Universidad Nacional Autónoma de México

Facultad de Ingeniería

“Sistema Tutorial de Fundamentos de Criptografía”

TESIS

QUE PARA OBTENER EL TÍTULO DE
INGENIERA EN COMPUTACIÓN

PRESENTA

ERIKA AGUILLÓN MARTÍNEZ

DIRECTOR DE TESIS

M.C. Ma. Jaquelina López Barrientos

México, D.F. **2012**

DEDICATORIA

A mis padres:

Gracias por el apoyo incondicional a lo largo de mis estudios y por haberlo hecho siempre con tanto entusiasmo, gracias por sus enseñanzas y su amor, este logro también es suyo ya que el camino lo hemos recorrido con el esfuerzo de los tres, muchas gracias por ser mi fortaleza en la vida.

A Jaquelina López Barrientos

Sin lugar a dudas, es usted una de las personas más extraordinarias que he conocido, me siento muy afortunada de haber tenido la oportunidad de ser su alumna y posteriormente que haya dirigido el presente trabajo, muchas gracias por su apoyo incondicional, su paciencia infinita y su tenacidad que siempre me motivaron para no claudicar, creo que no me alcanzará la vida para agradecerle…

ÍNDICE

INTRODUCCIÓN

Hoy en día, pensar en un mundo sin computadoras y comunicaciones resulta prácticamente imposible ya que muchas de las actividades cotidianas de millones de personas están basadas en el empleo de estas tecnologías. Sectores económicos, financieros, gubernamentales, religiosos, sociales e industriales requieren almacenar, enviar y obtener información eficientemente, en muchas ocasiones además de ello resulta primordial que dicha información mantenga un cierto grado de confidencialidad ya que de llegar a las manos equivocadas podrían existir repercusiones negativas.

Por ejemplo, con el crecimiento de Internet durante las últimas décadas cada vez más el manejo de información confidencial se realiza por esta vía, debido a ello en los últimos años se ha incrementado la frecuencia de los incidentes de seguridad informática, así como su sofisticación, no es extraño escuchar sobre ataques informáticos entre países o sobre fraudes o intentos de fraude.

De manera que conforme crece la utilización de recursos informáticos, también crece la necesidad de mantener protegida tanto como sea posible nuestra información de las amenazas existentes. Es en este punto donde la Seguridad de la Información juega un papel primordial dentro de los sistemas informáticos actuales ya que su objetivo es evitar que ocurra de manera accidental o intencional, la transferencia, modificación, difusión o destrucción no autorizada de la información.

Es por ello que atendiendo a la importancia que la Seguridad de la Información tiene en nuestros días, la Facultad de Ingeniería de la Universidad Nacional Autónoma de México durante la revisión de planes de estudio del año 2005 estableció que uno de los módulos de salida que se integrarían a la carrera de Ingeniería en Computación sería el de “Redes y Seguridad” con el objetivo de proporcionar a los estudiantes una base sólida sobre este campo de conocimiento y mostrando de esta manera su compromiso con la sociedad en la formación de profesionistas capaces de enfrentar problemas actuales.

Por otra parte, gracias al apoyo de la División de Ingeniería Eléctrica de la misma dependencia, se han realizado varias publicaciones referentes a Seguridad Informática, estos textos fueron escritos por académicos del área quienes tienen conocimientos profundos sobre el tema, a la par, estudiantes han desarrollado como proyecto de tesis varios portales de Internet que alojan tutoriales sobre las asignaturas que conforman el módulo de seguridad, todo ello con el objetivo primordial de proporcionar a las nuevas generaciones material de estudio para su buen aprendizaje de las bases que les servirán en el desarrollo de su vida profesional.

Así, el **objetivo del presente trabajo de tesis** es el diseño y desarrollo del sistema tutorial para la asignatura de Criptografía impartida en la Facultad de Ingeniería de la

UNAM, el desarrollo de este sistema busca proporcionar a los estudiantes de la carrera de Ingeniería en Computación que cursen la asignatura de Criptografía, herramientas de estudio adicionales a las obtenidas en sus clases, por lo que en él se exponen temas sobre los fundamentos que rigen la Criptografía de nuestros días, ejemplos y referencias bibliográficas relacionadas con el tema, todo ello disponible a través de un sitio de Internet, el cual está alojado en el servidor Web del laboratorio de Redes y Seguridad (http://redyseguridad.fi-p.unam.mx) de tal modo que los estudiantes puedan reafirmar los conocimientos adquiridos en el aula, así como asimilar nuevos que les permitan una formación sólida, además, al ser un sistema disponible a través de Internet, puede ser benéfico para cualquier persona en el mundo que esté interesada en el tema y que de alguna forma le sea útil.

El sistema se desarrolló tomando en cuenta que tenía que abarcar aquellos temas que son la base de la Criptografía, en donde lo más importante sería el usuario, por ello cada uno de los temas se trataron de explicar de una manera resumida y lo más claramente posible, además se presenta al usuario imágenes descriptivas y ejemplos animados e interactivos para facilitar la comprensión de algunos de los algoritmos criptográficos presentados a lo largo del tutorial, todo ello por medio de una interfaz amigable para el usuario del tal manera que pueda acceder al contenido de cualquier tema desde cualquier punto del sitio en el que se encuentre.

Para desarrollar cada uno de los temas, se consultaron textos especializados en Criptografía, seguridad informática, redes de computadoras, comunicaciones, arquitectura de computadoras y otros, además de varias páginas de Internet relacionadas con la Criptografía que contienen información clave sobre el tema, de modo que todo aquél que acceda al sitio tiene la garantía de encontrar información veraz sobre cada uno de los temas.

El capítulo uno describe las actividades que conformaron el presente trabajo (recopilación de información y la utilización de un CMS para desarrollar el sitio Web) así como conceptos generales de ingeniería de software necesarios para el desarrollo del sitio. El capítulo dos presenta el enfoque y el contenido de la asignatura. El capítulo tres muestra el análisis, diseño y desarrollo del sitio Web. Por último en el capítulo cuatro se muestra como se implementó el sistema en el servidor así como el proceso de pruebas que se llevó a cabo.

CAPÍTULO 1

Características del Proyecto y Marco Teórico

En este capítulo se describen las actividades que conformaron el presente trabajo (recopilación de información y la utilización de un CMS para desarrollar el sitio Web) así como conceptos generales de ingeniería de software necesarios para el desarrollo del mismo.

1.1 RECOPILACIÓN DE INFORMACIÓN

Parte del presente trabajo fue realizar investigación bibliográfica sobre Criptografía, basándose en el temario de la asignatura "Criptografía" impartida en la Facultad de Ingeniería de la Universidad Nacional Autónoma de México de tal modo que se desarrollaron siete temas primordiales:

1. Panorama general de Criptografía
2. Técnicas clásicas de cifrado
3. Gestión de claves
4. Criptografía simétrica o de clave secreta
5. Criptografía asimétrica o de clave pública
6. Funciones hash
7. Firmas digitales

Estos siete temas se presentan en el capítulo dos del presente trabajo bajo el título "Fundamentos de Criptografía".

Se consultaron textos especializados en Criptografía, seguridad informática, redes de computadoras, comunicaciones, arquitectura de computadoras y otros, además de varias páginas de Internet realizando de este modo una recopilación extensa de información siempre apegándose al contenido de la asignatura.

Cabe señalar que toda la información fue ampliamente revisada por la asesora de la presente tesis (M.C. Ma. Jaquelina López Barrientos) especialista en temas de redes y seguridad, asegurando de este modo la depuración de información de cada uno de los temas.

1.2 UTILIZACIÓN DE UN CMS

1.2.1 DEFINICIÓN Y VENTAJAS DE UN CMS

Un sistema de administración de contenidos (Content Management Systems o CMS) es un software que se usa para facilitar la creación de sitios Web, ya sea en Internet o en una intranet. El CMS permite manejar de manera independiente el contenido y el diseño. Así, es posible ingresar la información que contendrán las páginas y darle en cualquier momento un diseño distinto sin tener que modificar los contenidos nuevamente [27].

Este sistema de software permite crear una estructura de soporte para administrar y crear de modo cooperativo contenidos. Con frecuencia, un CMS es una aplicación Web usada para gestionar sitios Web y contenidos Web.

Consta de una interfaz que controla una o varias bases de datos en donde se aloja el contenido del sitio, permite una fácil y controlada publicación de contenidos mediante estados de los documentos y habilitando distintos permisos a los usuarios.

Se contempla que el sitio de Criptografía reciba mantenimiento, actualización y mejoramiento constante con el objetivo de apegarse a las disposiciones establecidas por la UNAM para sitios institucionales [28], así como para cumplir con los estándares publicados por la W3C.

Para lograrlo se decidió utilizar un Sistema de Gestión de Contenidos (CMS) para la creación del sitio Web ya que ello permite garantizar el constante mantenimiento del mismo tomando en cuenta que quizá no siempre se tendrán recursos humanos para dedicarle tiempo, así que cualquier persona (dentro del área y con las facultades para hacerlo) pueda hacer dichos cambios sin la necesidad de tener conocimientos profundos sobre desarrollo Web.

Otra ventaja que se tiene al utilizar un CMS es la fácil implementación de nuevas características; aunque en esta fase del sitio solamente se pretende plasmar toda la información recopilada sobre Criptografía, a futuro se contempla un crecimiento para tener un sitio interactivo y dinámico; nuevas secciones como blogs, exámenes en línea, participación del usuario para contribuir con nuevos temas, etc. pueden ser agregadas fácilmente con ayuda del gestor de contenidos, además la actualización, respaldo y reestructuración del sitio Web son mucho más sencillas al tener todos los datos importantes del sitio y los contenidos en una base de datos estructurada en el servidor.

1.2.2 EJEMPLOS DE CMS

A continuación se presentan características principales de tres de los CMS´s más populares bajo la tecnología LAMP (Linux, Apache, MySQL, PHP) ya que ésta es la tecnología con la que cuenta el servidor donde se alojó el sitio.

CMS	CARACTERÍSTICAS PRINCIPALES
Drupal [24]	– Código libre bajo la licencia GPLv2 – Aunque se recomienda usar MySQL soporta otras bases de datos como PostgreSQL, SQLite, Microsoft SQL Server y Oracle – Puede funcionar con Apache o Microsoft IIS como servidor Web – Gran variedad de módulos que proporcionan funcionalidades potentes – Un robusto sistema de ayuda online y páginas de ayuda para los módulos del “núcleo”, tanto para usuarios como para administradores – Todo el contenido en Drupal es totalmente indexado en tiempo real y se puede consultar en cualquier momento – Un robusto entorno de personalización está implementado en el núcleo de Drupal. Tanto el contenido como la presentación pueden ser individualizados de acuerdo las preferencias

	definidas por el usuario – Drupal usa el mod_rewrite de Apache para crear URLs que son manejables por los usuarios y los motores de búsqueda – Los usuarios se pueden registrar e iniciar sesión de forma local o utilizando un sistema de autenticación externo como Jabber, Blogger, LiveJournal o otro sitio Drupal – Los administradores de Drupal no tienen que establecer permisos para cada usuario. En lugar de eso, pueden asignar permisos a un "rol" y agrupar los usuarios por roles – El sistema de control de versiones de Drupal permite seguir y auditar totalmente las sucesivas actualizaciones del contenido – Todo el contenido creado en Drupal tiene un enlace permanente asociado a él para que pueda ser enlazado externamente sin temor de que el enlace falle en el futuro – El contenido creado en Drupal es, funcionalmente, un objeto (nodo). Esto permite un tratamiento uniforme de la información, como una misma cola de moderación para envíos de diferentes tipos, promocionar cualquiera de estos objetos a la página principal o permitir comentarios -o no- sobre cada objeto – El sistema de temas de Drupal separa el contenido de la presentación permitiendo controlar o cambiar fácilmente el aspecto del sitio Web – Exporta el contenido en formato RDF / RSS para ser utilizado por otros sitios Web – La API de Blogger permite que un sitio Drupal sea actualizado utilizando diversas herramientas, que pueden ser "herramientas Web" o "herramientas de escritorio" que proporcionen un entorno de edición más manejable – Incluye un potente agregador de Noticas para leer y publicar enlaces a noticias de otros sitios Web – Proporciona opciones para crear un sitio en varios idiomas – Puede mostrar en las páginas Web de administración informes sobre referrals (enlaces entrantes), popularidad del contenido, o de cómo los usuarios navegan por el sitio – Toda la actividad y los sucesos del sistema son capturados en un "registro de eventos", que puede ser visualizado por un administrador – Incorpora un mecanismo de control de congestión que permite habilitar y deshabilitar determinados módulos o bloques dependiendo de la carga del servidor – El mecanismo de cache elimina consultas a la base de datos incrementando el rendimiento y reduciendo la carga del servidor
Joomla! [25]	– Código libre bajo la licencia GPLv2 – Soporte para MySQL aunque se ha anunciado su pronta capacidad para soportar otras bases de datos – Puede funcionar con Apache o Microsoft IIS como servidor Web – Gran variedad de módulos que proporcionan funcionalidades potentes – Cuenta con un robusto sistema de ayuda online

	– Cuenta con un sistema de registro que permite a los usuarios configurar opciones personales. Existen nueve grupos de usuarios con diversos permisos tales como acceder, editar, administrar, publicar, etc. – Soporta múltiples protocolos de autenticación incluyendo LDAP, OpenID y Gmail, permitiendo así a los usuarios usar su información de cuenta existente para agilizar el proceso de registro – Soporte internacional para muchos lenguajes y codificación UTF-8 – El Media Manager es una herramienta para gestionar fácilmente archivos o carpetas, está integrado en el editor de artículos el cual permite tomar imágenes y otros archivos en cualquier momento – Es fácil de instalar banners en un sitio Web utilizando el gestor de anuncios, a partir de la creación de un perfil de cliente. – El gestor de contacto permite a los usuarios encontrar a la persona correcta y su información de contacto – Se pueden crear encuestas con múltiples opciones para obtener información sobre los usuarios – Una función de camuflaje de correo electrónico protege direcciones de correo electrónico de los spambots – Se puede crear contenido con el editor WYSIWYG, permitiendo a novatos combinar texto e imágenes de una manera fácil. – Exporta el contenido en formato RDF / RSS para ser utilizado por otros sitios Web – El gestor de menús permite crear fácilmente menús de cualquier tipo totalmente independientes de la estructura del contenido lo que permite ponerlos en cualquier lugar y con el estilo que se desee – El gestor de plantillas permite de una manera fácil y coherente que los sitios tengan el diseño que se desee e independiente del contenido – Tiene integrado un sistema de ayuda – Manejo de cache para carga rápida – Modo de depuración y presentación de informes de error disponibles para el administrador del sistema para solucionar problemas – La capa FTP permite operar con archivos como por ejemplo en la instalación de extensiones sin necesidad de dar permisos de escritura a carpetas y archivos – Los administradores pueden comunicarse eficiente y rápidamente con los usuarios vía mail – Se pueden integrar los servicios XML-RPC con las APIs de Blogger y Joomla
WordPress [23]	– Código libre bajo la licencia GPLv2 – En un principio estaba orientado a la publicación de blogs, actualmente su evolución contempla cada vez más su uso como CMS genérico – Soporte para MySQL

	– Se recomienda usar Apache o Nginx como servidor Web aunque se puede utilizar cualquiera que soporte MySQL y PHP – Gran cantidad plugins que dan mayor potencia a WordPress – Cuenta con una gran variedad de plantillas, además de que se pueden personalizar fácilmente – Estadísticas integradas que proporcionan información sobre las visitas al sitio – Es sencillo crear contenidos y agregar elementos como imágenes o videos, además se guardan continuamente los contenidos a medida que se va escribiendo – Utiliza Akismet, para prevenir el spam – Existe una comunidad de usuarios que colabora en los foros y un equipo de soporte. Tiene una buena documentación y un formulario de contacto con el soporte – Define diferentes roles para diferentes usuarios – Se puede importar contenido de Blogger, LiveJournal, Movable Type, TypePad o de otro blog WordPress – Disponible en más de sesenta idiomas – Tiene una característica llamada “páginas” que permite crear páginas Web estáticas con suma facilidad permitiendo de esta manera crear sitios completos – Tiene aplicaciones móviles para Android, iOS, Blackberry, Nokia, Windows Phone 7 y WebOS – URLs que son manejables por los usuarios – Exporta el contenido en formato RDF / RSS para ser utilizado por otros sitios Web

1.2.3 BENEFICIOS DE JOOMLA!

El CMS Joomla tiene los siguientes beneficios:

- Permite la colaboración de varios usuarios en el mismo trabajo debido a la gestión dinámica y fácil de usuarios y permisos.
- Hace énfasis especial en la usabilidad ya que permite organizar eficientemente los contenidos de un sitio en categorías, lo que facilita la navegabilidad para los usuarios y permite crear una estructura sólida, ordenada y sencilla para los administradores.
- Cuenta con una interfaz gráfica amigable, esto es primordial para que cualquier persona de mantenimiento al sitio una vez que éste se encuentre en producción.
- Permite escalabilidad e implementación de nuevas funcionalidades al sitio Web debido a su amplia variedad de módulos y extensiones, dando lugar a la construcción de aplicaciones potentes.
- Joomla es desarrollado y mantenido por una amplia comunidad, es por ello que cuenta con actualizaciones constantes, lo que garantiza mayor seguridad para las aplicaciones así como el seguimiento de estándares emitidos por la W3C.
- Es una solución de código abierto y está disponible libremente para cualquiera que desee utilizarlo.

1.3 FUNDAMENTOS DE INGENIERÍA DE SOFTWARE

1.3.1 CONCEPTOS GENERALES

- **Software**

Es el conjunto de los programas de cómputo y todos aquellos procedimientos, reglas, documentación y datos que son indispensables para hacer que dichos programas operen correctamente.

- **Ingeniería de software**

Es la disciplina de ingeniería que comprende los procesos técnicos del desarrollo de software, gestión de proyectos y el desarrollo de herramientas, métodos y teorías de apoyo a la producción de software desde la etapa de especificaciones del sistema hasta el mantenimiento de éste una vez que ya se está utilizando. En general se puede decir que comprende las formas prácticas para desarrollar y entregar un software de calidad.

- **Ciclo de vida del desarrollo de software**

También llamado proceso de software o fases del desarrollo de software es el conjunto de actividades y resultados asociados que dan como resultado un software de calidad.
Existen seis fases principales que conforman el ciclo de vida del desarrollo de software:

1. Análisis
2. Diseño
3. Desarrollo
4. Pruebas
5. Implementación
6. Mantenimiento

Estas fases pueden organizarse de diferentes formas y describirse en diferentes niveles de detalle para diferentes tipos de software.

En la sección 1.3.2 (Ciclo de vida del desarrollo de software) del presente capítulo se explica brevemente cada una de estas fases.

- **Gestión de proyectos**

La gestión de proyectos es una parte fundamental de la ingeniería de software ya que es responsable de la planificación y temporalización del desarrollo de proyectos.

Las actividades que se realizan son las siguientes:

- Redacción de la propuesta
- Calendarización del proyecto
- Estimación de costes del proyecto
- Supervisión y revisión del proyecto
- Selección y evaluación del personal
- Redacción y presentación de informes

1.3.2 CICLO DE VIDA DEL DESARROLLO DE SOFTWARE

- **ANÁLISIS**

En esta fase se define el software que se va a producir y las restricciones sobre su operación.

Una vez establecido que es lo que se va a producir se definen los requerimientos, los cuales especifican que es lo que el sistema debe hacer así como sus propiedades esenciales y deseables.

La figura 1.1 muestra las actividades del proceso de requerimientos, cabe señalar que no se llevan a cabo de forma estrictamente secuencial.

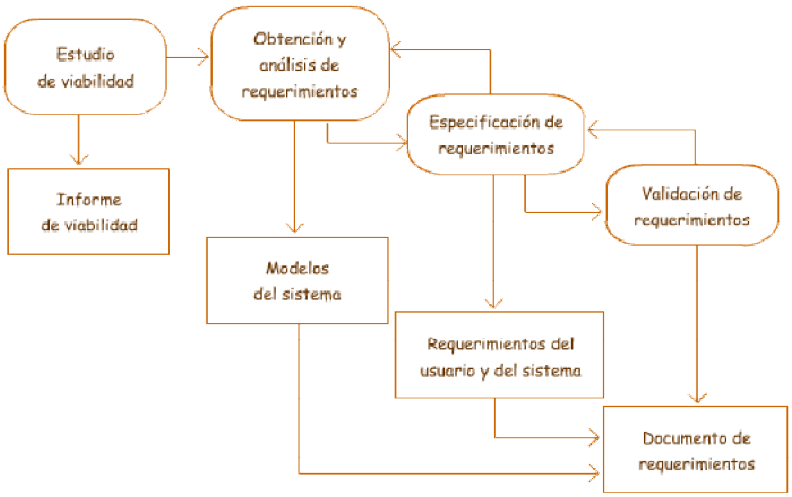


FIGURA 1.1 Proceso de requerimientos

Estudio de viabilidad: se analiza si el sistema es rentable y si es posible de desarrollar con el presupuesto destinado y con las tecnologías existentes de software y hardware.

Obtención y análisis de requerimientos: se obtienen requerimientos a través de usuarios potenciales del sistema, y estudiando los sistemas existentes en caso de existir, de este análisis puede resultar el desarrollo de uno o más modelos del sistema y prototipos cuya finalidad es ayudar al analista a comprender el sistema a especificar.

Especificación de requerimientos: se plasman en un documento tanto los requerimientos que hizo el cliente como aquellos requerimientos del sistema los cuales son una descripción más detallada de la funcionalidad a proporcionar.

Validación de requerimientos: se comprueba la veracidad, consistencia y completitud de los requerimientos y se hacen los ajustes o cambios en caso de ser necesarios.

Con la definición de los requerimientos se obtiene:

- Las funciones básicas que el sistema debe proporcionar.

- Identificación de usuarios y roles.
- Propiedades del sistema tales como disponibilidad, rendimiento y seguridad.
- Alcance y características que no debe mostrar el sistema.

- **DISEÑO**

El diseño del sistema es una descripción de la estructura del software que se va a implementar, para ello se elige una arquitectura de software la cual comprende los componentes del sistema, la relación entre ellos y el entorno que orienta la composición y restricciones de dichos elementos.

Además es en esta fase donde se eligen las tecnologías que más convengan para desarrollar el sistema.

La figura 1.2 muestra un modelo general de las actividades realizadas en el proceso de diseño, dichas actividades se pueden entrelazar y ser adaptarlas de acuerdo a las necesidades del sistema.

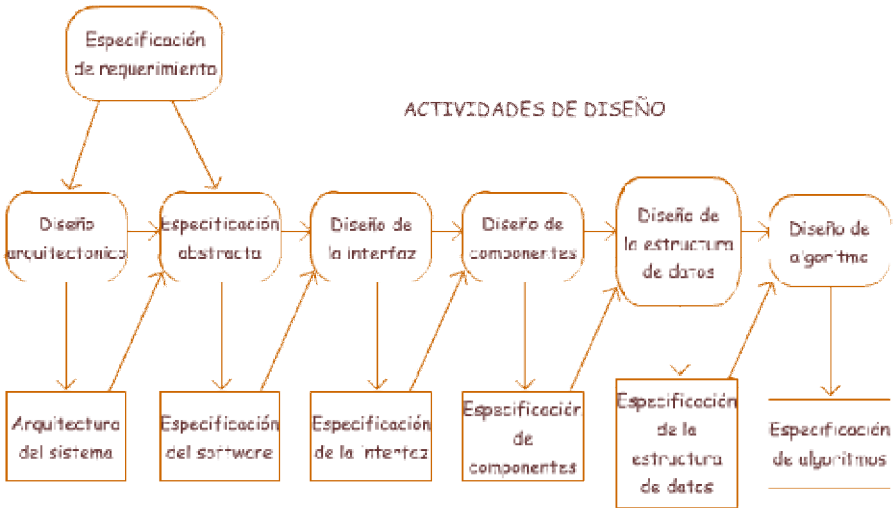


FIGURA 1.2 Proceso de diseño

Diseño arquitectónico: se identifican y documentan los subsistemas que forman el sistema.

Especificación abstracta: se realiza una especificación abstracta de cada uno de los subsistemas y se identifican sus restricciones.

Diseño de la interfaz: se diseña y documenta la interfaz que cada subsistema tendrá con otros subsistemas.

Diseño de componentes: se identifican funcionalidades de los componentes.

Diseño de la estructura de datos: se diseña a detalle y se especifica la estructura de datos que deberá utilizarse en el desarrollo del sistema.

Diseño de algoritmo: se diseñan a detalle y especifican los algoritmos utilizados para proporcionar los servicios.

El proceso que sigue cada uno de los subsistemas se resume en la figura 1.3:

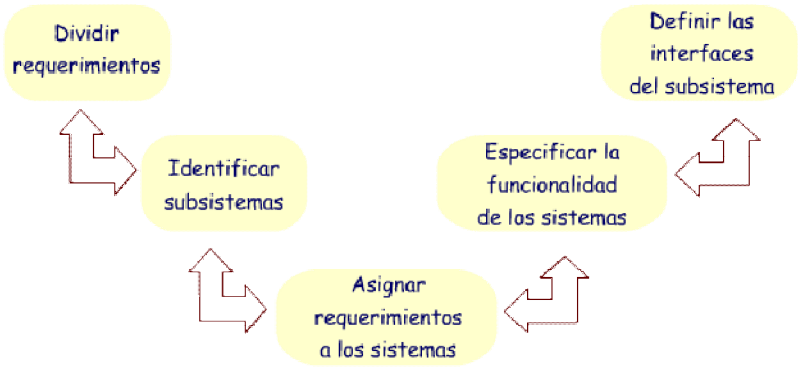


FIGURA 1.3 Proceso para cada uno de los subsistemas

- **DESARROLLO**

Basándose en el diseño, en esta fase se realizan las tareas de programación de cada uno de los subsistemas, también se desarrolla la base de datos que se deberá ocupar.

- **PRUEBAS**

Una vez desarrollado un subsistema éste debe ser probado individualmente y asegurarse que su funcionamiento es el esperado, a este tipo de pruebas se les llama unitarias.

Posteriormente deben integrase todos los subsistemas para obtener el sistema completo y realizar las llamadas pruebas de integración, existen varios enfoques para realizar dicha integración, uno de ellos es el llamado de “big bang” el cual consiste en integrar todos los subsistemas para así probar el sistema completo, otro enfoque es el creciente donde los subsistemas se integran uno a uno lo cual tiene como ventaja poder localizar más fácilmente los errores.

Una vez que el sistema ya está integrado se realizan las pruebas del sistema y es donde se verifica que el sistema cumple con los requerimientos del cliente.

Por último se realizan las pruebas de aceptación, las cuales consisten en probar el sistema con datos entregados por el cliente y se realizan antes de que se acepte que el sistema se ponga en funcionamiento.

La figura 1.4 muestra el proceso de pruebas en tres etapas: pruebas de cada uno de los componentes o pruebas unitarias, de integración del sistema y del sistema con los datos del cliente, cabe mencionar que cuando se descubren defectos en una de las etapas se puede requerir repetir otras etapas del proceso de pruebas tal y como muestra la figura.

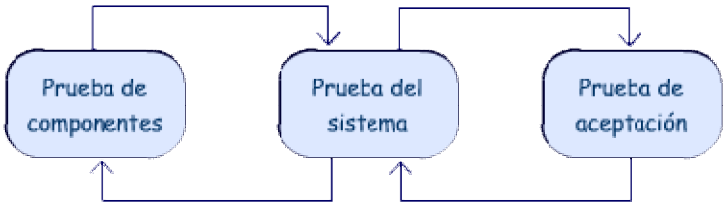


FIGURA 1.4 Proceso de pruebas

- **IMPLEMENTACIÓN**

Se prepara la infraestructura para realizar la instalación del sistema en donde quedará alojado de manera definitiva, aspectos como manejadores de bases de datos, seguridad, sistema operativo, e incluso el hardware deben ser atendidos en esta fase.

Una vez realizada la instalación se realizan las pruebas finales del sistema que implican revisar el óptimo funcionamiento del mismo ya en el equipo final, también se realizan actividades como capacitación del uso del sistema a los usuarios finales y entrega de manuales técnicos y de usuario.

- **MANTENIMIENTO**

Esta fase se refiere a lo que pasa después de que el sistema final es entregado; existen varias razones por los que el sistema puede ser modificado, por ejemplo para adaptarlo a los cambios requeridos por el cliente, por actualización de software o hardware, o bien por las necesidades del mercado.

La figura 1.5 muestra el proceso de mantenimiento que se debería seguir cuando el sistema requiera alguna modificación:

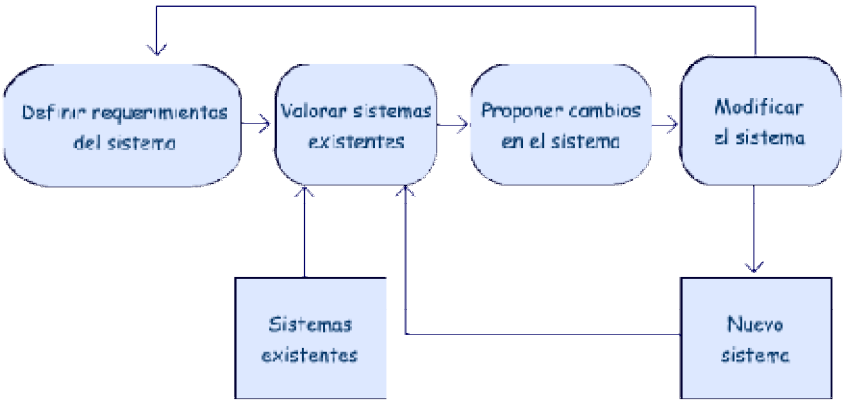


FIGURA 1.5 Proceso de mantenimiento

1.3.3 PARADIGMAS DEL DESARROLLO DE SOFTWARE

- **EL MODELO EN CASCADA**

Considera las fases que se muestran en la figura 1.6 y las representa como fases separadas transformándolas de este modo en actividades fundamentales de desarrollo. La siguiente fase no debe comenzar hasta que la fase previa quede terminada y aprobada.

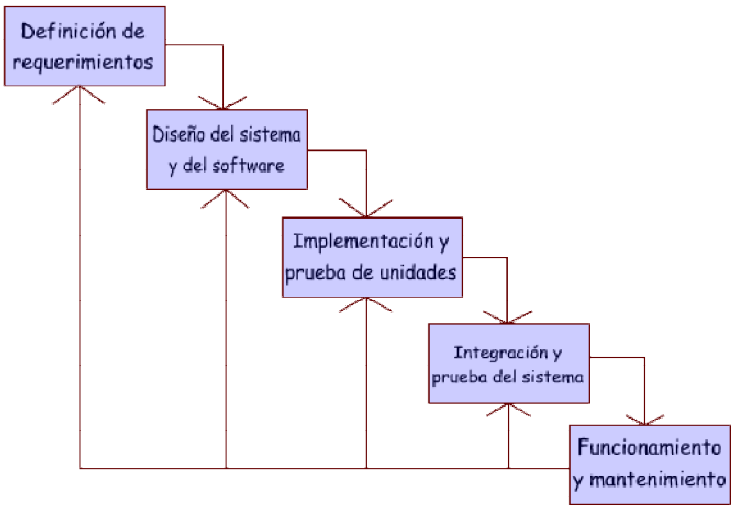


FIGURA 1.6 Modelo en cascada

- **DESARROLLO EVOLUTIVO**

Este enfoque entrelaza actividades de especificación, desarrollo y validación, tal y como se muestra en la figura 1.7.
Se desarrolla un sistema inicial de manera rápida partiendo de especificaciones abstractas, a partir de dicho sistema inicial se obtienen diferentes versiones ya que el cliente va haciendo comentarios y el sistema se va refinando basándose en los mismos.

FIGURA 1.7 Desarrollo evolutivo

- **INGENIERÍA DE SOFTWARE BASADA EN COMPONENTES**

Este enfoque se basa en la existencia de gran parte de los componentes del sistema a desarrollar, de este modo se enfoca a la integración de dichos componentes y no a desarrollarlos desde el principio, la figura 1.8 muestra esta técnica.

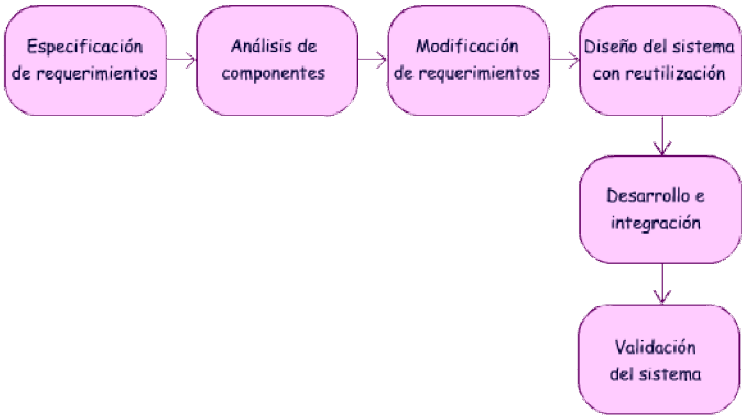


FIGURA 1.8 Ingeniería de software basada en componentes

Análisis de componentes: se buscan componentes existentes para cubrir los requerimientos.

Modificación de requerimientos: se analizan los requerimientos para ver si los componentes encontrados cumplen con las peticiones del cliente, de ser necesario y posible los componentes se modifican para que así sea; cuando no es posible realizar una modificación se regresa a la etapa anterior para buscar alternativas.

Diseño del sistema con reutilización: se diseña o se reutiliza un marco de trabajo para el desarrollo del sistema.

Desarrollo e integración: el software que no se pudo obtener externamente se desarrolla y los componentes que si se encontraron se integran.

1.3.4 PROCESOS DE LA INGENIERÍA DE SOFTWARE

Un proceso de la ingeniería de software es un enfoque estructurado para el desarrollo de software con el fin de facilitar la producción de software de alta calidad de una forma costeable. Incluye:

Descripciones del modelo del sistema: un modelo del sistema es la descripción simplificada del ciclo de vida del desarrollo de software presentada desde una perspectiva específica. Los modelos pueden incluir fases del desarrollo del software así como productos de software y el papel de las personas involucradas en la ingeniería del software. La mayoría de dichos modelos se basan en uno de los tres modelos generales o paradigmas de desarrollo de software (modelo en cascada, desarrollo iterativo e ingeniería basada en componentes) éstos se explican en la sección 1.3.3 (Paradigmas de desarrollo de software) del presente capítulo.

Notaciones: la notación utilizada en el modelo del sistema.
Reglas: restricciones que siempre aplican a los modelos del sistema.
Recomendaciones: sugerencias para obtener un modelo del sistema bien organizado.

Guías en el proceso: descripción de las actividades que deben seguirse para desarrollar los modelos del sistema y la organización de esas actividades.

- **MÉTODOS BASADOS EN UN PLAN**

Identifican etapas separadas dentro del ciclo de vida de desarrollo de software en donde los resultados de una fase son la entrada para el desarrollo de la siguiente, existen documentos formales para comunicarse entre etapas.

Los requerimientos deben conocerse de manera temprana y ser estables, lo cual da lugar a planear la fase de diseño y desarrollo como una serie de incrementos.

El Proceso Unificado de Rational es un ejemplo de este tipo de métodos.

- **MÉTODOS ÁGILES**

Fueron diseñados para apoyar al desarrollo de sistemas donde los requerimientos son muy cambiantes durante la fase de desarrollo, están pensados para entregar software funcional de forma rápida, se centran en el desarrollo y la implementación del software.

Los principios de los métodos ágiles son los siguientes:

Participación del cliente: los clientes deben ser implicados fuertemente en todo el desarrollo, proporcionando y priorizando nuevos requerimientos además de la evaluación de las iteraciones del sistema.

Entrega incremental: el software se desarrolla en incrementos, donde el cliente especifica los requerimientos que se deben incluir en cada incremento.

Personas, no procesos: a los miembros del equipo se les debe dejar desarrollar sus propias formas de trabajar, sin proceso formales.

Aceptar el cambio: el sistema se diseña para dar lugar a cambios en los requerimientos.

Mantener la simplicidad: se debe centrar en la simplicidad tanto en el software a desarrollar como en el proceso de desarrollo. Donde sea posible se debe eliminar la complejidad del sistema.

Ejemplos de métodos ágiles son: Extreme programming y Scrum.

1.4 DESCRIPCIÓN DEL PROCESO DE DESARROLLO DEL SITIO WEB DE CRIPTOGRAFÍA

El sitio Web se desarrolló bajo el modelo en cascada, la figura 1.9 muestra las fases que componen el ciclo de vida del desarrollo del sitio Web.

1. **Análisis**: determinar el objetivo, alcances, límites, el público al que estará dirigido así como el contenido y la funcionalidad del sitio.

2. **Diseño**: diseñar el sitio que cumpla con los requerimientos determinados en la fase de análisis.
3. **Desarrollo y pruebas**: realización del sitio valiéndose de la tecnología seleccionada y a la par ir probando componentes que se vayan obteniendo para realizar las modificaciones necesarias y al final hacer una prueba integral.
4. **Implementación**: Subir el sitio al servidor, realizar las últimas pruebas del sitio completo para ponerlo a disposición de los usuarios.
5. **Mantenimiento**: Reparar, mejorar y renovar el sistema cuando sea necesario.

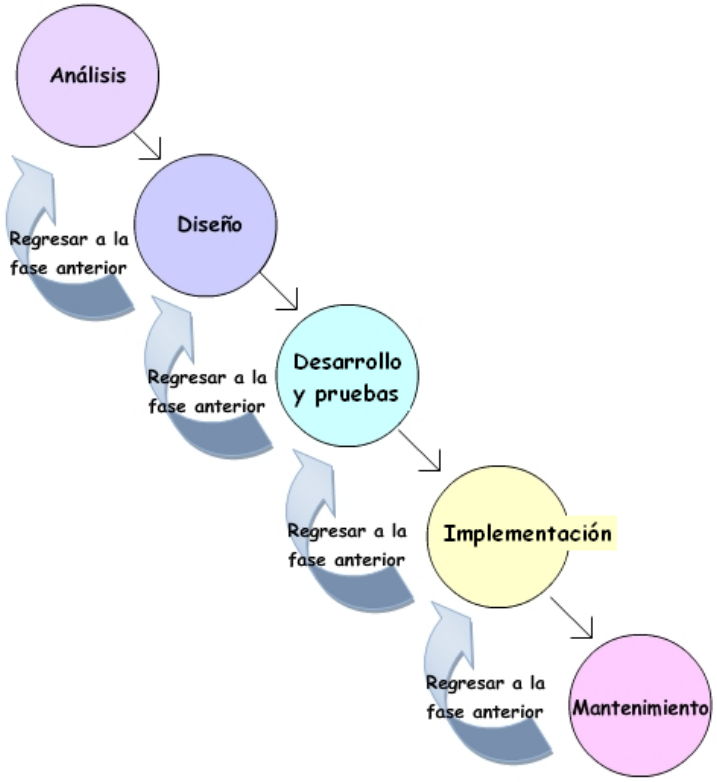


FIGURA 1.9 Ciclo de vida del desarrollo de un sitio Web

En el capítulo tres del presente trabajo se explican las fases de análisis, diseño y desarrollo, mientras que en el capítulo cuatro se explica el proceso de prueba y la implementación.

Dentro de este marco de desarrollo cabe señalar que se implementaron algunas prácticas de los procesos ágiles:

- Simplicidad: se buscó la forma más simple de realizar el trabajo para obtener los resultados esperados, ello implicaba un diseño lo más simple posible para garantizar los tiempos de desarrollo (no se contaba con un tiempo establecido, pero si se buscaba desarrollar en el menor tiempo posible) y mantenimiento del

sistema; como resultado se tuvo la determinación de desarrollar el sitio utilizando un CMS.

- Se considera el diseño y la implementación como actividades centrales en el desarrollo del software. Dado que lo más importante es obtener un resultado de calidad rápidamente, la atención se centra en estas fases, sin embargo, considerar el diseño e implementación como actividades centrales no implica que actividades como análisis y pruebas no se lleven a cabo.
- Los requerimientos y el diseño se desarrollaron en conjunto, una vez recabados los requerimientos estos se proyectaron en prototipos rápidos para pedir retroalimentación e ir obteniendo el producto final. Dichos prototipos consistieron en desarrollo rápido en HTML así como presentación de plantillas que cumplieran con los requerimientos.
- Se realizaron entregas pequeñas y frecuentes para lograr evaluación y corrección de fallos.
- Al realizar entregas pequeñas se tenía que realizar una integración continua de secciones, de tal manera de que gradualmente se iba construyendo el sistema final.

CAPÍTULO 2

Fundamentos de Criptografía

En este capítulo se presentan los antecedentes históricos de la Criptografía y su evolución a través del tiempo, se explican las técnicas clásicas de cifrado, los algoritmos simétricos y asimétricos de la Criptografía, además se exponen algunos aspectos sobre seguridad de la información dentro del mundo del cómputo y las redes.

2.1 PANORAMA GENERAL

2.1.1 CONCEPTO DE CRIPTOGRAFÍA

La Criptología (del griego criptos= oculto y logos= tratado, ciencia) es la ciencia que trata las escrituras ocultas, está comprendida por la Criptografía, el Criptoanálisis y la Esteganografía (Figura 2.1).

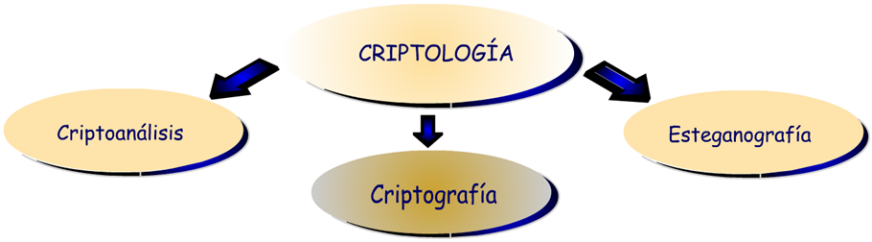


FIGURA 2.1 Ramas de la Criptología

Las raíces etimológicas de la palabra Criptografía son criptos (oculto), y graphos (escritura). Una definición clásica de Criptografía es la siguiente:

Arte de escribir mensajes en clave secreta o enigmáticamente

Anteriormente la Criptografía era considerada como un arte pero en la actualidad se considera una ciencia gracias a su relación con la estadística, la teoría de la información, la teoría de los números y la teoría de la complejidad computacional.

La **Criptografía** es la ciencia que se encarga del estudio de técnicas para transformar la información a una forma que no pueda entenderse a simple vista; sin embargo, el objetivo de la Criptografía no es sólo mantener los datos secretos, sino también protegerlos contra modificación y comprobar la fuente de los mismos.

El **Criptoanálisis** es la ciencia que se ocupa del análisis de un texto cifrado para obtener la información original sin conocimiento de la clave secreta, esto es, de forma ilícita rompiendo así los procedimientos de cifrado establecidos por la Criptografía, por lo que se dice que Criptoanálisis y Criptografía son ciencias complementarias pero contrarias.

La **Esteganografía** por su parte, estudia la forma de ocultar la existencia de un mensaje. Esta ciencia consiste en esconder en el interior de un mensaje, otro mensaje secreto, el cual sólo podrá ser entendido por el emisor y el receptor y pasará inadvertido para todos los demás.

2.1.2 HISTORIA DE LA CRIPTOGRAFÍA

La Criptografía nace debido a que el hombre a lo largo del tiempo se ha visto en la necesidad de comunicar información confidencial a otros individuos ya sea por motivos militares, diplomáticos, comerciales, etc., en donde mantener la información en secreto es la pauta para conservar la integridad de un individuo o en ocasiones de una comunidad completa.

Una de las primeras formas utilizadas para ocultar la información fue una técnica que consistía en realizar orificios sobre las letras del mensaje secreto para pasar sobre ellos un tipo de tejido que servía para ocultar dicho mensaje.

Alrededor del año 1500 a.C. los comerciantes asirios utilizaban tablillas de arcilla en donde tallaban escritos y algunas imágenes que establecían la forma de llevar a cabo sus transacciones comerciales, muchas veces dichas tablillas se colocaban en el interior de un contenedor de arcilla el cual era sellado.

Durante el siglo V a.C. los griegos crearon un instrumento para cifrar mensajes. Dicho instrumento es conocido como *Scítala de los Lacedemonios* y consistía en un cilindro de madera en el cual se enrrollaba una cinta de papiro o tela. Una vez enrollado el papiro se escribía el mensaje a lo largo de cada una de las generatrices del cilindro. Después se desenrollaba dicho papiro y era mandado con un mensajero al receptor, quien contaba con un cilindro con las mismas medidas que el del emisor, por lo que podía volver a enrollar el papiro en su cilindro y recuperar el mensaje original. Es importante mencionar que el mensaje en el papiro sin estar enrollado en el cilindro resultaba confuso e incoherente. En la figura 2.2 se muestra uno de estos cilindros.

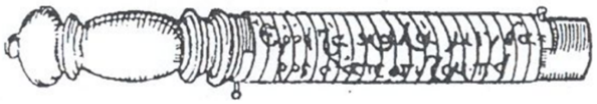


FIGURA 2.2 Escitalo de los Lacedemonios

Entre los años 500 y 600 a.C., escribanos hebreos utilizaban un alfabeto al revés (como ejemplo está el Libro de Jeremías), es decir cuando querían escribir la primera letra del alfabeto escribían la última y cuando querían escribir la última utilizaban la primera y así sucesivamente con todo el alfabeto, a esta forma de escribir se le llama código espejo o Atbash.

A mediados del siglo II a.C. surgió un procedimiento de cifrado atribuido al historiador griego Polybios, este procedimiento de cifrado consistía en la sustitución de un carácter por un par de caracteres que le correspondían según una tabla que se diseñaba con este propósito. En el subtema (Técnicas clásicas de cifrado) del presente capítulo se explica a detalle este método.

Cincuenta años más tarde, en el siglo I a.C. aparece un nuevo procedimiento de cifrado, el cual es conocido como *cifrador del César*, debido a que era usado por el militar y político romano Julio César. Este método consistía en sustituir cada carácter del mensaje original por otro situado tres posiciones después de él en un determinado alfabeto. En el subtema (Técnicas clásicas de cifrado) del presente capítulo se explica a detalle este método.

A finales del siglo I a.C. y principios del siglo I d.C., Augusto, el primer emperador de Roma propuso una nueva forma de enmascar los mensajes, ésta consistía en escribirlos en una tableta que posteriormente era cubierta con cera quedando así oculta la información.

Otra técnica interesante practicada en la antigua Roma, consistía en enviar el mensaje por medio de un esclavo quien era rapado y se escribía sobre su cabeza un mensaje, una vez que le crecía el pelo, era enviado con el receptor el cual debía rapar nuevamente al esclavo para obtener el mensaje. Era una práctica común que al esclavo se le cortara la lengua para que en el caso de ser interceptado por un contrario no pudiera decir que llevaba un mensaje escrito en la cabeza.

Durante la persecución de los primeros cristianos, éstos se vieron obligados a expresar la idea de un ser superior por medio de símbolos (llamados símbolos apostólicos), principalmente utilizaron marcas de talleres monetarios con el fin de que los perseguidores no pudieran relacionarlos con el cristianismo ya que eran símbolos comunes para todos. Los sacerdotes de aquella época mandaron poner dichos símbolos en todas las cosas relacionadas con la iglesia y se tienen vestigios de que se utilizaron hasta el siglo XVII a pesar de que en el año 313 el imperio romano promulgó el Edicto de Milán, el cual establecía el fin de las persecuciones a los cristianos. Ejemplos de estos símbolos son: un círculo representaba la eternidad, una línea horizontal representaba a Jesús, mientras que una X significaba Cristo.

Roger Bacon (1214-1294) un destacado filósofo y naturalista inglés en algunas de sus obras expone los conocimientos sobre Criptografía que se tenían hasta la época así como algunas observaciones.

Se tienen conocimientos de que para el año 1300 los árabes ya habían desarrollado alrededor de siete procedimientos de cifrado, los cuales se enlistan a continuación:

1. Reemplazar unas letras por otras
2. Escribir palabras al revés
3. Invertir letras alternadas en el texto del mensaje original
4. Dar a las letras un valor numérico y escribir dichos valores con símbolos
5. Reemplazar cada letra con otras dos de forma que la suma de sus valores numéricos fuera igual al valor numérico de la letra reemplazada
6. Sustituir cada letra con el nombre de una persona o un objeto
7. Sustituir las letras por signos lunares, pájaros, flores u otros signos inventados

Entre 1375 y 1383 Cicco Simonetta, consejero y secretario de los duques Sforza en Milán, desarrolla su obra llamada Liber Zifrorum, la cual es considerada como el tratado de descifrado más antiguo que se conoce. En dicha obra estudia y analiza diversos sistemas criptográficos.

A petición del Antipapa Clemente VII, Gabrieli di Lavinde en 1379, desarrolló un manual de Criptografía que consistía en utilizar determinados códigos para sustituir palabras del mensaje en claro.

León Battista Alberti quien fue secretario pontificio de la corte romana, en 1466 inventó un disco de cifrado utilizando discos concéntricos divididos en 24 casillas en donde cada una de ellas contenía un carácter (ver figura 2.3).

FIGURA 2.3 Cifrador de Leon Battista Alberti (1446)

El funcionamiento de este disco se explica en el subtema (Técnicas clásicas de cifrado) del presente capítulo.

En el siglo XVI el historiador y religioso benedictino alemán Trithemius, publica su obra "Poligraphiae" en donde desarrolla varios procedimientos de cifrado entre los que se encuentra la sustitución de letras por palabras, las cuales eran escogidas de tal modo que su yuxtaposición formarán un texto entendible, y que al leerlo no se sospechará que había un mensaje secreto oculto.

En ese mismo siglo el matemático Girolamo Cardano inventó el procedimiento de la trepa, mejor conocido como máscara rotativa, que consiste en una tableta con algunas perforaciones la cual se coloca sobre otra tableta que contiene distintas letras, de esta manera se va obteniendo el mensaje visualizando las letras a través de las perforaciones y girando en sentido horario la máscara (que es la tableta perforada), este método se ejemplifica en el subtema (Técnicas clásicas de cifrado) del presente capítulo.

En 1593, Giovanni Battista De la Porta modificó el disco de cifrado de Alberti sustituyendo el alfabeto del disco interior por símbolos extraños (Figura 2.4).

FIGURA 2.4 Cifrador de Giovanni Battista de la Porta

En 1595 el francés Blaise Vigenère inventó un método de cifrado que consistía en asignar un número a cada letra del alfabeto y sumar los números correspondientes a una clave con los del mensaje para obtener el criptograma. Este método es explicado a detalle en el subtema (Técnicas clásicas de cifrado) del presente capítulo.
Durante el siglo XVI fueron muy utilizados los libros de código para cifrar los mensajes, como ejemplo de ello están los libros de código de Felipe II (ver figura 2.5).

FIGURA 2.5 Libro de código de Felipe II

Estos libros consistían en poner las letras del alfabeto, en grupos de dos letras y tres letras e incluso palabras completas más usuales del lenguaje en un rectángulo para asignarles uno o varios símbolos extraños. De este modo tanto emisor como receptor debían poseer el libro para poder cifrar o descifrar los mensajes.

En 1790 Thomas Jefferson creó un cilindro formado por varios discos coaxiales en donde cada uno tenía escrito en la parte exterior un alfabeto, tal y como se muestra en la figura 2.6. Cada disco se ajustaba de tal modo que en una generatriz del cilindro se formara el mensaje en claro y el criptograma se obtenía de cualquiera de las otras generatrices.

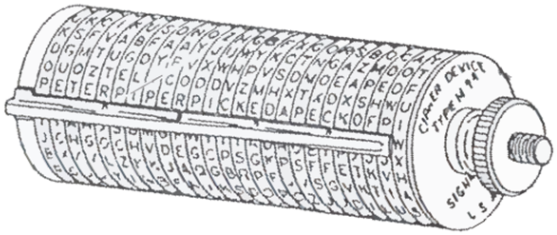


FIGURA 2.6 Cifrador de discos de Jefferson (1790)

En 1854 Sir. Charles Wheatstone diseñó un método de cifrado llamado Playfair, este método era parecido al de Polybios solo que ahora en vez de que cada carácter se sustituyera por dos caracteres sólo se sustituía por uno. Este método se explica en el subtema (Técnicas clásicas de cifrado) del presente capítulo.

Para 1867 Wheatstone había ideado un nuevo disco de cifrado que en realidad se trataba de una versión mecánica del disco de Alberti; esta nueva versión ocupaba en el disco exterior el alfabeto inglés más un signo de "+" colocados de manera ordenada en sentido de las manecillas del reloj y el disco inferior tenía solamente 26 casillas con el alfabeto colocado de manera desordenada (Figura 2.7). Las agujas estaban engranadas de tal manera que cuando la externa giraba 27 posiciones, la interna lo hacia 26, estableciendo de esta manera una correspondencia entre los dos alfabetos.

FIGURA 2.7 Cifrador de Wheatstone (1867)

En 1890 Étienne Bazeries, tomando como base el cilindro de Jefferson creó un cilindro que constaba de 20 discos coaxiales con 25 letras en cada uno de ellos (figura 2.8), la diferencia con el cilindro de Jefferson es básicamente la manera de cifrar el mensaje, ya que el disco de Bazeries al tener un disco adicional con números impresos, el criptograma del mensaje en claro podía conformarse con letras de varias generatrices estableciendo el número de la generatriz con que se cifraba cada carácter del mensaje en claro.

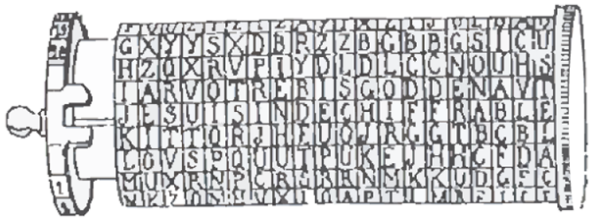


FIGURA 2.8 Cifrador de Bazeries (1890)

A principios de siglo XX, William F. Friedman considerado como el padre de la Criptografía moderna, publicó un estudio llamado "The index of coincidence and its applications in Criptography" uno de los primeros trabajos en donde se aplicaron principios matemáticos en la Criptografía.

En 1917 Gilbert Vernam desarrolló un algoritmo de cifrado que lleva su nombre (*cifrado Vernam*) el cual lleva al límite la idea del cifrado de Blaise Vigenère. Este método de cifrado se explica en el subtema (Técnicas clásicas de cifrado) del presente capítulo.

En 1923 Arthur Sherbius, un ingeniero alemán, dio a conocer una máquina llamada *Enigma*, se trataba de una máquina que en su exterior parecía una máquina de escribir común (ver figura 2.9) pero en su interior estaba compuesta por un mecanismo que trasformaba la letra tecleada en otra, estaba compuesta por un conjunto de ruedas cuyas caras tenían contactos eléctricos entre sí (figura 2.10).

Existieron varios modelos de esta máquina; el primero, contaba con cuatro ruedas con un alfabeto de 28 letras cada una, podía cambiarse entre el modo de cifrado y el de descifrado, además tenía un sistema de impresión. Las ruedas podían cambiarse de lugar por lo que la clave para el uso de esta máquina consistía en el orden en que eran colocadas las ruedas y la posición inicial de cada una. Varios gobiernos adquirieron esta máquina, algunos para estudiarla y otros para usarla en sus comunicaciones, por ejemplo, el gobierno de Alemania mandó modificarla con el fin de hacerla más segura para utilizarla plenamente en sus comunicaciones ya que creían que un criptograma obtenido con Enigma era indescifrable. Pero fue el gobierno Polaco el que hizo estudios intensos para descriptar los criptogramas obtenidos con Enigma, y lo lograron mediante el empleo de máquinas que ellos mismos desarrollaron y que les facilitaron el trabajo, estas máquinas fueron el Ciclómetro y la Bomba. Como la máquina Enigma utilizada por los alemanes era habitualmente modificada así como los protocolos para su manejo, los británicos mejoraron la Bomba y pudieron descriptar mensajes que los Polacos ya no pudieron.

FIGURA 2.9 Máquina ENIGMA

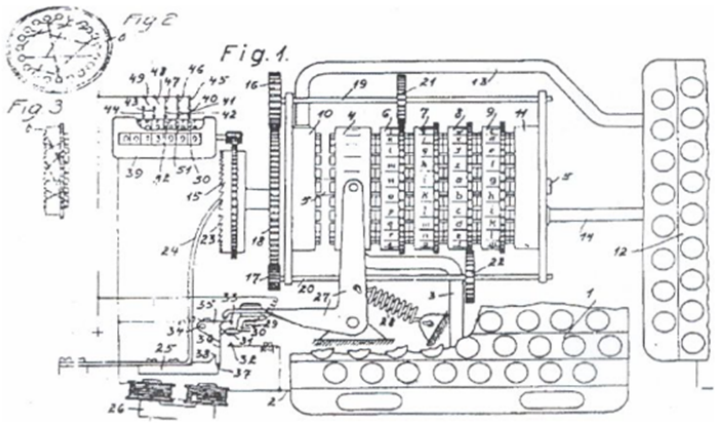


FIGURA 2.10 Vista del interior de una ENIGMA

Al igual que los alemanes utilizaban una máquina para cifrar sus mensajes durante la Segunda Guerra Mundial, Estados Unidos contaba con una máquina llamada Sigaba cuyo principio de funcionamiento era igual al de Enigma, los japoneses utilizaron una máquina llamada Purple criptoanalizada por Estados Unidos.

En 1940 los alemanes comenzaron a utilizar una nueva máquina de cifrado llamada "Máquina de Lorenz" parecida a Enigma pero con un mecanismo mucho más complicado, los británicos también fueron capaces de criptoanalizar la máquina de Lorenz con ayuda de la máquina Colossus.

Conforme pasaba el tiempo, la Criptografía tomaba un carácter matemático y gracias a los estudios realizados por Claude Shannon sobre la Teoría de la Información (1948) y posteriormente con la publicación de su trabajo "La Teoría de las Comunicaciones Secretas" en 1949, en donde sugería utilizar operaciones múltiples que mezclaran transposiciones y sustituciones, la Criptografía dejó de considerarse un arte y fue reconocida como una ciencia.

A partir de entonces, la invención de la computadora y el desarrollo de las matemáticas, permitieron que los nuevos algoritmos de cifrado fueran cada día más complejos.

En 1973, el National Bureau of Standards (NBS) hoy NIST (National Institute of Standards and Technology) realizó una convocatoria pública en el Registro Federal de los Estados Unidos para el desarrollo de sistemas criptográficos, como resultado de dicha convocatoria en 1975 IBM presentó el sistema de cifrado de llave secreta DES (Data Encryption Standard), el cual se explica a detalle en el subtema (Criptografía Simétrica) del presente capítulo. En 1977, el gobierno de E.U. adoptó este método como estándar y también lo hicieron varios gobiernos del mundo aunque algunos sólo aceptaron una parte. En 1981 DES fue estandarizado por la ANSI como ANSI X.3.92. y en 1998 fue descifrado en 56 horas por un ataque de fuerza bruta.

Otro hecho crucial para el desarrollo de la Criptografía ocurrió en 1976 cuando Whitfield Diffie y Martin Hellman ingenieros electrónicos de la Universidad de Stanford propusieron la Criptografía de clave pública.

En el año de 1977 el MIT dio a conocer un poderoso algoritmo criptográfico llamado RSA que debido a su robustez y efectividad es ampliamente utilizado hoy en día.

Desde entonces se han desarrollado varios algoritmos y utilizado diversas herramientas cada vez más sofisticadas.

2.1.3 SERVICIOS Y MECANISMOS DE SEGURIDAD

SERVICIOS DE SEGURIDAD

Los servicios de seguridad, son aquellos que garantizan en un sistema de información la adquisición, almacenamiento, procesamiento y transmisión de la información y para lograrlo se valen de uno o más mecanismos de seguridad.

- **CONFIDENCIALIDAD**

Este servicio asegura que sólo las personas o procesos autorizados tengan acceso a la información. Con ello se busca que un agente no autorizado no pueda leer, copiar o modificar la información.

El servicio de confidencialidad se puede diferenciar en dos tipos:

a) **Servicio de confidencialidad de contenido**: se busca proteger el contenido de un recurso del sistema, para ello se cifra la información para que en caso ser interceptada por alguien no autorizado, no pueda ser descubierta. Este servicio puede proporcionar protección a todos los datos transmitidos por un usuario durante una conexión o puede proteger sólo parte de ellos por ejemplo sólo a los mensajes con información importante o incluso se pueden proteger sólo algunos campos de un determinado mensaje.
b) **Servicio de confidencialidad del mensaje**: busca ocultar el flujo de un mensaje durante una conexión, para ello se cifra y se utiliza una técnica de envoltura con el objetivo de que si un atacante está realizando un análisis de tráfico, no pueda

descubrir por ejemplo quien está enviando la información ni quien la recibe ni la frecuencia con la que se envían los mensajes.

- **AUTENTICACIÓN**

Este servicio verifica la identidad de un agente que pretende acceder a la información. En una conexión entre dos entidades, el servicio verifica que las entidades sean quienes dicen ser, además de asegurar que un tercer individuo no pueda hacerse pasar por alguna de las entidades autorizadas y realizar una transmisión o recepción de datos.

- **INTEGRIDAD**

Este servicio asegura que el contenido de los datos no ha sido modificado y que la secuencia de los mismos se ha mantenido a lo largo de toda la transmisión, con ello se evita una réplica o un reordenamiento del mensaje por parte de un atacante.
Al igual que el servicio de confidencialidad, la integridad puede aplicarse a todos los datos transmitidos por un usuario durante una conexión, sólo a parte de ellos o sólo a algunos campos dentro del mensaje.

Cuando se tiene un ataque a la integridad de los datos, el sistema puede o no reportar dicha violación, por lo que se puede distinguir entre servicio de integridad con recuperación y servicio de integridad sin recuperación.

El servicio de integridad también se puede diferenciar entre servicio de integridad del contenido y servicio de integridad de la secuencia del mensaje:

a) **Servicio de integridad del contenido**: proporciona pruebas de que el contenido no ha sido alterado o modificado.
b) **Servicio de integridad de la secuencia del mensaje**: proporciona pruebas de que el orden de una secuencia de mensajes ha sido mantenida durante la transmisión.

- **NO REPUDIO**

Este servicio evita que las entidades en una conexión nieguen haber transmitido o recibido un mensaje.

Existen varios tipos de este servicio y cada uno de ellos proporciona pruebas de haberse llevado a cabo:

a) **No repudio de origen**: con este servicio, el emisor de un mensaje no puede negar haber sido él quien transmitió dicho mensaje.
b) **No repudio de envío**: comprueba que los datos fueron enviados.
c) **No repudio de presentación**: protege contra cualquier intento falso de negar que los datos fueron presentados para el envío.
d) **No repudio de transporte**: protege contra cualquier intento de negar que los datos fueron transportados.
e) **No repudio de recepción**: con este servicio, el receptor de un mensaje no puede negar haber recibido un mensaje.

- **CONTROL DE ACCESO**

El servicio de control de acceso es utilizado con el fin de restringir el acceso a los medios de almacenamiento de la información. Este servicio está muy relacionado con el de autenticación ya que cualquier agente que quiera tener acceso a algún recurso del sistema primero deberá identificarse para que le sea permitido el acceso a dicha información y de acuerdo a los permisos o privilegios que tenga podrá manipularla.

- **DISPONIBILIDAD**

El servicio de disponibilidad asegura que los agentes autorizados tengan acceso a la información en el momento en que ellos lo requieran y tantas veces como lo soliciten sin importar si ésta es correcta o no.

MECANISMOS DE SEGURIDAD

Un mecanismo de seguridad es un conjunto de elementos o procesos que implementan un servicio de seguridad.

- **CÓDIGO DE DETECCIÓN DE MODIFICACIÓN**

Se trata de una suma que se aplica a los datos a transmitir, el resultado de dicha suma se envía junto con los datos para que el receptor efectúe una prueba de comprobación; se debe obtener el mismo resultado tanto de la parte del emisor como del receptor para estar seguros de que los datos no fueron modificados. Cabe señalar que la suma es generada utilizando un algoritmo criptográfico.

- **CÓDIGO DE AUTENTICACIÓN DEL MENSAJE**

Este caso es muy parecido al anterior sólo que el resultado de la suma está cifrado y cuando el receptor realiza la prueba de comprobación se tendrá la certeza de que los datos están íntegros y que el emisor es quien se supone los envió.

- **FIRMA DIGITAL**

Una firma digital es una pieza de información que consiste en una transformación que por medio de una función relaciona de forma única un documento con la clave privada del firmante, es decir que las firmas digitales dependen del mensaje y de quien la genera, con el fin de que la información no sea modificada y al mismo tiempo sirve para proporcionar servicios de no repudio ya que el destinatario tendrá la certeza de que el mensaje fue enviado por quien esperaba.

- **NÚMERO DE SECUENCIA DEL MENSAJE**

Cuando un mensaje se divide en varios paquetes para ser transmitido; a cada paquete se le agrega un número el cual puede ir cifrado o no, dicho número es en realidad una secuencia de bits que identifica el número de secuencia del paquete; de esta manera el receptor tiene que comprobar que dicha secuencia de bits corresponde con el número de paquete que está recibiendo. Con este procedimiento se verifica si algún paquete fue insertado o sustraído por un tercer agente durante la transmisión.

- **CIFRADO**

Con el fin de que a individuos o procesos no autorizados les resulte inteligible la información, ésta se transforma por medio de los métodos de cifrado a una forma que no pueda entenderse a simple vista. Con la utilización de este mecanismo de seguridad se busca proteger la confidencialidad de los datos, aunque no es de uso exclusivo para este servicio ya que se puede usar conjuntamente con otros mecanismos para dar soporte a otros servicios.

- **CONTROL DE ACCESO**

Se emplean contraseñas para permitir el acceso a la información a todos aquellos agentes autorizados.

- **RELLENO DE TRÁFICO**

Se trata de transmitir unidades de datos falsos del mismo modo que se transmiten las unidades que llevan información correcta, con ello se busca que si un individuo está realizando un análisis de tráfico no conozca si las unidades llevan realmente información útil.

- **CONTROL DE ENCAMINAMIENTO**

Este mecanismo otorga la oportunidad de mandar la información por una ruta diferente cuando la conexión actual está siendo atacada.

- **CERTIFICACIÓN**

Se realiza una certificación por un tercer agente de confianza, el cual da fe de la integridad, secuencia y frecuencia de los datos así como el emisor y receptor de los mismos.

2.1.4 ATAQUES

Un ataque es una violación a la seguridad de la información realizada por intrusos que tienen acceso físico al sistema sin ningún tipo de restricción, su objetivo es robar la información o hacer que ésta pierda valor relativo, o que disminuyan las posibilidades de su supervivencia a largo plazo.

Un intruso puede obtener información como:

- Bloques de direcciones IP
- Localización de sistemas críticos (DNSs, WINS, DHCPs, Servidores de correo, etc.)
- Puntos de acceso para números telefónicos y VPNs
- Información personal de los trabajadores de la organización
- Organizaciones asociadas, subsidiarias, etc.

Existen dos tipos de ataques que amenazan las comunicaciones secretas:

1. **Pasivo**: es aquel en el cual el intruso sólo busca obtener la información y al hacerlo no la modifica, por lo que es difícil percatarse de que se está siendo atacado.
2. **Activo**: el intruso además de obtener la información la modifica de tal modo que sirva a sus intereses y al ser modificada es más fácil percatarse de que se está siendo atacado.

Los ataques activos se dividen en dos tipos:

a) Ataques a los métodos de cifrado
b) Ataques a los protocolos criptográficos

ATAQUES A LOS MÉTODOS DE CIFRADO

Este tipo de ataques se realizan con la intención de obtener la clave secreta para poder descifrar libremente cualquier criptograma, para ello se aprovechan las vulnerabilidades que pudiera tener el método de cifrado.

- **ATAQUE SÓLO CON TEXTO CIFRADO**

Este caso es cuando el criptoanalista sólo conoce el criptograma y el algoritmo con que fue generado; con esta información pretende obtener el texto en claro.

- **ATAQUE CON TEXTO ORIGINAL CONOCIDO**

En esta situación el criptoanalista conoce mensajes en claro seleccionados por él mismo y sus correspondientes criptogramas, así como el algoritmo con que éstos fueron generados; aquí el objetivo es conocer la clave secreta y poder descriptar libremente cualquier texto.

- **ATAQUE CON TEXTO CIFRADO ESCOGIDO**

El criptoanalista conoce el algoritmo de cifrado, así como un criptograma seleccionado por él mismo y su correspondiente texto en claro, su objetivo es obtener el mensaje en claro de todo criptograma que intercepte.

- **ATAQUE CON TEXTO ESCOGIDO**

En este caso el criptoanalista además de conocer el algoritmo de cifrado y el criptograma que quiere descriptar, también conoce el criptograma de un texto en claro que él elija y el mensaje en claro de un criptograma también elegido por él.

ATAQUES A LOS PROTOCOLOS CRIPTOGRÁFICOS

Este tipo de ataques no pretenden encontrar la clave secreta para poder conocer el mensaje en claro, sino que buscan obtener la información vulnerando los protocolos criptográficos, es decir, pretenden burlar la serie de pasos establecidos para alcanzar los objetivos de seguridad y que tienen que ser realizados por las entidades

involucradas en cierta comunicación. Ejemplos de este tipo de ataques son los siguientes:

- **ATAQUE CON CLAVE CONOCIDA**

El atacante conoce claves utilizadas en cifrados anteriores y con base en ellas intenta determinar nuevas claves.

- **SUPLANTACIÓN DE PERSONALIDAD**

El atacante asume la identidad de uno de los agentes autorizados en la red, y de esta manera obtiene libremente y sin tropiezos todos los mensajes en claro.

- **COMPILACIÓN DE UN DICCIONARIO**

Un diccionario es un archivo guardado en la memoria de la computadora que contiene contraseñas cifradas de los usuarios autorizados en el sistema. Si el método de cifrado con que se cifran las claves es público, el atacante puede generar claves aleatorias y después cifrarlas con el objeto de encontrar alguna contenida en el diccionario (previamente obtenido). Cuando una clave generada por el atacante coincide con una contenida en el diccionario, se ha encontrado una clave de acceso al sistema, mediante el usuario correspondiente a la clave encontrada.

- **BÚSQUEDA EXHAUSTIVA**

Este ataque se lleva a cabo generando aleatoriamente todos los valores posibles de las claves de acceso y probándolas hasta que una de ellas sea una clave válida en el sistema.

- **ATAQUE DE HOMBRE EN MEDIO**

El intruso se filtra en la línea de comunicación entre dos agentes autorizados en la red; obtiene la información de uno de ellos y se la envía al otro usuario una vez que la ha utilizado.

2.1.5 LA ARQUITECTURA DE SEGURIDAD DE OSI

En la figura 2.11 se muestran las capas que conforman el modelo OSI, cada una de ellas consiste en un proceso lógico y en conjunto tienen la finalidad de transferir mensajes en un sistema.

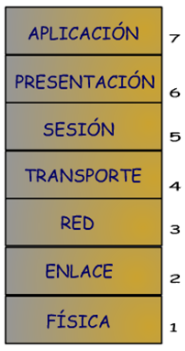


FIGURA 2.11 Niveles del Sistema OSI

En los documentos ISO 7498-2 y ITU-T X.800 se establecen los servicios y mecanismos de seguridad de la arquitectura OSI. Los servicios de seguridad pueden ser llamados por cada una de las capas que conforman el sistema OSI, según el nivel de seguridad que se requiera y según sea apropiado. Esto último porque debe respetarse siempre la funcionalidad de cada capa y respetar la independencia que cada una tiene.

SERVICIOS DE SEGURIDAD DE OSI

La arquitectura de seguridad OSI establece cinco servicios de seguridad (figura 2.12) los cuales se explican brevemente a continuación; si se requiere más información consultar la sección 2.1.3 (Servicios y mecanismos de seguridad) del presente capítulo en donde se explica con más detalle cada uno.

	SERVICIOS DE SEGURIDAD OSI
1	Servicio de autenticación de emisor y receptor
	Servicio de autenticación del origen de los datos
2	Servicio de control de acceso
3	Servicio de confidencialidad orientado a conexión
	Servicio de confidencialidad no orientado a conexión
	Servicio de confidencialidad de campo selectivo
	Servicio de confidencialidad de flujo de tráfico
4	Servicio de integridad orientado a conexión con recuperación
	Servicio de integridad orientado a conexión sin recuperación
	Servicio de integridad de campo seleccionado orientado a conexión
	Servicio de integridad no orientado a conexión
	Servicio de integridad de campo seleccionado no orientado a conexión
5	No repudio con prueba de origen
	No repudio con prueba de destino

FIGURA 2.12 Servicios de seguridad de OSI

- **SERVICIO DE AUTENTICACIÓN**

a) **Servicio de autenticación de emisor y receptor**: proporciona la capacidad de verificar que los comunicantes sean quienes dicen ser.
b) **Servicio de autenticación del origen de los datos**: proporciona la confirmación de que los datos recibidos son de la entidad con la que se está llevando a cabo la comunicación.

- **SERVICIO DE CONTROL DE ACCESO**

Impide el acceso a la información a aquellas personas o procesos no autorizados.

- **SERVICIO DE CONFIDENCIALIDAD**

a) **Servicio de confidencialidad orientado a conexión**: proporciona confidencialidad a todos los datos transmitidos durante toda la conexión.
b) **Servicio de confidencialidad no orientado a conexión**: proporciona confidencialidad de paquetes de datos.

c) **Servicio de confidencialidad de campo selectivo**: proporciona confidencialidad de campos específicos de los datos durante una conexión, o para un paquete de datos.
d) **Servicio de confidencialidad de flujo de tráfico**: este servicio busca ocultar información sobre el flujo de un mensaje durante una conexión.

- **SERVICIO DE INTEGRIDAD**

a) **Servicio de integridad orientado a conexión con recuperación**: proporciona integridad de los datos transmitidos durante toda la conexión y en caso de alguna violación a la integridad de ser posible recupera los problemas que ocasionaron dicha violación.
b) **Servicio de integridad orientado a conexión sin recuperación**: proporciona integridad a los datos durante toda la conexión, pero no se recuperan los problemas que ocasionaron la falla de integridad.
c) **Servicio de integridad de campo seleccionado orientado a conexión**: proporciona integridad de campos específicos de los paquetes transmitidos durante toda la conexión.
d) **Servicio de integridad no orientado a conexión**: proporciona integridad sólo a algunos paquetes de datos.
e) **Servicio de integridad de campo seleccionado no orientado a conexión**: proporciona integridad de campos específicos dentro de algunos paquetes de datos.

- **SERVICIO DE NO REPUDIO**

a) **Servicios de no repudio con prueba de origen**: sirve para proporcionar al destinatario una prueba del origen de los datos.
b) **Servicio de no repudio con prueba de destino**: sirve para proporcionar al emisario una prueba de que los datos se han entregado al destinatario.

En la tabla de la figura 2.13 se muestran los servicios que pueden ser implementados en cada una de las capas del modelo OSI, esta distribución no es definitiva ya que depende de los requerimientos de seguridad en el sistema.

SERVICIO DE SEGURIDAD	CAPA						
	1. Física	2. Enlace	3. Red	4. Transporte	5. Sesión	6. Presentación	7. Aplicación
Autenticación de emisor y receptor			■	■			■
Autenticación del origen de los datos			■	■			■
Control de acceso			■	■			■
Confidencialidad orientado a conexión	■	■	■	■			■
Confidencialidad no orientado a conexión		■	■	■			■
Confidencialidad de campo selectivo							■
Confidencialidad de flujo de tráfico	■		■				■
Integridad orientado a conexión con recuperación				■			■
Integridad orientado a conexión sin recuperación			■	■			■
Integridad de campo seleccionado orientado a conexión							■
Integridad no orientado a conexión			■	■			■
Integridad de campo seleccionado no orientado a conexión							■
No repudio con prueba de origen							■
No repudio con prueba de destino							■

FIGURA 2.13 Servicios de seguridad que se implementan en cada capa del modelo OSI

MECANISMOS DE SEGURIDAD ESPECÍFICOS

Estos mecanismos son usados para proporcionar algunos de los servicios descritos arriba. Dentro de la arquitectura de seguridad OSI se establecen ocho mecanismos de seguridad específicos; en la figura 2.14 se muestran estos mecanismos que por sí solos o en combinación con otros son apropiados para proveer cada servicio de seguridad. Es importante señalar que las relaciones mostradas en la tabla no son definitivas.

SERVICIO	MECANISMO							
	Cifrado	Firma digital	Control de acceso	Integración de datos	Autenticación	Relleno de tráfico	Control de encaminamiento	Certificación
Autenticación de emisor y receptor	■	■			■			
Autenticación del origen de los datos	■	■						
Control de acceso			■					
Confidencialidad orientado a conexión	■						■	
Confidencialidad no orientado a conexión	■						■	
Confidencialidad de campo selectivo	■							
Confidencialidad de flujo de tráfico	■					■	■	
Integridad orientado a conexión con recuperación	■			■				
Integridad orientado a conexión sin recuperación	■			■				
Integridad de campo seleccionado orientado a conexión	■			■				
Integridad no orientado a conexión	■	■		■				
Integridad de campo seleccionado no orientado a conexión	■	■		■				
No repudio con prueba de origen		■		■				■
No repudio con prueba de destino		■		■				■

FIGURA 2.14 Mecanismos de seguridad empleados para implementar cada servicio

Todos estos mecanismos ya fueron descritos con anterioridad en la sección 2.1.3 (Servicios y mecanismos de seguridad) del presente capítulo.

MECANISMOS DE SEGURIDAD GENERALIZADOS

Este tipo de mecanismos son utilizados para proporcionar mayor seguridad a una comunicación entre dos entidades y su uso depende el nivel de seguridad requerido. Dentro de la arquitectura de seguridad OSI se establecen cinco mecanismos de seguridad generalizados:

- **FUNCIONALIDAD DE CONFIANZA**

Se refiere a que todos los procedimientos que se realicen para proporcionar seguridad en la comunicación realmente cumplan con su objetivo tal y como se tiene previsto.

- **ETIQUETAS DE SEGURIDAD**

Los recursos del sistema pueden tener asociadas *etiquetas de seguridad* las cuales sirven para clasificar la información por niveles de seguridad: secreta, confidencial, no clasificada, etc., y con ello identificar la sensibilidad o nivel de protección requerido para cada uno de dichos recursos.

- **DETECCIÓN DE EVENTOS**

Se utiliza para detectar violaciones aparentes de la seguridad. Cabe señalar que se detectan todo tipo de eventos ocurridos es el sistema, es decir violaciones o accesos satisfactorios.

- **RASTREO DE AUDITORÍA DE SEGURIDAD**

Se trata de una revisión de los registros y actividades del sistema para verificar el cumplimiento de las políticas de seguridad establecidas con el fin de proporcionar eficientemente seguridad al sistema; en caso de encontrar alguna falla, se recomiendan los cambios para eliminar los problemas.

- **RECUPERACIÓN DE SEGURIDAD**

Después de alguna falla en la seguridad de un sistema se deben llevar a cabo acciones de recuperación mediante la aplicación de varias medidas establecidas previamente como resultado de ciertas pruebas de verificación de criterios y normas.

2.2 TÉCNICAS CLÁSICAS DE CIFRADO

2.2.1 INTRODUCCIÓN Y CLASIFICACIÓN DE LOS SISTEMAS DE CIFRADO

Un sistema de cifrado es aquel que permite que tanto emisor y receptor cuenten con determinada información confidencial. En la figura 2.15 se puede ver el esquema básico de un sistema de cifrado.

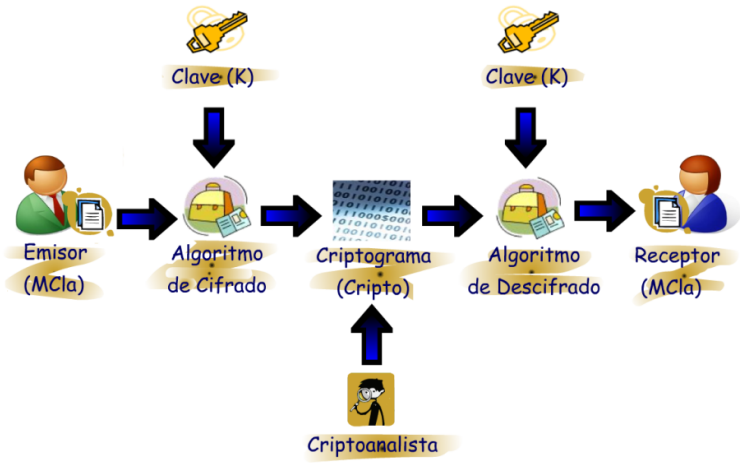


FIGURA 2.15 Sistema criptográfico

El emisor proporciona el mensaje original también denominado Mcla (mensaje en claro) o texto plano, para que el algoritmo de cifrado mediante un determinado procedimiento auxiliado por una clave cifre o transforme dicho mensaje en un mensaje cifrado (criptograma o comúnmente llamado Cripto) que se envía por un canal público. El receptor que conoce la clave, transforma ese criptograma en el texto original con ayuda de un algoritmo de descifrado.

El mensaje puede ser interceptado por un criptoanalista quien buscará descriptar el mensaje cifrado y encontrar el mensaje original.

Todo sistema criptográfico debe cumplir con los siguientes principios, los cuales fueron recomendados por Auguste Kerckhoffs en su trabajo "La Criptografía militar" en 1883:

1. El criptograma debe ser indescriptable.
2. El sistema debe ser compuesto por información pública (conocida por todos) como el algoritmo que se emplea para cifrar y por información privada (sólo emisor y receptor la conocen) como son las claves usadas para el cifrado y descifrado.
3. La clave debe ser fácil de memorizar y cambiar.
4. El criptograma debe ser enviado por los medios de transmisión habituales.
5. El sistema debe ser portátil y empleado por una sola persona.
6. El proceso de descifrado debe ser fácil de usar, su complejidad debe ser la suficiente para mantener la seguridad del sistema.

Los sistemas de cifrado se clasifican de acuerdo con:

a) El tipo de operación utilizada para obtener el criptograma a partir del texto en claro.
b) El número de claves utilizadas durante el proceso de cifrado/descifrado.
c) La manera en que el texto es procesado.

En las secciones 2.2.2 (Operaciones utilizadas), 2.2.3 (Número de claves) y 2.2.4 (Formas de procesamiento de datos) del presente capítulo se explica a detalle cada una de estas clasificaciones.

2.2.2 OPERACIONES UTILIZADAS

Existen dos operaciones que pueden ser utilizadas para obtener el criptograma a partir del texto en claro:

- **Sustitución**: Los caracteres que conforman el mensaje original se intercambian por otros que pueden ser del mismo alfabeto o de uno diferente. La sustitución causa confusión (término introducido por Shannon), es decir oculta la relación que existe entre el texto claro y el texto cifrado.
- **Transposición**: Se intercambian de lugar los caracteres que conforman un mensaje, por lo que el criptograma contiene los mismos caracteres que el mensaje en claro pero resultan incomprensibles a simple vista ya que están desordenados. La transposición causa difusión (término introducido por Shannon), es decir elimina la redundancia (demasiada abundancia de las mismas palabras) del texto en claro, esparciéndola a lo largo de todo el texto cifrado.

ALGORITMOS DE SUSTITUCIÓN

Los algoritmos de sustitución se dividen de la siguiente manera (figura 2.16):

FIGURA 2.16 Clasificación de los algoritmos de sustitución

- **SUSTITUCIÓN MONOALFABÉTICA**

En este tipo de sustitución sólo existe un alfabeto y siempre al texto en claro le va a corresponder el mismo criptograma.

Los siguientes tres métodos que se explicarán (Cifrado del César, Cifrado Atbash y Cifrado de Polybios) son métodos de cifrado de este tipo los cuales además consisten en sustituciones monográmicas debido a que se cifra letra por letra.

- **CIFRADO DEL CÉSAR**

Este método consiste en sustituir cada letra del texto original por otra situada tres posiciones delante de ella en el alfabeto que se esté utilizando.

Por ejemplo, si consideramos el alfabeto castellano de 27 letras, el cifrado de cada una de las letras es el siguiente (figura 2.17):

Letra en el MCla	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Letra cifrada	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

FIGURA 2.17 Cifrado del César para el alfabeto castellano

Ejemplo:
MCla: MUCHOS AÑOS DESPUES
Cripto: OXFKRV DQRV GHVSXHV

- **CIFRADO ATBASH**

Este método consiste en sustituir la primera letra por la última del alfabeto que se esté utilizando, la segunda por la penúltima, la tercera por la antepenúltima y así sucesivamente con todo el alfabeto.

Considerando el alfabeto castellano, el cifrado de cada letra es el que se muestra en la figura 2.18:

Letra en el MCla	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Letra cifrada	Z	Y	X	W	V	U	T	S	R	Q	P	O	Ñ	N	M	L	K	J	I	H	G	F	E	D	C	B	A

FIGURA 2.18 Cifrado Atbash para el alfabeto castellano

A este método también suele llamársele código espejo ya que el cifrado del criptograma es nuevamente el texto en claro.

Ejemplo:
MCla: FRENTE AL PELOTON DE FUSILAMIENTO
Cripto: UIVNGV ZO KVOLGLN WV UFHROZÑRVNGL

- **CIFRADO DE POLYBIOS**

La figura 2.19 muestra las matrices ideadas para llevar a cabo este cifrado, dichas matrices contienen el alfabeto a utilizar, en este caso el castellano, nótese que se dejaron fuera las letras ñ y w por ser poco comunes en la escritura, pero si fueran utilizadas en el mensaje en claro se tendrían que incluir y dejar fuera otras letras que no son utilizadas en el mensaje o bien compartiendo una casilla para dos letras.

	1	2	3	4	5
1	A	F	K	P	U
2	B	G	L	Q	V
3	C	H	M	R	X
4	D	I	N	S	Y
5	E	J	O	T	Z

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I	J
3	K	L	M	N	O
4	P	Q	R	S	T
5	U	V	X	Y	Z

FIGURA 2.19 Cifradores de Polybios

El cifrado consiste en sustituir cada letra por la pareja de valores correspondientes al renglón y la columna que definen su posición en la matriz.

Por ejemplo la letra E (en la matriz de la izquierda de la figura 2.19) está ubicada en el quinto renglón y la primera columna por lo que le corresponde el valor: 51

Ejemplo:
MCla: EL CORONEL HABIA DE RECORDAR

Utilizando la matriz de la figura 2.20:

	1	2	3	4	5
1	A	F	K	P	U
2	B	G	L	Q	V
3	C	H	M	R	X
4	D	I	N	S	Y
5	E	J	O	T	Z

FIGURA 2.20 Funcionamiento del cifrador de Polybios

Cripto: 5123 31533453435123 3211214211 4151 3451315334411134

A continuación se explica el método de cifrado de Playfair el cual consiste en sustitución monoalfabética poligrámica debido a que se cifra sobre grupos de caracteres.

- **CIFRADO DE PLAYFAIR**

El cifrado de Playfair requiere que se construya una matriz de 5x5 en donde se coloca el alfabeto después de haber colocado una clave.

Por ejemplo utilizando la clave AQUELLA TARDE REMOTA, se deben tener las siguientes consideraciones para la construcción de la matriz:

- La V y W comparten la misma casilla al igual que la Ñ y N, (esta disposición puede variar, de hecho es recomendable que la ñ y la n estén en diferente casilla con el objetivo de que no causen confusión en el mensaje).
- Se comienza poniendo la clave en la matriz (una letra por casilla de izquierda a derecha), si alguna letra se repite en la clave sólo se pone una vez y las demás veces se omite, (figura 2.21).

A	Q	U	E	L
T	R	D	M	O

FIGURA 2.21 Clave "AQUELLA TARDE REMOTA" en la matriz de 5x5

- Se verifica cada letra del alfabeto para ver si la tenemos que colocar en la matriz. Por ejemplo verificamos si la A ya está en la matriz, en este caso sí lo está, entonces seguimos con la B, la cual aún no está por lo que la colocamos en la siguiente casilla vacía, se sigue el mismo procedimiento hasta acabar con todas las letras del alfabeto, (figura 2.22).

A	Q	U	E	L
T	R	D	M	O
B	C	F	G	H
I	J	K	N/Ñ	P
S	V/W	X	Y	Z

FIGURA 2.22 Cifrador de Playfair utilizando la clave "AQUELLA TARDE REMOTA"

El método de cifrado trabaja con dos caracteres (bigrama) a la vez, por lo que el texto en claro se debe descomponer en parejas de dos caractetes. Cada una de las parejas de caracteres obtenidas después de la descomposición se sustituye por otra conforme a las siguientes reglas:

- Si las dos letras se encuentran en el mismo renglón de la matriz antes construida, cada una de ellas se sustituye con la letra que esté a su derecha.

Suponiendo que la pareja del texto en claro es: DO, la nueva pareja es: MT (figura 2.23).

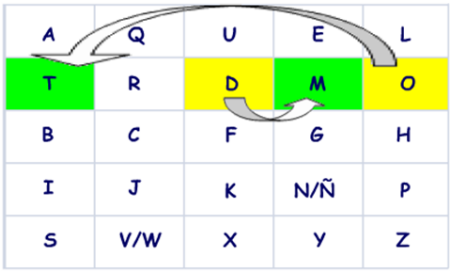


FIGURA 2.23 Cifrado Playfair de dos letras que se encuentran en el mismo renglón

- Si las dos letras se encuentran en la misma columna, cada una de las letras se sustituye por la letra que este debajo de ella.

Suponiendo que la pareja del mensaje en claro es: MY, la nueva pareja la cual es parte del criptograma es: GE (figura 2.24).

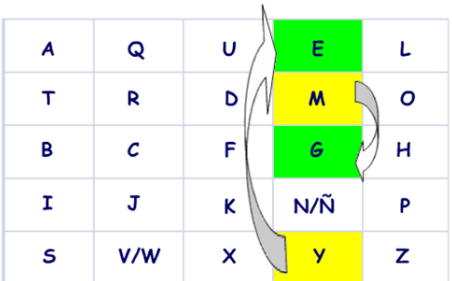


FIGURA 2.24 Cifrado Playfair de dos letras que se encuentran en la misma columna

- En otro caso, la primera letra de la pareja se sustituye por la que este en la intersección de su misma fila y la columna de la segunda letra, la segunda letra

se sustituye por la que este en la intersección de su misma fila y la columna de la primera letra.

Suponiendo que la pareja del mensaje en claro es: AH, la nueva pareja es: LB (figura 2.25).

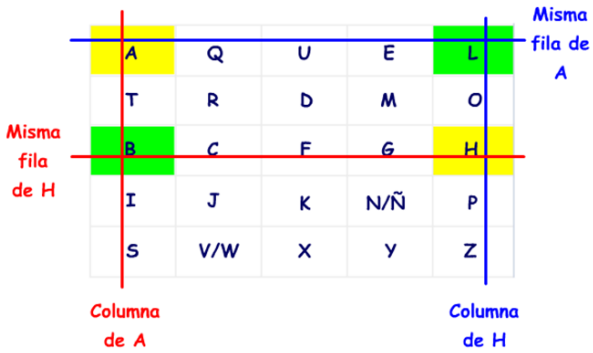


FIGURA 2.25 Cifrado Playfair de dos letras que se encuentran en diferente renglón y diferente columna

- Si la pareja está conformada por la misma letra, entonces se debe descomponer dicha pareja en dos nuevas parejas de la siguiente manera: suponiendo que la pareja es AA las nuevas dos parejas son AX y AX.

- Si el número de caracteres del mensaje en claro es impar, se debe agregar una 'X' para poder formar todas las parejas.

Ejemplo:
MCla: EN QUE SU PADRE

Descomposición en bigramas: EN QU ES UP AD RE

Utilizando el cifrador obtenido anteriormente con la clave "aquella tarde remota" (figura 2.26):

A	Q	U	E	L
T	R	D	M	O
B	C	F	G	H
I	J	K	N/Ñ	P
S	V/W	X	Y	Z

FIGURA 2.26 Ejemplo del cifrado Playfair

- Dado que E y N están en la misma columna: la E se sustituye por la M que es la letra debajo de ella y la N por la Y.
- Q y U están en el mismo renglón: la Q se sustituye por la U que es la letra que está a su derecha y la U por la E.

- E y S están en diferente renglón y diferente columna: la E se sustituye por la A (intersección del mismo renglón de E y columna de S) y la S por la Y (intersección del mismo renglón de S y columna de E).

Cripto: MY UE AY LK UT MQ

- **SUSTITUCIÓN POLIALFABÉTICA**

A diferencia de la sustitución monoalfábetica en donde al texto en claro siempre le corresponde el mismo criptograma, en la sustitución polialfabética el criptograma del texto en claro puede ser diferente dependiendo de la clave que se utilice para cifrar, por lo que se dice que existen múltiples alfabetos de cifrado, de ahí el nombre de sustitución polialfabética.

- **CIFRADO DE ALBERTI**

Este cifrado se lleva a cabo con la ayuda del cifrador que se muestra en la figura 2.27 ideado por Leon Battista Alberti.

FIGURA 2.27 Cifrador de Leon Battista Alberti

El cifrado se realiza haciendo coincidir una letra del disco exterior con la letra que se desee del disco interior. Esta pareja de caracteres son la clave que tanto emisor y receptor deberán conocer para llevar a cabo el proceso de cifrado/descifrado. El texto en claro se cifra letra por letra haciendo coincidir las letras del mensaje en claro con el disco exterior y sustituyéndolas por las letras correspondientes en el disco interior.

Ejemplo:

Supongamos que la clave para el cifrado es Ti.

Clave: Ti (Se debe entender que la letra i del disco interior se debe hacer coincidir con la T del disco exterior, tal y como se muestra en la figura 2.27).

MCla: LO LLEVO A CONOCER
Cripto: oc oozxc d gcfcgzy

➢ CIFRADO POR DESPLAZAMIENTO

Este método consiste en sustituir cada letra del texto original por otra situada k posiciones delante de ella en el alfabeto que se esté utilizando. Este método es la generalización del cifrado del César ya que ahora el desplazamiento (k) en vez de ser fijo (3 posiciones) puede variar entre el rango: $0 \leq k < n$, en donde n es el número de caracteres del alfabeto. El desplazamiento k es la clave del sistema.

Ejemplo:

Tomando en cuenta la tabla de la figura 2.28; para cifrar una letra del texto en claro sólo basta con sumar (módulo el número de caracteres en el alfabeto) el desplazamiento y la posición de la letra del mensaje en claro en el alfabeto y sustituirla por la nueva letra que indique el resultado de la suma (figura 2.29).

Alfabeto castellano	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Posición	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

FIGURA 2.28 Tabla de posiciones del alfabeto

Clave: Desplazamiento k= 9

MCla: EL HIELO

E	L	H	I	E	L	O	← MCla
4	11	7	8	4	11	15	← Posición de cada letra en el alfabeto
13	20	16	17	13	20	24	← MCla + Desplazamiento
N	T	P	Q	N	T	X	← Criptograma

FIGURA 2.29 Ejemplo del cifrado por desplazamiento

Cripto: NT PQNTX

➢ CIFRADO DE VIGENÈRE

Este cifrado consiste en realizar la suma (módulo el número de caracteres en el alfabeto) de la clave y el texto en claro una vez que se ha asignado un valor entero a cada carácter del alfabeto.

Ejemplo:

Considerando los valores numéricos asignados a cada carácter del alfabeto castellano de la tabla mostrada en la figura 2.30, el cifrado de Vigenère se realiza tal y como se muestra en la figura 2.31:

Alfabeto castellano	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Valor Numérico	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

FIGURA 2.30 Asignación de un valor numérico a cada carácter del alfabeto

Clave: ERA

MCla: ENTONCES UNA ALDEA

E	N	T	O	N	C	E	S	U	N	A	A	L	D	E	A
4	13	20	15	13	2	4	19	21	13	0	0	11	3	4	0
E	R	A	E	R	A	E	R	A	E	R	A	E	R	A	E
4	18	0	4	18	0	4	18	0	4	18	0	4	18	0	4
8	4	20	19	4	2	8	10	21	17	18	0	15	21	4	4
I	E	T	S	E	C	I	K	U	Q	R	A	O	U	E	E

MCla
Clave (se repite las veces que sea necesario)
MCla + Clave
Criptograma

FIGURA 2.31 Ejemplo del cifrado de Vigenère

Cripto: IETSECIK UQR AOUEE

El MCla se recupera realizando la resta (módulo el número de caracteres en el alfabeto) de la clave y el criptograma.

Otra manera de realizar el cifrado de un mensaje con el método de Vigenère es la siguiente:

Se utiliza una clave y una matriz cuadrada que contiene 26 alfabetos distribuidos tal y como muestra la figura 2.32.

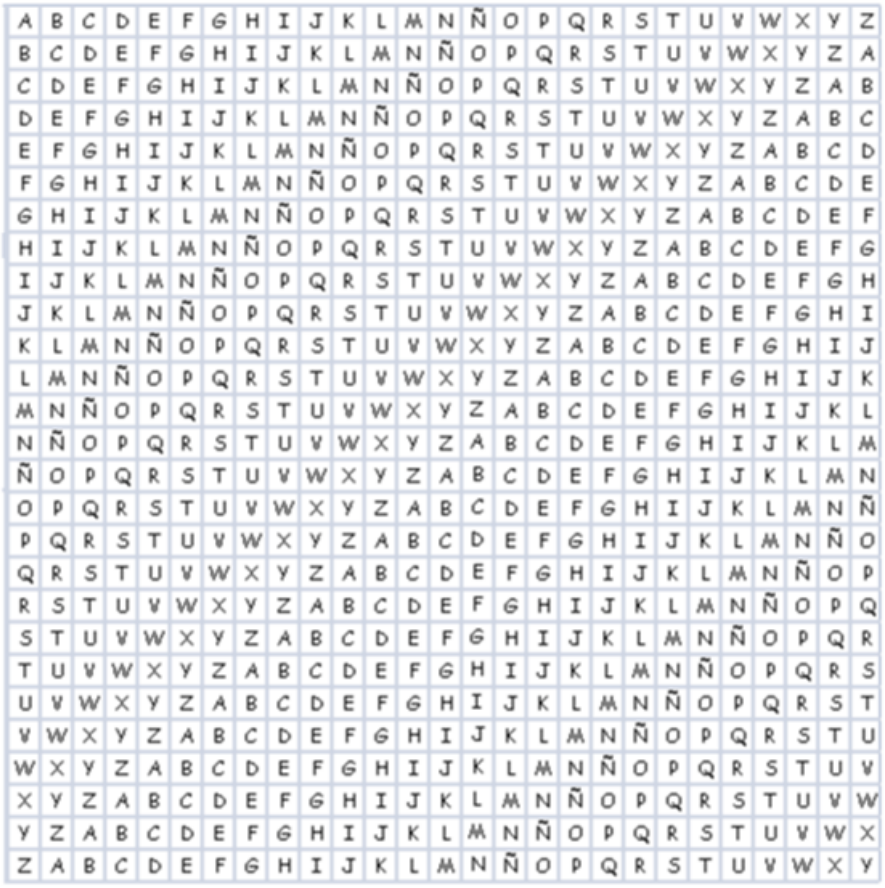


FIGURA 2.32 Cuadrado de Tritemio

Ejemplo:

Clave: ERA

MCla: ENTONCES UNA ALDEA

- El cifrado se realiza carácter por carácter, para ello a cada carácter del mensaje en claro se le hace coincidir con un carácter de la clave, si ésta es más corta que el mensaje en claro se repite las veces que sea necesario (figura 2.33 a).

- El primer renglón de la matriz corresponde a los caracteres de la clave y la primera columna a los caracteres del mensaje en claro (figura 2.33 b).
- El criptograma es aquel carácter que resulte de la intersección del renglón y la columna de donde se encuentren los caracteres de la clave y el mensaje en claro respectivamente (figura 2.33 b).

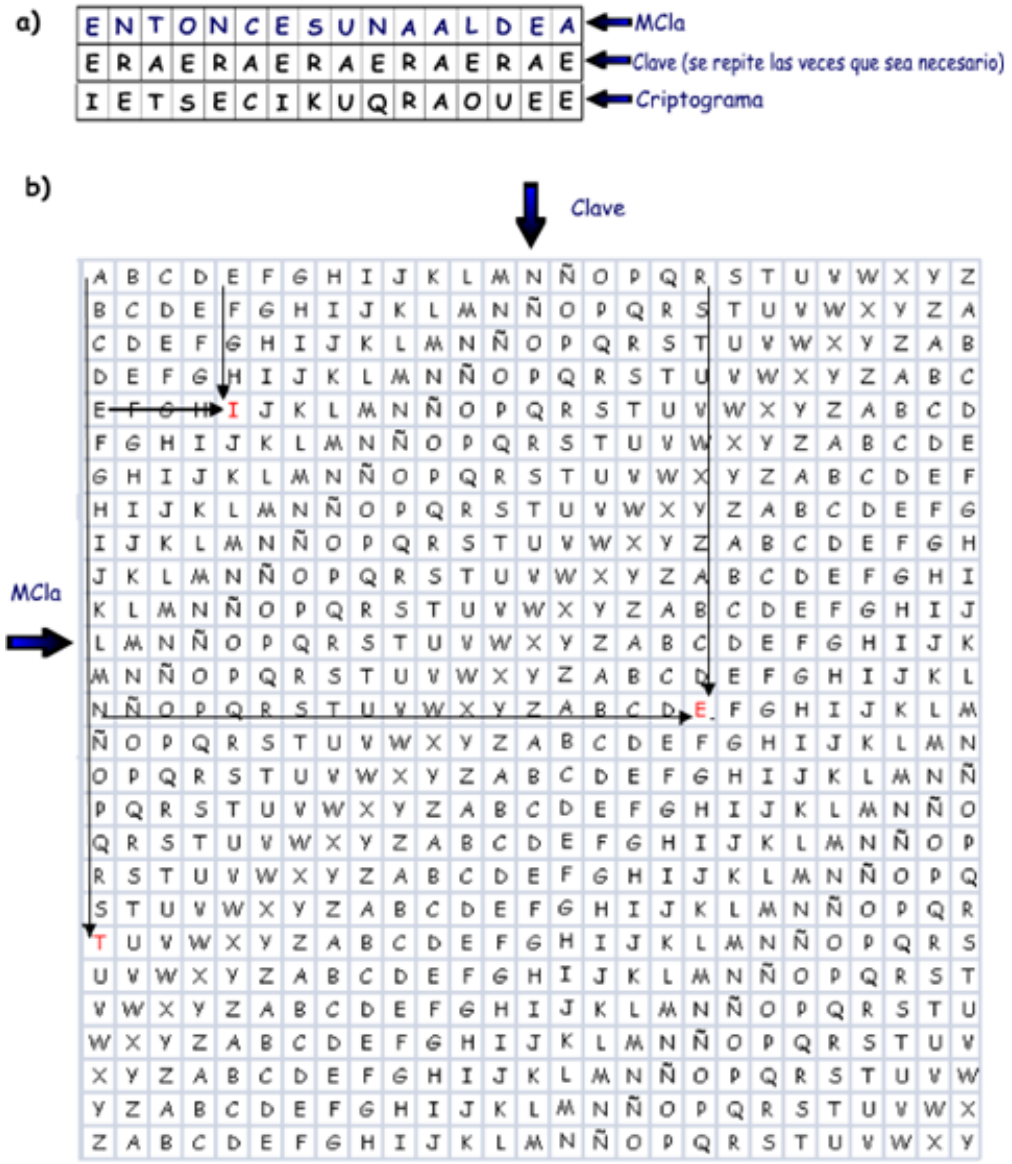


FIGURA 2.33 Cifrado de Vigenère utilizando el cuadrado de Tritemio

El mensaje en claro se recupera haciendo coincidir cada carácter del criptograma con uno de la clave (figura 2.34 a) y buscando en la columna de la letra de la clave el carácter del criptograma, la primera letra que esté en el renglón de dicho carácter es la letra del mensaje en claro, véase la figura 2.34 b:

Por ejemplo para buscar el MCla que corresponde al criptograma obtenido en el ejercicio anterior:

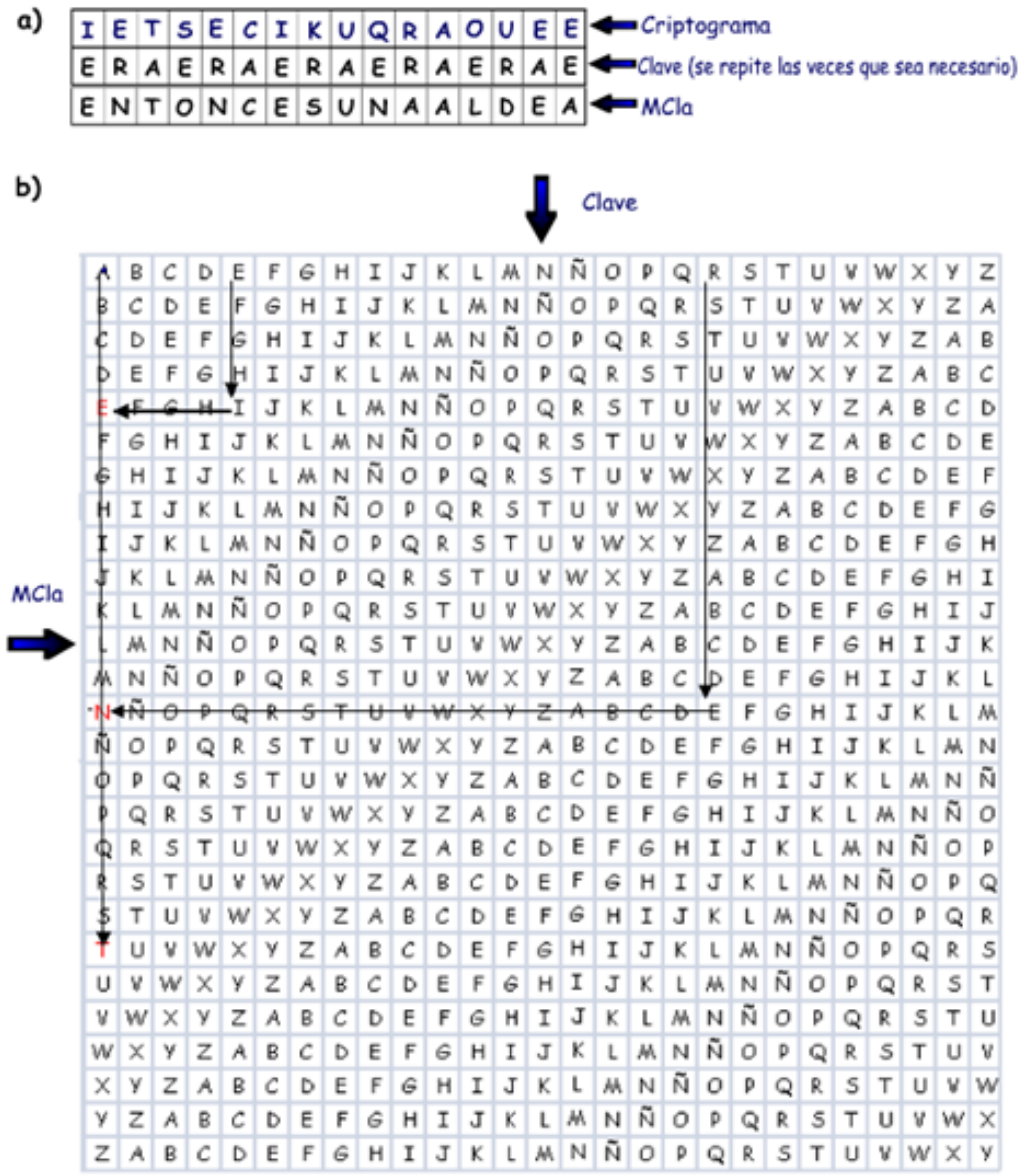


FIGURA 2.34 Recuperación del mensaje en claro del cifrado de Vigenère

- **CIFRADO DE VERNAM**

El cifrado de Vernam también llamado máscara desechable es parecido al cifrado de Vigenère solo que aquí la clave es aleatoria y tan larga como el mensaje, además se debe utilizar una sola vez. Claude Shannon en su trabajo "Teoría de las comunicaciones secretas" demostró que estas características hacen que este cifrado sea perfectamente seguro ya que no hay manera de criptoanalizarlo (es matemáticamente complicado).

Ejemplo:

Considerando los valores numéricos asignados a cada carácter del alfabeto castellano de la tabla de la figura 2.35, el cifrado de Vernam se realiza de la siguiente manera (figura 2.36):

Alfabeto castellano	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z
Valor Numérico	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

FIGURA 2.35 Asignación de un valor numérico a cada carácter del alfabeto

Clave: EDSAS A CETNIEVED
MCla: BARRO Y CAÑABRAVA

B	A	R	R	O	Y	C	A	Ñ	A	B	R	A	V	A
1	0	18	18	15	25	2	0	14	0	1	18	0	22	0
E	D	S	A	S	A	C	E	T	N	I	E	V	E	D
4	3	19	0	19	0	2	4	20	13	8	4	22	4	3
5	3	10	18	7	25	4	4	7	13	9	22	22	26	3
F	D	K	R	H	Y	E	E	H	N	J	V	V	Z	D

MCla
Clave (tan larga como el mensaje)
MCla + Clave
Criptograma

FIGURA 2.36 Ejemplo del cifrado de Vernam (suma módulo 27)

Cripto: FDKRH Y EEHNJVVZD

Otra forma de realizar el cifrado de Vernam es realizando la suma módulo 2; para el ejemplo siguiente emplearemos el código ASCII (figura 2.37) para obtener el equivalente binario de cada carácter (figura 2.38).

Caracteres de control ASCII

DEC	HEX	Símbolo ASCII	
00	00h	NULL	(carácter nulo)
01	01h	SOH	(inicio encabezado)
02	02h	STX	(inicio texto)
03	03h	ETX	(fin de texto)
04	04h	EOT	(fin transmisión)
05	05h	ENQ	(enquiry)
06	06h	ACK	(acknowledgement)
07	07h	BEL	(timbre)
08	08h	BS	(retroceso)
09	09h	HT	(tab horizontal)
10	0Ah	LF	(salto de linea)
11	0Bh	VT	(tab vertical)
12	0Ch	FF	(form feed)
13	0Dh	CR	(retorno de carro)
14	0Eh	SO	(shift Out)
15	0Fh	SI	(shift In)
16	10h	DLE	(data link escape)
17	11h	DC1	(device control 1)
18	12h	DC2	(device control 2)
19	13h	DC3	(device control 3)
20	14h	DC4	(device control 4)
21	15h	NAK	(negative acknowle.)
22	16h	SYN	(synchronous idle)
23	17h	ETB	(end of trans. block)
24	18h	CAN	(cancel)
25	19h	EM	(end of medium)
26	1Ah	SUB	(substitute)
27	1Bh	ESC	(escape)
28	1Ch	FS	(file separator)
29	1Dh	GS	(group separator)
30	1Eh	RS	(record separator)
31	1Fh	US	(unit separator)
127	20h	DEL	(delete)

Caracteres ASCII imprimibles

DEC	HEX	Símbolo	DEC	HEX	Símbolo	DEC	HEX	Símbolo
32	20h	espacio	64	40h	@	96	60h	`
33	21h	!	65	41h	A	97	61h	a
34	22h	"	66	42h	B	98	62h	b
35	23h	#	67	43h	C	99	63h	c
36	24h	$	68	44h	D	100	64h	d
37	25h	%	69	45h	E	101	65h	e
38	26h	&	70	46h	F	102	66h	f
39	27h	'	71	47h	G	103	67h	g
40	28h	(	72	48h	H	104	68h	h
41	29h	)	73	49h	I	105	69h	i
42	2Ah	*	74	4Ah	J	106	6Ah	j
43	2Bh	+	75	4Bh	K	107	6Bh	k
44	2Ch	,	76	4Ch	L	108	6Ch	l
45	2Dh	-	77	4Dh	M	109	6Dh	m
46	2Eh	.	78	4Eh	N	110	6Eh	n
47	2Fh	/	79	4Fh	O	111	6Fh	o
48	30h	0	80	50h	P	112	70h	p
49	31h	1	81	51h	Q	113	71h	q
50	32h	2	82	52h	R	114	72h	r
51	33h	3	83	53h	S	115	73h	s
52	34h	4	84	54h	T	116	74h	t
53	35h	5	85	55h	U	117	75h	u
54	36h	6	86	56h	V	118	76h	v
55	37h	7	87	57h	W	119	77h	w
56	38h	8	88	58h	X	120	78h	x
57	39h	9	89	59h	Y	121	79h	y
58	3Ah	:	90	5Ah	Z	122	7Ah	z
59	3Bh	;	91	5Bh	[	123	7Bh	{
60	3Ch	<	92	5Ch	\	124	7Ch	|
61	3Dh	=	93	5Dh	]	125	7Dh	}
62	3Eh	>	94	5Eh	^	126	7Eh	~
63	3Fh	?	95	5Fh	_			

FIGURA 2.37 Código ASCII

Ejemplo:
MCla: A LA ORILLA DE
Clave: `$/ 2")*6tn|

A	L	A	O	R	I	L	L	A	D	E	
01000001	01001100	01000001	01001111	01010010	01001001	01001100	01001100	01000001	01000100	01000101	← MCla
`	$	/	2	"	)	*	6	t	n	|	← Clave (tan larga como el mensaje)
01100000	00100100	00101111	00110010	00100010	00101001	00101010	00110110	01110100	01101111	01111100	
00100001	01101000	01101110	01111101	01110000	01100000	01100110	01111010	00110101	00101011	00111001	← MCla + Clave
!	h	n	}	p	`	f	z	5	t	9	← Criptograma

FIGURA 2.38 Ejemplo del cifrado de Vernam (suma módulo 2)

Cripto: ! hn }p`fz5 t9

ALGORITMOS DE TRANSPOSICIÓN

A diferencia de los algoritmos de sustitución en donde los caracteres que conforman el mensaje en claro son sustituidos por otros, los algoritmos de transposición los cambian

de posición dentro del mismo mensaje dando lugar al criptograma el cual no puede ser comprendido a simple vista.

- **TRANSPOSICIÓN INVERSA**

Para obtener el criptograma se debe reescribir el mensaje en claro al revés, es decir, la última letra del mensaje en claro será la primera del criptograma, la penúltima la segunda y así sucesivamente con todo el mensaje.

La figura 2.39 muestra un ejemplo de transposición inversa:

FIGURA 2.39 Transposición inversa

- **TRANSPOSICIÓN SIMPLE**

El procedimiento para realizar la transposición simple es el siguiente:

1. El mensaje en claro se reescribe en dos renglones: la primera letra en el primer renglón, la segunda en el segundo renglón, la tercera en el primer renglón, la cuarta en el segundo renglón y así uno y uno hasta acabar con todos los caracteres del mensaje en claro.
2. Se reescribe el mensaje por renglones.

La figura 2.40 muestra un ejemplo de transposición simple:

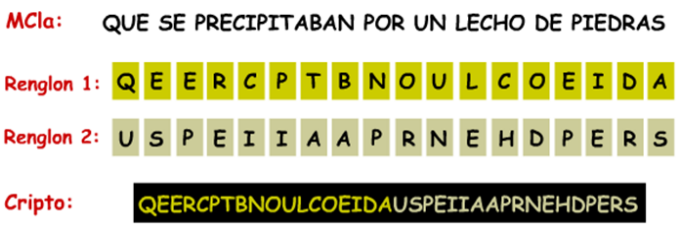


FIGURA 2.40 Ejemplo de transposición simple

Para recuperar el criptograma se realiza lo siguiente:

1. Se obtiene un número ‘n’ de la siguiente manera: el número de caracteres que conforman el criptograma se divide entre dos, al resultado se le suma el residuo de la división.
2. Se toman los primeros n caracteres del criptograma y se colocan en el primer renglón, los restantes se colocan en un segundo renglón.
3. El mensaje en claro se conforma tomando uno y uno de los caracteres de cada renglón.

En la figura 2.41 se muestra la recuperación del MCla a partir del criptograma obtenido en el ejemplo de la figura 2.40.

FIGURA 2.41 Obtención del MCla de una transposición simple

- **TRANSPOSICIÓN POR COLUMNAS**

Para obtener el criptograma se realiza lo siguiente:

1. Los caracteres que conforman la clave se enumeran por orden alfabético.
2. El mensaje en claro se reescribe debajo de la clave enumerada formando varios renglones, si alguno quedara incompleto se rellena con los caracteres que se deseen.
3. Se reescriben las columnas por orden numérico.
4. Se escribe el criptograma por columnas (no se toma en cuenta la clave).

Para dejar más claros los puntos anteriores se presenta el ejemplo de la figura 2.42:

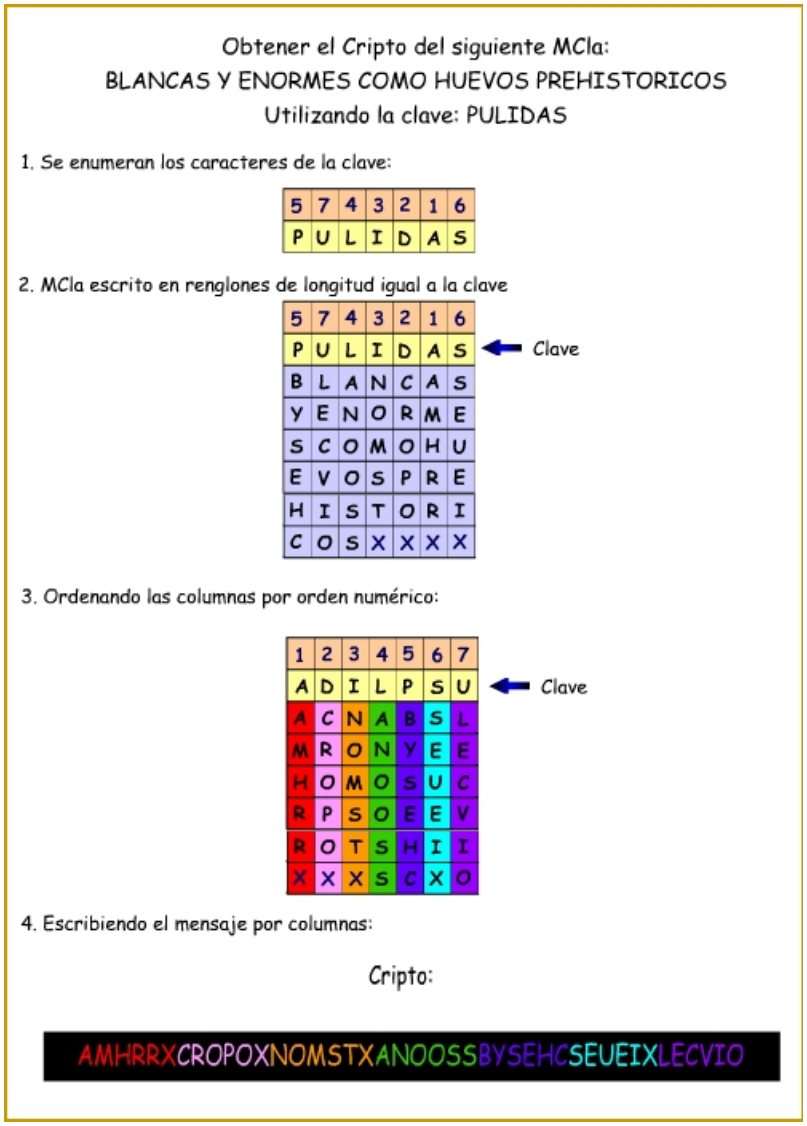


FIGURA 2.42 Transposición por columnas

Para recuperar el criptograma se realiza lo siguiente:

1. Se obtiene un número 'n' de la siguiente manera: el número de caracteres que conforman el criptograma se divide entre el número de caracteres en la clave.
2. Los caracteres que conforman la clave se enumeran por orden alfabético y se reacomoda por orden numérico.
3. Debajo de la clave se escribe el criptograma por columnas, el número de caracteres escritos debajo de cada carácter de la clave será 'n'.
4. Se reescriben las columnas en el orden original de la clave.
5. El MCla se obtiene leyendo por filas.

En la figura 2.43 se muestra la recuperación del MCla a partir del criptograma obtenido en el ejemplo de la figura 2.42.

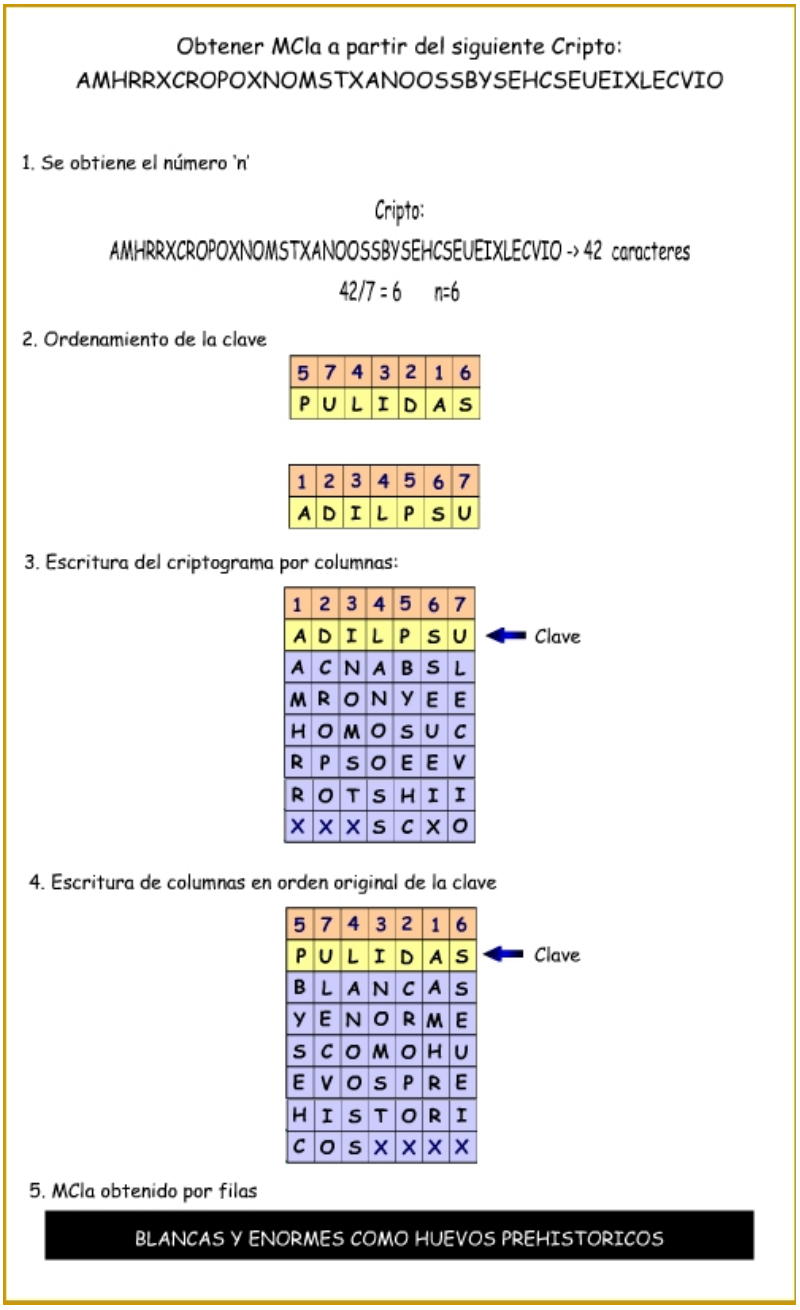


FIGURA 2.43 Obtención del MCla de una transposición por columnas

- **MÁSCARA ROTATIVA**

Este cifrado consiste en construir dos matrices de n x n, en una de ellas se escribe el mensaje en claro además de otros caracteres, en la otra se realizan varias perforaciones de tal modo que al colocarla sobre la primera matriz e irla girando se visualicen los caracteres que conforman el mensaje en claro, a esta matriz perforada se le llama máscara. Para descubrir por completo el mensaje en claro, la máscara se debe ir girando en sentido horario sobre la matriz que contiene los caracteres la cual siempre permanece en la misma posición.

Se deben tomar en cuenta las siguientes consideraciones para construir tanto la matriz que contiene el mensaje como la máscara.

- Las matrices deben de ser de n x n y se pueden hacer tan grandes hasta que se pueda escribir el mensaje en claro por completo, otra opción es utilizar varias matrices utilizando siempre la misma máscara para descubrir el mensaje.
- Los orificios de la máscara se deben de hacer de tal modo que no se sobrepongan caracteres, es decir que los orificios en una posición de la máscara no coincidan con otros orificios cuando se rota.
- Una vez escritos todos los caracteres del mensaje en claro en la matriz, las casillas restantes se deben rellenar con los caracteres que se deseen.

Para leer el mensaje se deben tomar las siguientes consideraciones:

- El mensaje se lee renglón por renglón.
- La máscara se rota en sentido horario.

Ejemplo:
MCla: EL MUNDO ERA TAN RECIENTE QUE

La máscara que se construyó para este ejemplo fue de 6 x 6. En la figura 2.44 se muestran las cuatro posiciones que puede tener la máscara y como el mensaje se va descubriendo:

FIGURA 2.44 Máscara rotativa

2.2.3 NÚMERO DE CLAVES

Durante el proceso de cifrado/descifrado se puede utilizar una sola clave o dos claves:

- **Sistemas de una clave**: Tanto emisor y receptor utilizan la misma clave para el proceso de cifrado y descifrado.

- **Sistemas de dos claves**: Se emplean dos claves, una pública, conocida por cualquiera y utilizada para cifrar y una clave privada que sólo es conocida por el receptor que es quien va a descifrar el mensaje.

SISTEMAS DE UNA CLAVE (CIFRADORES SIMÉTRICOS)

Este tipo de cifrado también es llamado de clave sencilla, de clave secreta o de secreto compartido ya que tanto emisor y receptor comparten la misma clave para cifrar y descifrar. Debido a ello la clave debe ser secreta y sólo el emisor y receptor deben conocerla para proporcionar seguridad al sistema. De hecho el principal riesgo de recibir un ataque con este tipo de cifrado es cuando se proporciona la clave al receptor, por lo que se debe hacer de una manera segura.

En la figura 2.45 se muestra el esquema de un sistema de cifrado simétrico.

FIGURA 2.45 Sistema criptográfico simétrico

SISTEMAS DE DOS CLAVES (CIFRADORES ASIMÉTRICOS)

Los sistemas de cifrado asimétrico fueron creados con la finalidad de dar solución al problema de intercambio de la clave secreta de los sistemas de cifrado simétricos, el cifrado asimétrico también conocido como de doble clave o de clave pública consiste en utilizar dos claves, una para cifrar y otra para descifrar.

La clave que se utiliza para cifrar el mensaje es la clave pública del receptor, y es pública porque es conocida por más personas que sólo el emisor y el receptor de un determinado mensaje. El descifrado se lleva a cabo por el receptor y lo hace con su clave privada, lo que implica que ya no hay un intercambio de claves ya que cuando un mensaje es cifrado con la clave pública del receptor, se está asegurando que sólo él puede descifrar el mensaje con su clave privada que se supone sólo está en su poder.

La clave pública de cada persona es derivada de su clave privada mediante funciones de un sentido, es decir que son fáciles de calcular en una dirección pero muy difíciles de calcular a la inversa, lo que hace posible que la clave para cifrar se haga pública.

La figura 2.46 muestra el esquema de un sistema de cifrado asimétrico.

FIGURA 2.46 Sistema criptográfico asimétrico

Es una práctica común utilizar los dos tipos de cifrado (simétrico y asimétrico) al momento de implementar un sistema criptográfico, con ello se busca aprovechar las mayores cualidades de cada uno; por un lado se intercambia de forma segura la clave con el cifrado asimétrico y por el otro el proceso cifrado/descifrado se realiza con rapidez y eficiencia con el cifrado simétrico.

La figura 2.47 muestra una técnica llamada *Sobre Digital*, la cual es utilizada para proteger la confidencialidad de una clave simétrica con la ayuda de un algoritmo asimétrico.

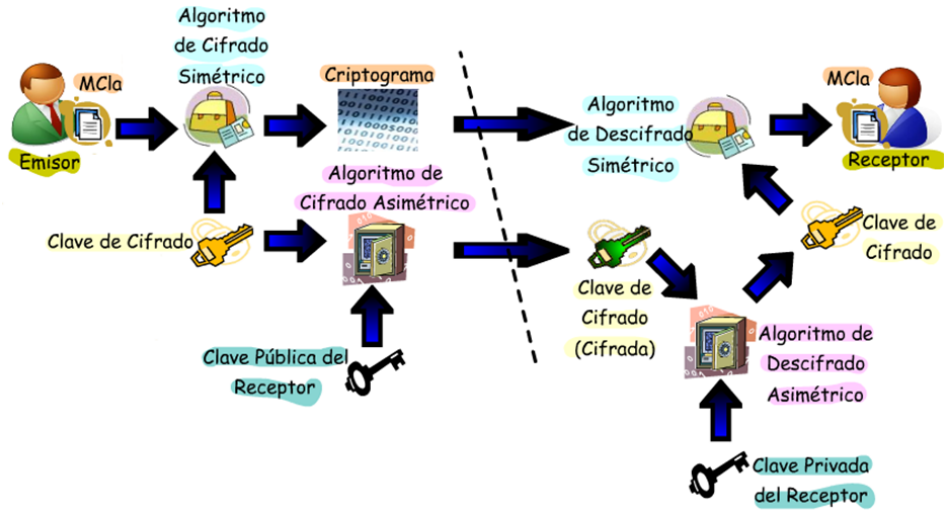


FIGURA 2.47 Sobre digital

El emisor utiliza una clave secreta para cifrar el MCla y por tanto un algoritmo simétrico, también cifra dicha clave pero con un algoritmo asimétrico y con la clave pública del receptor, con el objetivo de que solo éste pueda recuperarla ya que él es el único que tiene la clave privada para hacerlo, una vez que el receptor descifre la clave secreta podrá descifrar el mensaje con dicha clave.

2.2.4 FORMAS DE PROCESAMIENTO DE DATOS

Los sistemas de cifrado también se pueden clasificar de la siguiente manera:

- **Cifrador serial, continuo o en flujo**: El texto original es cifrado carácter por carácter.
- **Cifrador por bloque**: El texto original es dividido en grupos de caracteres (bloques) para ser cifrado.

CIFRADOR EN FLUJO

Los cifradores en flujo están formados por:

1. **Un generador de claves**: a partir de una clave de inicialización K produce una secuencia de bits igual a la longitud del mensaje, dicha secuencia de bits es empleada como la clave en el proceso de cifrado/descifrado. Tanto emisor como receptor cuentan con un generador de claves, los cuales producen claves idénticas en ambos extremos de la comunicación. En el subtema (Gestión de Claves) sección 2.3.3 (Generadores y distribución de claves) del presente capítulo se habla de cómo son generadas dichas claves.

2. **El algoritmo de cifrado**: realiza operaciones elemento a elemento, es decir que el algoritmo de cifrado se va aplicando a un elemento de información del MCla con un elemento de la clave (ya sean bits o caracteres según se esté trabajando), para obtener así el criptograma.

Los cifradores en flujo son apropiados para utilizarlos en los sistemas de comunicaciones de tiempo real, como lo es la telefonía móvil digital, debido a que el proceso de cifrado/descifrado se realiza elemento a elemento.

La figura 2.48 muestra el procedimiento del cifrado en flujo.

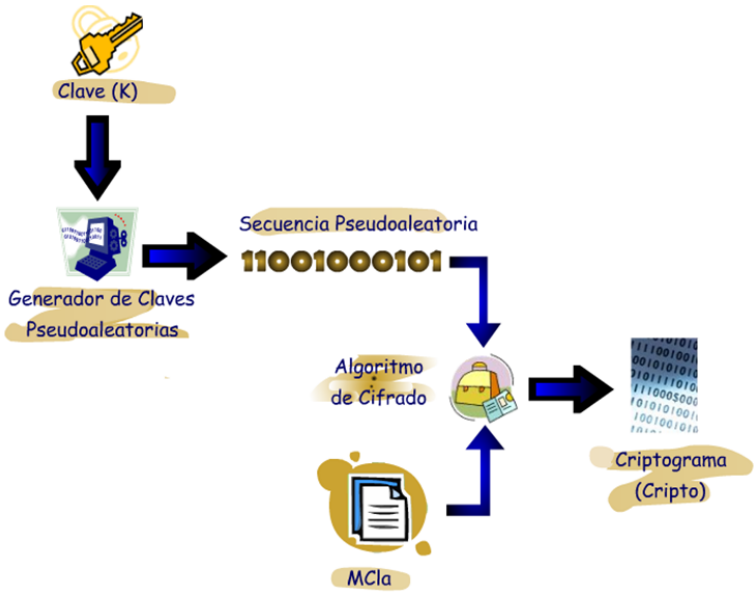


FIGURA 2.48 Cifrador en flujo

El cifrado de Vernam es tomado como referencia para muchos de los procedimientos de cifrado en flujo, recordemos que Vernam propuso que la clave fuera utilizada una sola vez, fuera aleatoria y tan larga como el mensaje a cifrar. Sin embargo, la secuencia generada por un generador de claves en realidad es una secuencia pseudoaleatoria ya que genera las secuencias mediante métodos determinísticos lo que ocasiona que la secuencia generada presente periodicidades, por lo que no cumple con la regla de Vernam de que la clave sea aleatoria, es por ello que se dice que el cifrado en flujo es una aproximación al cifrado de Vernam el cual es un cifrado perfectamente seguro según la teoría de la información.

Los cifradores en flujo pueden ser diferenciados en dos tipos:

1. **Cifrado en flujo síncrono**: la secuencia pseudoaleatoria es independiente del mensaje tal y como se muestra en la figura 2.49. Para poder realizar el proceso de descifrado correctamente tanto emisor como receptor deben tener señales de sincronización lo cual otorga la ventaja de que si se tiene un ataque en donde se inserten mensajes erróneos, es posible detectarlos ya que ello interrumpe la sincronía.

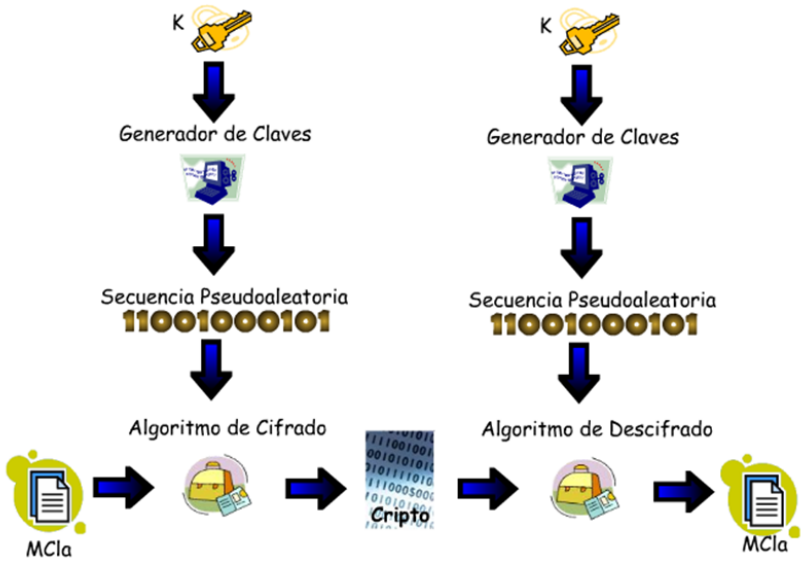


FIGURA 2.49 Cifrado en flujo síncrono

2. **Cifrado en flujo autosincronizante**: la secuencia pseudoaleatoria está en función del mensaje tal y como se muestra en la figura 2.50. En este caso no se necesitan señales de sincronización entre el emisor y receptor ya que en caso de pérdida de sincronía, ésta se recupera debido a la retroalimentación; la desventaja que tienen este tipo de cifradores es que son altamente vulnerables a un ataque de inserción de mensajes, pero ello se puede evitar enviando mensajes adicionales de identificación de mensaje.

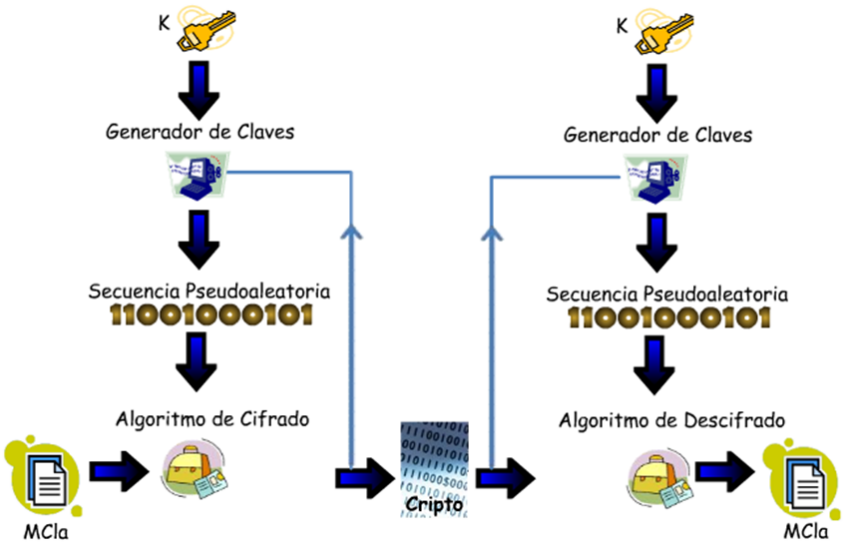


FIGURA 2.50 Cifrado en flujo autosincronizante

CIFRADOR POR BLOQUES

Este tipo de cifradores, realizan el cifrado por grupos de caracteres llamados bloques, en la actualidad se suele trabajar con grupos de bits debido a que los mensajes a cifrar

se codifican a esta forma previamente (utilizando el código ANSII, por ejemplo), es decir que cada algoritmo de cifrado/descifrado procesa un bloque de tamaño n a la vez, produciendo un bloque de tamaño n de salida por cada bloque de entrada. Como ejemplo de este tipo de cifradores están los algoritmos de sustitución monoalfabética poligrámica.

En la figura 2.51 se muestra el procedimiento del cifrado por bloques.

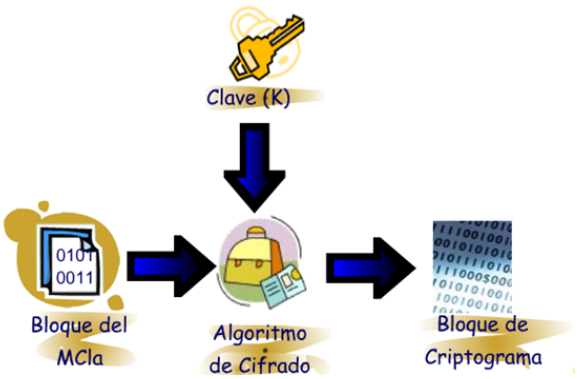


FIGURA 2.51 Cifrado por bloques

2.3 GESTIÓN DE CLAVES

2.3.1 POLÍTICAS DE GESTIÓN DE CLAVES

Una política de gestión de claves es un conjunto de reglas que establecen el modo de generación, almacenamiento, distribución, borrado, actualización, recuperación, protección y aplicación de claves en una red, en dicha política también se establece quién es la persona o grupo de personas autorizadas a realizar cada una de estas acciones.

Motivos por los que se debe establecer una política de gestión de claves:

- Es necesario renovar las claves frecuentemente ya que una clave queda expuesta cada vez que se usa.
- Se deben emplear claves diferentes para servicios diferentes (autenticación, transmisión, almacenamiento, etc.) con el fin de minimizar la exposición de las claves.
- Deben asignarse claves diferentes a cada persona o grupo que acceden a una red, de tal manera que sólo las personas autorizadas tengan acceso a determinada información.
- Las claves que por alguna razón se vuelven no seguras o aquellas que ya no son usadas por algún usuario o grupo deben ser eliminadas del sistema para evitar comprometer la información.

- **CICLO DE VIDA DE UNA CLAVE**

La figura 2.52 muestra el ciclo de vida de una clave:

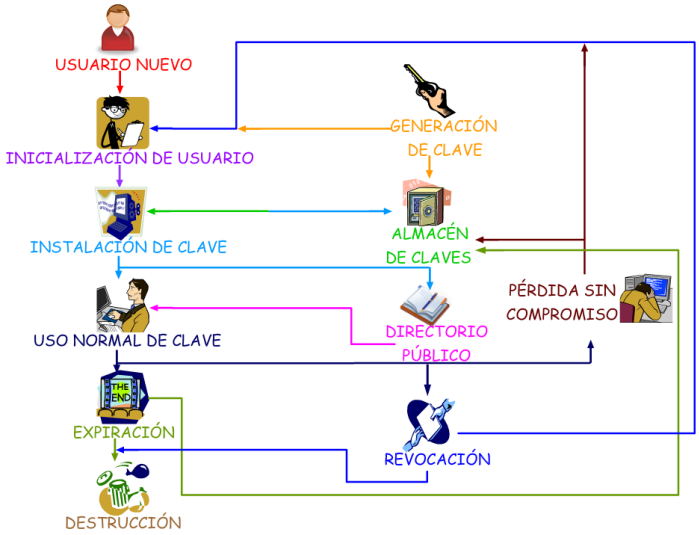


FIGURA 2.52 Ciclo de vida de una clave

- **GENERACIÓN**

Las claves pueden ser seleccionadas por el usuario o generadas automáticamente con la ayuda de generadores de claves de los cuales se habla en la sección 2.3.3 (Generadores y distribución de claves) del presente capítulo.

Cuando una clave es generada por el usuario se deben tomar en cuenta las siguientes buenas prácticas:

- Construirlas con letras mayúsculas, minúsculas, caracteres especiales y dígitos
- Evitar utilizar palabras de diccionario
- Longitud mínima de ocho dígitos
- No dejarlas en lugares visibles
- No contener información personal como fechas, nombres, gustos, etc.
- Fáciles de recordar pero difíciles de adivinar
- No divulgarlas
- No escribirlas en papel

- **ALMACENAMIENTO**

Se refiere a la ubicación que tendrán todas las claves de la red.

- **DISTRIBUCIÓN**

Se refiere a la manera en que el emisor envía la clave al receptor de un determinado mensaje para que pueda descifrarlo. Actualmente existen varias formas de hacerlo y ello se explica en la sección 2.3.3 (Generadores y distribución de claves) del presente capítulo.

- **BORRADO**

Se deben eliminar las claves que por alguna razón se consideren ya no son seguras o que ya no estén en uso en el sistema, este proceso lo debe realizar el administrador de la red.

- **ACTUALIZACIÓN**

La actualización la puede realizar el propio usuario que por alguna razón decida hacerlo, o bien la puede realizar el administrador de la red que con base en las políticas deba actualizar las claves.

- **RECUPERACIÓN**

Cuando un usuario se olvida de su contraseña y no existe alguna razón para desecharla, es posible volver a proporcionar la misma clave al usuario para que cumpla con su ciclo de vida, en la política de gestión de claves se debe contemplar este caso y establecer a detalle bajo qué condiciones una clave es recuperada.

- **PROTECCIÓN**

Es recomendable cifrar las claves antes de ser almacenadas para que en caso de una violación al acceso de dichas claves no represente un riesgo en la confidencialidad en las mismas, en la política de gestión de claves se debe establecer el algoritmo para cifrarlas así como las claves utilizadas.

- **APLICACIÓN**

Se refiere a la utilidad que tendrá cada una de las claves generadas.

2.3.2 TIPOS DE CLAVES

- **CLAVE ESTRUCTURAL**

A cada nivel de privilegios en la red le es asignada una clave estructural evitando así la comunicación entre entidades con distintos privilegios. La clave estructural es implementada en hardware o en memoria ROM o similar.

- **CLAVE MAESTRA**

Es generada aleatoriamente ya sea de forma manual o con un generador automático de claves, puede ser modificada por el usuario (el administrador de seguridad informática) y se usa para cifrar únicamente claves secundarias. Se almacena sin cifrar en un módulo de seguridad el cual se muestra en la figura 2.53.

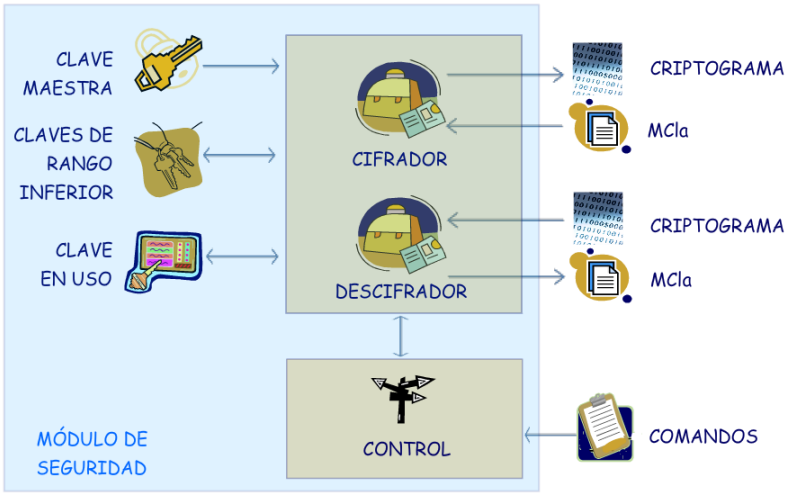


FIGURA 2.53 Módulo de seguridad de un equipo criptográfico

Un módulo de seguridad es un circuito integrado o bien una tarjeta chip en donde se almacena la clave maestra, el algoritmo de cifrado y descifrado y en ocasiones claves de rango menor a la maestra lo cual resulta poco aconsejable ya que resulta ser muy caro. Este módulo debe ser resguardado en un lugar seguro (físico) de la organización y sólo debe tener acceso a él personal encargado de la seguridad de la información.

- **CLAVE PRIMARIA**

Clave generada con la clave maestra que puede ser almacenada en una memoria no tan protegida como el módulo de seguridad, generalmente es utilizada para acceder a los sistemas o servicios.

- **CLAVE DE GENERACIÓN**

Es una clave primaria utilizada para generar claves de sesión o claves de archivos con la finalidad de protegerlas en la transmisión y almacenamiento.

- **CLAVE DE SESIÓN O DE MENSAJE**

Clave creada con una clave de generación, utilizada para iniciar una sesión o bien para cifrar los datos intercambiados entre dos entidades durante su conexión, una vez terminada la sesión la clave se destruye.

- **CLAVE DE CIFRADO DE ARCHIVOS**

Clave cifrada con una clave de generación, su finalidad es cifrar archivos. Es utilizada únicamente en el cifrado de un archivo y después se destruye.

La figura 2.54 muestra la jerarquía de los tipos de claves:

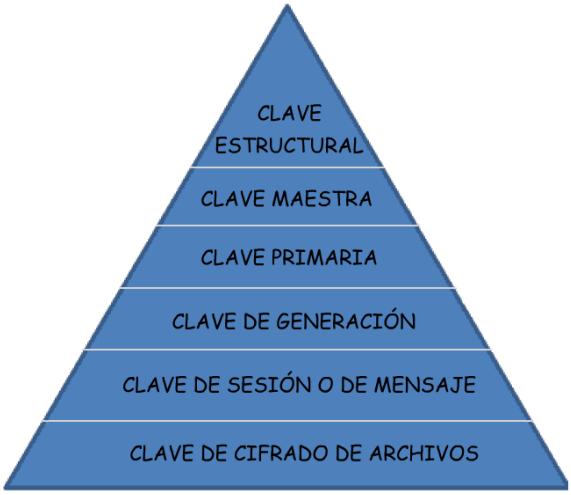


FIGURA 2.54 Jerarquía de claves

2.3.3 GENERADORES Y DISTRIBUCIÓN DE CLAVES

Las claves pueden ser creadas por el usuario o generadas automáticamente con la ayuda de generadores de claves los cuales se clasifican en dos tipos:

a) **Generadores aleatorios**: para generar secuencias cifrantes utilizan datos provenientes de ruido físico aleatorio (ruido de un micrófono, ruido térmico en un semiconductor, etc.) o bien provenientes del estado de una computadora (interrupciones, posición del ratón, actividad en la red, uso del teclado, etc.). Es conveniente combinar varias técnicas para que la secuencia resultante sea imposible de predecir. Este tipo de generadores se utilizan para generar claves cortas.

b) **Generadores pseudoaleatorios**: este tipo de cifradores no son totalmente aleatorios ya que para generar una secuencia obedecen a algún algoritmo o cierto procedimiento repetitivo.

La distribución de claves se refiere a los medios utilizados para distribuir una clave a dos entidades que quieran intercambiar datos. La distribución de claves es un tema primordial en un sistema de cifrado ya que de ello depende que las claves sólo sean conocidas por las entidades indicadas y así el método de cifrado sea efectivo.

Técnicas de Distribución de Claves:

- Distribución manual
- Distribución basada en centro
- Distribución basada en certificado

GENERADORES PSEUDOALEATORIOS

Existen cuatro requerimientos generales para que una secuencia de unos y ceros se considere segura para utilizarla como clave en Criptografía:

1. **Período**: la longitud de la secuencia de unos y ceros es decir de la clave debe ser al menos la longitud del mensaje en claro.

2. **Distribución de unos y ceros**: si se toman varias muestras de cierta longitud de una secuencia dichas muestras están uniformemente distribuidas a lo largo de toda la secuencia.

3. **Imprevisibilidad**: si consideramos una muestra de la secuencia, un criptoanalista no debería poder predecir el dígito siguiente con una probabilidad de acierto mayor a 0.5.

4. **Facilidad de implementación**: la velocidad de generación, costo, tamaño, consumo y otros factores no deben ser impedimentos para generar las secuencias pseudoaleatorias con medios electrónicos.

Golomb estableció tres postulados que debe cumplir una secuencia binaria para considerarse pseudoaleatoria, a continuación se exponen concretamente cada uno de dichos postulados:

1. **G1**: la probabilidad de aparecer a lo largo de la secuencia es igual para unos y ceros, es decir que el número de unos debe ser igual al número de ceros o bien la diferencia debe ser de uno.

2. **G2**: un cero tiene la misma probabilidad que un uno de aparecer después de un cierto n-grama (muestra de n dígitos consecutivos).

3. **G3**: el cómputo de coincidencias entre una secuencia y su versión desplazada no debe aportar ninguna información sobre el período de la secuencia.

En la práctica, uno de los procedimientos utilizados para generar una secuencia pseudoaleatoria es empleando registros de desplazamiento. Un registro de desplazamiento es una memoria de n celdas cuyo contenido se desplaza de acuerdo a los pulsos de un reloj de control, al mismo tiempo dicho contenido se va mezclando mediante operaciones establecidas y el resultado retroalimenta a la última celda. De acuerdo a estas características los registros de desplazamiento son de varios tipos; los principales se enlistan a continuación:

- **SR (Shift Register)**: registro de desplazamiento con filtrado de estados y con alimentación externa. El esquema de un SR se muestra en la figura 2.55.

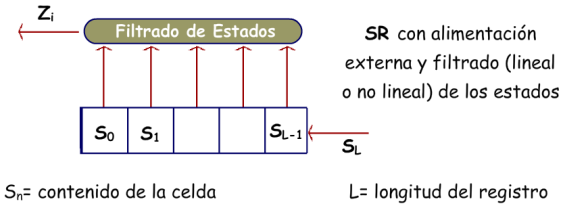


FIGURA 2.55 Registro de desplazamiento SR

- **FSR (Feedback Shift Register)**: registro con retroalimentación. El esquema de un FSR se muestra en la figura 2.56.

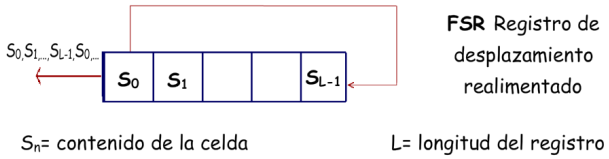


FIGURA 2.56 Registro de desplazamiento FSR

- **NLFSR (Non-Lineal Feedback Shift Register)**: registro con retroalimentación no lineal. El esquema de un NLFSR se muestra en la figura 2.57.

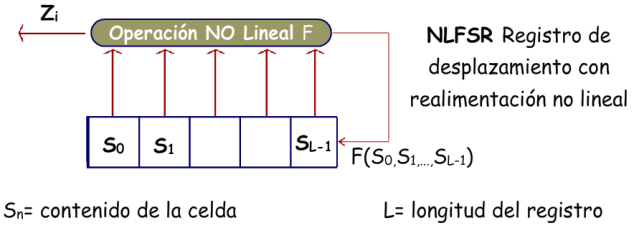


FIGURA 2.57 Registro de desplazamiento NLFSR

- **LFSR (Lineal Feedback Shift Register)**: registro con retroalimentación lineal. Estos registros proporcionan un alto grado de aleatoriedad, es por ello que son muy utilizados en Criptografía. El esquema de un LFSR se muestra en la figura 2.58.

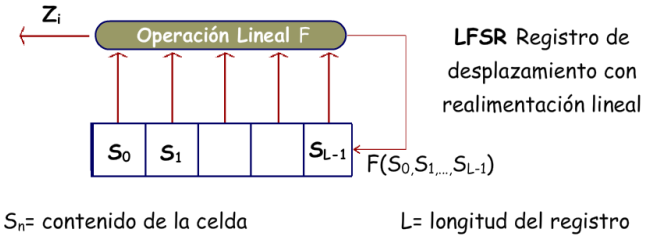


FIGURA 2.58 Registro de desplazamiento LFSR

Entre más grande es la longitud de un LSFR mayor es el periodo de las secuencias que genera, pero en la práctica es complejo implementar un LSFR de tamaño muy grande además de que representaría pérdida de mucha información en caso de que se perdiera la sincronía, es por ello que se han planteado varias formas de obtener secuencias de elevado periodo utilizando LSFRs de pequeña longitud. La figura 2.59 muestra dos de las formas en como son utilizados los LSFRs con este propósito:

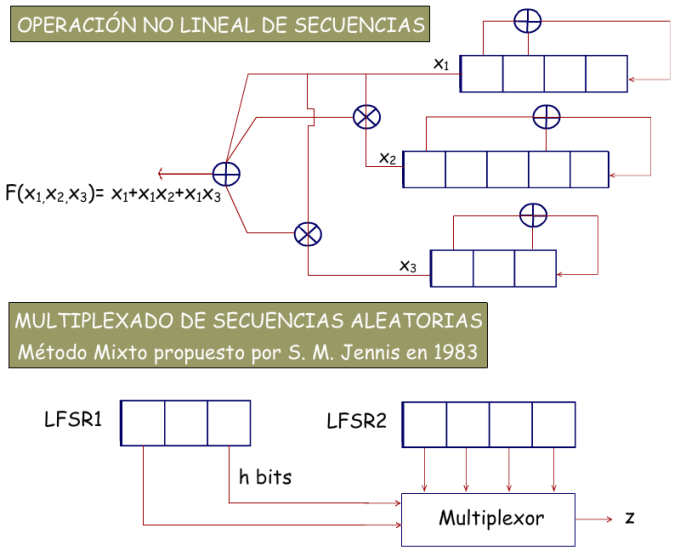


FIGURA 2.59 Empleo de LFSRs para obtener secuencias de periodo grande

Otros métodos para la generación de secuencias pseudoaleatorias se basan en la complejidad algorítmica, utilizando funciones de una sola dirección cuyo cálculo directo es fácil de obtener, pero el cálculo de sus inversas tienen una alta complejidad computacional.

DISTRIBUCIÓN DE CLAVES

- **DISTRIBUCIÓN MANUAL**

El envío de la clave no es por la línea de comunicación por la cual se mandan los mensajes cifrados, sino que se utilizan otros métodos, por ejemplo:

- Realizando la suma módulo dos de varias claves enviadas por distintos medios por ejemplo: carta certificada+vía telefónica+fax.

- Utilizando un inyector de claves; éste es un pequeño aparato en donde se almacena una clave la cual puede ser transferida una o más veces a un equipo, tiene un contador que registra el número de veces que la clave es transferida por lo que se puede controlar el número de instalaciones de la clave en otros equipos, el inyector debe ser trasportado por medio de una tercera entidad de gran confianza y de preferencia que no sea experto en el tema.

Este tipo de métodos dejan de ser prácticos cuando la cantidad de claves que se deben mandar o las distancias que se deban recorrer para realizar la entrega son muy grandes, lo cual hace que este método sea lento, caro y poco seguro.

- **DISTRIBUCIÓN BASADA EN CENTRO**

Las dos entidades interesadas en intercambiar datos tienen una conexión cifrada con una tercera entidad de confianza, esta tercera entidad es la encargada de entregar la clave a través de los enlaces cifrados a las otras dos entidades.

La figura 2.60 muestra diversos esquemas de la distribución basada en centro.

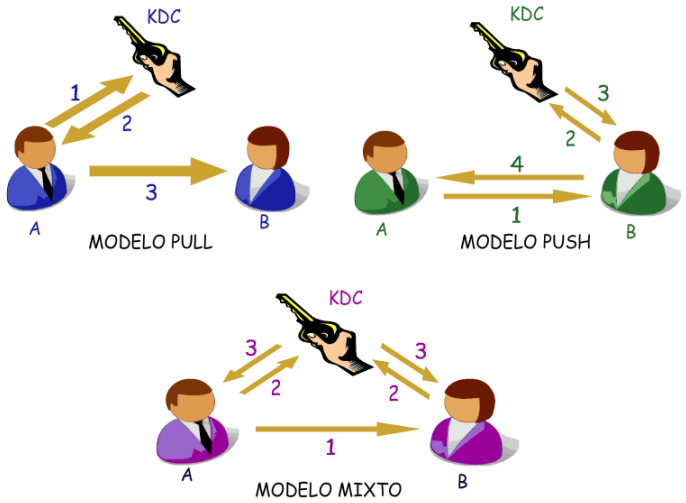


FIGURA 2.60 Distribución basada en centro

El modelo PULL requiere que el emisor A obtenga la clave de sesión del KDC, antes de comunicarse con B.

1. A solicita una clave de sesión al KDC.
2. El KDC envía a A la clave de sesión que utilizará para comunicarse con B y un paquete cifrado para que A lo entregue a B, dicho paquete está cifrado con la clave que sólo conocen B y el KDC y contiene la clave de sesión con la que B se comunicará con A así como un identificador de A.
3. A envía a B el paquete que le envío el KDC para B.

El modelo PUSH requiere que A primero contacte a B y después B debe obtener la clave de sesión del KDC.

1. A se comunica con B y le hace saber que requiere establecer una sesión.
2. B solicita una clave de sesión al KDC.
3. El KDC envía a B la clave de sesión que utilizará para comunicarse con A y un paquete cifrado para que B lo entregue a A, dicho paquete está cifrado con la clave que sólo conocen A y el KDC y contiene la clave de sesión con la que A se comunicará con B así como un identificador de B.
4. B envía a A el paquete que le envío el KDC para A.

El modelo mixto es la combinación del modelo PULL y el PUSH.

1. A se comunica con B y le hace saber que requiere establecer una sesión.
2. A y B solicitan una clave de sesión al KDC.
3. El KDC envía a A y B la clave de sesión que utilizarán para comunicarse.

Centro de distribución de claves (KDC – Key Distribution Center): verifica qué equipos tienen permiso de comunicarse con otros, cuando la conexión está permitida el KDC se

encarga de dar una clave de sesión para dicha conexión. El KDC puede ser una entidad centralizada en la red o ser un servicio distribuido en varios nodos.

Un centro de traducción de claves (KTC – Key Translation Center) está formado por el KDC y las entidades que desean establecer una sesión. La figura 2.61 muestra el esquema de un KTC.

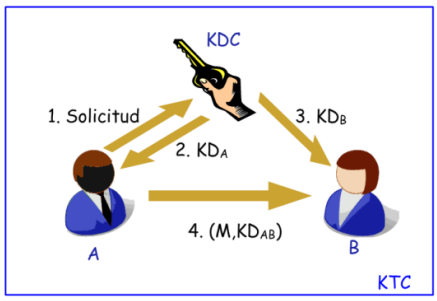


FIGURA 2.61 KTC

- **DISTRIBUCIÓN BASADA EN CERTIFICADO**

Podemos diferenciar dos técnicas para la distribución basada en certificado:

1. **Transferencia de claves**: El emisor genera localmente una clave y la cifra con un algoritmo asimétrico utilizando la llave pública del receptor, con el objetivo de que solo éste pueda recuperarla y así protegerla durante su transmisión.

   La figura 2.62 muestra el esquema de esta técnica.

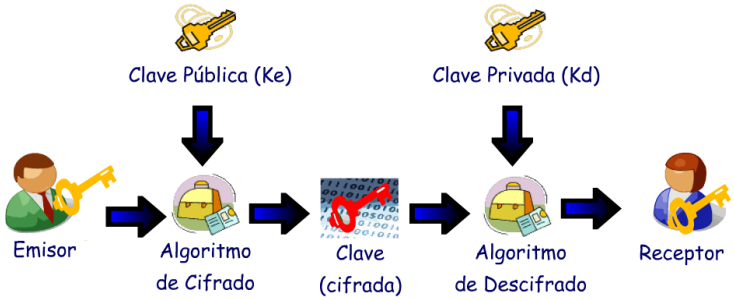


FIGURA 2.62 Transferencia de claves

2. **Intercambio de claves o acuerdo de claves**: la clave es generada por las dos entidades involucradas en la comunicación.

Dentro del esquema de distribución de claves basada en certificado, una autoridad de certificación (CA) debe autenticar las claves públicas de las entidades que desean intercambiar claves secretas, las claves públicas son parte de la información que proporciona un certificado. Por ejemplo identifiquemos a las dos entidades que intercambiarán claves como A y B y a la CA la llamaremos D, si A y B tienen certificados de la misma CA (en este caso D), A puede estar seguro de que una

determinada clave pública pertenece a B, obteniendo el certificado de B y comprobándolo con la clave pública de D.

2.4 CRIPTOGRAFÍA SIMÉTRICA O DE CLAVE SECRETA

2.4.1 INTRODUCCIÓN A LA CRIPTOGRAFÍA SIMÉTRICA

CARACTERÍSTICAS DE LOS ALGORITMOS SIMÉTRICOS

- La clave es la misma para cifrar que para descifrar un mensaje, por lo que sólo el emisor y el receptor deben conocerla.
- Se basan en operaciones matemáticas sencillas, por ello son fácilmente implementados en hardware.
- Debido a su simplicidad matemática son capaces de cifrar grandes cantidades de datos en poco tiempo.

HERRAMIENTAS MATEMÁTICAS

- **OPERACIONES LÓGICAS**

Este tipo de operaciones tienen la característica de que se aplica una función lógica por cada bit del operando, sin tomar en cuenta a los bits restantes.

- **OR**

Actúa sobre dos bits de acuerdo a la siguiente tabla lógica (figura 2.63).

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

FIGURA 2.63 Operación OR

Cuando uno de los dos bits es uno, el resultado es uno.

- **XOR**

Actúa sobre dos bits de acuerdo a la siguiente tabla lógica (figura 2.64).

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

FIGURA 2.64 Operación XOR

Cuando un bit es uno y el otro es cero el resultado es uno, cuando los dos operadores tienen el mismo valor el resultado es cero.

➢ **OTRAS**

Otras operaciones lógicas comúnmente usadas además de la OR y XOR son las mostradas en la figura 2.65.

A	B	AND
0	0	0
0	1	0
1	0	0
1	1	1

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

A	B	NOR
0	0	1
0	1	0
1	0	0
1	1	0

A	B	XNOR
0	0	1
0	1	0
1	0	0
1	1	1

B	NOT
0	1
1	0

FIGURA 2.65 Operaciones lógicas más comunes

- **CORRIMIENTOS O DESPLAZAMIENTOS**

Los desplazamientos se realizan recorriendo los bits de una palabra, dato o registro hacia la derecha o izquierda. Existen varios tipos:

- Desplazamientos lógicos
- Desplazamientos circulares
- Desplazamientos aritméticos
- Desplazamientos concatenados

➢ **DESPLAZAMIENTOS LÓGICOS**

Los valores que se van recorriendo se desechan y los extremos se completan con ceros. También es posible completar con unos aunque es menos común. La figura 2.66 muestra los desplazamientos lógicos a la derecha y a la izquierda.

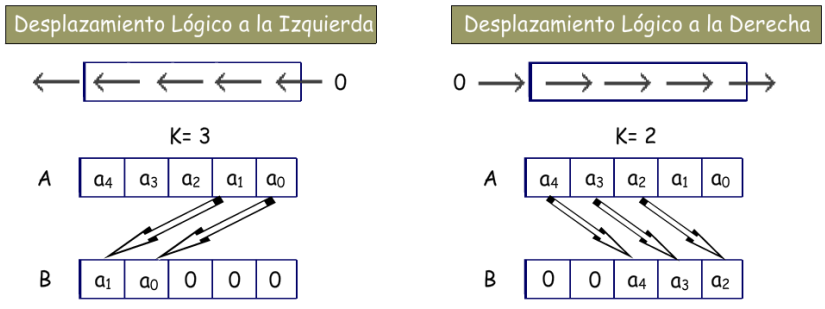


FIGURA 2.66 Representación gráfica del desplazamiento lógico a la izquierda y a la derecha

- **DESPLAZAMIENTOS CIRCULARES**

Los valores que se van recorriendo no se pierden ya que se colocan al extremo opuesto del que van saliendo. La figura 2.67 muestra los desplazamientos circulares a la derecha y a la izquierda.

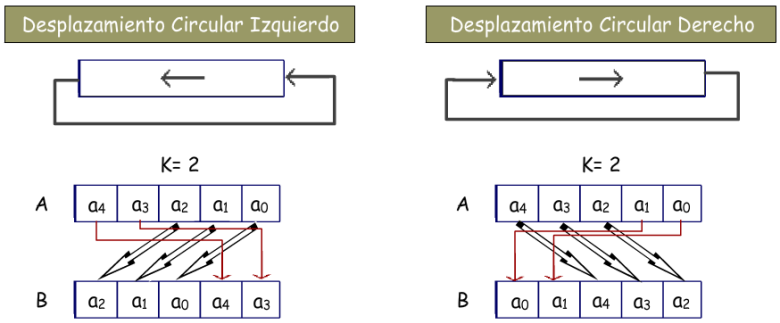


FIGURA 2.67 Representación gráfica del desplazamiento circular izquierdo y derecho

- **DESPLAZAMIENTOS ARITMÉTICOS**

Son parecidos a los desplazamientos lógicos sólo que el bit de signo se mantiene intacto. En la figura 2.68 se muestra el desplazamiento aritmético a la derecha y a la izquierda.

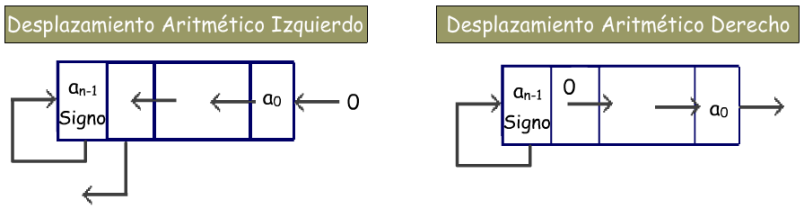


FIGURA 2.68 Representación gráfica del desplazamiento aritmético izquierdo y derecho

- **DESPLAZAMIENTOS CONCATENADOS**

Consisten en desplazamientos que afectan a un conjunto concatenado de dos o más elementos.

- **SISTEMAS DE NUMERACIÓN**

Un sistema de numeración posicional se caracteriza porque un dígito depende tanto de su valor absoluto como de su posición en el número.

Notación de un número entero: $N = d_0 r^n + d_1 r^{n-1} + ... + d_n r^0$

Donde:

r= radical o base del sistema de numeración. Especifica cuantos símbolos únicos existen en ese sistema.

d_n= entero

- **DECIMAL**

Número de símbolos únicos y base del sistema: *r*= 10

Símbolos únicos: $d_n \in \{0,1,2,3,4,5,6,7,8,9\}$

- **Conversión de base 10 a cualquier base**

Se divide el número en base 10 entre *r* de la base a la que se desea convertir tantas veces hasta que el cociente de la división sea cero tal y como se muestra en la figura 2.69. El resultado se obtiene de la columna de los residuos, el último residuo es el primer dígito en la nueva base y el primer residuo es el último dígito en la nueva base. A este método se le llama división por radical.

Ejemplo:
Convertir 14_{10} a base 2

	Cociente	Residuo
14/2	7	0
7/2	3	1
3/2	1	1
1/2	0	1
$14_{10}= 1110_2$		

FIGURA 2.69 Ejemplo de conversión de base 10 a base 2

- **Conversión de cualquier base a base 10**

Se multiplica cada dígito del número por la base en la que está dicho número elevada a la potencia de la posición del dígito (véase figura 2.70), finalmente se suman los resultados de todas las multiplicaciones.

Ejemplo:
Convertir 1110_2 a base 10

Base 2		1	1	1	0
Posición ←	...	3	2	1	0
$1x2^3 + 1x2^2 + 1x2^1 + 0x2^0 = 8 + 4 + 2 + 0= 14$					
$1110_2= 14_{10}$					

FIGURA 2.70 Ejemplo de conversión de base 2 a base 10

- **BINARIO**

Número de símbolos únicos y base del sistema: *r*= 2
Símbolos únicos: $d_n \in \{0,1\}$

Todas las computadoras que se fabrican actualmente, procesan y almacenan la información mediante el uso del sistema binario debido a la confiabilidad que proporciona el uso de sólo dos estados de energía, en donde un estado de energía representa un uno y otro un cero.

- **HEXADECIMAL**

Número de símbolos únicos y base del sistema: *r*= 16

Símbolos únicos: $d_n \in \{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$

El sistema hexadecimal es una notación condensada del sistema binario, que no es otra cosa más que una representación más corta y entendible de los dígitos binarios. Cada grupo de cuatro dígitos binarios se convierte en un dígito hexadecimal. La figura 2.71 muestra la tabla de conversión de hexadecimal a binario.

Hexadecimal	Binario
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

FIGURA 2.71 Tabla de conversión de base 16 a base 2

- **TEORÍA DE GRUPOS**

Un grupo *G* es un conjunto provisto de una operación asociativa que tiene elemento neutro y respecto a la cual cada elemento de *G* tiene inverso.

- Si un grupo presenta la propiedad conmutativa, se dice que el grupo es un grupo abeliano o conmutativo.
- El orden de un grupo finito es el número de elementos que conforman dicho grupo.
- Un elemento *g* de un grupo es un generador si cualquier elemento del grupo se puede escribir como una potencia de *g*, es decir, si $G = \{g^0 = 1, g^1 = g, g^2, ..., g^n, ...\}$. En este caso *G* es un grupo cíclico generado por *g*.
- *Teorema de Lagrange*: el orden de un subgrupo de un grupo finito divide al orden del grupo.
- Un grupo finito cuyo orden es un número primo no puede tener subgrupos propios.

- **TEORÍA DE CAMPOS**

Un campo es un conjunto *K* provisto de dos operaciones (suma "+" y producto "."), que satisfacen las siguientes propiedades:

- $(K,+)$ es un grupo aditivo conmutativo, se llama grupo aditivo del campo.

- $(K^* = K - \{0\}, \cdot)$ es un grupo conmutativo, es llamado grupo multiplicativo del campo.
- El producto tiene la propiedad distributiva respecto a la suma: $(a)\cdot(b+c) = a\cdot b + a\cdot c$

Teorema:

(i) El número de elementos de un campo finito *K* debe ser igual a la potencia de un número primo *p.* El entero *p* recibe el nombre de característica del campo y éste se representa por *GF* (p^m) (GF= Campo de Galois).

(ii) Sólo hay un campo finito p^m elementos. De hecho, fijado un polinomio irreducible *F(x)* de grado *m* con coeficientes en Z_p, los elementos de *GF* (p^m) se representan como polinomios con coeficientes en Z_p de grado < *m*; es decir:

$$GF(p^m) = \{\lambda_0 + \lambda_1 x + \lambda_2 x^2 + \cdots + \lambda_{m-1} x^{m-1}; \lambda_0, \lambda_1, \lambda_2, \lambda_{m-1} \in Z_p\}$$

- **SUSTITUCIÓN - CAJAS *S***

Si S_1 es la función definida en la tabla de la figura 2.72 y *B* es un bloque de 6 bits, entonces $S_1(B)$ está determinada como sigue:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

FIGURA 2.72 Caja S_1

- El primer y último bit de *B* representa en base 2 un número decimal en el rango de 0 a 3 (en binario de 00 a 11), llamaremos a dicho número decimal *i*.
- Los cuatro bits intermedios de *B* representan en base dos un número decimal en el rango de 0 a 15, llamaremos a dicho número decimal *j*.
- En la tabla se debe buscar el número en el *i*-ésimo renglón y la *j*-ésima columna. El número encontrado es un número decimal en el rango de 0 a 15, y debe encontrarse su equivalente en binario el cual es el resultado final.

Ejemplo:

Para *B*= 011011 y tomando en cuenta la tabla de la figura 2.72

- El primer bit de *B* es 0 y el último 1, formando "01" que en decimal es 1, indicando de este modo el renglón 1.
- Los cuatro bits intermedios de *B* son "1101" que en decimal es 13, indicando de este modo la columna 13.
- En la intersección del renglón 1 y columna 13 está el número 5, que en binario es 0101.

De este modo:
$S_1(B)=S_1(011011)= 0101$.
NOTA: En general el empleo de las cajas *S*, es el mostrado anteriormente independientemente del número de renglones y de columnas que empleen así como de los sistemas de numeración que utilicen.

- **PERMUTACIONES**

Las permutaciones consisten en realizar transposición de caracteres, es decir, se intercambian de lugar los caracteres que conforman la cadena de entrada de acuerdo a una tabla de referencia.

Un ejemplo de una tabla de referencia es la que se muestra en la tabla 2.73

5	20	10	13	11
1	2	8	5	7

FIGURA 2.73 Tabla de referencia de una permutación

- La tabla se recorre por renglones de arriba hacia abajo y recorriendo cada uno de izquierda a derecha.
- La primer casilla indica que el elemento número 5 de la cadena de entrada es el primer elemento de la cadena permutada.
- La segunda casilla indica que el elemento número 20 de la cadena de entrada es el segundo elemento de la cadena permutada.
- La tercer casilla indica que el elemento número 10 de la cadena de entrada es el tercer elemento de la cadena permutada.
- Se sigue el mismo procedimiento hasta terminar con todas las casillas.

Ejemplo:
Cadena de entrada: 10101011110100001110

Tomando como referencia la tabla de la figura 2.73, la cadena permutada queda tal y como se muestra en la tabla mostrada en la figura 2.74.

Pocisión de elemento	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Cadena de entrada	1	0	1	0	1	0	1	1	1	1	0	1	0	0	0	0	1	1	1	0
Cadena Permutada	1	0	1	0	0	1	0	1	1	1										

FIGURA 2.74 Ejemplo de permutación

Nótese que no necesariamente la longitud de la cadena permutada es de la misma longitud que la cadena original.

PRINCIPALES ALGORITMOS SIMÉTRICOS

- **IDEA (International Data Encryption Algorithm)**

Algoritmo de libre uso desarrollado y liberado en 1991 por Xuejia Lai y James L. Massey del Politécnico de Zurich. Opera con bloques de 64 bits y una clave de 128 bits realizando un total de 8 iteraciones. Una iteración del cifrado IDEA así como la transformación de salida final se muestra en la figura 2.75.

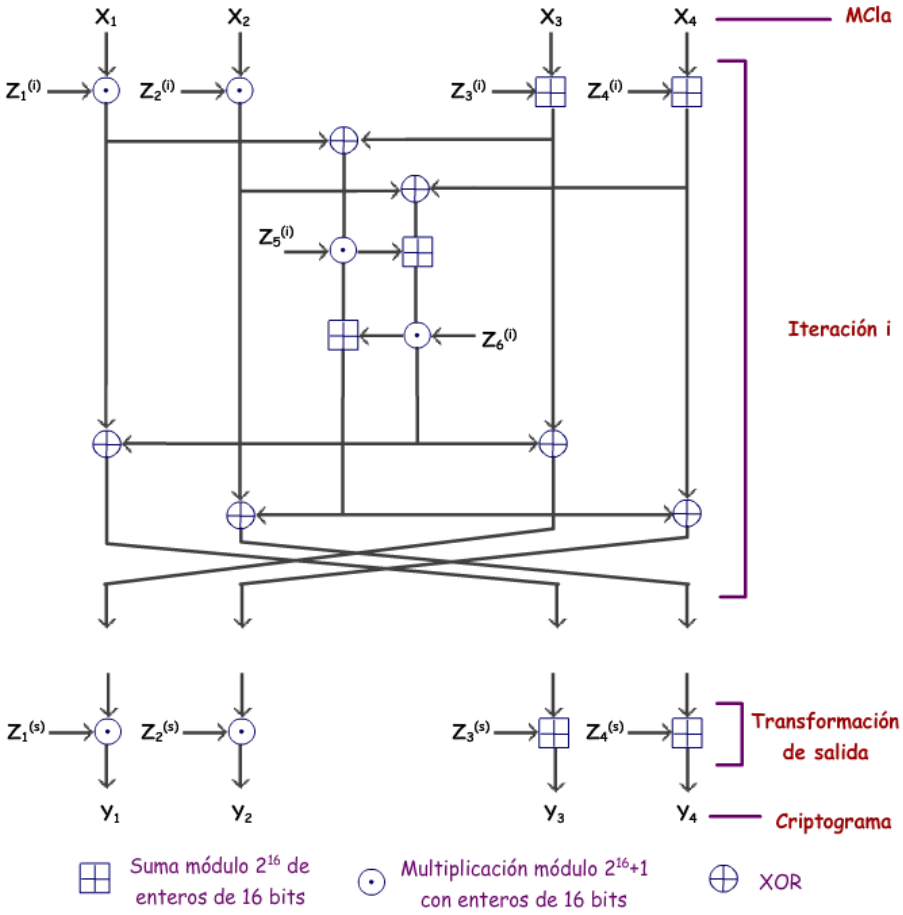


FIGURA 2.75 Algoritmo de cifrado IDEA

En cada una de las iteraciones se utilizan 6 subclaves de cifrado (en la figura 2.75 identificadas como $Z_n^{(i)}$ y cuatro en la transformación de salida haciendo un total de 52 subclaves en un cifrado (seis en cada una de las ocho iteraciones y cuatro de la transformación de salida) dichas subclaves son obtenidas a partir de la clave inicial de 128 bits de la siguiente manera:

- La clave inicial se divide en bloques de 16 bits formando así las primeras 8 subclaves.
- La clave inicial se desplaza 25 posiciones a la izquierda y se divide nuevamente en bloques de 16 bits para obtener las siguientes 8 subclaves.
- El paso anterior se repite hasta obtener las 52 subclaves.

- **BLOWFISH**

Desarrollado por B. Schneier, cifra bloques de 64 bits con una clave de longitud variable haciendo un total de 16 iteraciones en cada una de las cuales se realiza una función tal y como se muestra en la figura 2.76.

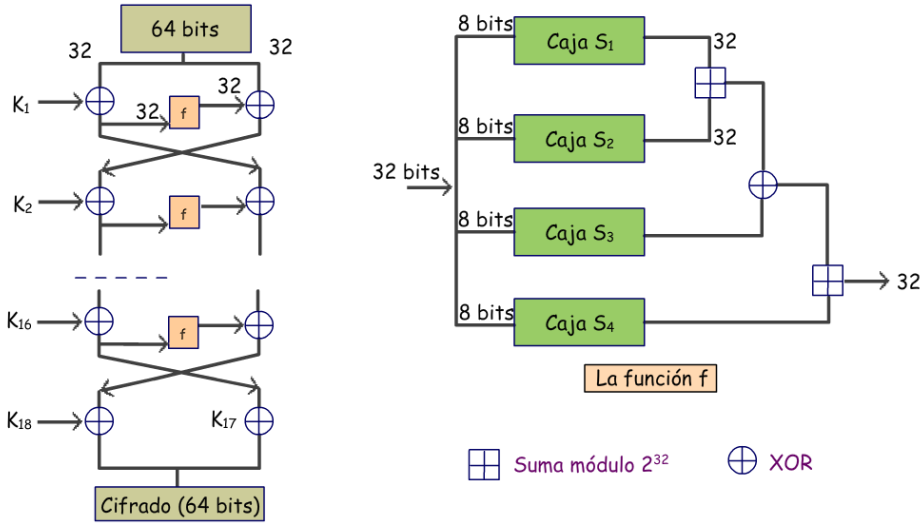


FIGURA 2.76 Cifrado BLOWFISH

- **RC5 (Rivest Cipher 5)**

Desarrollado por Ron Rivest, cifra bloques de longitud variable (w) utilizando claves de longitud variable (b) haciendo un número de iteraciones también variable (r).

Las operaciones que utiliza son:

- Suma módulo 2^w
- XOR bit a bit
- Corrimientos circulares, donde la rotación de x un número y de bits a la izquierda se denota por $x << y$.

La figura 2.77 muestra el diagrama de cifrado RC5:

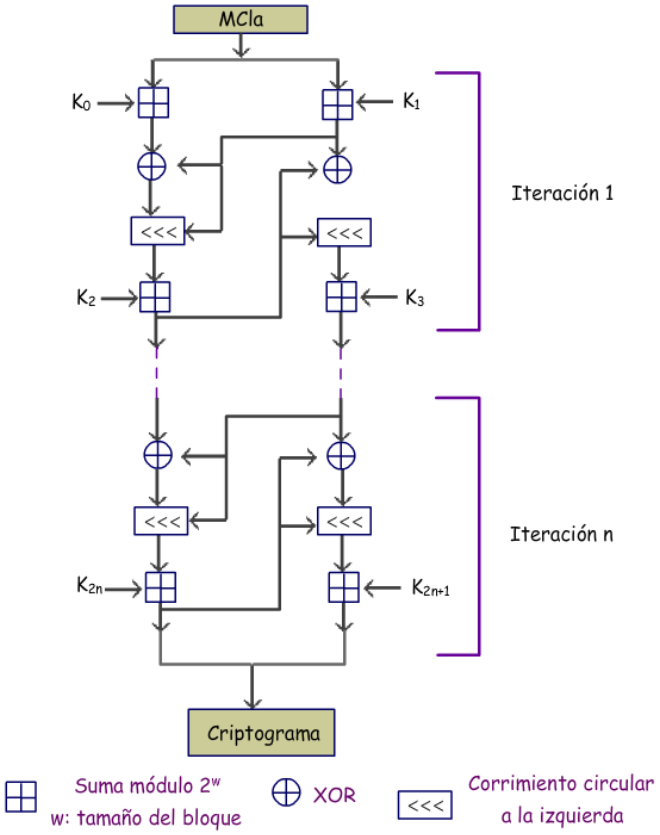


FIGURA 2.77 Cifrado RC5

- **DES (Data Encryption Standard)**

Fue creado en la década de los 70's. Es un cifrador en bloques que opera sobre grupos de datos de 64 bits, utiliza una clave de 56 bits y realiza un total de 16 rondas. Este algoritmo se explica con más detalle en la sección 2.4.2 (DES) del presente capítulo.

- **3DES o TDES**

Fue emitido por el NIST en 1999 como una versión mejorada de DES. Realiza tres veces el cifrado DES utilizando tres claves. La figura 2.78 muestra el diagrama que representa al algoritmo 3DES.

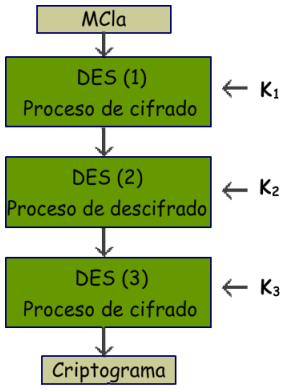


FIGURA 2.78 Triple DES

Cuando se descubrió que una clave de 56 bits (utilizada en el DES) no era suficiente para evitar un ataque de fuerza bruta, el 3DES fue elegido para agrandar la clave sin la necesidad de cambiar el algoritmo de cifrado.

Con tres claves distintas, 3DES tiene una longitud de clave efectiva de 168 bits aunque también se pueden usar dos claves haciendo $K_1=K_3$ (ver figura 2.78) con lo que se tiene una longitud de clave efectiva de 112 bits.

Actualmente el 3DES sigue siendo utilizado pero cada vez más está siendo sustituido por el algoritmo AES que ha demostrado ser muy robusto y más rápido.

- **AES (Advanced Encryption Standard)**

Algoritmo publicado por el NIST en el año 2001, opera sobre bloques de datos de 128 bits y la clave que utiliza puede ser de 128, 192 o 256 bits, el número de rondas que realiza depende del tamaño de la clave. En cualquier caso el criptograma siempre tiene la longitud de 128 bits. Utiliza primordialmente matemáticas polinomiales en estructuras de campos finitos. Este algoritmo se explica con más detalle en la sección 2.4.3 (AES) del presente capítulo.

2.4.2 DES (Data Encryption Standard)

ORÍGENES

- Desarrollado en 1975 por IBM como resultado de la convocatoria realizada por el NIST en la que se pedía el desarrollo de sistemas criptográficos.
- En 1977 fue adoptado como Estándar Federal de Procesamiento de la Información 46 (FIPS PUB-46) por el Buró Nacional de Estándares hoy en día conocido como Instituto Nacional de Estándares y Tecnología (NIST).
- Estandarizado en 1981 por la ANSI como ANSI X.3.92
- En 1994, el NIST establece que el DES seguirá siendo de uso federal por 5 años más.
- En 1998 fue descriptado por la Fundación de las Fronteras Electrónicas (EFF, Electronic Frontier Foundation) utilizando un ataque de fuerza bruta el cual duró 56 horas.

ALGORITMO DE CIFRADO

DES opera sobre bloques de datos de 64 bits y utiliza una clave de 56 bits. La figura 2.79 muestra el esquema del cifrado DES.

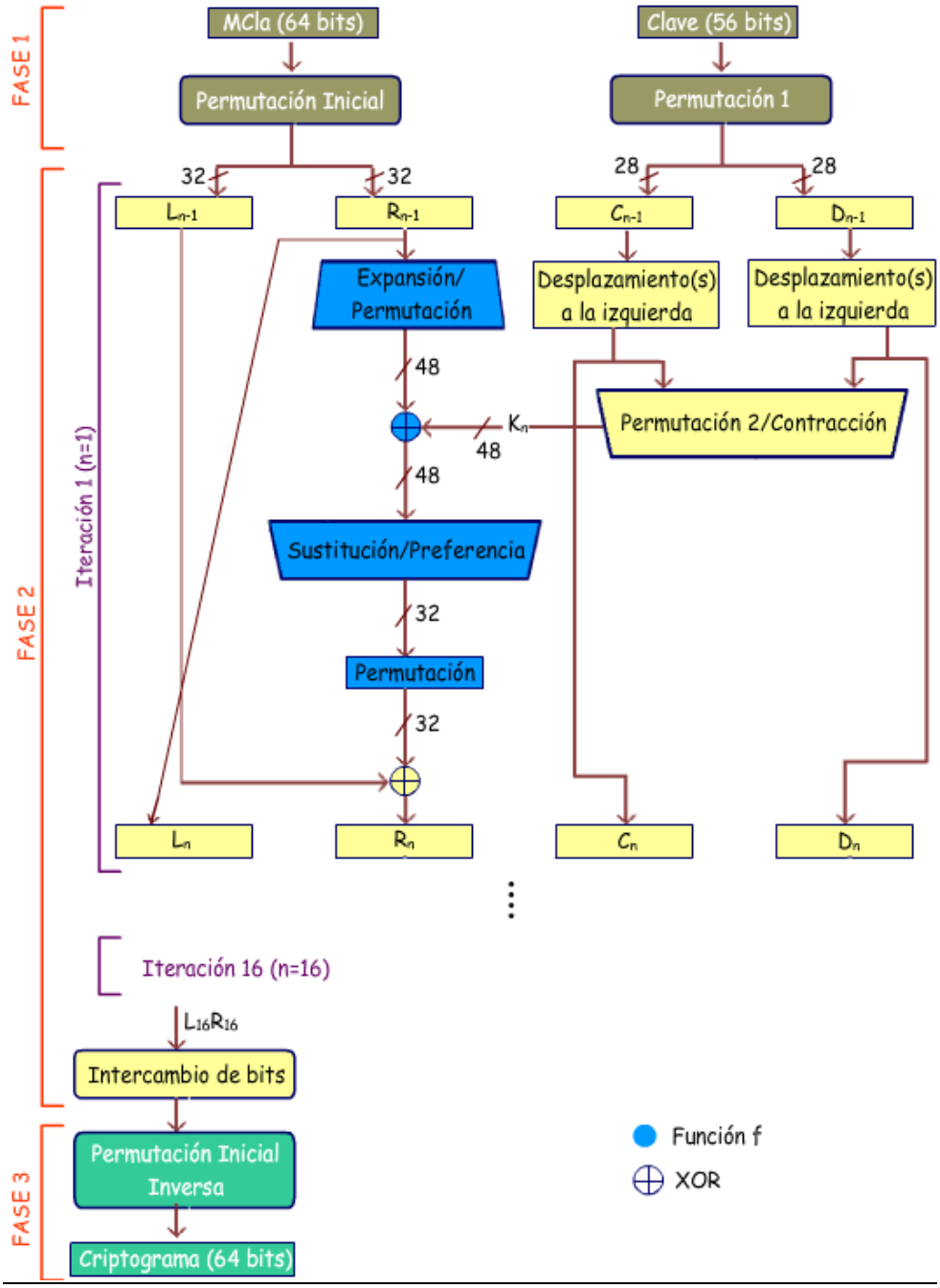


FIGURA 2.79 Algoritmo de cifrado DES

La figura 2.79 muestra el procedimiento para cifrar el texto en claro, puede notarse que dicho procedimiento consta de tres fases principales:

1. El texto original de 64 bits y la clave inicial pasan a través de una permutación.
2. Se realizan 16 iteraciones de la misma función, la salida de la iteración 16 contiene 64 bits los cuales son función del texto en claro y la clave, las mitades izquierda y derecha de dicha salida son intercambiadas para producir la presalida.

3. La presalida pasa a través de una permutación para producir el texto cifrado de 64 bits, dicha permutación es la inversa de la función de permutación inicial.

El algoritmo en cada una de las 16 iteraciones puede resumirse como sigue:

Los 64 bits correspondientes al mensaje que entran en cada una de las iteraciones son tratados como dos grupos de 32 bits, los primeros 32 y los últimos 32, en la figura 2.79 marcados como L(parte izquierda) y R(parte derecha). Las siguientes expresiones resumen el procedimiento en cada una de las iteraciones:

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f\left(R_{i-1}, K_i\right)$$

Los bits correspondientes a la clave que entran a cada una de las iteraciones se tratan como dos grupos de 28 bits cada uno, en la figura 2.79 marcados como C y D.

- En cada iteración C y D pasan por separado por un desplazamiento circular a la izquierda de 1 o 2 bits.
- Los bits resultantes de los desplazamientos circulares sirven como entrada de la siguiente iteración y como entrada a una función de permutación la cual produce una subclave K_i de 48 bits, dicha subclave corresponde a una de las entradas de la función $f\left(R_{i-1}, K_i\right)$, la cual se define en la figura 2.79.

ALGORITMO DE DESCIFRADO

El proceso de descifrado del DES es prácticamente el mismo que el de cifrado. En el descifrado el texto de entrada es el texto cifrado, debiéndose usar las subclaves en orden inverso, es decir en la primera iteración se debe usar la clave K_{16} y en la iteración 16 la K_1.

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

Para ilustrar a detalle el funcionamiento del algoritmo DES por medio de un ejemplo se realiza lo siguiente:

1. Partiendo de la clave original, se crean 16 subclaves, cada una de ellas con una longitud de 48 bits.
2. Se cifra el mensaje (bloque de 64 bits) utilizando las subclaves del paso anterior.

Para el ejemplo se emplea el siguiente mensaje en claro M=0123456789ABCDEF y la clave K=133457799BBCDFF1 ambos valores están en hexadecimal así que ambos se pasan a binario:

M= 0000000100100011010001010110011111000100110101011110011011110111
K= 0001001100110100010101110111100110011011101111001101111111110001

NOTA: Las tablas necesarias para cifrar con el algoritmo DES se encuentran en el anexo "Tablas del Algoritmo DES". Las operaciones tales como corrimientos, permutaciones, sustituciones utilizando cajas *S* y otras se explican en la sección 2.4.1 (Introducción a la Criptografía simétrica) del presente capítulo.

- **PASO 1. CREACIÓN DE LAS 16 SUBCLAVES**

1. Se reduce la clave original "K" a 56 bits. Para ello se realiza la permutación de K de acuerdo con la tabla PC-1, obteniéndose K+. Nótese que la tabla PC-1 sólo tiene 56 casillas razón por la cual sólo 56 bits de la clave original aparecen en la clave permutada K+.

K+= 11110000110011001010101011110101010101100110011110001111

2. Se divide K+ en dos bloques de 28 bits cada uno, los primeros 28 conforman el bloque 1 (C_0) y los últimos 28 el bloque 2 (D_0).

$C_{0=}$ 1111000 0110011 0010101 0101111
$D_{0=}$ 0101010 1011001 1001111 0001111

Con C_0 y D_0 definidos, se crean 16 bloques C_n y D_n 1<=n<=16. Cada par de bloques C_n y D_n se forman a partir del par previo C_{n-1} y D_{n-1}, utilizando la tabla de "desplazamientos izquierdos". Esto indica, por ejemplo que C_3 y D_3 se obtienen de C_2 y D_2, respectivamente, por dos desplazamientos a la izquierda, y C_{16} y D_{16} se obtienen de C_{15} y D_{15}, respectivamente, por un desplazamiento a la izquierda.

Siguiendo la tabla de "desplazamientos izquierdos" y usando el par C_0 y D_0 se obtienen los siguientes valores:

$C_{0=}$ 11110000110011001010101111
$D_{0=}$ 01010101011001100111100011111
$C_{1=}$ 11100001100110010101010111111
$D_{1=}$ 10101010110011001111000111110
$C_{2=}$ 11000011001100101010101111111
$D_{2=}$ 01010101100110011110000111101
$C_{3=}$ 00001100110010100101011111111
$D_{3=}$ 01010110011001111000111101010
$C_{4=}$ 00110011001010100101111111100
$D_{4=}$ 01011001100111100011110101010
$C_{5=}$ 11001100101010100111111110000
$D_{5=}$ 01100110011110000111101010101
$C_{6=}$ 00110010101010111111111000011

$D_{6=}$ 1001100111100011110101010101
$C_{7=}$ 1100101010101111111100001100
$D_{7=}$ 0110011110001111010101010110
$C_{8=}$ 0010101010111111110000110011
$D_{8=}$ 1001111000111101010101011001
$C_{9=}$ 0101010101111111100001100110
$D_{9=}$ 0011110001111010101010110011
$C_{10=}$ 0101010111111110000110011001
$D_{10=}$ 1111000111101010101011001100
$C_{11=}$ 0101011111111000011001100101
$D_{11=}$ 1100011110101010101100110011
$C_{12=}$ 0101111111100001100110010101
$D_{12=}$ 0001111010101010110011001111
$C_{13=}$ 0111111110000110011001010101
$D_{13=}$ 0111101010101011001100111100
$C_{14=}$ 1111111000011001100101010101
$D_{14=}$ 1110101010101100110011110001
$C_{15=}$ 1111100001100110010101010111
$D_{15=}$ 1010101010110011001111000111
$C_{16=}$ 1111000011001100101010101111
$D_{16=}$ 0101010101100110011110001111

3. Se forman las claves K_n, para 1<=n<=16, al aplicar la tabla de permutación PC-2 a cada uno de los pares. Cada par tiene 56 bits, pero PC-2 sólo utiliza 48.

Para obtener la primera clave se tiene:
C_1D_1= 11100001100110010101010111111010101011001100111100011110

Después de aplicar la permutación se obtiene:
K_1= 000110 110000 001011 101111 111111 000111 000001 110010

Las otras claves se obtienen de la misma forma, sus valores para el ejemplo son:

K_2= 011110 011010 111011 011001 110110 111100 100111 100101
K_3= 010101 011111 110010 001010 010000 101100 111110 011001
K_4= 011100 101010 110111 010110 110110 110011 010100 011101
K_5= 011111 001110 110000 000111 111010 110101 001110 101000
K_6= 011000 111010 010100 111110 010100 000111 101100 101111
K_7= 111011 001000 010010 110111 111101 100001 100010 111100
K_8= 111101 111000 101000 111010 110000 010011 101111 111011
K_9= 111000 001101 101111 101011 111011 011110 011110 000001
K_{10}= 101100 011111 001101 000111 101110 100100 011001 001111
K_{11}= 001000 010101 111111 010011 110111 101101 001110 000110
K_{12}= 011101 010111 000111 110101 100101 000110 011111 101001

K_{13}= 100101 111100 010111 010001 111110 101011 101001 000001
K_{14}= 010111 110100 001110 110111 111100 101110 011100 111010
K_{15}= 101111 111001 000110 001101 001111 010011 111100 001010
K_{16}= 110010 110011 110110 001011 000011 100001 011111 110101

- **PASO 2. CODIFICACIÓN DE CADA UNO DE LOS BLOQUES DE DATOS DE 64 BITS**

1. Se realiza una permutación inicial (IP) a los 64 bits del mensaje en claro M. Dicha permutación reacomoda los bits de acuerdo a la tabla denominada "tabla IP".

Para el ejemplo, tras aplicar la permutación inicial al bloque de texto M se obtiene:

M= 0000000100100011010001010110011111000100110101011110010110111101111

IP=1100110000000000110011001111111111110000101010101111000010101010

2. Se divide el bloque permutado IP en una mitad izquierda L_0 de 32 bits y una mitad derecha R_0 de 32 bits.

L_0= 1100 1100 0000 0000 1100 1100 1111 1111
R_0= 1111 0000 1010 1010 1111 0000 1010 1010

Se procede a realizar las 16 iteraciones, para 1<=n<=16, utilizando una función *f*, la cual opera sobre dos bloques (uno de datos de 32 bits y otro de la subclave K_n de 48 bits). La salida de la función *f* es de 32 bits.

Para 1<=n<=16 se calcula:

$$L_n = R_{n-1}$$
$$R_n = L_{n-1} \oplus f\left(R_{n-1}, K_n\right)$$

En cada iteración se toman los 32 bits derechos del resultado previo y se convierten en los 32 bits izquierdos del paso actual. Para obtener los 32 bits derechos del paso actual, se realiza el XOR de los 32 bits izquierdos del resultado previo con el cálculo de la función *f*.

Para calcular *f* se realiza lo siguiente:

a) Se expande cada bloque R_{n-1} de 32 bits a 48 bits utilizando la tabla llamada "tabla E de selección de bit". Los primeros tres bits de $E(R_{n-1})$ son los bits en las posiciones 32, 1 y 2 de R_{n-1}, mientras que los últimos dos bits de $E(R_{n-1})$ son los bits en las posiciones 32 y 1 de R_{n-1}.
b) Se realiza XOR de la clave K_n con la salida de $E(R_{n-1})$:

$$K_n \oplus E(R_{n-1})$$

c) Se escribe el resultado previo, el cual es de 48 bits en grupos de 6 bits:

$$K_n \oplus E(R_{n-1}) = B_1B_2B_3B_4B_5B_6B_7B_8$$

Donde cada B_i es un grupo de 6 bits.

Ahora se calcula:

$$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$$

Donde $S_i(B_i)$ se refiere a la salida de la *i*-ésima caja *S*

En cada una de las funciones S_1, S_2,..., S_8, se toma un bloque de 6 bits como entradas y arroja un bloque de 4 bits como salida.

d) Se realiza la permutación P definida en la tabla llamada "tabla S" a la salida del paso anterior. Con esto se termina con el cálculo de la función *f*.

$$f = P(S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8))$$

Para n=1, se tiene:

L_0= 1100 1100 0000 0000 1100 1100 1111 1111
R_0= 1111 0000 1010 1010 1111 0000 1010 1010
K_1= 000110 110000 001011 101111 111111 000111 000001 110010

L_1=R_0= 1111 0000 1010 1010 1111 0000 1010 1010
R_1= $L_0 \oplus f(R_0, K_1)$

Calculando la function *f*:

a) Expanción de R_0:

E(R_0)= 011110 100001 010101 010101 011110 100001 010101 010101

b) Cálculo de XOR de la clave K_1 con E(R_0).

$K_1 \oplus E(R_0) =$ 011000 010001 011110 111010 100001 100110 010100 100111

c) Calculo de $S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8)$.

B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
011000	010001	011110	111010	100001	100110	010100	100111

$S_1(B_1)S_2(B_2)S_3(B_3)S_4(B_4)S_5(B_5)S_6(B_6)S_7(B_7)S_8(B_8) =$

0101 1100 1000 0010 1011 0101 1001 0111

d) Aplicación de la permutación P al resultado del paso anterior:

f= 0010 0011 0100 1010 1010 1001 1011 1011

Ahora que ya tenemos el resultado de la función *f*, podemos obtener finalmente
$R_1 = L_0 \oplus f(R_0, K_1)$
R_1= 1100 1100 0000 0000 1100 1100 1111 1111 + 0010 0011 0100 1010 1010 1001 1011 1011
R_1= 1110 1111 0100 1010 0110 0101 0100 0100

3. Al final de la iteración 16 se tienen los bloques L_{16} y R_{16}. Entonces se invierte el orden de los dos bloques dentro del bloque de 64 bits $R_{16}L_{16}$ y se aplica la permutación final IP^{-1} como está definida en la tabla llamada "tabla IP^{-1}".

L_{16}= 0100 0011 0100 0010 0011 0010 0011 0100
R_{16}= 0000 1010 0100 1100 1101 1001 1001 0101

$R_{16}L_{16}$=0000101001001100110110011001010101000011010000100011001000110100

IP^{-1}= 1000 0101 1110 1000 0001 0011 0101 0100 0000 1111 0000 1010 1011 0100 0000 0101

Convertimos IP^{-1} a hexadecimal y obtenemos el criptograma C:

C= 85E813540F0AB405

2.4.3 AES (Advanced Encryption Standard)

ORÍGENES

- En 1997, el NIST convocó al desarrollo de un nuevo algoritmo que fuese la base de un nuevo estándar, los requerimientos dados fueron los siguientes:
  1. Algoritmo de cifrado simétrico
  2. Algoritmo de cifrado en bloques
  3. Manejo de bloques de 128 bits
  4. Soporte de manejo de claves de diferente longitud
  5. Claves de 128, 192 y 256 bits.

- Publicado por el NIST en 2001, con la finalidad de sustituir al DES.

ALGORITMO DE CIFRADO

El algoritmo de cifrado AES hace uso de matemáticas polinomiales en estructuras de campos finitos, en particular opera en el Campo de Galois $GF(2^8)$ Los campos finitos permiten manejar cada elemento del campo con una cantidad determinada de memoria, además siempre que se realiza una operación se tendrá una operación

inversa bien definida, por lo tanto las operaciones son bidireccionales permitiendo de este modo los procesos de cifrado y descifrado. La razón por la que AES opera en el GF(2^8) es que hace posible su implementación en varias plataformas debido a que los coeficientes están en el rango de 0 a 7, considerando así el sistema binario y a un byte como la palabra básica del algoritmo.

AES opera sobre bloques de datos de 128 bits y la clave que utiliza puede ser de 128, 192 o 256 bits. La figura 2.80 muestra el esquema del cifrado AES.

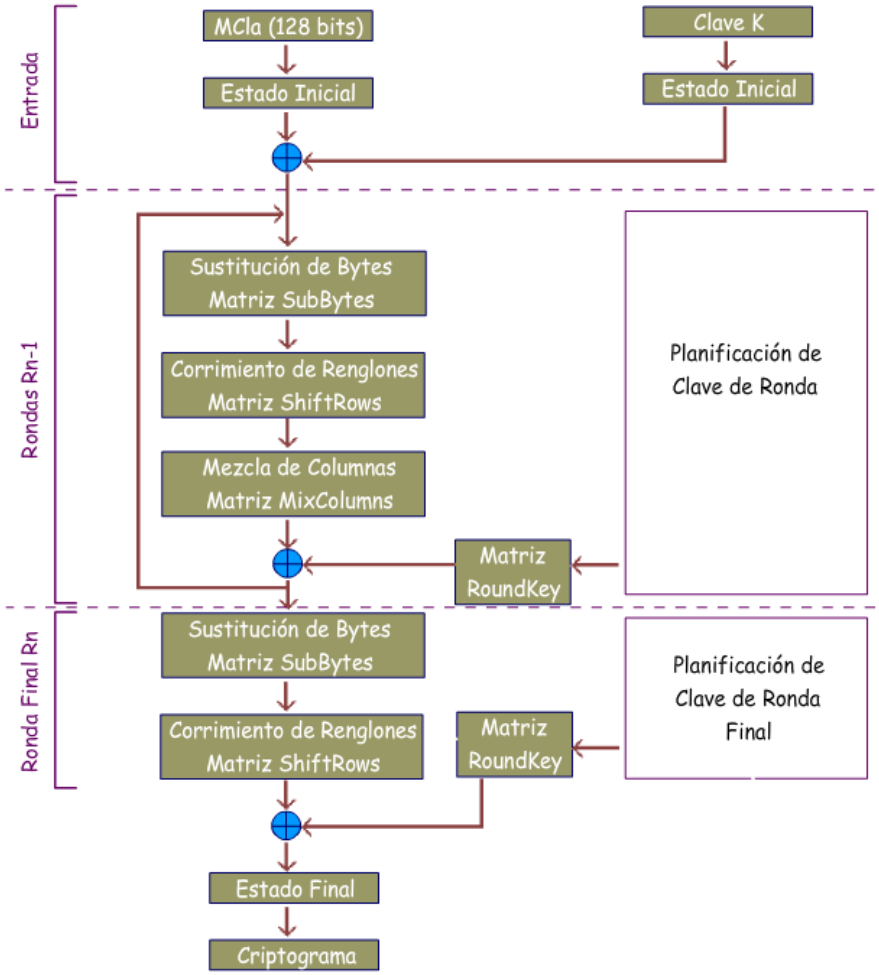


FIGURA 2.80 Algoritmo de cifrado AES

Dependiendo del tamaño de la clave que se emplee, AES realiza un número fijo de rondas tal y como se muestra en la figura 2.81.

	Longitud de clave Nk (palabras de 32 bits)	Longitud de bloque de datos Nb (palabras de 32 bits)	Número de rondas
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

FIGURA 2.81 Número de rondas de AES

Cada una de las rondas se compone de cinco matrices, la primera es la matriz de entrada o inicio de ronda y las cuatro restantes son transformaciones o funciones bien definidas:

1. **Matriz de entrada o inicio de ronda**: para la entrada su contenido corresponde al MCla, para cada una de las rondas es necesario calcular el texto de entrada.
2. **Matriz SubBytes**: sustituye individualmente cada byte del estado por otro de acuerdo a una tabla fija.
3. **Matriz ShiftRows**: toma cada renglón del estado completo y hace un corrimiento cíclico un determinado número de bytes o columnas que depende del renglón del que se trate.
4. **Matriz MixColums**: opera idénticamente con cada columna completa (4 bytes) aplicando una transformación lineal.
5. **Matriz Round Key**: modifica el estado de la clave sumándole módulo 2 (XOR) byte a byte la clave de la ronda correspondiente.

En la última ronda se omite el cálculo de MixColumns.

Independientemente del tamaño de K siempre se tienen las siguientes características:

- Matrices de 4x4
- Cada elemento de la matriz es de dos dígitos hexadecimales
- MCla siempre será procesado en bloques de 128 bits, manejados en las matrices en hexadecimal.

ALGORITMO DE DESCIFRADO

Las transformaciones utilizadas en el descifrado son:

1. **Matriz Cripto**: para la entrada su contenido corresponde al criptograma, para cada una de las rondas es necesario calcularla de la siguiente manera:

   Ronda 1: Cripto Original $\oplus$ RoundKey
   Ronda 2 a Ronda n: InvMixColumns(InvSubBytes $\oplus$ RoundKey)

   Utilizando la matriz de multiplicación fija inversa (figura 2.82):

0E	0B	0D	09
09	0E	0B	0D
0D	09	0E	0B
0B	0D	09	0E

FIGURA 2.82 Matriz de multiplicación fija inversa

2. **Matriz InvShiftRows (ISR)**: se sigue el mismo procedimiento que en el cifrado con la excepción de que los corrimientos son a la derecha.
3. **Matriz InvSubBytes (ISB)**: se sigue el mismo procedimiento que en el cifrado sólo que se utiliza la tabla Inverse S-box.
4. **Matriz RoundKey (RK)**: se obtiene de igual manera que en el cifrado, sólo que se utilizan en orden inverso.

5. **Matriz $ISB \oplus$ RK**: el resultado en la matriz $ISB \oplus RK$ de la última ronda es el MCIa.

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

A continuación se realiza un ejemplo para ilustrar a detalle el funcionamiento del algoritmo AES.

Considerando AES-128 se tiene:

- Tamaño de la clave de 128 bits.
- Tamaño del bloque de datos de 128 bits.
- Se realizan 10 iteraciones o rondas.

MCIa= 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34
K= 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

NOTA: Las tablas necesarias para cifrar con el algoritmo AES se encuentran en el anexo “Tablas del Algoritmo AES”. Las operaciones tales como XOR, corrimientos y otras se explican en la sección 2.4.1 (Introducción a la Criptografía simétrica) del presente capítulo.

A continuación se muestra el resultado de las diez rondas para el ejemplo:

Ronda	Matriz de inicio de ronda				Matriz SubBytes				Matriz ShiftRows				Matriz MixColumns				Matriz RoundKey			
Entrada	32	88	31	e0													2b	28	ab	09
	43	5a	31	37													7e	ae	f7	cf
	f6	30	98	07													15	d2	15	4f
	a8	8d	a2	34													16	a6	88	3c
R1	19	a0	9a	e9	d4	e0	b8	1e	da	e0	b8	1e	04	e0	48	28	a0	88	23	2a
	3d	f4	c6	f8	27	bf	b4	41	bf	b4	41	27	66	cb	f8	06	fa	54	a3	6c
	e3	e2	8d	48	11	98	5d	52	5d	52	11	98	81	19	d3	26	fe	2c	39	76
	be	2b	2a	08	ae	f1	e5	30	30	ae	f1	e5	35	9a	7a	4c	17	b1	39	05
R2	a4	68	6b	02	49	45	7f	77	49	45	7f	77	58	1b	db	1b	f2	7a	59	73
	9c	9f	5b	6a	de	db	39	02	db	39	02	de	4d	4b	e7	6b	c2	96	35	59
	7f	35	ea	50	d2	96	87	53	87	53	d2	96	ca	5a	ca	b0	95	b9	80	f6
	f2	2b	43	49	89	f1	1a	3b	3b	89	f1	1a	f1	ac	a8	e5	f2	43	7a	7f
R3	aa	61	82	68	ac	ef	13	45	ac	ef	13	45	75	20	53	bb	3d	47	1e	6d
	8f	dd	d2	32	73	c1	b5	23	c1	b5	23	73	ec	0b	c0	25	80	16	23	7a
	5f	e3	4a	46	cf	11	d6	5a	d6	5a	cf	11	09	63	cf	d0	47	fe	7e	88
	03	ef	d2	9a	7b	df	b5	b8	7b	7b	df	b5	93	33	7c	dc	7d	3e	44	3b
R4	48	67	4d	d6	52	85	e3	f6	52	85	e3	f6	0f	60	6f	5e	ef	a8	b6	db
	6c	1d	e3	5f	50	a4	11	cf	a4	11	cf	50	d6	31	c0	b3	44	52	71	0b
	4e	9d	b1	58	2f	5e	c8	6a	c8	6a	2f	5e	da	38	10	13	a5	5b	25	ad
	ee	0d	38	e7	28	d7	07	9a	94	28	d7	07	a9	bf	6b	01	41	7f	3b	00
R5	e0	c8	d9	85	e1	e8	35	97	e1	e8	35	97	25	bd	b6	4c	d4	7c	ca	11
	92	63	b1	b8	4f	fb	c8	6c	fb	c8	6c	4f	d1	11	3a	4c	d1	83	f2	f9
	7f	63	35	be	d2	fb	96	ae	96	ae	d2	fb	a9	d1	33	c0	c6	9d	b8	15
	e8	c0	50	01	9b	ba	53	7c	7c	9b	ba	53	ad	68	8e	b0	f8	87	bc	bc
R6	f1	c1	7c	5d	a1	78	10	4c	a1	78	10	4c	4b	2c	33	37	6d	11	db	ca
	00	92	c8	b5	63	4f	e8	d5	4f	e8	d5	63	86	4a	9d	d2	88	0b	f9	00
	6f	4c	8b	d5	a8	29	3d	03	3d	03	a8	29	8d	89	f4	18	a3	3e	86	93
	55	ef	32	0c	fc	df	23	fe	fe	fc	df	23	6d	80	e8	d8	7a	fd	41	fd
R7	26	3d	e8	fd	f7	27	9b	54	f7	27	9b	54	14	46	27	34	4e	5f	84	4e
	0e	41	64	d2	ab	83	43	b5	83	43	b5	ab	15	16	46	2a	54	5f	a6	a6
	2e	b7	72	8b	31	a9	40	3d	40	3d	31	a9	b5	15	56	d8	f7	c9	4f	dc
	17	7d	a9	25	f0	ff	d3	3f	3f	f0	ff	d3	bf	ec	d7	43	0e	f3	b2	4f
R8	5a	19	a3	7a	be	d4	0a	da	be	d4	0a	da	00	b1	54	fa	ea	b5	31	7f
	41	49	e0	8c	83	3b	e1	64	3b	e1	64	83	51	c8	76	1b	d2	8d	2b	8d
	42	dc	19	04	2c	86	d4	f2	d4	f2	2c	86	2f	89	6d	99	73	ba	f5	29
	b1	1f	65	0c	c8	c0	4d	fe	fe	c8	c0	4d	d1	ff	cd	ea	21	d2	60	2f
R9	ea	04	65	85	87	f2	4d	97	87	f2	4d	97	47	40	a3	4c	ac	19	28	57
	83	45	5d	96	ec	6e	90	ec	6e	4c	90	ec	37	d4	70	9f	77	fa	d1	5c
	5c	33	98	b0	4a	c3	4a	c3	46	e7	4a	c3	94	e4	3a	42	66	dc	29	00
	f0	2d	ad	c5	8c	d8	d8	95	a6	8c	d8	95	ed	a5	a6	bc	f3	21	41	6e
R10	eb	59	8b	1b	e9	cb	3d	af	e9	cb	3d	af					d0	c9	e1	b6
	40	2e	a1	c3	09	31	32	2e	31	32	2e	09					14	ee	3f	63
	f2	38	13	42	89	07	7d	2c	7d	2c	89	07					f9	25	0c	0c
	1e	84	e7	d2	72	5f	94	b5	b5	72	5f	94					a8	89	c8	a6

Entrada:

- En la matriz inicio de ronda se coloca por columnas los dígitos hexadecimales del MCla. Nótese que en cada celda se ponen dos dígitos.
- En la matriz RoundKey se coloca por columnas los dígitos hexadecimales de la clave. Nótese que en cada celda se ponen dos dígitos.

R_1:

- Sólo para obtener la matriz inicio de la ronda R_1 se calcula $MatrizInicio \oplus RoundKey$ elemento a elemento, tomando ambas matrices de la entrada.

Por ejemplo, para la celda [0][0] se obtiene el resultado mostrado a continuación:

	Hexadecimal	Binario
MatrizInicio de la Entrada Celda [0][0]	32	0011 0010
RoundKey Celda [0][0]	2b	0010 1011
MatrizInicio R_1 Celda [0][0]	19	0001 1001

- La matriz SubBytes se obtiene a partir de la matriz de inicio de la ronda actual. Se realizan sustituciones utilizando la tabla llamada “S-box” de la siguiente manera:

Por ejemplo, para la celda [0][0] se obtiene:

Matriz Inicio R_1 Celda [0][0]	1 9 | | x y
x= 1 , y=9 en S-box	↓
Matriz SubBytes R_1 Celda [0][0]	d 4

- La matriz ShiftRows se obtiene realizando corrimientos circulares izquierdos a la matriz SubBytes de la siguiente manera:

  Renglón 1: no se realizan corrimietos.
  Renglón 2: se realiza un corrimiento circular izquierdo.
  Renglón 3: se realizan dos corrimientos circulares izquierdos.
  Renglón 4: se realizan tres corrimientos circulares izquierdos.

- La matriz MixColumns se obtiene a partir de la matriz ShiftRows de la ronda actual, se deben calcular los valores de b1–b16 de la matriz mostrada en la figura 2.83:

b1	b5	b9	b13
b2	b6	b10	b14
b3	b7	b11	b15
b4	b8	b12	b16

FIGURA 2.83 Matriz MixColumns

Para obtener los valores de b, cada columna de la matriz anterior se multiplica por la matriz de multiplicación (la cual se muestra en la figura 2.84) de la siguiente manera:

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

FIGURA 2.84 Matriz de multiplicación

b1= (b1 * 2) ⊕ (b2 * 3) ⊕ (b3 * 1) ⊕ (b4 * 1)
b2= (b1 * 1) ⊕ (b2 * 2) ⊕ (b3 * 3) ⊕ (b4 * 1)
b3= (b1 * 1) ⊕ (b2 * 1) ⊕ (b3 * 2) ⊕ (b4 * 3)
b4= (b1 * 3) ⊕ (b2 * 1) ⊕ (b3 * 1) ⊕ (b4 * 2)

b5= (b5 * 2) ⊕ (b6 * 3) ⊕ (b7 * 1) ⊕ (b8 * 1)
b6= (b5 * 1) ⊕ (b6 * 2) ⊕ (b7 * 3) ⊕ (b8 * 1)
b7= (b5 * 1) ⊕ (b6 * 1) ⊕ (b7 * 2) ⊕ (b8 * 3)
b8= (b5 * 3) ⊕ (b6 * 1) ⊕ (b7 * 1) ⊕ (b8 * 2)
b9= (b9 * 2) ⊕ (b10 * 3) ⊕ (b11 * 1) ⊕ (b12 * 1)
b10= (b9 * 1) ⊕ (b10 * 2) ⊕ (b11 * 3) ⊕ (b12 * 1)
b11= (b9 * 1) ⊕ (b10 * 1) ⊕ (b11 * 2) ⊕ (b12 * 3)
b12= (b9 * 3) ⊕ (b10 * 1) ⊕ (b11 * 1) ⊕ (b12 * 2)

b13= (b13 * 2) ⊕ (b14 * 3) ⊕ (b15 * 1) ⊕ (b16 * 1)
b14= (b13 * 1) ⊕ (b14 * 2) ⊕ (b15 * 3) ⊕ (b16 * 1)
b15= (b13 * 1) ⊕ (b14 * 1) ⊕ (b15 * 2) ⊕ (b16 * 3)
b16= (b13 * 3) ⊕ (b14 * 1) ⊕ (b15 * 1) ⊕ (b16 * 2)

Por ejemplo para obtener la celda [0][0] de la matriz MixColumns de la ronda 1 tenemos que calcular b1:

b1= (b1 * 02) ⊕ (b2 * 03) ⊕ (b3 * 01) ⊕ (b4 * 01)
b1= (d4 * 02) ⊕ (bf * 03) ⊕ (5d * 01) ⊕ (30 * 01)

Para resolver la operación "*"primero tenemos que sustituir cada uno de los operadores por sus valores correspondientes en la tabla llamada "tabla L"; si un número está multiplicado por uno, el resultado es el mismo número.

b1= (d4 * 02) ⊕ (bf * 03) ⊕ 5d ⊕ 30

Los resultados de la tabla L son los mostrados en la figura 2.85:
Nótese que b3 y b4 se quedan igual debido a que están multiplicados por uno.

	b1	02	b2	03
Valores en la matriz ShiftRows R1	d 4 ↓ ↓ x y	0 2 ↓ ↓ x y	b f ↓ ↓ x y	0 3 ↓ ↓ x y
x,y en la tabla L	↓	↓	↓	↓
Resultado	41	19	9d	01

FIGURA 2.85 Resultados de tabla L

Se debe realizar la suma hexadecimal de los operadores una vez sustituidos por sus valores correspondientes en la tabla L, el resultado de la suma debe ser sustituido por su valor correspondiente en la tabla llamada "tabla E", ese será el resultado final. Si la suma hexadecimal diera como resultado un número mayor a FF, a ese número hay que restarle FF tantas veces hasta que el resultado de menor o igual a FF.

Continuando con el ejemplo, realizando las operaciones se tienen los siguientes resultados (figura 2.86):

	41+19	9d+01
Resultado de la suma hexadecimal	5 a ↓ ↓ x y	9 e ↓ ↓ x y
x,y en la tabla E	↓	↓
Resultado	b3	da

FIGURA 2.86 Resultados de tabla E

b1= b3 ⊕ da ⊕ 5d ⊕ 30
b1= 04

- Para calcular la matriz RoundKey se divide sólo para la primera ronda la clave original en N_k bloques de acuerdo a la longitud de clave que se esté empleando

(ver figura 2.87). En este caso estamos empleando una clave de 128 bits, por lo que la clave se dividirá en 4 bloques:

Longitud de clave (bits)	N_k
128	4
192	6
256	8

FIGURA 2.87 Bloques N_k

w_0= 2b 7e 15 16
w_1= 28 ae d2 a6
w_2= ab f7 15 88
w_3= 09 cf 4f 3c

Se realiza la planificación de clave de ronda:

w_i	temp w_{i-1}	Rotword	Subword	Rcon [1 / N_k]	Subword ⊕ Rcon	$w_{[i-Nk]}$	w_i = temp ⊕ $w_{[i-Nk]}$
w_4	09cf4f3c	cf4f3c09	8a84eb01	01000000	8b84eb0	2b7e1516	a0fafe17
w_5	a0fafe17	-	-	-	-	28aed2a6	88542cb1
w_6	88542cb1	-	-	-	-	abf71588	23a33939
w_7	23a33939	-	-	-	-	09cf4f3c	2a6c7605

Columna Rotword: se realiza un corrimiento a la izquierda de temp, el corrimiento es de una celda, es decir de dos dígitos hexadecimales.

Columna Subword: se toman los datos de Rotword y para cada celda se sustituye considerando la tabla S-box.

Columna Rcon: los valores de esta columna ya están calculados y son fijos (ver figura 2.88):

Columna w_i: los valores obtenidos en esta columna son los valores de la matriz RoundKey, se colocan por columnas y en cada casilla se ponen dos dígitos hexadecimales.

w_4	01000000
w_8	02000000
w_{12}	04000000
w_{16}	08000000
w_{20}	10000000
w_{24}	20000000
w_{28}	40000000
w_{32}	80000000
w_{36}	1b000000
w_{40}	36000000

FIGURA 2.88 Valores para Rcon

R_2 – R_9:

- Para obtener la matriz inicio de ronda se calcula $MixColumns \oplus RoundKey$ elemento a elemento, tomando ambas matrices de la ronda anterior.
- Las otras cuatro matrices se obtienen de la misma forma que en R_1.

R_{10}:

- Se realiza el mismo procedimiento que en la ronda anterior solo que no se obtiene MixColumns.

El criptograma se obtiene a partir de R_{10}, se realiza: $ShiftRows \oplus RoundKey$

39	02	dc	19
25	dc	11	6a
84	09	85	0b
1d	fb	97	32

Cripto: 3925841d02dc09fbdc118597196a0b32

2.5 CRIPTOGRAFÍA ASIMÉTRICA O DE CLAVE PÚBLICA

2.5.1 INTRODUCCIÓN A LA CRIPTOGRAFÍA ASIMÉTRICA

CARACTERÍSTICAS DE LOS ALGORITMOS ASIMÉTRICOS

- Se utiliza una clave para cifrar y otra para descifrar. El emisor emplea la clave pública del receptor para cifrar el mensaje, éste último lo descifra con su clave privada.

- Se basan en operaciones matemáticas complejas.
- Se ejecutan de 100 a 1000 veces más lento que los algoritmos simétricos.

HERRAMIENTAS MATEMÁTICAS

- **TEOREMA DE EUCLIDES**

Si un número primo divide a un producto, divide a uno de los factores.

- **TEOREMA FUNDAMENTAL DE LA ARITMÉTICA**

Todo número entero positivo se puede escribir de forma única (salvo en el orden de los factores) como producto de números primos.

Definiciones:

- **Máximo común divisor de dos números enteros *a* y *b*** $(mcd(a,b))$**:** mayor número entero que divide a *a* y a *b*.
- **Mínimo común múltiplo de dos números enteros a y *b*** $(mcm(a,b))$**:** es el menor entero positivo divisible por *a* y por *b*.
- **Dos enteros *a* y *b* son primos entre sí si** $mcd(a,b)=1$.

- **TEOREMA DE LA DIVISIÓN DE EUCLIDES**

Dados dos números enteros $a>b>0$ se verifica: $mcd(a,b)=mcd(b,r)$, esto es: $a=b\cdot q+r, b>r$, siendo *q* el cociente de la división de *a* entre *b* y *r* el residuo.

- **ALGORITMO DE EUCLIDES**

Permite calcular el máximo común divisor de dos números enteros.

El algoritmo de Euclides consiste en repetir de forma reiterada la propiedad del teorema de la división de Euclides.

Ejemplo:

Obtención del $mcd(1313,2017)$, empleando el algoritmo de Euclides:

a=2017 y *b*= 1313, *q*= cociente de *a*/*b* y *r*= residuo.

a	=	*b*	*	*q*	+	*r*	
2017	=	1313	*	1	+	704	
1313	=	704	*	1	+	609	
704	=	609	*	1	+	95	
609	=	95	*	6	+	39	
95	=	39	*	2	+	17	
39	=	17	*	2	+	5	
17	=	5	*	3	+	2	
5	=	2	*	2	+	1	→ **mcd**
2	=	1	*	2	+	0	

- **ALGORITMO DE EUCLIDES EXTENDIDO**

Si $mcd(a,b)=d$, con $a>b$, entonces existen enteros *u* y *v* tales que $d=u\cdot a+v\cdot b$; es decir, el *mcd* de dos números se puede expresar como la combinación lineal de esos dos números con coeficientes enteros.

El Algoritmo de Euclides Extendido permite determinar los valores de *u* y *v* de la igualdad anterior y es una aplicación directa del Algoritmo de Euclides sólo hay que ir despejando de la última división obtenida hasta llegar a la primera.

El algoritmo también permite calcular el inverso de un número.

- **EL ANILLO DE LOS NÚMEROS ENTEROS MÓDULO m**

Considerando el conjunto de los números enteros: $Z=\{...,-2,-1,0,1,2,...\}$

Definiciones:

- Sea *m* un número entero positivo. Dos números $a,b\in Z$ son congruentes módulo $m(a\equiv b(\mathrm{mod}\,m))$ si su diferencia es un múltiplo de *m*, es decir si $a-b=k\cdot m$
- Se llama clase de equivalencia definida por un número *a* módulo *m* y denotada por $([a])$ al conjunto de los números enteros que son congruentes con *a* módulo *m*, es decir:

$$[a]=\{n\in Z;n\equiv a(\mathrm{mod}\,m)\}$$

- El conjunto de clases de equivalencia se denota por Z_m, y se dice que es el conjunto de los números enteros módulo *m*.

  Por ejemplo, si $m=6$, entonces $[4]=\{...8,-2,4,10,16,...\}=\{4+6k;k\in Z\}$.

  En la práctica, $[a]$ se identifica con el residuo de la división de *a* entre *m*; por ejemplo, la clase de 14 módulo 6 se identifica con 2. De este modo se escribe que:

$$Z_m=\{0,1,2,...,m-1\}$$

- Las clases de equivalencia se pueden sumar y multiplicar, sólo con definir dichas operaciones como la suma y el producto de dos cualesquiera de sus elementos:

$$[a]+[b]=[a+b],[a]\cdot[b]=[a\cdot b]$$

  De este modo, Z_m se convierte en un anillo.

- **TEOREMA DE EULER**

Considerando que *N* es un número entero, la función de $Euler\phi(N)$ es el número de valores enteros positivos menores que *N* y relativamente primos con *N*.

Ahora que ya conocemos la función de *Euler* se puede comprender el teorema de *Euler* en donde *N* y *a* son valores enteros con *a* relativamente primo con *N*:

$$a^{\phi(N)} \equiv 1 \pmod{N}$$

- **TEOREMA DE FERMAT**

Considerando que *p* es un número primo y *a* un valor entero relativamente primo con *p*, entonces:

$$a^{p-1} \equiv 1 \pmod{p}$$

Es un caso particular del teorema de *Euler*, ya que para un número primo *p* la función de $Euler \phi(p)$ está dada por:

$$\phi(p) = p - 1$$

Corolarios:

1. Si *p* es primo, para cualquier valor entero *a* se verifica:

$$a^{p-1} a \equiv a \pmod{p}$$

2. Si *p* es primo, para cualesquiera valores enteros (*k*,*a*) se verifica:

$$a^{k(p-1)} a \equiv a \pmod{p}$$

3. Si *p* es primo, para cualesquiera valores enteros (*a*,*b*) se verifica:

$$a^{b} \equiv a^{b[\bmod(p-1)]} \pmod{p}$$

- **LOGARITMO DISCRETO**

La exponenciación modular dada por la ecuación:

$$y \equiv g^{x} \pmod{p}$$

es una función típica de una sola dirección. Con *g* y *x* enteros y *p* un número primo grande, el cálculo de la función *y* es posible, lo contrario sucede con el cálculo de su función inversa la cual es:

$$x \equiv \log_{g} y \pmod{p}$$

Esta función es denominada logaritmo discreto y su cálculo resulta inviable con los conocimientos matemáticos actuales ya que tiene una complejidad exponencial dada por $e^{\sqrt{\ln(p)\ln\ln(p)}}$.

5.1.3 PRINCIPALES ALGORITMOS ASIMÉTRICOS

- **DIFFIE-HELLMAN**

Se trata de un algoritmo de acuerdo de claves que permite a dos usuarios intercambiar una clave secreta a través de un medio inseguro. En otras palabras, los dos usuarios son capaces de acordar una clave, aún cuando los intercambios previos al acuerdo sean públicos. Este algoritmo se explica con más detalle en la sección 2.5.2 (Diffie-Hellman) del presente capítulo.

- **EL GAMAL**

Algoritmo utilizado en GNU Privacy Guard y PGP creado por Taher El Gamal en 1985, se trata de un algoritmo basado en el trabajo desarrollado por Diffie-Hellman cuyos parámetros son un número primo grande *p* y un entero *g* que es generador del grupo multiplicativo Z_p. Ambos valores son públicos. El emisor elige aleatoriamente su clave secreta *x* de tal modo que cumpla: $1 < x < p-1$, su clave pública *y* está dada por la ecuación:

$$y \equiv g^x \pmod{p}$$

Para cifrar un MCla *M* tal que: $1 < M < p$, se debe elegir un valor aleatorio *k* que sea relativamente primo con $(p-1)$ y cumpla con: $1 < k < p-1$. El cifrado está conformado por la pareja de valores enteros (r, s)definidos por las congruencias:

$$r \equiv g^k \pmod{p}$$
$$s \equiv My^k \pmod{p}$$

La recuperación del mensaje en claro *M* se obtiene mediante el cómputo de la siguiente congruencia:

$$M \equiv \frac{s}{r^x} \pmod{p}$$

Este algoritmo se explica con más detalle en la sección 2.5.3 (El Gamal) del presente capítulo.

- **RSA (Rivest Shamir Adleman)**

Fue desarrollado en el MIT en 1977, es el algoritmo de clave pública más popular en la actualidad utilizado tanto para cifrar texto como para generar firmas digitales. Multiplicando dos números primos, genera un número llamado módulo público el cual es utilizado para conseguir las claves pública y privada, la idea es que los números primos escogidos sean muy grandes ya que factorizar el resultado de multiplicar dos números primos es un problema computacionalmente imposible. Este algoritmo se explica a detalle en la sección 2.5.4 (RSA) del presente capítulo.

- **DSA (Digital Signature Algorithm)**

En 1991 el NIST propuso un estándar para firma digital (Digital Signature Standard, DSS), siendo su algoritmo el DSA, en 1994 este algoritmo fue anunciado formalmente como estándar y debe ser utilizado únicamente para generar firmas digitales. El algoritmo se apoya en el uso de funciones hash y está documentado en el FIPS 186.

- **CURVAS ELÍPTICAS**

En 1985, Neal Koblitz y Victor Miller aplicaron la teoría de las curvas elípticas a la Criptografía. Las ventajas que proporciona el uso de curvas elípticas en Criptografía es que cuando estas son definidas sobre campos finitos proporcionan grupos finitos abelianos, en donde los cálculos se efectúan con la eficiencia requerida en la Criptografía, además el cálculo de logaritmos resulta ser más difícil que en los campos finitos numéricos. Las curvas elípticas empleadas en Criptografía se explican a detalle en la sección 2.5.5 del presente capítulo.

- **CRIPTOGRAFÍA CUÁNTICA** [8]

Como se ha visto, en el caso de la Criptografía simétrica aun cuando los algoritmos sean robustos su problema radica en la distribución segura de las claves a utilizar, en tanto que en la Criptografía asimétrica aparentemente ese problema no existe, sin embargo, muchos de los sistemas de clave pública se apoyan fuertemente en el uso de entidades certificadoras, siendo éstas las que se encargan de calcular las claves correspondiente para los usuarios lo que significa que dichas entidades tienen son las que dan a conocer las claves públicas y deben repartir las claves privadas de los usuarios, distribución que debe ser hecha de manera segura y que de alguna forma este problema se puede resolver mediante Diffie-Hellman.

En este contexto, la Criptografía cuántica pretende dar una solución a dicha problemática de manera más rápida y eficiente que lo hecho por Diffie y Hellman. Para lo cual veamos en términos generales cómo opera. Suponiendo que un emisor desea enviar cierta información de manera segura, ésta será enviada en su forma binaria, esto es, en su representación de unos y ceros, para lo cual en Criptografía cuántica existen dos formas de enviar un cero e igual número de hacerlo para un uno binarios que obedecen a la polarización de los fotones que se van a transmitir, como se aprecia en la figura 2.89.

Polarización rectilínea		Polarización diagonal	
0°	90°	45°	135°
--	|	/	\
"1"	"0"	"0"	"1"

FIGURA 2.89 Polarización de fotones para el envío de bits

Por ejemplo una secuencia 00110100110 podría ser generada como //----|\/|\--|, ahora bien, la forma en que el receptor interprete esta serie de bits va a depender de la colocación de un filtro, el cual puede ser rectilíneo o diagonal; pero solamente uno de

ellos, de manera que si por ejemplo se coloca un filtro rectilíneo de la forma -- la información con polaridad rectilínea será bien interpretada (ingresan los unos, no ingresan los ceros); el problema se presentará con filtros diagonales ya que dejarán pasar parte de las señales rectilíneas.

En un sistema criptográfico de esta naturaleza lo que se hace es generar la secuencia a transmitir y el receptor elige un filtro para cada bit de manera que interpreta lo recibido y lo guarda, al finalizar la transmisión, el emisor envía la polarización utilizada para cada bit, de manera que entonces el receptor compara esta secuencia con la que él utilizó para determinar las posiciones de los filtros que usó correctamente, precisamente esta secuencia de los filtros usados correctamente es la que le hará saber al emisor y se convertirá en la clave secreta que utilizarán conjuntamente para compartir información secreta.

Finalmente, los equipos destinados a la Criptografía cuántica deben ser capaces de operar con partículas subatómicas (poco común en la actualidad) considerando que entre las propiedades de este tipo de partículas se dará el caso de que un bit cuántico (qubit) pueda representar un cero, un uno o ambos simultáneamente (fenómeno llamado superposición cuántica) propiedad que permitiría realizar por ejemplo la factorización de números de más de 1000 dígitos (en lo que se basa RSA), lo que en la actualidad es prácticamente imposible en un tiempo razonable ya que con la capacidad de cómputo de hoy día se llevaría miles años realizar en tanto que con una computadora cuántica se podría llevar a cabo en unos 20 minutos aproximadamente.

2.5.2 DIFFIE-HELLMAN

EL ALGORITMO

- Creado por Whitfield Diffie y Martin Hellman en 1976.
- Publicado en el periódico New Directions in Criptography.

Se trata básicamente de un protocolo que basa su seguridad en el problema del logaritmo discreto y que permite a dos usuarios intercambiar una clave secreta a través de un medio inseguro, dicho protocolo también es conocido como *"el cambio de clave de Diffie-Hellman".*

Descripción del algoritmo:

1. Los usuarios *A* y *B* seleccionan públicamente un grupo *G* de orden *n* que sea un número primo *p* y un elemento $\alpha \in G$.
2. *A* genera un número aleatorio *a*, calcula α^a en *G* y envía el resultado a *B*.
3. *B* genera un número aleatorio *b*, calcula α^b en *G* y envía el resultado a *A*.
4. *A* recibe α^b y calcula $\left(\alpha^b\right)^a$ en *G*.
5. *B* recibe α^a y calcula $\left(\alpha^a\right)^b$ en *G*.

De este modo *A* y *B* poseen un elemento común secreto: α^{ab} .

Como puede apreciarse en la descripción anterior, el algoritmo por sí sólo no utiliza autenticación, por lo que su implementación es comúnmente acompañada de la utilización de firmas digitales ya que de no hacerlo se podría tener un ataque de hombre en medio.

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

El siguiente ejemplo muestra cómo dos usuarios A y B intercambian una clave secreta por un medio inseguro:

Ejemplo:

1. *A* y *B* seleccionan: $p, n = 53, G = Z_{53}^{*}, \alpha = 2$
2.
   a) *A* elige *a*= 29

   $\alpha^a \in G = \alpha^a \pmod n$

   $\alpha^a \equiv X \pmod n$

   b) $\frac{\alpha^a}{n} = \frac{2^{29}}{53} = 10129639 sobran \rightarrow 45$

   $2^{29} \equiv 45 (\mathrm{mod} 53)$

   c) *A* envía a *B*: 45 (llave pública)
3.
   a) *B* elige *b*= 19

   $\alpha^a \in G = \alpha^a \pmod n$

   $\alpha^a \equiv Y \pmod n$

   b) $\frac{\alpha^b}{n} = \frac{2^{19}}{53} = 9892 sobran \rightarrow 12$

   $2^{19} \equiv 12 (\mathrm{mod} 53)$

   c) *B* envía a *A*: 12 (llave pública)
4. *A* recibe 12 y calcula $(\alpha^b)^a$

   $(\alpha^b)^a \in G = (\alpha^b)^a \pmod n$

   $\frac{(12)^{29}}{53} = 37323319779838019863042498652 7 sobran \rightarrow 21$

   $(12)^{29} \equiv 21 (\mathrm{mod} 53)$

   21 es la clave secreta.

5. *B* recibe 45 y calcula $\left(\alpha^{a}\right)^{b}$

$\left(\alpha^{a}\right)^{b} \in G = \left(\alpha^{a}\right)^{b} (\mathrm{mod}\, n)$

$\frac{(45)^{19}}{53} = 4861405994030186235679770415 68 sobran \rightarrow 21$

$\left(\alpha^{a}\right)^{b} \equiv (45)^{19} \equiv 21 (\mathrm{mod}\, 53)$

21 es la clave secreta.

La clave secreta que comparten A y B es 21.

2.5.3 EL GAMAL

EL ALGORITMO

Como se aprecia a continuación, el algoritmo es bastante similar a Diffie-Hellman. Consideremos nuevamente el par de usuarios A (Anita) y B (Benito) quienes:

1) Han seleccionado sus parámetros $n, \alpha \; y \; Z^{*}_{n}$ y los han hecho públicos.

2) Tanto A como B también han elegido un número aleatorio primo *a* y *b* dentro de *n* (para cada usuario su número primo representa su clave privada).

3) Cada usuario debe calcular $\alpha^{a} (\mathrm{mod}\, n)$ y $\alpha^{b} (\mathrm{mod}\, n)$ respectivamente, de manera que los valores obtenidos representan para cada uno *(A y B)* su clave pública.

Así, para que Anita envíe información confidencial hacia Benito y él sea la única persona que pueda leerla, Anita deberá cifrar el mensaje haciendo uso de la clave pública de Benito; de manera que Benito descifrará el criptograma recibido utilizando su clave privada, esto es:

- **Cifrado**: *A* cifra un mensaje *M* y lo envía a *B*

1. *A* genera un número aleatorio v que representa un número de sesión y utilizando los parámetros que ha hecho públicos *B* calcula $\alpha^{v} (\mathrm{mod}\, n)$.

2. Además, con la clave pública de *B* $k_{púb} = [\alpha^{b} (\mathrm{mod} n)]$ *A* calcula:

$$\left(\alpha^{b}\right)^{v} (\mathrm{mod}\, n) \;\; y \;\; M * \left(\alpha^{b}\right)^{v} (\mathrm{mod}\, n)$$

Así, el criptograma que *A* envía a *B* está formado por el par:

$$Cripto = [\alpha^{v} (\mathrm{mod}\, n), M * \left(\alpha^{b}\right)^{v} (\mathrm{mod}\, n)]$$

- **Descifrado**: B descifra el criptograma *Cripto* que le envió *A*

1) *B* recibe $Cripto = [\alpha^v (\text{mod } n), M * (\alpha^b)^v (\text{mod } n)]$

2) Y toma el primer dato, o sea $\alpha^v (\text{mod } n)$ para calcular $(\alpha^v)^b (\text{mod } n)$

3) Ahora *B* realiza la operación $M * (\alpha^b)^v (\text{mod } n) / ((\alpha^v)^b (\text{mod } n))$

B ha descifrado el criptograma ya que $(\alpha^b)^v (\text{mod } n) = (\alpha^v)^b (\text{mod } n)$, esto es, está en condiciones de leer el mensaje *M*:

$$M = [M * (\alpha^b)^v * \{inv(\alpha^v)^b, n\}] (\text{mod } n)$$

Donde el inverso multiplicativo de $[(\alpha^v)^b, n]$, que también puede expresarse como el inverso multiplicativo de $(\alpha^v)^b (\text{mod } n)$, es el valor *z* que hace cumplir la congruencia:

$$z * (\alpha^v)^b \equiv 1 (\text{mod } n)$$

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

Ahora realizaremos el envío de información confidencial a través de un canal inseguro mediante el algoritmo visto. Para el ejemplo consideremos los mismos usuarios A (Anita) y B (Benito), donde Anita será quien envíe ciertos datos (información confidencial) a Benito:

- **Cifrado: A (Anita) enviará a B (Benito) M = 10**

1. A genera un número aleatorio v= 5 y calcula $\alpha^v (\text{mod } n)$ donde α =13 y *n*= 53 por lo tanto se tiene: $13^5 (\text{mod } 53) = 28$.
2. Además, con la clave pública de B, $k_{púb} = \alpha^b (\text{mod } n) = 15$, A calcula:

   $(\alpha^b)^v (\text{mod } n) = 15^5 \text{ mod } 53 = 44$, y

   $M * (\alpha^b)^v (\text{mod } n) = (10 * 44) (\text{mod } 53) = 16$

   Así, el criptograma que A envía a B estará formado por el par:

   Cripto = [28, 16]

- **Descifrado**

1. B recibe Cripto = [28, 16]
2. Y toma $\alpha^v (\text{mod } n) = 28$ para calcular $(\alpha^v)^b (\text{mod } n)$ donde la clave privada de Benito es b = 7, así 28^7 mod 53 = 44
3. Ahora B debe realizar la operación $[M * (\alpha^b)^v * \{inv(\alpha^v)^b, n\}] (\text{mod } n)$ para lo cual primero calcula el inverso multiplicativo de $(\alpha^v)^b (\text{mod } n)$, esto es, calcular el valor de z = inv (53, 44), para ello con base en el teorema de la división de Euclides se plantea la ecuación lineal 53x – 44y = 1

1) 53 = 1(44) + 9
2) 44 = 4(9) + 8
3) 9 = 1(8) + 1

Ahora, aplicando el algoritmo extendido de Euclides:

de 3): 1 = 9-1(8)
de 2): 1 = 9-1(44-4(9))=5(9)-1(44)
de 1): 1 = 5(53-1(44)) – 1(44)= 5(53)-6(44)
y x

$z = n + x = 53 - 6 = 47$ de manera que se cumple $44 * 47 \equiv 1(\text{mod } 53)$

$[M * (\alpha^b)^v * \{inv(\alpha^v)^b, n\}](\text{mod } n) = (16 * 47)(\text{mod } 53) = 10$

2.5.4 RSA (Rivest Shamir Adleman)

EL ALGORITMO

- Creado en 1977 por Ronald Rivest, Adi Shamir y Len Adleman y publicado en 1978.

Cada usuario debe realizar lo siguiente para generar su par de claves (pública y privada):

1. Elegir dos números primos grandes, *p* y *q*, y calcular el número *n* mediante: $n = p \cdot q$. El grupo a utilizar por el usuario es Z_n^* de orden $\phi(n) = \phi(p \cdot q) = (p-1)(q-1)$.
2. Seleccionar un entero positivo *e*, $1 \le e \le \phi(n)$, de modo que sea primo con el orden del grupo, esto es: $mcd(e, \phi(n)) = 1$.
3. Calcular *d* que es el inverso de *e* en $Z_{\phi(n)}^*$, de manera que: $e \cdot d \equiv 1(\text{mod}\,\phi(n))$, con $1 \le d < \phi(n)$.
4. Por último debe publicar la pareja (n, e) que es su clave pública, su clave privada *d y l*os valores $p, q, \phi(n)$ deben permanecer en secreto.

Para enviar un mensaje, antes que nada se debe determinar su longitud '*j*' dado que debe de ser un elemento del grupo en el que se esté trabajando, de modo que según el sistema de numeración en el que esté codificado cumpla:

$$b^j < n$$

Donde *b* es la base del sistema de numeración y *n* el grupo utilizado por el receptor. Si la longitud del mensaje que se desea enviar es más grande, éste debe ser dividido en bloques de tamaño *j* y cifrarlos por separado.

El proceso de cifrado consiste en transformar el mensaje *M* de acuerdo con:

$$C = M^{e} \pmod{n}$$

Para ello se utiliza la clave pública del receptor.

Para descifrar el criptograma *C* se utiliza la clave privada *d* según la expresión:

$$M = C^{d} \pmod{n}$$

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

El objetivo de este ejemplo es que el usuario *A* envíe al usuario *B* el mensaje YES, para ello *A* debe cifrarlo utilizando la clave pública de *B*, y éste debe descifrarlo con su clave privada.

Cálculo de las claves pública y privada de B:

- B elige *p=281* y *q=167*
- Cálculo de $n = p \cdot q = (281)(167) = 46927$
- Cálculo de $\phi(n) = \phi(p \cdot q) = (p-1)(q-1) = (280)(166) = 46480$
- B elige *e*= 39423
- Cálculo de *d:*

Se plantea la ecuación lineal $ex - \phi(n)y = 1$

$$39423x - 46480y = 1$$

Aplicando el algoritmo de Euclides:
46480= 39423(1) + 7057
⋮
5= 2(2) + 1 ⟶ mcd
2= 1(2) + 0

Aplicando el algoritmo de Euclides Extendido:
1= 5- 2(2)
⋮
1=16720(46480)-19713(39423)

Se tiene que *x= -19713* y *y =-16720*

De donde d:

$$d = \phi(n) + x = 46480 - 19713 = 26767$$

Clave pública *(n,e)*= (46927,39423)
Clave privada *d*= 26767

Obtención del criptograma:

Para poder manipular cada una de las letras del alfabeto, se emplea la codificación que transforma las letras de la A a la Z en los números del 0 al 25, tal y como se muestra en la figura 2.90.

Letra en el MCla	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Codifica-ción	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

FIGURA 2.90 Codificación del alfabeto inglés

Dado que el usuario A desea mandar al usuario B un mensaje, la longitud *j* de éste debe cumplir:

$$b^j < n$$

Donde *b*= 26 debido a que estamos ocupando el alfabeto inglés el cual tiene 26 caracteres diferentes.

$26^0 = 1$
$26^1 = 26 < 46927$
$26^2 = 676 < 46927$
$26^3 = 17576 < 46927$
$26^4 = 456976 > 46927$

Por lo tanto la longitud del mensaje debe ser de 3 letras o menos.

El mensaje que A enviará a B sólo tiene 3 caracteres, pero si fuera más grande habría que dividirlo en bloques de 3 letras y cifrarlos por separado.

La codificación del mensaje M= YES es:

MCla	Y	E	S
Pocisión	2	1	0
$m_{cod} = 24x26^2 + 4x26^1 + 18x26^0 = 16346$			

Una vez codificado el mensaje se cifra mediante:

$$C = m_{cod}^{e_b} \left(\mathrm{mod} n_b\right) = \left(16346\right)^{39423} \mathrm{mod}\left(46927\right) = 21166$$

Calculando el criptograma que se debe enviar:

	Cociente	Residuo	Letra decodificada	
21166 / 26	814	2	C	D_0
814 / 26	31	8	I	D_1
31 / 26	1	5	F	D_2
1 / 26	0	1	B	D_3

C= BFIC

Descifrado del criptograma:

B recibe BFIC y lo descifra mediante: $m_{cod} = C^{d_b} (\mathrm{mod}\, n_b)$ con su clave privada.

BFIC= $1x26^3 + 5x26^2 + 8x26^1 + 2x26^0 = 21166$

$$m_{cod} = (21166)^{26767} (\mathrm{mod}\, 46927) = 16346$$

	Cociente	Residuo	Letra decodificada	
16346 / 26	628	18	S	D_0
628 / 26	24	4	E	D_1
24 / 26	0	24	Y	D_2

MCla= YES

2.5.5 CURVAS ELÍPTICAS

CURVAS ELÍPTICAS SOBRE NÚMEROS REALES

Dado un campo *K*, una curva elíptica sobre *K*, es la curva plana sobre K definida por la ecuación:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, a_i \in K, i = 1,...,6$$

Una curva elíptica es un caso especial de una curva algebraica plana que tiene la propiedad de que los puntos que la conforman son un grupo Abelino aditivo.

Reglas de la adición geométrica sobre una curva elíptica:

1. Si tres puntos se encuentran sobre la curva elíptica su suma es igual a 0.
2. El punto 0 sirve como elemento aditivo idéntico: $P + 0 = P$. Cuando $P \neq 0$.
3. El negativo de un punto P es el punto con la misma coordenada en x, pero con coordenada y, negativa: el negativo del punto $P = (x, y)$ es $-P = (x, -y)$. Debemos hacer notar que estos 2 puntos se representan como una línea vertical

y además que $P+(-P)=P-P=0$. Al punto 0 se le llama punto cero o punto en el infinito.

4. Considerando la curva elíptica E. Dados dos puntos $P,Q \in E$, la recta $\overline{PQ}$ corta a E en tres puntos *P*,*Q* y *R'* tal y como se muestra en la figura 2.91. La suma de *P* y *Q* se define como $P+Q=R$.

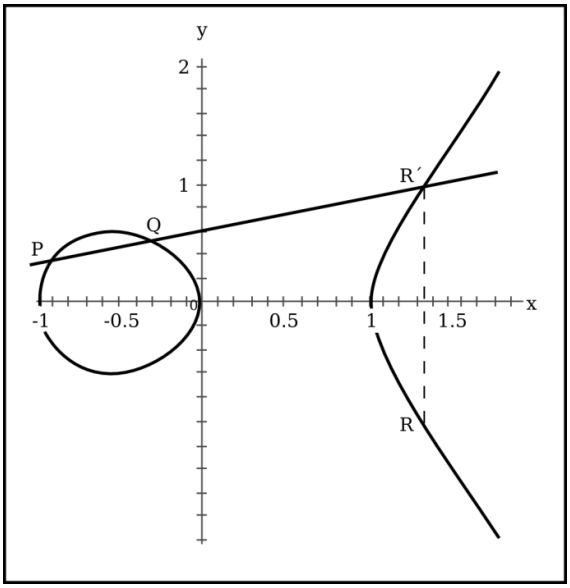


Figura 2.91 Curva elíptica

5. La interpretación geométrica anterior también se puede aplicar a dos puntos con la misma coordenada x como por ejemplo P y -P. En este caso: $P+(-P)=0$.
6. El punto doble P, traza una línea tangente a la curva y encuentra otro punto de intersección -R. Entonces $P+P=2P=R$.

CURVAS ELÍPTICAS EN CRIPTOGRAFÍA

Las curvas elípticas son utilizadas para implementar criptosistemas basados en el logaritmo discreto, con la ventaja de utilizar claves más pequeñas que repercute directamente en la utilización de menos memoria y hardware más pequeño.

En la Criptografía de curvas elípticas, el punto R se utiliza como clave pública que es la suma de los puntos P y Q, los cuales corresponden a la clave de tipo privada (ver figura 2.91).

De este modo, al conocer un punto de una curva, en este caso R, la información cifrada con este valor no se compromete, ya que la curva elíptica tiene infinidad de puntos y para poder descifrar un mensaje se necesitan conocer los parámetros P y Q para obtener su suma y así poder descifrar el mensaje.

En curvas elípticas, un ataque por fuerza bruta para conseguir la clave privada consiste en sumar todos los puntos de la curva elíptica hasta obtener como resultado el punto R. Para evitar que un ataque de este tipo sea exitoso y lograr que las claves sean lo más eficientes posible se escoge un campo de Galois lo suficientemente amplio, también se cuida que la curva elíptica elegida no sea singular, es decir que no pase por el punto cero, y por último se considera que la curva no sea anómala, es decir que el campo de Galois sobre el que se define la curva no tenga números racionales.

Debido a que las aplicaciones criptográficas requieren rapidez y precisión algebraica, en la práctica se utilizan el grupo de curvas elípticas sobre el campo finito de $GF(p)$ pertenecientes a los campos primos. Cabe mencionar que no existe una interpretación geométrica de la aritmética de curvas elípticas sobre los campos finitos.

El procedimiento para obtener las claves pública y privada es muy similar en la multiplicación de un punto por un escalar.

De tal forma que se utiliza un punto de una curva elíptica que sea un generador de todos los demás puntos de la misma para utilizarla como clave pública junto con el campo de Galois sobre el que definimos dicha curva y se utiliza un número entero aleatorio definido sobre el mismo campo de Galois que será la clave privada del algoritmo.
Así, se tiene que serán públicos el punto de la curva elíptica, el campo de Galois sobre el que se define dicha curva y finalmente el punto que se encuentra de multiplicar el punto generador por la clave privada elegida. La clave privada será el número entero aleatorio que pertenezca al mismo campo. Se debe considerar que el campo de Galois elegido debe ser un número primo lo suficientemente grande como para evitar ataques por fuerza bruta que se puedan presentar.

2.6 FUNCIONES HASH

2.6.1 INTRODUCCIÓN

Una función hash es una función matemática que reduce un mensaje original a una secuencia de bits de tamaño fijo que lo identifica (más pequeño); dicha secuencia es llamada valor hash o valor de dispersión. Las funciones hash proporcionan gran seguridad en la información ya que es muy difícil que dos textos tengan el mismo valor de dispersión, por ello son comúnmente utilizadas en grandes cantidades de datos.

Una función hash unidireccional es una función hash de modo que es prácticamente imposible encontrar dos mensajes originales con el mismo valor hash. Este tipo de función también es llamada función resumen y el valor hash es comúnmente llamado huella digital.

Dentro de las aplicaciones principales de los algoritmos hash se encuentra la creación de códigos de verificación:

- Creación del código MIC (Message Integrity Code): permite detectar si el contenido de un mensaje ha sido modificado. Por ejemplo, suponiendo que las entidades A y B han acordado una clave previamente:

  a) A concatena al mensaje la clave secreta y calcula el valor de dispersión de dicha concatenación.
  b) A envía a B el valor de dispersión y el mensaje.
  c) Al recibir el paquete, B concatena la clave secreta al mensaje que recibió y calcula el valor de dispersión de la concatenación.
  d) B compara el valor de dispersión que obtuvo con el que le envió A, si éstos coinciden, B puede tener la confianza de que el mensaje no ha sido modificado.

- Creación del código MAC (Message Authentication Code), el cual permite probar la integridad del contenido y la autenticación del origen de un mensaje, ya que asegura que el mensaje fue creado únicamente por alguien que conoce la clave.

2.6.2 SHA (Secure Hash Algorithm)

Algoritmo desarrollado por el NIST y publicado como estándar federal para el procesamiento de la información (FIBS PUB 180); en 1995 se publicó una versión revisada como FIBS PUB 180-1 conocida como SHA-1.

El algoritmo toma como entrada mensajes de longitud máxima de 2^{64} bits que son procesados en bloques de 512 bits; el resultado que produce es de 160 bits. La figura 2.92 muestra de manera general el algoritmo.

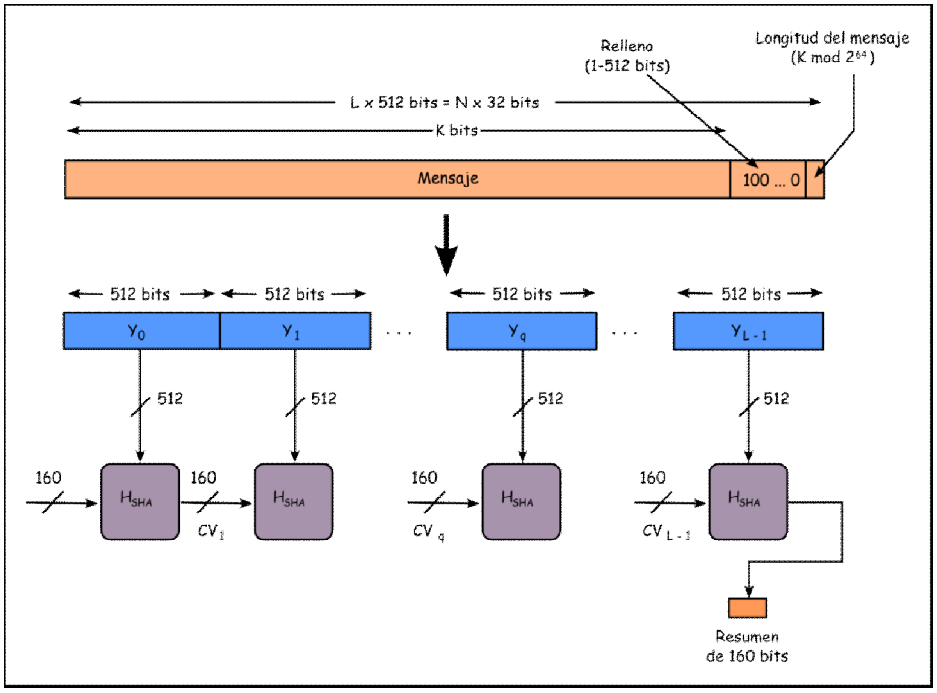


FIGURA 2.92 Algoritmo SHA-1

El procesamiento consta de cinco pasos los cuales se explican a continuación:

1. Se incorporan bits de relleno al mensaje de entrada de tal modo que cumpla: $longitud \equiv 448 \bmod 512$. El relleno consiste en un uno seguido de los ceros que sean necesarios. Aunque el mensaje ya tenga la longitud deseada, se debe efectuar el relleno, por lo que el número de bits de dicho relleno está en el rango de 1 a 512 bits.
2. A la salida del paso 1, se le añade un bloque de 64 bits que represente la longitud del mensaje original antes de ser rellenado.

3. Se inicializa la memoria temporal MD, la cual consta de 160 bits y su finalidad es almacenar los resultados intermedios y finales de la función de dispersión. La MD consta de 5 registros (A,B,C,D,E) de 32 bits cada uno, los valores con los que se inicializan son los siguientes (valores hexadecimales):

   A= 67452301

   B= EFCDAB89

   C= 98BADCFE

   D= 10325476

   C3D2E1F0
4. Se procesa el mensaje por bloques de 512 bits, cada uno pasa por un módulo que consta de 4 rondas de procesamiento de 20 pasos cada una. Las rondas tienen una estructura similar, con la excepción de que cada una ocupa una función lógica primitiva diferente (f_1, f_2, f_3 y f_4). Esta parte del algoritmo se muestra en la figura 2.93.

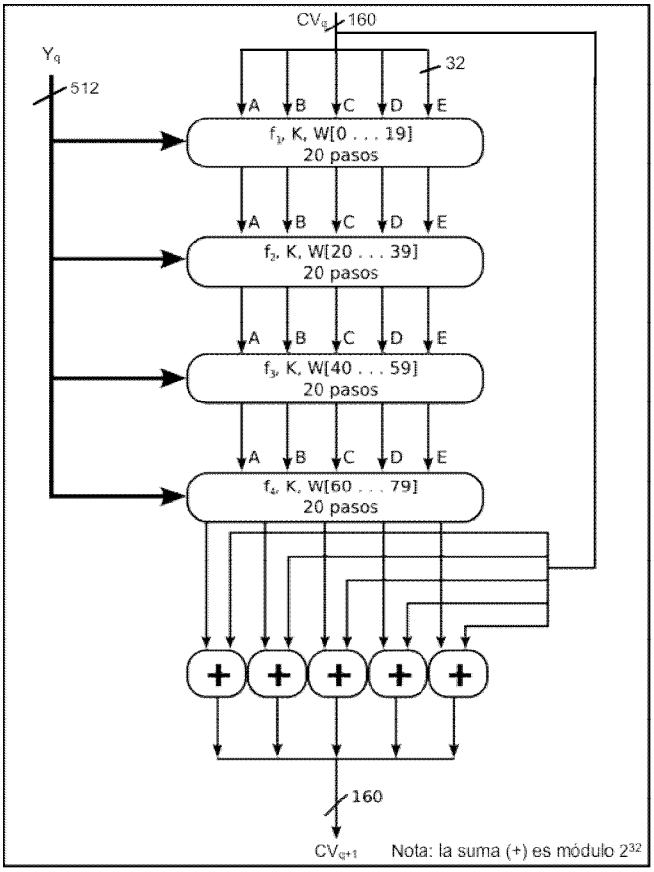


FIGURA 2.93 Procesamiento SHA-1 de un único bloque de 512 bits

La entrada a cada ronda consta del bloque de 512 bits que se esté procesando *(Y_q)* y los 160 bits de la memoria MD, nótese que cada bloque de 512 bits actualizará el valor de la memoria temporal. Cada ronda también hace uso de la constante aditiva K_t, donde $0 \leq t \leq 79$ indica uno de los 80 pasos a lo largo de las cuatro rondas. Los valores para dicha constante se muestran en la tabla de la figura 2.94.

Ronda	Número de paso	K_t (hexadecimal)
1	$0 \leq t \leq 19$	5A827999
2	$20 \leq t \leq 39$	6ED9EBA1
3	$40 \leq t \leq 59$	8F1BBCDC
4	$60 \leq t \leq 79$	CA62C1D6

FIGURA 2.94 Valores de la constante aditiva Kt en SHA-1

5. Una vez que se procesan los L bloques de 512 bits, el resumen del mensaje son los 160 bits de salida del último bloque.

2.6.3 MD4 (Message Digest Algorithm) [8]

El algoritmo MD4 fue desarrollado por Ron Rivest (uno de los creadores de RSA) y dado a conocer en 1990 a través de un RFC (Request For Comments), a menos de dos años, esto es, en la primavera de 1992 se publicó su actualización, nuevamente a través de un RFC (el 1320). Cabe señalar que esta publicación causó controversias dado que en esas mismas fechas se publicó MD5, algoritmo que corresponde a la siguiente versión de MD4 y que además persiguen los mismos objetivos:

- **Seguridad**. El algoritmo debe ser capaz de generar un "digest" (resumen) único para cada mensaje diferente, aún cuando la diferencia sea mínima (1 bit), dicho en otras palabras, que no exista posibilidad alguna de que dos mensajes diferentes generen el mismo bloque resumen.

- **Rapidez**. Que el algoritmo pueda ser fácilmente implementado (físicamente) con la finalidad de que su tiempo de procesamiento sea mínimo y entregue a la menor brevedad el bloque resumen. Para satisfacer dicho requerimiento la arquitectura de MD4 se implementó para trabajar palabras de 32 bits, una gran mejora con respecto a su antecesor, el MD2 que operaba por bytes (8 bits).

- **Sencillo y compacto**. Se pretende que el algoritmo pueda ser descrito de manera sencilla y que de igual forma se lleve a cabo su programación, con la finalidad de hacerlo ágil y fácil no solamente de implementar, sino de revisar y actualizar también.

2.6.4 MD5 (Message Digest Algorithm) [8]

El algoritmo MD5 fue desarrollado por Ron Rivest en 1992 en el MIT con la finalidad de robustecer el MD4 y a la fecha se trata del algoritmo hash más seguro y de mayor uso en el mundo (ampliamente documentado en el RFC 1321), entre las aplicaciones más recurrentes están la autenticación en el protocolo SSL y la firma digital en PGP.

MD5 procesa mensajes de cualquier longitud (longitud variable) y procesa bloques uniformes de 512 bits a la vez, hasta concluir con el mensaje total a fin de entregar a la salida un bloque "resumen" de 128 bits (longitud fija). El procesamiento consta de cinco pasos:

- **Paso 1**: En principio, para que el mensaje sea procesado en bloques de tamaño fijo se requiere que su longitud en bits sea: $long \equiv 448 \bmod 512$ de manera que su longitud sea de no menos de 64 bits de diferencia para completar un múltiplo de 512, así, de ser necesario se aplica un relleno, el cual varía en longitud en cada caso, esto es, el número de bits adicionados está en un rango de 1 a 512 bits con el formato 10000…0.

- **Paso 2**: Una representación de 64 bits de la longitud del mensaje original (antes de aplicar el relleno) se añade al resultado del paso 1. Estos dos pasos permiten hacer coincidir la longitud del mensaje en un múltiplo exacto de 512 bits de longitud

(requerido para el resto del algoritmo), en tanto adicionalmente se garantiza que mensajes distintos no serán iguales después del complemento de los bits.

- **Paso 3**: Es hasta este paso que da inicio el proceso de reducción, para ello se utiliza un registro de 128 bits que permite almacenar y mantener los resultados intermedios y final de la función hash. El registro se divide en cuatro secciones de 32 bits cada una, las cuales corresponden a las variables A, B, C, y D que son inicializadas con los valores hexadecimales:

  A = 67 45 23 01 B = EF CD AB 89 C = 98 BA DC FE D = 10 32 54 76

  Estas variables son llamadas variables de concatenación o variables de encadenamiento y se almacenan en formato little-endian de manera que aparecen:

  A = 01 23 45 67 B = 89 AB CD EF C = FE DC BA 98 D = 76 54 32 10

- **Paso 4**: El mensaje se procesa en bloques de 512 bits a la vez a través de 16 bloques de 32 bits cada uno, para lo cual las cuatro variables de concatenación se copian en variables distintas:

  a = A b = B c = C d = D

  La parte medular del algoritmo es una función de compresión que consta de cuatro rondas, las cuales tienen una estructura similar; pero cada una utiliza operaciones distintas durante 16 iteraciones; cada operación realiza una función no lineal sobre tres de las variables a, b, c y d, y el resultado es sumado a la cuarta variable que no fue elegida, un sub-bloque del texto y una constante.

  A ese resultado se le aplica una rotación circular a la izquierda un número variable de bits y se suma el resultado a una de las variables a, b, c, o d. Finalmente el resultado reemplaza a una de las variables a, b, c, o d. La salida de la cuarta ronda se suma a la entrada de la primera en una operación modular 2^{32} (figura 2.95).

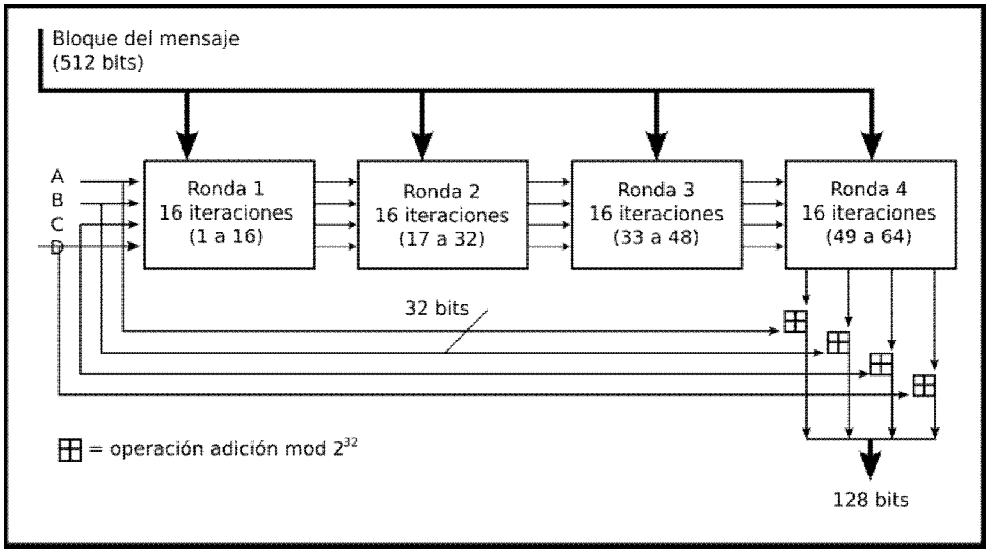


FIGURA 2.95 Función de compresión MD5

Hay cuatro operaciones no lineales utilizadas, una para cada ronda (véase la figura 2.96)

RONDA	FUNCIÓN OPERACIÓN
1	F(b, c, d) = (**b** AND **c**) OR (NOT **b** AND **d**)
2	G(b, c, d) = (**b** AND **d**) OR (**c** AND NOT **d**)
3	H(b, c, d) = **b** XOR **c** XOR **d**
4	I(b, c, d) = **c** XOR (**b** OR NOT **d**)

FIGURA 2.96 Funciones para MD5

Estas funciones son designadas de forma que si los bits que corresponden a b, c y d son independientes y no perjudiciales, cada bit del resultado también será independiente y no perjudicial. La función *F* es una función condicional: *If* b *then* c *else* d, de manera similar G: *If* d *then* b *else* c, en tanto la función *H* genera un bit de paridad.

Si *Mj* representa el *j-ésimo* sub-bloque del mensaje (desde 0 hasta 15), $y <<< s$ representa un cambio circular a la izquierda de *s* bits (véase figura 2.97), entonces las cuatro operaciones son:

$$FF(a,b,c,d,M_j,s,t_i)\_denota\_a = b + ((a + F(b.c.d) + M_j + t_i) <<< s)$$
$$GG(a,b,c,d,M_j,s,t_i)\_denota\_a = b + ((a + G(b.c.d) + M_j + t_i) <<< s)$$
$$HH(a,b,c,d,M_j,s,t_i)\_denota\_a = b + ((a + H(b.c.d) + M_j + t_i) <<< s)$$
$$II(a,b,c,d,M_j,s,t_i)\_denota\_a = b + ((a + I(b.c.d) + M_j + t_i) <<< s)$$

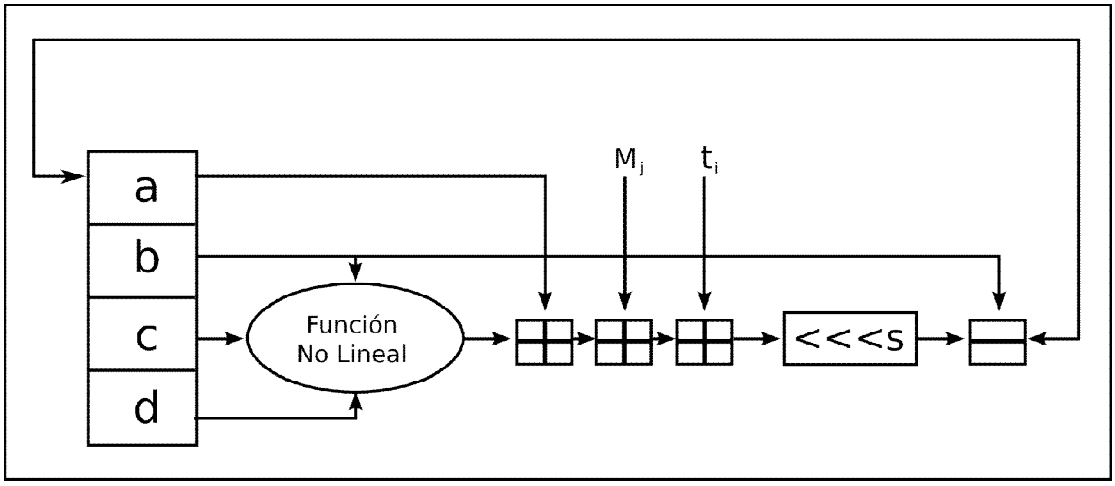


FIGURA 2.97 Operación en MD5

Las cuatro rondas (en total 64 pasos) serían de la siguiente forma:

Ronda 1:

$$FF(a,b,c,d,M_0,7,0xd76aa478)$$
$$FF(d,a,b,c,M_1,12,0xe8c7b756)$$
$$FF(c,d,a,b,M_2,17,0x242070db)$$
$$FF(b,c,d,a,M_3,22,0xc1bdceee)$$
$$FF(a,b,c,d,M_4,7,0xf57c0faf)$$
$$FF(d,a,b,c,M_5,12,0x4787c62a)$$
$$FF(c,d,a,b,M_6,17,0xa8304613)$$
$$FF(b,c,d,a,M_7,22,0xfd469501)$$
$$FF(a,b,c,d,M_8,7,0x698098d8)$$
$$FF(d,a,b,c,M_9,12,0x8b44f7af)$$
$$FF(c,d,a,b,M_{10},17,0xffff5bb1)$$
$$FF(b,c,d,a,M_{11},22,0x895cd7be)$$
$$FF(a,b,c,d,M_{12},7,0x6b901122)$$
$$FF(d,a,b,c,M_{13},12,0xfd987193)$$
$$FF(c,d,a,b,M_{14},17,0xa679438e)$$
$$FF(b,c,d,a,M_{15},22,0x49b40821)$$

Ronda 2:

$$GG(a,b,c,d,M_1,5,0xf61e2562)$$
$$GG(d,a,b,c,M_6,9,0xc040b340)$$
$$GG(c,d,a,b,M_{11},14,0x265e5a51)$$
$$GG(b,c,d,a,M_0,20,0xe9b6c7aa)$$
$$GG(a,b,c,d,M_5,5,0xd62f105d)$$
$$GG(d,a,b,c,M_{10},9,0x02441453)$$
$$GG(c,d,a,b,M_{15},14,0xd8a1e681)$$
$$GG(b,c,d,a,M_4,20,0xe7d3fbc8)$$
$$GG(a,b,c,d,M_9,5,0x21e1cde6)$$
$$GG(d,a,b,c,M_{14},9,0xc33707d6)$$
$$GG(c,d,a,b,M_3,14,0xf4d50d87)$$
$$GG(b,c,d,a,M_8,20,0x455a14ed)$$
$$GG(a,b,c,d,M_{13},5,0xa9e3e905)$$
$$GG(d,a,b,c,M_2,9,0xfcefa3f8)$$
$$GG(c,d,a,b,M_7,14,0x676f02d9)$$
$$GG(b,c,d,a,M_{12},20,0x8d2a4c8a)$$

Ronda 3:

$$HH(a,b,c,d,M_5,4,0xfffa3942)$$
$$HH(d,a,b,c,M_8,11,0x8771f681)$$
$$HH(c,d,a,b,M_{11},16,0x6d9d6122)$$
$$HH(b,c,d,a,M_{14},23,0xfde5380c)$$
$$HH(a,b,c,d,M_1,4,0xa4beea44)$$
$$HH(d,a,b,c,M_4,11,0x4bdecfa9)$$
$$HH(c,d,a,b,M_7,16,0xf6bb4b60)$$
$$HH(b,c,d,a,M_{10},23,0xbebfbc70)$$
$$HH(a,b,c,d,M_{13},4,0x289b7ec6)$$
$$HH(d,a,b,c,M_0,11,0xeaa127fa)$$
$$HH(c,d,a,b,M_3,16,0xd4ef3085)$$
$$HH(b,c,d,a,M_6,23,0x04881d05)$$
$$HH(a,b,c,d,M_9,4,0xd9d4d039)$$
$$HH(d,a,b,c,M_{12},11,0xe6db99e5)$$
$$HH(c,d,a,b,M_{15},16,0x1fa27cf8)$$
$$HH(b,c,d,a,M_2,23,0x4ac5665)$$

Ronda 4:

$$II(a,b,c,d,M_0,6,0xf4292244)$$
$$II(d,a,b,c,M_7,10,0x432aff97)$$
$$II(c,d,a,b,M_{14},15,0xab9423a7)$$
$$II(b,c,d,a,M_5,21,0xfc93a039)$$
$$II(a,b,c,d,M_{12},6,0x655b59c3)$$
$$II(d,a,b,c,M_3,10,0x8f0ccc92)$$
$$II(c,d,a,b,M_{10},15,0xffeff47d)$$
$$II(b,c,d,a,M_1,21,0x85845dd1)$$
$$II(a,b,c,d,M_8,6,0x6fa87e4f)$$
$$II(d,a,b,c,M_{15},10,0xfe2ce6e0)$$
$$II(c,d,a,b,M_6,15,0xa3014314)$$
$$II(b,c,d,a,M_{13},21,0x4e0811a1)$$
$$II(a,b,c,d,M_4,6,0xf7537e82)$$
$$II(d,a,b,c,M_{11},10,0xbd3af235)$$
$$II(c,d,a,b,M_2,15,0x2ad7d2bb)$$
$$II(b,c,d,a,M_9,21,0xeb86d391)$$

Las constantes t_i se seleccionaron de la siguiente manera:

En los pasos *i*, *ti* es la parte entera de $2^{32} * abs(sen(i))$, en donde *i* está en radianes.

- **Paso 5**: Al final de todos los ciclos, *a, b, c* y *d* son sumados a *A, B, C* y *D* respectivamente y el algoritmo continúa con el siguiente bloque de datos. Y después de que todos los bloques de 512 bits (cada uno) han sido procesados se obtiene la salida final, esto es, la concatenación de *A, B, C* y *D* que produce un bloque de 128 bits.

2.7 FIRMAS DIGITALES

2.7.1 INTRODUCCIÓN

Una firma digital es una transformación que por medio de una función relaciona de forma única un documento con la clave privada del firmante.

Clasificación de las firmas digitales:

a) **Implícitas**: contenidas en el mensaje.
b) **Explícitas**: añadidas como una marca inseparable del mensaje.
c) **Privadas**: sólo pueden identificar al remitente aquellos quienes compartan una clave secreta con éste.
d) **Públicas (o verdaderas)**: gracias a información públicamente disponible cualquiera puede identificar al remitente.
e) **Revocables**: el remitente puede negar que la firma le pertenece.
f) **Irrevocables**: el receptor puede probar que el remitente escribió el mensaje.

La firma digital comprende dos procesos principales:

1. **Firma** (el firmante *A* crea una firma digital *s* para un mensaje *M*):
   - **a)** Calcula $s = s_A(M)$, donde *s* es la firma de *A* sobre el mensaje *M* con la función de firma s_A.
   - **b)** Envía al receptor *B* la pareja *(M,s).*

2. **Verificación** (el receptor *B* verifica que la firma *s* sobre el mensaje *M* haya sido creada por *A*):
   - **a)** Obtiene la función de verificación V_A de *A*.
   - **b)** Calcula $v = V_A(M, s)$
   - **c)** Acepta la firma como creada por *A* si v= verdadero, y la rechaza si v= falso.

Las firmas digitales trabajan bajo el esquema de clave pública, en donde la clave privada se utiliza para firmar, y la pública para verificar la firma. Con el fin de evitar un ataque de hombre en medio, es aconsejable que se utilicen diferentes claves para el cifrado y para la firma digital.

Autenticación de identidad y de origen de datos, integridad y no repudio, son los servicios de seguridad que se proporcionan con el uso de firmas digitales.

En la práctica resulta poco eficiente firmar el documento completo, por lo que normalmente lo que se firma y envía es el hash del documento; en este caso tanto emisor como receptor además de acordar las funciones de firma y verificación, también deben acordar la función de dispersión.

2.7.2 DSA (Digital Signature Algorithm)

EL ALGORITMO

- Es un estándar del Gobierno Federal de los Estados Unidos de América o FIPS para firmas digitales.
- Propuesto por el Instituto Nacional de Normas y Tecnología de los Estados Unidos para su uso en su Estándar de Firma Digital (DSS), especificado en el FIPS 186.
- Se hizo público el 30 de agosto de 1991.
- Este algoritmo (como su nombre lo indica) sirve para firmar y no para cifrar información.
- Una desventaja de este algoritmo es que requiere mucho más tiempo de cómputo que RSA.
- El funcionamiento de DSA se divide en 3 etapas. Generación de claves, firma y verificación. Las dos primeras las realiza el emisor y la última el receptor tal y como se muestra en la figura 2.98.

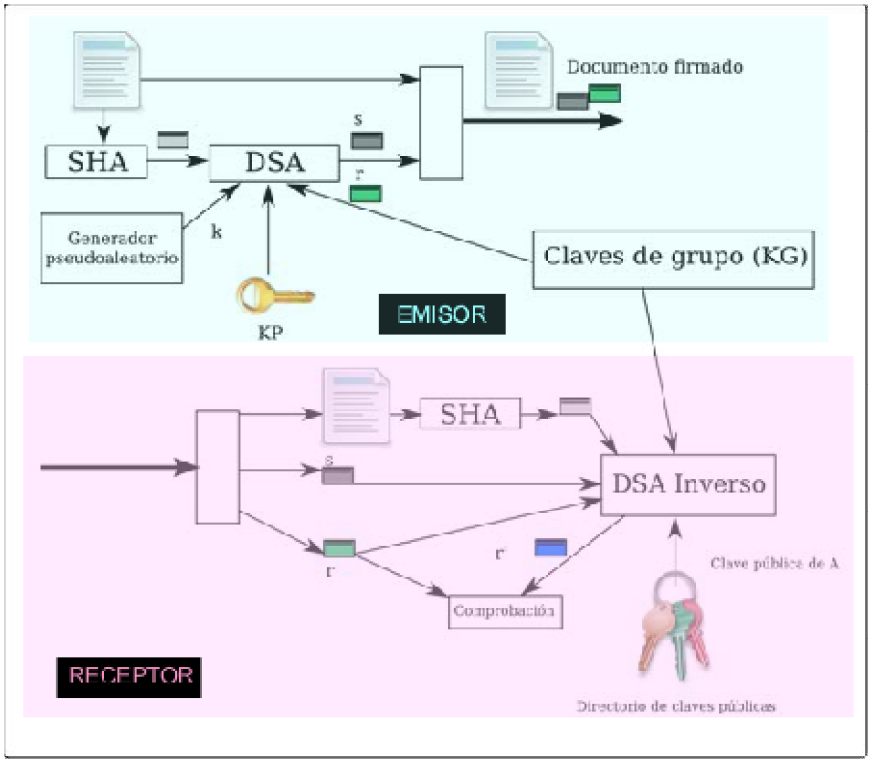


FIGURA 2.98 Diagrama DSA

- **PARÁMETROS**
  1. En la generación de claves: los datos son públicos excepto x

p	Número primo de 512 bits de longitud como mínimo
q	Número primo de 160 bits de longitud
g	Parámetro utilizado para calcular la clave pública
x	CLAVE PRIVADA DEL REMITENTE
y	CLAVE PÚBLICA DEL REMITENTE

  2. En la firma

k	Número pseudoaleatorio único para cada firma
s	Valor que corresponde a la firma
r	Valor de comprobación de la firma

  3. En la verificación

w	Valor que se requiere en el descifrado de la firma
u1	Dato relativo al valor del hash del mensaje en claro
u2	Dato relativo a la Integridad de la firma
v	Valor de comprobación y verificación de firma

APLICACIÓN DEL ALGORITMO: CASO PRÁCTICO

El objetivo de este ejemplo es que Benito envíe un mensaje en forma digital a Ana, el cual deberá estar firmado digitalmente para que ella pueda comprobar que él es quien lo envió. Posteriormente Ana recibe el mensaje junto con sus valores de verificación correspondientes, y realizará los cálculos necesarios para comprobar su autenticidad.

¿Qué necesita Benito para firmar su documento?

1. **Generar su clave privada**. La clave privada *x* deberá ser un número aleatorio de 160 bits el cual no es del todo "aleatorio" ya que debe cumplir con ciertas características según el estándar de DSS.

2. **Generar su clave pública**. Para generar la clave pública necesita:
   a) Obtener los números *p* y *q*, donde:
   - *p* será divisible por 64 y de longitud 512 bits
   - *q* será de longitud 160 bits y deberá de cumplir que

$p-1 = q * z$ (*z* es un número natural entero)

b) Calcular g utilizando la fórmula $g = h^{\frac{p-1}{q}} \bmod p$ donde: $1 < h < p-1$

c) Calcular la clave con la fórmula $y = g^x \bmod p$

3. **Número k correspondiente a la firma.** Éste número será único para cada firma y será un parámetro más para poder comprobarla, además que también debe cumplir con las características específicas del estándar donde: $0 < k < q$

4. **Generar la Firma de su documento.**

Calcular:
$$r = \left(g^k \bmod p\right) \bmod q$$
$$s = \left(k^{-1}\left(SHA(M) + x \cdot r\right)\right) \bmod q$$

Donde: SHA(M), es el resultado de aplicarle la función hash SHA-1 al documento, el cual no importando la longitud que tenga el SHA resultante tendrá siempre la misma longitud.

En el siguiente ejemplo práctico para efectos de comprobación de los parámetros del algoritmo, por practicidad, no se considerarán los estándares del algoritmo y únicamente se tomarán en cuenta números sencillos para conocer el funcionamiento correcto.

1. Clave privada. Se elige x= 5.

2. Clave pública.

a) obtener los números p y q (primos)

p= 23 q= 11

z=2 (z un número natural entero)

*se cumple que $p-1 = q * z$

b) calcular $g = h^{\frac{p-1}{q}} \bmod p$ donde: $1 < h < p-1$

usemos h= 10

$g = 10\text{^}2 \bmod 23$

g = 8

c) calcular $y = g^x \bmod p$

$y = 8\text{^}5 \bmod 23$

y = 16

Por lo tanto se tiene que:

- p=23
- q=11

- g=8
- x (clave privada) = 5
- y (clave pública) = 16

3. Número k correspondiente a la firma. Recordando que se debe cumplir $0 < k < q$, seleccionaremos k=4.

4. Generación de la firma.

Calculamos los parámetros de la firma:

$$r = \left(g^k \bmod p\right) \bmod q$$
$$r = \left(8\hat{}4 \bmod 23\right) \bmod 11$$
$$r = \left(2\right) \bmod 11$$
$$r = 2$$

Para el cálculo de *s*, necesitamos SHA(M), es decir, aplicarle la función hash al mensaje que se desea firmar, para ello procederemos de la siguiente manera:

v1	v2	v3	v1	v2	v3	v1	v2	v3	
H	o	l	a		A	n	a		suma
72	111	108	97	32	65	110	97	32	429
n	o	s		v	e	m	o	s	
110	111	115	32	118	101	109	111	115	-9031
	a		L	a	s		1	7	
32	97	32	76	97	115	32	49	55	-5430
SHA(M)= -14.032									-14032

Por cada carácter, incluyendo espacios, se buscará su valor en el código ascci, posteriormente se agruparan los caracteres de 3 en 3, haciendo la siguiente operación con los valores del código ascci $(v1 - v2) * v3$, finalmente se sumarán todos los datos obtenidos y ese será el SHA(M).

M= Hola Ana nos vemos a Las 17

Finalmente podemos calcular "s":

$$s = \left(k^{-1}\left(SHA(M) + x \cdot r\right)\right) \bmod q$$
$$s = \left(0.25\left(-14.032 + 5 \cdot 2\right)\right) \bmod q$$
$$s = \left(0.25\left(-14.032 + 10\right)\right) \bmod q$$
$$s = \left(0.25\left(-4.032\right)\right) \bmod q$$
$$s = \left(-1.008\right) \bmod 11$$
$$s = -1.008$$

s corresponde al valor de la firma

Ahora Ana recibe el mensaje en claro además de los datos correspondientes a la firma, y procede a comprobar la autenticidad de la misma.

Datos que recibe:

- Mensaje en claro junto con el valor HASH del mismo
  M= Hola Ana nos vemos a Las 17
  SHA(M)= -14.032
- Valores de la firma r' y s'
  $r' = 2$
  $s' = -1.008$
- Clave pública de Benito
  y= 16
- Otros
  p=23
  q=11
  g=8

Recordemos que en el esquema de firma digital, el receptor recibirá el mensaje en claro tal como el emisor lo desea enviar, ya que éste se encuentra firmado, no cifrado.

Ahora Ana procede a calcular los parámetros de verificación para comprobar la firma calculando los siguientes valores :

$$w = \left(s'\right)^{-1} \bmod q$$
$$u1 = \left(SHA(M)w\right) \bmod q$$
$$u2 = \left((r')w\right) \bmod q$$
$$v = \left(\left((g)^{u1}(y)^{u2}\right) \bmod p\right) \bmod q$$

Finalmente debe comparar el valor obtenido v con r':

- Si son iguales la firma se verifica y se puede confiar en la integridad del mensaje y asegurar que éste es de Benito.

- Si no son iguales el mensaje pudo haber sido alterado, o firmado incorrectamente, por lo que lo deberá rechazar.

Ana procede a calcular los parámetros de comprobación con los datos con los que cuenta.

- $w = (s')^{-1} \bmod q$
  $w = (-1.008)^{-1} \bmod 11$
  $w = -1 \bmod 11$
  $w = -1$
- $u1 = (SHA(M)w) \bmod q$
  $u1 = (-14.032 * -1) \bmod 11$
  $u1 = (14.032) \bmod 11$
  $u1 = 3$
- $u2 = (r' * w) \bmod q$
  $u2 = (2 * -1) \bmod 11$
  $u2 = (-2) \bmod 11$
  $u2 = -2$

Finalmente se calcula:

- $v = (((g)^{u1}(y)^{u2}) \bmod p) \bmod q$
  $v = (((8)^3(16)^{-2}) \bmod 23) \bmod 11$
  $v = ((512)(0.004) \bmod 23) \bmod 11$
  $v = (2 \bmod 23) \bmod 11$
  $v = 2 \bmod 11$
  $v = 2$

$v = r' = 2 = r$ por lo tanto se comprueba la firma

La figura 2.99 muestra la comunicación entre Ana y Benito.

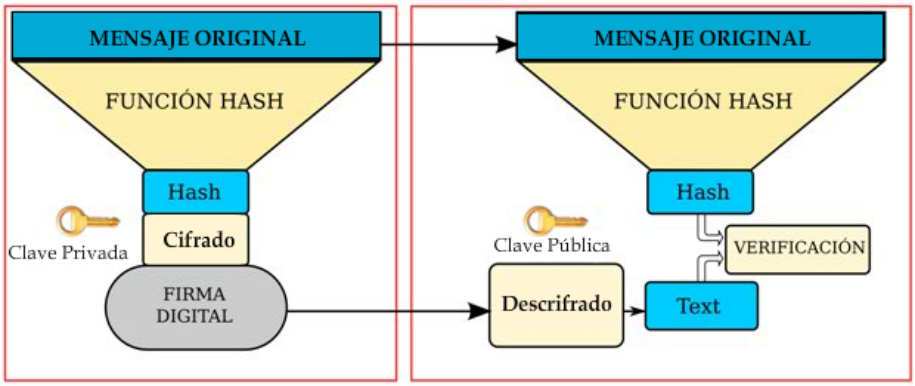


FIGURA 2.99 Comunicación entre Ana y Benito

CAPÍTULO 3

Análisis, Diseño y Desarrollo del Sistema

En este capítulo se presentan las fases de análisis, diseño y desarrollo para realizar el sitio Web que contendría a su vez el tutorial de fundamentos de Criptografía.

3.1 ANÁLISIS

3.1.1 DEFINICIÓN DE OBJETIVOS

El proyecto consiste en la creación de un sistema tutorial sobre fundamentos de Criptografía basándose en el temario de la asignatura "Criptografía" impartida en la Facultad de Ingeniería de la Universidad Nacional Autónoma de México. La finalidad es proporcionar a todas aquellas personas interesadas en el tema, información sobre los fundamentos que rigen la Criptografía de nuestros días, sin embargo, cabe señalar que está dirigido en especial a los estudiantes de la Facultad de Ingeniería de la UNAM que optan por el módulo terminal de Redes y Seguridad de tal modo que tengan una herramienta adicional para el aprendizaje de la materia, toda la información podrá consultarse a través de un sitio Web alojado en un servidor interno para tenerla disponible en el momento que se desee; en dicho sitio se podrá encontrar información de cada uno de los capítulos que conforman el temario de la asignatura, así como ejemplos y referencias bibliográficas para ampliar el conocimiento de cada uno de los temas.

3.1.2 IDENTIFICACIÓN DE ALCANCES Y LÍMITES

Recursos disponibles:

- Servidor ubicado en el laboratorio de Redes y Seguridad de la Facultad de Ingeniería, con la tecnología LAMP (Linux, Apache, MySQL, PHP), en donde ya se tienen hospedados los siguientes tutoriales: tutorial de Seguridad Informática, tutorial de Curvas Elípticas, redes Wi-Fi y Wimax, tutorial de Redes de Datos, Biometría Informática, entre otros.

Tomando en consideración los objetivos y los recursos con los que se cuenta para desarrollar el sistema, uno de los puntos primordiales es responder la pregunta ¿qué va a incluir el sistema?, con ello deben responderse más preguntas tales como ¿qué información se necesita?, ¿quién la necesita?, ¿cuándo?, ¿dónde? y ¿en qué forma?.

De tal modo que se identifican los siguientes alcances:

1. Disponibilidad vía Internet del tutorial de fundamentos de Criptografía por medio de un sitio Web alojado en el servidor del laboratorio de Redes y Seguridad junto con los demás tutoriales.
2. Disponibilidad de información teórica sobre los fundamentos de Criptografía.
3. Disponibilidad de ejercicios sobre los distintos temas así como referencias bibliográficas para ampliar el conocimiento de los mismos.
4. Proporcionar un medio de contacto, en este caso correo electrónico para solución de dudas sobre la información expuesta en el sitio.

Dentro de los límites se identifican los siguientes:

1. No se pretende dar información sobre cómo aplicar la Criptografía en casos prácticos, debe quedar claro que sólo se pretende dar a conocer las bases teóricas que han dado lugar a la Criptografía actual.
2. No se pretende tener mayor interacción con el usuario, aunque este punto queda establecido para una evolución y mejoramiento del sitio, ya que la plataforma permite hacerlo fácilmente.

3.1.3 ESTABLECIMIENTO DE REQUERIMIENTOS

Los requerimientos determinan las capacidades que debe tener el sistema para cumplir con los objetivos. Son establecidos para la funcionalidad, rendimiento, el equipo, la programación y las interfaces con el usuario, también pueden establecer estándares de desarrollo y de control de calidad del sistema.

En 2009 la Universidad Nacional Autónoma de México creó el Consejo Asesor en Tecnologías de Información y Comunicación (CATIC), que con la consultoría de expertos pretende promover estrategias para modernizar y actualizar de forma permanente la infraestructura tecnológica (Informática y Comunicaciones) universitaria, además de fortalecer la metodología de difusión; en este marco dicho consejo estableció lineamientos para el desarrollo de sitios Web institucionales permitiendo de este modo fortalecer la imagen institucional en la Red de las actividades de la Universidad, a nivel nacional e internacional.

Lineamientos generales emitidos por la CATIC para sitios Web institucionales [28]:

1. Los títulos de todas las páginas del sitio Web deben ser cortos pero descriptivos.
2. La página principal del sitio Web debe contener los siguientes elementos:

    - Escudo de la UNAM con los elementos establecidos en el Reglamento del Escudo y Lema de la UNAM. El escudo debe incluir un enlace hacia la página principal del portal de la UNAM.
    - Escudo o logo de la Facultad de Ingeniería.
    - Nombre completo de la UNAM (“Universidad Nacional Autónoma de México”) y de la Facultad de Ingeniería.
    - Enlace a la página de créditos (reconocimiento a colaboradores del sitio).
    - Correo electrónico de un contacto que dé asesoría o canalice solicitudes, dudas o comentarios de los usuarios acerca del sitio y su contenido.
    - Dirección postal de la Facultad de Ingeniería o de la DIE (División de Ingeniería Eléctrica)

3. Los contenidos deben contemplar los siguientes elementos de comunicación:

- Uso de lenguaje simple, claro y directo, que permita a los lectores concentrarse en el mensaje y comprenderlo de manera fácil y sencilla.
- Uso de palabras apropiadas al contexto del contenido y al perfil de la comunidad a la que está enfocado el contenido.
- Estructura gramatical, ortografía y redacción correctas.

4. Utilizar combinaciones de colores que identifiquen a la entidad universitaria.
5. El sitio Web debe contar con una distribución clara y consistente que permita al usuario identificar las áreas que la componen (navegación, menús, despliegue de información, entre otros).
6. La construcción de las páginas que conforman el sitio Web debe respaldarse en las recomendaciones y especificaciones para HTML publicadas por la W3C, verificando la compatibilidad con las versiones de los navegadores que pretendan utilizarse.

Con el afán de producir información de calidad y seguir los lineamientos establecidos por la CATIC se establecieron requerimientos operacionales para el presente trabajo tales como:

- **Confiabilidad**. Grado de seguridad con que el recurso realiza su función.
- **Disponibilidad**. El sistema debe ser accesible a los usuarios en el momento que ellos lo requieran.
- **Flexibilidad**. Habilidad del sistema para cambiar o adaptarse para satisfacer los requerimientos cambiantes de los usuarios.
- **Expectativa de vida y potencial de crecimiento**. El sistema debe ser capaz de satisfacer requerimientos durante un tiempo razonable y ser capaz de crecer si las necesidades cambian de manera significativa.
- **Capacidad de recibir mantenimiento**. Una vez que el sistema está en producción debe recibir mantenimiento en caso de existir fallas o para satisfacer solicitudes especiales.

3.2 DISEÑO

3.2.1 COMPONENTES DEL DISEÑO WEB

El diseño Web es la combinación de varios componentes que en conjunto permiten obtener sitios Web fáciles de entender, atractivos visualmente, fáciles de utilizar y técnicamente sólidos, es decir que se comporten como se espera.

De tal modo que es posible identificar cinco componentes del diseño Web (ver figura 3.1):

1. **Tecnología**: conjunto de herramientas que permiten la construcción de un sitio Web.
2. **Contenido**: información o mensaje que se presenta y razón por la cual los usuarios visitan un sitio Web.
3. **Arquitectura del sitio**: la forma en que se organiza el contenido.

4. **Diseño visual**: también llamada estética del sitio Web se refiere a la presentación visual proporcionando una identidad y coherencia visual a lo largo de todo el sitio.
5. **Interacción**: la forma en que se comporta el sitio al responder a las acciones del usuario.

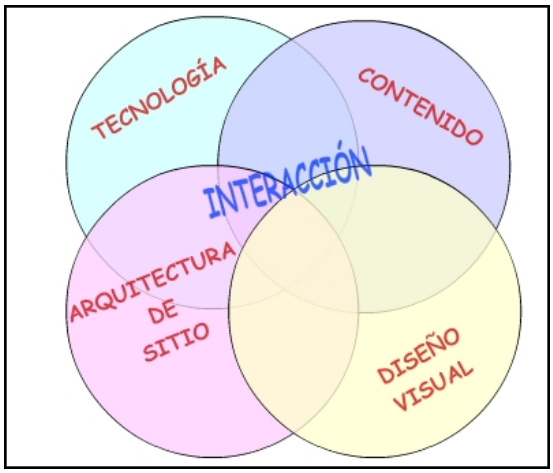


FIGURA 3.1 Componentes interdependientes del diseño Web

3.2.2 TECNOLOGÍAS UTILIZADAS

- **HTML/XHTML**

HTML (Hyper Text Markup Language) es un sistema de codificación basado en texto que ha sido utilizado para dar formato a los datos con el fin de que puedan ser visualizados en un navegador Web.

Está basado en etiquetas de marcado que indican al navegador cómo mostrar el contenido de un documento incluyendo texto, imágenes y otros medios de apoyo.

Debido al rápido crecimiento de la Web cada fabricante de navegadores comenzó a añadir sus propias etiquetas HTML con el fin de adaptarlo a los nuevos requerimientos, provocando de este modo problemas de compatibilidad entre plataformas, dando como resultado que un mismo archivo HTML se comportará de manera diferente en cada navegador.

A mediados del año 2000, el W3C desarrolló un nuevo estándar conocido como XHTML, el cual está basado en HTML, pero utilizando las reglas de formación de XML, lo que permite a los navegadores compatibles con este estándar validar los documentos antes de ser procesados y rechazar todos aquellos que no cumplan con las reglas.

Al conjunto formado por el texto y sus correspondientes etiquetas de marcado se le conoce como documento XHTML. Estos documentos pueden ser generados

dinámicamente por el servidor Web o, en caso de los documentos estáticos, estarán almacenados en archivos de texto con extensión .xhtml.

- **XML (Extensible Markup Language)**

Se trata de un estándar para definir información de forma estructurada y jerárquica mediante la utilización de etiquetas personalizadas las cuales no tienen un significado predefinido, sólo sirven para marcar los datos y que las aplicaciones receptoras del documento interpreten y manipulen los datos según sus necesidades.

Características principales de XML:

a) **Comprensible**: XML permite describir la información de manera que ésta sea fácilmente comprensible tanto para quien lee el documento como para las posibles aplicaciones que lo van a procesar.
b) **Basado en texto**: los documentos XML están basados en texto, lo que significa que no es necesario disponer de herramientas específicas para crear, interpretar y manipular información.
c) **Independiente**: XML es una tecnología desarrollada por el W3C que ha desarrollado una serie de estándares abiertos para ayudar a los programadores en el desarrollo de aplicaciones para XML. Esto posibilita que un documento generado por una aplicación escrita en un determinado lenguaje pueda ser interpretado por otra aplicación basada en una tecnología diferente.

Entre las principales aplicaciones de XML podemos destacar las siguientes:

- Intercambio de datos entre aplicaciones
- Almacenamiento intermedio de datos en aplicaciones Web
- Presentación de datos en la Web
- Utilización como base de datos

- **Hojas de estilo en cascada (CSS)**

Cascading Style Sheets (CSS), son un lenguaje formal utilizado para definir la presentación de un documento estructurado en HTML, XML, XHTML, etcétera; permitiendo separar la estructura del documento de su presentación. El W3C (World Wide Web Consortium) es el encargado de formular la especificación estándar de las hojas de estilo.

Las hojas de estilo se basan en la definición de una serie de propiedades de estilo asociadas a determinados tipos de etiquetas e independientes de su contenido, de manera que al ser utilizadas dichas etiquetas en el documento se apliquen automáticamente las diferentes propiedades y opciones de estilo definidas para las mismas.

- **Javascript**

Javascript es un lenguaje de programación que se puede utilizar para incorporar interactividad a las páginas Web, es soportado prácticamente en todos los navegadores y se caracteriza por poseer un reducido conjunto de instrucciones, a fin de reducir lo más posible la complejidad del intérprete de script del navegador y poder ser accesible a una mayor cantidad de usuarios.

Un script es un bloque de código que se incluye directamente en el documento HTML y que puede ser interpretado por el navegador. Mediante dichos scripts se pueden incluir instrucciones de código que correspondan a acciones de usuario y que sean capaces de modificar dinámicamente el aspecto de las páginas.

- **AJAX**

AJAX (Asynchronous Javascript And XML) es una técnica de programación, basada en el uso de un conjunto de tecnologías Web y estándares de cliente, consistente en la solicitud asíncrona de datos al servidor desde una página Web y la utilización de éstos para actualizar una parte de la misma, sin la necesidad por parte del navegador de realizar una recarga completa de toda la página.

- **PHP**

PHP, acrónimo de "Hypertext Preprocessor", es un lenguaje "Open Source" interpretado de alto nivel, especialmente pensado para desarrollos Web y el cual puede ser incrustado en páginas HTML. La meta de este lenguaje es permitir escribir a los creadores de páginas Web, páginas dinámicas de una manera rápida y fácil, aunque se pueda hacer mucho más con PHP.

El código PHP es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá los resultados de ejecutar el script, sin ninguna posibilidad de determinar qué código ha producido el resultado recibido.

- **Adobe flash**

Es una aplicación que trabaja sobre *fotogramas (*que se puede entender como un instante o momento de una película, es un equivalente a cuadro de un film*)* destinado a la producción y entrega de contenido interactivo sin importar la plataforma. Actualmente es escrito y distribuido por *Adobe Systems,* y utiliza gráficos vectoriales e imágenes ráster, sonido, código de programa, flujo de vídeo y audio bidireccional.

Flash inicio como una tecnología de animación de gráficos, pero hoy en día se puede utilizar para crear sitios Web completos.

- **Navegadores**

Un navegador es una aplicación que permite al usuario solicitar, recuperar y visualizar documentos de hipertexto, comúnmente en formato HTML o XHTML, cuando el navegador recibe el documento lo interpreta secuencialmente y muestra información contenida en el mismo de acuerdo con el formato definido por las etiquetas,

permitiendo así mostrar o ejecutar gráficos, secuencias de video, sonido, animaciones y programas diversos, así como texto y los hipervínculos o enlaces.

La figura 3.2 muestra el predominio de los navegadores hasta Diciembre de 2011 [26].

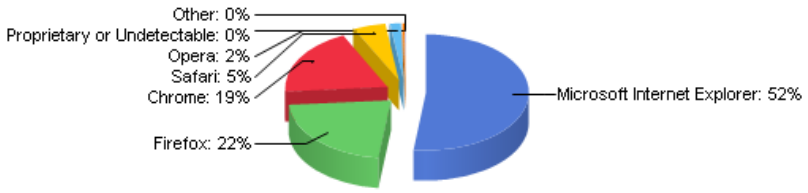


FIGURA 3.2 Predominio de navegadores, hasta Diciembre de 2011

3.2.3 ARQUITECTURA DE LA INFORMACIÓN

El contenido es la base de un sitio Web y el motivo por el cual las personas visitan las páginas. El diseño de la información implica la revisión y organización del contenido de un sitio Web y la interfaz de usuario para que el público objetivo encuentre fácilmente lo que busca. El diseño de la información puede definirse como: *"el arte y la ciencia de estructurar y clasificar sitios Web para ayudar al usuario a encontrar y gestionar la información."*

Así pues la arquitectura de la información es la organización y clasificación de contenidos, de tal forma que será de fácil entendimiento para el usuario final.

En la figura 3.3 se muestra la pantalla inicial del sitio Web desarrollado, la disposición de los distintos elementos tienen la finalidad de proporcionar una fácil navegación a los usuarios y que les resulte fácil hacer lo que deseen y encontrar lo que están buscando.

FIGURA 3.3 Pantalla inicial del sitio Web

3.2.4 USABILIDAD Y CALIDAD DE LA INFORMACIÓN

Los principios de usabilidad se basan en aumentar la satisfacción del usuario; cuando éste se siente satisfecho con lo que está viendo en un determinado sitio, permanece en él y si lo requiere regresa a visitarlo nuevamente.

Los factores de usabilidad para mejorar la satisfacción del usuario son los siguientes:

- **Evidencia**: el sitio debe ser fácil de utilizar, para ello se debe ofrecer una organización, presentación e interacción coherente y predecible.
- **Velocidad**: se debe navegar dentro del sitio con el mínimo de clics, se debe tratar que la información este lo más a la mano posible.
- **Retroalimentación:** En caso de que el usuario deba esperar por algún contenido se debe avisar de ello y no mostrar pantallas en blanco o incompletas.
- **Exactitud:** no deben existir vínculos rotos, imágenes perdidas, errores Javascript o cualquier otra cosa que no funcione.

La figura 3.4 muestra cómo se proporciona una fácil navegación a través de los contenidos.

FIGURA 3.4 Usabilidad del sitio Web

3.3 DESARROLLO EN JOOMLA

- **Frontend y Backend**

El frontend se refiere a la parte pública; comprende las áreas del sitio Web tal y como los visitantes las ven. El backend es el área de administración en donde sólo los usuarios con permisos tendrán acceso, en esta área se pueden crear usuarios con sus respectivos permisos y gestionar varias tareas del sitio Web.

Para la realización del presente trabajo solo se utilizó el usuario administrador.

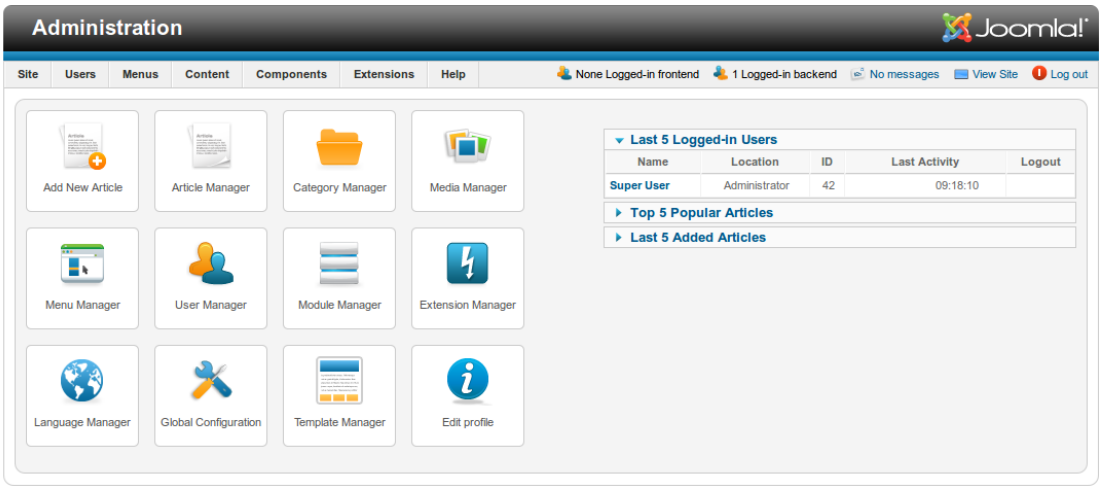


FIGURA 3.5 Interfaz de administración de Joomla

- **Archivos**

Joomla está formado por diversos archivos como imágenes, scripts PHP, archivos CSS, archivos Javascript, entre otros.

Los archivos necesarios para el backend de Joomla está localizado dentro del directorio *administrator* (figura 3.6), son los que se usan cuando se accede a la administración del sitio. Los directorios y archivos fuera de la carpeta *administrator* corresponden al frontend.

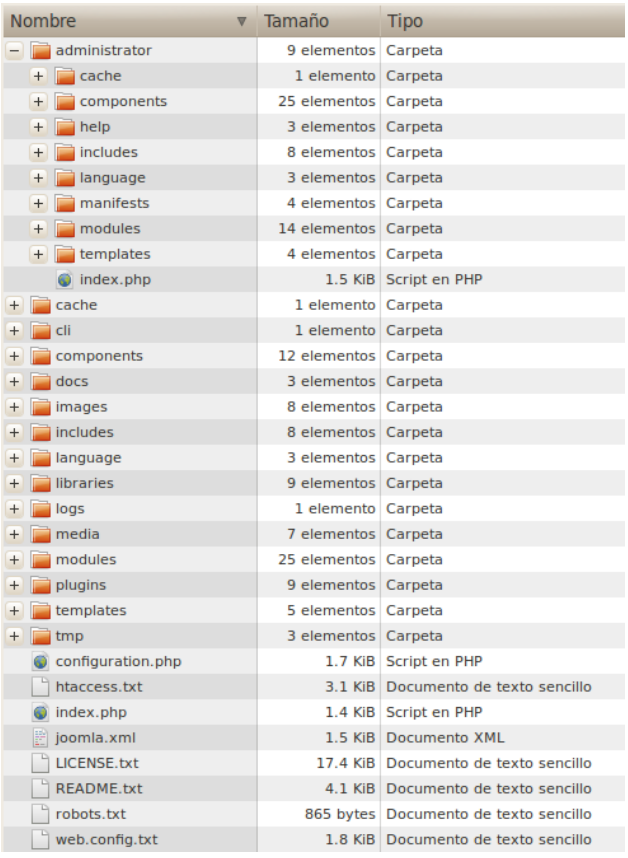


FIGURA 3.6 Archivos y directorios de Joomla

- **Base de datos MySQL**

Además de los archivos (gráficos, documentos, archivos de sistema, etc.) Joomla también utiliza una base de datos. Inicialmente consiste en 61 tablas en donde se almacena la mayor parte de configuración del sistema y de los contenidos del sitio Web.

Por ejemplo todos los contenidos (capítulos del temario) quedaron alojados en la tabla x_content.

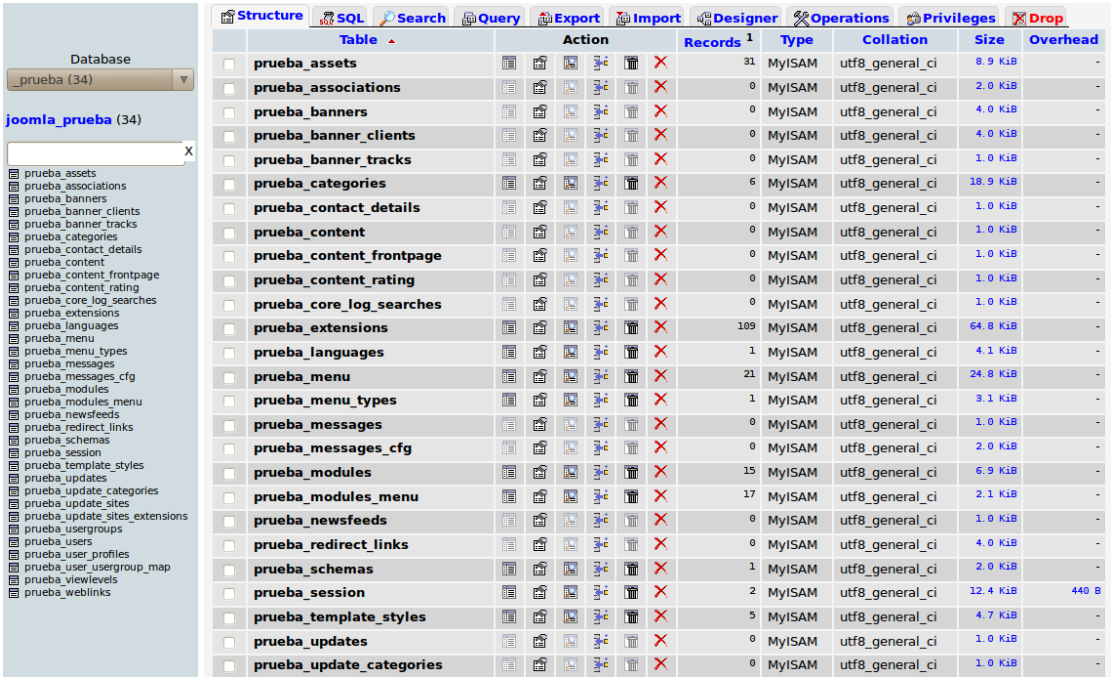


FIGURA 3.7 Tablas de la BD Joomla

Características de la base de datos

Nombre	Descripción
Sistema de gestión	MySQL
Número de tablas	61
Tamaño	0.7 MB
Motor de almacenamiento	MyISAM

- **Artículos, categorías y menús.**

Los artículos son básicamente textos con un título, aunque también pueden contener imágenes y otros tipos de medios.

Cada página desplegada en el sitio Web corresponde a un artículo, a su vez cada artículo debe estar asociado a una categoría, es así como se muestran y gestionan los artículos de forma clara y ordenada (figura 3.8).

Posteriormente cada artículo es ligado a un ítem del menú (figura 3.9 y figura 3.10).

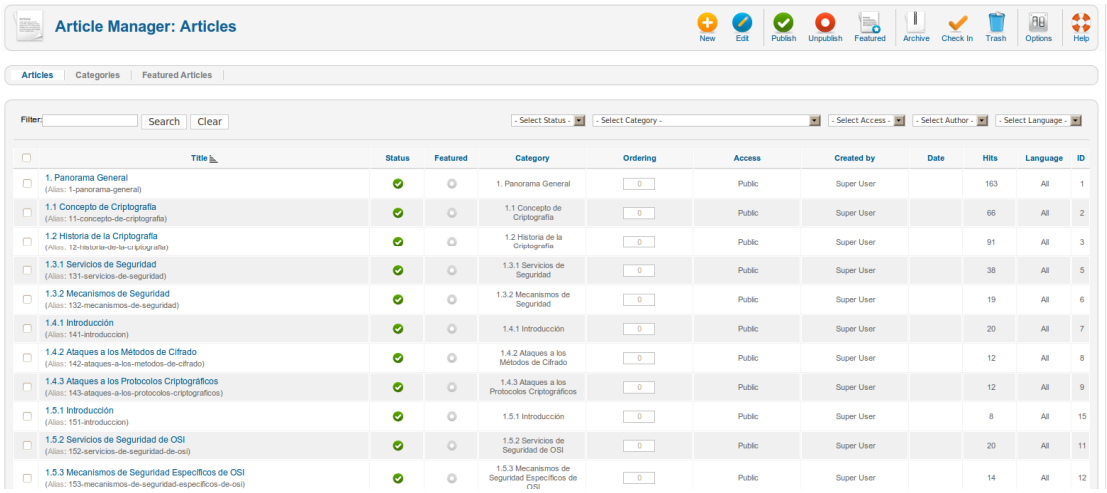


FIGURA 3.8 Artículos y categorías del sitio

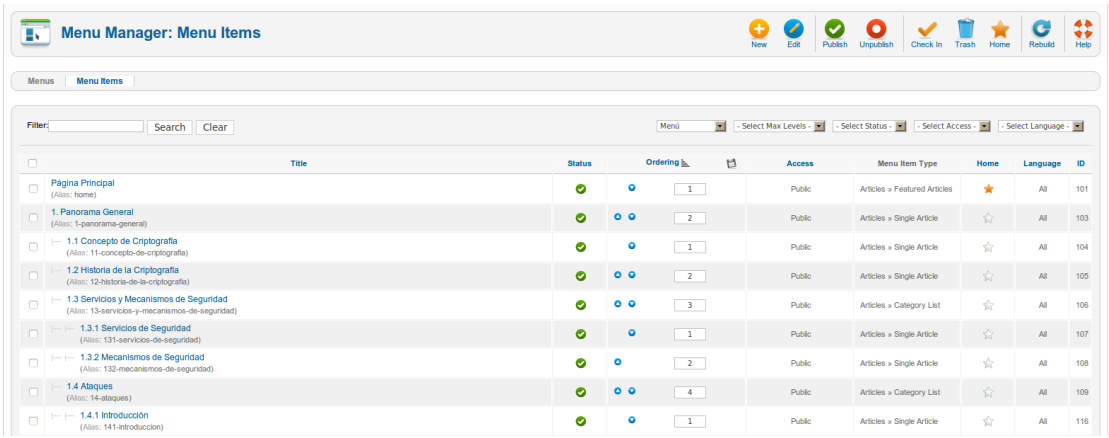


FIGURA 3.9 Elementos del menú lateral del sitio

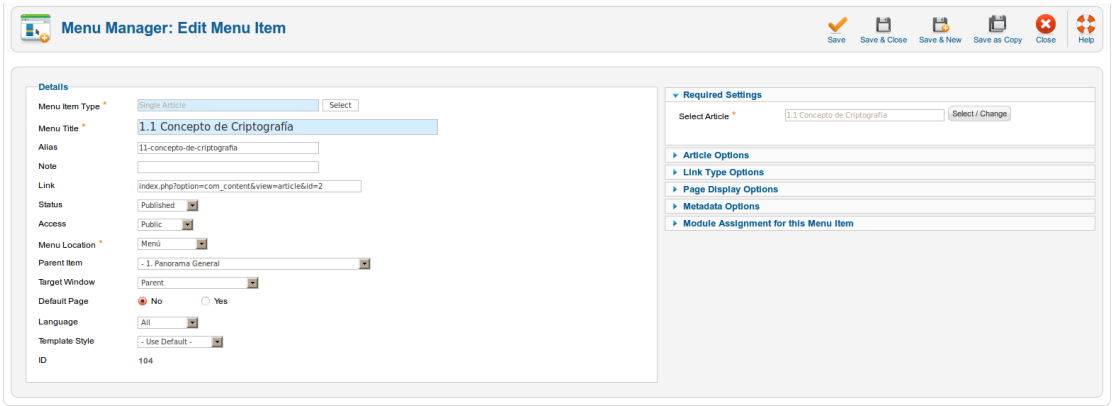


FIGURA 3.10 Cada artículo es asociado a un elemento del menú

- **Extensiones**

Las extensiones amplían las posibilidades de Joomla, existen diversos tipos:

- **Componentes**

Se pueden entender como una capa operativa dentro de la estructura de Joomla, operan sobre el núcleo, acceden a los contenidos y los interpretan para finalmente presentarlos y responder a la interacción del usuario, generalmente cada componente está vinculado a un tipo de información u operación concreta por ejemplo gestionar artículos, verificar las credenciales de los usuarios o realizar encuestas.

- **Módulos**

Son vínculos para mostrar en las páginas otros contenidos además del componente, cada elemento que se muestra en una página es el resultado de una "instancia" de un tipo de módulo, dicha instancia controla si un elemento se muestra o no, cuál será su contenido y en que páginas se muestra así como la posición establecida en la plantilla.

- **Plugins**

Son fragmentos de código que se ejecutan al producirse ciertos eventos en el sistema, en general ofrecen funcionalidades que no son exclusivas de un tipo de información, sino que pueden aplicarse a distintos tipos de datos, o bien amplían las posibilidades del núcleo de Joomla, ejemplos de plugins son el de autenticación, editores y búsqueda.

- **Idiomas**

Se trata de traducciones a prácticamente todos los idiomas del núcleo de Joomla y de las extensiones incluidas en él.

- **Plantillas**

Son las responsables del aspecto estético del sitio, la disposición de los elementos deben quedar plasmados en ella así como los colores, tipo de letra, etc. Las plantillas del frontend se instalan dentro de la subcarpeta *templates* (figura 3.11).

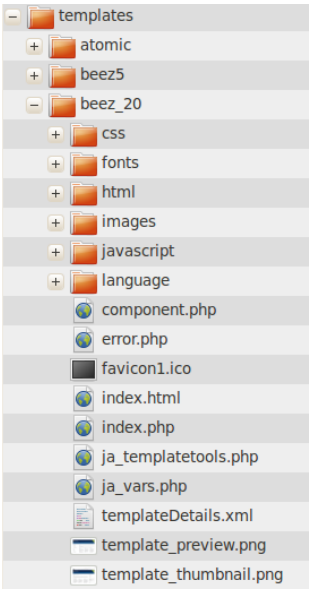


FIGURA 3.11 Estructura de subcarpetas del directorio "templates"

Algunos de los elementos importantes de esta carpeta son los siguientes:

- **CSS**: contiene archivos de hojas de estilo, dentro de esta carpeta se encuentra el archivo template.css que establece el aspecto de la plantilla.
- **Javascript**: contiene archivos de código Javascript.
- **templateDetails.xml**: contiene toda la información referente a la plantilla tales como nombre, referencias a todos los archivos de los que hace uso, posiciones y parámetros.
- **index.php**: establece la estructura jerárquica de la plantilla, es decir, configura los contenedores en los que se mostrarán las visas de componentes y módulos.

Cabe señalar que muchas de las extensiones de Joomla no son de un sólo tipo, sino que pueden incluir la combinación de varios de estos tipos, también existen extensiones que se basan en otras extensiones.

Algunas de las extensiones ocupadas en el sitio son las siguientes:

Extensiones

Nombre	Tipo	Versión	Descripción
beez_20	Plantilla	1.7.0	Plantilla contenida en Joomla y adaptada a las necesidades del sitio Web desarrollado.
Content-RokBox	Plugin	1.2	Despliegue de ventanas modales para imágenes y animaciones.
Maxi Menu CK	Módulo	5.24	Menú superior.
Content-Pagebreak	Plugin	1.7.0	Paginación de contenidos.

CAPÍTULO 4

Pruebas e Implementación

En este capítulo se expone el proceso de pruebas e implementación.

4.1 PROCESO DE PRUEBAS

El proceso de pruebas consistió en los tres pasos descritos a continuación e ilustrados en la figura 4.1:

1. Se incluyen pruebas de bajo nivel que verifiquen que cada sección se ha implementado correctamente y funciona adecuadamente como unidad. Dentro del sitio Web de fundamentos de Criptografía se puede identificar como una sección a cada capítulo y otras áreas como el contacto, glosario, etc.

   La mayoría de dichas secciones cuentan con las siguientes funcionalidades:

   - Menús y carga de contenidos
   - Paginación
   - Ventanas Modales
   - Animaciones

2. Se integran todas las secciones y se verifica trabajen conjuntamente. En esta fase se fueron integrando todos los temas, subtemas y demás secciones y se verificó su funcionamiento y adecuada integración con lo ya existente.
3. El software se combina con otros elementos del sistema como hardware, usuarios, bases de datos y se verifica que cada elemento trabaja en forma adecuada y que se alcanza la funcionalidad y el rendimiento del sistema en su totalidad.

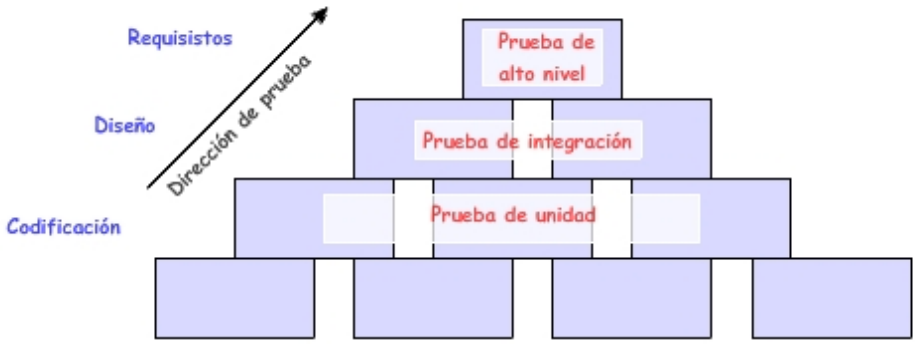


FIGURA 4.1 Pasos de la prueba del sitio

El proceso de pruebas de bajo nivel se muestra en la figura 4.2.

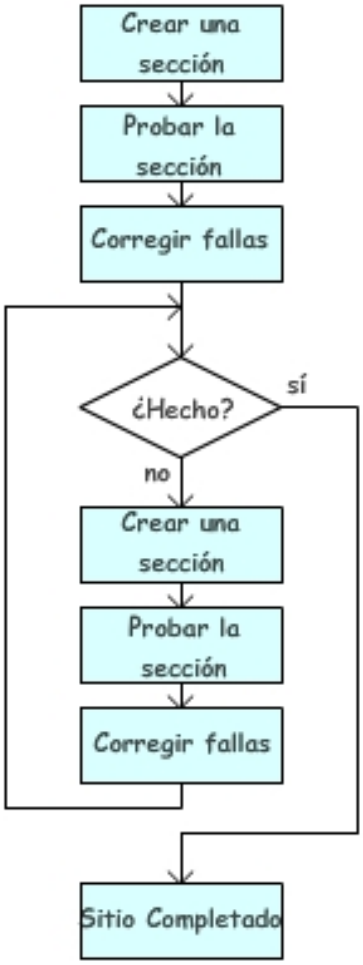


FIGURA 4.2 Proceso de pruebas

A cada sección desarrollada se aplican pruebas unitarias que consisten en reproducir el funcionamiento y verificar que el resultado sea el esperado, en caso de existir algún error se corrigen las fallas para posteriormente dar lugar a la creación de otra sección.

4.2 IMPLEMENTACIÓN

4.2.1 CARACTERÍSTICAS DE HARDWARE

Servidor HP ProLiant DL160 G6 Server

QuickSpecs	
See detailed specs	US QuickSpecs » html » pdf
Processor	
Processor family	Intel® Xeon® 5500 series Intel® Xeon® 5600 series
Number of processors	1 or 2
Processor core available	4 or 6
Memory	
Maximum memory	192 GB
Memory slots	18 DIMM slots
Memory type	DDR3 RDIMM or UDIMM
I/O	
Expansion slots	2
Network Controller	(1) 1GbE NC362i 2 Ports
Storage	
Maximum drive bays	(4) LFF SAS/SATA/SSD or (8) SFF SAS/SATA/SSD
Supported drives	Hot plug 2.5-inch SAS Hot plug 2.5-inch SATA Hot plug 3.5-inch SAS Hot plug 3.5-inch SATA Non-hot plug 3.5-inch SATA
Storage Controller	(1) Smart Array B110i SATA RAID (1) Smart Array P410/256MB BBWC

4.2.2 CARACTERÍSTICAS Y PROCEDIMIENTOS DE SOFTWARE

El sistema fue desarrollado en un equipo con las siguientes características de software:

Software	Versión
PHP	5.3.2
MySQL	5.1.41
Apache (con mod_mysql, mod_xml,y mod_zlib)	2.2.14
Sistema Operativo	Ubuntu

FIGURA 4.3 Sitio Web desplegado localmente

Tanto el desarrollo del sitio Web como las pruebas de bajo nivel y de integración se realizaron de manera local para después implementarlo en el servidor de producción (servidor del laboratorio de redes y seguridad).

Características de software (para Joomla 2.5.0) del servidor de producción:

Software	Versión
PHP	5.2.4 +
MySQL	5.0.4 +
Apache (con mod_mysql, mod_xml,y mod_zlib)	2.x +

La implementación del sitio en el servidor de producción consistió en:

a) Copiar la carpeta del sitio en el servidor Web.
b) Creación de la base de datos.
c) Modificación del archivo configuration.php para colocar datos correspondientes al servidor de base de datos tales como el usuario y password.
d) Realización de pruebas de alto nivel finales.

El sitio está disponible a través de la siguiente dirección electrónica:

http://redyseguridad.fi-p.unam.mx/proyectos/criptografia/criptografia

CONCLUSIONES

CONCLUSIONES

El desarrollo del presente trabajo culminó en una herramienta de apoyo para el aprendizaje de los fundamentos de la Criptografía. Será útil tanto para profesores como para alumnos ya que gracias a la disponibilidad de la información en un sitio Web da lugar a experimentar con otras formas de estudio y nuevas actividades dentro del aula.

El desarrollo de la herramienta ha sido un esfuerzo conjunto; por un lado la investigación de la información y construcción del sitio Web por mi parte y la revisión exhaustiva de dicha información por parte de la directora de la presente tesis, dando como resultado un material con información confiable y veraz.

Cabe señalar que toda la información contenida en el sitio Web es la base para el conocimiento de una ciencia tan amplia como es la Criptografía, el presente trabajo, se concentró en la realización de investigación bibliográfica y en plasmar los temas en un sitio Web con la explicación correspondiente y ejercicios prácticos en algunos casos. De este modo, si el usuario necesita o desea adentrarse profundamente en el conocimiento de la materia el conocimiento adquirido a través de este sitio le servirá de base para profundizar en el tema a través de otras bibliografías.

Se ha logrado tener una herramienta que para su subsistencia a largo plazo necesita ser actualizada y dársele el uso adecuado; una de sus mayores cualidades es que tiene posibilidades infinitas de crecimiento y puede ser mejorada de acuerdo a las necesidades que se tengan en un momento determinado, de tal modo que con el surgimiento de nuevas tecnologías; la imaginación y la disponibilidad de recursos serán los únicos límites para hacer de éste un proyecto en el cuál pueda colaborar toda la comunidad interesada, ya que es importante señalar que aunque en primera instancia está dirigido a alumnos de la Facultad de Ingeniería estará disponible para toda persona interesada en el tema en cualquier parte del mundo gracias a su disponibilidad vía Internet.

En el aspecto profesional, puedo decir que este proyecto me ha permitido poner en práctica muchos de los conocimientos adquiridos a lo largo de mis estudios en la licenciatura, desde los fundamentos de matemáticas; materias cursadas en la división de ciencias básicas (conocimientos que me permitieron comprender mejor las herramientas utilizadas en los algoritmos de cifrado) hasta los relacionados con las materias propiamente de computación como sistemas operativos, bases de datos, ingeniería de software, arquitectura de computadoras, entre otras, (conocimientos que en conjunto permitieron el desarrollo del software) sin olvidar por supuesto las pertenecientes a las de ciencias y humanidades ya que siempre se tuvo en mente que con la realización del presente trabajo se brindaría un servicio a la comunidad.

Es satisfactorio saber que el desarrollo del sistema tutorial de fundamentos de Criptografía no es un trabajo que al pasar los años sea sólo una propuesta o el desarrollo de una aplicación ficticia, mi satisfacción más grande es que este trabajo es una aplicación real y será de utilidad para toda la comunidad interesada, por lo que puedo decir sin temor a equivocarme que los objetivos planteados al inicio del desarrollo del presente proyecto se han cumplido.

ANEXOS

GLOSARIO

AGENTE:
Persona o proceso que desea acceder a la información de un sistema.

ALGORITMO:
Secuencia finita y ordenada de instrucciones elementales que, dados los valores de entrada de un problema devuelve la solución.

ALGORITMO CRIPTOGRÁFICO:
Un algoritmo define la forma en cómo los datos son transformados de texto en claro a texto cifrado. Tanto el emisor como el receptor deben conocer el algoritmo utilizado para la transformación de los datos, por lo que el mismo algoritmo debe ser utilizado para descifrar el texto cifrado.

AMENAZA:
Todo aquello que intenta o pretende violar la seguridad o causar daño a los recursos del sistema.

AMENAZA EXTERNA:
Códigos y usuarios maliciosos que intentan o pretenden violar la seguridad, así como intrusiones no autorizadas, spam, espionaje industrial, ataques DoS.

AMENAZA INTERNA:
Empleados internos, socios y usuarios de confianza de una organización que intentan o pretenden violar la seguridad, estas personas están familiarizadas con la red, saben que sistemas contienen la información valiosa y pueden tener acceso a los sistemas a través de su propia cuenta o mediante la cuenta de otro usuario.

AMENAZA NATURAL:
Fenómenos que no se pueden controlar y que pueden ocasionar la destrucción de la información de una organización tales como terremotos, inundaciones, tormentas, etc.

ANÁLISIS DE RIESGO:
Evaluación de amenazas y vulnerabilidades de la información y su impacto en el procesamiento de la información así como su probabilidad de ocurrencia.

ANÁLISIS DE TRÁFICO:
Inferencia de información a partir de la observación de flujos de tráfico (presencia, ausencia, cantidad, dirección y frecuencia).

ANSI (American National Standards Institute):

Organización fundada en 1918 dedicada a crear especificaciones para la industria de las computadoras con el fin de que se produzcan productos que sean interoperables, la ANSI está formada por más de 1300 miembros entre ellos todas las grandes empresas de computadoras.

ARQUITECTURA DE SEGURIDAD:
Descripción de alto nivel de la estructura de un sistema, con funciones de seguridad asignadas a los componentes de dicha arquitectura.

ATAQUE:
Es la realización de una amenaza.

ATAQUE ACTIVO:
Ataque que implica algún tipo de modificación del flujo de datos o la creación de un falso flujo de los mismos.

ATAQUE PASIVO:
Tipo de ataque que no altera en ningún momento la información, es decir, únicamente la observa, escucha, obtiene o monitorea mientras está siendo transmitida.

AUDITORÍA DE SEGURIDAD:
Revisión y examen independiente de los registros y actividades de un sistema para comprobar la adecuación de los controles, asegurar el cumplimiento de la políticas y procedimientos operacionales establecidos, detectar brechas en la seguridad y recomendar cambios en el control, la política y los procedimientos.

AUTENTICACIÓN:
Proceso de verificar la supuesta identidad de un principal.

CA (Certification Authority):
La Autoridad de Certificación también llamada Entidad Emisora de Certificados, es una tercera entidad de confianza que crea, asigna y distribuye certificados a usuarios, equipos, servicios o dispositivos de red.

CERTIFICADO:
Representación digital del usuario, equipo, servicio o dispositivo de red; consiste en un registro de datos que proporciona la clave pública del principal, junto con alguna información adicional relacionada con el nombre del principal y la autoridad de certificación que ha emitido el certificado.

CIFRAR:
Enmascarar una determinada información de carácter confidencial.

CLAVE:

Secuencia de símbolos que controla las operaciones de cifrado y descifrado. Es utilizada como una entrada para el algoritmo criptográfico, junto con los datos en texto plano, para que el algoritmo pueda convertir los datos en texto cifrado o bien, la clave y el texto cifrado son la entrada al algoritmo para que el texto cifrado sea convertido a texto en claro.

CLR (Lista de Revocación de Certificados):
Se trata de una lista de certificados que son revocados por la CA. La fecha de revocación y la razón son registradas en dicha lista.

CONFIDENCIALIDAD:
Propiedad de que la información no sea revelada ni puesta a disposición de las partes no autorizadas.

CONTROL DE ACCESO:
Proceso de evitar el uso no autorizado de recursos.

CRIPTOANÁLISIS:
Es la ciencia que se ocupa del análisis no autorizado de un texto cifrado para obtener la información original sin conocimiento de la clave secreta, provocando de este modo una ruptura o derrota a la criptografía.

CRIPTOANALISTA:
Persona no autorizada que intenta conocer la clave o el mensaje original utilizando para ello el criptoanálisis.

CRIPTOGRAFÍA:
Es la ciencia que se encarga del estudio de técnicas para transformar la información a una forma que no pueda entenderse a simple vista. El objetivo de la criptografía es triple; mantener los datos secretos, proteger los datos contra modificación y comprobar la fuente de los datos.

CRIPTÓGRAFO:
Máquina o dispositivo utilizado para cifrar.

CRIPTOGRAMA:
Texto o mensaje cifrado.

CRIPTOLOGÍA:
Es la ciencia que trata las escrituras ocultas, está comprendida por la criptografía, el criptoanálisis y la esteganografía.

CRIPTÓLOGO:
Persona que crea de forma legítima algoritmos criptográficos para proteger la información.

DES (Data Encryption Standard):
Algoritmo de cifrado simétrico, utiliza una clave de 56 bits generada de forma aleatoria.

DESCIFRAR:
Obtener el texto en claro a partir del criptograma utilizando la clave.

DESCRIPTAR:
Recuperar el mensaje original a partir del criptograma y sin conocimiento de la clave.

DIGRÁMICO:
El mensaje en claro se cifra tomando sus caracteres por parejas, en lugar de carácter a carácter.

DISPONIBILIDAD:
Propiedad de ser accesible y utilizable bajo demanda por una entidad autorizada.

DNS (Domain Name System):
Es un conjunto de protocolos y servicios (base de datos distribuida) que permite a los usuarios utilizar nombres en vez de tener que recordar direcciones IP numéricas.

DSA (Digital Signature Algorithm): Este algoritmo es utilizado para firmar datos, no para cifrar. El proceso de firma de DSA es realizado a través de una serie de cálculos basados en un número primo seleccionado. Aunque está orientado a tener un tamaño de clave máximo de 1024 bits, ya son soportados tamaños de clave más grandes.

ESTÁNDAR:
Acuerdo documentado que contiene especificaciones técnicas u otros criterios precisos para ser utilizados, contiene reglas, guías o definiciones de características para asegurar que los materiales, productos, procesos y servicios son adecuados para su propósito.

GESTIÓN DE CLAVES:
Generación, almacenamiento, distribución, borrado, actualización, archivado y aplicación de claves en una red de acuerdo a una política de seguridad.

INFORMACIÓN:
Todo mensaje (conjunto de datos) que al receptor le interese, le entienda o lo ignore antes de recibirlo.

INTEGRIDAD:

Propiedad de asegurar que los datos se transmiten desde una fuente a un destino sin alteraciones no detectadas.

INTRUSO:
Aquel que puede realizar cualquier conjunto de acciones que puede comprometer la integridad, confidencialidad o disponibilidad de la información o un recurso informático.

ISO (International Organization For Standardization):
La Organización Internacional para la Estandarización es una federación a nivel mundial no gubernamental establecida en 1947, con el objetivo de proporcionar el desarrollo de la estandarización y sus actividades relacionadas en todo el mundo en vista a facilitar el intercambio internacional de bienes y servicios.

MD5 (Message Digest 5):
Este algoritmo tomo un mensaje de cualquier longitud y produce un *message digest* (ver hash) de 128 bits.

MECANISMO DE SEGURIDAD:
Conjunto de elementos o procesos que implementan un servicio de seguridad, es decir, es aquel mecanismo que está diseñado para detectar, prevenir o recobrarse de un ataque de seguridad.

NO RECHAZO:
Propiedad de un receptor de ser capaz de probar que el remitente de algunos datos ha enviado efectivamente los datos.

PERPETRADOR:
Es un individuo que se basa en cualquier medio para cometer un delito o culpa grave.

POLÍTICA DE SEGURIDAD:
Conjunto de leyes, reglas y prácticas que regulan la manera de dirigir, proteger y distribuir recursos en una organización para llevar a cabo los objetivos de seguridad.

PRINCIPAL:
Persona o sistema registrada y autentificable para una red de computadoras o sistema distribuido.

PROTOCOLO CRIPTOGRÁFICO:
Serie de pasos que las entidades involucradas en una comunicación tienen que realizar para lograr los objetivos de seguridad.

RECHAZO:
Negación por parte de alguna de las entidades participantes en una comunicación de haber participado totalmente o en parte.

RELLENO DE TRÁFICO:
Generación de realizaciones espurias de comunicaciones, de unidades de datos o de datos dentro de dichas unidades.

RIESGO:
Posibilidad de sufrir algún daño o pérdida.

SEGURIDAD:
Confianza, tranquilidad, certidumbre procedente de la idea de que no hay peligro que temer.

SEGURIDAD DE LA INFORMACIÓN:
Se refiere a la prevención y a la protección, a través de ciertos mecanismos, para evitar que ocurra de manera accidental o intencional, la transferencia, modificación, difusión o destrucción no autorizada de la información.

SEGURIDAD INFORMÁTICA:
Nombre genérico dado a una colección de herramientas diseñadas para proteger datos y detener a los perpetradores.

SERVICIO DE SEGURIDAD:
Es aquél que está dirigido a evitar ataques de seguridad desde un aspecto particular buscando la seguridad de un sistema de información y el flujo de la información de una organización.

SSL (Secure Sockets Layer):
Protocolo desarrollado por Netscape en 1994 para garantizar la confidencialidad, la autenticación y la integridad de los mensajes en el intercambio de datos entre un navegador y un servidor Web.

TEXTO EN CLARO:
También llamado texto plano; es aquél que se desea cifrar, por tanto es la entrada a una función de cifrado o salida de una función de descifrado.

VULNERABILIDAD:
Debilidad que puede ser explotada para violar un sistema o la información que contiene.

TABLAS DES

Número de iteración	Número de desplazamientos a la izquierda
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Tabla PC-1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Tabla PC-2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Tabla IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tabla E de selección de bit					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

		Número de columna															
		0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***11***	***12***	***13***	***14***	***15***
Número de renglón	***0***	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Tabla S1

Tabla S1															
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Tabla S2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Tabla S3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Tabla S4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Tabla S5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

Tabla S6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

Tabla S7															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

Tabla S8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Tabla S			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Tabla IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

TABLAS AES

Tamaños de claves

- K = 128 bits
- K = 192 bits
- K = 256 bits

Independientemente del tamaño de la clave K *las matrices siempre serán del mismo tamaño:*

- Matrices de 4 x 4
- Cada elemento de la matriz es de 2 dígitos Hex.
- Mcla siempre será procesado en bloques de 128 bits; pero siempre manejados en las matrices en Hex.

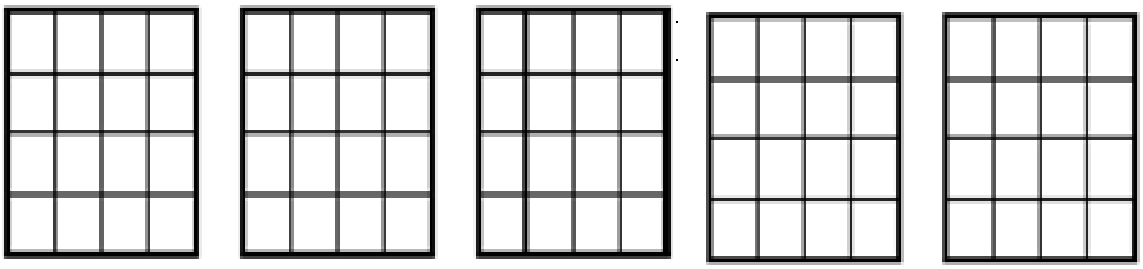
	Key Length (Nk words)	Block Size (Nb words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Nk - Núm. de palabras de 32 bits

Nb - Núm. de columnas ✓✓✓

Nr - Núm. de iteraciones o rondas, según el tamaño de K

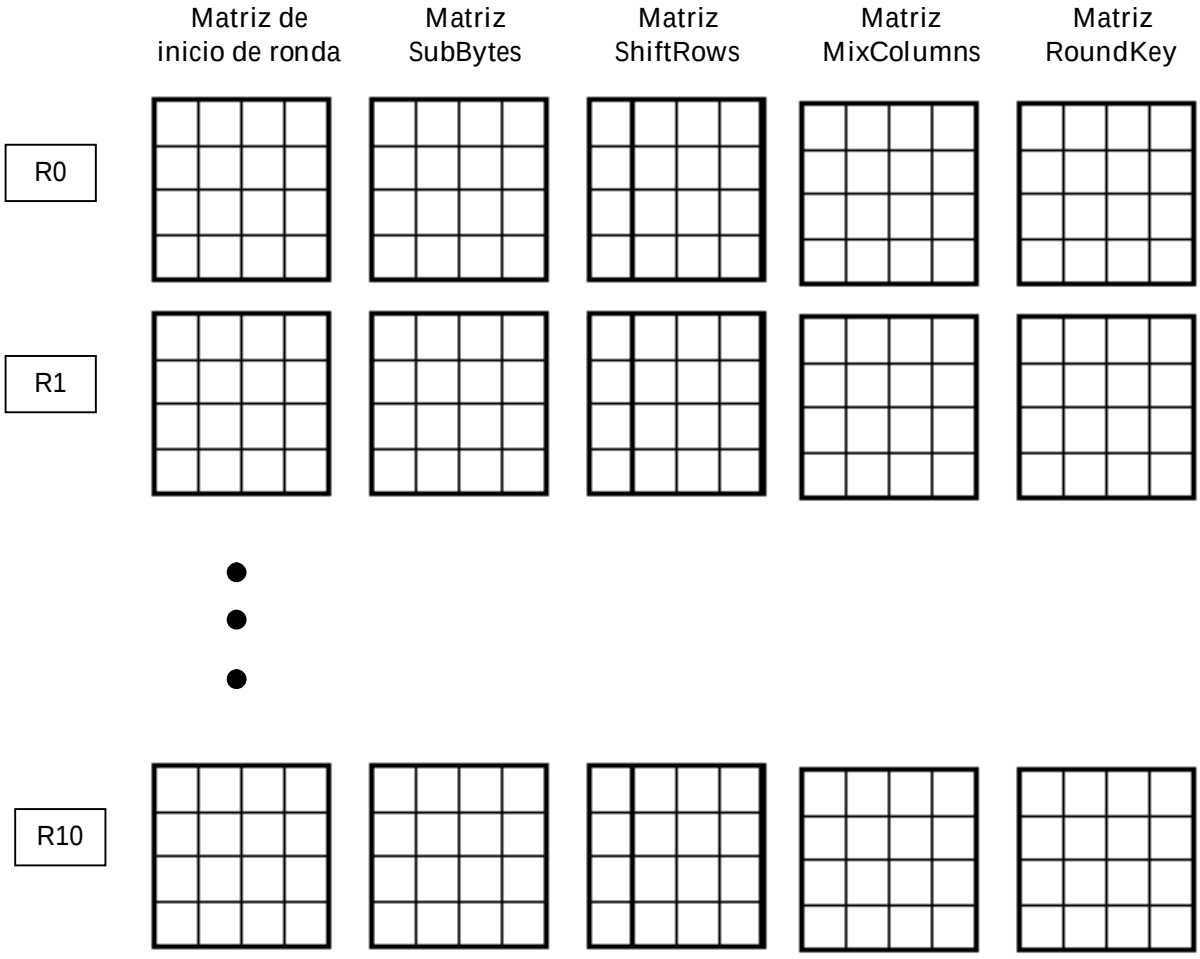
Matriz de inicio de ronda | Matriz SubBytes | Matriz ShiftRows | Matriz MixColumns | Matriz RoundKey



- Matriz de entrada o inicio de ronda
  Para la ronda R0 su contenido corresponde al Mcla, para las siguientes rondas será necesario calcular la matriz de entrada.
- Matriz SubBytes
  Sustituye individualmente cada byte del estado por otro de acuerdo a una tabla fija.
- Matriz ShiftRows
  Toma cada renglón del estado completo (Nb bytes) y hace un corrimiento cíclico un determinado número de bytes o columnas que depende del renglón del que se trate.
- Matriz MixColumns
  Opera idénticamente con cada columna completa (4 bytes) aplicando una transformación lineal.
- Matriz RoundKey
  Modifica el estado de la clave sumándole módulo 2 (XOR) byte a byte la clave de la ronda correspondiente.

EJEMPLO: considerando AES-128 ➔ Nb = 4 y Nk = 4

Mcla =	32	43	f6	a8	88	5a	30	8d	31	31	98	a2	e0	37	07	34
Ki =	2b	7e	15	16	28	ae	d2	a6	ab	f7	15	88	09	cf	4f	3c



Tablas a utilizar durante el proceso de cifrado AES

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 7. S-box: substitution values for the byte xy (in hexadecimal format).

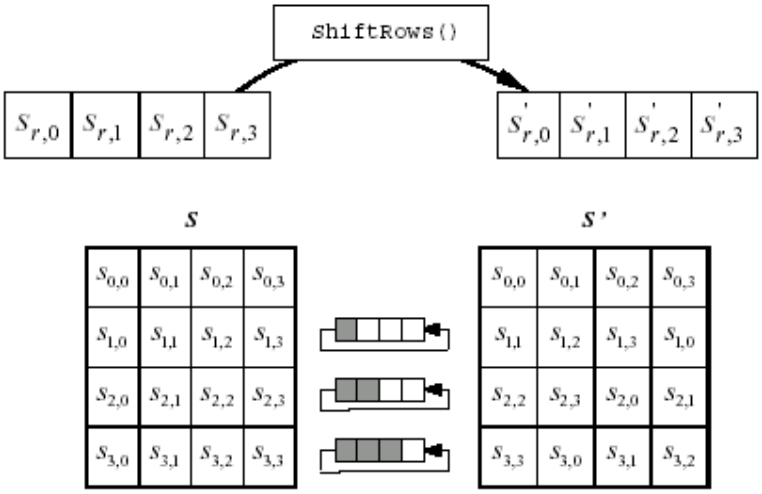


Figure 8. `ShiftRows()` cyclically shifts the last three rows in the State.

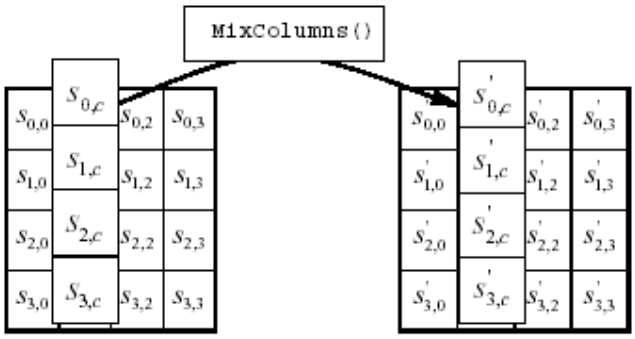


Figure 9. `MixColumns()` operates on the State column-by-column.

Multiplication Matrix

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

16 byte State

b1	b5	b9	b13
b2	b6	b10	b14
b3	b7	b11	b15
b4	b8	b12	b16

The first column will include state bytes 1-4 and will be multiplied against the matrix in the following manner:

```
b1 = (b1 * 2) XOR (b2*3) XOR (b3*1) XOR (b4*1)
b2 = (b1 * 1) XOR (b2*2) XOR (b3*3) XOR (b4*1)
b3 = (b1 * 1) XOR (b2*1) XOR (b3*2) XOR (b4*3)
b4 = (b1 * 3) XOR (b2*1) XOR (b3*1) XOR (b4*2)
```

(b1= specifies the first byte of the state)

The second column will be multiplied against the second row of the matrix in the following manner.

```
b5 = (b5 * 2) XOR (b6*3) XOR (b7*1) XOR (b8*1)
b6 = (b5 * 1) XOR (b6*2) XOR (b7*3) XOR (b8*1)
b7 = (b5 * 1) XOR (b6*1) XOR (b7*2) XOR (b8*3)
b8 = (b5 * 3) XOR (b6*1) XOR (b7*1) XOR (b8*2)
```

And so on until all columns of the state are exhausted.

E Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	03	05	0F	11	33	55	FF	1A	2E	72	96	A1	F8	13	35
1	5F	E1	38	48	D8	73	95	A4	F7	02	06	0A	1E	22	66	AA
2	E5	34	5C	E4	37	59	EB	26	6A	BE	D9	70	90	AB	E6	31
3	53	F5	04	0C	14	3C	44	CC	4F	D1	68	B8	D3	6E	B2	CD
4	4C	D4	67	A9	E0	3B	4D	D7	62	A6	F1	08	18	28	78	88
5	83	9E	B9	D0	6B	BD	DC	7F	81	98	B3	CE	49	DB	76	9A
6	B5	C4	57	F9	10	30	50	F0	0B	1D	27	69	BB	D6	61	A3
7	FE	19	2B	7D	87	92	AD	EC	2F	71	93	AE	E9	20	60	A0
8	FB	16	3A	4E	D2	6D	B7	C2	5D	E7	32	56	FA	15	3F	41
9	C3	5E	E2	3D	47	C9	40	C0	5B	ED	2C	74	9C	BF	DA	75
A	9F	BA	D5	64	AC	EF	2A	7E	82	9D	BC	DF	7A	8E	89	80
B	9B	B6	C1	58	E8	23	65	AF	EA	25	6F	B1	C8	43	C5	54
C	FC	1F	21	63	A5	F4	07	09	1B	2D	77	99	B0	CB	46	CA
D	45	CF	4A	DE	79	8B	86	91	A8	E3	3E	42	C6	51	F3	0E
E	12	36	5A	EE	29	7B	8D	8C	8F	8A	85	94	A7	F2	0D	17
F	39	4B	DD	7C	84	97	A2	FD	1C	24	6C	B4	C7	52	F6	01

L Table

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		00	19	01	32	02	1A	C6	4B	C7	1B	68	33	EE	DF	03
1	64	04	E0	0E	34	8D	81	EF	4C	71	08	C8	F8	69	1C	C1
2	7D	C2	1D	B5	F9	B9	27	6A	4D	E4	A6	72	9A	C9	09	78
3	65	2F	8A	05	21	0F	E1	24	12	F0	82	45	35	93	DA	8E
4	96	8F	DB	BD	36	D0	CE	94	13	5C	D2	F1	40	46	83	38
5	66	DD	FD	30	BF	06	8B	62	B3	25	E2	98	22	88	91	10
6	7E	6E	48	C3	A3	B6	1E	42	3A	6B	28	54	FA	85	3D	BA
7	2B	79	0A	15	9B	9F	5E	CA	4E	D4	AC	E5	F3	73	A7	57
8	AF	58	A8	50	F4	EA	D6	74	4F	AE	E9	D5	E7	E6	AD	E8
9	2C	D7	75	7A	EB	16	0B	F5	59	CB	5F	B0	9C	A9	51	A0
A	7F	0C	F6	6F	17	C4	49	EC	D8	43	1F	2D	A4	76	7B	B7
B	CC	BB	3E	5A	FB	60	B1	86	3B	52	A1	6C	AA	55	29	9D
C	97	B2	87	90	61	BE	DC	FC	BC	95	CF	CD	37	3F	5B	D1
D	53	39	84	3C	41	A2	6D	47	14	2A	9E	5D	56	F2	D3	AB
E	44	11	92	D9	23	20	2E	89	B4	7C	B8	26	77	99	E3	A5
F	67	4A	ED	DE	C5	31	FE	18	0D	63	8C	80	C0	F7	70	07

Para el proceso de descifrado

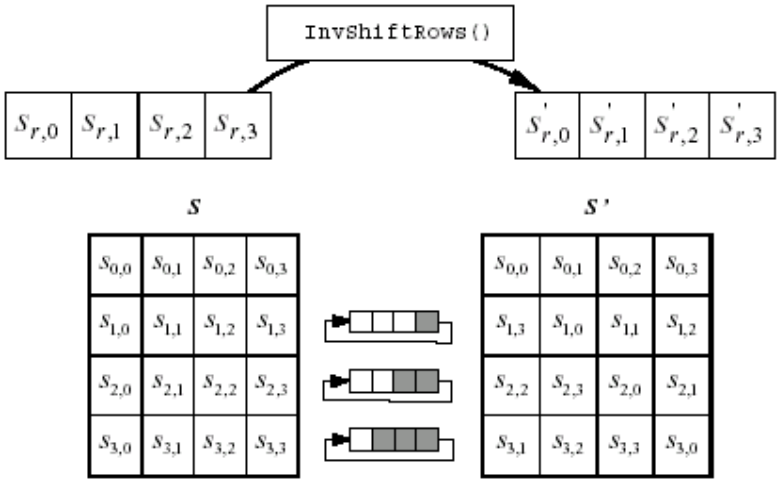


Figure 13. `InvShiftRows()` cyclically shifts the last three rows in the State.

5.3.2 `InvSubBytes()` Transformation

`InvSubBytes()` is the inverse of the byte substitution transformation, in which the inverse S-box is applied to each byte of the State. This is obtained by applying the inverse of the affine transformation (5.1) followed by taking the multiplicative inverse in GF(2^8).

The inverse S-box used in the `InvSubBytes()` transformation is presented in Fig. 14:

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Figure 14. Inverse S-box: substitution values for the byte xy (in hexadecimal format).

5.3.3 `InvMixColumns()` Transformation

`InvMixColumns()` is the inverse of the `MixColumns()` transformation. `InvMixColumns()` operates on the State column-by-column, treating each column as a four-term polynomial as described in Sec. 4.3. The columns are considered as polynomials over $GF(2^8)$ and multiplied modulo $x^4 + 1$ with a fixed polynomial $a^{-1}(x)$, given by

MATRIZ FIJA

$$\begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix}$$

BIBLIOGRAFÍA

BIBLIOGRAFÍA

LIBROS

[1] Black, Uyless D., *Redes de computadoras: Protocolos, normas e interfaces*, México D.F., Macrobit Editores S.A de C.V., 1990, 421 p.

[2] Caballero Gil, Pino, *Seguridad informática: técnicas criptográficas*, México, D.F., Alfaomega, 1997, 135 p.

[3] Daltabuit, Enrique, *La seguridad de la información*, México, Limusa, 2007, 774 p.

[4] Fuster Sabater, Amparo, et al., *Técnicas criptográficas de protección de datos*, México, D.F., 2ª Edición, Alfaomega: Ra-Ma, 2001, 372 p.

[5] Gómez Vieites, Álvaro, *Enciclopedia de la seguridad informática*, México, Alfaomega, 2007, 664 p.

[6] Gratton, Pierre, *Protección informática: en datos y programas; en gestión y operación; en equipos y redes; en Internet*, México, Trillas, 1998, 272 p.

[7] Howlett, Tony, *Software libre: herramientas de seguridad*, Madrid, Anaya Multimedia, 2005, 656 p.

[8] López Barrientos María Jaquelina, *Criptografía*, México, 1ª Edición, UNAM, Facultad de Ingeniería, 2009, 275 p.

[9] López Barrientos María Jaquelina y Quezada Reyes Cintia, *Fundamentos de seguridad informática*, México, 1ª Edición, UNAM, Facultad de Ingeniería, 2006, 223 p.

[10] McIntire, Penny, *Técnicas innovadoras en Diseño Web*, Madrid, Anaya Multimedia, 2009, 320 p.

[11] Menezes, Alfred J., et al., *Handbook of Applied Cryptography*, Boca Raton, Florida, CRC, 1997, 780 p.

[12] Nash, Andrew, et al., *PKI - Infraestructura de claves públicas: la mejor tecnología para implementar y administrar la seguridad electrónica de su negocio*, Colombia, Osborne McGraw-Hill, 2002, 512 p.

[13] Oppliger, Rolf, *Sistemas de autentificación para seguridad en redes*, Bogotá, Alfaomega, 1998, 194 p.

[14] Pastor Franco, José, et al., *Criptografía digital : fundamentos y aplicaciones*, Zaragoza, España, Prensas Universitarias de Zaragoza, 1998, 597 p.

[15] Rodríguez Prieto, Amador, *Protección de la información: Diseño de criptosistemas informáticos*, Madrid, Paraninfo, 1986, 255 p.

[16] Schmidt, Jeff , *Guía avanzada: seguridad en Microsoft Windows 2000*, España, Prentice Hall, 2001, 802 p.

[17] Sommerville, Ian, *Ingeniería del software*, Madrid, 7ª Edición, Pearson Educación, S.A., 2005, 687 p.

[18] Stallings, William, *Comunicaciones y redes de computadores*, Madrid, 6ª Edición, Prentice Hall, 2000, 747 p.

LIBROS ELÉCTRONICOS

[18] Lucena López, Manuel J., *Criptografía y Seguridad en Computadores*
[en línea], disponible en:
http://wwwdi.ujaen.es/~mlucena/wiki/pmwiki.php?n=Main.LCripto
recuperado: enero de 2012, 307 p.

[19] Ramió Aguirre, Jorge, *Seguridad Informática y Criptografía*
[en línea], disponible en:
http://www.criptored.upm.es/guiateoria/gt_m001a.htm
recuperado: enero de 2012, 1106 p.

[20] S.a., Criptosistemas Clásicos. In S.a., *Aplicaciones criptográficas*, Escuela Universitaria de Informática de la Universidad Politécnica de Madrid España
[en línea], disponible en:
http://www.criptored.upm.es/descarga/CriptoClasica.zip
recuperado: enero de 2012, 104 p.

TESIS

[21] Silva Sarabia, Christopher Román, *Criptografía y curvas elípticas*, Tesis Licenciatura (Matemático), Universidad Nacional Autónoma de México, Facultad de Ciencias, 2006, 183 p.

[22] Zuñiga González, María Guadalupe, *Introducción histórica a la criptografía*, Tesis Licenciatura (Ingeniero en Computación), Universidad Nacional Autónoma de México, Facultad de Estudios Superiores: Aragón, 2001, 157 p.

REFERENCIAS DE INTERNET

[23] Características del CMS WordPress
http://codex.wordpress.org/WordPress_Features
[en línea], enero de 2012

[24] Características del CMS Drupal
http://drupal.org.es/caracteristicas
[en línea], enero de 2012

[25] Características del CMS Joomla!
http://www.joomla.org/core-features.html
[en línea], enero de 2012

[26] Información sobre tendencias más significativas para el uso de Internet
http://marketshare.hitslink.com/
[en línea], enero de 2012

[27] Recomendaciones en el uso de CMS
http://recursosweb.unam.mx/recomendaciones-en-el-uso-de-cms-administradores-de-contenido/
[en línea], enero de 2012

[28] Lineamientos CATIC (Consejo Asesor en Tecnologías de Información y Comunicación, UNAM) para páginas institucionales de la UNAM
http://recursosweb.unam.mx/recursos-web/lineamientos-unam/
[en línea], enero de 2012