Diseño de una herramienta para validación de niveles de seguridad de contraseñas en sistemas de información empresariales

Juan David Bojato Pacheco
Dpto. de Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
jbojato@uninorte.edu.co

Giovanni Camilo Moreno Vargas
Dpto. de Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
cgmoreno@uninorte.edu.co

Daniel Andrés Donado Avendaño
Dpto. de Ingeniería de Sistemas
Universidad del Norte
Barranquilla, Colombia
adavendano@uninorte.edu.co

***Abstract*—The following paper compiles the creation of an application that allows the verification of password security levels. For this, we combined the use of PassGAN, a more humane password generation algorithm, as well as the estimation of password strength with PESRank, optimizing it and allowing its update. This represents a novelty compared to other password estimators since the set of evaluated passwords is periodically updated, improving the estimation of passwords. Finally, a web application is provided which expresses the strength of a password and the time it would take an average computer to enumerate it.**

***Index Terms*—Password, security, PassGAN, PESRank, bits**

***Resumen*—En el siguiente trabajo se recopila la creación de una aplicación que permite la verificación de niveles de seguridad de contraseñas. Para esto, se combinó el uso de PassGAN, un algoritmo de generación de contraseñas mas humanas, así como la estimación de fuerza de la contraseña con PESRank, optimizándolo y permitiendo su actualización. Lo anterior representa una novedad frente a otros estimadores de contraseñas, puesto que al mantenerse actualizado de forma periódica el conjunto de contraseñas evaluadas, existe una mejora en la estimación de contraseñas. Finalmente se entrega una aplicacion web en la que se expresa la fortaleza de una contraseña y el tiempo que tardaría una computadora promedio en enumerarla.**

***Palabras clave*—contraseña, seguridad, PassGAN, PESRank, bits**

I. Introducción

En la actualidad cualquier tipo de sistema electrónico que se encargue de almacenar información perteneciente a una empresa debe contar con un mínimo grado de seguridad, para evitar que cualquier persona ajena a la organización tenga acceso a la información contenida en estos dispositivos. No obstante, los clientes y/o usuarios necesitan poder acceder a su información privada, para ello se requieren maneras de identificar que la persona que desea acceder es realmente la persona que dice ser. Para tal problemática se usa la autenticación, en ella destacamos el método más popular de autenticación el cual consiste en proporcionar un usuario o correo electrónico y una contraseña para así tener acceso. Por lo general, las personas escogen como contraseña combinaciones de caracteres que les sean fáciles de recordar tales como nombres, fechas, números telefónicos etc. sin tener en cuenta lo vulnerable que estas pueden ser. En consecuencia, debido al uso de este tipo de contraseñas los atacantes o hackers se valen de técnicas y herramientas de vulneración y crackeo de contraseñas (Security, 2020).

Muchas empresas han estado preocupadas por esta problemática, puesto que poseen empleados cuyas contraseñas son muy vulnerables o sencillas y que esto represente una vulnerabilidad a explotar por hackers y en consecuencia, obtener el acceso a información de vital importancia para la empresa. Debido a esta problemática, se han ido implementando mecanismos y técnicas que permitan mejorar la seguridad de las contraseñas tales como: el uso de mayúsculas, números, símbolos intercalados e incluso muchas personas han llegado a utilizar diferentes aplicaciones que generan contraseñas seguras, creando así una cadena aleatoria de letras mayúsculas, minúsculas números y símbolos que se considere segura o en su defecto, difícil de vulnerar. Estas practicas aunque mejoran la seguridad de las contraseñas y hace que sean menos vulnerables dificulta el proceso de recordarlas y muchas personas para no olvidarlas las anotan en libretas o archivos de texto haciendo que sean aun mas fáciles de obtener.

Por otra parte, es importante que las contraseñas sean tanto seguras como fáciles de recordar, con la finalidad de dificultarle el trabajo a los atacantes y facilitárselo a las personas o empleados. Para hacer posible lo anterior, es necesaria la existencia de una aplicación que le permita a los empleados de una empresa saber que tan segura es su contraseña manteniendo la facilidad para recordarla y evitar comprometer la información privada de la empresa.

La mayor preocupación de vulnerabilidad que podemos encontrar están en los sistemas de información, ya que estos manejan datos que son de vital importancia para el funcionamiento de muchas empresas, es por esto que es necesario que las organizaciones definan e implementen los respectivos controles administrativos, operativos y técnicos que permitan llegar a un nivel de seguridad aceptable, que los riesgos de seguridad que se manejen sean permitidos. El obtener esta

garantía que el sistema de información es segura se traduce en que brinda la confianza de tener privacidad y ningún riesgo de alta magnitud. (Spears, Barki, & Barton, 2013)

El manejo de la seguridad de la información no solo trata del acceso a los datos que se encuentran protegidos, ya que esto solo hace referencia a la confidencialidad, y hay que tener en cuenta dos aspectos más, la integridad y la disponibilidad,es decir, seguir los lineamientos de la ÇIA "por sus iniciales en inglés. Es fundamental que los datos solo sean accesibles por aquellos a los que está destinado esta función (confidencialidad), garantizar que los datos no cambien (integridad) y por supuesto, estar disponibles la mayor cantidad de tiempo posible (disponibilidad)(Olcott, 2019).

II. Definición del problema

La información en la actualidad ha cobrado un grado de importancia tan alto a tal punto que se puede llegar a comparar su valor con el valor del dinero. Es por esto que los datos se han convertido en el objetivo de los atacantes cibernéticos, lo que conlleva a que las empresas realicen grandes inversiones en ciberseguridad, sin embargo, por mayor que sea la inversión hay un apartado que depende de los clientes, empleados, asociados, a fin de cuentas, factor humano.

La privacidad de los datos es fundamental para cualquier empresa que tenga en sus manos cualquier tipo de información sobre sus clientes. Por tanto es de gran importancia proteger estos datos ante cualquier ataque de terceros. Sin embargo, hay un apartado cuya seguridad se ve influenciada por la capacidad del usuario: las cuentas cuyo acceso implica un usuario y una contraseña. Por tal motivo, se quiere fortalecer las contraseñas que son utilizadas por las personas midiendo su grado de fortaleza, con la finalidad de reducir las vulnerabilidades que se presentan ante ataques maliciosos para obtenerlas. Para realizar esto se va a hacer uso de las herramientas Passgan y PESrank que permitirán crear un conjunto de contraseñas de características humanas y clasificar las contraseñas de los usuarios por la fortaleza que tengan, respectivamente. La problemática presentada y sus interrogantes se reflejan en la **Figura 1**.

III. Objetivos

A. Objetivo General

Diseñar, desarrollar y evaluar una herramienta para definir la fortaleza de contraseñas en aplicaciones empresariales.

B. Objetivos Específicos

- Entrenar modelos que nos permitan generar diccionarios de contraseñas acordes con los sistemas empresariales.
- Implementar un sistema de clasificación de contraseñas que exprese que tan fuerte es una contraseña.
- Utilizar los diccionarios de contraseñas generados para entrenar el sistema de clasificación de contraseñas.
- Realizar pruebas de la herramienta.
- Evaluar los resultados obtenidos.

IV. Metodología

Para el desarrollo de esta solución se escoge la metodología XP, extreme programming, dado que, se basa en iteraciones cortas (de 1 a 2 semanas) lo cual abre la posibilidad de avanzar de manera más rápida en el desarrollo del proyecto, además, éste permite más flexibilidad para realizar modificaciones durante la ejecución de la tarea, en caso de que haya que realizar un cambio no previsto durante la planeación. De igual manera, posee prioridades definidas para el desarrollo de las tareas, ya que esto es dado por el cliente, esto con la finalidad de procurar su satisfacción (Beck, 2000).

Debido a la naturaleza misma del proyecto, la cual se centra más en la programación y el desarrollo que en la documentación, se considera apropiado la implementación de XP, ya que a su vez, proporciona valores y prácticas definidos para una buena ejecución. Finalmente, XP, permite el constante testeo de las funcionalidades, lo cual ayuda a estar en constante retroalimentación, al estar seguros de que todos los componentes están funcionando, reduciendo así, riesgos de que el programa y/o el proyecto fracasen.

V. Justificación

La información ha llegado a tal valor que existen servicios que no requieren de ningún pago económico para hacer uso de este, solo con el hecho de suministrar información del cliente es más que suficiente para tener un negocio rentable. Es por esto que los datos se han vuelto objetivo de los atacantes malintencionados, lo que ha provocado mayor entusiasmo en aumentar la seguridad de estos (Spears et al., 2013).

Entre las principales vulnerabilidades que se encuentran en los sistemas son los famosos sistemas de cuentas con usuario y contraseñas, las cuales no pueden controlarse su seguridad, ya que estos dependen del cliente en cuestión. La explotación de estas vulnerabilidades puede desencadenar en robos de identidad, obtención de datos del cliente, entre otros, los cuales pueden tener generar resultados negativos para la empresa. Por tanto, se desarrolla esta aplicación que permite verificar la fortaleza de las contraseñas suministradas, para que dependiendo su facilidad de encontrarse, proponer las medidas necesarias que permitan incrementar su resistencia y sean menos susceptibles a ser descifradas mediante cualquier tipo de ataque.

VI. Marco Teorico

"La fuerza de la contraseña se mide en los intentos realizados para adivinarla"(David & Wool, 2020a). Conocemos una contraseña como una palabra secreta o combinación de letras, números o símbolos, usada para obtener acceso a un lugar, una información o un sistema de cómputo y juegan un rol importante en en nuestra vida diaria en diferentes aplicaciones computacionales como las de cajeros, servicios de internet, autenticación en el computador, en el teléfono, etc.

El principal objetivo de una contraseña es restringir el acceso de personas no autorizadas al sistema y por lo tanto siempre queremos que sean lo suficientemente seguras (Raza, Iqbal, Sharif, & Haider, 2012). Partiendo de esta definición

Figura 1: Mapa conceptual del problema

encontramos herramientas como PESrank que se encarga de estimar la fuerza de una contraseña a través de unos rangos que permiten determinar que tan probable es que se encuentre una contraseña. Todas las contraseñas son descifrables, sin embargo, el que tengan una mayor fuerza va a permitir que los atacantes se tomen más recursos y más tiempo para poder extraerlas.

Entre las técnicas de ataque más comunes podemos encontrar los ataques de diccionario y los ataques de fuerza bruta. Un ataque de diccionario prioriza la velocidad de ataque aunque esto conlleve a un menor espectro de posibilidades, mientras que un ataque de fuerza bruta cubre muchas más posibilidades reduciendo considerablemente la velocidad. Un ataque de diccionario usa un conjunto de palabras comunes para ser probadas, mientras que los de fuerza bruta prueban todas las combinaciones de caracteres posibles para formar cadenas de caracteres (Bošnjak, Sreš, & Brumen, 2018). Pero no todo es blanco o negro, es posible utilizar ambas técnicas según la información que tengamos sobre las contraseñas, tal y como un patrón particular que pueda ser replicado.

Los ataques de diccionario son los mas utilizados porque ayudan a ahorrar bastante tiempo y los diccionarios son creados con contraseñas reales y comunes que han utilizado diferentes personas. Gracias a esto se han creado muchos diccionarios a lo largo de los años con los que se realizan este tipo de ataques, y recientemente se están llegando a conocer aplicaciones que generan diccionarios de contraseñas como PassGAN.

A. ¿Como actúa un atacante?

Un atacante que busca atacar una contraseña tiene a su disposición diferentes herramientas para perpetrar como el uso de Wordlists que son simples archivos de texto que contienen palabras para probar contraseñas que es el ya mencionado ataque de diccionario. Lo más importante de este ataque es la precisión y esto los atacantes lo saben, por lo que utilizan diccionarios relevantes y de temas específicos basados en algún tema, cultura, industrias específicas y basados en regiones geográficas para lograr contener en el las contraseñas más comúnmente usadas. Estos diccionarios se pueden encontrar fácilmente buscando en internet y también pueden modificarlos y aumentar su efectividad con herramientas como CeWL y John the Ripper.

El ataque de diccionario es el más común o el más utilizado por atacantes por la precisión que tiene, pero a veces este lo pueden combinar con un ataque de fuerza bruta o directamente utilizar fuerza bruta. Este ataque se basa en explorar cada una de las combinaciones posibles, pero la viabilidad de esto es el dominio y el tamaño de la contraseña. Por ejemplo, (Mendoza, 2017) afirma "para 4 caracteres existen 14,766,336 de combinaciones entre mayúsculas, minúsculas y dígitos pero ya para 8 hay $2{,}18x10^{14}$ y entre más caracteres o si se agregan símbolos el número de combinaciones aumenta y un computador puede probar alrededor de 1 millón de contraseñas por segundo". Lo anterior quiere decir que aquellas contraseñas de corta longitud, pueden ser descifradas en segundos, pero contraseñas cuyo tamaño sea mayor, pueden tomar años en ser descifradas sin importar que tanta capacidad computacional se tenga.

Los ataques de canal lateral (SCA) constituyen una de las formas más fáciles y potentes de ejecutar contra las implementaciones criptográficas, y sus objetivos van desde primitivas, protocolos, módulos y dispositivos hasta sistemas uniformes (Zhou & Feng, 2005). Es decir, que representan una grave amenaza para la seguridad de los productos de hardware criptográfico. De esta forma, para llevar a cabo dichos ataques, se utilizan soluciones a problemas como **key enumeration problem** y **rank estimation problem**.

- **Key enumeration problem:** Dado un conjunto d de espacios de subclaves independientes *k1, ..., kd*, cada una de tamaño *n* y con su correspondiente probabilidad, el problema radica en enumerar el espacio de clave completos en orden de probabilidad decreciente, desde la clave más probable hasta la menos probable, teniendo en cuenta que la probabilidad de una clave completa es el producto de las probabilidades de sus subclaves y una vez hecho esto, ir probando hasta encontrar la clave deseada (David & Wool, 2020a).
- **Rank estimation problem:** Continuando con los valores anteriores, este problema consiste en hallar la posición de la clave *k** en la lista de n^d posibles claves, cuyas probabilidades están ordenadas de forma decreciente. Dicha posición estará acotada por un limite inferior y uno superior, cuya suma determinará la cantidad de bits necesarias para enumerar la cantidad de contraseñas candidatas necesarias para recuperar una contraseña. La solución a este problema está dada por el algoritmo PESRank (David & Wool, 2020b).

B. Tablas de búsqueda

En el proceso de revisión de literatura se tomaron un total de cinco bases de datos para realizar distintas búsquedas asociadas con conceptos claves trabajados en este articulo. De esta forma, se realizaron tres búsquedas principales, de los siguientes términos: Passgan, Password Security y Cracking, así como también, se consultaron artículos cuyos abstracts tuviesen similitud a lo presentado. Las búsquedas realizadas se hicieron teniendo en cuenta una serie de filtros (año, exactitud), obteniendo los resultados disponibles en los **cuadros I, II y III**.

Base de datos	Passgan	Password Security	Cracking
IEEE	4	4213	13146
ACM	3	113801	5494
Oxford Academic Journals	0	1840	36848
Web of Science	4	5913	84525
Springer Link	9	39449	385485
Total	20	165216	525035

Cuadro I: Consulta sin filtros en bases de datos

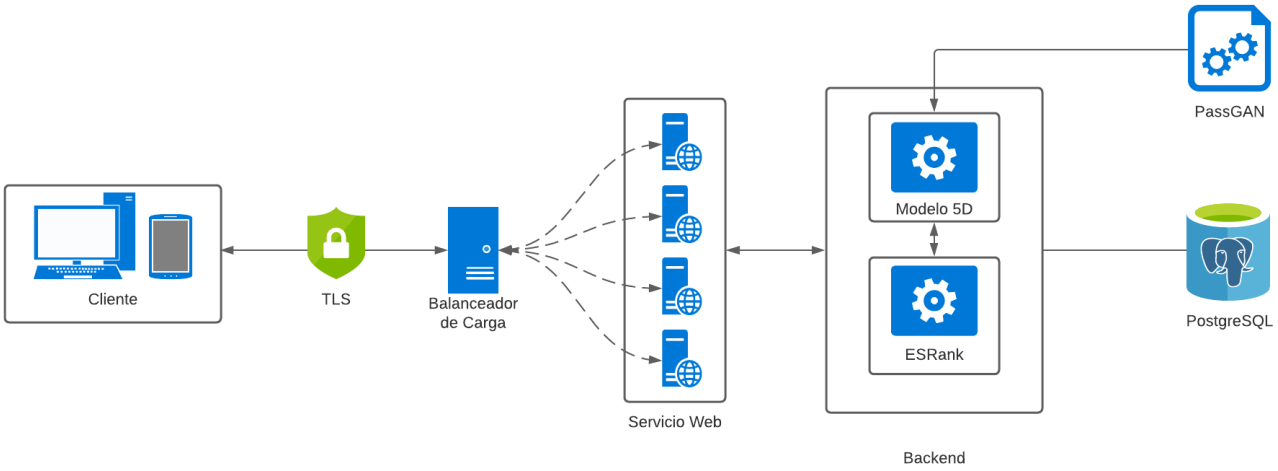

Figura 2: Arquitectura de la solución

Base de datos	Passgan	Password Security	Cracking
IEEE	3	132	216
ACM	3	13121	1785
Oxford Academic Journals	0	238	6681
Web of Science	2	475	22865
Springer Link	6	5167	3667
Total	14	19133	35214

Cuadro II: Consulta con filtros en bases de datos

Base de datos	de articulos
IEEE	31
ACM	24
Oxford Academic Journals	16
Web of Science	27
Springer Link	10
Total	108

Cuadro III: Abstracts similares obtenidos en bases de datos

VII. Arquitectura lógica

La arquitectura de la solución se puede observar en **la Figura 2.** En esta figura se representa la estructura del algoritmo implementado y las tecnologías empleadas, la cual se fundamenta en la creación de una API o servicio web, que pueda ser consumida vía web o mediante aplicación móvil, y en la que se desarrolle el algoritmo ESRank (presente en PESRank) (L. & A, 2019), el cual se alimenta con un conjunto de millones de contraseñas creadas mediante PassGAN (Hitaj, Gasti, Ateniese, & Perez-Cruz, 2017), cuyas probabilidades se almacenan en una base de datos conocida como PostgreSQL (Stonebraker, Rowe, & Hirohama, n.d.).

VIII. Desarrollo e implementación de la solución

Cómo se menciona anteriormente, los ataques del canal lateral permiten obtener información de un sistema informático mediante sus características o implementación física. Para la solución presentada en este articulo, se simuló la información obtenida por estas técnicas mediante la construcción de un modelo de contraseñas compuesto por un conjunto de dimensiones en el que se aplican soluciones a las problemáticas presentadas anteriormente y con ello, es posible realizar una estimación de la fortaleza de un conjunto de contraseñas. Para realizar la implementación de la solución se necesita un diccionarios de contraseñas acordes con las de un sistema empresarial, ya que los sistemas empresariales en comparación a otros sistemas, poseen información más sensible y confidencial por lo que su seguridad debe ser mayor. Para obtener los diccionarios de contraseñas se utiliza la herramienta PassGAN que se define como una nueva aproximación a la generación de contraseñas basado en Deep Learning y Generative Adversarial Networks (GANs) (Hitaj et al., 2017). Las redes adversarias generativas (conocidas como GAN por sus siglas en inglés) plantean el proceso de entrenamiento como un juego entre dos redes separadas: una red generadora y una segunda red discriminatoria que trata de clasificar las muestras como procedentes de la distribución verdadera o la distribución del modelo. Cada vez que el discriminador nota una diferencia entre las dos distribuciones, el generador ajusta ligeramente sus parámetros para hacerlo desaparecer, hasta que al final (en teoría) el generador reproduce exactamente la distribución de datos verdadera y el discriminador está adivinando al azar, incapaz de encontrar una diferencia (Baruffaldi & Uzal, 2017).

Las redes GAN son usualmente usadas para la generación de imágenes, pero PassGAN está diseñado para aprender la distribución de la información de las contraseñas a partir de contraseñas filtradas, genera contraseñas sin la intervención de usuarios e iguales que las utilizadas por las personas.

Actualmente PassGAN cuenta con su código abierto, con una versión para Python 3 en https://github.com/d4ichi/PassGAN y se puede experimentar con él, usando diversos modelos pre entrenados como por ejemplo el famoso diccionario RockYou y con ello generar conjuntos de contraseñas. Así mismo, también se puede entrenar con tu propio conjunto de datos para

generarlas. Como nuestro objetivo es generar contraseñas lo más parecidas a las que podemos encontrar en un sistema empresarial utilizaremos un dataset de contraseñas filtradas de LinkedIn, configuraremos PassGAN para que seleccione las contraseñas de hasta 17 caracteres y realice 100000 iteraciones. Los demás parámetros se dejan por defecto. Finalmente, una vez PassGAN entrenado podemos generar diccionarios de contraseñas con la cantidad deseada.

Por otra parte, para llevar a cabo la solución a la problemática de hallar la fuerza de una contraseña , fue necesario la creación de una API en el lenguaje de programación Python que utiliza el algoritmo ESrank como base para entrenar el modelo, mediante un conjunto de probabilidades previamente calculadas (almacenadas en la base de datos PostgreSQL), las cuales permitieron obtener el número de bits necesarios para encontrar una contraseña dada, y con ello, estimar su fuerza. El algoritmo desarrollado calcula las 5 dimensiones de una contraseña, y posteriormente, sus probabilidades, las cuales nos permiten obtener dos listas denominadas L1 y L2 cuya finalidad es la de entrenar el modelo a través del algoritmo ESrank. Finalmente, una vez entrenado dicho modelo, se realizan las solicitudes a la API como servicio web, teniendo como parámetro la contraseña a evaluar, y se retorna un mensaje que expresa la fortaleza o no de la contraseña digitada.

A. Modelo 5D

Para llevar a cabo la estimación de probabilidades, es necesario calcular 5 dimensiones en las que se divide una contraseña, tales como: Prefijo, palabra base, sufijo, patrón de cambio y transformación 133t.

1) **Prefijo:** consiste en todos aquellos dígitos y símbolos que se encuentran a la izquierda de la primera letra que se encuentre en la contraseña al leerla de izquierda a derecha. Un ejemplo de un prefijo es el siguiente: Si tenemos la contraseña 123@joseherediaxD, el prefijo será 123@ ya que la primera letra que se encuentra de izquierda a derecha es la j, por lo que todo lo que esté antes de la j será considerado como prefijo. También pueden existir casos en donde el prefijo es vacío, como por ejemplo, en la contraseña josedavid2009.
2) **Palabra base:** es la subcadena que se encuentra entre la letra más a la izquierda y la letra mas a la derecha de la contraseña. Por ejemplo: si la contraseña es hola12345, la palabra base es hola. Si no hay números o símbolos en la contraseña, la misma contraseña se considera como palabra base. Adicionalmente, las mayúsculas se convierten en minúsculas.
3) **Sufijo:** consiste en todos aquellos dígitos y símbolos que se encuentran a la derecha de la primera letra que se encuentre en la contraseña al leerla de derecha a izquierda. Un ejemplo de un prefijo es el siguiente: Si tenemos la contraseña 123abc456, el sufijo será 456 ya que la primera letra que se encuentra de derecha a izquierda es la c, por lo que todo lo que esté después de la c será considerado como sufijo. También pueden existir casos en donde el sufijo es vacío, como por ejemplo, en la contraseña 23josedavid.
4) **Patrón de cambio:** se define como un lista de índices positivos y negativos que indican la posición en la que aparecen letras mayúsculas en una contraseña. Los índices negativos se cuentan desde el final de la palabra, siendo el valor de -1 la posición correspondiente a la letra más a la derecha. En cuanto a los índices positivos, se empiezan a contar desde el principio de la palabra, con el valor de 0 representando la letra más a la izquierda. Para evitar la ambigüedad, tanto los índices negativos como los positivos no superan el índice medio (la mitad de la longitud total de la contraseña). Un ejemplo de esto se puede observar en la contraseña RealMadriD123, en la cual el patrón de cambio es [0, 4, -4].
5) **Transformación l33t:** se trata de mutar las contraseñas, reemplazando aquellos dígitos y símbolos que visualmente sean similares en las palabras base, por aquellas que correspondan a su transformación l33t, retornando los índices asociados a cada transformación, como se evidencia en el **cuadro V**.

Índice	Letra original	133t
1	o	0
2,3	a	[@,4]
4,5	s	[$,5]
6	e	3
7,8	g	[6,9]
9, 10	t	[+,7]
11	z	2
12 , 13	i	[1, !]
14	x	%

Cuadro IV: Transformaciones l33t

Por ejemplo: Para la contraseña ”g00dPa$$w0rD”su transformación 133t tiene como resultado los valores [1,4].

Un ejemplo completo con todas las dimensiones es el siguiente:

Contraseña	Prefijo	Palabra base	Sufijo	P.C	l33t
23jormi5	23	jormi	5	vacío	vacío
S0l2020	vacío	sol	2020	[0]	[1]
bkB25dgeK	vacío	bkb25dgek	vacío	[2,-1]	[5,11]

Cuadro V: Ejemplo con las 5 Dimensiones

Con el cálculo de las dimensiones, es posible calcular las probabilidades asociadas a cada contraseña, y con ello, implementar el algoritmo ESRank, el cual tiene como entrada los siguientes parámetros:

- **L1 y L2**: son datasets obtenidos al ejecutar previamente una función presente en el mismo módulo de ESRank.
- **b:** Es la mínima longitud de las listas de probabilidades obtenidas.
- **contraseña:** Es la contraseña a la que se busca evaluar su fortaleza.

- **d:** representa la cantidad de dimensiones utilizadas (como es un modelo 5D, el valor es 5).
- **gamma:** Valor obtenido al implementar la fórmula (b+1)/2.
- **p:** Probabilidad aleatoria.

La salida del algoritmo ESRank es un número de bits que al enumerarlos representan la cantidad de contraseñas candidatas necesarias para obtener la contraseña solicitada como petición al servicio. No obstante, esta enumeración será interpretada de otra manera, de tal forma que si el número obtenido se encuentra en un rango específico, el mensaje de salida indique que tan fuerte o no es una contraseña como se evidencia en la figura X.

B. Etapa de entrenamiento

Para entrenar el modelo se tomó un dataset original del sitio web extinto hashes.org, el cual contiene 60'525.522 millones de contraseñas (Dorsey, 2017), y se configuró la herramienta PAssGAN para entrenar un modelo basado en el dataset anterior, cuyas contraseñas debían tener una longitud de 17 caracteres. Una vez entrenado el modelo, fue posible crear diccionarios de contraseñas cuyo tamaño se podía variar dependiendo de la cantidad deseada, y con ello, se pudo entrenar nuestra herramienta desarrollada , y posteriormente realizar comparaciones en los tiempos de ejecución obtenidos al utilizar PESrank contra los tiempos de ejecución de nuestro algoritmo.

Para el entrenamiento de los modelos utilizando PESrank se usó un equipo propio de un integrante del equipo con las siguientes especificaciones:
Procesador: AMD Ryzen 5 2600X Six-Core Processor (12 CPUs) 3.6GHz.
Memoria: 16384MB RAM.
Tarjeta Grafica: NVIDIA GeForce GTX 1060 6GB.

Como estos entrenamientos requieren grandes cantidades de recursos, entre mejor sean los componentes del equipo empleado, los tiempos de ejecución del algoritmo serán menores y se obtendrá un mejor desempeño.

C. Fortaleza de una contraseña

De hecho, estudios recientes evalúan la seguridad de la contraseña como la cantidad de intentos que necesitaría un ataque para adivinarla. Con esta métrica, una contraseña cuya fuerza sea 2^x puede considerarse tan fuerte como una clave de cifrado simétrica de x bits, ya que un atacante puede adivinar con el mismo esfuerzo (Dell'Amico & Filippone, 2015). Basado en esto, se establecieron una serie de rangos para determinar la fortaleza de una contraseña, siendo considerada débiles aquellas contraseñas cuya cantidad de bits sean menores a 30 bits, y muy fuertes aquellas superiores a 70 bits. Estos rangos y los restantes se pueden observar en el **cuadro VI**, en el que x representa el número de bits.

Por otra parte, es posible enumerar la cantidad de bits obtenida por el algoritmo para así poder expresar ya sea en segundos, horas, días e incluso hasta años, el tiempo en que se tardaría una computadora promedio en enumerar dicha cantidad de bits por medio de un ataque de fuerza bruta. En este articulo, se emplea como computadora promedio aquella con el procesador Intel's Core i7-970, el cual ejecuta 109 GFLOPs (Operaciones de coma flotante por segundos) (Williams, 2010), y con ello fue posible calcular el tiempo promedio aproximado en segundos que se necesita para enumerar una contraseña, basados en la siguiente fórmula:

Rango	Nivel de fortaleza
$x \leq 30$	muy débil
$30 < x \leq 40$	débil
$40 < x \leq 60$	aceptable
$60 < x \leq 70$	fuerte
$x > 70$	muy fuerte

Cuadro VI: Fortaleza de contraseñas

$$t = \frac{2^x}{109 \cdot 2^{30}} \tag{1}$$

IX. Resultados

Con base a la API creada en el lenguaje de programación Python, se realizó la implementación de un servicio web y móvil que envía peticiones a dicha API y cuya respuesta expresa tanto el grado de fortaleza de la contraseña ingresada (como se observó previamente en el **cuadro VI**) así como también el tiempo en que una computadora promedio tardaría en enumerar su cantidad de bits. El código fuente de la implementación web se encuentra disponible en el enlace: https://github.com/Juandavid716/APIPasswordchecker así como su URL de acceso: http://password-checker.surge.sh/. Por otra parte, su implementación a nivel móvil se encuentra disponible en el enlace: https://drive.google.com/file/d/1N9cHzezjeOP3EWo4tHy9lZa-Wz85ZeKl/view?usp=sharing. Con ello, fue posible comprobar qué tan fuerte es una contraseña respecto a mas de 10 millones de contraseñas entrenadas con anterioridad usando PassGAN. A su vez, se pudo observar que dependiendo del conjunto de contraseñas (almacenadas en la base de datos Postgresql) utilizadas como fuente de entrenamiento para el modelo, una contraseña que bajo la lógica humana es vulnerable y fácil, para el algoritmo implementado no necesariamente lo será. Es por ello que PassGAN es importante y representa una novedad en comparación a lo implementado en PESRank, puesto que ya que cada cierto período de tiempo en días, las contraseñas usadas para entrenar el modelo son reentrenadas, lo que mejora la capacidad del algoritmo para identificar contraseñas que con anterioridad no eran detectadas como fáciles de descifrar, y con ello se garantiza una mejor eficacia en la detección de contraseñas vulnerables y no vulnerables.

Por otra parte, la eficacia en la detección de la seguridad de contraseñas mejora a medida que se incrementan el conjunto de contraseñas en la base de datos. A su vez, se observa que

entre menor sea la probabilidad de cada dimensión de una contraseña dada, mayor será el número de bits necesarios para enumerarla, y en consecuencia, será más segura.

Finalmente, el mensaje que se muestra para cada contraseña va acompañado por el tiempo en que se demoraría una computadora promedio en enumerar la contraseña mediante un ataque de fuerza bruta, permitiendo así una mayor claridad respecto a la seguridad de la contraseña ingresada.

A. Tiempos de entrenamiento

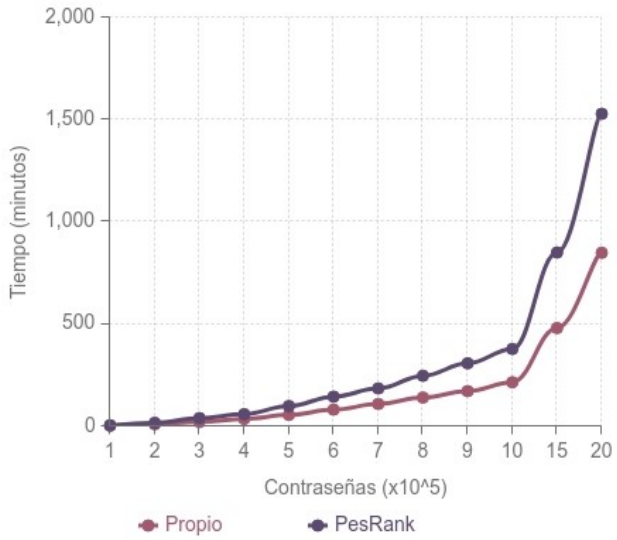

Figura 3: Comparación de los tiempos de ejecución para grandes valores

A partir de la **Figura 3**, se puede observar un comportamiento creciente en los tiempos de ejecución de ambos algoritmos, siendo a partir de 1'000.000 contraseñas, el punto clave en el que se puede apreciar claramente la diferencia entre los tiempos del algoritmo implementado en este artículo vs el de PESRank. La mejora en los tiempos de ejecución se justifica en que el algoritmo de PESRank realiza múltiples operaciones en memoria a través del uso de listas, lo que causa un mayor grado de complejidad, en comparación al algoritmo propuesto que emplea operaciones directamente con la bases de datos PostgresSQL, cuyas consultas (queries) resultan más rápidas y por ende, el tiempo de ejecución es menor. Por otra parte, la diferencia también es apreciable para valores pequeños de contraseñas (1000 a 90000) como se observa en la **Figura 4**.

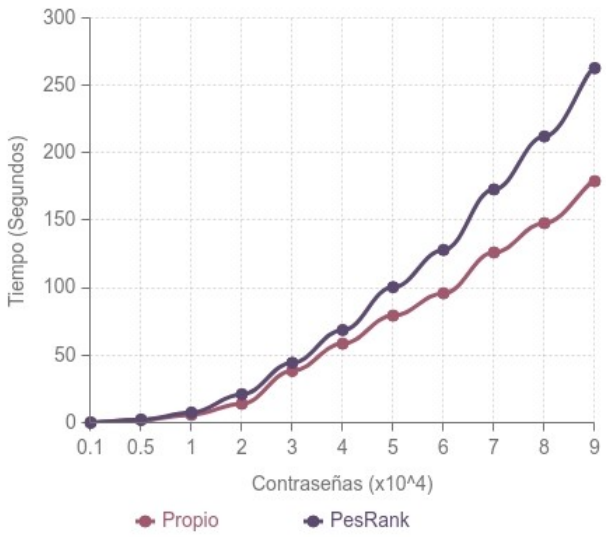

Figura 4: Comparación de los tiempos de ejecución para pequeños valores

B. Desempeño

Con base a la realización de 1000 peticiones a nivel local a la API, se obtuvo como tiempo promedio aproximado de respuesta el valor de 0.10 segundos y una desviación estándar de 0.05610, lo que permite evidenciar en la **Figura 5**, lo cercano que se encuentran entre sí los valores de tiempo, y la rapidez que posee el servidor para responder en promedio las peticiones de contraseñas.

Por otra parte, al realizar las peticiones de manera online, se obtuvo como tiempo promedio aproximado de respuesta el valor de 6.03 segundos y una desviación estándar de 0.82 , y así fue posible observar en la **Figura 6** una diferencia de tiempo en comparación a lo realizado a nivel local,

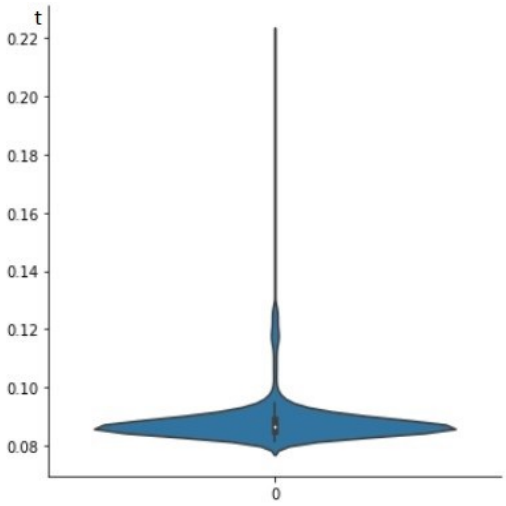

Figura 5: Comparación de los tiempos de ejecución para pequeños valores

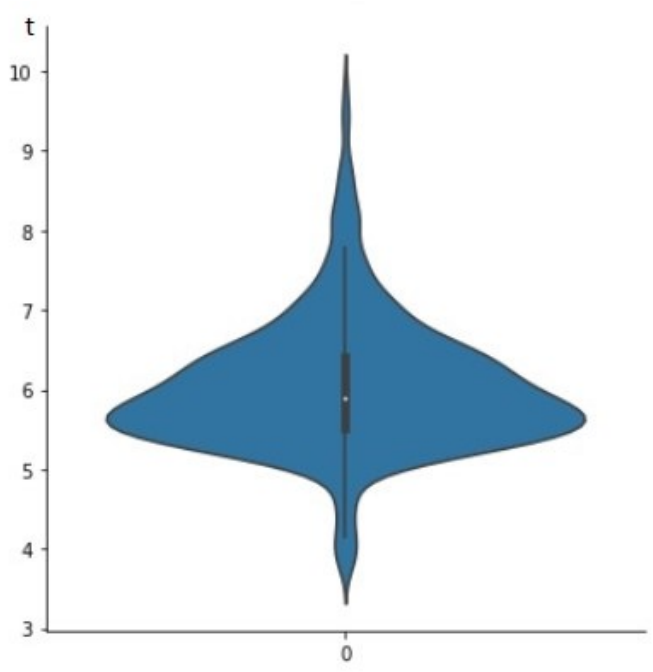

Figura 6: Comparación de los tiempos de ejecución para pequeños valores

Conclusión

Se desarrolló una herramienta capaz de estimar el grado de fortaleza de cualquier contraseña ingresada, con base a un entrenamiento previo de conjuntos de contraseñas proporcionados por PassGAN. Esta misma se realiza con la intención de alertar a los usuarios de los niveles de seguridad que están manejando en sus diferentes cuentas, para que de esta manera sean conscientes si se encuentran en un nivel aceptable o son blancos fáciles de robar credenciales. La herramienta se desplegó tanto como aplicación móvil como web, se consiguió una mejora considerable en los tiempos de uso con respecto a los algoritmos bases. Adicionalmente, se presenta una actualización constante de contraseñas, lo cual permitirá adaptarse a contraseñas nuevas, que se traduce en una mejor estimación a medida que se extienda su uso. Finalmente, este proyecto permite a las empresas poner a prueba las credenciales de sus usuarios, para tomar las decisiones correspondientes a los niveles de seguridad que se encuentren, y con ello obtener una mejora en la seguridad de sus datos.

Referencias

Baruffaldi, J. M., & Uzal, L. (2017). Redes neuronales adversarias para el reconocimiento de malezas. In *Xx concurso de trabajos estudiantiles-jaiio 46 (córdoba, 2017)*.

Beck, K. (2000). *Extreme programming explained: embrace change*. addison-wesley professional.

Bošnjak, L., Sreš, J., & Brumen, B. (2018). Brute-force and dictionary attack on hashed real-world passwords. In *2018 41st international convention on information and communication technology, electronics and microelectronics (mipro)* (pp. 1161–1166).

David, L., & Wool, A. (2020a). *Online password guessability via multi-dimensional rank estimation.* Retrieved from https://arxiv.org/abs/1912.02551

David, L., & Wool, A. (2020b). *Pesrank algorithm.* Retrieved from https://github.com/barak-itkin/PESrank

Dell'Amico, M., & Filippone, M. (2015, October). Monte Carlo Strength Evaluation: Fast and Reliable Password Checking. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (pp. 158–169). Denver Colorado USA: ACM. Retrieved 2021-05-05, from https://dl.acm.org/doi/10.1145/2810103.2813631 doi: 10.1145/2810103.2813631

Dorsey, B. (2017). Passgan dataset.. Retrieved from https://github.com/brannondorsey/PassGAN/releases

Hitaj, B., Gasti, P., Ateniese, G., & Perez-Cruz, F. (2017). Passgan: A deep learning approach for password guessing. In *International conference on applied cryptography and network security* (pp. 217–237).

L., D., & A, W. (2019). Poly-logarithmic side channel rank estimation via exponential sampling. In M. Matsui (Ed.), *Topics in cryptology – ct-rsa 2019. ct-rsa 2019. lecture notes in computer science* (Vol. 11405). Cham: Springer. Retrieved from https://doi.org/10.1007/978-3-030-12612-4_17 doi: 10.1007/978-3-030-12612-4_17

Mendoza, M. (2017). *Una mirada a la complejidad computacional y seguridad en la practica de los algoritmos md5 y des.* Retrieved from http://repositorio.utp.edu.co/dspace/bitstream/handle/11059/8945/T005.8%20M539.pdf?sequence=1&isAllowed=y

Olcott, J. (2019). *Cybersecurity vs. information security.* Retrieved from https://www.bitsight.com/blog/cybersecurity-vs-information-security

Raza, M., Iqbal, M., Sharif, M., & Haider, W. (2012). A survey of password attacks and comparative analysis on methods for secure authentication. *World Applied Sciences Journal*, *19*(4), 439–444.

Security, O. (2020). *Penetration testing with kali linux*.

Spears, J. L., Barki, H., & Barton, R. R. (2013). *Theorizing the concept and role of assurance in information systems security* (Vol. 50) [Pages 598-605,]. Information Management. Retrieved from https://doi.org/10.1016/j.im.2013.08.004. doi: 10.1016/j.im.2013.08.004.

Stonebraker, M., Rowe, L., & Hirohama, M. (n.d.). Retrieved from https://dsf.berkeley.edu/papers/ERL-M90-34.pdf

Williams, R. (2010). *Intels core i7-980x extreme edition ready for sick scores?* Retrieved

from https://techgage.com/article/intels_core_i7-980x_extreme_edition_-_ready_for_sick_scores/8/

Zhou, Y., & Feng, D. (2005). *Side-channel attacks: Ten years after its publication and the impacts on cryptographic module security testing.* Retrieved from https://eprint.iacr.org/2005/388.pdf