Universidad de La Habana
Facultad de Matemática y Computación

Implementación de una API de autenticación gráfica basada en Passpoints

Trabajo de Diploma
presentado en opción al título de
Licenciado en Ciencias de la Computación

Autor:
Alex Sánchez Saez

Tutores:

MSc. Joaquín Alberto Herrera Macías

MSc. Evaristo José Madarro Capó

MSc. Lisset Suárez Plasencia

La Habana, 24-12-2024

Dedicación

Agradecimientos

Agradecimientos

Opinión del tutor

Opiniones de los tutores

Resumen

La autenticación es crucial para la protección de los usuarios y sus datos. Debido a las debilidades que aparecen en las contraseñas alfanuméricas por la acción de los usuarios, se han desarrollado nuevos enfoques como son los basados en autenticación gráfica. Uno de estos sistemas es el *Passpoints* que se destaca por su seguridad y facilidad de uso. En este trabajo se presenta una implementación propia de dicho sistema, resultado de un exhaustivo estudio del sistema en cuestión. Dicho estudio abordó tanto el funcionamiento como la seguridad del sistema *Passpoints*, identificando sus debilidades y explorando las propuestas existentes para mitigarlas. Para la implementación principal de este sistema, se llevaron a cabo otras implementaciones intermedias esenciales para su desarrollo completo. Para ello se realizó un análisis exhaustivo de los métodos de discretización disponibles con el fin de seleccionar el más efectivo y eficiente para su posterior traducción a código de programación, así como una investigación referente a la adaptación de este sistema a la variedad de resoluciones de pantalla y tamaños de imagen actuales, permitiendo la adaptación de esta implementación a cualquier tipo de dispositivo. Este proceso es fundamental para convertir el sistema en un producto real que pueda ser evaluado por usuarios reales en diferentes medios.

Abstract

Authentication is crucial for protecting users and their data. Due to the weaknesses that appear in alphanumeric passwords as a result of user actions, new approaches have been developed, such as those based on graphical authentication. One of these systems is Passpoints, which stands out for its security and ease of use. This work presents our own implementation of this system, the result of an exhaustive study of the system in question. This study addressed both the functioning and security of the Passpoints system, identifying its weaknesses and exploring existing proposals to mitigate them. For the main implementation of this system, other essential intermediate implementations were carried out for its complete development. To achieve this, a comprehensive analysis of available discretization methods was conducted to select the most effective and efficient for subsequent translation into programming code, as well as research regarding the adaptation of this system to the variety of current screen resolutions and image sizes, allowing the adaptation of this implementation to any type of device. This process is fundamental to turning the system into a real product that can be evaluated by real users across different media.

Índice general

Índice de figuras

Ejemplos de código

Introducción

Desde la popularización del internet en los años 90, es creciente la tendencia a almacenar enormes cantidades de datos en los distintos servicios en línea, desde comercios electrónicos, redes sociales, aplicaciones bancarias, hasta servicios de *streaming*. Servicios como estos han crecido exponencialmente, así como la cantidad de usuarios que los consumen.

La necesidad de mantener segura y privada la información de los usuarios, muchas veces sensible, originó el uso de sistemas de autenticación seguros, condicionados al hecho de ser eficientes y fáciles de utilizar. La autenticación es el proceso mediante el cual un sistema verifica la identidad del usuario que intenta acceder a un recurso protegido, por ello constituye un pilar fundamental de la seguridad informática. Existen tres tipos principales de autenticación, basada en tokens, basadas en conocimiento y la autenticación biométrica. Cada uno de estos puede verse como la respuesta a una pregunta, de cara a quien intenta acceder al recurso protegido: ¿qué tienes?, ¿qué sabes? o ¿quién eres?

La autenticación basada en tokens responde a la pregunta: ¿qué tienes?, y se basa en que el usuario posea un token de identidad que demuestre su autenticidad, como, por ejemplo, tarjetas de crédito, llaves físicas y digitales. Uno de los ejemplos más utilizados en diferentes tipos de aplicaciones es JWT (*JSON Web Token*) 28, que son utilizados sobre todo para conectar un cliente con un servidor web. Este consiste en el uso de un token de acceso que contiene los permisos y datos del usuario al sistema, este token iría codificado en la cabecera de la petición. OAuth 5 es un protocolo para la autenticación multifactor, usa tokens de vida corta y permisos granulares, lo que reduce el riesgo de robo de credenciales y phishing. Es ampliamente utilizado para la integración de autenticación con redes sociales en distintos tipos de aplicaciones.

¿Quién eres?, es la pregunta cuya respuesta está vinculada a la autenticación biométrica. En esta se pide al usuario que demuestre su identidad a partir de datos que provengan de sí mismo, como pueden ser las huellas dactilares en el conocido Touch ID 21, reconocimientos faciales3 y reconocimiento de voz. A pesar de que este tipo de autenticación destaca por su seguridad, tiene la desventaja de requerir hardware especial para su uso.

Por último, el método más utilizado es el basado en conocimiento, el cual se

puede ver como la respuesta a la pregunta ¿qué sabes? Desde hace muchos años, las contraseñas alfanuméricas han sido el estándar en este tipo. Sin embargo, debido a la evolución de la capacidad de cómputo y el desarrollo de diferentes tipos de ataques, como son los ataques de diccionarios 27, fuerza bruta 1 o rainbow table 46, se ha visto debilitada su seguridad al punto de hacerlas inseguras para el usuario común. Estudios han demostrado que, para los usuarios es difícil recordar contraseñas alfanuméricas con un alto nivel de aleatoriedad y de gran longitud. Creando contraseñas débiles y fáciles de predecir computacionalmente. Esto plantea la necesidad de desarrollar alternativas más robustas y adaptadas a los desafíos actuales. En respuesta a esto se han propuesto las contraseñas gráficas como una alternativa.

La principal diferencia entre las contraseñas gráficas y alfanuméricas reside en la naturaleza de la información a memorizar. En el caso de las alfanuméricas se memorizan conjuntos de caracteres y en el caso de las gráficas se utiliza información visual que es más fácil de recordar por los usuarios. Estudios como 31, 37, 29 avalan la anterior idea a través de la comparación de la capacidad de memoria visual y verbal, con la demostración de que las imágenes se analizan tanto verbal como visualmente 37, a diferencia de las palabras que se analizan solo verbalmente. Esto sitúa a las contraseñas gráficas como una buena alternativa, más segura y fácil de usar que las alfanuméricas.

Entre los sistemas basados en contraseñas gráficas destaca el *Passpoints* 47 por su seguridad y usabilidad. Este es un sistema en el que el usuario selecciona 5 puntos ordenados de una imagen. Llevar a cabo la implementación de este sistema, así como analizar su seguridad y resistencia a ataques es una problemática de interés, pues puede ayudar a mejorar la seguridad de los datos en diferentes aplicaciones. Así como proporcionar una mayor seguridad sobre todo a personas mayores, cuya memoria o poca adaptación a las tecnologías modernas puede conducir a poner en riesgo su seguridad en línea al utilizar contraseñas predecibles.

El presente trabajo propone una implementación del sistema de autenticación gráfica *Passpoints*, a través de una aplicación práctica, así como una validación de la seguridad del mismo. Como novedad de esta investigación se tiene la implementación personalizada de la contraseña gráfica *Passpoints* cuyo análisis de seguridad reafirma su superioridad en cuanto a seguridad y resistencia ante ataques respecto a las contraseñas alfanuméricas.

Problema Científico

El problema científico planteado en el presente trabajo es: ¿cómo hacer una implementación práctica del sistema de autenticación gráfica *Passpoints*?.

Objeto de Estudio y Campo de Acción

El objeto de estudio es: implementación práctica del sistema de autenticación gráfica *Passpoints*. El campo de acción, el *Passpoints*.

Hipótesis

Se plantea la hipótesis: se puede crear una implementación práctica, usable y segura del sistema de autenticación Passpoints.

Objetivos

Objetivo General

El objetivo general del presente trabajo es hacer una implementación práctica del sistema de autenticación gráfica Passpoints.

Objetivos Específicos

- Valorar la usabilidad del sistema implementado.
- Valorar la seguridad del sistema implementado.
- Crear una plataforma para recolectar datos para futuros estudios.

Estructura de la tesis

El presente trabajo está dividido en 3 capítulos. En el primero se presenta el estado del arte de la autenticación gráfica, enunnciando los diferentes tipos de contraseñas gráficas, así como ejemplos e implementaciones de algunos de ellos. Se muestra en que consiste *Passpoints* así como su origen, ventajas y desventajas, variaciones e implementaciones del mismo. Se explicarán además conceptos utilizados en el desarrollo del presente trabajo, región de tolerancia, punto r-seguro y problema del hash. Se enuncian y explican los diferentes métodos de discretización estudiados durante la investigación como son la Discretización Robusta, Discretización Centrada, Discretización Optimal y Discretización mediante polígonos de Voronoi. Se presenta un análisis de estos métodos así como un análisis de los inconvenientes que trae consig utilizarlos.

En el segundo capítulo se hace una propuesta de implementación para este sistema, definiendo que discretización escogida y los problemas que suponen utilizar dicho

método. Se explica como se manejan los diferentes tamaños de imágenes y pantallas para mantener la consistencia de la contraseña en diferentes dispositivos. Además se muestran los ataques de fuerza bruta y diccionario escogidos para validar la resistencia de la implementación a los mismos.

El tercer capítulo aborda la fase de implementación y experimentación del sistema de autenticación propuesto. Inicialmente, se describen las decisiones arquitectónicas y de diseño que guiaron la implementación, junto con una descripción de la estructura del código y los componentes principales. Se proporciona una visión general de la implementación de los algoritmos de discretización y manejo de la interfaz de usuario para diferentes dispositivos. Posteriormente, se presenta el diseño experimental concebido para evaluar la robustez del sistema frente a ataques específicos. Se justifica la selección de estos ataques y se describe el protocolo experimental seguido. La presentación de los resultados de estos experimentos se acompaña de un análisis exhaustivo, destacando los puntos fuertes y las áreas de mejora identificadas durante el proceso de validación. Este análisis permite extraer conclusiones iniciales sobre la viabilidad y seguridad del sistema implementado.

Capítulo 1

Preliminares

1.1. Tipos de contraseñas gráficas

La amplia gama de contraseñas gráficas disponibles se puede organizar en una clasificación que facilita su estudio y comparación. A continuación, se presenta el análisis de cuatro categorías que abarcan la mayoría de los enfoques existentes: contraseñas basadas en el reconocimiento, las cuales dependen de la identificación de elementos visuales; contraseñas basadas en el dibujo, donde el usuario crea un patrón personalizado; contraseñas basadas en *Clicks*, que utilizan la selección secuencial de puntos en una pantalla; y, por último, los esquemas híbridos, que combinan elementos de las categorías anteriores. Cada una de estas categorías representa un enfoque distinto en cuanto a la interacción del usuario y presenta diferentes ventajas y desventajas en términos de seguridad y usabilidad

1.1.1. Contraseñas basadas en reconocimiento

Los sistemas basados en el reconocimiento son aquellos donde el usuario debe reconocer su contraseña de entre un conjunto de otras contraseñas o elementos distractorios, de tal forma que solo el usuario auténtico sea capaz de acceder al recurso protegido.

El sistema *Deja Vu*, desarrollado por 12), ilustra un enfoque de contraseña gráfica basado en reconocimiento. En este esquema, el usuario configura su contraseña seleccionando un conjunto de imágenes que conforman un portafolio 3.1. El proceso de autenticación requiere que el usuario identifique correctamente las imágenes de su portafolio, las cuales se presentan mezcladas con imágenes señuelo.

Para garantizar la memorabilidad de la contraseña se pide hacer un pequeño entrenamiento por parte del usuario en el que deberá identificar las imágenes del portafolio de un conjunto con imágenes señuelo, esto disminuye la experiencia de usuario en la

fase de registro.

Este sistema aprovecha la habilidad humana para recordar imágenes previamente vistas, lo que, según 12, las hace más resistentes a ataques de ingeniería social al dificultar la elección de contraseñas débiles y su intercambio.

Pass Faces 45, 42 es otro ejemplo de sistemas basados en el reconocimiento, esta vez basado en la capacidad de reconocimiento facial. En este sistema el usuario debe seleccionar un conjunto de caras, al igual que en el anterior, en el momento de autenticación el usuario debe seleccionar 4 caras de una cuadrícula de $3x3$ caras 3.2.

Este sistema se ha demostrado que es predecible 45, pues la selección de las caras está sujeta a características étnicas, raciales o tendencias a seleccionar caras más atractivas.

1.1.2. Contraseñas basadas en dibujo

En contraste, las contraseñas basadas en dibujo son aquellas donde se le pide al usuario dibujar a mano su contraseña, con el objetivo de reconocer los trazos o secuencias que cree el mismo. Están inspirados en las firmas que se utilizan para garantizar la veracidad de las personas en documentos oficiales o elementos similares.

Un ejemplo muy conocido de este tipo de contraseña gráfica,son los patrones de desbloqueo de *Android*, permiten al usuario crear un patrón enlazando puntos en una cuadrícula 3.3. Aunque presentan vulnerabilidades como los ataques de *Smudge* 4, en el que después de colocado el patrón queda una marca del mismo en la pantalla producto de la grasa o suciedad de los dedos 3.4 y una reducción del espacio de contraseñas al no permitir la repetición de puntos. Otra vulnerabilidad de este sistema es el ataque *Shoulder Surfing*, este consiste en que un atacante observa, graba o registra los eventos de la pantalla del usuario mientras este dispone su contraseña durante fase de autenticación o registro.

Free Form Draw 26 ofrece mayor libertad al permitir dibujar cualquier figura y registrarla como contraseña. Dicho dibujo debe ser reproducido durante la fase de autenticación. En 26 siguen la premisa de que un usuario al escribir o dibujar realiza los trazos de forma inconsciente y automatizada, una persona deja no solo su estado mental, sino un estado de la consciencia por lo que se tiene el dibujo resultante como: "un diagrama del inconsciente", por lo que estos trazos son únicos para cada individuo.

Dibujar exactamente lo mismo con precisión quirúrgica en el momento de reproducir cada trazo es prácticamente imposible, es necesario tener una forma de verificar todos los posibles futuros dibujos que el usuario utilizará en la fase de autenticación. Para esto se le pide al usuario ingresar durante la fase de registro su contraseña o firma digital varias veces 3.5 y 3.6, esto se procesa y se utiliza un modelo de Regresión Polinomial 13 para poder predecir y verificar la contraseña del usuario en caso de que sus trazos tengan muchas fluctuaciones 3.7.

Si la cantidad de fluctuaciones es mayor, se utiliza un modelo de regresión B-spline 20, este fusiona la suavidad polinomial y la precisión de la influencia local de la aproximación o interpolación lineal 3.8.

1.1.3. Contraseñas basadas en *Clicks*

Las contraseñas basadas en *Clicks* se basan en la selección de elementos en un orden específico solo conocido por el usuario auténtico. Ejemplos son las contraseñas basadas en historias o narrativas 30, 19, donde se seleccionan puntos en imágenes semánticamente relacionadas para crear una historia.

En el caso de *Semantic Lock* 30 definen una contraseña como una secuencia de k imágenes seleccionadas de un conjunto de $n > k$ imágenes, dichas k imágenes deben estar dispuestas por el usuario de forma que su unión y orden represente una historia 3.9.

Passpoints 47 fue diseñado en el 2005 por Susan Wiedenbeck,inspirado en el modelo propuesto por *Blonder* 8, basa su funcionamiento en que un usuario seleccione un conjunto ordenado de 5 puntos en una imagen como su contraseña en la fase de registro 3.10.

Cued Click Points 11 propone una mejora en usabilidad al seleccionar un punto por cada imagen de un conjunto de imágenes en lugar de varios puntos de una misma imagen, esto aumentaría la carga computacional de implementar dicho sistema.

De Passpoints se han derivado variantes como *PassPositions* y *PassPositions* 2 48, que agregan información posicional para aumentar la seguridad y fortaleza ante ataques. Logran esto haciendo que el cálculo del *hash* de un punto se haga tomando como referencia la posición del anterior 3.11.

Pass Maps 41, emplea un mapa mundial para aumentar el espacio de contraseñas 3.13 y, por tanto, su resistencia a ataques de diccionario o fuerza bruta 2.

Otro esquema basado en clics es *PassGo* 44, inspirado en el juego "Go". En este los usuarios deben selecccionar intersecciones en una cuadrícula.

Cada intersección tiene un área sensible alrededor para compensar pequeñas inexactitudes en la entrada del usuario. Se muestran indicadores de puntos y líneas para las intersecciones seleccionadas y las líneas dibujadas entre ellas 3.12. Luego la contraseña se codifica como una secuencia de pares de coordenadas bidimensionales .

1.1.4. Esquemas Híbridos

Los esquemas híbridos, como *Gra-Pin* 22 o *PassMatrix* 43, combinan diferentes enfoques de contraseñas gráficas (incluso con alfanuméricas), como la combinación de un PIN con la selección de imágenes o la selección de secuencias de imágenes como un PIN.

En el caso de *Gra-Pin* 22 se pide al usuario seleccionar un número secreto de 2 dígitos, luego se le pide seleccionar una imagen secreta de entre otras 9 opciones, se pide seleccionar una operación aritmética y una posición secreta para un Pin.

Esta combinación de elementos compone su contraseña. En la fase de autenticación se pide al usuario contar cuantas veces aparece su imagen secreta de una cuadrícula de $5x5$ imágenes, realiza la operación aritmética seleccionada en la fase anterior entre el número de veces que aparecía su imagen secreta y el número secreto de dos cifras seleccionado durante el registro, por último se pone el resultado de esta operación en la posición seleccionada en el registro en el Pin, poniendo el resto de dígitos aleatorios como se puede ver en 3.14 y 3.18.

Este método está pensado para mantener la resistencia que las contraseñas gráficas ofrecen presentando mayor resistencia ante ataques de tipo *Shoulder Surfing* 24

1.2. Passpoints

Como se mencionó anteriormente, el sistema Passpoints 47 se basa en la memorización de una secuencia de cinco puntos seleccionados en una imagen durante la fase de registro. En la fase de autenticación el usuario tendrá que seleccionar dichos puntos en el mismo orden que en la fase de registro. En este sistema cualquier imagen puede ser utilizada (pinturas, fotos naturales, fotos familiares, etc), y puede ser seleccionada por el usuario o proveídas por el sistema. La imagen debe tener cientos de puntos probables de ser seleccionados y deben estar diseminados de forma homogénea para mayor seguridad. Por motivos de seguridad, el sistema no almacena de forma explícita la contraseña, sino un hash de la misma, esto trae consigo el problema de identificar el usuario legítimo, ya que es muy poco probable que se digiten exactamente los mismos puntos durante las fases de registro y autenticación, haciendo que los hashes sean diferentes si esto no ocurriese. Para solventar este problema es necesario agregar un margen de error para la selección de los puntos; una región de tolerancia. Para lograr esto se utiliza una discretización de la imagen, lo que reduce el espacio de contraseñas y aporta información relevante para llevar a cabo ataques de diccionario 49. además permite la aparición de falsos positivos y negativos en la autenticación debido a la forma de las regiones calculadas utilizando la discretización. Una discusión acerca de la importancia del mecanismo de discretización en los esquemas de contraseñas gráficas y de los diferentes métodos de discretización conocidos hasta el momento puede verse en 7, 10, 6, 23. Otros aspectos negativos a destacar en este sistema son: algunas regiones en la imagen son más propensas a ser seleccionadas por el usuario para formar su contraseña 32. Dado que este sistema basa su funcionamiento en la selección de 5 puntos en la imagen, la fase de registro y de autenticación pueden extenderse lo que conlleva a que sean vulnerables ante ataques de tipo *Shoulder-Surfing* 33. Si el conjunto de puntos seleccionados por el usuario no sigue un patrón alea-

torio es considerada débil y es susceptible a ataques de diccionarios 33. En varios artículos publicados recientemente 18, 40, 39, 14, 15, 16, 17 se proponen tests para evitar el registro de contraseñas gráficas no aleatorias. Estos resultados unidos a la propuesta en 25 de un modelo probabilístico capaz de medir el nivel de autenticidad de un usuario, conllevan a un aumento significativo en la seguridad de este sistema y por consiguiente lo convierten en una de las alternativas más prometedoras ante las contraseñas alfanuméricas

1.2.1. ¿Por qué usar Passpoints?

La notoria falta de seguridad de las contraseñas alfanuméricas, debido a la contradicción que presentan las mismas 50, es una señal de que se requieren opciones más seguras y sencillas de usar. En este escenario, el sistema *Passpoints* se erige como una buena alternativa a las tradicionales contraseñas alfanuméricas. La seguridad de este método de autenticación gráfica reside en su espacio de contraseñas Q^L, donde Q es el tamaño del alfabeto y L la longitud de la contraseña 25. Mientras que en el caso de las contraseñas alfanuméricas este espacio consiste en la cantidad de cadenas que se pueden formar con un alfabeto específico, en *Passpoints* es la cantidad de combinaciones de puntos (píxeles) de la imagen que se utilice.

En la actualidad, los tamaños de imágenes que se manejan rondan los cientos de miles de píxeles, lo que incrementa aún más el espacio de contraseñas de *Passpoints*. Esto tiene un impacto positivo en su nivel de seguridad y resistencia a ataques de fuerza bruta. Al ser un método novedoso y poco conocido, no existen grandes bases de datos de contraseñas *Passpoints*, a diferencia de la enorme cantidad de información y recopilaciones de contraseñas alfanuméricas disponibles en Internet. Esta falta de información le otorga una mayor resistencia a *Passpoints* ante ataques de diccionario 35.

Además, al explotar la capacidad humana de reconocer patrones en imágenes, *Passpoints* facilita el recuerdo de las contraseñas para cualquier tipo de persona, desde niños y adolescentes hasta personas de la tercera edad. Esto contrasta con las contraseñas alfanuméricas, donde, para garantizar la seguridad, se hace necesario que los usuarios memoricen largas cadenas con altos niveles de aleatoriedad. En 34 se ratifica la posición de *Passpoints* como el método de autenticación gráfica más conveniente, a través de una comparación y evaluación crítica de los diferentes métodos existentes.

1.2.2. Discretización

Es sencillo percatarse de la baja probabilidad de que un usuario seleccione siempre exactamente el mismo píxel en las fases de registro y autenticación, situación que

se agrava aún más en escenarios como el de los teléfonos móviles, cajeros y otras situaciones donde el usuario no posee un puntero digital. Es por esto que se hace necesario dar un margen de error a cada punto de la contraseña *Passpoints*. A la región definida por el punto y su margen de error se le conoce como región de tolerancia. Una forma eficiente y segura de introducir esta región de tolerancia en un sistema de autenticación gráfica *Passpoints* es utilizando una discretización.

1.2.3. Región de Tolerancia

La región de tolerancia 9, 25 de un punto p se define como el conjunto de puntos de la imagen que son aceptados como válidos durante la autenticación para el punto original p_0, y se denota como R_T. Sea I el conjunto de píxeles de la imagen, y f una función tal que para todo punto de la imagen devuelve 1 si es aceptado y 0 en caso contrario. Entonces, R_T quedaría definido como:

$$R_T = \{p, p \in I \cap f(p) = 1\} \tag{1.1}$$

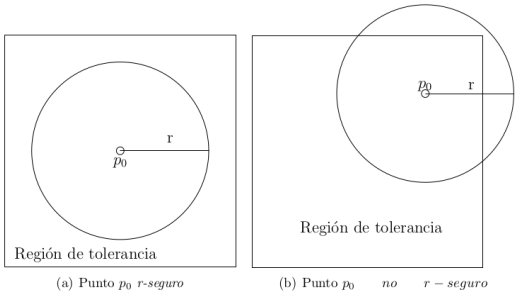
Puede interpretarse la región de tolerancia como el error permitido al usuario en el momento de seleccionar su contraseña.

1.2.3.1. Punto r-seguro

Un punto se considera $r - seguro$ 9, 25 para un radio r si y solo si todo punto que está a una distancia r de él se incluye en la región de tolerancia. Sea I el conjunto de píxeles de una imagen, p_0 un punto de la imagen y R_T la región de tolerancia. Se dice que p_0 es $r - seguro$ si y solo si:

$$\forall p \in I : d(p_0, p) < r \Rightarrow p \in R_T \tag{1.2}$$

2



(a) Punto p_0 *r-seguro* (b) Punto p_0 *no* $r - seguro$

Figura 1.1: Ejemplo de punto r-seguro y punto no r-seguro, Fuente: 25

1.2.4. Problema del Vértice

Este problema surge durante la fase de registro 7, 9, 25, cuando el usuario puede seleccionar un punto que no es $r-seguro$. Hay dos casos posibles: seleccionar un punto localizado exactamente en los vértices o aristas de la región de la partición, o seleccionar un punto que está situado a una distancia $d < r$ de los mismos. El primer caso plantea un problema de decisión para determinar la región de tolerancia del punto. Es necesario discretizar las imágenes de tal manera que cada punto pertenezca a una región de tolerancia.

1.2.5. Problema del Hash

Por temas de seguridad, las contraseñas no pueden almacenarse en texto claro. Es necesaria una forma segura de representarlas tal que para cada contraseña exista un identificador único y que estas no sean recuperables desde dicho identificador. Las funciones *hash* 9, 25 encajan perfectamente con esta definición, por lo que son un buen recurso a utilizar para almacenar las contraseñas. Sin embargo, usando un *hash* surge la problemática de que cada punto de la región de tolerancia tendrá un *hash* diferente, impidiendo así que guardar el *hash* de los puntos seleccionados sea una buena opción. Utilizando como parámetro de la función *hash* no solo un punto, sino toda su región de tolerancia, no solo se garantiza que se puedan guardar los *hashes* de las contraseñas, sino que también aumenta la cardinalidad del espacio de entrada de la función *hash*, lo que dificulta los ataques de fuerza bruta.

1.2.6. Métodos de Discretización

Es sencillo percatarse de la baja probabilidad de que un usuario seleccione siempre exactamente el mismo pixel en las fases de registro y autenticación, situación que se agrava aún más en escenarios como el de los teléfonos móviles, cajeros y otras situaciones donde el usuario no posee un puntero digital. Es por esto que se hace necesario dar un margen de error a cada punto de la contraseña Passpoints, a la región definida por el punto y su margen de error se le conoce como región de tolerancia. Una forma eficiente y segura de introducir esta región de tolerancia en un sistema de autenticación gráfica Passpoints es utilizando una discretización

1.2.6.1. Discretización Robusta

Para evitar el problema del vértice 7, se utiliza un conjunto de tres particiones diferentes de la imagen. Esto garantiza que cada punto es $r-seguro$ en al menos una de dichas particiones. Esto se logra asegurando una separación de al menos r píxeles entre el punto y el borde de alguna de las tres particiones 10, 49. Se toman cuadrículas

de dimensiones $6r \times 6r$ y cada partición debe estar separada una distancia $2r$ del resto. Durante la fase de autenticación, debido a la construcción de las particiones, cualquier punto a una distancia $d \leq r$ del punto original pertenecerá al mismo cuadrante, lo que garantiza la autenticación del usuario ya que la salida de la función *hash* será la misma. Por otro lado, cualquier punto a una distancia mayor a $5\sqrt{2}r$ pertenecerá a otro cuadrante, lo que garantiza la no autenticación del usuario ilegítimo.

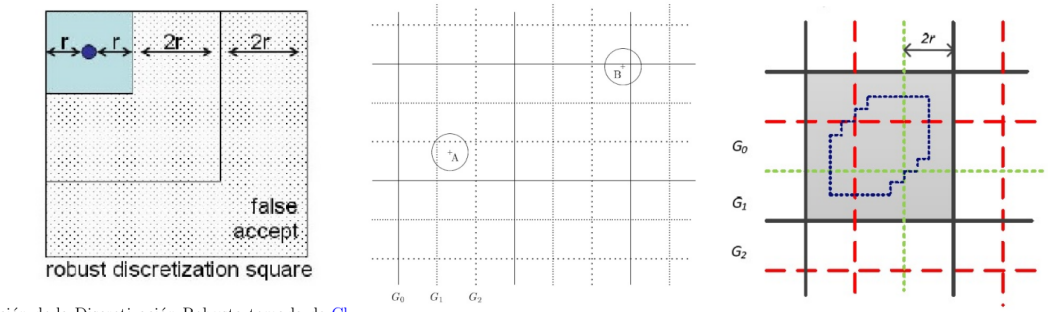


Figura 1.2: Regiones de la discretización robusta, Fuentes: 49 y 10.

1.2.6.2. Discretización Centrada

La Discretización Centrada 10 ofrece mejoras en usabilidad y seguridad en comparación con la discretización robusta. Esta técnica garantiza que la región de tolerancia esté centrada en el punto seleccionado para la contraseña, resolviendo así el problema del vértice. Al determinar una región de dimensiones $2r \times 2r$ centrada en el punto. Este método funciona encontrando, en cada dimensión de la imagen $(x,\ y)$, un segmento de longitud $2r$ en el cual el centro sea el punto originalmente seleccionado en el registro. Sea x un punto en la semirrecta numérica que comprende los valores entre 0 y m, donde m es el ancho o largo de la imagen, dependiendo de la dimensión que se quiera calcular. A partir de ese segmento, se divide el resto del intervalo $[0, m]$ en subintervalos de igual longitud.

En la mayoría de los casos, habrá sobrantes de tamaño d, donde d pertenece al intervalo $[0, 2r]$, por lo que si se almacena el valor de d es posible reconstruir la partición realizada comenzando en d, donde uno de estos subintervalos estará centrado en x. Una vez establecido el radio r y el punto de la contraseña x, se puede calcular la región de tolerancia. Se calcula el sobrante d que se utilizará luego en la fase de autenticación:

$$d = (x - r) \mod 2r \tag{1.3}$$

Determinar el intervalo exacto *i* donde se encuentra *x*:

$$i = \left\lfloor \frac{x - r}{2r} \right\rfloor \tag{1.4}$$

Una vez seleccionado el punto *x'* durante la fase de autenticación, se halla el intervalo *i'* donde este se encuentra:

$$i' = \left\lfloor \frac{x' - r}{2r} \right\rfloor \tag{1.5}$$

Nótese que *i* no está centrado en *x'*, pero $|x - x'| < r \rightarrow i = i'$. Por tanto, se utiliza *i* como componente de la contraseña.

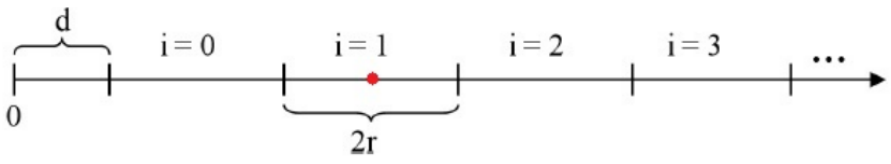


Figura 1.3: Ejemplo de recta numérica discretizada. centralmente, Fuente: 6

Este método de autenticación supone una mejora sustancial en cuanto a su complejidad de implementación, ya que elimina la necesidad de crear varias particiones en la imagen. Sin embargo, este método introduce un nuevo problema: la necesidad de almacenar el valor d en texto claro para poder efectuar la autenticación correctamente. Una posible solución a este problema sería encriptar este valor de forma reversible y almacenarlo junto al *hash* de la contraseña concatenándolo al mismo. Esto permitiría al sistema, durante la fase de autenticación, recuperar estos datos y determinar si los puntos seleccionados son válidos o no.

1.2.6.3. Discretización Óptima

Este método mantiene la filosofía de particionar la imagen generando una región centrada en el punto original de la contraseña, pero utiliza propiedades de la aritmética modular para construirla 6. Sean r el radio de tolerancia y X el punto seleccionado por el usuario, se calcula:

$$\begin{aligned} X \mod 2r \geq r &\rightarrow \phi = X \mod r \\ X \mod 2r < r &\rightarrow \phi = (X \mod 2r) - r \end{aligned} \tag{1.6}$$

De forma análoga se calcula:

$$\begin{aligned} Y \mod 2r \geq r &\rightarrow \varphi = Y \mod r \\ Y \mod 2r < r &\rightarrow \varphi = (Y \mod 2r) - r \end{aligned} \tag{1.7}$$

Estos valores (ϕ) y (φ) se almacenan en claro en el sistema junto a los *hashes*:

$$S_X = \frac{X - \phi}{2r}, \quad S_Y = \frac{Y - \varphi}{2r} \tag{1.8}$$

Durante la fase de autenticación, el usuario selecciona el píxel (X', Y'), utilizando los valores de (ϕ) y (φ) almacenados para ese usuario se calculan los *hashes* $S_{X'}$ y $S_{Y'}$ cumpliéndose que se garantiza la autenticación.

$$\|(X,Y) - (X',Y')\| < r \rightarrow S_{X'} = S_X \wedge S_{Y'} = S_Y \tag{1.9}$$

Este método es más eficiente que los descritos anteriormente, ya que su implementación a nivel computacional tiene una menor complejidad. Al tomar como base la aritmética modular se reduce la complejidad de los cálculos necesarios. No soluciona el problema del anterior método debido a que mantiene la necesidad de guardar texto claro junto con las contraseñas, en este caso son los valores (ϕ) y (φ). Aunque esto tiene una solución rápida que comparte con la discretización centrada

1.2.6.4. Discretización mediante Polígonos de Voronoi

Otras propuestas de discretización encontradas en la bibliografía es la hecha por Kirovski et al. 23, donde proponen utilizar diagramas de Voronoi. Partiendo de los puntos más probables de ser seleccionados en la imagen (conocidos como *Hotspots* en la literatura), propone aplicar una discretización de Voronoi ponderada usando una heurística para maximizar la entropía $H(P_w)$, tratando de obtener polígonos equiprobables. Su ventaja principal es que todos los polígonos de la partición obtenida poseen aproximadamente la misma probabilidad a priori de que el usuario escoja un punto de ese polígono. Esta propiedad parece ofrecer mejor resistencia a los ataques de diccionario basados en *Hotspots*. Sin embargo, en **zhu2013securityl** se afirma que la propuesta de 23 sigue dejando información en claro, útil para ataques de diccionario.

1.2.6.5. Problemas de los Métodos de Discretización

El uso de los métodos de discretización conlleva a ciertas limitaciones durante la autenticación. Una de estas es la falta de diferenciación entre los puntos que se encuentran dentro de la región de tolerancia. Todos los puntos reciben el mismo tratamiento, lo cual contradice el comportamiento esperado por parte del usuario legítimo, que debería seleccionar con mayor frecuencia los puntos más cercanos al punto original. Además, al representar la región de tolerancia como un polígono en lugar de un círculo, existe la posibilidad de obtener falsos positivos, es decir, puntos que se encuentran a una distancia mayor que el radio de tolerancia establecido pero que aún se consideran válidos como parte de la contraseña 8, 9. Otro problema se

presenta cuando existen puntos situados a la misma distancia del punto seleccionado como parte de la contraseña, siendo ambos válidos para el usuario legítimo. Sin embargo, uno de ellos puede quedar dentro de la región de tolerancia determinada por la discretización y el otro no, lo que genera falsos negativos. Además, al segmentar la imagen en cuadrículas, no se toman todos los puntos que deberían determinar la región de radio r alrededor del punto seleccionado como contraseña. Esto puede afectar la experiencia de usuario al utilizar el sistema 6, 9.

En general, las limitaciones de los métodos de discretización radican en el hecho de que definen la región de tolerancia como un polígono, mientras que la distancia se plantea en términos de un círculo. Además, el criterio utilizado para determinar si un punto es válido o no se basa únicamente en la distancia. Otra debilidad de estos métodos es la necesidad de almacenar información adicional para garantizar la autenticación 9, esto podría ser explotado para aumentar la efectividad de ataques de tipo diccionario. Por lo tanto, es importante abordar en trabajos futuros estas limitaciones y buscar mejoras en los métodos de discretización para lograr una mayor precisión en la autenticación y brindar una mejor experiencia de uso al usuario.

Capítulo 2

Propuesta

Como se vio en capítulos anteriores, la autenticación gráfica, específicamente las contraseñas gráficas basadas en Passpoints, ofrece una alternativa prometedora a la autenticación tradicional. En el presente capítulo, se propone una implementación de dicho sistema en una aplicación práctica y simple, un blog de notas privado. La elección de este ejemplo se fundamenta en su relevancia como aplicación que requiere un mecanismo de autenticación seguro para proteger la privacidad de los datos almacenados, además de permitir evaluar la escalabilidad del sistema. Esta implementación permitirá validar en un entorno real las ventajas de la autenticación con Passpoints, demostrando su viabilidad y seguridad. Asimismo, se desarrollará la API REST necesaria para integrar el sistema de autenticación con el blog de notas, así como la interfaz de usuario que permita la creación y gestión de Passpoints.

2.1. Región de tolerancia seleccionada

Debido a la variabilidad de tamaños de imagen y densidad de píxeles de pantalla, el radio de la región de tolerancia se redefinió como un valor $0 < r < 1$, el cual representa el porcentaje de píxeles de la imagen que abarca dicha región. De este modo, el tamaño en píxeles de la región de tolerancia es variable en función de la imagen. Esta estrategia abre la puerta a futuros trabajos, ofreciendo la posibilidad al usuario de seleccionar el tamaño de la región de tolerancia para su contraseña y simplificando el proceso de prueba de varios tamaños de la región. También ayuda a ajustar la representación de la región de tolerancia en distintos tamaños de pantalla. Para obtener el valor en píxeles del radio de la región de tolerancia se multiplica r por el mínimo entre el largo y ancho de la imagen,es decir, sea a el alto de la imagen, b el ancho, $d_{pixel} = min(b, a)$

2.2. Método de discretización seleccionado

Por todo lo explicado anteriormente se seleccionó la Discretización Óptima utilizando la variante enunciada en 6 con verificación de *hash* múltiple. Esto funciona introduciendo un error en el cálculo de la discretización de cada punto y repitiendo el *hash* del punto de forma recursiva k veces, es decir hallar el *hash* de la salida del anterior. Al hacer esto se vuelve la función de *hash* de cada punto más lenta, dando una capa de seguridad extra ante ataques de diccionario y al repetir *hash* varias veces previene que los puntos sean hallados utilizando *rainbow tables*, dando aún más seguridad al sistema. Para introducir este error en el cálculo de la discretización de cada punto se hace lo siguiente:

Sea

$$d((x,y)) : [0,a] \times [0,b] \to \left(\left\lfloor \frac{\lfloor x - \phi \rfloor}{q} \right\rfloor, \left\lfloor \frac{\lfloor y - \varphi \rfloor}{q} \right\rfloor \right)$$

el valor resultante de discretizar el punto (x,y) para la imagen I.

Sea r la región de tolerancia, q el cuantil o tamaño de las secciones en que se divide la imagen en cada dimensión, y ϕ y φ los offsets correspondientes a cada una. Para que el valor de la discretización tenga un error 0, es decir, que la división de la imagen en la discretización sea del mismo tamaño que la región de tolerancia, el valor de q debe ser $2r$.

Esto se debe a que el error es:

$$\pm \left\lfloor \frac{r}{q} \right\rfloor,$$

por lo que valores de $q \leq r$ hacen que haya un error $e \geq 1$.

Luego, durante la fase de verificación de un punto, se verifican el valor del punto y cada error de este. Es decir, sea h una función de hash definida como:

$$h(x,k) = \begin{cases} h'(x), & \text{si } k = 0, \\ h(h'(x), k-1), & \text{si } k > 0, \end{cases}$$

Entonces, sean $x = (a,b)$, el punto originalmente seleccionado por el usuario, y $x' = (a',b')$, un punto seleccionado durante la autenticación. Se cumple $d(x) = d(x')$ si $\exists i \in [-e,e]$ y $\exists j \in [-e,e]$ tales que:

$$h(d(x),k) = h(d(x' + (i,j)),k).$$

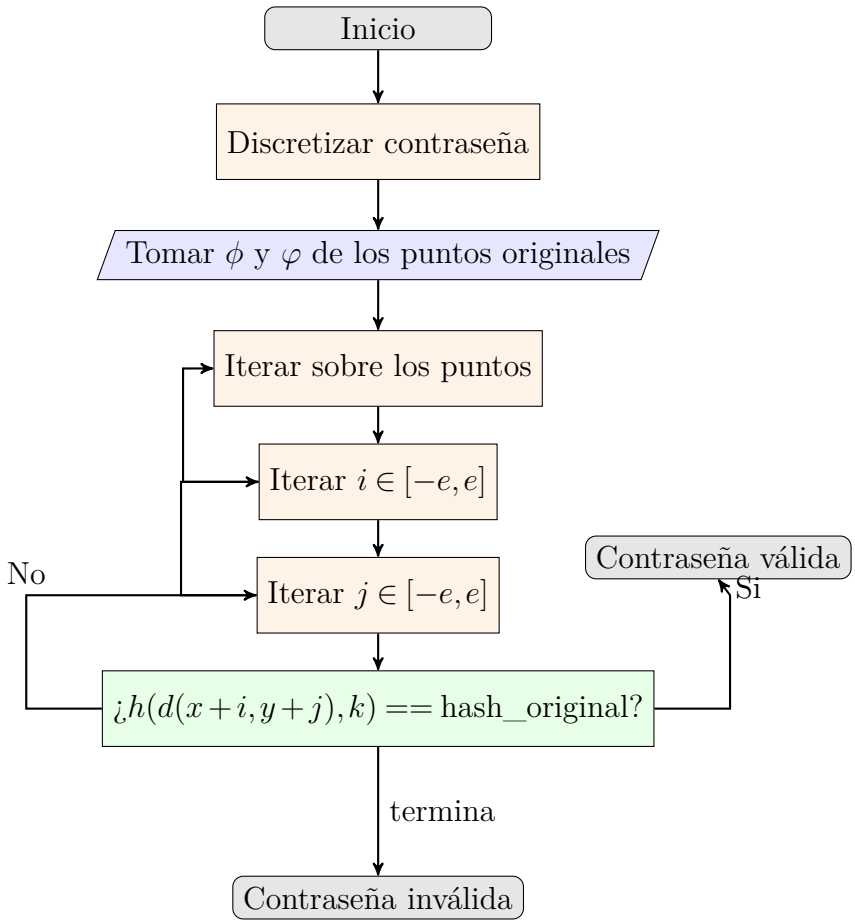


Figura 2.1: Diagrama de flujo de verificación de contraseña usando la Discretización Óptima

2.3. Funcionamiento de la aplicación

2.3.1. Flujo en la API

Para acceder al sistema, el usuario debe registrarse introduciendo sus credenciales y estableciendo su contraseña utilizando *Passpoints*. Durante el registro, se solicita al usuario que seleccione 5 puntos sobre una imagen, de una lista predeterminada por el sistema.La región de tolerancia se verá como un recuadro rojo semitransparente centrado en el punto seleccionado por el usuario en la imagen. En el servidor, se calcula la discretización de la imagen y el hash de los puntos seleccionados, almacenándose junto con los datos necesarios para la posterior autenticación. Para llevar a cabo la discretización de la imagen, se empleó la Discretización Óptima. Los datos necesarios para la autenticación, como (ϕ) y (φ), son guardados en texto claro para posteriores

estudios, pero en una aplicación real estos deben ser cifrados de forma reversible antes de ser guardados. En la fase de autenticación el usuario vuelve a seleccionar los 5 puntos anteriormente fijados en la fase de registro, teniendo el cuenta la región de tolerancia del sistema, dichos puntos irán a la API en donde se calculará su hash, si este coincide con el almacenado por el usuario legítimo se le dará acceso a las notas, en caso contrario se le devuelve un error. Los puntos utilizados para la autenticación y su estatus de fallido o no son almacenados también para posteriores estudios.

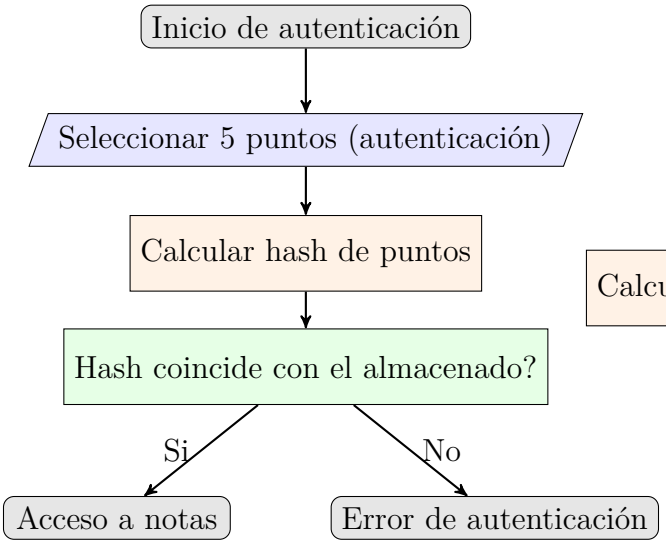


Figura 2.2: Diagrama de flujo autenticación con Passpoints

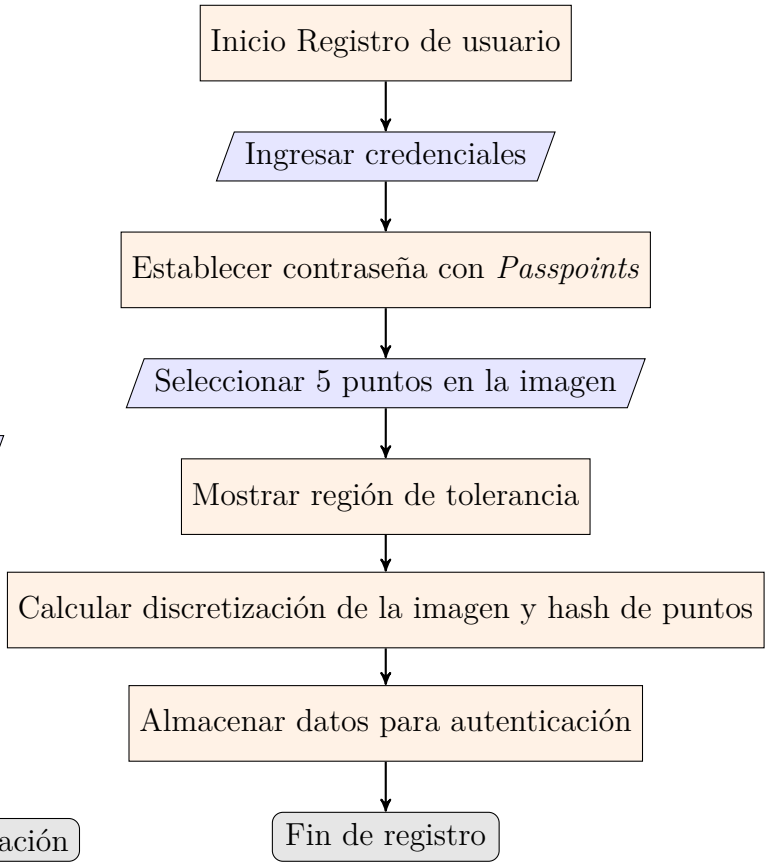


Figura 2.3: Diagrama de flujo registro con Passpoints

2.3.2. Adaptación a distintos tamaños de pantalla

Para hacer la aplicación adaptable a diferentes tamaños de pantalla se cambió la forma de tomar los puntos y transformarlos a coordenadas de imagen. Sean (I_X, I_Y)

las coordenadas de imagen del punto p_0 seleccionado por el usuario, (S_X, S_Y) sus coordenadas de pantalla, S_W el ancho de la ventana, S_h el alto de la ventana, I_W el ancho de la imagen, e I_h el alto de la imagen. Entonces, se calculan las coordenadas de imagen como:

$$I_X = \left\lfloor \frac{S_X}{S_W} \cdot I_W \right\rfloor$$
$$I_Y = \left\lfloor \frac{S_Y}{S_h} \cdot I_h \right\rfloor$$

Esto garantiza la consistencia en las coordenadas de los puntos en diferentes tamaños de pantalla e imagen.

2.4. Tecnologías a utilizar

Para la implementación de la propuesta, se utilizó el lenguaje *JavaScript* tanto en la interfaz de usuario como en la API. Específicamente, para la API se empleó *Supabase*, que proporciona una base de datos *PostgreSQL* y funciones *serverless* ejecutadas en entornos *JavaScript*. En cuanto a la creación de la interfaz de usuario, se utilizaron tecnologías web comunes como *HTML*, *CSS* y *JavaScript*, todas integradas mediante el *framework Vue*. El uso de *Vue* permitió agilizar el desarrollo de sitios web interactivos.

2.5. Validación del sistema

Para realizar una validación de usabilidad de este sistema se pide a los usuarios que den una valoración de su experiencia, para luego ser evaluadas automáticamente utilizando algún modelo de procesamiento de lenguage como **team2023gemini**. En cuanto a la seguridad se utilizará el ataque de diccionario descrito en **van2010purely**. Este ataque consiste en la creación de 3 diccionarios de contraseãs, los dos primeros utilizando herramientas de procesamiento de imágenes, específicamente segmentación y detección de bordes. Otro diccionario generado utilizando el modelo de atención visual **itti2000saliency**. Para generar el diccionario de ataque se propone en **van2010purely** un método de clusterización de los puntos para reducir la dimensionalidad de los diccionarios y evitar puntos redundantes, funciona haciendo que los puntos que estén a una distancia N, no necesariamente el radio de la región de tolerancia, se cuenten como el mismo. También se propone una forma de generar los diccionarios gastando menos recursos computacionales y timepo, se llega a esto a través de heurísticas para encontrar patrones comunes de los usuarios.

Capítulo 3

Detalles de Implementación y Experimentos

Conclusiones

Conclusiones

Recomendaciones

Recomendaciones

43

Bibliografía

Apostol, K. (2012). Brute-force Attack. https://api.semanticscholar.org/CorpusID:63833721 (vid. pág. 2).

ArunPrakash, M., & Gokul, T. (2011). Network security-overcome password hacking through graphical password authentication. *2011 National Conference on Innovations in Emerging Technology*, 43-48. https://doi.org/10.1109/NCOIET.2011.5738831 (vid. pág. 7).

Atan, N. D. F., Rahmadewi, R., Susanto, D. A., & Jati, W. K. (2024). 3. Implementation of an identification system with facial image processing (eigenface) using matlab application. *Media Elektrik: Jurnal Kelistrikan Gagasan dan Hasil Penelitian*. https://doi.org/10.59562/metrik.v21i2.1706 (vid. pág. 1).

Aviv, A. J., Gibson, K., Mossop, E., Blaze, M., & Smith, J. M. (2010). Smudge attacks on smartphone touch screens. *4th USENIX workshop on offensive technologies (WOOT 10)* (vid. págs. 6, 31).

Ayyagiri, A., Jain, S., & Aggarwal, A. (2023). 2. Innovations in Multi-Factor Authentication: Exploring OAuth for Enhanced Security. *Innovative research thoughts*. https://doi.org/10.36676/irt.v9.i4.1461 (vid. pág. 1).

Bicakci, K. (2008). Optimal discretization for high-entropy graphical passwords. *2008 23rd International Symposium on Computer and Information Sciences*, 1-6 (vid. págs. 8, 13, 15, 17).

Birget, J.-C., Hong, D., & Memon, N. (2006). Graphical passwords based on robust discretization. *IEEE Transactions on Information Forensics and Security*, *1*(3), 395-399 (vid. págs. 8, 11).

Blonder, G. E. (1996). *Graphical Passwords* (United States Patent N.º 5559961). (Vid. págs. 7, 14).

Borrego, E. A., Navarro, P. E., & Legón, C. M. (2018). Debilidades de los métodos de discretización para contraseñas gráficas. En I. (Sociedad Cubana de Matemática y Computación (Ed.), *IV Seminario Científico Nacional de Criptografía*. (Vid. págs. 10, 11, 14, 15).

Chiasson, S., Srinivasan, J., Biddle, R., & Van Oorschot, P. (2008). Centered discretization with application to graphical passwords (full paper). *Proceedings*

of the 1st Conference on Usability, Psychology, and Security (UPSEC'08), 6 (vid. págs. 8, 11, 12).

Chiasson, S., Van Oorschot, P. C., & Biddle, R. (2007). Graphical password authentication using cued click points. *Computer Security–ESORICS 2007: 12th European Symposium On Research In Computer Security, Dresden, Germany, September 24—26, 2007. Proceedings 12*, 359-374 (vid. pág. 7).

Dhamija, R., & Perrig, A. (2000). Deja {Vu–A} User Study: Using Images for Authentication. *9th USENIX Security Symposium (USENIX Security 00)* (vid. págs. 5, 6, 30).

Heiberger, R. M., Neuwirth, E., Heiberger, R. M., & Neuwirth, E. (2009). Polynomial regression. *R Through Excel: A Spreadsheet Interface for Statistics, Data Analysis, and Graphics*, 269-284 (vid. pág. 6).

Herrera, A., Suárez, L., Legón, C. M., & Sosa, G. (2023). Comparación y combinación de dos test efectivos en la detección de contraseñas gráficas no aleatorias en Passpoints. *Revista Cubana de Ciencias Informáticas*, *17*(1) (vid. pág. 9).

Herrera, J. A., Suárez, L., & Legón, C. M. (2023a). Nuevo test para detectar contraseñas gráficas agrupadas en Passpoints. *Congreso Internacional Matemático COMPUMAT 2023* (vid. pág. 9).

Herrera, J. A., Suárez, L., & Legón, C. M. (2023b). Nuevo test para detectar contraseñas gráficas regulares en Passpoints. *Congreso Internacional Matemático COMPUMAT 2023* (vid. pág. 9).

Herrera-Macías, J. A. [J. A.], Suárez-Plasencia, L., Legón-Pérez, C. M., Sosa-Gómez, G., & Rojas, O. (2024). New test to detect clustered graphical passwords in Passpoints based on the perimeter of the convex hull. *Information*, *15*(8), 447 (vid. pág. 9).

Herrera-Macías, J. A. [Joaquín Alberto], Legón-Pérez, C. M., Suárez-Plasencia, L., Piñeiro-Díaz, L. R., Rojas, O., & Sosa-Gómez, G. (2021). Test for Detection of Weak Graphic Passwords in Passpoint Based on the Mean Distance between Points. *Symmetry*, *13*(5). https://doi.org/10.3390/sym13050777 (vid. pág. 9).

Hoover, C. C. (2015). *Narrative Passwords: Potential for Story-Based User Authentication*. University of Idaho. (Vid. pág. 7).

Imoto, S., & Konishi, S. (2000). B-spline nonparametric regression models and information criteria. *Proceedings of 2nd Int. Symp. on Frontiers of Time Series Model*, 240-241 (vid. pág. 7).

Jin, L., Yingmin, F., Liyan, H., Guoqi, R., Jing, Z., Shuangbing, L., Jianbo, W., & Bingran, S. (2019). *5. Machine room access control system and method based on Touch ID*. (Vid. pág. 1).

Kausar, N., Din, I. U., Khan, M. A., Almogren, A., & Kim, B.-S. (2022). GRA-PIN: A graphical and PIN-based hybrid authentication approach for smart devices. *Sensors*, *22*(4), 1349 (vid. págs. 7, 8, 35, 36).

Kirovski, D., Jogic, N., & Roberts, P. (2007). *Click Passwords* (inf. téc.). Microsoft Research. One Microsoft Way, Redmond, WA 98052, USA. (Vid. págs. 8, 14).

Lashkari, A. H., Farmand, S., Zakaria, D. O. B., & Saleh, D. R. (2009). Shoulder surfing attack in graphical password authentication. *arXiv preprint arXiv:0912.0951* (vid. pág. 8).

Legón, C. M., Socorro, R., Navarro, P., Rodríguez, O., & Borrego, E. (2019). Nuevo modelo probabilístico en autenticación gráfica. *Ingeniería Electrónica, Automática y Comunicaciones*, *40*(3), 92-104 (vid. págs. 9-11).

Lin, A. J., & Cheng, F. (2009). A Free Drawing Graphical Password Scheme. *Computer-Aided Design and Applications*, *6*(4), 553-561 (vid. págs. 6, 32).

Narayanan, A., & Shmatikov, V. (2005). Fast dictionary attacks on passwords using time-space tradeoff. *Proceedings of the 12th ACM Conference on Computer and Communications Security*, 364-372. https://doi.org/10.1145/1102120.1102168 (vid. pág. 2).

Nardone, M., & Scarioni, C. (2023). 5. JSON Web Token (JWT) Authentication. https://doi.org/10.1007/979-8-8688-0035-1_9 (vid. pág. 1).

Nelson, D. L., Reed, V. S., & Walling, J. R. (1976). Pictorial superiority effect. *Journal of experimental psychology: Human learning and memory*, *2*(5), 523 (vid. pág. 2).

Olade, I., Liang, H.-N., & Fleming, C. (2023). Story-based authentication for mobile devices using semantically-linked images. *International Journal of Human-Computer Studies*, *171*, 102967 (vid. págs. 7, 33).

Paivio, A. (2013). *Imagery and verbal processes*. Psychology Press. (Vid. pág. 2).

Renaud, K., & De Angeli, A. (2004). De Angeli, A.: My password is here! An investigation into visuo-spatial authentication mechanisms. Interacting with Computers 16, 1017-1041. *Interacting with Computers*, *16*, 1017-1041 (vid. pág. 8).

Rodriguez, O. (2019). *Algoritmo para la detección de claves débiles en la técnica de autenticación gráfica passpoints* [M.Sc. thesis]. Universidad de la Habana, Facultad de Matemática y Computación, Instituto de Criptografía. (Vid. págs. 8, 9).

Rodriguez, O., Legón, C. M., & Socorro, R. (2018). Seguridad y usabilidad de los esquemas y técnicas de autenticación gráfica. *Revista Cubana de Ciencias Informáticas*, *12*(Especial UCIENCIA), 13-27 (vid. pág. 9).

Rodriguez Valdés, O., Legón, C., & Socorro, R. (2019, diciembre). *Algoritmo para la detección de claves débiles en la técnica de autenticación gráfica Passpoints* [Tesis doctoral]. https://doi.org/10.13140/RG.2.2.26847.20647 (vid. pág. 9).

Salehi-Abari, A., Thorpe, J., & Van Oorschot, P. C. (2008). On purely automated attacks and click-based graphical passwords. *2008 Annual Computer Security Applications Conference (ACSAC)*, 111-120.

Shepard, R. N. (1967). Recognition memory for words, sentences, and pictures. *Journal of verbal Learning and verbal Behavior*, *6*(1), 156-163 (vid. pág. 2).

Suárez Plasencia, L., Legón, C., Socorro, R., & Rodríguez, O. (2020). Discretización de Voronoi en autenticación gráfica. *Actas del Congreso Internacional de Ciencias de la Computación*, 123-130.

Suárez-Plasencia, L., Herrera-Macías, J. A. [Joaquín Alberto], Legón-Pérez, C. M., Sosa-Gómez, G., & Rojas, O. (2022). Detection of DIAG and LINE Patterns in PassPoints Graphical Passwords Based on the Maximum Angles of Their Delaunay Triangles. *Sensors*, *22*(5). https://doi.org/10.3390/s22051987 (vid. pág. 9).

Suárez-Plasencia, L., Legón, C., Herrera, J., Socorro, R., Rojas, O., & Sosa Gómez, G. (2022). Weak PassPoint Passwords Detected by the Perimeter of Delaunay Triangles. *Security and Communication Networks*, *2022*, 1-14. https://doi.org/10.1155/2022/3624587 (vid. pág. 9).

Sun, H.-M., Chen, Y.-H., Fang, C.-C., & Chang, S.-Y. (2012). PassMap: a map based graphical-password authentication system. *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, 99-100. https://doi.org/10.1145/2414456.2414513 (vid. págs. 7, 35).

Suo, X., Zhu, Y., & Owen, G. (2005). Graphical Passwords: A Survey., 463-472. https://doi.org/10.1109/CSAC.2005.27 (vid. págs. 6, 30).

Tabrez, S., & Sai, D. J. (2017). Pass-matrix authentication a solution to shoulder surfing attacks with the assistance of graphical password authentication system. *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, 776-781. https://doi.org/10.1109/ICCONS.2017.8250568 (vid. pág. 7).

Tao, H., & Adams, C. (2008). Pass-go: A proposal to improve the usability of graphical passwords. *Int. J. Netw. Secur.*, *7*(2), 273-292 (vid. págs. 7, 34).

Tuscano, G., Tulasyan, A., Shetty, A., & Shetty, A. (2015). Graphical password authentication using Pass faces. https://api.semanticscholar.org/CorpusID:61812491 (vid. pág. 6).

WAHAB, F., KHAN, I., & SI, K. (2024). Investigating offline password attacks: A comprehensive review of rainbow table techniques and countermeasure limitations. *Romanian Journal of Information Technology and Automatic Control*, *34*(1), 81-96 (vid. pág. 2).

Wiedenbeck, S., Waters, J., Birget, J.-C., Brodskiy, A., & Memon, N. (2005). PassPoints: Design and longitudinal evaluation of a graphical password system. *International journal of human-computer studies*, *63*(1-2), 102-127 (vid. págs. 2, 7, 8, 33).

Yang, G.-C. (2017). PassPositions: A secure and user-friendly graphical password scheme. *2017 4th International Conference on Computer Applications and In-*

formation Processing Technology (CAIPT), 1-5. https://doi.org/10.1109/CAIPT.2017.8320723 (vid. págs. 7, 33).

Zhu, B. B., Wei, D., Yang, M., & Yan, J. (2013). Security implications of password discretization for click-based graphical passwords. *Proceedings of the 22nd international conference on World Wide Web*, 1581-1591 (vid. págs. 8, 11, 12).

Zimmermann, V., & Gerber, N. (2020). The password is dead, long live the password - A laboratory study on user perceptions of authentication schemes. *International Journal of Human Computer Studies*, *133*, 26-44 (vid. pág. 9).

Anexos

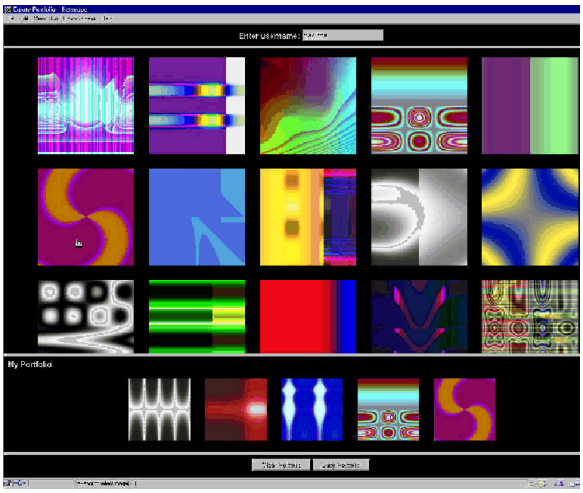


Figura 3.1: Selección de portafolio. Fuente: 12

Figura 3.2: Sistema PassFaces. Fuente: 42

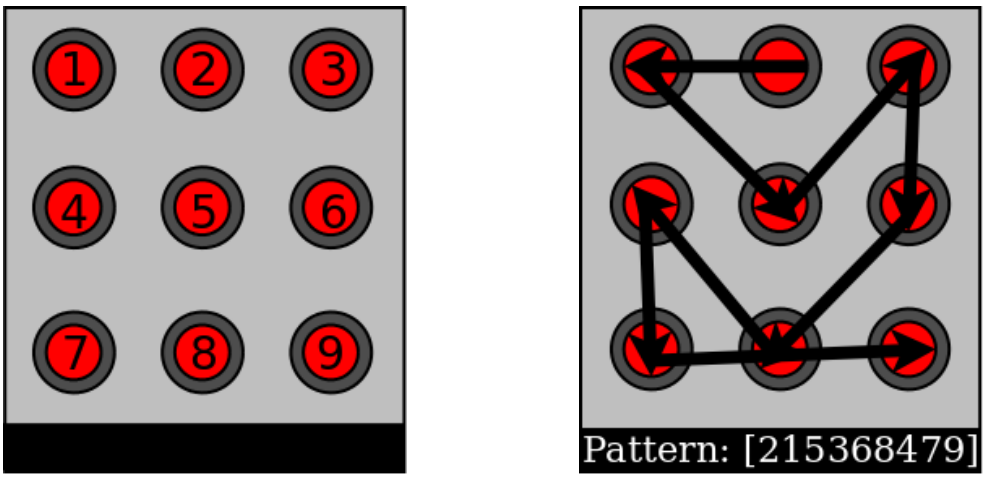


Figura 3.3: Funcionamiento de un patrón de desbloqueo, Fuente: 4

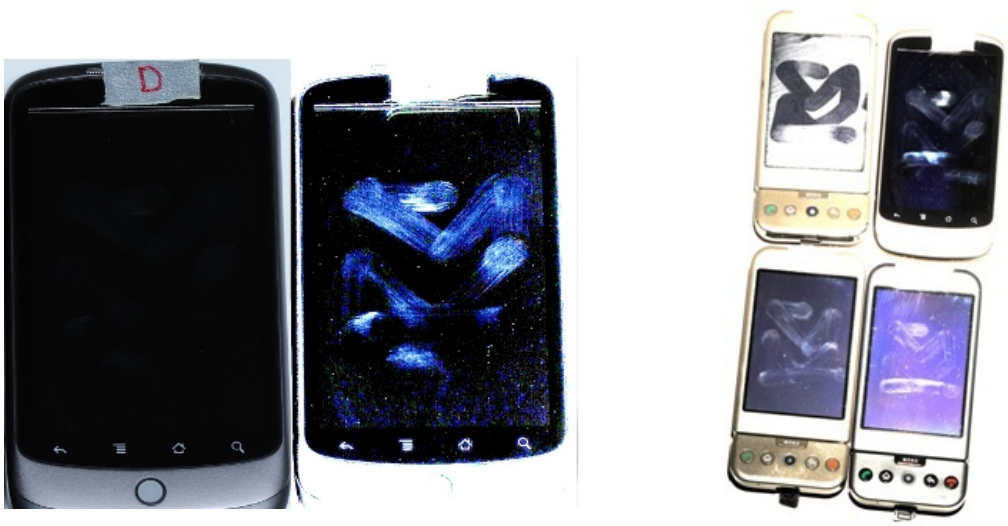


Figura 3.4: Grasa en pantalla usada en los ataques de smudge, Fuente: 4

Figura 3.5: Dibujado 6 veces usando el mouse, Fuente: 26

Figura 3.6: Dibujado 6 veces usando stylus, Fuente: 26

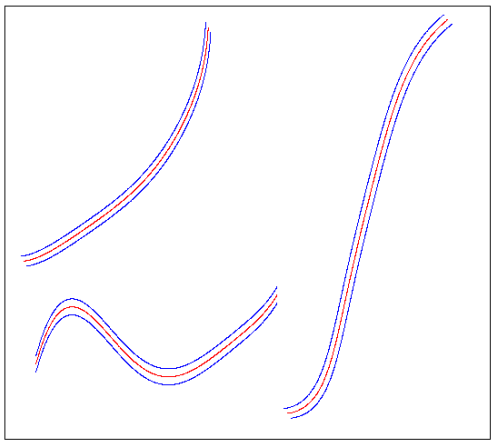


Figura 3.7: Valores predichos e intervalos de predicción generados por el modelo de regresión polinomial, Fuente: 26

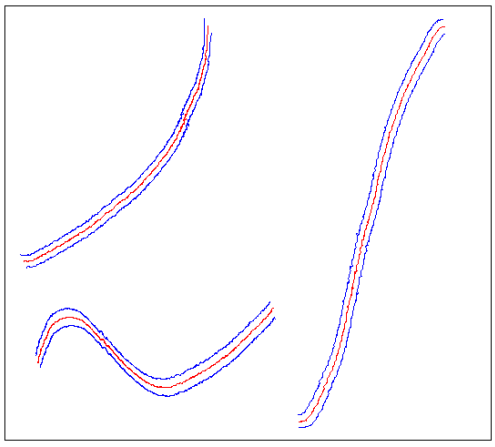


Figura 3.8: Valores predichos e intervalos de predicción generados por el modelo de regresión B-spline, Fuente: 26

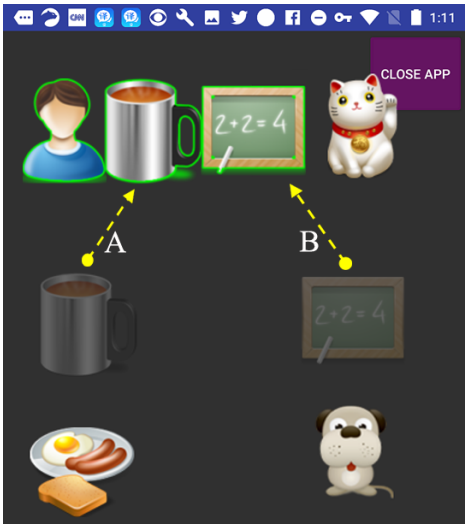


Figura 3.9: Funcionamiento de Semantic Lock, Fuente: 30

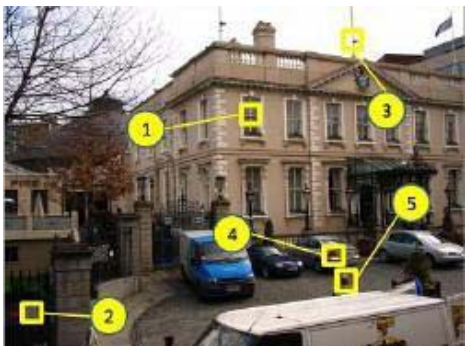


Figura 3.10: Ejemplo de Passpoints, Fuente: 47

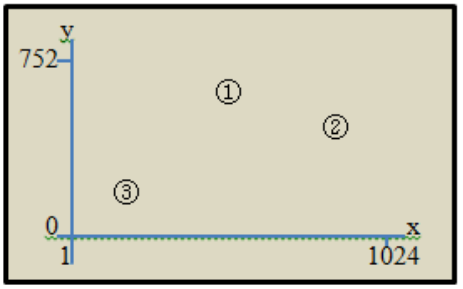


Figura 3.11: Cálculo de hash de un punto Passpositions, Fuente: 48

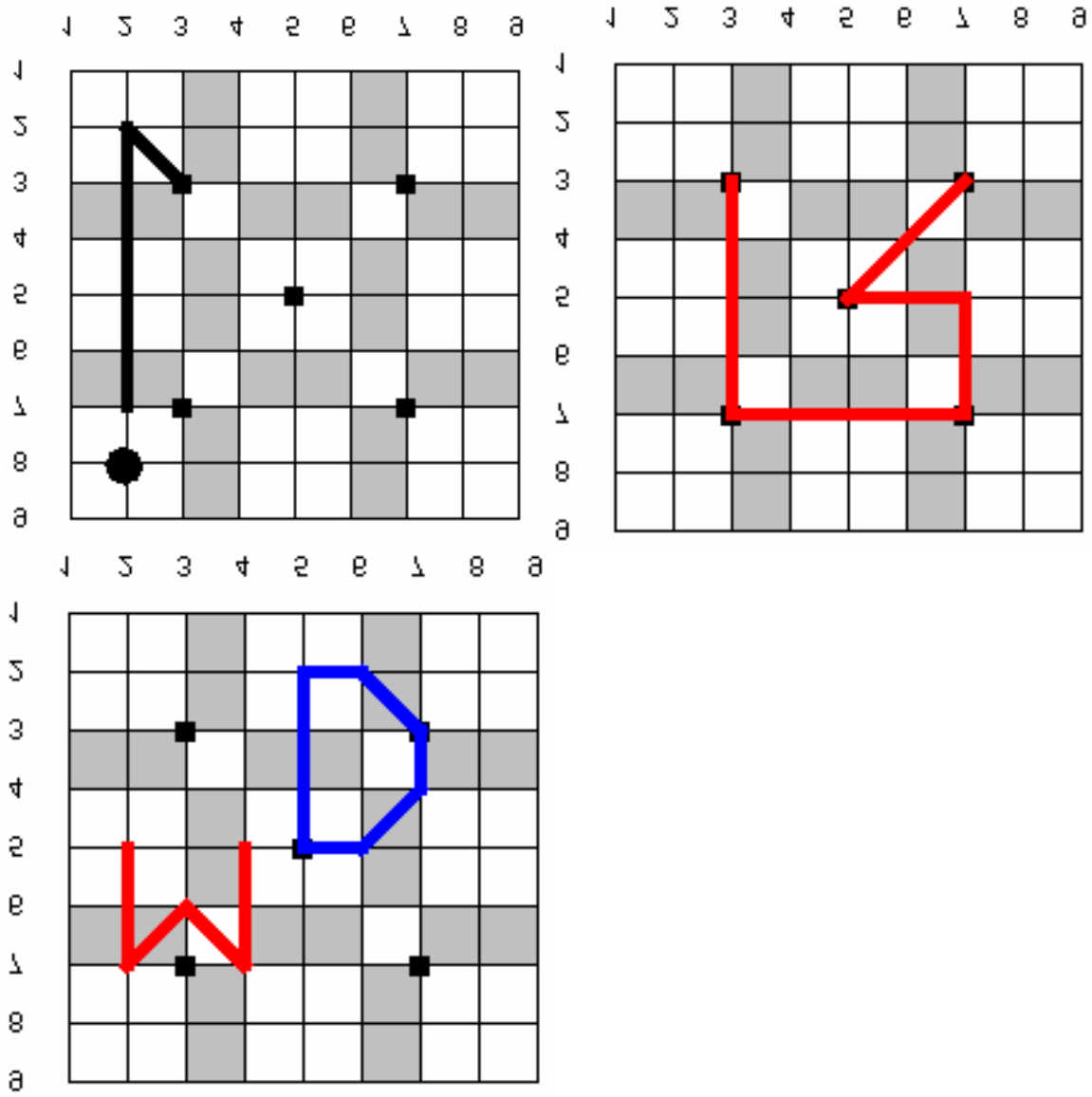


Figura 3.12: Ejemplos de contraseñas usando Pass Go, Fuente: 44

Figura 3.13: Información del mapa en diferentes lugares y niveles de zoom, Fuente: 41

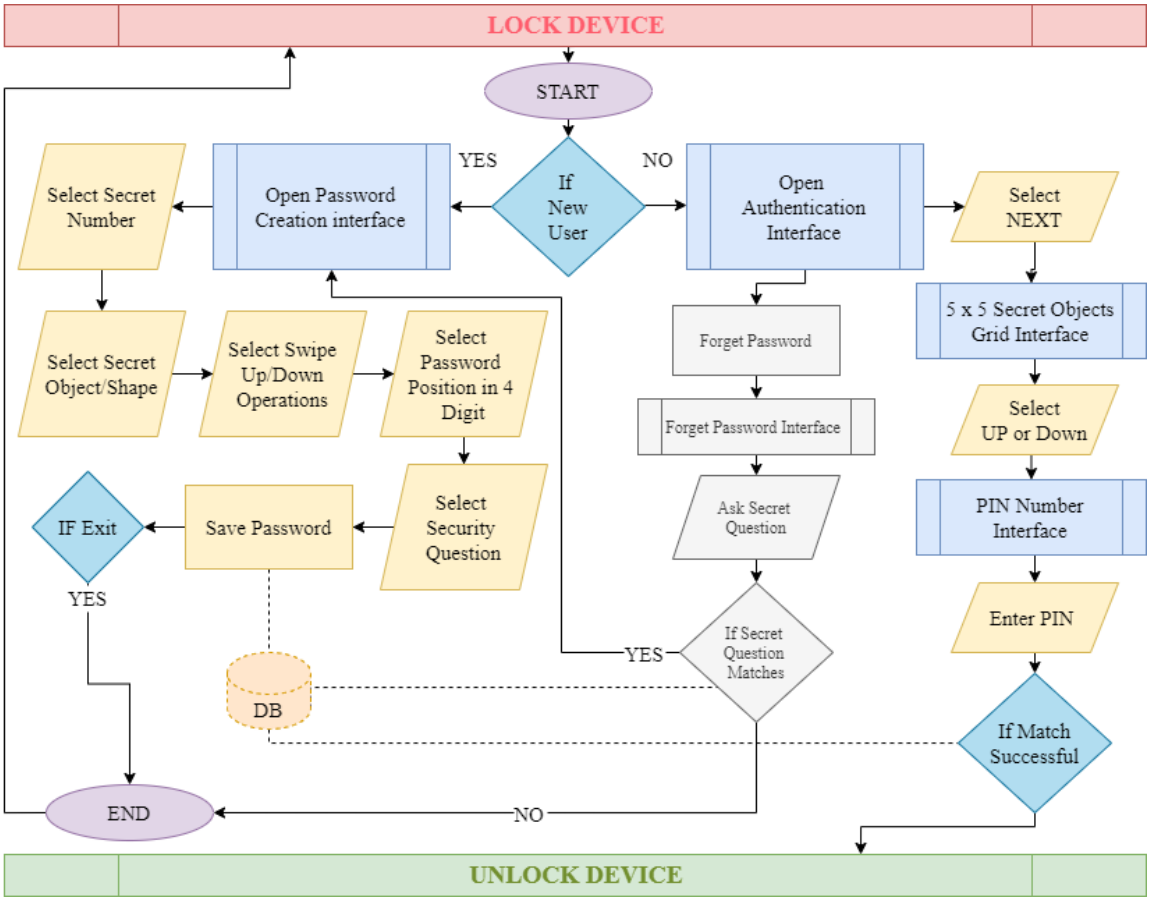


Figura 3.14: Proceso de autenticación *Gra-Pin*, Fuente: 22

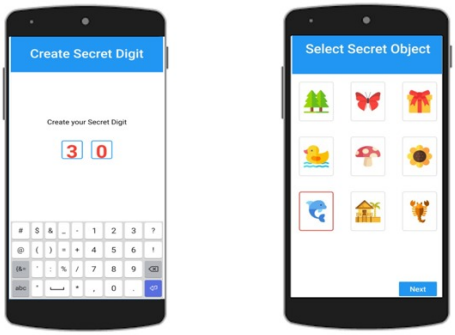


Figura 3.15: Selección del número e imagen secretos, Fuente: 22

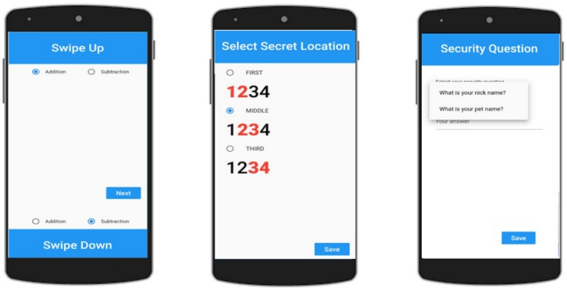


Figura 3.16: Selección de la operación y posición en el Pin de la clave, Fuente: 22

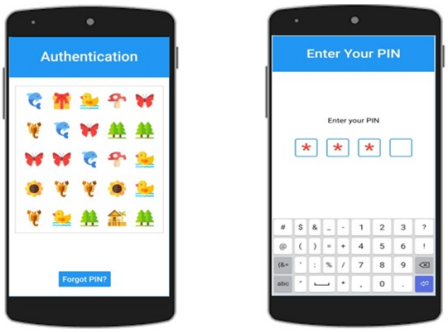


Figura 3.17: Pantalla de autenticación, Fuente: 22

Figura 3.18: Pantallas de autenticación de *Gra Pin*, Fuente: 22