Introducción.

Capítulo I:

Planteamiento del problema.

Actualmente, una gran cantidad de personas hacen uso del internet y de las nuevas tecnologías para comunicarse. Con ello, también se incrementa la cantidad de información que se transmite y/o almacena. En diversas ocasiones, esta información es susceptible a sufrir distintos tipos de ataques, tales como acceso no autorizado, modificación o destrucción de la misma, entre otros. Adicionalmente, cada día aparecen nuevos tipos de ataques a los sistemas de información. Por lo tanto, surge la necesidad de proteger dicha información.

Una de las tecnologías ampliamente usada para comunicarse es el correo electrónico [1]. Los mensajes que envían y reciben los usuarios de correo electrónico pueden ser de diferentes tipos: personales, transaccionales, de notificación o de publicidad.  Por lo tanto, cada vez que se escribe y envía un correo electrónico, se está revelando información acerca de las preferencias y/o intereses del usuario. Estos datos, son el insumo más importante, para distintas entidades, entre las cuales están empresas que realizan publicidad en línea, proveedores de internet, instituciones de gobierno, entre otros [3].  El propósito de tener estos datos puede ser realizar publicidad efectiva, vender los datos a empresas de publicidad o averiguar si determinado usuario es una amenaza para el gobierno. Para obtener información acerca de los intereses y/o preferencias del usuario, se hace uso de programas de cómputo denominados clasificadores. Los clasificadores son herramientas informáticas que analizan una gran cantidad de información, haciendo uso de técnicas de aprendizaje maquina [4], y posteriormente  clasifican un mensaje en determinada categoría o perfil.  En este contexto, los clasificadores pueden constituir una amenaza para algunos usuarios del correo electrónico, por tal motivo de ahora en adelante a los programas que clasifican se les denominará adversarios clasificadores.

Ante tal escenario, surge la pregunta ¿cómo se puede proteger un usuario contra los adversarios clasificadores? Una posible respuesta es hacer uso de algoritmos de cifrado estándar. Sin embargo, hacer uso de tales algoritmos, implica que los participantes en la comunicación acuerden una clave de cifrado. Desafortunadamente, acordar una clave, no es un proceso sencillo para el usuario común. Otra desventaja de esta primera solución, es que los algoritmos de cifrado estándar ofrecen un alto nivel de seguridad, el cual resulta excesivo cuando se considera los recursos y el objetivo de un adversario clasificador [5].

Lo que se pretende en este trabajo terminal, es ofrecer al usuario una solución alternativa para proteger al correo electrónico, contra adversarios clasificadores. Dicha solución se considera más fácil de usar y además ofrece un nivel de seguridad adecuado, teniendo en cuenta las características de dichos adversarios. La solución que se propone, hará uso de CAPTCHAs y de un algoritmo criptográfico, conocido como secreto compartido. Los CAPTCHAs son programas de cómputo, cuyo  propósito es distinguir si están interactuando con una máquina o  con un ser humano.

El secreto compartido es una técnica criptográfica, que permite dividir un secreto en “n” partes, este secreto puede ser por ejemplo un CAPTCHAs, finalmente las “n” partes se dividen entre diferentes entidades. Para poder recuperar el secreto no es necesario recuperar las “n” partes en las que se dividió el secreto, sólo es necesario recuperar un  número mínimo de partes para deducir el secreto, este número mínimo debe ser menor o igual al número de partes en que se dividió el secreto y está definido por la implementación de la técnica criptográfica; en caso de no recuperar el número mínimo de partes del secreto la técnica nos dicta que no es posible recuperar el secreto.

Objetivos.

Desarrollar una herramienta para un cliente de correo electrónico que permita cifrar el contenido de los correos para evitar su clasificación, basándonos en la técnica de criptográfica de secreto compartido y asegurando el envío y recepción de los CAPTCHAs a través de un repositorio, basándonos en un esquema de cifrado por identidad.

2.1 Objetivos Específicos

* Desarrollar un algoritmo de cifrado y descifrado basado en secreto compartido.
* Desarrollar una herramienta en un cliente de correo electrónico para el envío y recepción de los correos cifrados y la generación, envío y recepción de CAPTCHAS.
* Desarrollar un servidor de llaves que reciba, aloje y envíe los CAPTCHAS a los usuarios para descifrar los correos electrónicos.

Justificación.

Lo que se pretende en este trabajo terminal, es ofrecer al usuario una solución alternativa para proteger al correo electrónico, contra los clasificadores. Dicha solución se considera más fácil de usar y además ofrece un nivel de seguridad adecuado, teniendo en cuenta las características de los clasificadores. La solución que se propone, hará uso de CAPTCHAs y de un algoritmo criptográfico, conocido como *secreto compartido.* Los CAPTCHAs son programas de cómputo, cuyo  propósito es distinguir si están interactuando con una máquina o  con un ser humano. El secreto compartido es una técnica criptográfica, que permite dividir un secreto *K* en *w* partes, de tal manera que si se tienen al menos *u* de *w*, donde *u<=w*, es posible recuperar *K*. Si se tienen menos de *u* partes no es posible recuperar *K* [11]*.*

Este algoritmo criptográfico tiene una gran ventaja al momento de la generación de llaves, ya que sus llaves son simétricas a diferencia de muchos algoritmos estándares que utilizan una generación de un par de llaves asimétricas, estos algoritmos criptográficos asimétricos nos llevan a necesitar una comunicación previa entre el emisor y el receptor del mensaje enviado. Mientras que al usar secreto compartido solo es necesario que el emisor genere una llave, la divida y envíe los CAPTCHAs, y que el receptor resuelva los CAPTCHAs, recupere la llave y posteriormente el mensaje de correo electrónico.

El envío de los correos se propone hacerlo enviando el correo electrónico con el contenido cifrado a un servidor de correo electrónico donde se alojará hasta que el cliente de correos electrónico instalado en el equipo de cómputo del receptor lo descargue por medio del protocolo POP3, al mismo tiempo se enviarán los CAPTCHAs que permiten recuperar la llave de cifrado a un tercer agente llamado servidor de llaves, el cuál verificara que la petición hecha por el receptor es válida y que el receptor es un usuario valido. El protocolo SSL se usará para garantizar el envío de los CAPTCHAs  por parte del emisor, la correcta recepción por parte el servidor de llaves y la recuperación por parte del receptor para la correcta recuperación de la llave de cifrado y posterior descifrado de los correos electrónicos. El servidor de llaves es fundamental para el funcionamiento de este sistema, ya que esto le da un nivel de seguridad un poco más elevado que si lo enviáramos todo en un sólo paquete.

Capítulo II:

Antecedentes.

La única referencia que tenemos sobre un esquema criptográfico contra adversarios clasificadores es el que encontramos en el artículo “Defending Email Communication

Against Profiling Attacks” de Philippe Golle y Ayman Farahat, ambos miembros del “Palo Alto Research Center” [12].

En su artículo aborda el tema del ataque de los adversarios clasificadores proponiendo un esquema de cifrado simple de llave simétrica en el cual la llave que se usa para cifrar está dada por la cabecera del correo electrónico (que consiste en la dirección del remitente, la dirección del destinatario, la hora a la que se envía el correo electrónico y potencialmente otros campos).

Esto le permite al destinatario calcular la llave fácilmente ya que estos datos son públicos, este esquema es inseguro contra un adversario normal, pero es seguro contra adversarios clasificadores, ya que los clasificadores al procesar la información por volumen si quisieran descifrar el contenido del correo tardarían en descifrar todos por la gran cantidad de correos que procesan en cada lote.

Tomando en cuenta que esta es la única referencia sobre el tema que estamos abordando, podemos decir que trabajamos en el mismo sentido ya que esta propuesta de solución está basada en el artículo “On Securing Communication from Profilers” en el que se propone algo similar en cuanto al cifrado, nos lleva a tomar muy en cuenta el poder del algoritmo de cifrado que usaremos en nuestra propuesta de solución.

Marco teórico.

En el siguiente capítulos hablaremos sobre el correo electrónico y algunos de los intermediarios que hacen posible que este servicio sea usado en toda la red de internet, también hablaremos de las amenazas que a las que se tiene que enfrentar este servicio para hacer llegar información de un usuario a otro de una manera segura y cuales han sido las respuestas de los proveedores de servicio de correo electrónico para proporcionar dicha seguridad a los usuarios.

1.-Correo electrónico.

El correo electrónico es el servicio de internet por el cual se pueden enviar mensajes entre 2 usuarios que cuenten con este servicio. El correo electrónico o “e-mail” por sus siglas en inglés, basa su funcionamiento en las oficinas postales. Tomando esa analogía podemos decir que los usuarios tienen cartas (mensajes de correo electrónico) que desean enviar a otros usuarios; estas cartas son enviadas a las oficinas postales (servidores de correo electrónico) donde se almacenan con otras cartas que van dirigidas a otros usuarios en diferentes ciudades; estas oficinas postales envían las cartas a otras oficinas postales si es necesario; y por último cada oficina postal se encargan dejan en un buzón asignado a un usuario (buzón de correos electrónicos) las cartas que otros usuarios le han enviado.

2.- Mensaje de correo electrónico.

Los mensajes de correo electrónico al igual que las cartas tienen la dirección del receptor, la dirección del emisor y el cuerpo del mensaje, pero a diferencia de las cartas los correos electrónicos son enviados por internet y necesitan sus propios formatos.

Para poder enviar un mensaje de correo electrónico es necesario tener los elementos como mínimo:

Dirección del remitente: Esta dirección se compone por dos elementos importantes, el primero es el nombre de usuario seguido de un carácter [1] “@”, el segundo elemento es el dominio donde está alojado el servicio de correo electrónico.

Direcciones de los receptores: En un correo electrónico debe haber al menos una dirección de receptor para ser enviado, esta dirección de receptor tiene la misma estructura que la dirección del remitente.

Contenido del mensaje: Es el texto que se desea transmitir entre el remitente y el receptor.

El mensaje de correo electrónico cuenta con más elementos pero estos son opcionales y se pueden consultar en el RFC 2821 extensión MIME.

3.- Dominio de internet.

Es un nombre que identifica a los diferentes dispositivos interconectados en una red sin la necesidad de aprenderse un número de red. Estos dispositivos pueden proporcionar diferentes servicios como el servicio de correo electrónico.

4.- Buzón de correo electrónico.

Un buzón de correo electrónico es un espacio virtual proporcionado por el servicio de correo electrónico que sirve para almacenar los mensajes enviados y recibidos.

5.- Usuario de correo electrónico.

Se le conoce como usuario de correo electrónico a la persona que se registra en un dominio para obtener los servicios de mensajería proporcionados por un servidor de correo electrónico.

6.- Cliente de correo electrónico.

El cliente de correo electrónico es una interfaz por la cual el usuario de correo electrónico puede administrar sus mensajes enviados y recibidos.

El cliente de correo electrónico puede ser de diferentes tipos como son:

1. Cliente de correo electrónico para la pc.

Estos clientes de correo electrónico son instalados en una computadora y es configurado para sincronizarse con un servidor de correo electrónico cada cierto tiempo o cuando el usuario se lo indique.

Una de las principales características de éste cliente de correos es la capacidad de descargar los mensajes del servidor a la computadora para ser leídos sin la necesidad de tener una conexión de internet y en cuanto tiene conexión con el servidor otra vez descarga los nuevos mensajes y envía los que se tienen pendientes en la computadora.

1. Cliente de correo electrónico web.

Los clientes web necesitan un explorador de internet y una conexión permanente a la red para que funcione, estos clientes normalmente son proporcionados por el mismo servidor de correo electrónico y nunca descarga los correo en los ordenadores donde son utilizados.

1. Cliente de correo electrónico para móviles.

Un cliente de correo móvil se caracteriza por ser aplicaciones instaladas en los dispositivos móviles, se sincronizan con el servidor continuamente pero no descargan ningún mensaje de correo pero podemos tener cargados los últimos mensajes temporalmente.

7.- Servidor de Correo electrónico.

Un servidor de correo electrónico es un programa que se encarga de enviar y recibir los mensajes de correo electrónicos de sus usuarios registrados, este servidor puede recibir mensajes de usuarios de otros servidores de correos que sean dirigidos a sus usuarios registrados.

Este servidor tiene que seguir algunos estándares que existen en internet para él envió de mensajes de correo electrónico (protocolo smtp) y recepción de mensajes de correo electrónico (protocolo pop3 o imap).

8.- Protocolo SMTP

El protocolo SMTP significa “protocolo para transferencia simple de correo” o “Simple Mail Transfer Protocol” por sus siglas en inglés, el cual se encarga de enviar los mensajes de correo electrónicos entre dispositivos que se encuentran interconectados en la red o en internet. Este protocolo solo se utiliza para mandar los mensajes entre servidores o entre el usuario emisor y su servidor de correo electrónico. Podemos revisar el protocolo completo en el RFC5321 [15].

9.- Protocolo POP3.

Este protocolo se encarga de descargar los mensajes del servidor a un cliente de correo electrónico que el usuario haya configurado previamente. Una de las características es que  solo se sincroniza para la descarga de los mensajes de correo y no deja una copia de seguridad en el servidor de correo electrónico. Podemos consultar el funcionamiento detallado en el RFC1939 [16].

10.- Protocolo IMAP

Este protocolo, al igual que el protocolo POP3, se encarga de la descarga de los mensajes del servidor a un cliente de correo electrónico con la diferencia de que la sincronización entre el servidor de correos electrónico y el cliente de correos es continua, manteniendo una copia de seguridad en el servidor. Con este protocolo es posible tener varios clientes de correo configurados con la misma cuenta y los cambios que se realicen en cualquiera de los clientes de correo se verán reflejados en el servidor y en los diferentes clientes sincronizados. Este protocolo puede ser consultado en el RFC 6851[17].

Como hemos podido ver en el presente documento, el servicio de correo electrónico es un canal de comunicación que se ayuda de varios elementos para completar la comunicación entre dos usuarios de correo electrónico, no importando que estos estén registrados en 2 servidores de correo diferentes. Pero para poder entender por completo el comportamiento de este servicio tenemos que definir ciertas características de este servicio.

Este servicio establece una comunicación no orientada a conexión, lo que significa que el receptor no necesita estar conectado al servicio de correo electrónico para recibir el mensaje en su buzón de correo electrónico.

Los mensajes enviados por medio del correo electrónico transitan por el internet como archivos en texto plano, esto quiere decir que cualquiera que tenga una copia del mensaje puede abrir el correo electrónico y leer el mensaje siguiendo la estructura del protocolo SMTP.

Además de mandar mensajes tiene la facultad de enviar archivos multimedia en el mismo mensaje de correo electrónico, esta característica ha sido explotada bastante por empresas privadas y de gobierno para tener comunicados a sus empleados, departamentos, proveedores, socios, etc.

El correo permite enviar el mismo mensaje a más de un usuario sin la necesidad de hacer un mensaje para cada usuario, esto lo han utilizado muchas empresas de publicidad para hacer marketing a gran escala y a muy bajo costo, a este servicio se le llama SPAM.

Este medio de comunicación es bastante rápido ya que se estima que un mensaje de correo electrónico tiene que llegar a su destino a más tardar en 5 minutos, no importando la ubicación geográfica del servidor de correo electrónico.

Las características mencionadas nos dan a notar que el correo electrónico tiene una baja seguridad al momento de enviar los mensajes, ya que la información que mandamos en sumamente fácil de leer por cualquiera que pueda tener una copia del mensaje de correo electrónico. Si tomamos en cuenta que un mensaje de correo electrónico tiene que saltar por varios servidores de correo antes de llegar a su destino, es fácil suponer que se pueden interceptar una copia de los mensajes en el envío de servidor a servidor.

Por estos motivos los servidores de correo electrónico han implementado diversos candados de seguridad para blindar la transferencia de mensajes entre servidores. Una de las maneras que han encontrado para proporcionar dicha seguridad ha sido a través de la implementación de técnicas criptográficas.

La criptografía moderna se basa en dos corrientes metodológicas que son la criptografía simétrica y la criptografía asimétrica. Estas dos técnicas tienen como propósito ocultar el contenido de un mensaje con el fin de que solo sea leído por aquellos que estén autorizados, a esto se le llama cifrado.

12.- Cifrado

Es el procedimiento por el cual la información es transformada, esta transformación se realiza por medio de un algoritmo de cifrado junto con una clave para obtener un texto ilegible, el cual es llamado texto cifrado. Para poder obtener el texto original es necesario transformar el texto cifrado por medio de un algoritmo inverso al algoritmo de cifrado y la llave con que fue cifrado, esto es llamado algoritmo de descifrado.

Como ya fue mencionado la criptografía simétrica y asimétrica llevan a cabo la misma tarea pero a través de diferentes esquemas. Esto les proporciona diferentes características, tanto en seguridad como en implementación.

13.- Cifrado Simétrico

El Cifrado Simétrico es un método criptográfico en el cual se usa una misma clave para cifrar y descifrar mensajes.

La clave al ser usada tanto en el cifrado como en el descifrado se convierte en la parte más vulnerable del método, esta vulnerabilidad es contrarrestada con el tamaño de la clave. El tamaño de la clave es medido por medio del número de bits que la conforma, porque al tener una clave con más bits tendremos un campo de posibles claves más amplio. Por ejemplo, una clave de 56 bits tiene 256 combinaciones que a su vez son 72.057.594.037.927.936 claves posibles, este espacio de claves tan amplio dificulta al atacante el obtener fácilmente la clave.

Otra de las condiciones que se deben de cumplir para este esquema de cifrado se lleve a cabo es el intercambio de claves entre los participantes de la comunicación. Este intercambio de claves tiene que darse antes del envío de los mensajes.

14.- Cifrado Asimétrica

Este método criptográfico propone la generación de dos claves, de las cuales una es privada y está al resguardo del propietario del par de claves; la otra clave es pública y puede ser vista por cualquier usuario que quiera enviarle un mensaje al propietario.

Estas claves son mucho más grandes que las claves de cifrado simétrico y siempre se generan por pares, haciéndolas claves dependientes una de la otra, esto con el fin de que solo exista una clave privada que descifre lo que cifra su clave pública y viceversa. Este método de cifrado también cambia el esquema de envío de mensajes entre usuario ya que antes de empezar la comunicación los interlocutores tienen que buscar las claves públicas y una vez que tienen las claves públicas los mensajes son cifrados de la siguiente manera:

1. El interlocutor 1 toma su mensaje y es cifrado con la clave pública del interlocutor 2 obteniendo un texto cifrado.
2. Se envía el texto cifrado al interlocutor 2.
3. El interlocutor 2 toma el texto cifrado y lo descifra usando su clave privada para obtener el mensaje del interlocutor 1.

Este método de cifrado propone dar solución a 2 problemas, el primer problema es que para hacer el intercambio de llaves en un cifrado simétrico es muy difícil ya que al pasar la clave por un canal inseguro y se intercepta esta clave la comunicación quedaría expuesta a un ataque. Este problema es resuelto con la clave pública ya que esta llave puede ser vista por cualquier persona y tiene la cualidad de que si se quiere obtener la clave privada a partir de la pública sus cálculos son demasiado difíciles de resolver.

El segundo problema es el manejo de claves para establecer comunicaciones seguras con más de un usuario. Tomando el modelo de cifrado simétrico necesitaríamos tener una clave diferente para cada conexión segura que se desea establecer, mientras que en el cifrado asimétrico se necesita tener un par de claves, pública y privada, por cada usuario y esas claves son suficientes para hacer las conexiones seguras que se requieran. El beneficio que se tiene al utilizar claves públicas se ve al momento de establecer comunicaciones seguras, por ejemplo si tuviéramos 6 usuarios en una red con un esquema de cifrado asimétrico solo se necesitaría publicar las claves públicas de los usuarios en un lugar común para los 6, y cada vez que se quiera establecer una comunicación segura con algún usuario solo tienen que buscar su clave pública y cifrar el mensaje sin la necesidad de ponerse de acuerdo con cada uno de los usuarios de la red para establecer una clave para cada comunicación que se desea establecer como en el cifrado simétrico.

A pesar de la implementación de estas técnicas criptográficas los ataques a las comunicaciones siguen existiendo y estos ataques se han clasificado dependiendo de cuanta información tenga disponible el adversario para poder romper el cifrado y obtener la información deseada.

Nosotros nos enfocaremos a un tipo de ataque llamado “ataque de texto cifrado” o  COA que hace referencia a su nombre en inglés “*Ciphertext-only attack”*. [19] Este ataque solo cuenta con los textos cifrados que va recopilando de un canal o base de datos, estos textos cifrados los utiliza para hacer un análisis criptográfico de cómo se comportan la técnica de cifrado y trata de resolver el cifrado de los textos cifrados que ya tiene y los nuevos textos que va recopilando.

Este tipo de ataques es muy común en el internet aunque con muy baja efectividad cuando se implementa en comunicaciones altamente protegidas, y cuando se implementa en canales de comunicación desprotegidos la información obtenida llega a ser muy pobre. En los últimos años se han dado cuenta que si pones a este tipo de agentes a atacar comunicaciones sin cifrado obtienes características valiosas sobre los usuarios que utilizan este tipo de canales de comunicación, este tipo de ataques son ejecutados por programas llamados agentes clasificadores.

15.-Agentes Clasificadores.

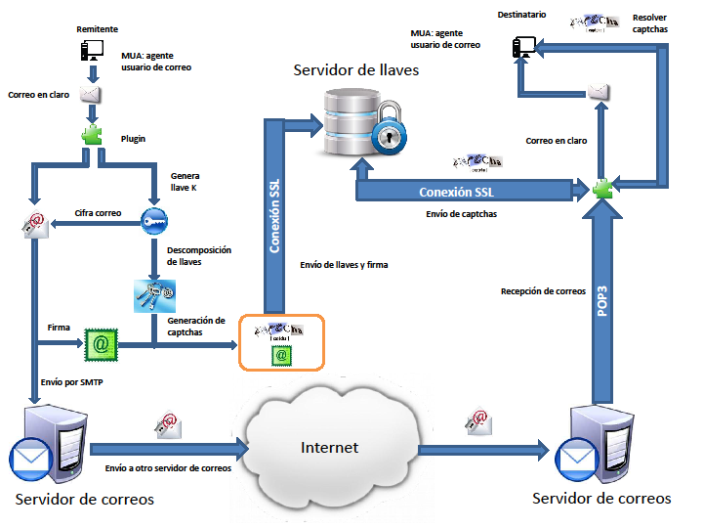
Los agentes clasificadores son programas que se dedican a observar los mensajes que se intercambian entre los usuarios de correo electrónico, con el fin de clasificarlos e identificar a todos los usuarios que cumplan con cierto criterio. Esta clasificación se hace de manera masiva y la realiza haciendo una búsqueda de palabras claves dentro de los mensajes de los usuarios. Por ejemplo, el clasificador puede estar interesado en los mensajes que contienen la palabra clave "Bomba", así que todos los mensajes que contengan esta palabra serán etiquetados en una clasificación en específico, este proceso se lleva a cabo por medio de técnicas de “Reconocimiento de patrones” y “Aprendizaje Maquina” para encontrar y clasificar los mensajes que intercepta.[12]

La clasificación de estos mensajes tiene diversos usos, ya que pueden ser clasificados con fines demográficos, con fines comerciales o con fines gubernamentales. Todo esto con el propósito de generar las estadísticas de comportamientos e intereses de los usuarios de correo electrónico.

Después de conocer el funcionamiento de los agentes, podríamos deducir que la solución más simple y obvia es cifrar el correo haciendo que este sea ilegible y siendo que, el agente clasifica los correos electrónicos semánticamente dependiendo de las palabras que encuentra en el correo. Esta solución resuelve el problema de que un correo sea clasificado por el agente clasificador, pero esta solución no es óptima ya que la infraestructura de los cifrados asimétricos comunes es demasiado para este problema.

14.-Propuesta (Esquema y Acotación)

1.-Esquema



2.-PGP

3.-CAPTCHA

4.-SSE.

5.-Tecnologias.

Cifrado

Tablas HASH

Generadores de CAPTCHAS

Servidores

Protocolos

S.O.

Lenguajes de Programación

Capítulo III:

Metodología aplicada.

Diagramas de caso de uso.

Diagrama general de casos de uso.

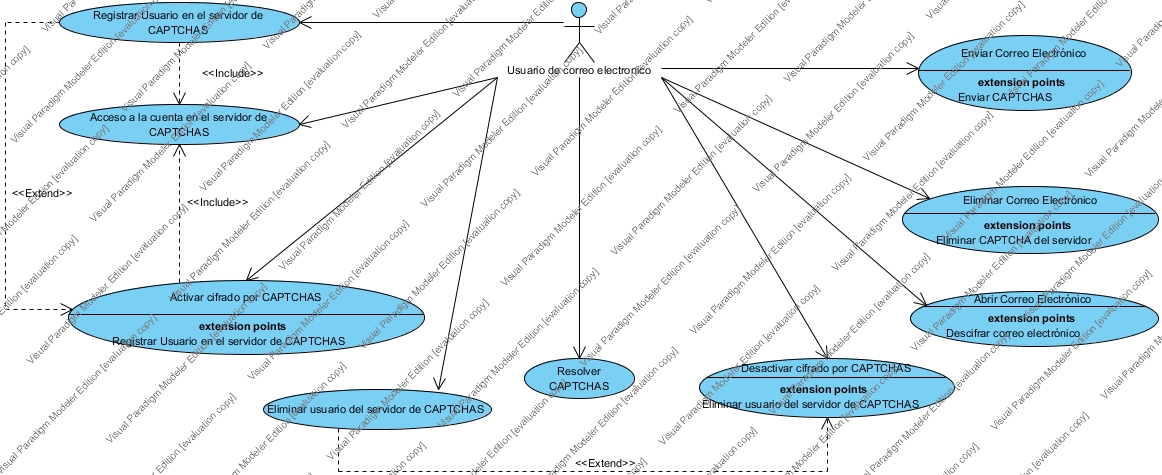
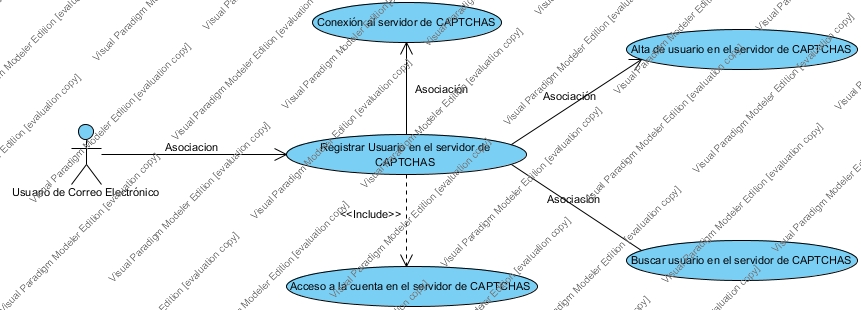


Diagrama de caos de uso CU Registrar usuario en el servidor de CAPTCHAS.



|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Registrar Usuario en el servidor de CAPTCHAS | | |
| **Actor** | Actor1. Usuario de Corro Electrónico | | |
| **Descripción** | Describe los pasos necesarios para registrar un nuevo usuario en el servidor de CAPTCHAS. | | |
| **Pre-condiciones** | Tener una cuenta de correo electrónico. | | |
| **Post-condiciones** | Activación del módulo de cifrado por CAPTCHAS. | | |
| **Puntos de inclusión** | Acceso a la cuenta en el servidor de CAPTCHAS. | | |
| **Puntos de extensión** |  | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El usuario selecciona la opción registrarse en el servidor de CAPTCHAS |
|  | 2 | Sistema | El cliente de correo despliega un formulario con la información necesaria para dar de alta en el servidor de CAPTCHAS. |
|  | 3 | Actor | Completa el formulario y oprime el botón de registrar. |
|  | 4 | Sistema | El sistema valida los datos proporcionados por el usuario. |
|  | 5 | Sistema | Se conecta con el servidor y valida si el usuario ya está registrado. <FA01 - Usuario ya registrado> <FA02 - Falla en la conexión con el servidor> |
|  | 6 | Sistema | Manda la información del usuario y lo da de alta. |
|  | 7 | Sistema | Despliega el siguiente mensaje "El usuario se ha dado de alta correctamente" |
|  |  |  | Fin del flujo principal. |
|  | FA01 - Usuario ya registrad | | |
| **Flujo alternativo** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | Despliega el siguiente mensaje "El usuario ya está registrado favor de proporcionar otra cuenta de correo electrónico" |
|  | 2 |  | El flujo continúa en el paso 3 del flujo principal. |
|  | 3 |  | Fin del flujo alternativo |
|  | FA02 - Falla en la conexión con el servidor | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | Despliega el siguiente mensaje " La conexión con la red es nula o limitada, favor de realizar esta operación más tarde" |
|  | 2 |  | El flujo continúa en el paso 1 del flujo principal. |
|  | 3 |  | Fin del flujo alternativo |

Diagrama de casos de uso CU Acceso a la cuenta en el servidor de CAPTCHAS

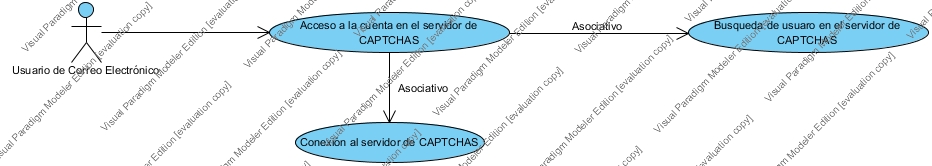
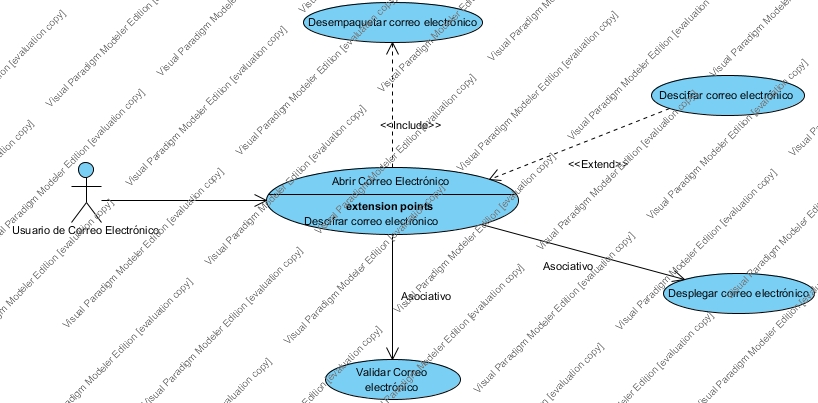
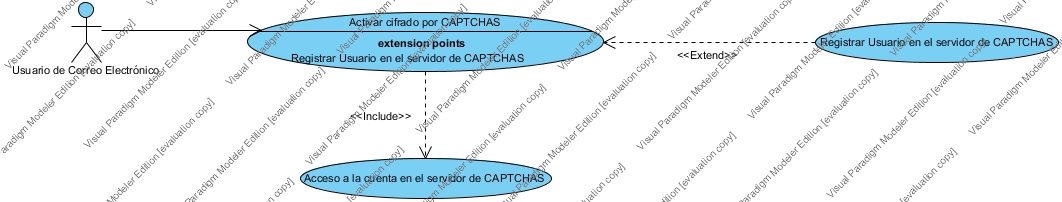


Diagrama de casos de uso CU Abrir Correo Electrónico.



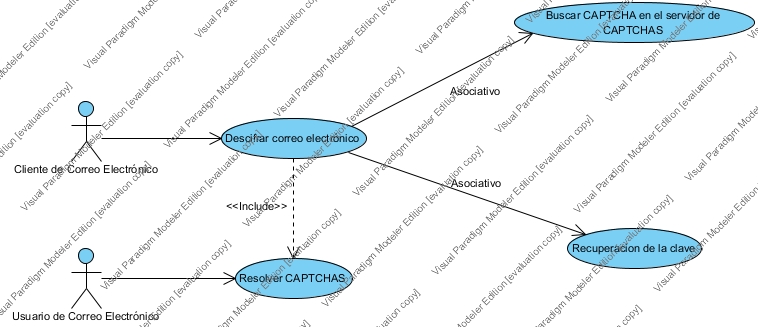
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Abrir correo electrónico | | |
| **Actor** | Actor 1. Usuario de correo electrónico | | |
| **Descripción** | Describe los pasos necesarios para abrir un mensaje de correo electrónico. | | |
| **Precondiciones** | 1. Iniciar sesión con su servidor de correo electrónico. 2. Descargar el correo electrónico que se desea abrir. | | |
| **Post-condiciones** | Despliegue del mensaje de correo electrónico descifrado. | | |
| **Puntos de inclusión** | Desempaquetar correo electrónico | | |
| **Puntos de extensión** | Descifrar correo electrónico | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El caso de uso comienza cuando el usuario selecciona el correo que desea abrir. |
|  | 2 | Sistema | El sistema manda a llamar a la función de desempaquetar correo electrónico. |
|  | 3 | Sistema | Valida si el mensaje viene timbrado. <FA01 - El mensaje no viene timbrado> |
|  | 4 | Sistema | Invoca al caso de uso <CU Descifrar correo electrónico> |
|  | 5 | Sistema | Recibe el mensaje de correo electrónico descifrado |
|  | 6 | Sistema | Despliega el contenido completo del mensaje al usuario |
|  | 7 |  | Fin del flujo principal |
|  | FA01 - El mensaje no viene timbrado | | |
| **Flujo alternativo** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | El flujo continúa en el paso 6 del flujo principal. |
|  | 2 |  | Fin del flujo alternativo |

Diagrama casos de uso CU Activar cifrado por CAPTCHAS.



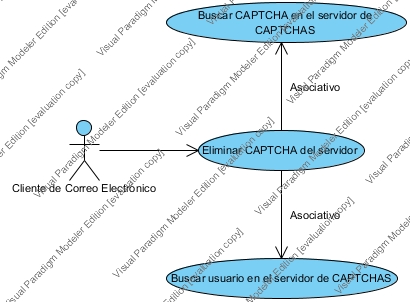
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Activar cifrado por CAPTCHAS | | |
| **Actor** | Actor 1. Usuario de correo electrónico | | |
| **Descripción** | Describe los pasos necesarios para activar el módulo de cifrado CAPTCHAS en el cliente de correo electrónico. | | |
| **Pre-condiciones** | 1. Instalar el módulo de cifrado por CAPTCHAS | | |
| **Post-condiciones** | Activación del cifrado y descifrado por CAPTCHAS. | | |
| **Puntos de inclusión** |  | | |
| **Puntos de extensión** | Registrar usuario del servidor de CAPTCHAS | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El caso de uso inicia cuando el actor seleccionar la opción "Activar cifrado por CAPTCHAS" |
|  | 2 | Sistema | El sistema verifica si la dirección de correo del usuario está registrada en el servidor de CAPTCHAS<FA01 -Usuario no registrado en el servidor> |
|  | 3 | Sistema | Despliega una ventana con el mensaje "Activación del módulo de cifrado por CAPTCHAS" |
|  | 4 |  | Fin del flujo básico. |
| **Flujos alternativos** | FA01 -Usuario no registrado en el servidor | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | El sistema despliega una ventana con las opciones de "Registrarse" y "Cancelar". <FA02 Cancelar activación> |
|  | 2 | Actor | Oprime el botón de "Registrarse" |
|  | 3 | Sistema | El sistema invoca al caso de uso <CU Registrar usuario en el servidor de CAPTCHAS> |
|  | 4 | Sistema | El sistema obtiene una respuesta satisfactoria del registro |
|  | 5 |  | El flujo continúa en el paso 2 del flujo principal. |
|  | 6 |  | Fin del flujo alterno. |
|  | FA02 - Cancelar activación. | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El Actor selecciona "Cancelar" |
|  | 2 | Sistema | Cierra la ventana de selección |
|  | 3 |  | El flujo continúa en el paso 1 del flujo principal. |
|  | 4 |  | Fin del flujo alterno. |

Diagrama de casos de uso CU Descifrar correo electrónico.



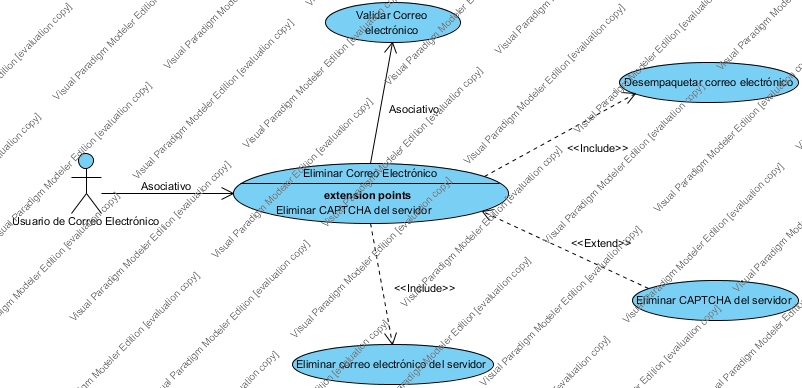
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Desactivar cifrado por CAPTCHAS | | |
| **Actor** | Actor 1. Usuario de correo electrónico | | |
| **Descripción** | Describe los pasos necesarios para desactivar el módulo de cifrado CAPTCHAS en el cliente de correo electrónico | | |
| **Pre-condiciones** | 1. Activar cifrado por CAPTCHAS. 2. Registrar usuario en el servidor de CAPTCHAS | | |
| **Post-condiciones** | Desactivación del cifrado y descifrado por CAPTCHAS. | | |
| **Puntos de inclusión** |  | | |
| **Puntos de extensión** | Eliminar usuario del servidor de CAPTCHAS | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El caso de uso inicia cuando el actor seleccionar la opción "Desactivar cifrado por CAPTCHAS" |
|  | 2 | Sistema | El sistema despliega la venta con las opciones de "Desactivar cifrado" y "Eliminar usuario" <FA01 - Eliminar usuario> |
|  | 3 | Actor | Selecciona la Desactivación del cifrado por CAPTCHAS |
|  | 4 | Sistema | El sistema desactiva el módulo de cifrado por CAPTCHA |
|  | 5 |  | Fin del flujo principal. |
| **Flujos alternativos** | FA01 - Eliminar usuario | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El Actor selecciona "Eliminar usuario" |
|  | 2 | Sistema | El sistema despliega una ventana con las opciones de "Aceptar" y "Cancelar" para confirmar la eliminación del usuario. <FA02 - Cancelar acción eliminar usuario> |
|  | 3 | Actor | Oprime el botón de "Aceptar" |
|  | 4 | Sistema | Establece la conexión con el servidor de CAPTCHAS <FA03 - Fallo en la conexión con el servidor> |
|  | 5 | Sistema | Busca y elimina al usuario de la base de datos desplegando la confirmación del servidor. |
|  | 6 | Actor | Oprime el botón de "Aceptar" |
|  | 7 | Sistema | Desactiva el módulo de cifrado por CAPTCHA |
|  | 8 |  | Fin del flujo alternativo |
|  | FA02 - Cancelar acción eliminar usuario | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | El Actor selecciona "Cancelar" |
|  | 2 | Sistema | Cierra la ventana de confirmación |
|  | 3 |  | Fin de flujo alternativo |
|  | FA03 - Fallo en la conexión con el servidor | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | Despliega una ventana de alerta con el mensaje "No sé a podido establecer la conexión con el servidor, es probable que no se tenga conexión a internet. Favor de intentarlo más tarde" |
|  | 2 | Actor | Cierra la ventana de alerta |
|  | 3 |  | El flujo continua en el paso 1 del flujo principal |
|  | 4 |  | Fin del flujo alterno |

Diagrama de caos de uso CU Eliminar CAPTCHA del servidor.



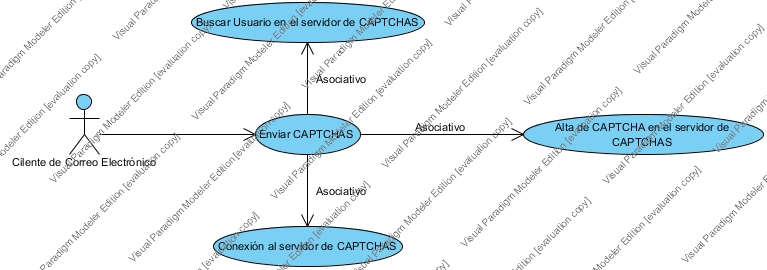
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Eliminar CAPTCHA del servidor | | |
| **Actor** | Actor 1. Cliente de correo electrónico. | | |
| **Descripción** | Describe los pasos necesarios para eliminar los CAPTCHAS del servidor de CAPTCHAS. | | |
| **Pre-condiciones** | 1. Solicitar eliminar un mensaje de correo electrónico | | |
| **Post-condiciones** |  | | |
| **Puntos de inclusión** |  | | |
| **Puntos de extensión** |  | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | Solicita eliminar CAPTCHA del servidor de CAPTCHAS |
|  | 2 | Sistema | Busca al usuario y el CAPTCHA a eliminar en el servidor de CAPTCHAS |
|  | 3 | Sistema | Elimina el CAPTCHA solicitado |
|  | 4 | Sistema | Regresa la confirmación de que se eliminó el CAPTCHA. |

Diagrama de casos de uso CU Eliminar correo electrónico.



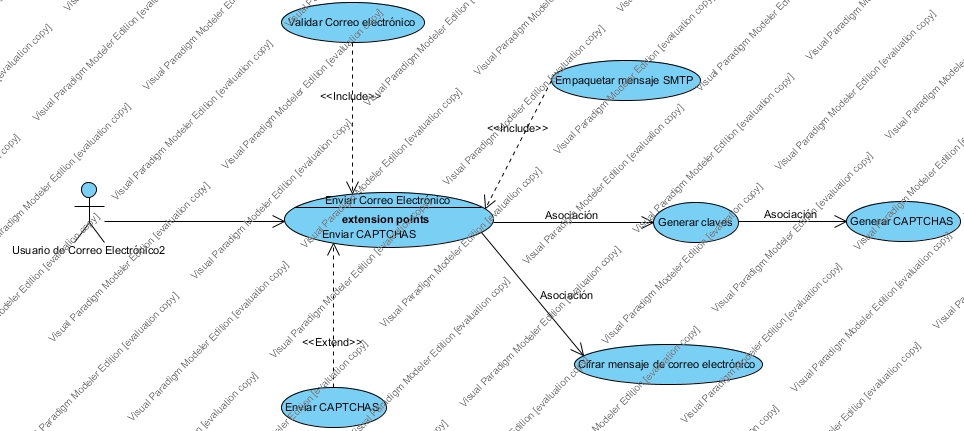
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Eliminar correo electrónico | | |
| **Actor** | Actor 1. Usuario de correo electrónico. | | |
| **Descripción** | Describe los pasos necesarios para eliminar un mensaje de correo electrónico. | | |
| **Precondiciones** | 1. Seleccionar un mensaje de correo electrónico | | |
| **Post-condiciones** | Mensaje y CAPTCHA eliminados. | | |
| **Puntos de inclusión** | 1. Desempaquetar correo electrónico. 2. Eliminar correo electrónico del servidor | | |
| **Puntos de extensión** | Eliminar CAPTCHA del servidor de CAPTCHAS | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | Selecciona un mensaje de correo electrónico a eliminar. |
|  | 2 | Sistema | Desempaqueta el mensaje de correo electrónico |
|  | 3 | Sistema | Valida si el mensaje esta timbrado. <FA01 - Mensaje no timbrado> |
|  | 4 | Sistema | Invoca al caso de uso <CU Eliminar CAPTCHA del servidor> |
|  | 5 | Sistema | Elimina el mensaje de correo electrónico y despliega el mensaje "El correo se ha eliminado correctamente" |
|  | 6 |  | Fin del flujo principal. |
| **Flujo alternativo** | FA01 - Mensaje no timbrado | | |
|  |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | El sistema continúa a partir del paso 5 del flujo principal. |
|  |  |  | Fin del flujo alternativo |

Diagrama de casos de uso CU Enviar CAPTCHAS



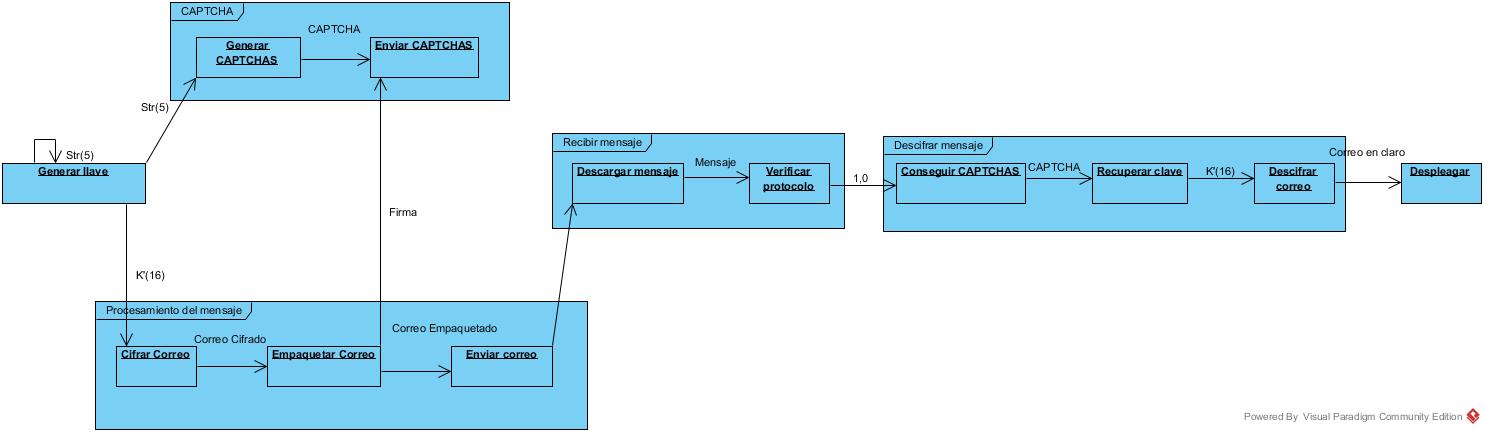
|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Enviar CAPTCHAS | | |
| **Actor** | Actor 1. Cliente de correo electrónico. | | |
| **Descripción** | Describe los pasos necesarios para enviar el CAPTCHA el servidor de CAPTCHAS | | |
| **Precondiciones** | 1. Solicitar el envió de un nuevo mensaje de correo electrónico. | | |
| **Post-condiciones** | Envío del CAPTCHA al servidor de CAPTCHAS | | |
| **Puntos de inclusión** |  | | |
| **Puntos de extensión** |  | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | Solicita el envío de CAPTCHA al servidor |
|  | 2 | Sistema | Abre la conexión y busca al usuario en el servidor de CAPTCHAS |
|  | 3 | Sistema | Da de alta el CAPTCHA en el servidor asociándolo con el usuario. |
|  | 4 | Sistema | Regresa la confirmación de que se dio de alta el CAPTCHA |
|  | 5 |  | Fin del flujo principal |

Diagrama de casos de uso CU Enviar correo electrónico.



|  |  |  |  |
| --- | --- | --- | --- |
| **Caso de Uso** | CU Enviar correo electrónico | | |
| **Actor** | Actor 1. Usuario de correo electrónico. | | |
| **Descripción** | Describe los pasos necesarios para enviar un mensaje de correo electrónico cifrado a otro usuario de correo electrónico. | | |
| **Precondiciones** | 1. El usuario tiene que redactar un mensaje de correo electrónico que contenga la dirección del destinatario. | | |
| **Post-condiciones** | Envió de un mensaje cifrado al servidor de correo electrónico y el registro del CAPTCHA en el servidor de CAPTCHAS. | | |
| **Puntos de inclusión** | 1. Validar correo electrónico. 2. Empaquetar mensaje de correo electrónico SMTP. | | |
| **Puntos de extensión** | Enviar CAPTCHA | | |
| **Flujo principal** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Actor | Oprime el botón "Enviar" |
|  | 2 | Sistema | Valida que el mensaje de correo electrónico contenga los datos mínimos.<FA01 - Campos no completados> |
|  | 3 | Sistema | Genera una llave de cifrado |
|  | 4 | Sistema | Con la llave generada genera el CAPTCHA y cifra el mensaje de correo electrónico. |
|  | 5 | Sistema | Toma el mensaje cifrado y es empaquetado para enviarse al servidor de correo electrónico |
|  | 6 | Sistema | Toma el CAPTCHA generado y se envía al caso de uso <CU Enviar CAPTCHA> |
|  | 7 | Sistema | Despliega el mensaje de "envío satisfactorio" |
|  | 8 |  | Fin del flujo principal |
|  | FA01 - Campos no completados | | |
| **Flujo alternativo** |  | Actor/Sistema | Acción a realizar |
|  | 1 | Sistema | Notifica al usuario cuales campos mal proporcionado para poder enviar el mensaje correctamente |
|  | 2 | Actor | Modifica los campos solicitados |
|  | 3 |  | El flujo continua en el paso 1 del flujo principal |
|  |  |  | Fin del flujo alternativo |

**Diagrama a bloque general del sistema**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Generar clave** | **Generar CAPTCHA** | **Procesamiento del mensaje** | **Recibir mensaje** | **Descifrar mensaje** | **Desplegar** |
| **Entradas** | * Señal de activación | * Cadena de 5 caracteres: Str(5) | * Clave de 16 bytes: K’(16) * Mensaje de correo | * Correo Cifrado | * Verificación (1,0) | * Correo en claro |
| **Salidas** | * Cadena de 5 caracteres: Str(5) * Clave de 16 bytes: K’(16) | * Señal de envió | * Correo Cifrado | * Verificación (1,0) | * Correo en claro | * - |
| **Descripción** | Se activa el proceso generar clave, este crea una palabra de 5 caracteres(Str(5)), procesa la palabra Str(5) por medio de una función hash obteniendo una palabra de 256 caracteres(K(256)) y recorta esta clave a una palabra de 16 caracteres(K’(16)) | Toma la entrada Str(5) y la convierte en una imagen CAPTCHA, Posteriormente inicia una conexión con el servidor de CAPTCHAS para mandarlo a este. | Cifra el mensaje de correo con la clave K’(16), posteriormente lo firma y genera un timbre para saber que fue creado con este esquema y lo empaqueta para su envió. | El cliente hace una petición al servidor y descarga el mensaje de correo electrónico, lo des empaqueta verifica la firma y el timbrado para saber de quién viene y si está cifrado bajo este esquema | Se hace una petición al servidor de CAPTCHAS, se descargan los CAPTCHAS asociados al correo, ya con el CAPTCHA este se resuelve y se recupera la cadena Str(5), esta se pasa por una función hash y se recupera K(256), esta se corta a K’(16), con esto se descifra el mensaje | Se muestra el correo descifrado en la interface del cliente de correo electrónico |

**Generar clave**



|  |  |  |  |
| --- | --- | --- | --- |
|  | **Generar Str(5)** | **Aplicar Función Hash** | **Recortar Hash K’** |
| **Entradas** | * Llamada a Función | * Cadena de 5 caracteres: Str(5) | * Digesto K(128) |
| **Salidas** | * Cadena de 5 caracteres: Str(5) | * Digesto K(128) | * K’(16) |
| **Descripción** | Toma una función random módulo 67, para formar una palabra con 5 caracteres aleatorios tomados del siguiente conjunto.  Anillo67{-.,+\*[a-z][A-Z]} | Se pasa la cadena Str(5) por una función hash SHA-1 para obtener un digesto único de esta palabra | Se copian a otro string lo primeros 16 caracteres del digesto K(128) para formar la clave K’(16) |

**Cifrar Correo**

|  |  |
| --- | --- |
|  | **Cifrar** |
| **Entrada** | * Clave K’(16) * Mensaje de correo |
| **Salida** | * Correo cifrado |
| **Descripción** | Se cifra el mensaje con un algoritmo de llave simétrica (AES o DES) usando una llave de 16bytes o 128bits |

**Empaquetar Correo**

C:\Users\Administrador\Documents\VPProjects\3Empaquetar Correo.jpg

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Empaquetamiento SMTP** | **Timbrar Correo** | **Firmar Correo** |
| **Entrada** | * Mensaje Cifrado | * Correo Empaqueta | * Correo Timbrado |
| **Salida** | * Correo Empaquetado | * Correo Timbrado | * Correo Coreo Timbrado y Firmado * Firma |
| **Descripción** | Se toma el correo y se integra en el formato del correo marcado en el RFC822 | Se timbra el mensaje colocando una marca después del final del mensaje. Para señalar que el correo enviado está cifrado bajo este protocolo | El mensaje se firma a través de ….. para verificar que usuario mando el mensaje. |

**Enviar correo**

C:\Users\Administrador\Documents\VPProjects\4Enviar correo.jpg

|  |  |  |
| --- | --- | --- |
|  | **Abrir conexión SMTP** | **Envió de Correo por SMTP** |
| **Entradas** | * Petición | * Correo empaquetado |
| **Salidas** | * Canal de comunicación | * Confirmación de envió |
| **Descripción** | Se genera una petición para conexión SMTP | Se manda el correo electrónico al servidor por medio del protocolo SMTP |

**Generar CAPTCHA**

C:\Users\Administrador\Documents\VPProjects\5Generar CAPTCHAS.jpg

|  |  |  |
| --- | --- | --- |
|  | **Generar respuesta del CAPTCHA** | **Transformar palabra** |
| **Entradas** | * Cadena de caracteres: Str(5) | * Señal de confirmación |
| **Salidas** | * Señal de confirmación | * Imagen CAPTCHA |
| **Descripción** | Genera un archivo con la respuesta del CAPTCHA | Convierte el Str(5) en una imagen distorsionada que llamaremos CAPTCHA |

**Enviar CAPTCHAS (Usuario existente)**

C:\Users\Administrador\Documents\VPProjects\6Enviar_CAPTCHA.jpg

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Abrir conexión** | **Verificar Usuario** | **Mandar CAPTCHA** | **Registrar en base de datos** | **Cerrar Conexión** |
| **Entradas** | * CAPTCHAS | * Datos de Usuario | * Verificación de usuario | * Datos de usuario * CAPTCHA | * Verificación |
| **Salidas** | * Datos de usuario | * Verificación de usuario | * CAPTCHA | * Verificación |  |
| **Descripción** | Se genera una petición para poder entablar una conexión con el servidor de CAPTCHAS | Se verifica la existencia del usuario en el servidor, si existe se le da acceso | Ya verificado el usuario se manda el CAPTCHA al servidor | Se registran los datos del CAPTCHA en la base de datos y se envía una verificación | Se cierra la conexión y se guardan los datos |

**Enviar CAPTCHAS (Usuario inexistente)**

**C:\Users\Administrador\Documents\VPProjects\7bisEnviar CAPTCHAS usuario inexistente.jpg**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Abrir conexión** | **Registrar usuario en Base de Datos** | **Verificar Usuario** | **Mandar CAPTCHA** | **Registrar en base de datos** | **Cerrar Conexión** |
| **Entradas** | * CAPTCHAS | * Datos de usuario | * Verificación de registro | * Verificación de usuario | * Datos de usuario * CAPTCHA | * Verificación |
| **Salidas** | * Datos de usuario | * Verificación de registro | * Verificación de usuario | * CAPTCHA | * Verificación |  |
| **Descripción** | Se genera una petición para poder entablar una conexión con el servidor de CAPTCHAS | Se da de alta al usuario en la base de datos | se le da acceso al usuario | Ya verificado el usuario se manda el CAPTCHA al servidor | Se registran los datos del CAPTCHA en la base de datos y se envía una verificación | Se cierra la conexión y se guardan los datos |

**Descargar mensaje**C:\Users\Administrador\Documents\VPProjects\8Descargar mensaje.jpg

|  |  |  |
| --- | --- | --- |
|  | **Abrir conexión al servidor de correo** | **Descargar mensaje por IMAP o POP3** |
| **Entradas** | * Señal de activación | * Confirmación |
| **Salidas** | * Confirmación | * Correo electrónico |
| **Descripción** | El receptor se conecta al servidor de correo electrónico e inicia la sesión | Descarga del servidor de correo electrónico todos los mensajes que aún no se hallan descargado. |

**Verificar protocolo (con protocolo valido)**

C:\Users\Administrador\Documents\VPProjects\9Verificar_CAPTCHA.jpg

|  |  |  |
| --- | --- | --- |
|  | **Abrir mensaje** | **Verificar mensaje** |
| **Entradas** | * Correo electrónico | * mensaje |
| **Salidas** | * Mensaje | * verificación |
| **Descripción** | Se toma el mensaje descargado del servidor y se des empaqueta para dejar solo el texto del mensaje | Se verifica que el mensaje tenga la bandera correspondiente a que está cifrado con este esquema |

**Verificar protocolo (con protocolo invalido)**

|  |  |  |
| --- | --- | --- |
|  | **Abrir mensaje** | **Verificar mensaje** |
| **Entradas** | * Correo electrónico | * mensaje |
| **Salidas** | * Mensaje | * verificación |
| **Descripción** | Se toma el mensaje descargado del servidor y se des empaqueta para dejar solo el texto del mensaje | Si la verificación es negativa se manda directamente al bloque de Despliegue |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Abrir Conexión** | **Validar Usuario** | **Accesar a cuenta** | **Verificar existencia de CAPTCHA** | **Enviar CAPTCHA** | **Cerrar Conexion** |
| **Entradas** | * Confirmacion | * Datos usuario | * Contraseña | * Petición de CAPTCHAS | * Confirmacion | * Peticion |
| **Salidas** | * verificacion | * Contraseña | * confirmacion | * confirmacion | * CAPTCHA | * Confirmacion |
| **Descripción** | Se abre una conexión con el servidor de CAPTCHAS | Se verifica que el usuario este dado de alta en el servidor mandándole una petición a la base de datos, si el usuario existe se accesa | Si esta dado de alta en el servidor se manda la contraseña para que pueda tener acceso a los CAPTCHAS de su cuenta | Se verifica que los CAPTCHAS que están ligados al mensaje que realizo la petición existan | Si existen estos CAPTCHAS son enviados de regreso al mensaje | Ya enviados los CAPTCHAS se cierra la conexión con el servidor |

**Conseguir CAPTCHAS (Usuario existente)**C:\Users\Administrador\Documents\VPProjects\10Conseguir_captcha.jpg

**Recuperar clave**C:\Users\Administrador\Documents\VPProjects\10Recuperar clave.jpg

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Resolver CAPTCHA** | **Aplicar Función Hash** | **Recortar llave** |
| **Entradas** | * CAPTCHA | * Cadena de 5 caracteres: Str(5) | * Digesto K(128) |
| **Salidas** | * Str(5) | * Digesto K(128) | * K’(16) |
| **Descripción** | Se despliega el CAPTCHA para que el usuario pueda resolverlo | Se pasa la cadena Str(5) por una función hash SHA-1 para obtener un digesto único de esta palabra | Se copian a otro string lo primeros 16 caracteres del digesto K(128) para formar la clave K’(16) |

**Descifrar correo**

|  |  |
| --- | --- |
|  | **Descifrar** |
| **Entrada** | * Clave K’(16) * Mensaje de correo |
| **Salida** | * Correo descifrado |
| **Descripción** | Se descifra el mensaje con un algoritmo de llave simétrica (AES o DES) usando una llave de 16bytes o 128bits |

Capítulo IV:

Análisis e interpretación de resultados.

**Objetivo del prototipo**

Conocer el uso, funcionamiento e implementación de herramientas de cifrado, hashing y generación de CAPTCHAS, con el fin de conocer la integración de estos módulos en diferentes lenguajes de programación.

**Prototipo 1**

Se implementó un módulo de cifrado de mensajes de texto en lenguaje C++. Tratando de simular el proceso de cifrado del esquema que estamos usando.

La primera parte del proceso es abrir el mensaje para lo cual usamos las bibliotecas estándar stdio.h, posteriormente crearemos una palabra aleatoria de 10 caracteres usando una función Rand()%100 casteando el valor de salida a un char.

Al resultado se pasa por una función de hashing, esta función no es nativa de ninguna biblioteca estándar de C++ ni de C, por lo que se tuvo que conseguir una en internet y probar que efectivamente funcionara como lo necesitamos.

Posteriormente este hash se usara como llave para cifrar el mensaje que ya abrimos, para esto necesitaremos una función AES o DES, ninguna de estas es estándar de alguna biblioteca de C o C++, así que tendremos que buscarlas y verificar su funcionamiento.

**Conclusión**

Podemos ver que en C++ el proceso es simple pero se necesita buscar muy bien las bibliotecas externas que se usaran, ya que no siempre están funcionando correctamente, en algunos casos están ni siquiera compilan.

Este caso fue particularmente evidente al buscar una biblioteca de C o C++ que pudiera realizar el cifrado con AES o DES, encontrándonos con bibliotecas que cifraban mal ya que al meter la misma llave no descifraban e incluso bibliotecas que no logramos hacer que compilaran.

Main.cpp

#include <iostream>

#include <fstream>

#include "sha256.h"

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <sstream>

using namespace std;

int main()

{

FILE \*archivo;

srand(time(NULL));

archivo=fopen("/home/allan/Dropbox/Allan/TT/prueba 1/Correo1.txt","r");

char linea[128],p;

int c=0,n=0;

string input = "grape";

string output1;

stringstream output2;

n=10;

char keyy[n];

//long contador = 0L;

while(!(feof(archivo)))

{

p=fgetc(archivo);

linea[c]=p;

c++;

}

cout << linea << "\n" << endl;

fclose(archivo);

for(int i=0;i<n;i++)

{

keyy[i]=(char)((rand())%100);

}

output2<<keyy;

output2>>input;

input=string(keyy);

cout << "tamaño: " << n << " palabra: " << keyy << ". ." << input << ".\n";

output1=sha256(input);

cout << "sha256('"<< input << "'):" << output1 << endl;

return 0;

}

Sha256.h

#ifndef SHA256\_H

#define SHA256\_H

#include <string>

class SHA256

{

protected:

typedef unsigned char uint8;

typedef unsigned int uint32;

typedef unsigned long long uint64;

const static uint32 sha256\_k[];

static const unsigned int SHA224\_256\_BLOCK\_SIZE = (512/8);

public:

void init();

void update(const unsigned char \*message, unsigned int len);

void final(unsigned char \*digest);

static const unsigned int DIGEST\_SIZE = ( 256 / 8);

protected:

void transform(const unsigned char \*message, unsigned int block\_nb);

unsigned int m\_tot\_len;

unsigned int m\_len;

unsigned char m\_block[2\*SHA224\_256\_BLOCK\_SIZE];

uint32 m\_h[8];

};

std::string sha256(std::string input);

#define SHA2\_SHFR(x, n) (x >> n)

#define SHA2\_ROTR(x, n) ((x >> n) | (x << ((sizeof(x) << 3) - n)))

#define SHA2\_ROTL(x, n) ((x << n) | (x >> ((sizeof(x) << 3) - n)))

#define SHA2\_CH(x, y, z) ((x & y) ^ (~x & z))

#define SHA2\_MAJ(x, y, z) ((x & y) ^ (x & z) ^ (y & z))

#define SHA256\_F1(x) (SHA2\_ROTR(x, 2) ^ SHA2\_ROTR(x, 13) ^ SHA2\_ROTR(x, 22))

#define SHA256\_F2(x) (SHA2\_ROTR(x, 6) ^ SHA2\_ROTR(x, 11) ^ SHA2\_ROTR(x, 25))

#define SHA256\_F3(x) (SHA2\_ROTR(x, 7) ^ SHA2\_ROTR(x, 18) ^ SHA2\_SHFR(x, 3))

#define SHA256\_F4(x) (SHA2\_ROTR(x, 17) ^ SHA2\_ROTR(x, 19) ^ SHA2\_SHFR(x, 10))

#define SHA2\_UNPACK32(x, str) \

{ \

\*((str) + 3) = (uint8) ((x) ); \

\*((str) + 2) = (uint8) ((x) >> 8); \

\*((str) + 1) = (uint8) ((x) >> 16); \

\*((str) + 0) = (uint8) ((x) >> 24); \

}

#define SHA2\_PACK32(str, x) \

{ \

\*(x) = ((uint32) \*((str) + 3) ) \

| ((uint32) \*((str) + 2) << 8) \

| ((uint32) \*((str) + 1) << 16) \

| ((uint32) \*((str) + 0) << 24); \

}

#endif

Sha256.cpp

#include "sha256.h"

#include <cstring>

#include <fstream>

const unsigned int SHA256::sha256\_k[64] = //UL = uint32

{0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5,

0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5,

0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3,

0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174,

0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc,

0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da,

0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7,

0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967,

0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13,

0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85,

0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3,

0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070,

0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5,

0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3,

0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208,

0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2};

void SHA256::transform(const unsigned char \*message, unsigned int block\_nb)

{

uint32 w[64];

uint32 wv[8];

uint32 t1, t2;

const unsigned char \*sub\_block;

int i;

int j;

for (i = 0; i < (int) block\_nb; i++) {

sub\_block = message + (i << 6);

for (j = 0; j < 16; j++) {

SHA2\_PACK32(&sub\_block[j << 2], &w[j]);

}

for (j = 16; j < 64; j++) {

w[j] = SHA256\_F4(w[j - 2]) + w[j - 7] + SHA256\_F3(w[j - 15]) + w[j - 16];

}

for (j = 0; j < 8; j++) {

wv[j] = m\_h[j];

}

for (j = 0; j < 64; j++) {

t1 = wv[7] + SHA256\_F2(wv[4]) + SHA2\_CH(wv[4], wv[5], wv[6])

+ sha256\_k[j] + w[j];

t2 = SHA256\_F1(wv[0]) + SHA2\_MAJ(wv[0], wv[1], wv[2]);

wv[7] = wv[6];

wv[6] = wv[5];

wv[5] = wv[4];

wv[4] = wv[3] + t1;

wv[3] = wv[2];

wv[2] = wv[1];

wv[1] = wv[0];

wv[0] = t1 + t2;

}

for (j = 0; j < 8; j++) {

m\_h[j] += wv[j];

}

}

}

void SHA256::init()

{

m\_h[0] = 0x6a09e667;

m\_h[1] = 0xbb67ae85;

m\_h[2] = 0x3c6ef372;

m\_h[3] = 0xa54ff53a;

m\_h[4] = 0x510e527f;

m\_h[5] = 0x9b05688c;

m\_h[6] = 0x1f83d9ab;

m\_h[7] = 0x5be0cd19;

m\_len = 0;

m\_tot\_len = 0;

}

void SHA256::update(const unsigned char \*message, unsigned int len)

{

unsigned int block\_nb;

unsigned int new\_len, rem\_len, tmp\_len;

const unsigned char \*shifted\_message;

tmp\_len = SHA224\_256\_BLOCK\_SIZE - m\_len;

rem\_len = len < tmp\_len ? len : tmp\_len;

memcpy(&m\_block[m\_len], message, rem\_len);

if (m\_len + len < SHA224\_256\_BLOCK\_SIZE) {

m\_len += len;

return;

}

new\_len = len - rem\_len;

block\_nb = new\_len / SHA224\_256\_BLOCK\_SIZE;

shifted\_message = message + rem\_len;

transform(m\_block, 1);

transform(shifted\_message, block\_nb);

rem\_len = new\_len % SHA224\_256\_BLOCK\_SIZE;

memcpy(m\_block, &shifted\_message[block\_nb << 6], rem\_len);

m\_len = rem\_len;

m\_tot\_len += (block\_nb + 1) << 6;

}

void SHA256::final(unsigned char \*digest)

{

unsigned int block\_nb;

unsigned int pm\_len;

unsigned int len\_b;

int i;

block\_nb = (1 + ((SHA224\_256\_BLOCK\_SIZE - 9)

< (m\_len % SHA224\_256\_BLOCK\_SIZE)));

len\_b = (m\_tot\_len + m\_len) << 3;

pm\_len = block\_nb << 6;

memset(m\_block + m\_len, 0, pm\_len - m\_len);

m\_block[m\_len] = 0x80;

SHA2\_UNPACK32(len\_b, m\_block + pm\_len - 4);

transform(m\_block, block\_nb);

for (i = 0 ; i < 8; i++) {

SHA2\_UNPACK32(m\_h[i], &digest[i << 2]);

}

}

std::string sha256(std::string input)

{

unsigned char digest[SHA256::DIGEST\_SIZE];

memset(digest,0,SHA256::DIGEST\_SIZE);

SHA256 ctx = SHA256();

ctx.init();

ctx.update( (unsigned char\*)input.c\_str(), input.length());

ctx.final(digest);

char buf[2\*SHA256::DIGEST\_SIZE+1];

buf[2\*SHA256::DIGEST\_SIZE] = 0;

for (int i = 0; i < SHA256::DIGEST\_SIZE; i++)

sprintf(buf+i\*2, "%02x", digest[i]);

return std::string(buf);

}

**Prototipo 2**

Se implementó un módulo de cifrado, descifrado y generación de CAPTCHAS en Python, simulando el proceso antes del envió del correo y el que se hace después de la recepción de los correos electrónicos.

Para este se usó el formato estándar del correo electrónico especificado en el RFC 822, también se usaron bibliotecas ya estandarizadas de python para la implementación de las funciones de hashing, funciones de cifrado y descifrado (AES o DES), funciones aleatorias y la generación de los CAPTCHAS.

**Conclusión**

Se logró generar todo el proceso de envió y parte de el de recibimiento de mensajes. En cuanto a el envió se logró leer el mensaje, crear el strig a partir de funciones random, crear la llave con esa string y cifrar el correo exitosamente, además de esto se logró leer el archivo de mensaje de correo electrónico y cifrar únicamente el mensaje que viene en este.

Por su parte el módulo de generación de CAPTCHAS mostró muchos problemas para generarlos, ya que no logramos hacer que el intérprete pudiera encontrar correctamente las funciones de la biblioteca de generación CAPTCHAS por lo que al no poder generar un CAPTCHA la recuperación no se puede realizar como se planteó, para verificar únicamente que las funciones sirven se implementó el descifrado del mensaje en el mismo método.

p.py

#! /usr/bin/env python

from Crypto.Hash import SHA256

from Crypto.Cipher import AES

import random

hash = SHA256.new()

semilla=""

r=0

for i in range(5):

r=random.randrange(100)

semilla=semilla+chr(r)

print str(i)+" "+str(r)+" "+chr(r)+" "+semilla

print semilla+"\n"

hash.update(semilla)

otra=hash.digest()

llave = ""

print otra

for i in range(16):

llave=llave+otra[i]

print str(i)+" "+otra[i]+" "+llave

print "\n"

print llave

archy=open('llave.txt','w')

archy.write(semilla)

archy.close()

arc=open('cifrado.txt','w')

archi=open('1443750804.V805Idc01e2M920300.jonnytest:2,S','r')

obj = AES.new(llave, AES.MODE\_ECB)

lineas=' '

c=0

while lineas!="":

c=c+1

lineas=archi.read(16)

if (((len(lineas))<16)and((len(lineas))>0)):

c=16-(len(lineas))

aux=lineas

for i in range(c):

aux=aux+" "

else:

aux=lineas

ciphertext = obj.encrypt(aux)

arc.write(ciphertext)

print str(c) +" " + lineas + " "+str(len(lineas))+" "+str(len(aux))+" "+ciphertext

for i in range(12):

archi.readline()

num=archi.tell()

print num

archi.seek((num+9))

lineas=archi.readline()

lineas=(lineas.replace("\n",""))

print lineas

print "v" + str(len(lineas))

for i in range(len(lineas)):

print lineas[i]

if (((len(lineas))<16)and((len(lineas))>0)):

c=16-(len(lineas))

aux=lineas

for i in range(c):

aux=aux+" "

else:

aux=lineas

ciphertext = obj.encrypt(aux)

print ciphertext + str(len(aux))

desif=obj.decrypt(ciphertext)

print desif

archi.close()

arc.close()

re.py

#! /usr/bin/env python

from Crypto.Hash import SHA256

from Crypto.Cipher import AES

import random

hash = SHA256.new()

llav=open('llave.txt','r')

sem=llav.read(5)

print sem

hash.update(sem)

otra=hash.digest()

llave = ""

for i in range(16):

llave=llave+otra[i]

print str(i)+" "+otra[i]+" "+llave

print "\n"

print llave

obj = AES.new(llave, AES.MODE\_ECB)

arc=open('cifrado.txt','r')

arch=open('si.txt','w')

lineas=' '

while lineas!="":

lineas=arc.read(16)

if (((len(lineas))<16)and((len(lineas))>0)):

c=16-(len(lineas))

aux=lineas

for i in range(c):

aux=aux+" "

else:

aux=lineas

deciphertext = obj.decrypt(lineas)

arch.write(deciphertext)

print lineas + " "+str(len(lineas))+" "+str(len(aux))+" "+deciphertext

cap.py

#! /usr/bin/env python

from captcha.audio import AudioCaptcha

from captcha.image import ImageCaptcha

audio = AudioCaptcha(voicedir='voices')

image = ImageCaptcha(fonts=['correo\_prueba1 / A.ttf'])

#data = audio.generate('1234')

#assert isinstance(data, bytearray)

#audio.write('1234', 'out.wav')

data = image.generate('1234')

assert isinstance(data, BytesIO)

image.write('1234', 'out.png')

Referencia

1. “Email”, Internet: http://en.wikipedia.org/wiki/Email, Mayo, 2015
2. B. Templeton, “Essays in Junk E-mail (Spam)”, Internet: http://www.templetons.com/brad/spam/
3. Interactive Advertising Bureau, Marcelo Brodsky, “Reflexiones jurídicas sobre el e-marketing en Chile”, Internet: http://www.iab.cl/reflexiones-juridicas-sobre-ele-marketing-en -chile.
4. D. Jurafsky, Text Classification, Stanford University Natural Language Processing.
5. S. Díaz Santiago y D. Chakraborty. “On Securing Communication from Profilers.” Proceedings of International Conference on Security and Cryptography, Secrypt 2012, pp.154-162, Rome, Italy, 2012.
6. Trend Micro, “Email encryption”, Internet: http://www.trendmicro.es/productos/email-encryption/
7. Thunderbird, “Firma digital y cifrado de mensajes”, Internet: https://support.mozilla.org/es/kb/firma-digital-y-cifrado-de-mensajes
8. D. R. Stinson, “Cryptography Theory and Practice”,3a ed, Ontario, Canada.Chapman&Hall/CRC. 2006
9. Philippe Golle y Ayman Farahat. “Defending Email Communication Against Proﬁling Attacks” Proceedings of the 2004 ACM workshop on Privacy in the electronic society, ACM New York, NY, USA ©2004 pp 39-40

[10]”Dominio de Internet”, Internet: <https://es.wikipedia.org/wiki/Dominio_de_Internet>, Noviembre2015

[11] “Servidor de correos”, Internet: <http://servilinux.galeon.com/>, Noviembre2015

[12] J. Klensin, “Simple Mail Transfer Protocol”, Patent 5321, October 2008.

[13] J. Myers, “Post Office Protocol - Version 3”, Patent 1939, May 1996.

[14] A. Gulbrandsen, “Internet Message Access Protocol (IMAP) - MOVE Extension”, Patent 6851, January 2013.

[15] “Cifrado”, Internet: <https://es.wikipedia.org/wiki/Cifrado_(criptograf%C3%ADa)>, Noviembre2015.

[16] “Ciphertext-only attack”, Internet: <https://en.wikipedia.org/wiki/Ciphertext-only_attack>, Noviembre2015