Ingeniería Civil en Informática y Telecomunicaciones

Criptografía y seguridad en redes

Tarea 4 Criptografía y seguridad en redes

Autor:
Diego Lagos

Profesor:
Nicolás Boettcherh

22 de Junio de 2021

Índice

1. Introducción

En el ramo Çriptografía y Seguridad en Redes"de la carrera Ingeniería Civil en Informática y Telecomunicaciones Universidad Diego Portales se les pide a los alumnos hacer variadas tareas.

Este informe comprende la tarea 4 de dicho ramo, donde se le pide a los alumnos aplicar la materia de cifrado simétrico, además de la materia del ramo de hashes y conexión entre procesos usando sockets (materia del ramo Redes de Datos)

A los alumnos se les pasa unos archivos que corresponden a textos hasheados y otros archivos que son diccionarios para lograr descrifrar los textos ofuscados

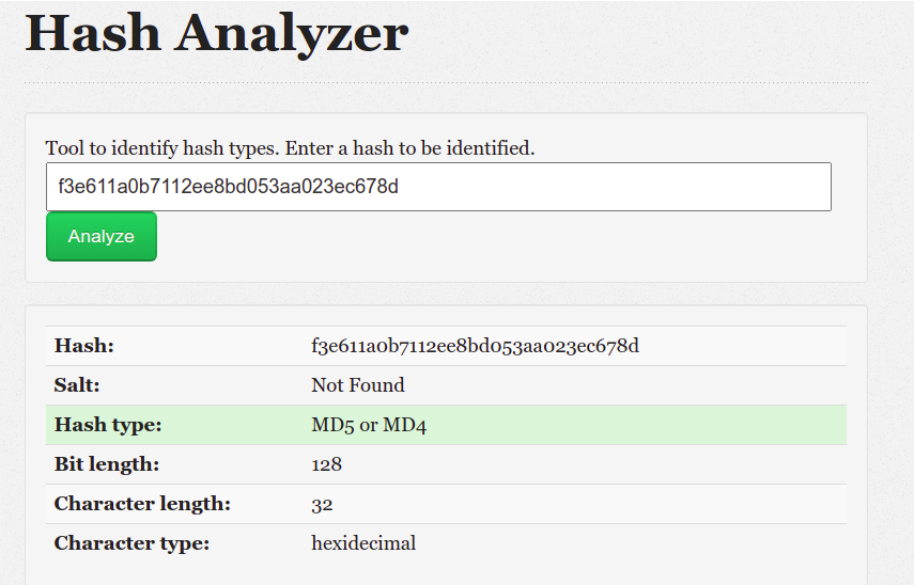
En este informe se narrará el proceso de crear los códigos que puedan deshashear los archivos y luego aplicar un hash y una criptografía asimétrica. Explicar el proceso de forma en que una persona con conocimientos técnicos básicos pueda comprender.

2. Hashes e identificación

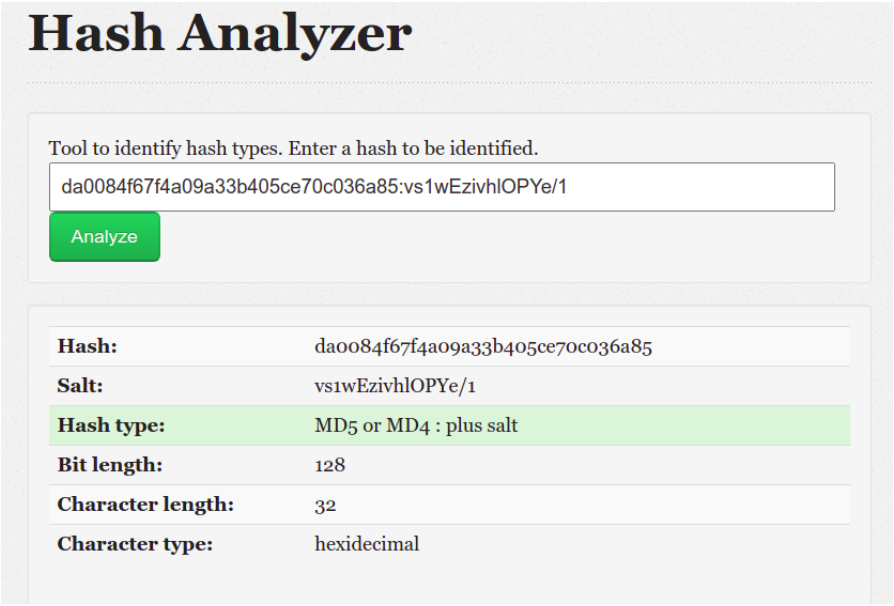
¿Que´ tipo de Hash se utiliza en cada archivo?

El primer paso es identificar el tipo de hash que se ocupa en cada archivo, para esto se utiliza la herramienta Hash Analyzer.

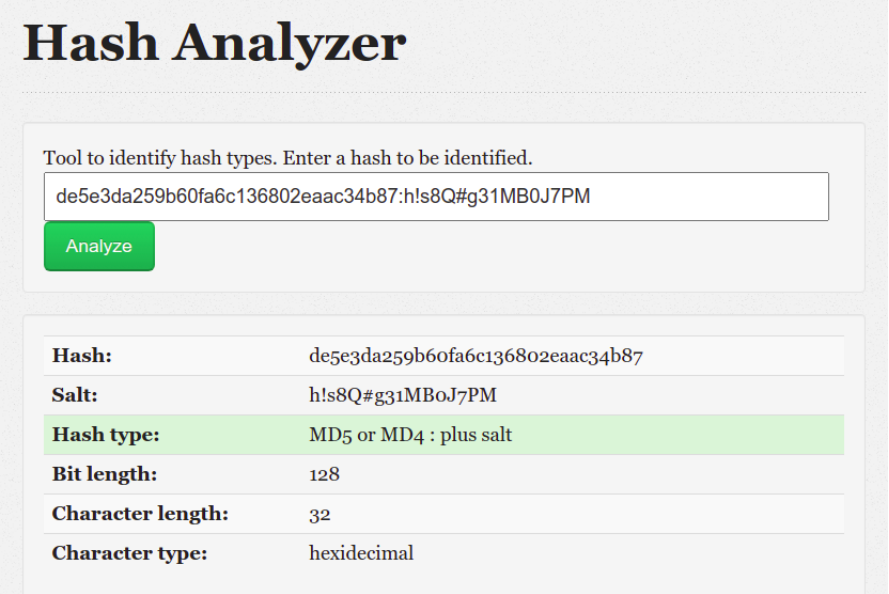
Para el primer archivo se encuentra que es hash tipo MD5



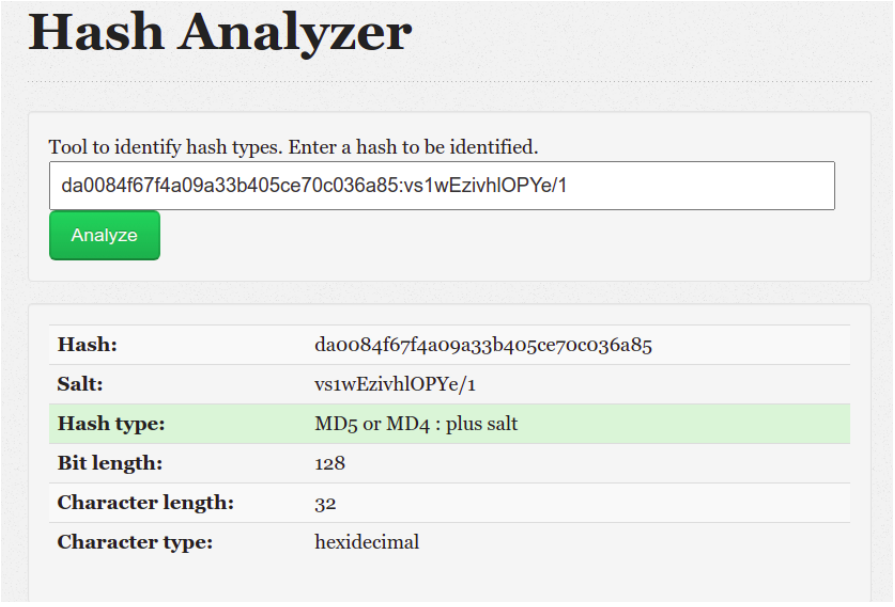
Para el segundo archivo se encuentra que es hash tipo MD5 con salt



Para el tercer archivo se encuentra que es hash tipo MD5 con salt

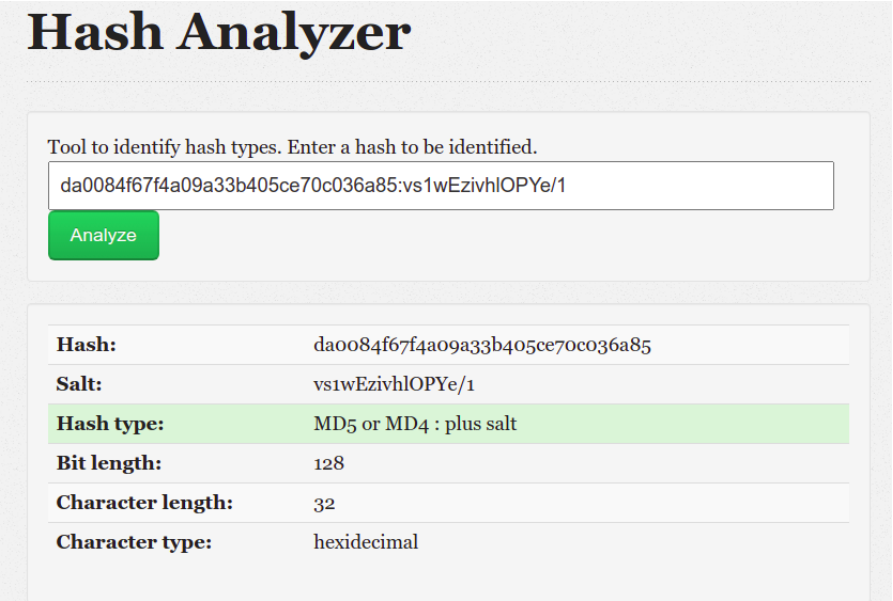


Para el cuarto archivo se encuentra que es hash tipo MD5 con salt



Pero probando con hashcat arroja error, hasta que se logró con NTML (probando otros hashes que tienen el mismo largo).

Para el quinto archivo no se encuentra que tipo de hash es.



Pero buscando ejemplos de hashes se encuentra que en la documentación de hashcat se encuentra el sha512crypt 6

3. Algoritmos y Tecnologías

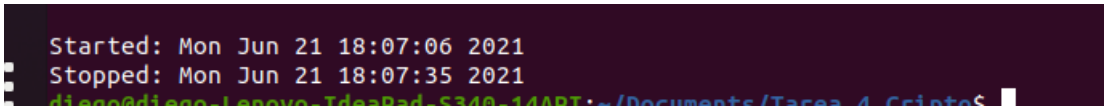
Las indicaciones que funcionaron en hashcat y dieron un resultado satisfactorios fueron las siguientes:

- Para el archivo 1 "hashcat -a 0 -m 0 -w 4 Ofuscado/archivo_1 Diccionarios/diccionario_2.dict"
- Para el archivo 2 "hashcat -a 0 -m 10 -w 4 Ofuscado/archivo_2 Diccionarios/diccionario_2.dict"
- Para el archivo 3 "hashcat -a 0 -m 10 -w 4 Ofuscado/archivo_3 Diccionarios/diccionario_2.dict"
- Para el archivo 4 "hashcat -a 0 -m 1000 -w 4 Ofuscado/archivo_4 Diccionarios/diccionario_2.dict"
- Para el archivo 5 "hashcat -a 0 -m 1800 -w 4 Ofuscado/archivo_5 Diccionarios/diccionario_2.dict"

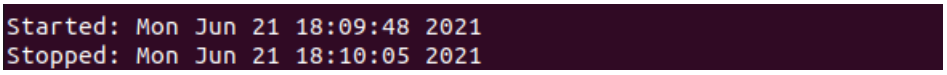
Los tiempos de ejecución son los siguientes

Tiempo de ejecución del archivo 2

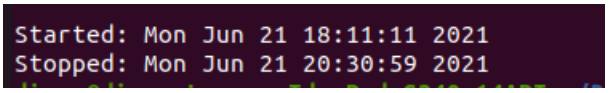
Tiempo de ejecución del archivo 3



Tiempo de ejecución del archivo 4



Tiempo de ejecución del archivo 5



Como se puede observar

4. Análisis de resultados

¿a que tribuiria la diferencia de tiempo?

Los tiempos de ejecución varian demasiado especialmente entre los tiempos del archivo 5 con los otros, este primero tardó más de dos horas. Esto se puede deber a que SHA-512 es un hasheo más complejo y -a diferencia de MD5- es todavía utilizado y respetado en la industria.
El problema de esto son los grandes tiempos de ejecución no solo al momento de hashear si no al momento de dehashear.

¿cual cree que es más seguro?

Definitivamente SHA (especialmente SHA-2) por que es más nuevo, entonces no existen demasiados ataques exitosos en contra de el, entonces no existen muchos diccionarios fáciles de encontrar.
Realmente es mejor si se compara a MD5, un hash antiguo que se comprobó que posee colisiones en sus hasheos, creando el mismo resultado de diferentes inputs. Además de ser antiguo y tener mucha historia de ataques (entonces existen muchas tablas/diccionarios disponibles para hacer ataques).

5. Códigos Python

Una vez deshasheados los textos, se le pide al alumno volver a hashearlos y luego encriptarlos en encripción asimétrica.
Para lograrlos, se analizó el archivo potfile que deja hashcat y se divide en 5 archivos diferentes llamados potfilearchivo(número).txt

Explique el algoritmo de hash que utilizará y por que

El hash que se utilizará es SHA-256 por que demeustra ser un hash que todabía es vigente en la industria, a pesar de su mayor tiempo de cálculo.

Explique el cifrao asimétrico y por que decidió usarlo

El cifrado asimétrico que se usa es el cifrado Paillier, se eligió por tener los beneficios de ser menos predecible que otros competidores (RSA por ejemplo) al tener diferentes resultados de el mismo input, entonces se convierte en un cifrado que piensa "fuera de la caja".

5.1. Código hasheo

rehasheo.py > ...

```python
import socket
import hashlib


port = 50557
s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.bind(("127.0.0.1",port))
s.listen(1)
client, addr = s.accept()

texto1 = open("hashcat/potfilearchivo1", "r")
texto2 = open("hashcat/potfilearchivo2", "r")
texto3 = open("hashcat/potfilearchivo3", "r")
texto4 = open("hashcat/potfilearchivo4", "r")
texto5 = open("hashcat/potfilearchivo5", "r")


textos = []
textos.append(texto1.readlines())
textos.append(texto2.readlines())
textos.append(texto3.readlines())
textos.append(texto4.readlines())
textos.append(texto5.readlines())

for i in range (0, len(textos)):
    hashlib.sha256(textos[i+1]).hexdigest()
    len(hashlib.sha256(textos[i+1]).digest())


```

Este código corresponde a rehasheo.py, la parte que hace el hash.

En las primeras líneas se importan dos liberías, socket para poder conectarse con el otro código y hashlin para poder hashear en SHA.
Luego crea el socket mediante la función socket.socket, y se pone a escuchar en la linea 1 del puerto 50557.
Luego este abre los resultados del potfile del deshasheo anterior y los guarda en un arreglo para poder hashearlos en SHA-256.

6. Github

El github conteniendo los códigos está en:
https://github.com/lagossully/CriptografiaTarea4

7. Conclusión

En esta tarea se aprende como se implementan las encriptaciones asimétricas, donde existen dos llaves una pública y una privada. la llave privada la posee el creador de la encriptación y solo sirve para encriptar, la llave privada solo sirve para desencriptar.
A diferencia de la criptografía simétrica, los tiempos de cálculo de encriptación son bastante largos, ya que a haber 2 llaves se acompleja todo el cálculo.

Pero la principal ventaja de este tipo de encriptación es el hecho de que al existir dos llaves no existe la posibilidad de ataques de externos, al tener control total de quien tiene que llave se puede controlar como se utiliza la encriptación. Aunque el mayor peligro es que una llave caiga en manos equivocadas.