RELAZIONE MINICHALLENGE
BRUTE-FORCING PASSWORD

1° scenario

Password cracking offline

Questo primo scenario rigurarda il recupero di password avendo i valori di hash(digest).

Le tecniche utilizzate saranno brute force puro con approcci incrementali e ibridi dizionario-bruteforce, rainbow table.

Sperimenterò principalmente con il file shadow delle password dei sistemi UNIX , con stringhe di varia lunghezza e di varia complessità(create casualmente o formate da parole della lingua comune).

JOHN THE RIPPER

Il più famoso tra i tool, si dice che sia il più veloce tra i brute-forcer ed il più semplice da utilizzare, per mia esperienza invece mi è risultato il più lento ed il più complicato principalmente per il fatto che i comandi non funzionano il 50% delle volte, se si esegue un semplice cracking brute-force(non ibrido) puro senza regole aggiuntive ,su una password hashata con MD5 o SHA1 relativamente semplice(4 caratteri in lowercase o alternati), sembra non finire mai(per una password di 6 caratteri tutti minuscoli hashata con MD5 o SHA1 ci ha messo all'incirca 4 ore eseguendolo in background). Ho tentato di risolvere cambiando il formato di cracking e specificandolo sia a riga di comando (con --format che seguendo le istruzioni sulla pagina ufficiale non funziona, cioè john --format:FORMATO, che continuava a darmi come errori unknown cyphertext oppure non trovava hash nel file quando in realtà erano presenti ed in grandi quantità) e ottimizzando utilizzando la GPU(che non usa quasi per niente dato che anche riconoscendomi i device, non li usa, andando a guardare il codice, non utilizza molto la memoria locale di openCL e altre ottimizzazioni tipiche per GPGPU come la vettorizzazione). Successivamente ho scoperto uno dei possibili motivi per cui non riusciva a crackare password così semplici in poco tempo, john the ripper non supporta digest MD5 con caratteri maiuscoli(almeno così ho letto). Un altro problema è che john the ripper lavora meglio con password e shadow files di UNIX, quindi utilizzarlo per altri motivi risulta deleterio rispetto ad altri tool/algoritmi (Hashcat funziona molto meglio per quasi qualsiasi formato e per qualsiasi situazione).

Il comando utilizzato per crackare un digest MD5 utilizzando un approccio incrementale(di default, si possono definire varianti ibride e limitate andando a cambiare le opzioni nel file john.conf) è il seguente:

```
sudo john –format=raw-md5 –incremental hashfile.txt
sudo john –show hashfile.txt
```

La parte importante di questo scenario è principalmente il cracking di password tenute in shadow(i digest), per utilizzare john si deve prima eseguire unshadow per rendere le passwd processabili da john, poi si procede con il cracking, le linee di comando per cercare di trovare le password di sistema sono le seguenti:

```
unshadow /etc/passwd /etc/shadow > mypasswd.txt
john mypasswd.txt
sudo john –show mypasswd.txt
```

Tutte le password che john the ripper "riesce" a recuperare vengono messe in un file(john.pot) che ha una locazione sbagliata rispetto a quello che viene dichiarato(almeno per me non era nella home del mio user, ma era tenuta insieme ai file di configurazione in /etc/john/,probabilmente dipende dal sistema operativo ma non saprei specificare l'assurdo motivo per cui è successo).

Il primo approccio utilizzato da john the ripper è --single che utilizza un attacco dizionario(possiamo usare i nostri dizionari definendo liste di parole nella directory in cui si trovano i file di configurazione e le altre liste). Se non si sono trovate password procede con quello incrementale(brute-force puro). È possibile definire un approccio ibrido utilizzando un file di wordlist e cambiando il file di john.conf nella directory di $JOHN(a me non funzionava e sono andato a trovarlo nella cartella /usr/share/).

un esempio di approccio incrementale è il seguente (nel file di conf):

```
[Incremental:LowerNum]
File = /usr/share/john/lowernum.chr
MinLen = 1
MaxLen = 13
CharCount = 36
```

Dove file definisce il file dove trovare i caratteri(file charset), minlen e maxlen sono minima e massima lunghezza per le password da provare, charcount sono i possibili caratteri da considerare nella ricerca delle password(se superano quelli contenuti nel file verrà notificato). Esistono anche altre opzioni e modi di fare il brute-forcing(è possibile definire un approccio ibrido), ma tutta la parte di scripting e programmazione è abbastanza complicata.

Per utilizzare una modalità incrementale si usa l'opzione --incremental=NOME.

Personalmente non mi è piaciuto come tool per come è stato strutturato e per la eccesiva difficoltà del suo uso, inoltre i manuali su internet non sono molto chiari, le informazioni che mi servivano le ho prese da terzi, e le uniche cose che non trovavo erano come definire approcci ibridi tramite i file di conf(che poi sono andato a vedere nelle pagine ufficiali in modo abbastanza approssimativo data la natura molto riassunta delle informazioni).

Resta comunque uno dei tool più utili per il cracking di password di sistema data la sua storia.

HASHCAT

Questo tool utilizza principalmente algoritmi scritti ed ottimizzati per essere eseguiti in parallelo, andando a guardare il codice usano in modo più massivo le proprietà basilari di openCL e della programmazione GPGPU basilari. Durante questo scenario infatti ho trovato questo tool molto più veloce e comprensibile(e coeso nelle scelte che hanno fatto gli sviluppatori di questo software).

Dove john the ripper è principalmente utilizzato e specializzato per recuperare le password di sistema, hashcat è molto utilizzato nel caso in cui si hanno grandi database di digest e si vogliono trovare password in modo veloce (rispetto ad altri scenari che considererò riguardanti bruteforcing).

Hashcat supporta tanti formati, lo scenario che considererò però si concentrerà principalmente su MD5 e SHA-1.

Il comando per trovare una password usando un'approccio di puro bruteforce senza nessuna regola è il seguente:

```
hashcat -m 0  hash.txt
```

dove l'opzione -m 0 esprime che il formato dell'hash è in MD5(per SHA1 sarà 100).

se si vogliono stabilire delle regole il comando è il seguente:

```
hashcat -m 0 -a 3 hash.txt ?l?l?l?l?a
```

dove -a stabilisce la strategia da seguire(3 è brute-force, 0 è dizionario, 1 è combinatoriale con più dizionari, 6 è ibrida, 7 stessa cosa ma le stringhe sistematiche vengono prima di quelle del dizionario), e ?l è un carattere minuscolo mentre ?a è alphanumerico.

Altri esempi di comandi sono i seguenti:

```
hashcat -m 0 -a 0 hash.txt example.dict

hashcat -m 0 -a 1 hash.txt example1.dict example2.dict

hashcat -m 0 -a 6 hash.txt example.dict  ?d?d?d?d?d

hashcat -m 0 -a 7 hash.txt ?d?d?d?d?d example.dict
```

Ovviamente il dizionario può essere generato prima utilizzando tool come **crunch** che creano stringhe sistematicamente seguendo regole stabilite dall'utente, ma questo lo affronterò quando parlerò di attacchi online con **hydra**.

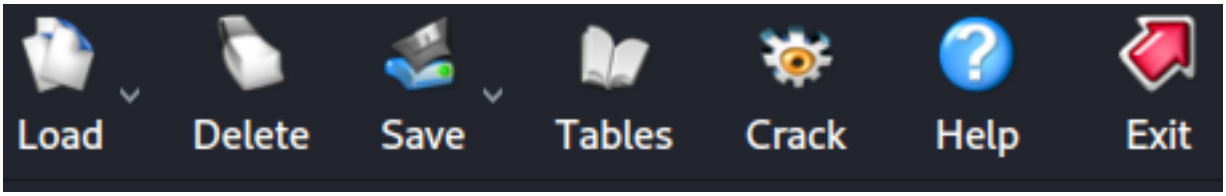
OPHCRACK

Strumento molto importante e famoso, anche per una versione live bootable per il recupero di password windows, poco utile per il cracking quotidiano dato che non contiene tools per la creazione di rainbow-tables e l'uso di proprie rainbow tables(probabilmente mi sto sbagliando ma su internet non ho trovato informazioni).

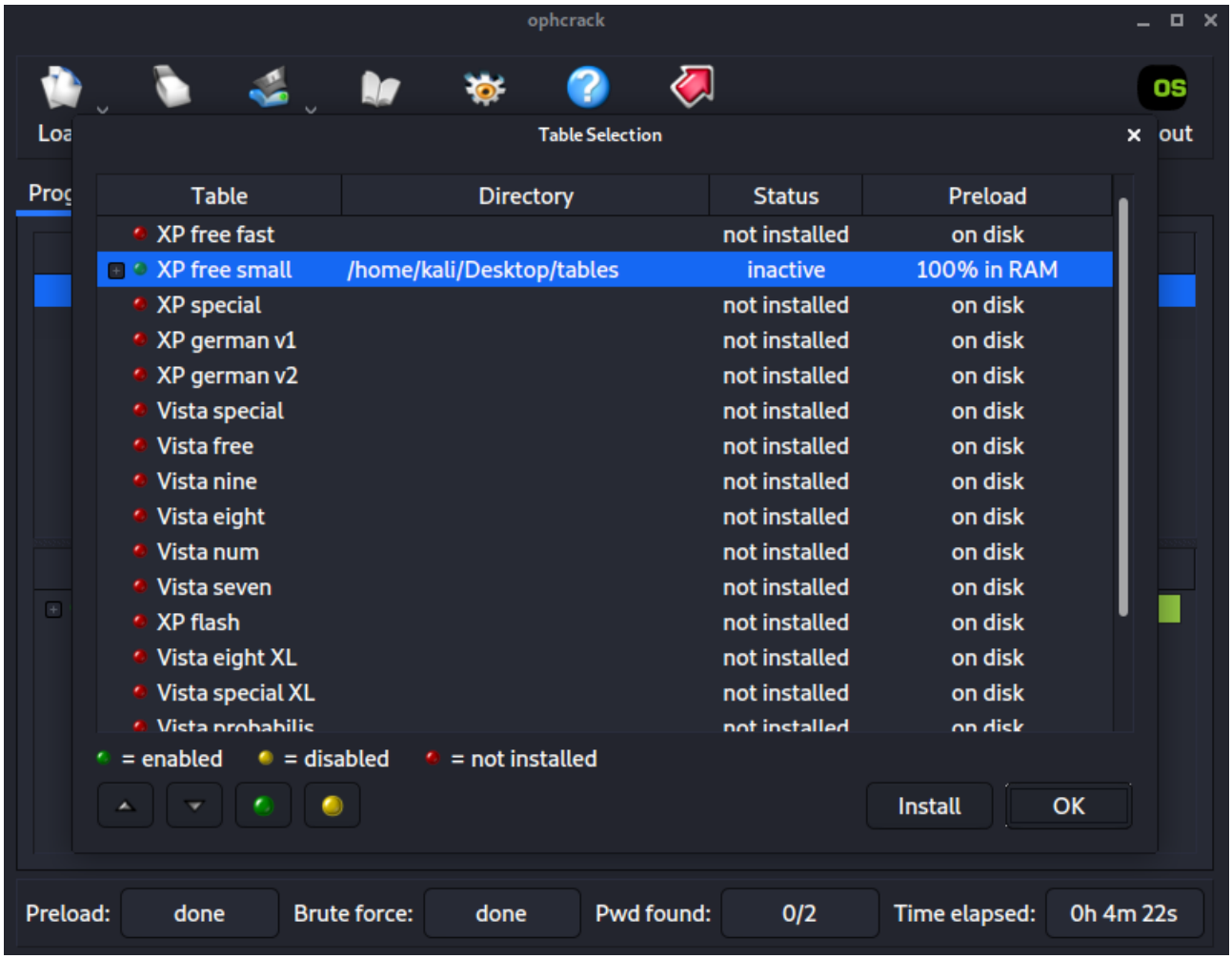
Ophcrack utilizza rainbow tables, utilizzerò una rainbow table abbastanza piccola come esempio(XP free table) anche perchè solitamente sono molto grandi per il cracking vero e proprio.

Per questo scenario, utilizzero la UI, ma i ragionamenti e procedimenti sono uguali per la CLI.

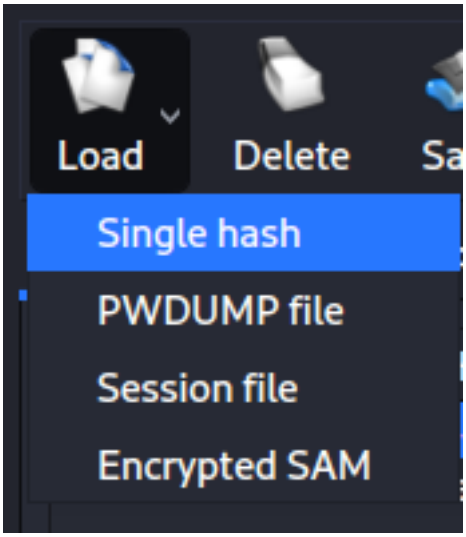
Nella UI si hanno diversi comandi:



Si parte caricando le rainbow tables scaricate in memoria andando su "tables", apparirà una schermata con le tavole di ophcrack, se si ha sul discouna tavola si carica in memoria e si usa:



Dopo aver caricato le tavole, carichiamo il database o i singoli hash:



Dopo averlo fatto, si inizia la crack, e ophcrack scorrera le rainbow table che sono caricate fin quando o le avrà finite, o avrà finito di recuperare le password.

RAINBOW-CRACK

Per creare le rainbow table, usiamo un tool diverso che si chiama rainbowcrack, che permette la definizione di proprie rainbow table insieme all'uso. In particolare questo tool genera rainbow chains.

I comandi per la generazione di rainbow-table sono i seguenti

```
rtgen md5 loweralpha-numeric 1 7 0 2400 24652134 0
rtgen md5 loweralpha-numeric 1 7 1 2400 24652134 0
rtgen md5 loweralpha-numeric 1 7 2 2400 24652134 0
rtgen md5 loweralpha-numeric 1 7 3 2400 24652134 0
rtgen md5 loweralpha-numeric 1 7 4 2400 24652134 0
rtgen md5 loweralpha-numeric 1 7 5 2400 24652134 0
rtsort .
```

Dopo la creazione delle rainbow-tables, per usare in modo efficente vengono ordinate.

Per utilizzare rainbow crack con le rainbow table definite i comandi sono i seguenti:

```
rcrack <path-to-rainbow-table.rt> -h  digest
```

dove path-to-rainbow-table.rt è il path assoluto alla tabella, -h è un'opzione che esprime singolo hash e *digest* è il valore di hash da crackare.

2° scenario

Password cracking online

Durante questo scenario, ci saranno comunque le stesse considerazioni sulle password, ma in questo caso la violazione avviene in remoto utilizzando dei protocolli di comunicazione e collegamento in remoto che necessitano identificazione e autorizzazione attraverso credenziali.

Questo scenario si focalizzera su un host con un certo indirizzo ip(simulato tramite macchina virtuale in rete interna), e avverrà un recupero di credenziali di accesso tramite tentativi di accesso remoto con SSH(gli stessi discorsi di brute-forcing valgono per altri protocolli e programmi che richiedono autenticazione con credenziali).

CRUNCH + HYDRA

Crunch

Lo strumento per creare stringhe di lunghezza arbitraria da un alfabeto definito (o da una lista di parole). Il generico template per l'uso di crunch è il seguente :

crunch min max charset options

min e max sono ovviamente il minimo ed il massimo numero di caratteri nelle stringhe generate(per -p non hanno nessuna utilità), charset è una lista di caratteri che può essere definita a riga di comando o in un file .lst, options sono delle possibili opzioni che includono:

-b per la massima dimensione di parole generate

-c per il massimo numero di righe generate

-d numero massimo di caratteri duplicati

-e definire una stringa dopo il quale si deve smettere di generare parole

-f per specificare una lista di charset nel file .lst

-o l'output

-p non stampa le permutazioni in cui i caratteri si ripetono

-r ritorna ad una vecchia sessione di crunch lasciata in sospeso

-t per specificare un certo pattern(@ lettere minuscole , lettere maiuscole % numeri ^ caratteri speciali)

-z per comprimere il file di stringhe generate

-s stringa prima di ogni parola generata

esempi di utilizzo sono i seguenti:

```
crunch 5 5 abcd1234 -s why54 -c 69 -b 40mb -o try.txt -z
crunch 5 5 abcde14 -t @@@09 -d 2@ -o try2.txt -z
crunch 1 1 -p  passw ord bird -o try3.txt
```

Dopo aver generato le stringhe possiamo procedere con Hydra

Hydra

Hydra è un brute-force cracker tool che lavora online, la sintassi per svolgere un recupero di credenziali ssh è il seguente:

```
hydra -L utenti.txt -P password.txt $(indirizzoIP_vittima) ssh
```

dove utenti.txt è una lista di utenti che può essere generata casualmente o i nomi veri e propri degli utenti vittime, password.txt è il file creato per esempio con crunch dove sono tenute le guesses da provare, indirizzoIP_vittima è l'indirizzo IP della vittima, e ssh è il protocollo da utilizzare(altre vulnerabilità o protocolli da utilizzare per recuperare le credenziali in remoto si possono trovare utilizzando ***nmap***).

Se dopo aver finito tutte le password, trova le credenziali che permettono l'accesso, le stampa a video.