Universitat Oberta de Catalunya uoc.edu

Autenticación de doble factor mediante OTPs

Javier Guerrero Ramírez
Master Inter-universitario en Seguridad de las Tecnologías de la Información y las Comunicaciones (MISTIC)
Seguridad de autenticación y autorización

Henric Hernández Jiménez
Víctor García Font

Junio/2019

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Autenticación de doble factor mediante OTPs*
Nombre del autor:	*Javier Guerrero Ramírez*
Nombre del consultor/a:	*Henric Hernández Jiménez*
Nombre del PRA:	*Víctor García Font*
Fecha de entrega (mm/aaaa):	06/2019
Titulación::	*Master Inter-universitario en Seguridad de las Tecnologías de la Información y las Comunicaciones*
Área del Trabajo Final:	*Sistemas de autenticación y autorización*
Idioma del trabajo:	*Español*
Palabras clave	*Autenticación, factor, OTP*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados i conclusiones del trabajo.*

El gran avance de las tecnologías de la información y la comunicación en las aplicaciones ha llevado a las personas al gran reto de mantener en secreto sus datos personales y un constante desafío por mantener seguro una forma de acceder a ellos.

El proyecto aborda en un principio el concepto de la seguridad: desde cuales fueron las primeras percepciones que tuvo el ser humano de querer mantener de forma segura las cosas y buscar su propia seguridad a los principales conceptos que engloban hoy en día un sistema de autenticación de cualquier aplicación. Tras ello ha introducido al tema principal de este proyecto, la autenticación, y más en concreto la que se realiza a través de un sistema de doble factor utilizando contraseñas de un único uso.

En un segundo bloque se han analizado los dos principales algoritmos definidos por la IETF Internet Engineering Task Force) para la generación de contraseñas de un solo uso basándose en el sistema de encriptación HMAC y basado en el tiempo.

Finalmente se aborda un proyecto local en el que se pone de manifiesto todo lo anterior para obtener una aplicación con un sistema de autenticación y autorización que requiera el empleo de un segundo factor mediante contraseña de un solo uso.

Abstract (in English, 250 words or less):

The great advance of information and communication technologies in applications has led people to the great challenge of keeping their personal data secret and a constant challenge to keep a form of access to them safe.

The project initially addresses the concept of security: from what were the first perceptions that the human being had of wanting to keep things secure and seek their own security to the main concepts that today encompass an authentication system of any application After this has introduced the main topic of this project, authentication, and more specifically that is done through a double factor system using passwords for a single use.

In a second block we analyzed the two main algorithms defined by the IETF Internet Engineering Task Force for the generation of one-time passwords based on the HMAC encryption system and based on time.

Finally, a local project is addressed in which all of the above is highlighted in order to obtain an application with an authentication and authorization system that requires the use of a second factor through a one-time password.

Índice

1. Introducción 1
1.1 Contexto y justificación del Trabajo 1
1.2 Objetivos del Trabajo 2
1.3 Enfoque y método seguido 2
1.4 Planificación del Trabajo 3
1.5 Breve sumario de productos obtenidos 3
1.6 Breve descripción de los otros capítulos de la memoria 3
2. Estado del arte 4
2.1 Seguridad 4
2.2 Identificación 5
2.3 Autenticación 5
2.4 Autorización 7
2.5 Criptografía 7
2.6 Problemática en la autenticación convencional 8
2.7 Autenticación de doble factor 8
2.8 Códigos OTP 9
3. LA AUTENTICACIÓN Y OTROS FACTORES 10
3.1 La autenticación de un único factor 10
3.2 Errores comunes al crear una contraseña 11
3.3 Contraseña robusta 12
3.4 Ataques a las contraseñas 13
3.5 La autenticación de múltiples factores 14
3.5.1 Categorías de factores 16
3.5.2 No es algo nuevo, se lleva utilizando años 16
3.6 Autenticación de doble factor 17
3.6.1 OTPs 17
3.6.2 Envío de código PIN de un solo uso a dispositivo móvil por SMS o correo electrónico 19
3.6.3 Tokens 20
3.6.4 Uso de la biometría 20
3.6.5 Seguridad en los sistemas biométricos 21
3.6.6 Otros puntos débiles de la autenticación de doble factor 22
4. ANÁLISIS DE PROTOCOLOS 23
4.1 Sistema One-Time Password 23
4.1.1 Generación de un One-Time Password o contraseña de un solo uso 24
4.1.2 Verificación de una contraseña de un solo uso 24
4.2 HOTP - HMAC based One-Time Password 24
4.2.1 Requerimientos del algoritmo 25
4.2.2 El algoritmo HTOP 26
4.2.3 Ejemplo de obtención de un valor HOTP 28
4.2.4 Consideraciones de seguridad 29
4.2.5 Validación de valores HOTP 29
4.2.6 La resincronización del contador incremental 29
4.3 TOTP - TIME BASED ONE-TIME PASSWORD 30
4.3.1 Requerimientos del algoritmo 30

4.3.2 El algoritmo TOTP 31
4.3.4 Consideraciones de seguridad 31
4.3.5 Validación y el tamaño del paso de tiempo 32
4.3.6 Resincronización 32
5.1 Google Authenticator 34
5.2 Authy 35
5.3 Latch 35
6. IMPLEMENTACIÓN DEL CONCEPTO 37
6.1 Servidor NodeJS 38
6.2 Librerías principales 38
6.3 Base de datos MongoDB. Modelos 39
6.3.1 Modelos de datos 39
6.3.2 Modelo <users> 39
6.3.3 Modelo <otps> 40
6.4 Algoritmo de Python para la generación de códigos OTP 40
6.4.1 Diagrama de flujo del programa 41
6.4.2 Función Main() 41
6.4.3 Función GenerarTOTP() 43
6.4.4 Función GenerarHOTP() 44
6.4.5 Twilio, plataforma para envío de SMS 44
6.5 Vistas 46
6.5.1 Login 46
6.5.2 Register 46
6.5.3 Profile 47
6.6 Rutas 48
6.6.1 Raíz o / 48
6.6.2 /login 48
6.6.3 /register 48
6.6.4 /generateotp 48
6.6.5 /profile 48
6.6.6 /logout 48
7. Conclusiones 49
4. Glosario 50
5. Bibliografía 51
6. Anexos 55
ANEXO 1 – DIAGRAMA DE GANT PLANIFICACIÓN ESTIMADA DEL TRABAJO 55

Lista de figuras

Ilustración 1 - Mecanismo de autenticación de un único factor. 10
Ilustración 2 - Herramienta de generación de contraseñas ofrecida por LastPass 14
Ilustración 3 - Mecanismo de autenticación de doble factor. 15
Ilustración 4 - Dispositivo USB generador de tokens RSA SecurID 18
Ilustración 5 - Tarjeta de coordenadas con OTPs 19
Ilustración 6 - PIN-pad utilizado como terminal de pago mediante tarjetas 26
Ilustración 7 - Orden natural de una cadena hexadecimal mediante big-endian [4] 27
Ilustración 8 - Proceso de generación de un código OTP mediante HOTP 28
Ilustración 9 - Bytes que componen el resúmen devuelto por una función HMAC-SHA-1 28
Ilustración 10 - Aplicación móvil Google Authenticator ofreciendo un TOTP para distintos servicios 34
Ilustración 11 - Códigos de un sólo uso para distintos servicios online con la aplicación móvil Authy 35
Ilustración 12 - Opción de utilizar OTP en Latch para inicio de sesión 36
Ilustración 13 – Esquema general del servidor levantado en NodeJS 37
Ilustración 14 – Diagrama de flujo de Python para la generación de OTPs 41
Ilustración 15 – Código de la función main de Python – Parte 1 42
Ilustración 16 – Código de la función main de Python – Parte 2 43
Ilustración 17 – Código de la función GenerarTOTP de Python 43
Ilustración 18 - Código de la función GenerarHOTP de Python 44
Ilustración 19 – Panel de control de la cuenta de Twilio 45
Ilustración 20 – Vista de la pantalla de login 46
Ilustración 21 – Vista de la pantalla register 47
Ilustración 22 – Vista de la pantalla de perfil de usuario 47

1. Introducción

1.1 Contexto y justificación del Trabajo

No cabe duda que hoy en día se vive en un mundo tecnológico, relleno de información que muchas veces se ha de proteger (debido a que puede ser sensible, personal, etc.) y lo primero que se puede venir a la cabeza es usar una contraseña para ello. Error.

Con el paso del tiempo ha quedado demostrado que la protección mediante contraseñas no es todo lo eficaz y seguro que se creía.

Por un lado, el usuario ha de pensar una contraseña, normalmente que le sea familiar o fácil de recordar, sin embargo no sería conveniente proteger todo con la misma contraseña, ya que si un intruso se hace con ella tendría acceso a todo el contenido que el usuario no quería que quedase expuesto. Igualmente, pensar que una contraseña de longitud elevada hará más segura la protección es otro error común, ya que por el mismo motivo anterior un intruso podría tener acceso a los datos protegidos.

Por otro lado, lo ideal sería poder demostrar que el usuario es quien dice ser realmente, sin depender de una palabra o combinación de letras, números y/o caracteres, y es aquí donde entra en juego una segunda forma de demostrarlo, un segundo factor de autenticación, e incluso pueden existir varios factores juntos (autenticación multifactor).

El NIST define [20] tres factores básicos en un paradigma de autenticación estándar:

- "*Algo que el usuario sabe*", como puede ser una contraseña.
- "*Algo que el usuario tiene*", como puede ser una clave criptográfica o un identificador.
- "*Algo que el usuario es*", como puede ser una huella dactilar, el iris u otro dato biométrico.

Hoy en día son muchos los servicios que integran y ofrecen al usuario la posibilidad de utilizar una autenticación de doble factor, sin embargo también ha quedado demostrado como otros factores han podido ser suplantados como por ejemplo huellas dactilares impresas. De lo que se trata en este trabajo es de utilizar un sistema de doble autenticación en forma de clave criptográfica y que no solamente el usuario tenga, sino que sólo pueda utilizarlo una sola vez. Así si un intruso consigue éste, aun conociendo la contraseña, no podrá tener acceso al contenido protegido ya que esta clave ya no volverá a ser válida. Esto es conocido como OTP (*One Time Password*) o contraseña de un solo uso.

1.2 Objetivos del Trabajo

El objetivo principal del trabajo es realizar un estudio preliminar o análisis de una capa de autenticación basada en OTP's, para finalmente intentar realizar una prueba de campo que ponga en práctica lo tratado.

En el estudio o análisis se tratarán diversos aspectos, entre ellos:

- La elaboración de un estado del arte para conocer en qué estado se encuentra esta tecnología actualmente.
- La realización de un compendio o resumen breve del tema a tratar en este trabajo.
- Conocer la importancia de la autenticación en un medio digital como es internet.
- Mostrar distintos métodos o factores alternativos que pueden complementar a la autenticación convencional por contraseña.
- La implementación de un estudio comparativo a nivel de mercado actual que utilice esta tecnología.
- Profundizar en materia para conocer los tipos de protocolos, las funciones criptográficas, tipos de códigos de un solo uso, etc. que se utilizan.
- Formas de generar de contraseñas de un solo uso.
- Una vez cubierto el análisis, se procederá a realizar una prueba de campo en la que se apliquen los conceptos anteriormente tratados y que pueda servir de ejemplo a lo que este trabajo quiere desarrollar.

1.3 Enfoque y método seguido

La estrategia para llevar a cabo el trabajo será la búsqueda de información en distintas fuentes (publicaciones, libros, internet, etc.) hasta llegar a una comprensión a fondo de los aspectos más relevantes del tema para poder implementarlo en una prueba de campo final.
Se tratará de adaptar un producto existente o una reinterpretación de éste ya que ello permitirá conocer a fondo lo que se trata en este trabajo y lo que se pretende es conocer el funcionamiento de estas capas de autenticación, en concreto las que atañen a este trabajo que son las OTPs.

1.4 Planificación del Trabajo

Teniendo en cuenta que se disponen de 3 meses (aproximadamente 12 semanas) para la elaboración del trabajo, se puede estimar que la duración o tiempo de dedicación de las siguientes tareas sean:

- Estado del arte: 1 semana
- Compendio: 1 semana
- Estudio comparativo: 2 semanas
- Análisis de protocolos: 3 semanas
- Propuesta de arquitectura: 1 semana
- Implementación del concepto: 5 semanas
- Pruebas: 1 semana
- Redacción de la memoria: 2 semanas

Hay que tener en cuenta que algunas de estas fases pueden realizarse en paralelo, por lo cual el conjunto del proyecto no corresponde a la suma de esas cantidades estimadas.

A modo de ejemplo puede elaborarse un diagrama de Gantt que haga una idea del tiempo estimado a cada una de las partes que queda representado en el Anexo 1.

1.5 Breve sumario de productos obtenidos

Se ha obtenido una visión global de los sistemas de autenticación de doble factor y la puesta en marcha de un sistema que lo aplica a modo de prueba.

1.6 Breve descripción de los otros capítulos de la memoria

En los primeros capítulos se expondrá un estado del arte y los principales conceptos relacionados con la seguridad, hasta llegar a los sistemas de autenticación actuales.

A la mitad del proyecto se analizarán los protocolos establecidos por la IETF para la generación de códigos de un solo uso.

Finalmente se pondrá en marcha un servidor local con una aplicación web/móvil que se desarrollará siguiendo el contenido de este proyecto para que permita una autorización de doble factor mediante OTP.

2. Estado del arte

En la era digital actual, donde todos los dispositivos se encuentran interconectados y se guarda gran cantidad de información personal y confidencial en ellos o en servicios de la Red, es imprescindible contar con contraseñas robustas que impidan a cualquiera acceder a dichos datos. Quizás no sea muy conocido, pero [11] desde hace varios años, el primer jueves de mayo se celebra el Día Mundial de las Contraseñas, para concienciar al usuario sobre su importancia, ya que son la llave de entrada a la información personal.

En los años 60, Fernando Corbató, conocido también como el padre de la contraseña moderna, ideó la contraseña para que cada usuario accediera a los archivos que le correspondían, pero su existencia data de la época de los romanos.

En este apartado se va a realizar una descripción de los distintos conceptos relacionados con la autenticación, recorriendo los primeros signos que mostró el ser humano de querer proteger cosas a la problemática actual de no recaer todo el peso de la seguridad en una única contraseña, entrando así en juego distintos factores que añadirán una capa de seguridad extra que permitirán a los usuarios demostrar que son ellos y que alguien sin autorización expresa no pueda acceder con esta funcionalidad.

2.1 Seguridad

La seguridad es un concepto controvertido ya que no existe un consenso generalizado sobre su significado, dependiendo de las personas, su cultura y el contexto en el que se aplique. En un tiempo remoto el concepto de seguridad estaba relacionado con el concepto de confianza, mientras que actualmente se relaciona mejor con la responsabilidad de proteger [1].

Los primeras evidencias de seguridad se pueden asociar a la que aplicaban los humanos al utilizar por ejemplo llaves para mantener a buen recaudo sus pertenencias, o utilizar criptografía para cifrar mensajes escritos (la más simple y usada [2] fue el cifrado por desplazamiento de César), o incluso el utilizar armas (como puede ser una espada) para salvaguardar sus vidas.

El cifrado de César corresponde a sustituir cada letra de un texto original con la letra que corresponde un número fijo de posiciones más adelante en el alfabeto y se dice [2] que Julio César utilizaba un desplazamiento de número tres para proteger sus mensajes importantes de contenido militar. Estos primitivos signos de preocupación por la seguridad y la confidencialidad fueron evolucionando con el tiempo a la vez que lo hizo análogamente la tecnología. Se podría destacar la máquina Enigma [3] , que consistía en una máquina de rotores que permitía usarla tanto para

cifrar mensajes como para descifrar mensajes, aunque finalmente quedó demostrado que presentaba vulnerabilidades (como un análisis de frecuencia de letras) y pudieron ser descifrados mensajes escritos con ella.

La evolución de la seguridad también llegó [4] al ámbito de las empresas en los que también está implícito el concepto de seguridad:

- En la década de los 70, la seguridad se centraba en garantizar un buen uso de la información, pero tenía sus inconvenientes como poca información de los riesgos, falta de copias de seguridad o medidas físicas inexistentes, a la vez que se iba quedando obsoleta por el gran avance de la tecnología [4].

- En la década de los 80 aparecieron los primeros virus, y con ellos la aparición y comercialización de antivirus debido a la inexistente seguridad en los ordenadores.

- En la década de los 90 empiezan a ocurrir los primeros ataques por internet y la preocupación por la protección de la red.

- En los años 2000 comienzan los ataques a corporaciones y la noción de seguridad de los datos.

- En los años 2010 se toma conciencia de la seguridad en dispositivos móviles y el uso de cifrado de información a nivel corporativo y personal.

2.2 Identificación

La identificación es el momento en el que el usuario se da a conocer al sistema y corresponde al proceso de distinguir un individuo de otro. Tiene la capacidad de identificar de forma exclusiva a un usuario de un sistema o aplicación que se esté ejecutando en un sistema [8]. Muchas veces este término se confunde con la autenticación y la diferencia es que esta última es cuando el sistema valida que el individuo es realmente quien se está identificando.

2.3 Autenticación

La autenticación es otro concepto importante relacionado con la seguridad. Hoy en días las personas se autentican múltiples veces cada día. Por ejemplo, cuando abren una puerta de un edificio se están autenticando demostrando la posesión de la llave, o cuando compran algo con una tarjeta de crédito se autentican al introducir el código PIN asociado a la tarjeta.

Se define [5] esta como el acto o proceso de confirmar que algo o alguien es quien dice ser.

Entre las características que ha de poseer un sistema de autenticación están:

- Que sea fiable con una probabilidad muy elevada.
- Que soporte ciertos tipos de ataques.
- Que su precio no sea superior al valor de los que se quiere proteger.
- Que lo acepten los usuarios, ya que ellos son quienes van a utilizarlo.
- Que tenga una respuesta inmediata.

En un acto de autenticación se diferencian dos partes, el que se identifica o probador, y el que verifica la identidad o verificador y para ello se distinguen tres categorías:

- Autenticación basada en algo conocido por el probador: Como por ejemplo una contraseña, un "*passphrase*" (PGP), un Número de Identificación Personal (PIN), un desafío-respuesta o una pregunta de seguridad.
- Autenticación basada en algo poseído o de propiedad por el probador: una tarjeta de identidad o tarjeta inteligente ("*smartcard*"), un dispositivo USB tipo "*epass token*", una tarjeta de coordenadas, un dongle criptográfico.
- Autenticación basada en una característica existente o inherente al probador: verificación por voz, patrones oculares, una secuencia de ADN, huellas dactilares u otro identificador biométrico.

El mecanismo de autenticación [5] es un proceso muy sencillo en el cual:

- El probador solicita acceso a un sistema.
- El verificador solicita al usuario que se autentique
- El probador aporta las credenciales que le identifican y permiten verificar la autenticidad de la identificación.
- El verificador valida según sus reglas si las credenciales son suficientes para dar acceso al usuario o no.

2.4 Autorización

Otras veces el concepto de autenticación es confundido con el de autorización, sin embargo este corresponde a un paso posterior a la autenticación, ya que se encarga de proteger los recursos permitiendo que solo sean usados por aquellos individuos a los que se les ha concedido autorización para ello. Es decir, mientras que la autenticación verifica que la persona es quien dice ser, la autorización comprueba si esa persona tiene permiso de acceso a lo que intenta acceder (ya sea un dato, una funcionalidad, un servicio, etc).

Uno de los mecanismos de autorización más utilizados es el Control de Acceso Basado en Roles (RBAC - *Role-Based Access Control*) el cual es [13] un mecanismo de control de acceso de política neutral definido en base a unos roles y unos privilegios. Existen otros mecanismos de control de acceso como los basados en atributos, también los basados en riesgos, basados en localización, basados en listas o basados en organización.

2.5 Criptografía

La criptografía es otro concepto ligado a la autenticación. Para que un sistema de comunicación se considere seguro, los requisitos [7] que debe proporcionar son:

- La autenticación de cada parte, es decir, que cada individuo que participa en la comunicación sean quienes dicen ser.
- La confidencialidad de los datos, es decir, que solo los individuos que toman parte pueda conocer el mensaje original.
- La integridad de los datos, es decir, que no se pueda modificar el contenido original encriptado.
- El no repudio, es decir, que los individuos no puedan negar haber sido ellos

Para ello se utiliza la criptografía.

La criptografía es el ámbito de la criptología que se ocupa de las técnicas de cifrado o al diseño de cifras. Una cifra, o también llamado criptosistema, es [6] un método secreto de escritura mediante el cual un texto en claro se transforma en un texto cifrado (a veces llamado criptograma). El cifrado es el proceso de transformar un texto en claro en texto cifrado se denominado cifrado, mientras que el proceso inverso se denomina descifrado. Ambos procesos son controlados por una o más claves criptográficas.

Existen [6] dos criptosistemas elementales:

- Las cifra de transposición, que reordenan los bits o caracteres del texto claro. en este caso la clave criptográfica de la cifra es el criterio de reordenación utilizado.

- La cifra de sustitución, que cambia los bits, caracteres o bloques de estos por sustitutos. En este caso la clave de la cifra es el criterio de sustitución utilizado.

Por otro lado, según la resistencia que tengan a ataques, las cifras pueden ser rompibles o débiles; computacionalmente seguras o fuertes; o incondicionalmente seguras.

2.6 Problemática en la autenticación convencional

En el paso de los últimos años, ha sido muy común conocer a alguien cuya cuenta online haya sido comprometida, empresas que han tenido una brecha de seguridad y se han producido filtraciones de datos que dan acceso a infinitas cuentas online de sus diversos servicios. Por ello ha quedado demostrado que la autenticación que solo utiliza un nombre de usuario y una contraseña genera tantos problemas de seguridad que los investigadores han tenido que buscar constantemente nuevas formas de que los usuarios puedan autenticarse de una forma rápida y segura. Es por ello por lo que una acción cada vez más popular es que los servicios ofrezcan a sus usuarios distintos factores de autenticación.

2.7 Autenticación de doble factor

Debido a que la autenticación de un único factor ha quedado demostrado que no es un método infalible debido a diversos motivos como contraseñas no lo suficientemente robustas, contraseñas repetidas, ataques que comprueban millones de combinaciones en segundos, ataques de suplantación de identidad, ataques de ingeniería social, mediante infección por malware, etc. Esto hace que autenticarse en un servicio suplantando la identidad de un usuario y tener acceso a los datos protegidos haya sido una actividad fácil para muchos ciberdelincuentes.

Por ello la autenticación de dos factores se ha convertido en el estándar de seguridad dentro del mundo del internet. Es también conocida [9] por distintos nombres, como “autenticación en dos fases”, “autenticación de dos factores”, “verificación en dos pasos” o la forma abreviada “2FA” de “*Two-Factor authentication*”.

Entre los factores complementarios más comunes que se pueden utilizar en la autenticación de doble factor están los códigos PIN, las huellas dactilares, reconocimiento facial o las tarjetas de coordenadas o los OTPs entre otros muchos.

2.8 Códigos OTP

Este concepto viene del anglicismo "*One-Time Password*", lo que viene a significar una contraseña válida para una autenticación, sesión de inicio o transacción. Esta una contraseña que se genera dinámicamente y que como su nombre indica sólo puede ser utilizada una vez, a veces durante un periodo corto de tiempo (minutos o incluso hasta segundos) desde que haya sido generado.

La adopción de las contraseñas de un sólo uso puede ofrecer [10] una alternativa más segura o incluso complementar a una contraseña estática en un proceso de autenticación multifactor. Su potencial implica que si un atacante llegase a obtener esta contraseña dinámica, no sería capaz de volverla a utilizar de forma válida en un segundo intento, quedando así el usuario protegido si sus credenciales estáticas hubieran sido comprometidas por algún mecanismo o ataque.

3. LA AUTENTICACIÓN Y OTROS FACTORES

El sistema de autenticación basado en usuario y contraseña sigue siendo el más extendido para acceder a distintos servicios online. Ello no significa que sea el más seguro ya que la mayoría de usuarios optan por contraseñas débiles, ya sea por la facilidad de recordarlas o por el desconocimiento acerca de lo fácil que resulta para un ciberdelincuente obtenerlas. En este apartado se va a analizar la autenticación empezando desde el punto de vista de un único factor hasta el uso de distintos factores que pueden complementarse para verificar que el usuario que quiere acceder a un recurso protegido es quien realmente dice ser.

3.1 La autenticación de un único factor

La utilización de contraseñas es válida para autentificar al usuario frente al proceso de verificación de identidad de cualquier servicio que lo requiera. De esta forma, se asegura que el usuario es realmente quien dice ser y no un impostor.

Un mecanismo de autenticación que solamente utiliza unas credenciales basadas en un identificador de usuario y su contraseña de acceso es lo que se denomina como autenticación de un único factor o las siglas SFA de "*Single Factor Authentication*". En la Ilustración 1 se muestra de forma gráfica un mecanismo de autenticación de un único factor.

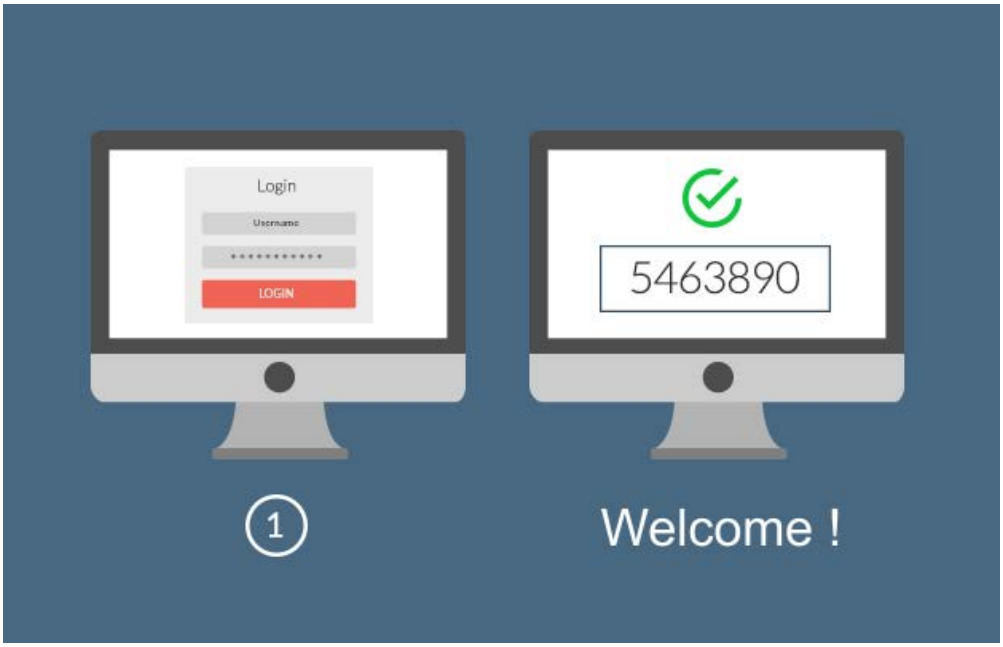


Ilustración 1 - Mecanismo de autenticación de un único factor.

Según la Real Academia Española, una contraseña es [15] una seña secreta que permite el acceso a algo, a alguien o a un grupo de personas antes inaccesibles. Esta seña podrá ser una combinación formada por caracteres alfanuméricos y símbolos especiales. Idealmente la contraseña debe ser algo que nadie pueda adivinar.

3.2 Errores comunes al crear una contraseña

El problema reside en la cantidad de claves y combinaciones que los usuarios han de recordar ya que existen una infinidad de servicios online que requieren de un método de autenticación, es por ello por lo que finalmente los usuarios recurren a utilizar contraseñas fáciles de recordar, que a su vez son precisamente las más vulnerables. Aunque parezca tediosa la tarea de crear una contraseña robusta, son peores los riesgos a los que se enfrenta el usuario de no dedicar un tiempo extra en ello.

La compañía de seguridad SplashData publicó [14] como todos los años la lista de las contraseñas más comúnmente utilizadas en la web. A continuación se listan las 10 primeras que copan el ranking, siendo la primera la más utilizada y ello da una idea de cómo vulnerable puede llegar a ser un servicio de un único factor. Además se compara con su posición respecto a las que el mismo medio publicó en 2017.

c	Contraseña 2018	Contraseña 2017
1	123456	No cambia
2	password	No cambia
3	123456789	Sube 3 posiciones
4	12345678	Baja 1 posición
5	12345	No cambia
6	111111	Nueva
7	1234567	Sube 1 posición
8	sunshine	Nueva
9	qwerty	Baja 5 posiciones
10	iloveyou	No cambia

Tabla 1 - Las 10 peores contraseñas más utilizadas en 2018

Otro error muy común es anotar las contraseñas en un cuaderno para no tener que recordarlas. Un estudio de Kaspersky indica que uno de cada cinco españoles anota sus contraseñas en un cuaderno. Esto no es una buena práctica ya que cualquiera podría verlo y acceder a los servicios.

A modo de resumen, los errores más comunes [16] a la hora de crear una contraseña son:

- Utilizar contraseñas débiles o fáciles.

- Utilizar la misma contraseña en distintos servicios online.

- Memorizar contraseñas en función del teclado (como “12345” o “*qwerty*”).

- Anotar la contraseña en sitios visibles, como en una mesa de trabajo.

- Utilizar expresiones hechas.

- Revelarlas a familiares, amistades o entorno de trabajo

3.3 Contraseña robusta

Una vez visto que la sencillez de una contraseña es un error muy común, por internet pueden encontrarse algunas reglas básicas para crear contraseñas robustas. Según Panda Security [18], algunas de estas reglas pueden ser:

- Obtener la contraseña a partir de una frase. Escoger una frase que tenga mucho significado para la persona y para nadie más. Mejor si tiene alguna mayúscula y algún número o símbolo. Después coger el comienzo de cada palabra. Por ejemplo, la frase “El bar de Juan pone tapas grandes a 3 euros” quedaría como “EbdJptga3€”. En el caso de no tener una frase, se puede elegir el título de una canción favorita.

- Combinar dos palabras. Para ello hay que escoger dos palabras que tengan significado sólo para la persona y construir la contraseña combinando cada letra de una con la otra. Por ejemplo para las palabras “Bigotes” y “Mechero” la contraseña quedaría “BMiegcohteerso”.

- Convertir las vocales en números. Para ello hay que ir intercambiando las vocales partiendo de que la vocal “a” es el número “1” y la vocal “u” correspondería al número “5”. Este método es muy conocido ya por los ciberdelincuentes, pero se puede combinar con otros sistemas. Por ejemplo con el ejemplo anterior quedaría “BM32gc4ht22rs4”.

- Sin vocales. En vez de reemplazarlas por números, este método propone eliminarlas. Por ejemplo para el ejemplo anterior quedaría “BMgchtrs”.

- Involucrar el servicio en la contraseña. Como normalmente un usuario está registrado en muchos servicios y no es recomendable utilizar la misma contraseña, puede utilizar su contraseña robusta antes creada y añadirle el nombre del servicio aplicando alguno de los sistemas ya vistos. Por ejemplo

“F1c2b44k_EbdJptga3€” para un servicio y “Tw3tt2r_EbdJptga3€” para otro.

- Reemplazar por líneas en el teclado. Para este sistema habría que pensar inicialmente en un número no muy extenso, aproximadamente de 6 dígitos, como pudiera ser un código postal, y reemplazar cada número por él mismo y las letras que tiene por debajo. Por ejemplo “06006” quedaría “0pñ6yhn0pñ0pñ6yhn”. Si finalmente si cambia alguna letra por su mayúscula o algún número por su símbolo especial, quedará más robusta, por ejemplo: “0Pñ&Yhn0Pñ=Pñ6Yhn”.

Sin importar lo difícil robusta o compleja que sea una contraseña, sigue siendo la única barrera entre un ciberdelincuente y lo que el usuario intenta proteger, por esta razón merece la pena activar una capa adicional de seguridad, como un segundo factor de autenticación extra más aún si cabe si la información es sensible o es información personal identificable.

3.4 Ataques a las contraseñas

Según un estudio [17] realizado por la empresa Deloitte, casi el 90% de las contraseñas de los usuarios de todo el mundo son vulnerables a los ataques de los ciberdelincuentes. El mismo estudio alerta que el concepto de contraseña robusta que se tiene hoy en día muy pronto dejará de serlo debido al avance de la tecnología.

Existen diversos sistemas para atacar contraseñas, entre ellos:

- El uso de “*keyloggers*”, un malware instalado en el sistema para capturar las teclas pulsadas por el usuario en el teclado. Existe una función denominada “teclado virtual” o “teclado táctil” que puede ayudar a evitar el robo de dato personales en el caso de creer que un equipo está infectado.

- Los “*sniffers*” son programas para interceptar el flujo de datos en las comunicaciones de los equipos de una red, pudiendo extraer de estas posibles contraseñas.

- Existen programas de fuerza bruta que se encargan de crear combinaciones posibles con un abecedario. Para este caso, las contraseñas más largas son más difíciles de obtener.

- También existen los ataques mediante diccionario, en el cual se utiliza un fichero que contiene una cantidad inmensa de contraseñas (por ejemplo las más típicas, las obtenidas en brechas de seguridad anteriores de otros servicios, etc) y van comprobando una a una.

- Otro ataque bastante común está basado en la suplantación de identidad o “phising”, por lo general el atacante utiliza el envío de correos electrónicos suplantando ser una empresa o servicio que requiere al usuario que introduzca sus credenciales para solucionar alguna incidencia (por ejemplo, simular ser un banco diciendo que su cuenta está bloqueada y necesita desbloquearla, o que el propio administrador de la plataforma le solicita que vuelva a introducir sus credenciales sino quiere que eliminen su cuenta, etc).

Otra opción es que el usuario puede optar por algún generador de contraseñas robustas online. Existen multitud de páginas que ofrecen esta herramienta, aunque mejor confiar en alguna que sea de un organismo o empresa fiable. Por ejemplo LastPass, es un popular gestor de contraseñas “freemium” que almacena las contraseñas encriptadas en la nube que a su vez ofrece en su página web [19] un generador de contraseñas robustas. En él, puede elegirse la longitud de la contraseña, si utilizar mayúsculas, minúsculas, números, símbolos entre otras opciones.

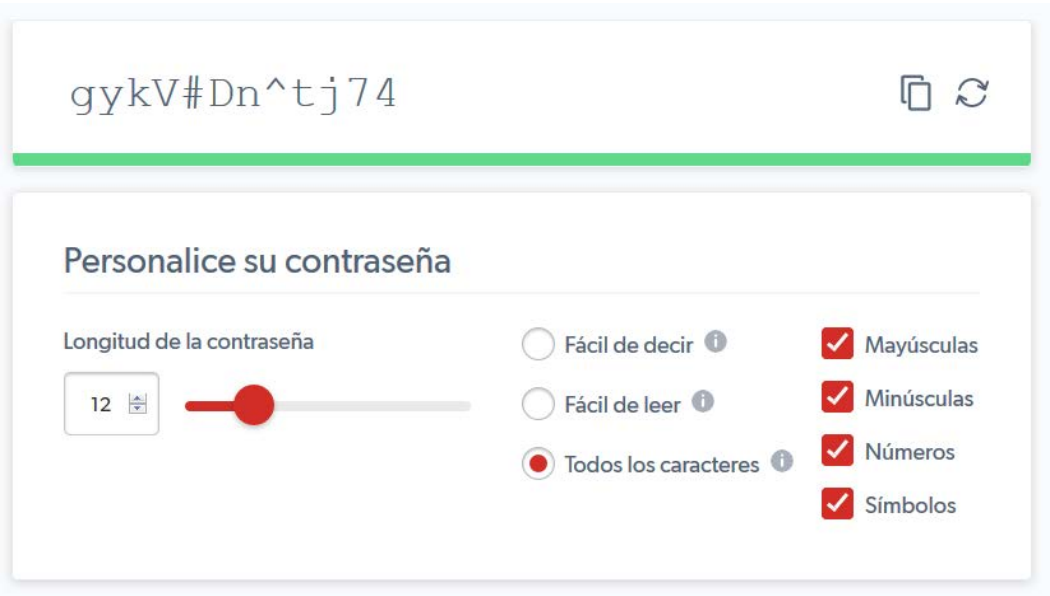


Ilustración 2 - Herramienta de generación de contraseñas ofrecida por LastPass

Como se puede verificar en la Ilustración 2, LastPass ha generado la contraseña robusta “gykV#Dn^tj74”.

Finalmente recordar que una contraseña robusta puede dejar de serlo en el mismo momento en el que se anota sobre un papel, se comparte con alguien o es interceptada por algún malware. Es por ello la importancia de utilizar distintos factores de autenticación para no facilitar el acceso a cualquier ciberdelincuente que obtenga la contraseña de un usuario.

3.5 La autenticación de múltiples factores

Vista la problemática que surge en torno a la utilización de un único factor de autenticación y la facilidad con la que ciberdelincuentes pueden autenticarse impostando su identidad, los expertos en seguridad empezaron a recomendar utilizar una capa adicional de seguridad en la fase de autenticación requiriendo al usuario una información extra que pueda verificar su identidad.

La dureza de un sistema de autenticación es ampliamente determinada por el número de factores incorporados al sistema. Es por ello que implementaciones que usen dos factores son más fuertes que los que utilicen un único factor.

Se denomina con las siglas "MFA" ("*Multi-Factor Authentication*") a aquellos sistemas que utilicen múltiples factores de autenticación, pero el sistema más común es el uso de dos factores, denominándose "2FA" ("*Two-Factor Authentication*"). Aunque como se ha dicho previamente, cuantos más factores se utilicen más robusto será el sistema, tampoco es viable ni cómodo para el usuario que el proceso de autenticación le demore demasiado tiempo, ya que a fin de cuentas es el usuario el que va a utilizar el servicio y no se sentirá cómodo teniendo que emplear demasiado tiempo en demostrar que es él realmente quien está autenticándose.

En la Ilustración 3 se muestra de forma gráfica un mecanismo de autenticación de doble factor en el que el usuario, tras realizar su inicio de sesión con sus credenciales, recibe un código PIN en su terminal móvil para que lo devuelva al sistema de autenticación y ser verificado correctamente.

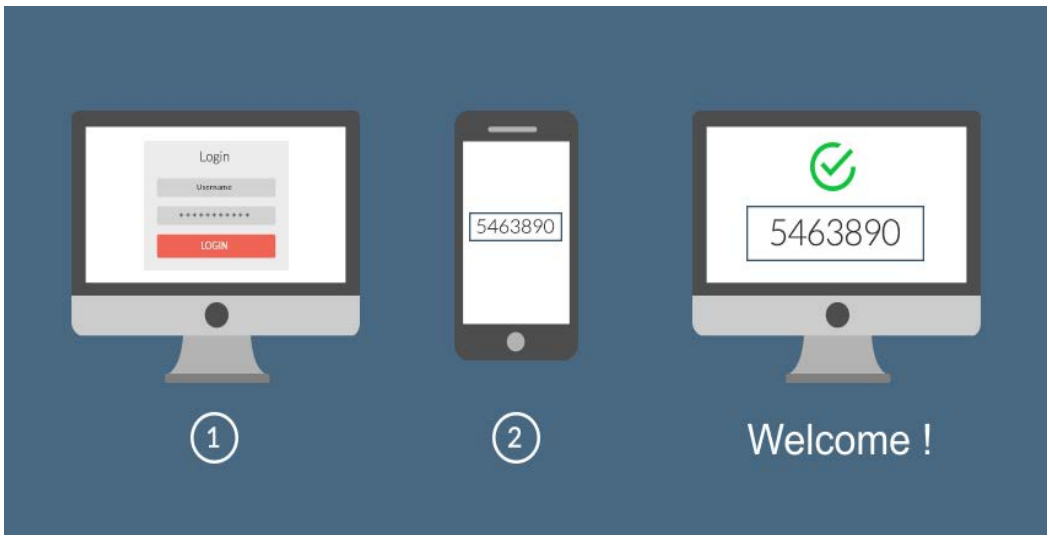


Ilustración 3 - Mecanismo de autenticación de doble factor.

3.5.1 Categorías de factores

Aunque el NIST [20] define en sus reglas generales sobre la autenticación electrónica tres categorías de factores de autenticación:

- "**Algo que el usuario sabe**": Este factor incluye cualquier cosa que pueda comprometer al conocimiento del usuario. Por ejemplo puede ser una contraseña, o un desafío-respuesta (preguntas de seguridad) para recuperar una contraseña olvidada. El problema que presenta es que puede ser algo que otra persona también conozca fácilmente sobre el usuario.

- "**Algo que el usuario tiene**": Este factor incluye algo que físicamente posea el usuario, como puede ser un número de teléfono donde recibir un código, o como puede ser una clave criptográfica o un identificador.

- "**Algo que el usuario es**": Este factor es uno de los niveles más altos de seguridad para el usuario, como puede ser una huella dactilar, el iris u otro dato biométrico.

… aunque con el avance del tiempo se han catalogado dos más:

- "**Algo que el usuario hace**": Este factor utiliza matices de comportamiento diario con las cosas y las personas, como puede ser una firma o dar un paso, u otras tan concretas como [21] "¿Cuál es la primera aplicación a la que accede el usuario en su teléfono después de encenderla el 95% del tiempo? o ¿Qué sitio web comprueba el usuario a primera hora cada mañana?.

- "**Dónde está el usuario**": Este factor utiliza la tecnología de seguimiento de direcciones, como las direcciones MAC e IP, utilizándose para verificar los intentos de acceso y bloquear futuros posibles accesos. Por ejemplo si una person saca dinero de un cajero en un lugar concreto, sería físicamente imposible un intento en un corto espacio de tiempo de sacar dinero de un cajero en un lugar remoto.

Estas dos últimas categorías aún no son reconocidas para lograr una autenticación válida pero si son útiles para evitar comportamientos sospechosos que puedan poner en riesgo los datos del usuario.

3.5.2 No es algo nuevo, se lleva utilizando años

Para la mayoría de las personas en el mundo, las contraseñas y los Números de Identificación Personal (PINs) son la manera en cómo los usuarios se autentican ellos así mismos a sistemas y servicios, sin embargo la combinación de dos factores se presenta de forma cotidiana en algunos actos.

Como por ejemplo cuando una persona va a un cajero a sacar dinero, para ello utiliza su tarjeta (algo que el usuario tiene) e introduce su código PIN (algo que el usuario sabe). O también cuando la persona utiliza su tarjeta de crédito para pagar utilizando la propia tarjeta (algo que tiene) y una firma (algo que es inherente al usuario). Un último ejemplo sería cuando una persona entra en un país extranjero, teniendo que presentar su pasaporte (algo que el usuario tiene) y mostrar su biometría facial para la comparación de la foto (algo inherente a la persona).

3.6 Autenticación de doble factor

Esta autenticación representa una capa adicional de seguridad que complementa el uso de una contraseña. Su principal objetivo es asegurarse de que el usuario no solamente conoce la contraseña para acceder al servicio, sino que además es quien dice ser aportando una información extra, como por ejemplo un código, que posee. Esta información puede obtenerla de la siguiente forma:

- A través de un llamada de teléfono o un SMS enviado por el servicio.
- Haciendo uso de una tarjeta inteligente ("*token*") física o virtual.
- Utilizando un dispositivo biométrico.

3.6.1 OTPs

Un "One-Time Password" (OTP) o también conocido como "One-Time Pin" es una contraseña válida para un único uso, es decir, una única sesión o transacción, en un sistema u otro dispositivo digital. Muchas implementaciones incorporan autenticación de doble factor asegurándose que la contraseña única requiere acceso a algo que la persona tiene (como un pequeño llavero de seguridad con calculadora OTP o tarjeta inteligente o teléfono específico) como también algo que la persona conoce (como un PIN).

Las OTPs evitan los defectos asociados a los tradicionales sistemas de autenticación basados en contraseñas estáticas, las cuales son vulnerables a ataques de reproducción o de reinyección ("replay attacks") en el que una transmisión de datos válida es maliciosa o fraudulentamente repetida por alguien que intercepta la información y la retransmite como parte de un ataque enmascarado para conseguir una suplantación de identidad o una denegación de servicio [26].

Las formas de entrega de un OTP son:

- Teléfonos: ya sea un teléfono móvil convencional o inteligente. La tecnología común que utiliza es la mensajería de texto ya que ésta es un canal de comunicación omnipresente, estando disponible en casi todos los móviles o mediante conversión texto a voz a telefonía fija, alcanzando a todos los consumidores que con coste total muy bajo para implementarlo. en teléfono inteligentes puede ser enviado a través de aplicaciones móviles, que incluyen autenticación dedicada, como Authy, Duo o google authenticator. Estas aplicaciones no comparten las vulnerabilidades de los SMS y no requieren de una conexión a una red móvil para usarlas.

- Tokens propietarios: Es un tipo de token basado en sincronización de tiempo. Es un dispositivo físico que genera OTPs sin necesidad de conexión a internet. SecurID de la empresa RSA Security (Ilustración 4) es uno de los ejemplos más conocidos. sin embargo estos pueden ser perdidos, dañados o robados, aparte del inconveniente de la batería que incorporan que se agota o que no son reemplazables. Estos dispositivos han ido evolucionado, integrándose incluso en tamaño de tarjetas de crédito. Otros modelos recientes se alimentan directamente a un USB.

Ilustración 4 - Dispositivo USB generador de tokens RSA SecurID

- Métodos basados en web: Proveedores de autenticación como servicio ofrecen varios métodos de entrega de contraseñas de un único uso sin necesidad de tokens. Uno de los métodos depende de la habilidad del usuario en reconocer categorías pre-elegidas de una serie de imágenes generadas de forma aleatoria.

- OTP escritas en papel: En algunos países, los bancos envían al usuario una lista numerada de OTPs impresas sobre un papel o una tarjeta. A veces estas contraseñas vienen ocultas y el usuario ha de rascarlas para obtener el código. es el banco el que solicita al usuario un número de OTP concreto de la lista que le ha dado, algunos sistemas lo hacen de forma secuencial y otros pseudoaleatorio. Estas OTPs a veces son denominadas como TANs (“Transaction Authentication Numbers”) o mTANs si son enviadas a un móvil vía SMS.

Ilustración 5 - Tarjeta de coordenadas con OTPs

La clasificación más importante de OTPS [32] puede ser:

- HOPT (*HMAC-based One-Time Password*), contraseña de un solo uso basada en eventos: está basada en el algoritmo criptográfico HMAC y depende de dos tipos de información. El primero es la clave secreta llamada “inicialización” la cual solo conoce el token y el servidor que valida los códigos OTP enviados. el segundo es el factor móvil, que es un contador almacenado en el token y el servidor. El contador del token incrementa al pulsar el botón del token, mientras que el contador del servicio solo se incrementa cuando se valida correctamente una OTP.

- TOPT (*Time-based One-Time Password*), contraseña de un solo uso basada en tiempo: Está inspirada en la anterior, HOTP, pero su factor móvil es el tiempo en lugar de un contador. TOPT emplea tiempo en incrementos llamado “timestep”, que suele ser de 30 o 60 segundos. De esta forma, cada OTP será válida mientras dura el “timestep”.

3.6.2 Envío de código PIN de un solo uso a dispositivo móvil por SMS o correo electrónico

Un dispositivo móvil, como parte en una autenticación de doble factor, puede resultar muy conveniente, ya que es algo que el usuario siempre lleva consigo, es fácil de usar y tiene conexión a internet o a la red de telefonía.

Por otro lado los mensajes de texto suelen ser el paso más débil en una autenticación de doble factor ya que son fáciles de interpretar y no debería asumirse su seguridad. Con una simple ingeniería social un ciberdelincuente podría [25] redirigir los mensajes o llamadas a una tarjeta SIM diferente interceptando así los códigos de acceso debido a un defecto en el protocolo que utilizan la mayoría de los operadores de telecomunicaciones “*Signaling System Number 7*“ (SS7).

Un OTP mediante mensaje de texto debe ser encriptado usando un estándar A5/x, el cual varios grupos de hackers han reportado que puede ser desencriptado con éxito en cuestión de minutos o segundos.

Muchos expertos consideran que un SMS no es algo que el usuario sabe, ni que el usuario tiene, ni algo que es, sino que es algo que le envían. A mediados de 2016, el Instituto Nacional de Estándares y Tecnología de los Estados Unidos (NIST) declaró [22] inseguros a los SMS como método de autenticación en dos pasos.

En lugar de utilizar algo que envía un tercero como un SMS, se pueden utilizar alternativas como aplicaciones dedicadas para ello, como es el caso de Google Authenticator, Authy, RSA SecurID, OTP Authenticator, etc.

3.6.3 Tokens

Los tokens son generalmente algo que el usuario posee y controla que puede ser utilizado para autenticar la identidad del usuario. En la autenticación electrónica, el usuario se autentica a un sistema o aplicación a través de la red, por lo tanto el token utilizado es un secreto que debe ser protegido.

Un token puede ser por ejemplo una llave criptográfica que está protegida por encriptación bajo una contraseña. Un atacante tendría que robar la clave encriptada y encontrar la contraseña para usar el token.

3.6.4 Uso de la biometría

Frente al fraude de documentos y el robo de identidad con nuevas amenazas como el terrorismo o la delincuencia informática se están implementando nuevas soluciones tecnológicas. Una de estas es la biometría, estableciéndose como el medio más pertinente para identificar y autenticar personas de una manera rápida y confiable mediante el uso de características biológicas únicas.

La biometría es [31] la ciencia del análisis de las características físicas o del comportamiento, propias de cada persona, con el fin de autenticar su identidad. Se pueden establecer dos categorías de tecnologías biométricas:

- Mediciones fisiológicas: contemplan los análisis morfológicos, como las huellas dactilares, forma de la mano, del dedo, el patrón de venas, el ojos (iris y retina) y la forma de la cara; y los análisis biológicos, como el ADN, la sangre o la orina.

- Mediciones del comportamiento: analizan el reconocimiento de la voz, la dinámica de la firma, la dinámica de pulsación de teclas, la

manera de usar objetos, la marcha, el sonido de los pasos, los gestos, etc.

La autenticación biométrica es el proceso en el que se comparan los datos de las características de una persona con la plantilla biométrica previamente obtenida de esa persona con el fin de determinar su semejanza. El modelo de referencia se almacena en bases de datos o en un elemento seguro portátil, como una tarjeta inteligente.

3.6.5 Seguridad en los sistemas biométricos

Con el gran avance de la tecnología en los últimos años, es muy común ver ya en los dispositivos móviles sensores de lectura de huella dactilar. El primero en dar el paso al frente fue Apple con su iPhone 5S. Aunque parezca un método novedoso no ha dejado de estar bajo escrutinio entre los profesionales de la seguridad de la información, ya que la asociación de hackers *Chaos Computer Club* fue capaz de burlar el sensor del iPhone 5S dos semanas después de su presentación.

Los sistemas biométricos fueron concebidos para proporcionar un mecanismo de identificación dirigido a varios objetivos, aunque el más común es proporcionar seguridad a un recurso, como la autenticación.

Un sistema de seguridad, utilice o no información biométrica, puede ser objeto de una serie de ataques [24], entre los que destacan:

- Ataques de suplantación de la persona ("*spoofing*"): consiste en suplantar la identidad de un usuario para obtener un acceso ilícito a un recurso deseado.

- Ataque de ofuscación biométrica: está dirigida a falsear o enmascarar datos biométricos para evitar ser reconocidos. Este ataque suele ser para evitar ser reconocido por un sistema como por ejemplo un aeropuerto.

- Ataque de denegación de servicio: está dirigido a retrasar, detener o degradar la calidad de un sistema.

- Ataque por conspiración y coacción: este ataque lo ejecutan los usuarios legítimos del sistema posiblemente por soborno, amenaza o chantaje.

El punto débil de seguridad en la mayoría de sistemas biométricos ha sido la utilización de biometría falsa, es decir biometría sintética obtenida a partir de fotografías o creada mediante moldes.

Por ejemplo para las huellas dactilares, se demostró [23] que utilizando una imagen de alta resolución (2400 dpi) de una huella impresa en algún objeto, tras tratarla con un software de retoque fotográfico y aislar los

contornos, se imprimió el negativo sobre una lámina transparente. Aprovechando que las impresoras láser utilizan toners que dejan un relieve se aplicó una película de un material flexible y translúcido similar a la piel que fué capaz de desbloquear un dispositivo como si un dedo se tratase.

Por otro lado, en referencia a los sistemas de reconocimiento facial, ha sido demostrado cómo se han podido burlar estos mecanismos mediante el uso de fotografías de personas ante el sensor del mecanismo de reconocimiento.

Para protegerse de estos ataques algunos sistemas han empezado a implantar tecnología para aplicar la denominada “vida de la muestra”, es decir, que el propio sistema sea capaz de reconocer si la muestra que está identificando tiene vida o no. Por ejemplo, reconocer los vasos sanguíneos, las pulsaciones del corazón, el calor humano, que el iris tenga movimiento, etc.

3.6.6 Otros puntos débiles de la autenticación de doble factor

La autenticación de doble factor no es la salvación a los problemas de autenticación, más bien resuelve los problemas de seguridad que se tenían hace 10 años pero no los problemas de seguridad actuales.

Desafortunadamente la naturaleza de los ataques ha cambiado [28] en las últimas dos décadas. Anteriormente las amenazas eran todas pasivas, como escuchas a escondidas o adivinar contraseñas en modo “*offline*”. Hoy en día son más activas, como los ataques de “*phising*” o caballos de Troya (troyanos):

- Ataque de intermediario o ataque “*Man-in-the-Middle*” (MitM o JANUS): es un ataque [29] en el que se adquiere la capacidad de leer, insertar y modificar a voluntad, interceptando mensajes entre las dos víctimas y procurando que ninguna de las víctimas conozca que el enlace entre ellos ha sido comprometido. Por ejemplo, un atacante levanta una página web falsa de un banco y persuade a un usuario a tal página. El usuario escribe su contraseña y el atacante lo envía a la página real del banco. En este punto el atacante conoce la contraseña del usuario y puede realizar una transacción fraudulenta.

- Ataque de caballo de Troya (troyano): es un malware [30] que se presenta al usuario como un programa aparentemente legítimo e inofensivo, pero que al ejecutarlo le brinda al atacante un acceso remoto al equipo infectado. Por ejemplo con un troyano instalado en el ordenador, cuando el usuario se identifica en la página web del banco, el atacante se suma a la sesión mediante el troyano para realizar las transacciones fraudulentas que quiera.

4. ANÁLISIS DE PROTOCOLOS

Como ya se ha visto, la generación de códigos de un solo uso utiliza la aleatoriedad o más bien pseudoaleatoriedad para dificultar a los atacantes la predicción de sucesivos códigos. Aparte, se utilizan funciones de resumen o también llamadas “hash” mediante técnicas criptográficas lo cual implica que sea más difícil revertir u obtener la información de origen ya que la mayoría de técnicas crean resúmenes de un único sentido.

En esta sección primeramente se va a describir el protocolo de un sistema OTP (contraseña de un solo uso) y después se van a describir los protocolos utilizados en las dos formas más empleadas de generación de contraseñas de un solo uso: las basadas en criptografía HMAC (HOTP) y las basadas en tiempo (TOTP).

Para ello se van a utilizar los documentos RFC (“*Resquests For Comments*”) que la IETF (“*Internet Engineering Task Force”*), una comunidad abierta que desarrolla y promueve los estándares de Internet. Los documentos publicados por esta organización para estos protocolos son el RFC-2289 [38] para el sistema OTP, el RFC-4226 [33] para el HOTP y el RFC-6238 [34] para el TOTP.

Los RFC son [35] publicaciones que la IETF sobre diversos aspectos del funcionamiento de Internet y otras redes de computadoras, como protocolos, procedimientos, etc. y comentarios e ideas sobre estos las cuales constituyen un monográfico escrito en texto plano que los expertos han hecho llegar a la IETF.

Aunque ya se han descrito muy brevemente en el capítulo anterior, se procede a entrar en más profundidad de estos protocolos.

4.1 Sistema One-Time Password

La seguridad de un sistema OTP está basada en la posibilidad de no poder invertir la función de un resumen hash. Tales funciones deben ser manejables para un ordenador en el sentido de la generación pero computacionalmente incapaz de revertirlo.

Las tres principales funciones criptográficas son MD4, MD5 y SHA. Tanto el generador de códigos como el validador han de usar el mismo algoritmo para poder interoperar entre ellos. Estos algoritmos tienen la capacidad de aceptar una entrada de longitud arbitraria y de producir una salida de longitud fija de 64 bits, una longitud que se considera suficiente para ser segura y lo suficientemente corta para ser entrada manualmente por un usuario.

4.1.1 Generación de un One-Time Password o contraseña de un solo uso

Este proceso [38] consiste en un paso inicial en el cual todas las entradas son combinadas, un proceso computacional en el que todas las funciones resumen son aplicadas y por último una función de salida donde la contraseña de 64bits es convertida a una forma legible por humanos.

En el paso inicial, la contraseña secreta del usuario puede ser de cualquier longitud pero se recomienda que al menos contenga 10 caracteres. Todas las implementaciones deben soportar frases de contraseña de hasta 63 caracteres. En este paso, la frase de contraseña es concatenada con una semilla que es transmitida desde el servidor en texto claro. Esta semilla no secreta permite a los clientes usarla misma frase de contraseña secreta en múltiples máquinas (usando diferentes semillas). El resultado de la concatenación se pasa a través de una función resumen y es reducida a 64bits.

En el paso computacional, una secuencia de contraseñas de un sólo uso es generada aplicando la función resumen segura múltiples veces a la salida del paso inicial (el anterior descrito).

4.1.2 Verificación de una contraseña de un solo uso

Una aplicación en el servidor de verificación que requiere una autenticación OTP espera expedir un desafío con unos parámetros, que combinado con la frase de contraseña secreta, el generador puede computar el OTP que ha sido enviado al servidor para ser verificado.

El servidor tiene una base de datos que contiene por cada usuario el OTP desde su última autenticación con éxito o el primer OTP de una nueva secuencia iniciada. Para autenticar al usuario, el servidor decodifica el OTP recibido del generador a una clave de 64 bits y ejecuta esta clave a través de una función de resumen segura. Si el resultado de esta operación coincide con el OTP previamente almacenado, la autenticación tiene éxito y el OTP aceptado es almacenado en la base de datos

4.2 HOTP - HMAC based One-Time Password

HTOP es un algoritmo de autenticación que permite la generación de contraseñas de un sólo uso basadas en el mecanismo criptográfico HMAC, surgido de la necesidad de un mecanismo de autenticación robusto y que supliese a los esquemas de autenticación débiles. Este protocolo fue propuesto por OATH (*Open AuTHentication*) y fue publicado en Diciembre 2005 debido [33] al gran crecimiento de las amenazas en la red antes las contraseñas estáticas permitiendo con este protocolo implementar autenticación de doble factor tanto por software

como por hardware en dispositivos que puedan trabajar en un amplio rango de aplicaciones asegurando en todo momento la interoperabilidad. Las OTPs a veces son preferidas a otras formas de autenticación fuertes como PKI (*Public-Key Infrastructure*) o biometría.

La IETF publicó el algoritmo a través de una implementación en Java y desde entonces ha sido adoptado por muchas compañías, convirtiendo éste en un estándar abierto.

4.2.1 Requerimientos del algoritmo

Entre los requerimientos del algoritmo se puso mucho énfasis en la usabilidad por parte del consumidor final así como en la habilidad de poder ser implementado mediante hardware de bajo coste, sin embargo la IETF define [33] los siguientes requerimientos

- El algoritmo debe ser secuencial o basado en un contador.

- El algoritmo debería ser económicamente fácil de implementar en un dispositivo, minimizando los requerimientos de batería, número de botones, potencia computacional y el tamaño de la pantalla LCD.

- El algoritmo debe trabajar con tokens que no soporten entrada numérica, salvo excepciones con dispositivos seguros como los PIN-pads (normalmente usados como terminales de pago, Ilustración 6).

- El valor mostrado en el token debe ser legible e introducido posteriormente por el usuario: esto requiere que la longitud sea razonable, de al menos 6 dígitos.

- Debe ser existir mecanismos para resincronizar el contador.

- El algoritmo debe usar un secreto compartido fuerte con una longitud de al menos 128 bits, aunque la IETF recomiendo una longitud del secreto compartido de 160 bits.

Ilustración 6 - PIN-pad utilizado como terminal de pago mediante tarjetas

4.2.2 El algoritmo HTOP

El algoritmo está basado en un valor de un contador incremental y una clave simétrica estática únicamente conocida por el token (el cual posee el usuario) y el servicio de validación.

Para generar el valor se usará el algoritmo HMAC-SHA-1, definido por la IETF en el RFC-2104. Normalmente ha sido considerada una función segura, pero ya en 2004 se dio a conocer un número significante de ataques mediante colisiones contra funciones similares y en 2017 Google anunció [5] que había sido capaz de crear dos ficheros PDF distintos con el mismo resumen SHA-1, lo cual podría considerarse una vulnerabilidad muy grande.

Como la salida de la función HMAC-SHA-1 genera un resumen de 160 bits (20 bytes) se debe realizar una operación de truncado a “algo” que sea fácilmente después retornado por el usuario, ya que un código de muchos dígitos dificultaría el acto de autenticación y empeoraría la experiencia final del usuario. Es importante que los valores generados por el algoritmo HOTP sean tratados como big-endian [36], es decir, representar los bytes en el orden natural (desde la izquierda) y evitar así problemas de interpretación.

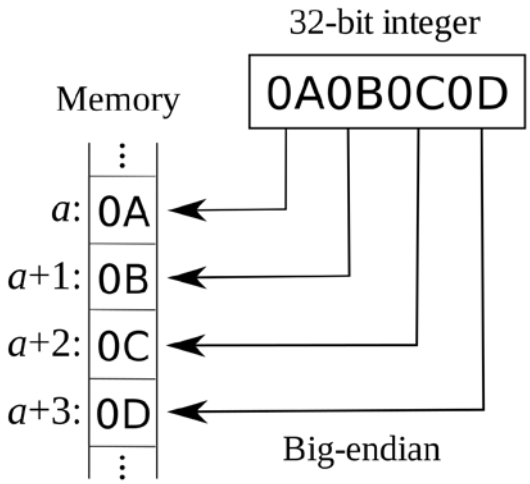


Ilustración 7 - Orden natural de una cadena hexadecimal mediante big-endian [4]

El proceso de generación de un código OTP (reflejado en la Ilustración 8) se puede definir de la siguiente manera:

$$\mathrm{HOTP}(K, C) = \mathrm{Truncate}\,(\mathrm{HMAC} - \mathrm{SHA} - 1(K, C))$$

donde K es la clave compartida secreta y C el valor del contador incremental. Se puede dividir en tres pasos:

Generación de un valor HMAC-SHA-1 tal que:

$$HS = HMAC - SHA - 1(K, C)$$

Generación de una cadena de 4 bytes de longitud mediante una operación de truncado dinámico (DT) tal que Sbits= DT(HS)

Se toman los 4 últimos bits del último byte de HS (HS[19]), que será el offset del byte a partir del cual se toman los 4 bytes quitándole el bit más significativo para evitar problemas con el signo, es decir, 31 bits.
Se calcula el valor HOTP, para ello se convierten los 31bits tomados en un número N entre 0 y 2^{31}-1. El valor final HOTP se obtiene tomando el módulo N%$10^{(digitos)}$ y por tanto obteniendo un número entre 0 y $10^{(dígitos)}$ - 1.
El número de dígitos debe ser 6 como mínimo, permitiéndose implementaciones con 7 u 8 dígitos también, dependiendo de los requisitos de seguridad.

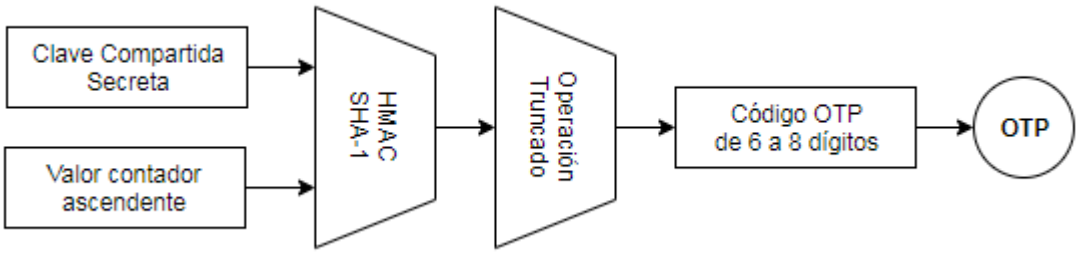


Ilustración 8 - Proceso de generación de un código OTP mediante HOTP

4.2.3 Ejemplo de obtención de un valor HOTP

A modo de ejemplo la IETF escenifica [33] estos tres pasos de la siguiente manera. En principio se supone que el resultado del mecanismo criptográfico HMAC-SHA-1 de 160 bits (20 bytes) para una clave secreta compartida determinada y un valor concreto del contador es el contenido de la Ilustración 9.

```
-------------------------------------------------------------
| Byte Number                                               |
-------------------------------------------------------------
|00|01|02|03|04|05|06|07|08|09|10|11|12|13|14|15|16|17|18|19|
-------------------------------------------------------------
| Byte Value                                                |
-------------------------------------------------------------
|1f|86|98|69|0e|02|ca|16|61|85|50|ef|7f|19|da|8e|94|5b|55|5a|
--------------------------------***********----------------++|
```

Ilustración 9 - Bytes que componen el resúmen devuelto por una función HMAC-SHA-1

Es decir, HS=1F 86 98 69 0E 02 CA 16 61 85 50 EF 7F 19 DA 8E 94 5B 55 5A.

Siguiendo con los pasos mencionados anteriormente, hay que extraer el byte HS[19], es decir el último: HS[19] = 0x5A,del cual, los 4 bits más bajos son 0xA que convertido a bits es b1010y en decimal equivale al número 10, por lo cual ese será el offset.

El valor de los 4 bytes comenzando en el byte 10 son 0x50EF7F19, sin embargo como el MSB de 0x50 es 0, no habrá que tener cuidado ya que no va a cambiar el signo, por lo que en decimal equivale a 1357872921

Finalmente, se hace el módulo para obtener el valor final de forma que:

- Para un HOTP de 6 dígitos: HOTP = 1357872921 modulo 10^6 = 872921
- Para un HOTP de 7 dígitos: HOTP = 1357872921 modulo 10^7 = 7872921
- Para un HOTP de 8 dígitos: HOTP = 1357872921 modulo 10^8 = 57872921

4.2.4 Consideraciones de seguridad

Las conclusiones de seguridad que presenta [33] la IETF indica que la salida de la operación de truncado dinámico sobre valores contadores distintos es uniforme e independientemente distribuida. Los análisis de seguridad detallan el impacto de la conversión de una cadena hexadecimal a un entero y la reducción final módulo 10 dependiente del número de dígitos deseado la cual introduce una insignificante tendencia la cual no impacta en la seguridad del algoritmo HOTP. Finalmente la IETF observa que podría ser vulnerable a ataques de fuerza bruta, el cual podría ser reducido utilizando un parámetro de estrangulamiento ante sucesivos intentos fallidos.

4.2.5 Validación de valores HOTP

Un cliente HOTP (un token por software o hardware) incrementa su contador y entonces calcula el siguiente valor HOTP del cliente HOTP. Si el valor recibido en el servidor de autenticación coincide con el valor calculado por el cliente HOTP entonces éste es validado. En este caso el servidor incrementa su valor en uno.

Si el valor recibido en el servidor de autenticación no coincide con el de cliente HOTP, el servidor inicia el protocolo de resincronización antes de pedir otra contraseña.

Si la resincronización falla, el servidor pregunta por otra contraseña de autenticación del protocolo hasta que se alcance el máximo de intentos permitidos.

Si esto llega a ocurrir, el servidor debería bloquear la cuenta e iniciar un procedimiento de informar al usuario.

La operación de truncado del valor HMAC-SHA-1 a un valor más corto hace posible un ataque de fuerza bruta, por lo tanto, el servidor de autenticación necesita detectarlo y pararlo. Se recomiendo establecer un parámetro llamado “de estrangulamiento” el cual define el número máximo de intentos posibles para validar un OTP. Este valor se recomiendo que sea lo más bajo posible mientras se asegura que no haga impacto en la usabilidad. Otra opción sería implementar un retardo para evitar un ataque de fuerza bruta de forma que cada vez que ocurra un intento fallido el servidor espere por un tiempo en segundos proporcional al número sucesivo de intentos fallidos.

4.2.6 La resincronización del contador incremental

Aunque el valor del contador del servidor solamente se incrementa después de una autenticación HOTP con éxito, el contador en el token que posee el usuario se incrementa cada vez que solicita uno nuevo. Debido a esto, los valores del contador en el servidor y en el token pueden desincronizarse. Para ello se recomienda establecer un parámetro llamado “ventana” con el cual el servidor puede recalcular los siguientes valores HOTPs y validarlos frente al recibido por el cliente HOTP.

El sistema puede requerir al usuario enviar una secuencia de varios valores HOTP más consecutivos ya que sería más difícil que adivinar un valor individual. El valor de ventana debe ser fijado lo más bajo posible

4.3 TOTP - TIME BASED ONE-TIME PASSWORD

TOTP es un algoritmo de autenticación que permite la generación de contraseñas de un solo uso con un periodo de validez limitado o temporal. Está basado en el algoritmo HOTP visto en el capítulo anterior pero se diferencia en que éste usa como factor variable el tiempo en vez de un contador incremental, evitando así problemas de sincronización del contador entre el servidor y el cliente.

La IETF recomienda utilizar HMAC-SHA-256 o HMAC-SHA-512 en vez de HMAC-SHA-1 especificada para HOTP.

4.3.1 Requerimientos del algoritmo

Los requerimientos [34] a tener en cuenta para diseñar el algoritmo TOTP son:

- Tanto el usuario (que posee el token) y el servidor de validación deben conocer y ser capaces de obtener el tiempo Unix (es decir, el número de segundos transcurridos desde la medianoche UTC del 1 de Enero de 1970).

- Tanto el usuario como el servidor deben compartir el mismo secreto o la transformación secreta para generar una clave compartida.

- El algoritmo debe usar el algoritmo HOTP.

- El usuario y el servidor deben usar el mismo paso de tiempo, es decir, modificar el OTP cada mismo tiempo ambos.

- Debe existir un único secreto o clave para cada usuario.

- Las claves deben ser generadas de forma aleatoria o derivadas usando algoritmos de derivación.

- Las claves deben ser almacenadas en un dispositivo resistente a falsificación y protegido contra acceso y uso no autorizado.

4.3.2 El algoritmo TOTP

Esta variante del algoritmo HOTP se centra en el cálculo de un valor OTP basado en una representación de un contador incremental como un factor de tiempo. Se define de la siguiente manera:

$$TOTP = HOTP\,(K,T)$$

Para este algoritmo se definen las siguientes notaciones:

- X representa el paso de tiempo en segundos y es un parámetro del sistema. Por defecto el valor de X es 30.

- T0 es el tiempo Unix para empezar a contar pasos de tiempo. Por defecto T0 vale 0, y es también un parámetro del sistema.

Básicamente se define [34] T como un número entero y representa el número de pasos de tiempo entre el tiempo inicial T0 y el tiempo actual Unix.

$$T = \frac{(\,Current\ Unix\ Time - T0\,)}{X}$$

Se utilizará la función de redondeado hacia abajo por defecto. Por ejemplo, si T0=0, X=30, T=1 si el tiempo actual UNIX es 59 segundos, y T=2 si el tiempo actual UNIS es 60 segundos.

El algoritmo debe soportar un valor T más mayor que un número entero de 32 bits cuando se supere el año 2038. Los valores de los parámetros del sistema X y T0 son preestablecidos durante el proceso de aprovisionamiento y comunicado entre el usuario y el servidor.

4.3.4 Consideraciones de seguridad

La seguridad y fortaleza de este algoritmo dependen de las propiedades que recaen sobre el bloque HOTP, el cual es una construcción basada en HMAC usando SHA-1 como función resumen.

Al estar basado en el protocolo RFC 4226 que definía al algoritmo HOTP, tiene las mismas consideraciones de seguridad, por lo cual puede ser vulnerable a un ataque de fuerza bruta.

Como las claves deben ser elegidas de forma aleatoria se usará un función pseudoaleatoria lo suficientemente criptográficamente fuerte, alimentada por un valor aleatorio, definiéndose algunas recomendaciones en el RFC 4086 que trata la generación de número pseudoaleatorios.

Todas las comunicaciones deben ser realizadas por un canal seguro, como por ejemplo SSL/TLS o conexiones IPsec.

El almacén de claves debe estar en un área segura para evitar ataques directos en el sistema de validación, quedando limitado el acceso a procesos y programas requeridos únicamente por el sistema de validación.

4.3.5 Validación y el tamaño del paso de tiempo

Un OTP generado dentro del mismo paso de tiempo deberá ser el mismo. Cuando un OTP es recibido en el servidor de validación no conoce cuál fue el sello de tiempo del usuario cuando el OTP fue generado. El sistema de validación utiliza el sello de tiempo para realizar la comparación, sin embargo habrá que tener en cuenta posibles latencias en la red, debiendo crear una ventana de margen para que el sistema de validación compare contra anteriores OTPs.

El paso de tiempo tiene impacto tanto en la seguridad como en la usabilidad. Cuanto más grande sea significa una ventana mayor de validación para que un código de un sólo uso sea aceptado por el sistema de validación, exponiendo a posibles ataques. Se recomienda por defecto un paso de tiempo de 30 segundos (x = 30s).

Otro problema de poner un paso de tiempo muy elevado es que la espera entre un OTP y el siguiente. El usuario debe esperar hasta que el reloj llegue al siguiente paso de tiempo para obtener otro código de un sólo uso tras una autenticación válida. Por ejemplo, un paso de 10 minutos no sería cómodo para la típica autenticación de internet ya que un usuario no sería capaz de obtener el siguiente OTP dentro de 10 minutos y además tendrá que reautenticarse al mismo sitio en 10 minutos. El servidor no aceptará un segundo intento del OTP después de una validación correcta.

4.3.6 Resincronización

Debido a la posible diferencia del reloj entre el usuario y el servidor de validación, se recomienda que el validador sea ajustado con un límite específico al número de pasos que el usuario puede estar desincronizado antes de ser rechazado.

Este límite puede ser ajustado tanto hacia delante como hacia atrás del OTP calculado. Por ejemplo, si el paso de tiempo está ajustado a 30 segundos y el validador está ajustado a aceptar sólo dos pasos de tiempo hacia atrás, entonces el máximo de tiempo transcurrido ronda los 89 segundos (29 en el paso calculado y 60 segundos de los dos pasos hacia atrás).

5. SISTEMAS QUE INTEGRAN OTP COMO SOLUCIÓN

Tras haber expuesto los conceptos más importantes referentes a la autenticación de doble factor y exponer los protocolos de generación de contraseñas de un solo uso, antes de poner en práctica el concepto se expone en esta sección una serie de servicios empresariales que ayudan hoy en día en la implementación de una capa de seguridad extra a aplicaciones web y móviles.

5.1 Google Authenticator

Está desarrollado por Google y es [41] un software basado en autenticación con contraseña de un solo uso. Ofrece un código numérico de seis dígitos que el usuario debe proporcionar además de su nombre de usuario y contraseña. Estos códigos que genera utilizan los estándares abiertos por “The Initiative for Open Authenticator” (OATH).

Google Authenticator puede generar también códigos para aplicaciones de terceras partes.

El proveedor de servicio genera una clave secreta de 80 bits para cada usuario. Esta se suministra como una cadena de 16 caracteres codificados en base32 o como un código QR. El cliente crea un mensaje HMAC-SHA1 usando esta clave secreta, quedando un mensaje generado basado en el número de periodos de 30 segundos transcurridos desde el comienzo Unix o un contador que es incrementado con cada nuevo código.

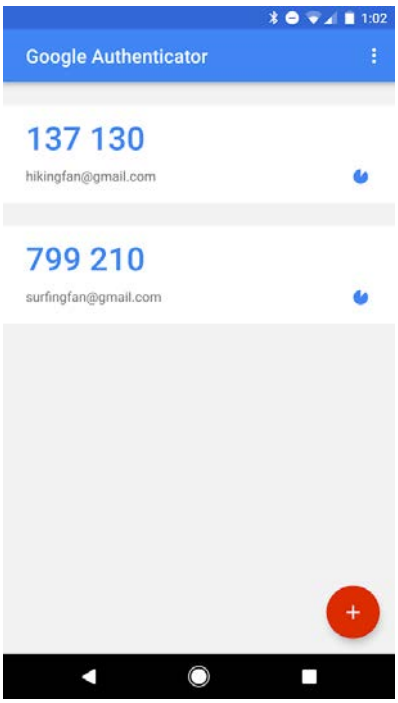


Ilustración 10 - Aplicación móvil Google Authenticator ofreciendo un TOTP para distintos servicios

5.2 Authy

Es [42] un cliente que permite gestionar distintas cuentas de servicios online utilizando autenticación de doble factor de una forma fácil y con opción de sincronización entre varios dispositivos, sincronización en la nube y generación de códigos de un sólo uso de forma offline. Es una solución compatible con las principales plataformas, como Android, iOS, Windows, macOS o Chrome.

Para acceder a los tokens de doble factor se puede utilizar TouchID, un código PIN o una contraseña.

Ilustración 11 - Códigos de un sólo uso para distintos servicios online con la aplicación móvil Authy

5.3 Latch

Es una aplicación de propiedad del Grupo Telefónica, y como su propio nombre indica en inglés (pestillo) permite [39] a los usuario activar o desactivar sus cuentas de redes sociales para evitar el robo de información por parte de ciberdelincuentes. Una de sus características es que permite bloquear todo tipo de servicios online durante un tiempo establecido y obtener una notificación de intentos de acceso no autorizados mediante de control del cliente, así como poder bloquear las cuentas por la noche de forma automática. Esto permite añadir una capa extra de seguridad.

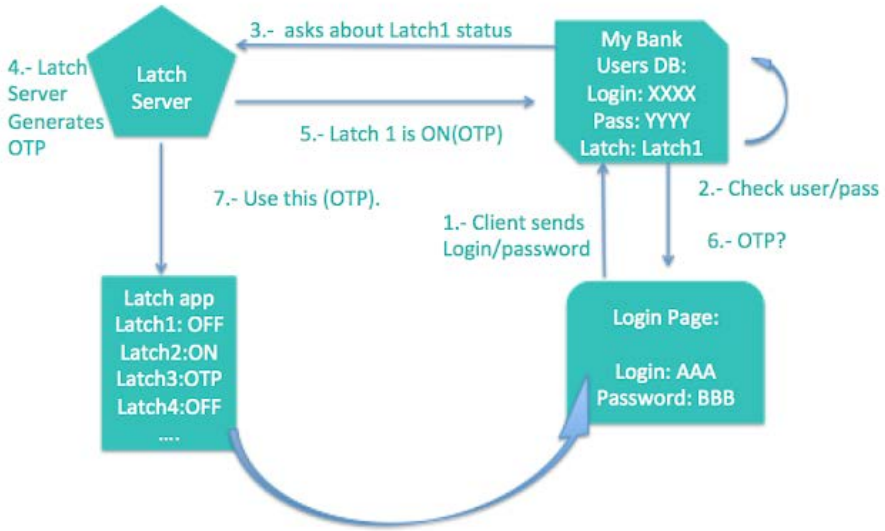


Ilustración 12 - Opción de utilizar OTP en Latch para inicio de sesión

Se puede utilizar con servicios tan variados como bancos, cuentas de correo empresarial y redes sociales tanto en iOS, Android y Windows Mobile.

Latch ofrece un nuevo factor de autenticación basado en la generación de contraseñas tipo OTP generadas a partir de información biométrica obtenida de la voz del usuario [40]. El proceso de capturar y procesar los datos biométricos que se utilizarán a su vez para generar una clave, son realizados desde el smartphone del usuario, no requiriendo de ningún otro dispositivo adicional. Esta clave criptográfica generada a parte de esta información se utilizará para cifrar y descifrar la información intercambiada con el servidor.

6. IMPLEMENTACIÓN DEL CONCEPTO

Como última parte de este proyecto se ha propuesto realizar la implementación de un sistema de autenticación usando un doble factor tal como un código de un solo uso.

La idea que se pretende realizar en esta práctica final es la implementación de una página de autenticación o "*login*" en la cual un usuario introduzca sus credenciales, y le sea requerido introducir un código de un solo uso (OTP) que reciba bien vía SMS, o vía correo electrónico, según las capacidades y servicios que se puedan usar de forma gratuita.

Debido a la limitación que puede suponer el envío de mensajes SMS con los códigos finalmente se ha optado por utilizar una cuenta de pruebas en la plataforma Twilio la cual ofrece un mínimo de saldo gratis al registrarse para hacer pruebas.

Se va a utilizar un servidor montado con stack tecnológico formado por NodeJS, MongoDB y ExpressJs.

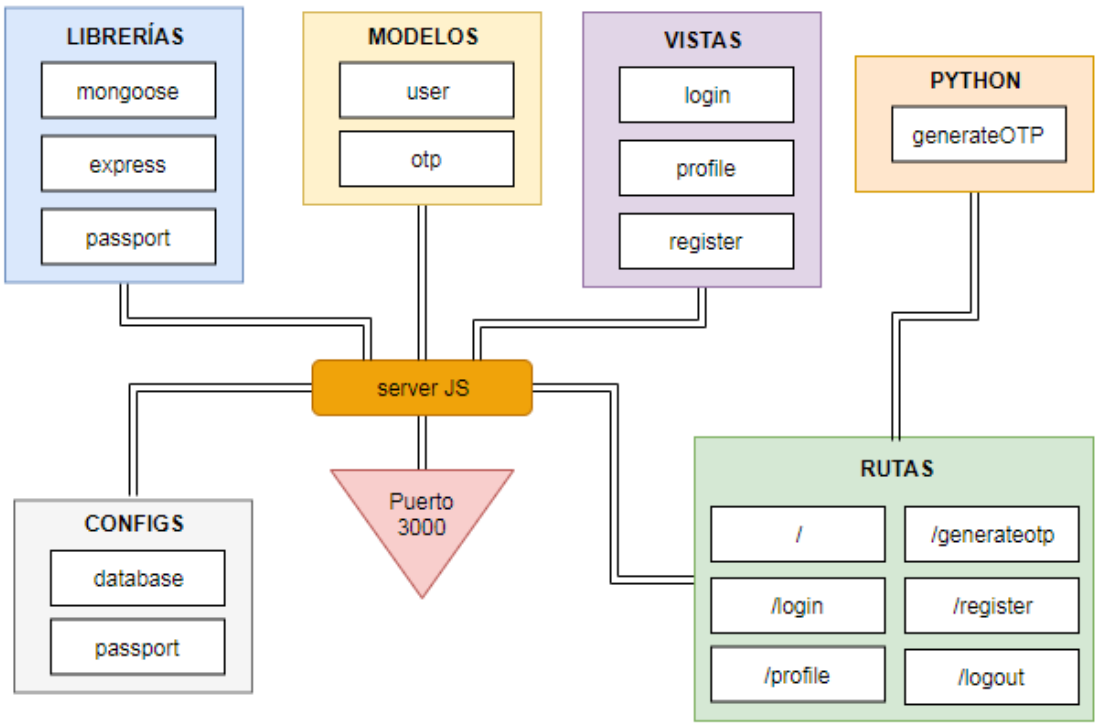


Ilustración 13 – Esquema general del servidor levantado en NodeJS

6.1 Servidor NodeJS

Tal y como se ha mencionado en la introducción de este capítulo, se levantará un servidor con tecnología NodeJS, la cual es [45] un entorno de ejecución multiplataforma de código abierto para la capa servidor, basado en el lenguaje de programación ECMAScript (lenguaje de programación basado en el popular javascript), asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Esto último permite a Node proporcionar un entorno de ejecución del lado del servidor que ejecuta javascript a velocidades elevadas. La gran diferencia con javascript es que se ejecuta en el lado del servidor y no del cliente.

En la Ilustración 138 se describe un esquema general de las conexiones entre las distintas partes del proyecto, como son las librerías, las vistas, las rutas, los modelos y las configuraciones.

6.2 Librerías principales

Entre las librerías principales que se van a utilizar en el proyecto con NodeJS están:

- Mongoose: permite una increíble cantidad de funcionalidades para crear y trabajar con esquemas Mongoose, los cuales permiten definir objetos con un esquema fuertemente tipado que se asigna a un documento MongoDB. También permitirá realizar las conexiones a la base de datos y las operaciones sobre ella.

- Express: es [46] el framework web más popular de Node, y es la librería subyacente para un gran número de otros frameworks web de Node populares. Proporciona mecanismos para:

    - Escritura de manejadores de peticiones con diferentes verbos HTTP en diferentes caminos URL (rutas).

    - Integración con motores de renderización de "vistas" para generar respuestas mediante la introducción de datos en plantillas.

    - Establecer ajustes de aplicaciones web como qué puerto usar para conectar, y la localización de las plantillas que se utilizan para renderizar la respuesta.

    - Añadir procesamiento de peticiones "middleware" adicional en cualquier punto dentro de la tubería de manejo de la petición

- Passport: La documentación [47] de Passport lo describe como un simple "*middleware*" no intrusivo de autenticación para NodeJS. Proporciona más de 140 mecanismos de autenticación para escoger, aunque en este proyecto solo se ha usado el más básico (una autenticación local con usuario y contraseña). Trabaja mediante estrategias que pueden ser definidas por el propio usuario.

6.3 Base de datos MongoDB. Modelos

Para almacenar los datos de la aplicación se ha optado por utilizar una base de datos NoSQL llamada MongoDB. Está orientado a documentos y es de código abierto. MongoDB [44] guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.

Para este proyecto se ha utilizado la versión de MongoDB 3.2.4 sobre un sistema operativo Windows 7 de 64 bits. Se ejecuta sobre el puerto por defecto 27017.

6.3.1 Modelos de datos

Se ha decidido usar dos tablas, o mejor dicho, colecciones de documentos, una para la gestión de usuarios de la aplicación y otra para la gestión de los códigos OTPs generados. Estas están formadas por los siguientes atributos.

6.3.2 Modelo <users>

En esta colección se almacenará el email del usuario, su contraseña y una credencial que consiste en la concatenación del correco electrónico con el código hash de la contraseña convertido esto a base64

atributo	tipo
email	String
password	String
credential	String
phone	String

Tabla 2 – Modelo de datos de la colección users

Este modelo cuenta con dós métodos:

- generarHash: el cual se encarga de generar una cadena hash con encriptación bcrypt.

- validarPassword: el cual se encarga de verificar que el password enviado por el usuario coincide con el hash almacenado

6.3.3 Modelo <otps>

En esta colección se almacenará el email la credencial del usuario, para identificarle y relacionado con la de la tabla users, el código otp, un indicador de si ha sido utilizado o no y una variable de fecha para saber si está dentro de su validez o ha expirado y hay que generar uno nuevo.

atributo	tipo
credential	String
otpcode	String
used	Boolean
valid	Date

Tabla 3 - Modelo de datos de la colección users

Este modelo cuenta con un método:

- validarOTP: el cual se encarga de verificar que el OTP introducido por el usuario es válido. Para ello comprueba que no esté usado, que sea el mismo que está almacenado, y que esté dentrodel periodo de validez de éste.

6.4 Algoritmo de Python para la generación de códigos OTP

Para la generación de códigos OTP o de un solo uso se ha optado por utilizar el lenguaje de programación Python. Este es [43] un lenguaje interpretado que hace hincapié en una sintaxis que favorezca un código legible. Es también un lenguaje multiparadigma, ay que soporta orientación a objetos, programación imperativa y programación funcional. Es de código abierto con una licencia general pública GNU

Se han tenido las siguientes consideraciones:

- Se ha realizado un algoritmo que genera códigos OTP de 6 dígitos y con una ventana de tiempo de 30 segundos y una validez de 2 minutos más por cada OTP generado para así evitar problemas de caducidad por posibles retrasos en la entrega del código, como se vio en capítulos anteriores.

- Los códigos son generados mediante el protocolo TOTP, por lo cual se aplica el protocolo HOTP sobre el sello de tiempo actual o "*timestamp*".

- Los códigos son generados a partir de la credencial del usuario, la cual es la combinación del email con la contraseña utilizando encriptación bcrypt.

- Estos códigos son almacenados en la base de datos “otps” sobre MongoDB.

6.4.1 Diagrama de flujo del programa

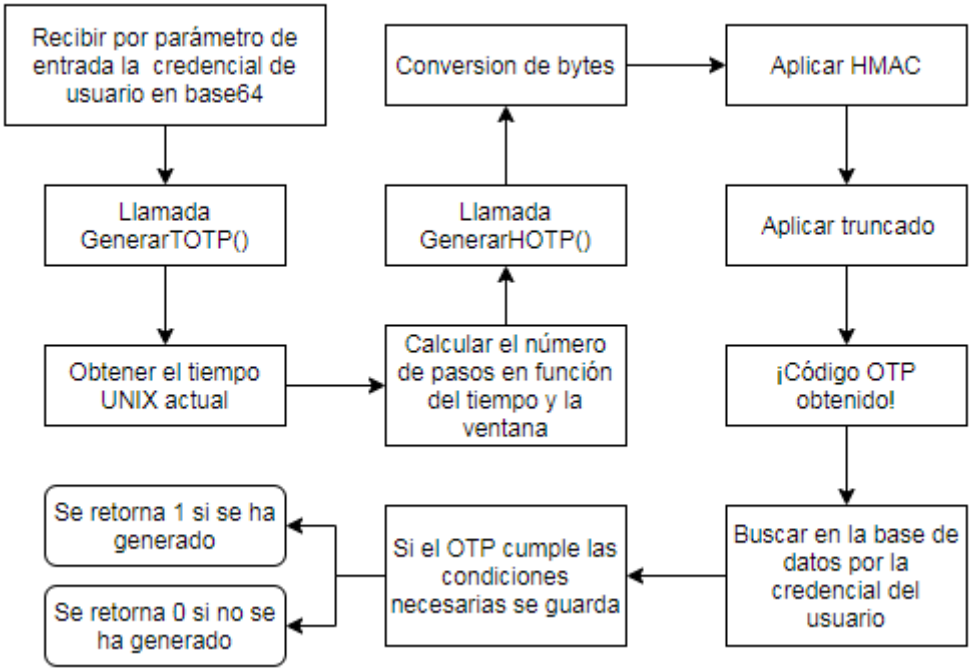


Ilustración 14 – Diagrama de flujo de Python para la generación de OTPs

6.4.2 Función Main()

Esta función cuando es llamada tendrá que recibir un parámetro de entrada que será la credencial del usuario (recordar que es el correo electrónico concatenado al hash encriptado de la contraseña mediante bcrypt y todo esto convertido a base64).

Tras la conexión a la base de datos MongoDB, se comprueba que existe un código ya asociado a esta credencial (ya que por defecto al dar de alta un usuario en la aplicación, se le asigna ya un código OTP en esta colección que por defecto es 000000 y está usado, es decir, tanto el indicador de uso está a verdadero y el campo fecha ya cumplida, por lo cual sin validez)

Para generar el código se llama a la función TOTP, que esta a su vez llamará a la función HOTP. Finalmente se obtiene el código de 6 dígitos.

Para que un OTP sea vuelto a ser actualizado en la base de dato se tiene que cumplir que:

- El código OTP sea el mismo y el indicador de uso esté a falso (es decir, el usuario a regenerado el mismo OTP dentro de su misma ventana de tiempo de validez)
- El OTP es distinto y el campo de validez de fecha ha expirado (es decir, es uno nuevo generado)

Si ambos se cumplen, el OTP generado se actualiza en la base de datos. Además si se da esta condición, se enviará por SMS el código generado usando Twilio.

El código fuente de la función main() puede verse en las Illustraciones 15 y 16.

```
59  if __name__ == '__main__':
60      # Si no hay dos parametros minimo, error, salir
61      if len(sys.argv) == 1:
62          print "0"
63          sys.exit()
64      # La funcion recibira un parametro que sera la credencial
65      #  del usuario -> email:password(bcrypt) convertido a base64
66      credential = sys.argv[1]
67      # Generar codigo OTP
68      otpcode = GenerarTOTP(credential)
69      # Conexion a la base de datos mongoDB y almacenarlo
70      # Abrir una nueva conexion a una base de datos MongoDB
71      client = pymongo.MongoClient('mongodb://localhost:27017')
72      db = client['otpproject']  # Base de datos 'otpproject'
73      otpdb = db['otps']  # Seleccionar la collecion 'otps'
74      userdb = db['usuarios']  # Seleccionar la collecion 'users'
75
76      # Buscar un documento por su credenial
77      otp_doc = otpdb.find_one({'credential': credential})
78
79      # Chequear si encuentra algun registro
80      if otp_doc == None:
81          print "0" #Error
82      else:
83          # Comprobar que el otp actual no es el mismo y no esta usado (para no reactivarlo) o sigue activo
84          if (otp_doc['otpcode'] == otpcode and otp_doc['used'] == False)
85              or (otp_doc['otpcode'] != otpcode and datetime.datetime.utcnow() > otp_doc['valid']):
86
87              # Calcular la validez temporal del OTP. Sera valido durante sus 30 segundos de ventana
88              # aparte de 120 segundos mas por desfase en el de envio de SMS/correo electronico
89              valid = datetime.datetime.utcnow() + datetime.timedelta(seconds=120)
90
91              # Actualizar el documento de mongoDB asignando el nuevo otp generado y su validez
92              otpdb.update({'_id': otp_doc['_id'], 'credential':otp_doc['credential'] },{"$set": {
93                  'otpcode' : otpcode,
94                  'used': False,
95                  'valid': valid}})
```

Ilustración 15 – Código de la función main de Python – Parte 1

```
90            # Actualizar el documento de mongoDB asignando el nuevo otp generado y su validez
91            otpdb.update({'_id': otp_doc['_id'], 'credential':otp_doc['credential'] },{"$set": {
92                'otpcode' : otpcode,
93                'used': False,
94                'valid': valid}})
95
96            # Buscar telefono del usuario
97            user_doc = userdb.find_one({'credential': credential})
98            phone = "+34"+str(user_doc['phone'])
99            # Enviar por SMS usando Twilio
100           account_sid = 'AC13b4129c5d[redacted]'
101           auth_token = 'a2865b9e32b2d[redacted]'
102           twclient = Client(account_sid, auth_token)
103
104           message = twclient.messages \
105                           .create(
106                               body="El codigo OTP para la aplicacion TFM OTP Project es "+otpcode+". Su validez es h
107                               from_='+34955160[redacted]',
108                               to=phone
109                           )
110           print "1" # Ok
111       else:
112           print "0" # Error
113   client.close() # Cerrar la conexion a la base de datos MongoDB
```

Ilustración 16 – Código de la función main de Python – Parte 2

6.4.3 Función GenerarTOTP()

Esta función recibe como parámetro la clave, y como parámetro opcional la ventana de tiempo, que si no se indica por defecto es de 30 segundos.

En esta función se obtiene primeramente el tiempo actual UNIX y se calcula el número de pasos en el tiempo mediante la ventana de tiempo.

Finalmente se invoca a la función GenerarHOTP pasándole como parámetro la clave y el número de pasos del tiempo.

El código fuente de la función GenerarTOTP se puede ver en la Ilustración 17.

```
38  def GenerarTOTP(clave, paso_tiempo=30):
39      '''
40          TOTP es una variante de HOTP basado en el tiempo. Solamente requiere el
41          parametro de la clave, siendo el paso de tiempo un parametro opcional.
42          El paso de tiempo permitira mantener el mismo valor para el contador
43          basado en el tiempo segun su valor, es decir, por defecto es 30seg
44          por lo que el numero de pasos se mantendra igual y el codigo OTP que
45          se generara sera el mismo tambien durante ese periodo de tiempo.
46          Esta funcion retorna un codigo de numero enteros de longitud 6
47      '''
48      #print "- - Funcion TOTP"
49
50      # Se calcula el numero de pasos desde el comienzo del tiempo UNIX
51      tiempo_unix_actual = time.time()
52      #print "     El tiempo UNIX actual es "+str(tiempo_unix_actual)
53      N = long(math.floor(tiempo_unix_actual / paso_tiempo))
54      #print "     El numero de pasos N es "+str(N)
55      return GenerarHOTP(clave, N)
```

Ilustración 17 – Código de la función GenerarTOTP de Python

6.4.4 Función GenerarHOTP()

Esta función recibe como parámetros la clave y el número de pasos de tiempo a modo de contador.

Primeramente hay que convertir el contador a un valor hexadecimal de 8 bytes y esto será el mensaje para la función criptográfica HMAC-SHA1.

Finalmente hay que realizar un truncado del valor obtenido para finalmente obtener los últimos 6 dígitos.

Este será el código OTP generado.

El código fuente de la función GenerarHOTP se puede ver en la Ilustración 18.

```
9   def GenerarHOTP(clave, contador):
10      '''
11          Este algoritmo HOTP genera un codigo de numeros enteros de longitud 6.
12          Requiere como parametros una clave compartida y un contador.
13          Se basa en el RFC4226 de la IETF.
14      '''
15      #print "- - Funcion HOTP"
16
17      # Primeramente hay que convertir el contador a un valor
18      #  hexadecimal de 8 bytes (16 cararteres), sino tiene esa longitud habra
19      #  que rellenar con ceros a la izquierda. Esto sera el "message" de
20      #  la funcion hmac_sha1.
21      # Convertir el long a un array de bytes
22      cont_bytes = array.array('B')
23      for i in reversed(range(0, 8)):
24          cont_bytes.insert(0, contador & 0xff)
25          contador >>= 8
26
27      # Llamada a la funcion criptografica hmac_sha1
28      hmac_sha1 = hmac.new(key=clave, msg=cont_bytes, digestmod=hashlib.sha1).hexdigest()
29      #print "     HMAC-SHA1 antes de truncar: "+hmac_sha1
30
31      # Hay que realizar un truncado para la conversion de un valor HMAC-SHA-1 en
32      #  un valor HOTP tal y como se define en la seccion 5.3 del RFC4226 de la IETF
33      offset = int(hmac_sha1[-1], 16)
34      hmac_sha1_truncado = int(hmac_sha1[(offset * 2):((offset * 2) + 8)], 16) & 0x7fffffff
35      #print "     HMAC-SHA1 despues de truncar: "+str(hmac_sha1_truncado)
36      return str(hmac_sha1_truncado) [-6:]
```

Ilustración 18 - Código de la función GenerarHOTP de Python

6.4.5 Twilio, plataforma para envío de SMS

Entre las diversas formas que un usuario tiene de recibir su código de un solo uso OTP, las más usuales son mediante correo electrónico, vi mensaje SMS o mediante una aplicación móvil. En capítulos anteriores ya se han visto las ventajas y desventajas de cada una de ellas, sin embargo para realizar esta parte práctica se ha optado por buscar algún servicio que ofrezca el envío de mensajes SMS, con además la

condición de que tenga una API para poder utilizarla desde nuestro código.

Tras ver diversas opciones se ha decidido que una opción correcta para abordar este reto podría ser Twilio.

Twilio es [48] una plataforma de desarrollo que permite a los desarrolladores construir aplicaciones de comunicación en la nube y sistemas web. Ha tomado la red global de telecomunicaciones y proporcionado API de comunicaciones en la nube, que se integran con sus aplicaciones y sistemas web para comunicarse a nivel mundial.

Una vez se ha efectuado el registro en su plataforma, Twilio , como se puede observar en la Ilustración 19, obsequia a las cuentas de prueba con un saldo inicial de 11.50 dólares para utilizar el servicio haciendo pruebas, sin tener que pagar nada antes ni después. Esto supondrá un servicio perfecto para las pruebas que se quieren realizar en esta práctica final.

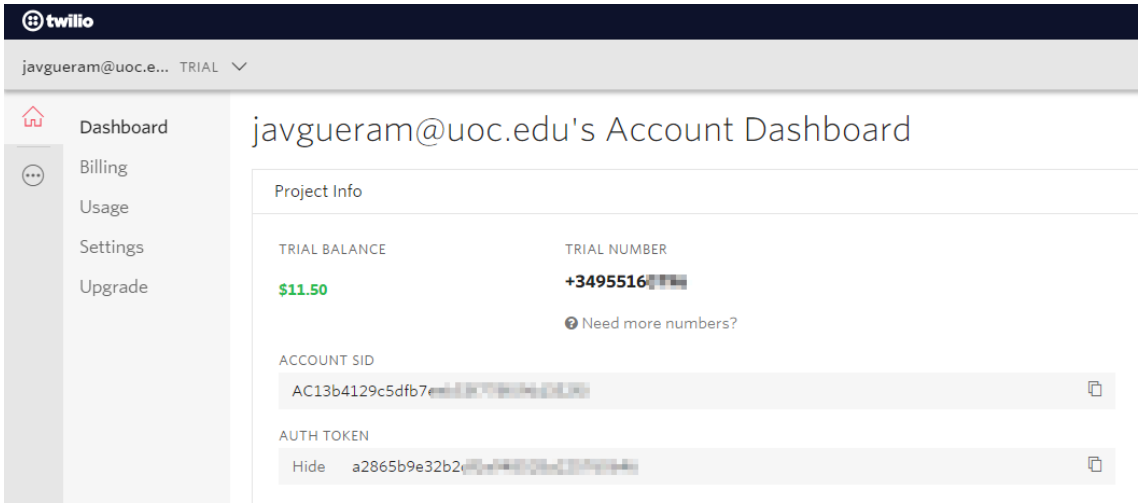


Ilustración 19 – Panel de control de la cuenta de Twilio

A su vez, Twilio, a la cuenta de prueba permite obtener un número de teléfono asociado desde el cual se enviarán los mensajes SMS. Para este caso práctico ha asignado el +34955160XXX.

Una vez se tiene acceso al panel de usuario de Twilio, tan sólo es necesario obtener los siguientes valores para integrarlos después en la API:

- Account SID: identificador en twilio
- Auth token: token de autenticación
- Trial Number: número de teléfono de pruebas asignado

Tras las primeras pruebas, se ha estimado que con la cuenta obtenida se van a poder enviar hasta 138 mensajes SMS, lo cual da de sobra para el acometido de esta práctica.

6.5 Vistas

La aplicación web se compone de tres vistas: La página de inicio de sesión o "*login*", la página de registro de nuevo usuario o "*register*" y la página del perfil de usuario o "*profile*".

6.5.1 Login

La pantalla de login permite al usuario realizar la autenticación utilizando su correo electrónico y su contraseña. Además dispone de un botón para generar el código OTP y recibirlo mediante SMS, y otro botón para enviar la información al servidor.

En la Ilustración 20 se muestra la pantalla de login.

Ilustración 20 – Vista de la pantalla de login

6.5.2 Register

La pantalla de registro permite al usuario darse de alta en el sistema. Tan sólo tiene que introducir un correo electrónico, una contraseña y un número de teléfono (en el cual recibirá el código de un solo uso al autenticarse).

En la Ilustración 21 se muestra la pantalla de registro.

Ilustración 21 – Vista de la pantalla register

6.5.3 Profile

La pantalla de perfil de usuario tan sólo muestra la información del usuario que está logueado. No tiene ninguna finalidad más que indicar que hay una sesión abierta. Pulsando el botón de salir se puede cerrar la sesión.

En la Ilustración 22 se muestra la pantalla de perfil de usuario

Ilustración 22 – Vista de la pantalla de perfil de usuario

6.6 Rutas

La aplicación dispone de distintas rutas que gracias a la librería Express permite realizar la navegación por éstas.

6.6.1 Raíz o /

La ruta raíz no tiene ninguna vista asociada. Tan solo comprueba si hay una sesión de usuario abierta. Si es así redirige a la ruta /profile y sino, redirige a la página de /login.

6.6.2 /login

Esta ruta muestra la vista de login antes descrita. Permite tanto petición GET como POST.

6.6.3 /register

Esta ruta muestra la vista de reigster antes descrita. Permite tanto petición GET como POST.

6.6.4 /generateotp

Esta ruta permite la generación de un nuevo código de un solo uso. Es llamada cuando el usuario rellena su correo electrónico y su contraseña en la pantalla de login y pulsa el botón “generar OTP”. Internamente hace una llamada al script de Python “generateOTP.py” el cual genera el código, lo actualiza en su base de datos correspondiente y lo envía por SMS al usuario.

6.6.5 /profile

Esta ruta muestra la vista de profile antes descrita.

6.6.6 /logout

Esta ruta realiza un cierre de sesión del usuario activo y retorna a la ruta raíz (/).

7. Conclusiones

Durante el transcurso de este trabajo se ha profundizado en la problemática actual de los sistemas de autenticación de un único factor que relegan todo el peso en el componente de la contraseña. Tal y como se ha podido contemplar en este trabajo, el uso de más de un factor de autenticación permite mantener la integridad de los datos de una forma más segura si uno de los factores falla (descuido humano, ataque, etc).

Al finalizar el trabajo con la práctica final quedan alcanzados los objetivos planteados al inicio del proyecto, permitiendo probar un sistema de autenticación de doble factor mediante códigos de un solo uso. Además se ha creado un sistema que genera los códigos utilizando lenguaje Python y siguiendo el protocolo descrito por la IETF para los códigos TOTP.

Respecto a la planificación propuesta al principio del proyecto ha permitido perfectamente la realización del trabajo en el tiempo establecido.

Las posibles líneas futuras del presente trabajo serían fundamentalmente aquellas que cubriesen las carencias y posibles vulnerabilidades del sistema desarrollado hasta ahora. Se podría mejorar la obtención del código OTP utilizando por ejemplo una aplicación móvil, así no hay que depender del saldo disponible para enviar SMS y para un sistema real reduciría el coste de uso. Finalmente, dado que el objetivo es mejorar la seguridad de acceso, debería hacerse un análisis sobre la fortaleza del sistema aplicado pudiendo sacar posibles vulnerabilidades y mejoras.

4. Glosario

2FA: Two-Factor Authentication

HOTP: HMAC based One-Time Password

IETF: Internet Engineering Task Force

MFA: Multi-Factor Authentication

OTP: One-Time Password

RBAC: Role-Based Access Control

RFC: Requests For Comments

Spoofing: Suplantación de identidad

SS7: Signaling System Number 7

TAN: Transaction Authentication Numbers

Timestamp: Sello de tiempo

TOTP: Time based One-Time Password

5. Bibliografía

[1] La evolución del concepto de seguridad - Instituto Español de Estudios Estratégicos. http://www.ieee.es/Galerias/fichero/docs_marco/2011/DIEEEM05-2011EvolucionConceptoSeguridad.pdf. Última visita: 20-Marzo-2019.

[2] El cifrado por desplazamiento de César - Wikipedia. https://es.wikipedia.org/wiki/Cifrado_C%C3%A9sar. Última visita 21-Marzo-2019

[3] Máquina Enigma - Wikipedia. https://es.wikipedia.org/wiki/Enigma_(m%C3%A1quina). Última visita: 21-Marzo-2019.

[4] Evolución de la seguridad en la empresa - INCIBE. https://www.incibe.es/protege-tu-empresa/blog/seguridad-desde-inicio. Última visita: 21 - Marzo-2019.

[5] Autenticación - Wikipedia. https://es.wikipedia.org/wiki/Autenticaci%C3%B3n. Última visita: 20-Marzo-2019.

[6] Módulo 1 - Introducción a la criptografía - Apuntes UOC de asignatura Criptografía. P05/75024/00985. Josep Domingo Ferrer.

[7] Introducción a la criptografía - Universidad Complutense de Madrid. https://www.ucm.es/data/cont/docs/72-2016-11-10-2%20Comunicaciones%20seguras.pdf. Última visita: 23-Marzo-2019.

[8] Identificación y autenticación - IBM. https://www.ibm.com/support/knowledgecenter/es/SSFKSJ_7.5.0/com.ibm.mq.sec.doc/q009740_.htm. Última visita: 23-Marzo-2019.

[9] Introducción a la autenticación - Welivesecurity by ESET. https://www.welivesecurity.com/la-es/2016/05/04/autenticacion-como-probar-que-eres-tu/. Última visita: 24-Marzo-2019

[10] One-Time Password - Vasco Data Security. https://www.vasco.com/es-es/glossary/one-time-password.html. Última visita: 24-Marzo-2019

[11] 4 de mayo, Día Mundial de las contraseñas - Oficina de Seguridad del Internauta OSI. https://www.osi.es/es/actualidad/blog/2017/05/03/4-de-mayo-dia-mundial-de-las-contrasenas. Última visita: 24-Marzo-2019

[12] Día de la contraseña, una breve historia de su origen. Welivesecurity by ESET. https://www.welivesecurity.com/la-es/2017/05/04/dia-de-la-contrasena-origen/. Última visita: 24-Marzo-2019

[13] Role-based Access Control - Wikipedia. https://www.welivesecurity.com/la-es/2017/05/04/dia-de-la-contrasena-origen/. Última visita: 24-Marzo-2019

[14] The worst passwords of 2018 - SplashData. https://www.teamsid.com/100-worst-passwords-top-50/. Última visita: 25-Marzo-2018

[15] Contraseña - RAE. https://dle.rae.es/?id=Ad1mU8W. Última visita: 25-Marzo-2019

[16] Típicos errores que cometemos al usar nuestras contraseñas - Oficina de Seguridad del Internauta. https://www.osi.es/es/actualidad/blog/2019/02/20/tipicos-errores-que-cometemos-al-usar-nuestras-contrasenas-y-como. Última visita: 25-Marzo-2019

[17] Think your password will protect your data? Think again - Deloitte. https://www2.deloitte.com/ca/en/pages/press-releases/articles/think-your-password-will-protect-your-data-think-again.html. Última visita: 25-Marzo-2019

[18] 10 trucos para contraseñas seguras y fáciles de recordar - Panda Security. https://www.pandasecurity.com/spain/mediacenter/seguridad/10-trucos-para-crear-contrasenas-seguras/. Última visita: 25-Marzo-2019

[19] Generador de contraseñas - LastPass. https://www.lastpass.com/es/password-generator. Última visita: 25-Marzo-2019

[20] Electronic Authentication Guidelines - NIST. Página 20. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf

[21] Factores de autenticación - Medium. https://medium.com/@marvin.soto/factores-de-autenticaci%C3%B3n-5d8a886c6b4d. Última visita: 26-Marzo-2019

[22] NIST declares the age of SMS-based 2-factor authentication over - Tech Crunch. https://techcrunch.com/2016/07/25/nist-declares-the-age-of-sms-based-2-factor-authentication-over/?guccounter=1&guce_referrer_us=aHR0cHM6Ly93d3cuZ2VuYmV0YS5jb20vc2VndXJpZGFkL3Bvci1xdWUtbm8tZGViZXMtY29uZmlhci1lbi1sYS1hdXRlbnRpY2FjaW9uLWVuLWRvcy1wYXNvcy1hLXRyYXZlcy1kZS1zbXM&guce_referrer_cs=YUacyx6ptHvP-_KX47BX1g. Última visita: 26-Marzo-2019

[23] Chaos Computer Club hackt AppleID - Chaos Computer Club CCC. https://www.ccc.de/updates/2013/ccc-breaks-apple-touchid. Última Visita: 26-Marzo-2019

[24] Módulo 6 - Seguridad en los sistemas biométricos - Apuntes UOC de asignatura Biometría. PID_00200719. Albert Solé Ribalta.

[25] Hackers can read your private SMS and listen to phone calls - The Hacker News. https://thehackernews.com/2014/12/hackers-can-read-your-private-sms-and.html. Última visita: 26-Marzo-2019

[26] Ataque de replay - Wikipedia. https://es.wikipedia.org/wiki/Ataque_de_REPLAY. Última visita: 26-Marzo-2019

[27] One-Time Password - Wikipedia. https://en.wikipedia.org/wiki/One-time_password. Última visita: 26-Marzo-2019

[28] The failure of Two-Factor Authentication - Schneier on Security - https://www.schneier.com/blog/archives/2005/03/the_failure_of.html. Última vistia: 27-Marzo-2019

[29] Ataque de intermediario - Wikipedia. https://es.wikipedia.org/wiki/Ataque_de_intermediario. Última visita: 27-Marzo-2019

[30] Troyano - Wikipedia. https://es.wikipedia.org/wiki/Troyano_(inform%C3%A1tica). Última visita: 27-Marzo-2019

[31] Biometría para identificación y autenticación - Gemalto. https://www.gemalto.com/latam/sector-publico/inspiracion/biometria. Última visita: 28-Marzo-2019

[32] HOTP frente a OTP - Microcosm. https://es.microcosm.com/blog/hotp-totp-what-is-the-difference. Última visita: 28-Marzo-2019

[33] IETF - RFC-4226.https://www.ietf.org/rfc/rfc4226.txt. Última visita: 25-Abril-2019

[34] IETF - RFC-6238. https://tools.ietf.org/html/rfc6238. Última visita: 27-abril-2019

[35] Wikipedia - Requests For Comments. https://es.wikipedia.org/wiki/Request_for_Comments. Última visita: 18-Abril-2019

[36] Wikipedia - Endianness. https://es.wikipedia.org/wiki/Endianness. Última visita: 18-Abril-2019

[37] Security Google Blog. Announcing the first SHA1 collision. https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html. Última visita: 18-Abril-2019

[38] IETF - RFC 2289. https://tools.ietf.org/html/rfc2289. Última visita: 27-Abril-2019

[39] Wikipedia - Latch. https://es.wikipedia.org/wiki/Latch_(aplicaci%C3%B3n). Última visita: 28-Abril-2019

[40] ElevenPath Blog. Latch Voice: Generación Cript-biométrica del OTP. https://blog.elevenpaths.com/2017/10/latch-voice-generacion-cripto.html. Última visita: 28-abril-2019

[41] Wikipedia - Google authenticator. https://es.wikipedia.org/wiki/Google_Authenticator. Última visita: 28-Abril-2019

[42] Authy. https://authy.com/. Último acceso: 28-Abril-2019

[43] Wikipedia – Python. https://es.wikipedia.org/wiki/Python. Última visita: 01-Junio-2019

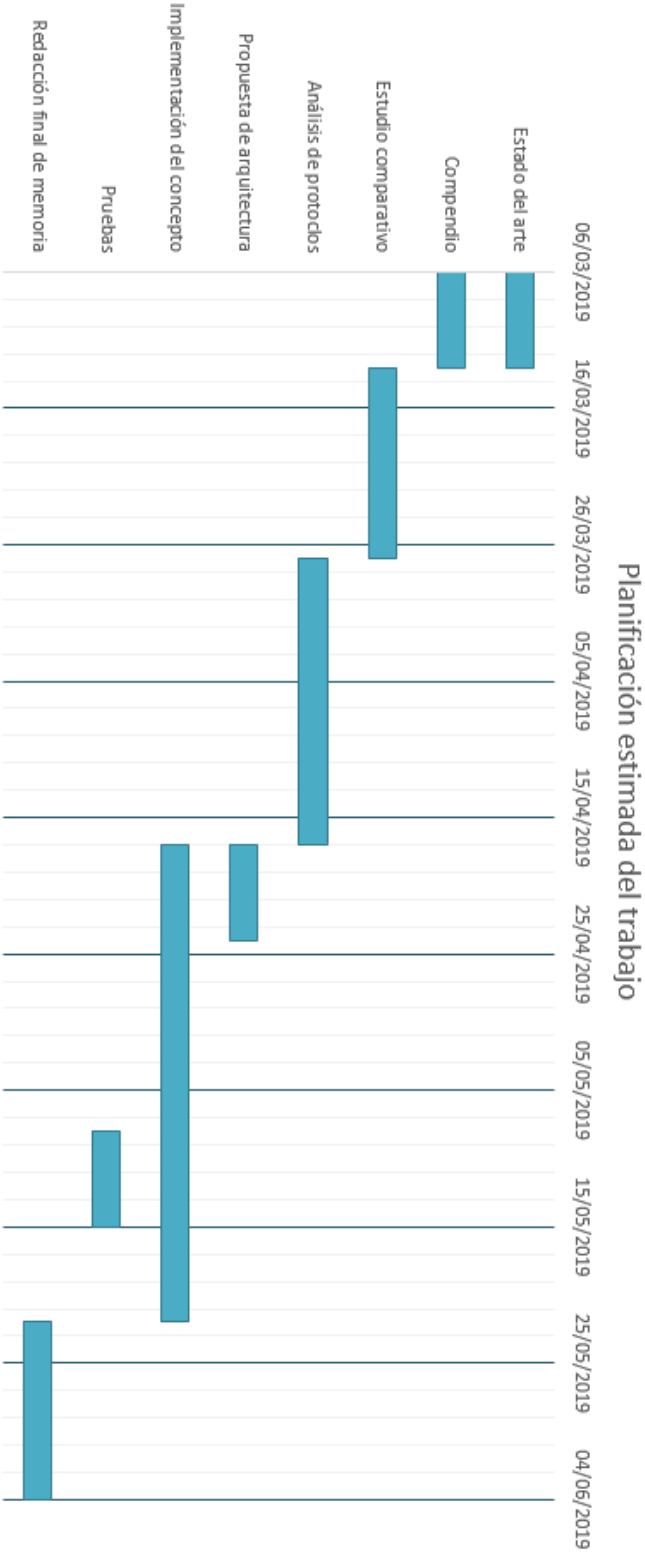
[44] Wikipedia – MongoDB. https://es.wikipedia.org/wiki/MongoDB. Última visita: 01-Junio-2019

[45] Wikipedia – NodeJS. https://es.wikipedia.org/wiki/Node.js. Última visita: 02-Junio-2019

[46] Mozilla Developer - Express NodeJS. https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction. Última visita: 02-Junio-2019

[47] Passport. https://github.com/jaredhanson/passport. Última visita: 02-Junio-2019

[48] Twilio. https://www.twilio.com/. Última visita: 02-Junio-2019

6. Anexos

ANEXO 1 – DIAGRAMA DE GANT PLANIFICACIÓN ESTIMADA DEL TRABAJO