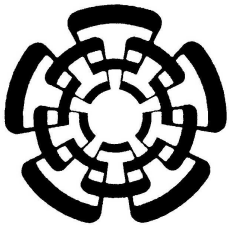
CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL



UNIDAD ZACATENCO
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE COMPUTACIÓN

Estudio, diseño y evaluación de protocolos de autenticación para redes Inalámbricas

---

Tesis que presenta
Laura Itzelt Reyes Montiel

Para obtener el grado de
Maestra en Ciencias

En la especialidad de
Ingeniería Eléctrica

Opción: Computación

Director de Tesis:
Francisco José Rambó Rodríguez Henríquez

Ciudad de México, D.F. Noviembre 2003

Agradecimientos

Agradezco al Consejo Nacional de Ciencia y Tecnología por el respaldo financiero otorgado por medio de la beca crédito nivel maestría. Así como el apoyo brindado para la conclusión de este trabajo de Tesis por medio del proyecto CONACyT .

Agradezco a los profesores de la sección de computación del CINVESTAV-IPN que en mayor o menor medida ampliaron mi visión sobre las Ciencias de la Computación.

Agradezco al Dr. Francisco José Rambó Rodríguez Henríquez la oportunidad que me dio al proponerme este tema de tesis; la experiencia al desarrollar el trabajo resultó muy nutritiva y gratificante.

En mayor medida agradezco a mis padres, Gregorio Reyes e Irma Montiel por su apoyo durante esta etapa de mi vida. A mis hermanos Alfredo, Gabriela, Adriana y Gregorio por el ánimo que me brindaron. A mi amiga Karina por hacer la vida en esta ciudad más fácil y ayudarme, como siempre, en esta etapa de mi vida.

A José Juan por su comprensión y apoyo incondicional, estuviera donde estuviera.

Índice general

Introducción

La globalización de las comunicaciones inalámbricas ha permitido el desarrollo de nuevos estándares y productos que están produciendo cambios en la vida diaria. La movilidad se ha vuelto un requerimiento cada vez mayor dentro de los ambientes de trabajo y gracias a las redes inalámbricas se ha obtenido una movilidad *real* en los dispositivos *móviles*. Debido a ello, actualmente se exige de las redes inalámbricas el mismo tipo de servicios que se brindan en las redes cableadas. Estándares inalámbricos emergentes tales como IEEE 802.11, IEEE 802.15, Bluetooth, HiperLAN/2, HomeRF en combinación con otras tecnologías no tan recientes como la telefonía celular aunado con nuevos protocolos como el WAP permitirán la interconexión de las redes actuales e Internet a dispositivos móviles como teléfonos celulares, asistentes personales digitales (*PDAs*), radiolocalizadores (*pagers*) de dos vías y otros dispositivos portátiles.

La navegación por Internet a través de los dispositivos inalámbricos, hace que el intercambio de información por este medio, incluyendo datos de alto valor, sea una práctica común para los usuarios de las redes inalámbricas, por lo que actualmente se ha puesto un especial énfasis a la seguridad en tales medios de comunicación.

Por otro lado, las redes inalámbricas de área local, donde el estándar 802.11 ha destacado, están ocupando un nicho muy importante en la industria. Este tipo de redes ofrecen una alternativa de bajo costo a los sistemas cableados, brindan capacidad de incorporar de manera fácil a nuevos usuarios a la red, y además, se tiene la posibilidad ubicua para acceder a cualquier base de datos o a cualquier aplicación localizada dentro de la red. Pero, en este tipo de redes, es posible captar los datos desde cualquier punto dentro del radio de alcance correspondiente con un receptor compatible.

En las redes inalámbricas el canal de comunicación es peculiarmente inseguro y en aplicaciones de comercio móvil o electrónico la inseguridad es una característica no deseada. Para tratar de atenuar este defecto, se deben poner en práctica servicios que garanticen la seguridad computacional, tales servicios son la confidencialidad, la integridad, la disponibilidad y la autenticación.

Un sistema posee la propiedad de *confidencialidad* si los recursos manipulados por éste no son puestos al descubierto por usuarios, entidades o procesos no autorizados. Un sistema posee la propiedad de *integridad* si los recursos manipulados por éste no son alterados o destruidos por usuarios, entidades o procesos no autorizados. Un sistema posee la propiedad

de *disponibilidad* si los recursos brindan servicio en el momento en que así lo deseen los usuarios, entidades o procesos autorizados. La *autenticación* es el proceso de verificar y asegurar la identidad de las partes involucradas en una transacción. Si este servicio no se llevara a cabo cabe la posibilidad de que una entidad desconocida asuma una identidad falsa, comprometiendo de esta manera la privacidad y la integridad de la información.

La seguridad es muy importante para que el desarrollo e implementación de las redes inalámbricas sean explotados de una manera eficaz y confiable; desafortunadamente, las características inherentes de las redes inalámbricas pueden ser un punto en contra de tal seguridad. Es por ello, que en este trabajo de tesis se aborda, desde diferentes perspectivas, uno de los servicios de la seguridad mencionado anteriormente: la autenticación. Para las redes inalámbricas de área local se estudia el estado del arte en este campo y para las redes inalámbricas de área amplia se plantea el estudio e implementación del protocolo donde se lleva al cabo la autenticación, el protocolo WTLS (*Wireless Transport Layer Security*).

Con el propósito de establecer una sesión segura el protocolo de Negociación del WTLS admite el uso de únicamente dos criptosistemas de llave pública: RSA y Criptosistemas de Curvas Elípticas (CCE). Tradicionalmente RSA ha sido la opción criptográfica más utilizada en las implementaciones de WTLS reportadas hasta la fecha. Ello a pesar que CCE nos brinda niveles de seguridad similares a los proporcionados por RSA con tamaños de llaves hasta 10 veces menores.

En esta trabajo de tesis se ha desarrollado un prototipo que simula la funcionalidad del protocolo de Negociacón de WTLS para evaluar el desempeño de cada uno de estos criptosistemas. El objetivo primordial de este trabajo es presentar un estudio cuantitativo del impacto que el uso de cada uno de los dos sistemas criptográficos mencionados provoca en el tiempo de ejecución del protocolo de Negociación. Con esa finalidad se elaboró un sistema para cada una de las partes involucradas en la comunicación, un sistema que corresponde al cliente y un sistema que corresponde al servidor. El cliente y el servidor del prototipo del protocolo WTLS se comunican, en esta ocasión, mediante sockets utilizando TCP/IP a diferencia de la especificación inalámbrica: WPD. Los resultados experimentales obtenidos en nuestras pruebas permiten corroborar que efectivamente CCE es una opción criptográfica más eficiente para la implementación de WTLS con tiempos de ejecución 2 y 5 veces más rápidos que RSA de 1024 y 2048 bits, respectivamente.

El resto de este documento de tesis está organizado como sigue: en el Capítulo 1 se describen los conceptos relacionados con la criptografía de llave pública, como los son: primitivas de cifrado/descifrado, firma/verificación, protocolos para el acuerdo de llaves, etc; necesarios para establecer las bases de este trabajo. En el Capítulo 2 se discuten los métodos de autenticación en general. El Capítulo 3, dedicado a redes inalámbricas, se define el término seguridad computacional y su relación con las redes inalámbricas. Asimismo, se describe brevemente la estructura general del protocolo WAP y su protocolo de seguridad, WTLS. Además se introduce el tema de seguridad dentro de las redes 802.11. Posteriormente, en el Capítulo 4 se revisan los elementos de la implementación del protocolo de negociación de WTLS. En el Capítulo 5 se incluye la evaluación y análisis de los hallazgos experimentales

obtenidos. Finalmente en el capítulo 6 se dan las conclusiones obtenidas en este trabajo.

Capítulo 1

Criptografía de Llave Pública

Hoy en día el intercambio de información valiosa por Internet, como números de tarjetas de crédito, es una práctica común. En general, en todo tipo de redes la información que se comunica de una entidad a otra está expuesta a la posibilidad de un ataque de un adversario, por lo que el proteger los datos intercambiados se ha vuelto una tarea prioritaria en todo ámbito. Es por ello que la premisa básica de cualquier comunicación y administración de datos confiable es cumplir con los principios de la seguridad computacional, que se resumen en los siguientes servicios [4, 2]:

- **Confidencialidad**. Los componentes del sistema serán accesibles sólo por aquellos usuarios autorizados.
- **Integridad**. Los componentes del sistema sólo pueden ser creados y modificados por los usuarios autorizados.
- **Disponibilidad**. Los usuarios deben tener disponibles todos los componentes del sistema cuando así lo deseen.
- **Autentificación**. La autentificación es el proceso de verificar y asegurar la identidad de las partes involucradas en una transacción.
- **No Repudio**. El no repudio está asociado a la aceptación de un protocolo de comunicación entre emisor y receptor (cliente y servidor) normalmente este proceso se lleva a cabo a través de la autentificación.

Si se cumplen estos 5 servicios se considera que los datos están protegidos y seguros [4]. La *criptografía* es la responsable directa de brindar sistemas criptográficos a los servicios de seguridad para llevar a cabo su tarea. Como se muestra en la figura 1.1, las aplicaciones de los algoritmos criptográficos son los servicios de seguridad mencionados.

En la siguiente sección se ofrece una breve introducción a la criptografía para enfocarse después en una de sus variantes, la criptografía de llave pública donde se estudian los

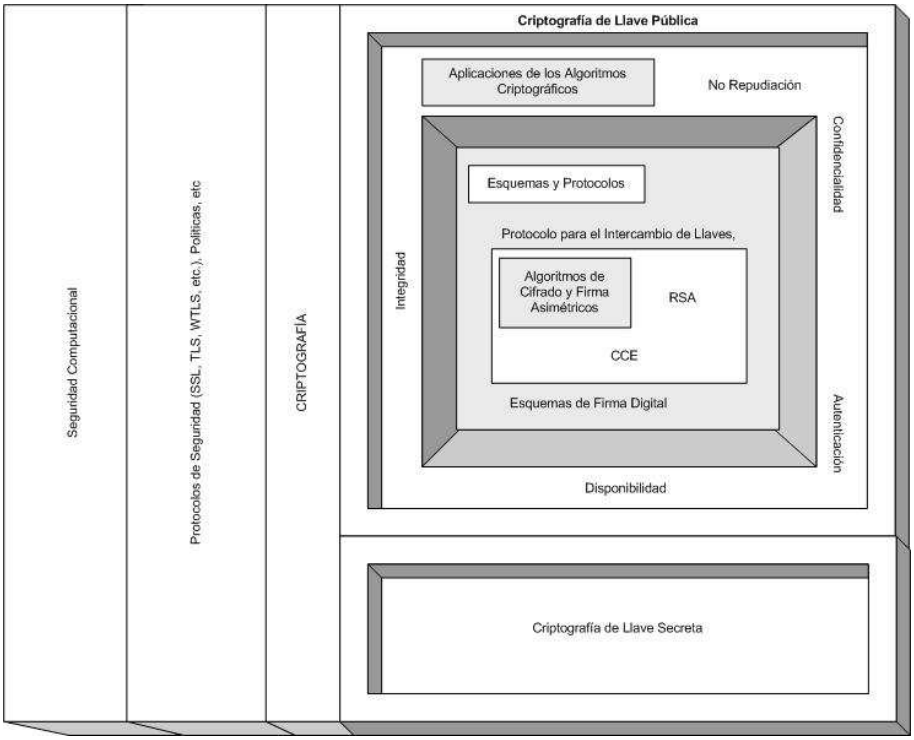


Figura 1.1: Relación de la Criptografía con la Seguridad Computacional

esquemas de Criptografía de Curvas Elípticas y RSA. En la sección consecutiva se provee una introducción a los protocolos necesarios para hacer segura la transmisión de datos: *los protocolos de intercambio de llave*; este tipo de protocolo es muy importantes para establecer una sesión segura y logran su propósito a través del uso de la criptografía de llave pública. En la última sección se muestran las primitivas de firma digital y las diferentes formas de generarlas. Todo esto como marco teórico de un concepto que se abordará en el siguiente capítulo, la autentificación.

1.1. Introducción

La **criptografía** es el proceso de diseñar sistemas basados en técnicas matemáticas para obtener una comunicación segura en canales que no lo son [35]. La criptografía da lugar a diferentes tipos de sistemas criptográficos que permiten asegurar los principios de la seguridad informática: confidencialidad, integridad, disponibilidad, autentificación y no repudio de emisor y receptor. Así pues, los protocolos de seguridad permiten llevar al cabo las metas de seguridad, pero la criptografía es la principal herramienta matemática de la que se sirven dichos protocolos para brindar los servicios necesarios que protegen tanto a los datos como a las entidades involucradas en la comunicación.

Figura 1.2: Modelo Convencional de Cifrado con Llave Simétrica

La criptografía contiene primitivas para llevar a cabo la seguridad de la información, tales primitivas son *cifrado/descifrado* y *firma /verificación*. Los *criptosistemas* pueden clasificarse, de manera general, en dos categorías: *criptografía de llave pública* y *criptografía de llave simétrica*.

Matemáticamente un **criptosistema** puede ser definido como la quíntupla $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, donde [15]:

$\mathcal{P}$ es el conjunto finito de los posibles textos en claro.

$\mathcal{C}$ es el conjunto finito de los posibles textos cifrados.

$\mathcal{K}$ el espacio de llaves, es un conjunto finito de todas las llaves posibles.

$\forall\, K \in \mathcal{K}\ \exists E_K \in \mathcal{E}$ (regla de cifrado), $\exists\, D_K \in \mathcal{D}$ (regal de descifrado).

Cada $E_K : \mathcal{P} \to \mathcal{C}$ y $D_K : \mathcal{C} \to \mathcal{P}$ son funciones tales que $\forall x \in \mathcal{P}, D_K(E_K(x)) = x$.

En los criptosistemas, el término **llave** se refiere a un valor numérico utilizado para alterar información haciendola segura y visible únicamente a los individuos que tienen la llave correspondiente para recuperar dicha información [1].

Los **algoritmos de llave simétrica** (o llave secreta) utilizan una llave con la cual es posible codificar y decodificar los mensajes comunicados entre dos o más partes. Existen muchos algoritmos capaces de realizar cifrado con llaves simétricas como lo son: el estándar

Figura 1.3: Modelo Convencional de Cifrado con Llave Pública

de cifrado de datos (DES), Rijndael (AES), IDEA, Blowfish, 3DES, Twofish, RC2, entre otros [2]. Estos distintos algoritmos tienen diferentes grados de seguridad de acuerdo al tamaño en bits de la llave. Las llaves utilizadas para cifrar y descifrar datos van desde 40 bits (64 para ser realmente consideradas seguras) hasta 256 bits. Este tipo de algoritmos tienen como ventaja la sencillez matemática de los problemas en que se fundamentan, lo que implica una alta velocidad de cifrado. Sin embargo, la principal debilidad de este tipo de esquemas es la administración de las llaves, es decir, la protección y distribución de las llaves.

El **cifrado con llave pública** o asimétrica fue introducido por Whitfield Diffie y Martin Hellman en el año 1976, en el artículo "*New directions in Cryptography*". [34, 35]. El algoritmo de cifrado Diffie-Hellman produjo una verdadera revolución en el campo de la criptografía, ya que fue el punto de partida para los sistema asimétricos o de llave pública. En este tipo de sistemas existen dos llaves distintas: una para cifrar y otra para descifrar. La llave para cifrar es conocida públicamente y, por ende, se denomina *llave pública*. La llave para descifrar sólo es conocida por el receptor del mensaje, por lo que se denomina *llave privada*. La ventaja de estos sistemas criptográficos es que la denominada llave pública puede ser usada por cualquier persona para cifrar mensajes bajo la premisa que sólo quien posea la llave privada podrá descifrar dichos mensajes. Aunque los métodos de llave pública son muy poderosos, tienen un costo computacional elevado: son por lo menos mil veces más lentos que los algoritmos simétricos [6].

Actualmente la mayoría de los protocolos de seguridad ocupan ambas técnicas criptográficas en diferentes etapas del proceso. La criptografía de llave simétrica se utiliza para cifrar grandes cantidades de datos y la criptografía de llave pública se prefiere para acordar

una *llave de sesión*[1] que será utilizada con el método de cifrado simétrico.

1.2. Algoritmos Asimétricos de Cifrado

La criptografía asimétrica se fundamenta en la teoría elemental de números. De manera general, un sistema criptográfico de llave pública se compone de una función $f(x)$ no lineal de sólo ida tal que dado x, calcular $y = f(x)$ sea trivial; pero que dado $y = f(x)$, calcular x sea un problema difícil.

Según el problema matemático en el cual base su seguridad, la criptografía asimétrica se clasifica en 2 grupos principales [17]:

Grupo 1: Problema de Factorización Entera.

El problema de factorización entera se define como:

"dado un entero positivo n, encuentre sus factores primos"

esto es, escriba $n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$, donde cada p_i es un primo y cada $e_i > 1$.

El sistema de llave pública RSA pertenece a esta categoría.

Grupo 2: Problema del Logaritmo Discreto. A partir de la función módulo discreto (mod), se tiene: dado un primo p, sean α, $\beta \in \mathbb{Z} \not\equiv 0 \bmod p$ y suponga $\beta \equiv \alpha^x \pmod{p}$. El problema de encontrar x se conoce como el problema del logaritmo discreto.

El sistema criptográfico de curvas elípticas se construye en una analogía de este problema. Así mismo, el estándar norteamericano de firma digital, DSA, basa su seguridad en este tipo de problema.

1.2.1. RSA

El algoritmo RSA debe su nombre a las iniciales de sus tres inventores: Ronald Rivest, Adi Shamir y Leonard Adleman. Este sistema de cifrado, como se menciona anteriormente, basa su seguridad en la conjetura matemática que sostiene que el problema de factorizar números enteros en sus factores primos tiene una complejidad computacional prohibitiva para el estado del arte de la tecnología de hoy en día y los tamaños en bits de los números utilizados.

Las llaves pública y privada $(K_{pub}; K_{priv})$ se calculan a partir de un número que se obtiene como producto de dos primos grandes. El proceso para generar dicho par de llaves es el siguiente:

1. Se elige de manera aleatoria dos números primos grandes, p y q.

[1] En la página 14 se introducirá el concepto de llave de sesión, su importancia y su funcionalidad.

2. Se calcula el producto $n = pq$.

3. Se elige ahora un número e primo relativo con $(p-1)(q-1)$.

4. La llave pública será (e, n). Nótese que e debe tener inversa módulo $(p-1)(q-1)$ para garantizar que existirá un número d tal que

$$de \equiv 1 \mathrm{mod}((p-1)(q-1))$$

es decir, que d es el inverso de e mod $(p-1)(q-1)$.

5. La llave privada será (d, n).

Por lo que

$$K_{pub} = (e, n)$$

y

$$K_{priv} = (d, n)$$

Para poder garantizar un margen de seguridad aceptable en transacciones comerciales electrónicas, diferentes estándares recomiendan usar RSA con pares de llaves públicas y privadas con un tamaño no menor a 1024 bits [35, 21].

1.2.2. Criptografía de Curvas Elípticas

Las curvas elípticas constituyen un formalismo matemático conocido y estudiado desde hace más de 150 años. Las primeras propuestas de uso de las curvas elípticas en la criptografía fueron hechas por Neal Koblitz y Victor Millar, de manera independiente, en 1985. La criptografía de curvas elípticas (**CCE**) fundamenta su seguridad en el alto grado de dificultad que supone resolver el problema del logaritmo discreto en el grupo abeliano formado por curvas elípticas definidas sobre campos finitos.

De forma general, una curva elíptica $E(\mathbb{F}_q)$ se define como el conjunto de puntos que satisface la ecuación:

$$E: \ y^2 = x^3 + ax + b,$$

donde a y b están en un campo finito apropiado $\mathbb{F}_q$ de orden q, el cual puede ser el grupo de los números racionales, números complejos, enteros módulo n, campos de Galois, etc. Los coeficientes a y b caracterizan de manera unívoca cada curva.

Se define también un **punto en el infinito**, denotado como $\mathcal{O}$, a un punto imaginario situado por encima del eje de las abcisas a una distancia infinita, y que por lo tanto no tiene

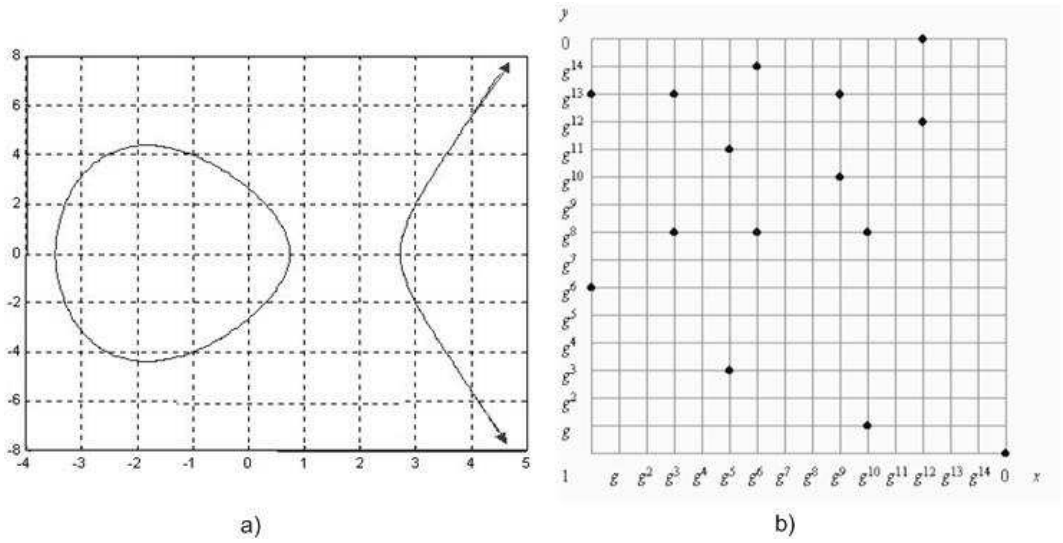


Figura 1.4: Gráficas de curvas elípticas: a) $y^2 = x^3 - 10x + 7$ sobre $\mathbb{R}$; b) $y^2 + xy = x^3 + g^4x^2 + 1$ sobre $F(2^4)$

un valor concreto. Existe en el grupo la suma y una operación conocida como multiplicación escalar: si k es un entero y $P \in E(\mathbb{F}_q)$ es un punto, entonces kP es el punto obtenido al sumar k copias de P. El elemento neutro es $\mathcal{O}$ [35, 25].

Las curvas elípticas definidas en un *Campo de Galois* $GF(P)$ siendo P un número primo, forman un grupo donde todos los elementos, con excepción del cero, tienen inversa, por lo que se puede sumar, restar, multiplicar y dividir. Los puntos de estas curvas cumplen la ecuación

$$y^2 = x^3 + ax + b \pmod{P}$$

definiendo de esta forma el grupo $E(GF(P))$ [2].

En la figura 1.4 se muestran curvas elípticas definidas en el conjunto $\mathbb{R}$, [17], y en el campo $F(2^4)$.

Tómese un punto G cualquiera de una curva elíptica E. Se denominará $\langle G \rangle$ al conjunto $\{\mathcal{O}, G, 2G, 3G, \ldots, \}$. En $E(GF(P))$ y $E(GF(2^m))$ los conjuntos de esta naturaleza deberán necesariamente ser finitos, ya que el número de puntos de la curva es finito. Por lo tanto, si se dispone de un punto $Q \in \langle G \rangle$, debe existir un número entero k tal que $kG = Q$.

El *problema de logaritmo discreto para las curvas elípticas* consiste en hallar el número k a partir de G y Q.

Debido a la enorme complejidad computacional que dicho problema matemático representa, es posible obtener con CCE niveles de seguridad similares a los proporcionados por otros sistemas de cifrado al precio de operaciones de campos finitos mucho menores a las

requeridas por los otros esquemas. Las operaciones sobre campos finitos menores conducen al uso de llaves públicas y secretas también menores lo que a su vez tiene como resultado una mayor velocidad, y menores requerimientos de memoria y de poder de cómputo en las implementaciones de los algoritmos que conforman al esquema.

1.3. Protocolos Criptográficos

Un escenario típico en transacciones electrónicas incluye a diferentes actores, tales como individuos, compañías, computadoras, lectores de tarjetas magnéticas, etc., los cuales se comunican usando una variedad de canales (teléfono, correo electrónico, radio, redes, etc.) o dispositivos físicos (tarjetas bancarias, pasajes, cédulas, etc.). La comunicación entre los actores está gobernada por reglas basadas en protocolos criptográficos para soportar ataques de carácter malicioso [12].

Un **protocolo criptográfico** es un algoritmo distribuido definido por una secuencia de pasos que especifican, de manera precisa, las acciones requeridas por dos o más entidades para llevar a cabo un objetivo de seguridad específico [2]. Dichos protocolos sirven para llevar a cabo los servicios de seguridad como el de autentificación, confidencialidad, etc.

Los protocolos criptográficos usan funciones de criptografía en algunos o todos los pasos que los componen. Para que un protocolo criptográfico que involucra intercambio de mensajes tenga una ejecución exitosa se requiere de una definición precisa de los mensajes a intercambiarse y que el protocolo y los participantes cumplan las siguientes premisas:

1. Todos los participantes deben conocer los pasos del protocolo de antemano.
2. Todos los participantes deben estar de acuerdo en seguir el protocolo.
3. El protocolo no admite ambigüedades.
4. El protocolo debe ser completo, es decir, debe definir qué hacer en cualquier circunstancia posible.
5. No debe ser posible hacer más que lo que el protocolo define.

Los protocolos pueden ser clasificados en base a su modo de funcionamiento en:

Protocolos arbitrados: Aquellos en los que es necesaria la participación de un tercero para garantizar la seguridad del sistema. Los protocolos arbitrados pueden ser ineficientes, o incluso vulnerables a ataques contra el árbitro. La figura 1.5 muestra un esquema del protocolo arbitrado.

Este protocolo se lleva a cabo, por ejemplo, en el método de compra y venta usando un notario.

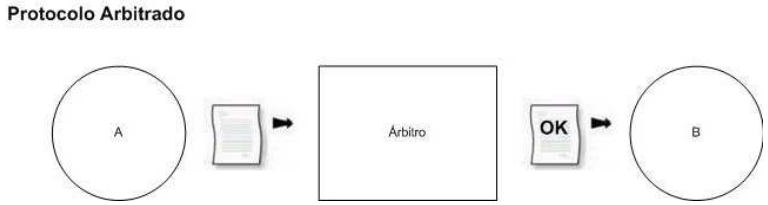


Figura 1.5: Esquema de Protocolo Arbitrado

Protocolos adjudicados: Son protocolos que tienen dos partes, una parte no-arbitrada y una arbitrada. El subprotocolo arbitrado se activa sólo en caso de que surja un disputa sobre el contrato. La figura 1.6 muestra un esquema de este tipo de protocolo.

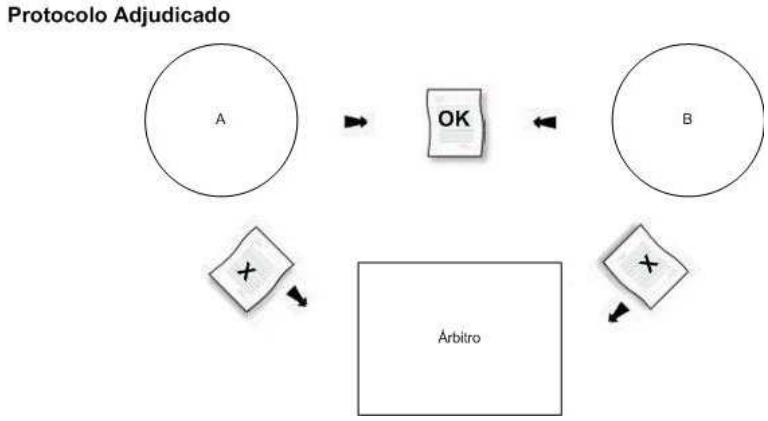


Figura 1.6: Esquema de Protocolo Adjudicado

Por ejemplo, las partes A y B quieren suscribir un contrato, para ello:

1. Negocian los términos
2. A firma
3. B firma
4. Si hay una disputa, apelan ante un juez, quien resuelve el problema.

Protocolos auto-reforzados o autoadjudicados: No se requiere de una árbitro para garantizar el protocolo. Se puede abortar la secuencia de pasos en cualquier momento, dejando sin efecto las acciones tomadas (similar a una transacción atómica). La figura 1.7 muestra un esquema de este tipo de protocolo. Cabe mencionar que no todas las situaciones son apropiadas para este tipo de protocolos.

Existen diferentes ataques a los cuales son vulnerables los protocolos criptográficos, estos ataques pueden ser *activos* o *pasivos*. Uno de los ataques activos más conocidos es el denominado "Intruso en Medio"(*Man In The Middle*).

Figura 1.7: Esquema de Protocolo Auto-Adjudicado

Un ejemplo de este tipo de ataque cuando se usa un protocolo autoadjudicado, se muestra en la figura 1.8. Aquí A y B deciden usar sus llaves públicas para el intercambio inicial de información, pero un intruso M está escuchando. Lo que sucede entonces es lo siguiente:

1. A manda a B su llave pública.
2. M intercepta esta llave y envía a B su propia llave pública.
3. B le envía a A su llave pública, M la intercepta y le envía a A su propia llave pública.
4. Cuando A envía a B un mensaje cifrado con su llave pública, realmente usa la de M, quien puede descifrar el mensaje y luego cifrarlo usando la llave pública de B.
5. Cuando B envía un mensaje a A, M puede hacer lo mismo que con A.

El problema es que B cree, sin evidencia, que el primer mensaje que recibe proviene de A, y viceversa. Sin apoyo directo o indirecto de un tercero, en que ambos confíen, este esquema nunca puede ser seguro. Una solución es el uso de los protocolos arbitrados, por ejemplo, donde A y B compartan una llave secreta (permanente) con una autoridad T en que ambos confíen.

Para reducir la oportunidad de ataques, además de agregar a la entidad T, se introduce una modalidad conocida como **llave de sesión** con la idea de usar una llave nueva cada vez que se inicia una nueva sesión de intercambio de mensajes. Evidentemente la llave no puede ser transmitida en claro sobre el canal. Por ello se necesita un protocolo inicial para acordar o establecer tal llave (aleatoria) para cada sesión. El protocolo inicial también tiene que ser seguro. Una posibilidad podría ser usar llaves públicas para el protocolo inicial, pero en general, son demasiado ineficientes para la sesión misma. Un protocolo que use estos conceptos sería el siguiente:

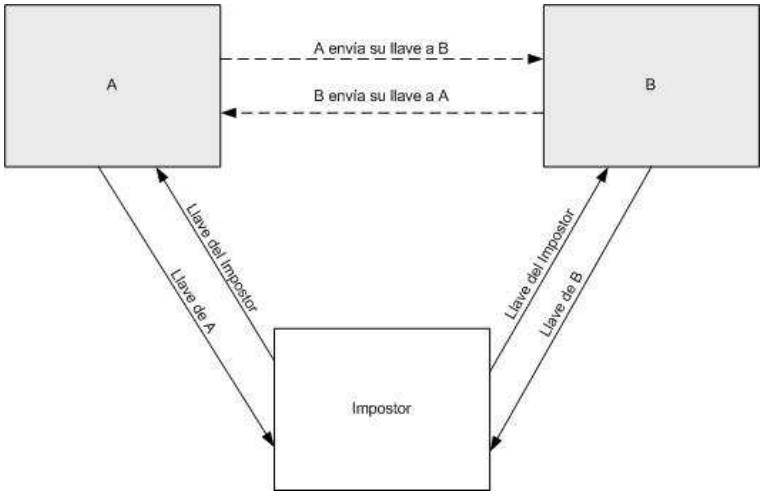


Figura 1.8: Ataque del intruso en medio

1. A llama a T y le solicita un llave de sesión para comunicarse con B.
2. T genera una llave de sesión aleatoria y cifra dos copias de esta llave, una usando la llave que comparte con A y la otra usando la llave que comparte con B.
3. T le envía ambas copias a A. A descifra su copia de la llave de sesión.
4. A le envía a B su copia de la llave de sesión (cifrada por T).
5. B descifra el mensaje de T y recupera a su vez su llave de sesión.
6. B y A usan esta llave de sesión para comunicarse en forma segura.

El protocolo Needham-Schroeder es la versión formal de lo anterior [3]. Estos protocolos, conocidos como **protocolos de acuerdo de llaves**, consisten en un conjunto de técnicas para establecer una llave, es decir, procesos donde un secreto compartido se hace disponible para dos o más entidades, para su uso criptográfico subsiguiente, donde cada parte aporta información para generar el secreto, y ninguna de las partes puede predeterminar el valor de salida [2]. Estos secretos compartidos son llamados o usados para derivar las *llaves de sesión*.

Idealmente una llave de sesión debe ser un secreto efímero, es decir, debe tener un uso restringido a un periodo corto, tal como una sola conexión de telecomunicación (o sesión), después todo rastro de ella debe ser eliminado [2]. La motivación para las llaves de sesión con tiempo de vida efímero son las siguientes:

1. limitar el texto cifrado disponible (bajo una misma llave) para evitar ataques de análisis de texto cifrado;

2. limitar la exposición, con respecto tanto al tiempo y cantidad de datos, en el caso de que alguna llave de sesión esté comprometida;

3. evitar el almacenamiento de distintas llaves, generando llaves únicamente cuando se requieran;

4. crear independencia entre sesiones o aplicaciones.

Uno de los protocolos más eficientes para el acuerdo de llaves es el algoritmo de *Diffie-Hellman* que es un algoritmo asimétrico, basado en el problema de Diffie-Hellman que se emplea fundamentalmente para acordar una llave común entre dos interlocutores, a través de un canal de comunicación inseguro. La gran ventaja de este sistema es que no son necesarias llaves públicas en el sentido estricto, sino una información compartida por los dos comunicantes.

1.3.1. Algoritmo de Diffie-Hellman

Matemáticamente, el algoritmo de cifrado de Diffie-Hellman se basa en las potencias de los números y en la función modular. Uniendo estos dos conceptos se define la potencia discreta de un número como $Y = X^a \bmod q$. Si bien el cálculo de potencias discretas es fácil, la obtención de su función inversa, el logaritmo discreto, no tiene una solución analítica para números grandes. Por lo que la seguridad del algoritmo Diffie-Hellman radica en la dificultad de resolver el problema de los logaritmos discretos, que se considera computacionalmente equivalente a factorizar números enteros grandes.

Para implementar el sistema se siguen los siguientes pasos:

- Se busca un número primo grande llamado q.
- Se busca β raíz primitiva de q.
- β y q son llaves públicas.

A partir de tal información generada por cada parte se sigue el diagrama de la figura 1.9.

Para generar una llave simétrica compartida entre dos usuarios, A y B, ambas entidades generan un números pseudoaleatorio, X_a y X_b. Éstos son las llaves privadas de A y B. Con estos números y las llaves públicas β y q que ambos conocen, cada uno genera un número intermedio, Y_a e Y_b, mediante las fórmulas:

$$Y_a = \beta^{X_a} \bmod q$$

$$Y_b = \beta^{X_b} \bmod q$$

Figura 1.9: Diffie-Hellman

Estos números son intercambiados entre ambos, y luego cada uno opera con el que recibe del otro, obteniendo en el proceso el mismo número ambos:

$$K = Y_b X_a \bmod q$$

$$K = Y_a X_b \bmod q$$

El número K es la llave simétrica que a partir de ese momento ambos comparten, y que pueden usar para establecer una comunicación cifrada mediante cualquiera de los sistemas simétricos.

1.3.2. ECDH, Acuerdo de Llaves con CE

El problema del logaritmo discreto es la base para la seguridad de muchos sistemas criptográficos incluyendo al de curvas elípticas. La criptografía de curvas elípticas utiliza al grupo de puntos definidos en una curva elíptica sobre un campo finito para obtener una variante del algoritmo para el acuerdo de llaves convencional Diffie-Hellman, *ECDH*[31].

La generación de llaves de sesión utilizando ECDH requiere ciertos parámetros de las curvas elípticas: la curva a ser utilizada, el punto generador (P), el orden de P, (n) y otros más.

El cliente y el servidor, generan cada uno por su parte, un entero aleatorio r_c y r_s, dentro del intervalo $[1, n-1]$, calculando la llave pública de ECDH $K_a = r_a P$ y $K_b = r_b P$. Esta llave pública es enviada a la contraparte dentro del protocolo de Negociación. Entonces cada entidad calcula el punto $Z_b = r_b K_a$ y $Z_a = r_a K_b$. Los puntos Z_a y Z_b son idénticos y serán utilizados como la llave de sesión por ambas partes [2, 31]. Este proceso se muestra en la figura 1.10.

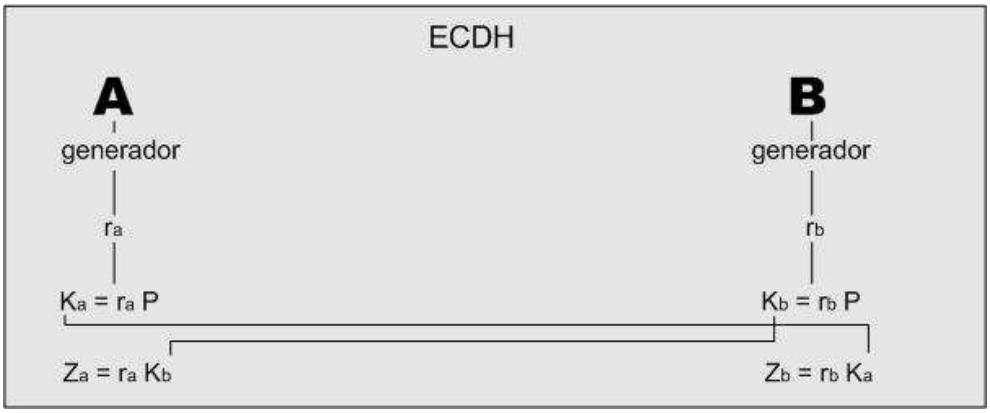


Figura 1.10: Variante de Diffie-Hellman para Curvas Elípticas

1.3.3. Acuerdo de Llaves con RSA

Cuando se utiliza RSA para el acuerdo de llaves, el cliente genera un valor secreto de 20 bytes S, lo cifra con la llave pública del servidor y lo envía al servidor. El servidor utiliza su llave privada para descifrar el mensaje enviado por el cliente para obtener el valor secreto S. La llave de sesión es el valor secreto concatenado a la llave pública del servidor [9].

Las operaciones de llave pública de cifrado y descifrado de RSA se realizan siguiendo las especificaciones del estándar PKCS #1 (*Public Key Cryptography Standard*)[22].

1.4. Esquema de Firma Digital

Una primitiva criptográfica que es fundamental en la autentificación, autorización y no repudio, es la *firma digital*. El propósito de una firma digital es proveer a una entidad un medio para enlazar su identidad a una pieza de información. El proceso de *firma* dentro de los esquemas de llave pública se puede ver como el proceso de cifrado con la llave privada y el proceso de *verificación* se puede ver como el proceso de descifrado con la llave pública. El esquema general de firma digital se muestra en la figura 1.11. Como se observa, al mensaje se le aplica una función *hash*[2] cuyo resultado será firmado con la llave privada del signatario y anexado al mensaje para ser enviados al destinatario. El destinatario separa los dos componentes: el mensaje y la firma. Le aplica la misma función *hash* al mensaje obteniendo el valor v_1 y a la firma la verifica con la llave pública del signatario obteniendo el valor v_2, si $v_1 = v_2$ se dirá que el mensaje no ha sido alterado en la transmisión y que la autenticidad del origen ha sido confirmada.

[2]Una función *hash* toma un mensaje como entrada de longitud arbitraria, lo digiere y produce una salida conocida como *digestión* de longitud fija. Los tipos de funciones *hash* como MD5 y SHA1 se describen en la sección 2.2.

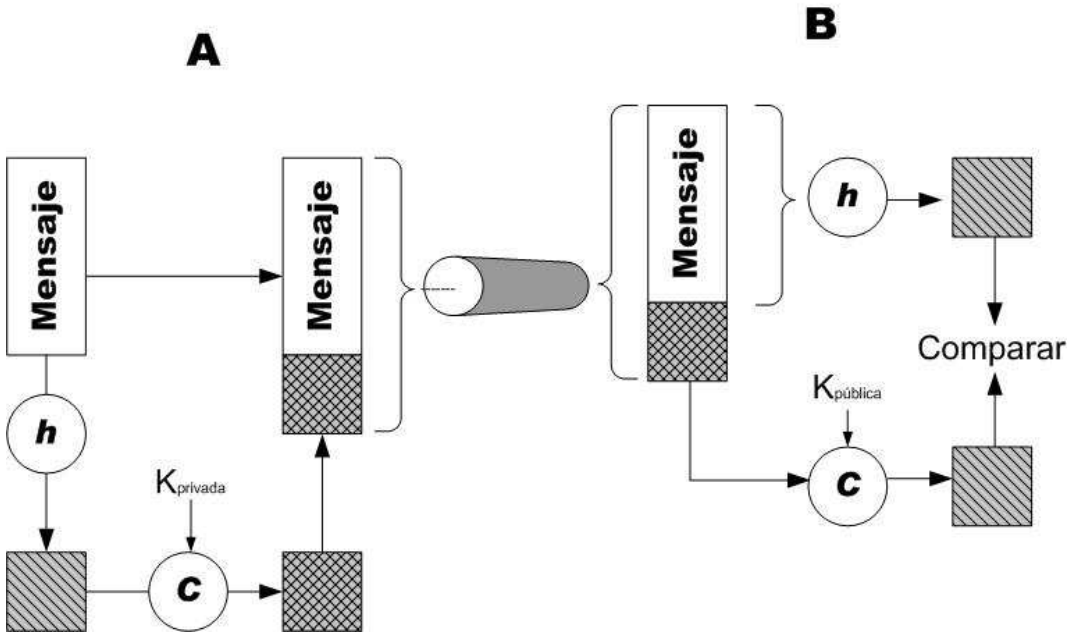


Figura 1.11: Esquema de firma digital con apéndice

Existen diferentes esquemas de firmas dependiendo del sistema criptográfico de llave pública que se utilice, tal como se aprecia a continuación.

1.4.1. El Algoritmo de Firma Digital, DSA

El Instituto Nacional de Estándares y Tecnología de Estados Unidos (NIST) propuso en 1991 el algoritmo de firma digital, DSA por sus siglas en inglés, y se adoptó como estándar en 1994. DSA es un esquema de firma digital con apéndice, y como otros esquemas, se firma un mensaje digerido. Por lo que en el algoritmo es asumido que el mensaje ya se le ha aplicado una función de *hash*. Para generar el par de llaves de DSA se debe seguir una fase de inicialización:

1. A encuentra un primo q de longitud 160 bits y elige un primo p tal que $q|p-1$. El problema del logaritmo discreto debe ser difícil para esta elección de p.
2. Sea g un raíz primitiva mod p y sea $\alpha \equiv g^{(p-1)/q} \pmod{p}$. Entonces $\alpha^q \equiv 1 \pmod{p}$.
3. A elige un secreto a tal que $1 \leq a < q-1$ y calcula $\beta \equiv \alpha \pmod{p}$.
4. A publica (p, q, α, β) se hace público, y mantiene a a secreto.

A firma un mensaje m con el siguiente procedimiento:

1. Selecciona un número aleatorio, entero k, que mantiene secreto, tal que $0 < k < q - 1$.
2. Calcular $r = (\alpha^k \pmod p) \pmod q$.
3. Calcular $s \equiv k^{-1}(m + ar) \pmod q$.
4. La firma de A para m es (r, s), la cual se envía a B junto con m.

Para que B verifique la firma debe:

1. Obtener los datos públicos de A, (p, q, α, β).
2. Calcular $u_1 \equiv s^{-1}m \pmod q$, y $u_2 \equiv s^{-1}r \pmod q$
3. Calcular $v = (\alpha^{u_1}\beta^{u_2} \pmod p) \pmod q$.
4. Se acepta la firma si y sólo si $v = r$.

1.4.2. ECDSA

El algoritmo de firma digital con curvas elípticas, ECDSA por sus siglas en inglés, es el análogo para la criptografía de llave pública de curvas elípticas y DSA. Fue aceptado en 1999 como un estándar ANSI, y fue adoptado en 2000 por la IEEE y la NIST como estándar.

A diferencia de los problemas ordinarios de logaritmo discreto y problemas de factorización, no se conoce algún algoritmo de tiempo subexponencial para el problema de logaritmo discreto con curvas elípticas. Por esta razón, la fortaleza por cada bit de la llave es substancialmente mayor en un algoritmo que utiliza curvas elípticas.

Para firmar un mensaje m con ECDSA, una entidad A con parámetros de dominio $D = (q, FR, a, b, G, n, h)$ y un par de llaves asociado (d, Q) hace lo siguiente:

1. Selecciona un entero aleatorio k, $1 \leq k \leq -1$.
2. Calcular $kG = (x_1, y_1)$ y convertir x_1 a un entero $\bar{x}_1$.
3. Calcular $r = x_1 \bmod n$. Si $r = 0$ entonces ir al paso 1.
4. Calcular $k^{-1} \bmod n$.
5. Calcular $SHA - 1(m)$ y convertir esta cadena de bits a un entero e.
6. Calcular $s = k{-}1(e + dr) \bmod n$. Si $s = 0$ entonces ir a paso 1.
7. La firma de A para el mensaje m es (r, s).

Para verificar la firma con ECDSA de A de (r, s) sobre m, B obtiene una copia de los parámetros de dominio de A $D = (q, FR, a, b, G, n, h)$ y la llave pública asociada Q. B entonces realiza lo siguiente:

1. Verificar que r y s sean enteros dentro del intervalo $[1, n-1]$.
2. Calcular $SHA-1(m)$ y convierte esta cadena de bits en un entero e.
3. Calcular $w = s^{-1} \bmod n$.
4. Calcular $u_1 = ew \bmod n$ y $u_2 = rw \bmod n$.
5. Calcular $X = u_1 G + u_2 Q$.
6. Si $X = 0$, entonces rechazar la firma. De otro modo, convertir la $x-coordenada$ x_1 de X a un entero $\bar{x}_1$, y calcular $v = \bar{x}_1 \bmod n$.
7. Aceptar la firma si y sólo si $v = r$.

1.4.3. Firmas con RSA

Si se utiliza el esquema de llave pública RSA para firma digital los pasos son los siguientes:

1. Recapitulando, A genera dos primos grandes p, q y calcula $n = pq$. A elije e_A tal que $1 < e_A < \phi(n)$ con $mcd(e_A, \phi(n)) = 1$, y calcula d_A tal que $e_A d_A \equiv 1 \pmod{\phi(n)}$. A publica (e_A, n) y mantiene privado d_A, p, q. El proceso de generación de llaves se da por hecho al iniciar el procedimiento de firma.
2. La firma de A es
$$y \equiv m^{d_A} \pmod{n}.$$
3. Entonces el par (m, y) se hace público.

B puede verificar que A firmó el mensaje siguiendo los siguientes pasos:

1. Obtener (e_A, n) de A.
2. Calcular $z \equiv y e_A \pmod{n}$. Si $z = m$, entonces B puede aceptar la firma como válida; de otra manera la firma no es válida.

El sistema criptográfico RSA presenta algunos inconvenientes para las firmas digitales parecidos a los que presenta como sistema de cifrado. En particular, no se sabe a ciencia cierta si es tan difícil de romper como la factorización de grandes enteros. Incluso aunque así fuera, dados un mensaje original elegido m y la llave de cifrado de otro usuario (e, n), calcular la firma digital s tal que $m - s^e (\text{mod } n)$ puede ser mucho más fácil si se tiene, además, (s', m'), donde s' es la firma digital del usuario legítimo para un mensaje m' muy parecido al mensaje m. En otras palabras, podría resultar fácil falsificar firmas digitales para algún mensaje dado después de haber visto las firmas digitales auténticas de varios mensaje parecidos.

Lo arriba mencionado sugiere que podría resultar más favorable para el diseño de esquemas de firmas digitales el empleo de sistemas probabilísticos, en vez de los sistemas de llave pública. Sin embargo, ésta es una tarea difícil, ya que, por ejemplo, se ha demostrado que el sistema probabilístico de Blum-Goldwasser es inútil para firmas digitales [24]. Debido a este tipo de ataques para la firma y verificación, el estándar de criptografía de RSA **PKCS** #1 versión 2.1 da recomendaciones para la implementación de los esquemas criptográficos de llave pública basados en RSA: primitivas criptográficas, esquemas de cifrado, esquemas de firma, y la sintaxis ASN.1 para representar a las llaves.

Las figuras 1.12 y 1.13 indican el procedimiento de la firma y verificación para RSA en este estándar. Para cifrar el mensaje m, se digiere con una función *hash* dando como resultado una digestión que es codificada de acuerdo al estándar en una cadena de octetos. A continuación el resultado se divide en bloques y cada cadena de octetos es transformada a enteros. A partir de ahí se aplica la primitiva de firma de RSA vista anteriormente y el resultado es convertido de enteros a octetos, teniendo de esta manera la firma digital.

Para el proceso de verificación dentro del estándar a partir de la firma y el mensaje m, el primer paso es convertir la cadena de octetos de la firma en cadena de enteros, a lo cual se le aplica la verificación de RSA, la cadena resultante de enteros se convierte a cadena de octetos nuevamente y se le aplica un análisis para recuperar del bloque la digestión $h(m)'$, se digiere el mensaje m y el resultado, $h(m)$ debe ser idéntico a $h(m)'$.

Estos esquemas resisten los ataques principalmente al dar formato a los bloques, representado en la figura 1.12 por el paso 3, ya que todos los mensajes grandes o pequeños se codifican a bloques de tamaño normalizado de k bytes (a través del uso de bits de relleno).

Se han cubierto los conceptos básicos de la criptografía de llave pública, los cuales se utilizan en el utilizan en el siguiente capítulo, que trata sobre la Autentificación.

Figura 1.12: Firma en RSA

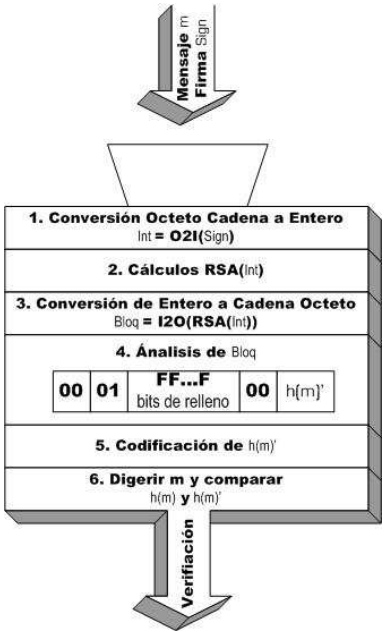


Figura 1.13: Verificación en RSA

Capítulo 2

Autentificación

Como se discutió en el capítulo anterior, la principal meta de la criptografía es garantizar que se cumplan los cuatro servicios de la seguridad computacional: la Confidencialidad, la Integridad de los datos, la Disponibilidad la Autentificación y el No-Repudio. La **autentificación** es, a grandes rasgos, el proceso mediante el cual se verifica y asegura la identidad de las partes involucradas en una transacción. Si este proceso no se llevara a cabo cabría la posibilidad de que una entidad desconocida asuma una identidad falsa, comprometiendo de esta manera la privacidad y la integridad de la información.

La autentificación es necesaria en los sistemas de llave pública, como se verá a continuación. Puede parecer que los sistemas de llave pública son ideales y que no requieren de un canal seguro para transportar la llave de cifrado. Esto implicaría que dos entidades pueden comunicarse sobre un canal inseguro sin haberse nunca encontrado para intercambiar llaves. Desafortunadamente, este no es el caso. El ataque conocido como "intruso en medio" muestra como un adversario activo puede burlar el modelo sin romper el criptosistema. Esto resalta la necesidad de autenticar a las llaves públicas para lograr una certificación del origen de datos de las llaves públicas en sí.

En la primera sección se señalarán los diferentes tipos de autentificación que existen. En la sección 2.4 se introduce a la infraestructura de llave pública para tener un panorama más amplio de lo que los sistemas de llave pública pueden cubrir, principalmente los certificados, que es un elemento de la autentificación.

2.1. Métodos de Autentificación

La autentificación es cualquier proceso a través de cuál se demuestra y se verifica cierta información referente a un objeto, como el origen de un documento, la identidad del remitente, momento en que un documento fue enviado y/o firmado, la identidad de una computadora o usuario, etc.

Los métodos de autentificación se clasifican en cinco tipos [2]:

Autentificación del origen de datos. La autentificación del origen de datos es un tipo de autentificación donde la identidad de una de las partes es corroborada como la fuente original de datos específicos creados en algún momento (típicamente sin ser señalado) en el pasado. Por definición, este tipo de autentificación incluye integridad de datos.

Autentificación de mensaje. Esta autentificación sucede cuando se quiere garantizar la procedencia de un mensaje conocido, de forma que se pueda asegurar de que no es una falsificación. La autentificación de mensaje provee autentificación del origen de los datos con respecto a la fuente del mensaje original. Provee integridad de datos pero no una garantía sobre la línea del tiempo.

Autentificación de transacción. La autentificación de transacción denota autentificación de mensajes aunado a una garantía de existencia única y temporal, es decir, que identifique el momento preciso de creación.

Autentificación de entidad. Esta autentificación es el proceso por el cual una de las partes, mediante la adquisición de evidencia que se puede corroborar, está seguro de la identidad de la otra parte involucrada en el protocolo, y que esa otra parte está activa en ese justo momento. Los términos *Identificación* y Autentificación de entidad se usan comúnmente como sinónimos. La identificación está basada en una o más de estas características: *algo que se conozca* (contraseña, NIP, etc.); *algo que se posea* (por ejemplo, una tarjeta de identificación); y *algo que sea inherente* a un individuo (huellas digitales u otras características biométricas).

Autentificación de llave. La autentificación de llave es la propiedad por la cual, una parte, está segura de que ninguna otra entidad además de una segunda parte identificada (o un conjunto de partes confiables) tiene acceso a una llave secreta particular.

Las principales herramientas para llevar a cabo la autentificación de origen de datos son:

- las funciones *hash*,
- los esquemas de firma digital,
- las infraestructuras de llave pública, y
- los certificados

El enfoque de este trabajo es en la autentificación de origen de datos y se abarca cada una de estas herramientas para llevarla a cabo. Es importante tener en cuenta que la firma digital, como se resalta más adelante, no se aplica al mensaje en sí, si no que se aplica al valor resultante de una función llamada *hash*. Previo al tema de autentificación mediante firma digital se presenta este tipo de funciones.

2.2. Funciones hash para Firma Digital: MD5 y SHA-1

Las funciones *hash*, también conocidas como funciones de digestión son una herramienta fundamental en la criptografía. Las funciones *hash* son usadas principalmente para resolver problemas de la integridad de los mensajes, así como en los procesos de verificación de la autenticidad de mensajes y de su origen. Como ha sido mencionado, los sistemas de llave pública son muy lentos, por lo que en vez de firmar digitalmente el mensaje completo, la firma se aplica sobre el mensaje digerido.

Una función **hash** toma un mensaje con entrada de longitud arbitraria, lo digiere y produce una salida conocida como *digestión* de longitud fija. Definida de manera formal, una función *hash* h mapea cadenas de bits de longitud arbitraria finita a cadenas de longitud fija de n bits.

La idea básica de las funciones *hash* es que el digerido sirve como una imagen representativa compacta, llamada *huella digital* o *mensaje digerido*, de una cadena de entrada y puede utilizarse como si fuera una identificación única de la cadena de entrada.

Para un domino D y un rango R se tiene que $h : D \rightarrow R$ y $|D| > |R|$, implicando que la función es de muchos a uno y que existe la existencia de colisiones, es decir, de pares de valores de entrada que corresponden a un valor de salida idéntico. Sin embargo, para generar colisiones en una función *hash* "aleatoria perfecta"de n bits, la *paradoja del cumpleaños* nos indica que la probabilidad de que esto suceda es de $2^{1/n}$ donde n es el número de bits.

La *paradoja del cumpleaños* es la respuesta a la siguiente pregunta:

¿Cuántas personas necesitamos reunir, de forma aleatoria, para que la probabilidad de que por lo menos dos de ellas tengan su cumpleaños el mismo día sea $1/2$*?*

La respuesta es un número sorprendentemente pequeño: 23 personas. En general, si se desea elegir con repetición de una colección de n objetos, se necesitan $1.77\sqrt{n}$ intentos para obtener al menos una repetición con probabilidad de 50 %.

Las propiedades matemáticas son deseables para una función *hash* son las siguientes:

- La función $h(m)$ debe ser fácil de calcular para cualquier m.
- La función *hash* debe ser de sólo ida, es decir, si se conoce $h(m)$ encontrar m debe implicar calcular todos los m posibles.
- La función *hash* debe ser resistente a las colisiones, es decir, no debe ser posible (computacionalmente) encontrar m y m' con $m \neq m'$ tales que $h(m) = h(m')$.

Las funciones *hash* se emplean para integridad de datos en unión con los esquemas de firma digital, donde, por varias razones, un mensaje típicamente es digerido primero y entonces, el valor *hash*, como una representación del mensaje, es firmado en lugar del mensaje original, tal como se muestra en la figura 1.11.

Generalmente, las funciones *hash* asocian una cadena de longitud de 160 bits que los hace más manejables para el propósito de firma digital. Entre los algoritmos más importantes de las funciones *hash* están: MD5 y SHA-1.

El Algoritmo MD5 es el resultado de una serie de mejoras sobre el algoritmo MD4, diseñado por Ron Rivest, procesa los mensajes de entrada en bloques de 512 bits, y produce una salida de 128 bits. En los últimos tiempos el algoritmo MD5 ha mostrado ciertas debilidades, aunque sin implicaciones prácticas reales, por lo que se sigue considerando en la actualidad un algoritmo seguro, si bien su uso tiende a disminuir.

El algoritmo SHA-1 fue desarrollado por la NSA, para ser incluido en el estándar DSS (*Digital Signature Standard*). Al contrario que los algoritmos de cifrado propuestos por esta organización, SHA-1 se considera seguro y libre de puertas traseras [1]. Produce tramas de 160 bits, a partir de bloques de 512 bits del mensaje original.

2.3. Firmas Digitales para Autentificación

Las firmas digitales han sido la herramienta criptográfica que ha resuelto varios aspectos de la autentificación e integridad de datos. A través de estos esquemas ha sido posible sustituir documentos legales en papel por documentos digitales de manera confiable y eficaz. La firma digital, a diferencia de la autógrafa, está ligada tanto al signatario como al documento o mensaje que está siendo acreditado. Los esquemas de firma digital consisten de dos pasos: el proceso de *firma* y el proceso de *verificación*. Cualquier alteración en el documento firmado digitalmente hará que el proceso de verificación falle y la firma no sea aceptada.

La firma digital, como mencionamos anteriormente, es un proceso criptográfico que permite asegurar la identidad del autor de un documento, y la inalterabilidad del contenido del documento firmado; para que ello sea posible, la firma digital debe ser:

- única
- infalsificable
- fácil de verificar

Para hacer más eficientes los esquemas, los mensajes son preprocesados por una función de digestión mejor conocida como funciones *hash*. Partiendo de una función *hash* h pública, se toma el mensaje m a procesar y se calcula $h(m)$, la salida de $h(m)$ es significativamente más pequeña que m y por lo tanto firmar el valor *hash* puede hacerse más rápido que procesar el mensaje completo.

[1] Una puerta trasera o *BackDoor* es una característica oculta de algunas aplicaciones o algoritmos que permite a su creador acceder a opciones especiales que son inaccesibles para los usuarios

El resultado de la operación *hash* se cifra con la llave privada del remitente, $sig(h(m))$ y se usa como la firma digital del mensaje. Para verificar la firma, se envía el par $(m, sig(h(m)))$, tal como se ilustra en la figura 1.11. Al mensaje m recibido se le realiza el *hash*, $h'(m)$, se descifra $sig(h(m))$ con la llave publica del remitente y el resultado debe concordar con el resultado de la operación *hash*, $h'(m) = h(m)$.

Los diferentes esquemas de firmas digitales apoyados en la criptografía de llave pública vistos en la sección 1.4 pueden combinarse con los diferentes algoritmos de *hash*. Así por ejemplo, dentro de las especificaciones de los certificados, como se ve en la sección 2.4.1, se puede encontrar:

- SHA1 con RSA,
- SHA1 con ECDSA,
- MD5 con RSA,
- MD5 con ECDSA,
- etc.

2.4. Infraestructura de Llave Pública

Una **infraestructura de llave pública** o PKI por sus siglas en inglés (*Public Key Infrastructure*) es el término utilizado para referirse a la infraestructura de seguridad, basada en criptografía de llave pública, que permite la gestión de certificados digitales. La meta de una infraestructura de llave pública es cumplir las necesidades del *control de acceso*, de la *identificación automatizada* y de la *autentificación* de manera determinista [28].

Una PKI consiste de políticas que definen las reglas bajo las cuales los sistemas criptográficos operarán y los procedimientos para generar y publicar las llaves y certificados. Todas las PKIs consisten de operaciones de *certificación* y *validación*. La certificación y la validación garantizan que los certificados sean legítimos.

Un **certificado** es una porción de información que ha sido firmada digitalmente por una tercera parte confiable, a quien se le refiere comúnmente como la autoridad certificadora.

Una **autoridad certificadora** (CA) es una organización (o una subdivisión de una organización) responsable de verificar los atributos de seguridad de los usuarios de un sistema de computacional, e introducir esta información verificada en el sistema.

Un modelo arquitectural simplificado de una PKI se muestra en la figura 2.1. Los componentes de este modelo son:

Entidad Final: son los usuarios de los certificados de la PKI y/o los usuarios de los sistemas que están en el asunto de un certificado;

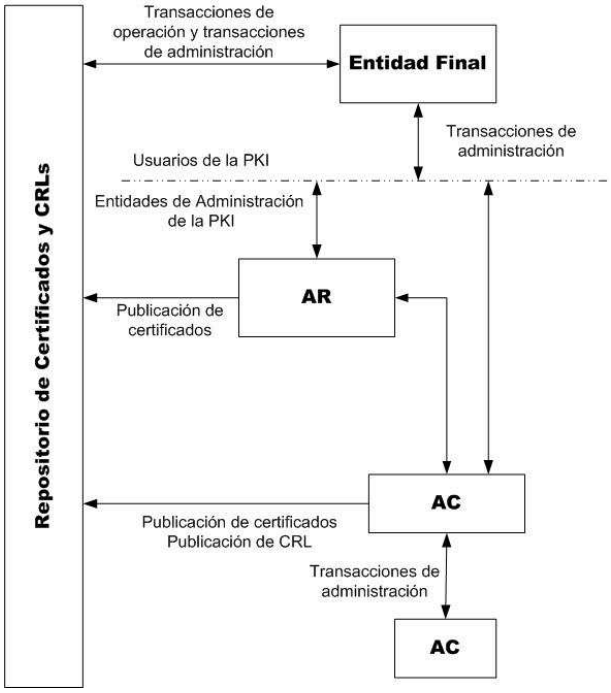


Figura 2.1: Entidades de las PKI

AC: Una autoridad certificadora;

AR: Una autoridad de registro, es decir, un sistema opcional al cual le delega una AC ciertas funciones de administración;

Repositorio: Un sistema o una colección de sistemas distribuidos que almacenan certificados y *listas de revocación de certificados* (CLRs), y sirve como medio de distribución de esos certificados y CRLs a las entidades finales.

2.4.1. Certificados

Los servicios que una PKI ofrece determinan los atributos contenidos en los certificados así como la información de control dentro de los certificados tales como las restricciones de la ruta de los certificados y las políticas que definen a los datos englobados en él, etc [28].

Los usuarios de una llave pública deben de estar convencidos que la llave privada correspondiente sea la del usuario correcto, ya sea una persona o sistema, con la que se utiliza un mecanismo de cifrado o firma digital. Esta confianza se obtiene con el uso de los certificados de llave pública, que son estructuras de datos asociados a los valores de la llave pública del usuario.

Como se ha mencionado, los certificados digitales se utilizan para identificar de manera

única a las personas y recursos en las redes o en el Internet. Los certificados habilitan, de esta manera, comunicaciones seguras y confidenciales entre dos partes.

El enlace de la información del usuario con la llave pública correspondiente se realiza cuando una autoridad certificadora confiable firma digitalmente cada certificado. Un certificado incluye diferentes campos con información relativa a su propietario y a la autoridad certificadora que lo respalda, tales campos son:

- El nombre del usuario y un conjunto de datos que lo identifican de manera única. Los datos pueden ser: país, estado, nombre de la organización, nombre de la persona y alguna otra información como la URL del servidor Web que contiene al certificado, o la dirección de correo electrónico del usuario.
- La llave pública del usuario.
- El nombre de la autoridad certificadora que extiende el certificado
- Un número de serie.
- El periodo de validez, es decir, el tiempo de vida del certificado (incluyendo una fecha de inicio y una de fin).

Al crear el certificado, la información contenida en él es firmada digitalmente por la CA. La firma digital de la CA sirve para detectar que el contenido del certificado no ha sido alterado y autenticar la llave del usuario. A continuación se detallará el tipo de certificado más difundido, el certificado conocido como X.509.

2.4.2. Certificados X.509

El estándar, internacionalmente aceptado, para certificados digitales, es el definido en la infraestructura de llave pública X.509, en su versión 3. La especificación **X.509** define el formato y la semántica de los certificados y de las listas de revocación de certificados para las PKI del Internet. Se describen también los procedimientos para el proceso de las rutas de certificación en el ambiente de Internet y, entre otras características, se proveen las reglas de codificación de los algoritmos criptográficos a utilizarse [28].

La primera versión del formato X.509 apareció en 1988, siendo la propuesta más antigua para una PKI a nivel mundial. Esto junto con su origen ISO/ITU han hecho de X.509 el PKI más ampliamente utilizado. En 1993 fue extendida a la versión 2, agregando únicamente dos campos a los certificados: el identificador del emisor y el usuario del certificado. La versión 3 de X.509 amplía la funcionalidad del estándar X.509, e introduce cambios significativos en el estándar. El cambio fundamental es el hacer el formato de los certificados y los CRLs extensible, es decir, se agregan campos denominados *extensiones* donde se pueden definir características de los certificados inherentes a cada entidad [10].

Los certificados están codificación usando el estándar X.208, que define las reglas de codificación distinguidas ASN.1, *DER* por sus siglas en inglés. La codificación DER ASN.1 incluye una etiqueta, la longitud del elemento y el valor codificado para cada elemento [28].

Los campos de un certificado X.509 versión 3 se muestra en la figura 2.2. El certificado esta compuesto de tres áreas principales:

1. El Certificado *TBS*, que contiene la *versión* del certificado, el *número de serie*, el *identificador del algoritmo* de la firma, el *nombre del emisor*, el periodo de *validez* del certificado, el *usuario* que esta siendo certificado, la *información de la llave pública* del usuario. Es opcional su presencia del *identificador único* del emisor, del *identificador único* del usuario y de las extensiones.

2. El *Identificador del Algoritmo de Firma* que toma un código preestablecido en [16].

3. El *Valor de la Firma* que es una cadena de bits.

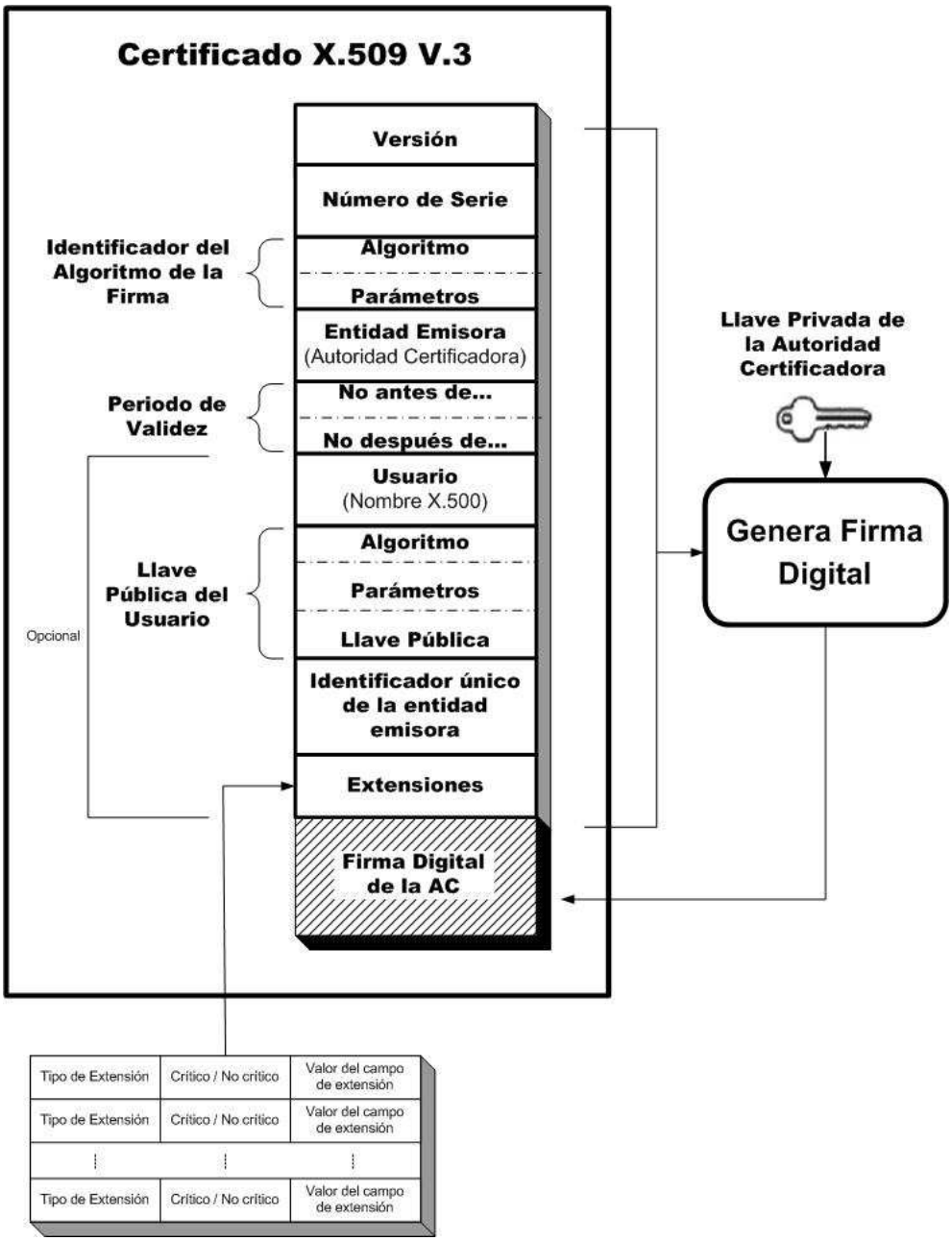


Figura 2.2: Certificado X.509 V3

Capítulo 3

Seguridad en Redes de Dispositivos Móviles

En el siglo XXI, se prevé que Internet tendrá más de 100 millones de servidores, habrá más de un millón de redes enlazadas accesibles en cualquier parte del mundo. Estará disponible por cualquier medio, transportará miles de aplicaciones comerciales, educativas, de distracción, de investigación. Cambiará las fronteras políticas, institucionales, económicas, culturales [33]. Desafortunadamente, este crecimiento incluye el incremento de individuos maliciosos dispuestos a atacar de manera activa o pasiva a los sistemas de información. Es por ello que se deben establecer normas o políticas que garanticen la *seguridad computacional* dentro de una organización.

El caso de las redes inalámbricas no es una excepción, debido a la creciente cantidad de transacciones comerciales que se realizan diariamente en este tipo de redes, la seguridad electrónica ha adquirido un papel fundamental para tales medios de comunicación. La autentificación, como se vio en el capítulo anterior, es de gran importancia en cualquier sistema que intercambie información valiosa, como se apreciará más adelante, si una entidad desconocida asume una identidad falsa, la privacidad y la integridad de la información se verán comprometidas.

En la primera sección de este capítulo se ofrece una visión general de las redes inalámbricas, en la que se incluye una taxonomía referente a su alcance. Dentro de esta sección se dará una introducción a la seguridad computacional enfocada a las redes inalámbricas. A continuación se presentan a detalle dos de los protocolos para el acceso a redes inalámbricas de diferentes alcance: el IEEE 802.11 y WAP. En la sección 3.2 se presenta una panorámica del protocolo IEEE 802.11 y su esquema de seguridad. En la sección 3.3 se revisará la estructura de WAP que incluye dentro de su arquitectura un protocolo de seguridad robusto llamado WTLS. El estudio se enfocará en este protocolo, en específico en el protocolo de mayor costo computacional, el protocolo de Negociación.

3.1. Redes Inalámbricas

La movilidad se ha vuelto un requerimiento cada vez mayor dentro de los ambientes de trabajo, ahora, se debe tener la información precisa en forma instantánea, es decir, la comunicación debe ser inmediata, en tiempo real y en cualquier lugar. Existen ya en el mercado diversos dispositivos ligeros que permiten llevar información y poder de cómputo a diversos lugares, pero las redes inalámbricas agregan una movilidad real a tales dispositivos. Dentro de las redes inalámbricas existen tres categorías: redes de área amplia o metropolitana (WAN/MAN), redes de área local (LAN) y redes de área personal (PAN):

- En la primera categoría, WAN/MAN, se tienen a las redes que cubren miles de kilómetros; la tecnología que utilizada son las redes celulares (GSM, TDMA, CDMA, 3G, etc) [5].
- Las redes inalámbricas tipo LAN (WLAN) son redes que tienen un alcance de decenas de metros [5].
- La última categoría, PAN, cubre distancias cortas y cerradas, algunas de las tecnologías que son utilizadas aquí son Bluetooth, 802.15 y Homero [5], pero no serán objeto de nuestro estudio.

El principal acceso al medio de las redes inalámbricas **WAN/MAN** es a través de la telefonía celular. Aunque originalmente la telefonía celular fue utilizada para la transferencia de voz, se han desarrollado protocolos importantes para poder transferir datos a través de esta tecnología inalámbrica lo que ha evolucionado a tener acceso a Internet en este tipo de redes. Un ejemplo de estos protocolos es CDPD (*Celullar Digital Packet Data*), desarrollado a mediados de los 90s por AT&T. CDPD provee la transmisión inalámbrica de datos digitales como Internet a través de la telefonía celular. Sin embargo el acceso es limitado debido a que CDPD está basado en el protocolo de Internet TCP/IP, sin tomar en cuenta las limitaciones de los dispositivos móviles [5].

Otro protocolo que provee acceso a Internet es WAP (*Wireless Access Protocol*). Con WAP son posibles las comunicaciones de datos entre redes inalámbricas a celulares y otros dispositivos portátiles como PDAs, radiolocalizadores, teléfonos inteligentes, etc. Las especificaciones de WAP soportan la mayoría de los servicios y protocolos de las redes celulares de hoy en día tales como GSM, PDC, TDMA, CDMA y CDPD. Uno de los principales objetivos de la especificación WAP es permitir que dispositivos portátiles se interconecten con las redes inalámbricas independientemente de sistemas operativos y protocolos. Es por eso que WAP utiliza un lenguaje conocido como WML (Wireless Markup Language) que permite la conexión entre las redes y los dispositivos portátiles. Con WAP y WML el contenido de Internet puede ser formateado para uso en una pequeña pantalla de un dispositivo portátil [5]. Las redes WLANs se han extendido rápida y ampliamente. De hecho, se considera que este tipo de redes es el mayor suceso en la historia de la tecnología en las últimas décadas. Su uso más extendido es en mercados como hospitales, fábricas, tiendas de autoservicio y

departamentales, áreas académicas, entre otros. Las ventajas más difundidas de este tipo de redes son: la capacidad de incorporar de manera fácil a nuevos usuarios a la red; ofrecer una alternativa de bajo costo a los sistemas cableados y además, la posibilidad para acceder cualquier base de datos o cualquier aplicación localizada dentro de la red de manera ubicua. Las tecnologías que pueden formar este tipo de redes son HiperLAN/2 e IEEE 802.11, ésta última de mayor aceptación y que actualmente permiten una funcionalidad comparable a la de Ethernet [26].

La seguridad juega un papel muy importante para que el desarrollo e implementación de las redes inalámbricas WAN y LAN sean explotados de una manera eficaz y confiable. En la siguiente sección se presentará un breve estudio de la seguridad computacional y cuál es su importancia en las redes inalámbricas.

3.1.1. Impacto de la Seguridad en Redes Inalámbricas

En esta sección se describirán las características de la seguridad computacional y el efecto que tiene en el funcionamiento de las redes inalámbricas. La **seguridad computacional** se define como el conjunto de políticas y mecanismos que permiten garantizar los servicios de confidencialidad, integridad y disponibilidad de los recursos de un sistema.

En las últimos años, los avances en las tecnologías de comunicación digital y móvil para las redes celulares WAN/MAN han cambiado la forma de transmisión de datos, a tal grado que Internet se ha extendido a este tipo de redes, dando como resultado a Internet Móvil. Se espera que el número de usuarios de los servicios inalámbricos rebase los mil millones en 2004, y un número importante de ellos tendrá acceso a Internet móvil [7]. Existen diversos servicios que se ofrecen para Internet móvil tales como predicción del clima y del tráfico, noticias e información de la bolsa de valores, acceso a bancos, venta de boletos y en algunos lugares se brinda el servicio de pago de multas, etc., todos estos servicios son opciones de comercio electrónico.

Independientemente de las diferencias en contenido, cada servicio que se ofrece mediante comercio electrónico necesita de una seguridad sólida y confiable que incluya un intercambio seguro de datos así como transacciones de pago seguro. Las transacciones móviles seguras tienen cada vez mayor demanda. Estudios de mercado en el mundo han mostrado que la banca móvil es la aplicación más solicitada cuyas peticiones abarcan el 85 % del total de transacciones [33]; por lo que establecer un marco de seguridad será crucial para el desarrollo de los nuevos servicios de comercio móvil.

Por otro lado, la tecnología de redes de diferente alcance como las WLAN han ocupado un nicho muy importante en la industria. Sin embargo, las características inherentes de este tipo de redes pueden, en ciertos casos, ser un punto en contra en términos de seguridad al compararlas con las redes tradicionales y dar como resultado un pobre desempeño para el usuario. Por ejemplo, debido a que al transmitir los datos no se tiene un medio físico para delimitar la señal es posible captar los datos desde cualquier punto dentro del radio

de alcance correspondiente, teniendo de esta manera un acceso completo a la información transmitida, lo que implica que la confidencialidad se vea totalmente comprometida.

Otro de los puntos débiles de la seguridad en las redes inalámbricas es el acceso a la red inalámbrica de algún intruso encubierto como usuario autorizado. Una vez dentro de la red, el intruso puede violar la confidencialidad y la integridad del tráfico de la red al enviar, recibir, alterar o falsificar mensajes. Este último ataque, es un ataque activo, y dentro de las redes WLANs puede llevarse acabo utilizando un adaptador inalámbrico que sea compatible con la tarjeta de red objeto del ataque, o en ambos casos de redes, al utilizar un dispositivo inalámbrico comprometido (robado, por ejemplo) que tenga acceso a la red.

El ancho de banda es otra restricción importante, cada mensaje enviado incrementa el costo de la comunicación. Es por ello que se debe minimizar el intercambio de mensajes cuando se establece una sesión segura. Debido a que los dispositivos son homogéneos, con algunos de ellos de poder de cómputo y memoria limitados, como en los celulares o asistentes digitales, las implementaciones de seguridad no deben hacer cálculos extensivos del lado del usuario. Por lo tanto se requiere reducir la complejidad computacional de los algoritmos a ejecutarse en estos dispositivos.

En todo caso, brindar malos servicios de seguridad, en especial de autentificación, puede traer consecuencias desfavorables a los usuarios de las redes inalámbricas ya que la mejor protección contra el acceso no autorizado es la implementación de mecanismos de autentificación para asegurar que únicamente usuarios autorizados puedan acceder a la red. En la siguiente sección se verán dos protocolos de redes inalámbricas y cómo cada uno de ellos enfrenta el reto de brindar seguridad a sus usuarios.

3.2. IEEE 802.11

En el ámbito de las redes WLANs el estándar que más ha destacado es la especificación de la IEEE: 802.11. Liberada en 1997, hoy es la especificación más utilizada ya que brinda a sus usuarios flexibilidad, simplicidad de uso y efectividad de costos. Este estándar especifica los parámetros de dos capas del modelo OSI: la capa física (PHY) y la capa de control de acceso al medio (MAC).

La capa MAC tiene tres funciones principales: controlar el canal de acceso, mantener la calidad de servicio (QoS) y proveer seguridad. La capa MAC del IEEE 802.11 soporta servicios de seguridad para las aplicaciones de las capas superiores tales como la autentificación y la privacidad, pero la especificación IEEE 802.11 sólo da un método débil de autentificación y para asegurar la privacidad cuenta con una opción llamada *Wired Equivalent Privacy* (WEP) que no ha cumplido con su propósito.

Al inicio, el 802.11 especificaba un bajo índice de transferencia real, hasta de 2Mbps. El estándar ha sido mejorado en dos diferentes especificaciones: el estándar 802.11b conocido como Wi-Fi, que permite, en teoría, una funcionalidad inalámbrica comparable con Ethernet

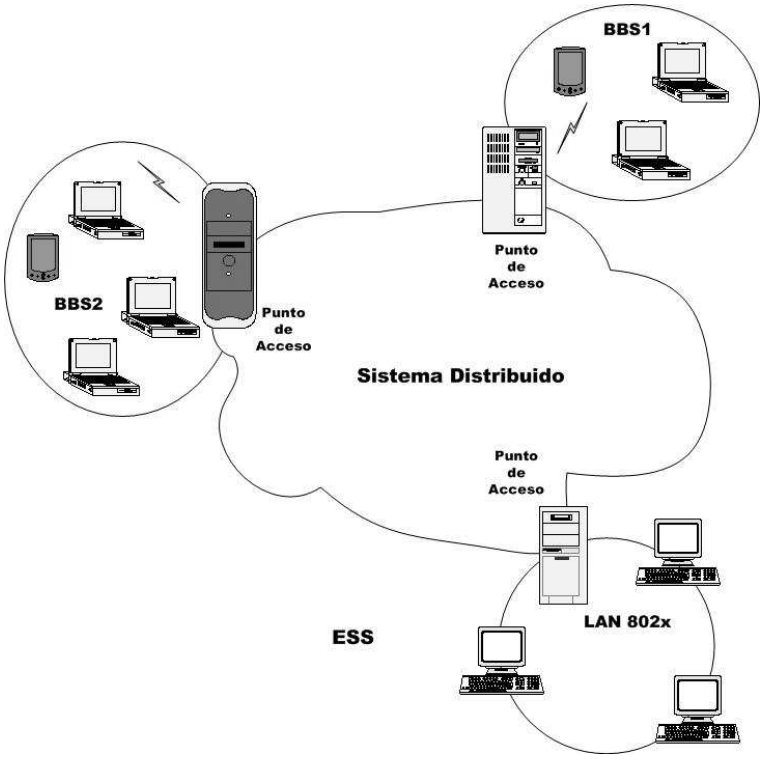


Figura 3.1: Topología de una red IEEE's 802.11

con un índice de transferencia real de hasta 11Mbps en la banda comercial y el estándar 802.11a que permite hasta 54Mbps en la banda industrial, científica y médica.

La arquitectura de una WLAN IEEE 802.11 consiste, generalmente, de un conjunto de servicios básicos (BSS) que se interconectan a un sistema de distribución (DS) para formar un conjunto de servicios extendidos (ESS) como se muestra en la figura 3.1. Cada estación puede transmitir directamente a cualquier otra estación en el mismo BSS (modo ad-hoc). Por otro lado, para transmitir a estaciones pertenecientes a diferentes BSS, las estaciones pasan a través de un punto de acceso (PA) que es una unidad de enlace que implementa ambos protocolos MAC, el de la IEEE 802.11 y el del DS (modo de infraestructura).

3.2.1. Seguridad en 802.11

En una WLAN, una de las mayores preocupaciones es la escucha no autorizada, esto debido a la facilidad con que se captura una transmisión. A pesar de ello y al hecho de que la capa MAC debe ser la encargada de dar servicios de autentificación y privacidad, la especificación del estándar 802.11 provee a las WLANs servicios de seguridad débiles.

- Para la autentificación, el 802.11 especifica dos modalidades: *OSA* (Autentificación de

Sistema Abierto) y la *Autentificación de Llave Compartida*. La modalidad de autentificación OSA está predispuesta a los ataques debido al uso de llaves previamente compartidas, no utiliza la criptografía de llave pública para obtener una llave en un medio inseguro, de hecho, no se utiliza ningún protocolo de intercambio de llaves, lo cual implica en la práctica, que no hay autentificación verdadera. Por otra parte, la autentificación de llave compartida únicamente admite a aquellas terminales móviles que posean una llave cifrada estática. El proceso de autentificación de llave compartida se muestra en la figura 3.2.

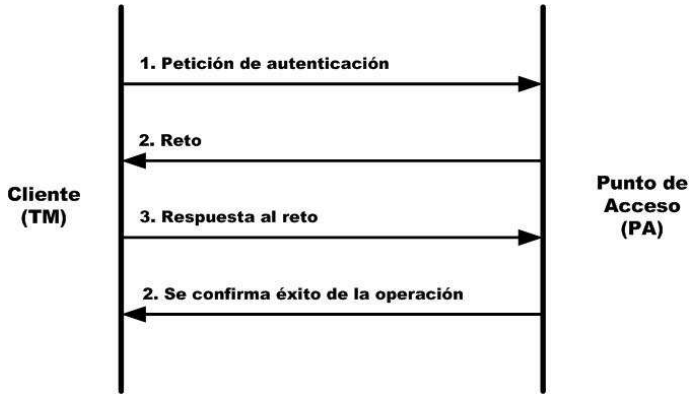


Figura 3.2: Autentificación de Llave Compartida

Cuando el PA autentica a una terminal, todo lo que hace es asegurarse de que la terminal pertenezca a su grupo de dispositivos móviles. El PA no tiene forma de determinar la identidad exacta de la terminal móvil que está requiriendo acceso. Aún más, la mayoría de las implementaciones del 802.11 comparten las llaves a través de los puntos de acceso incrementando el tamaño del grupo en el cual un dispositivo móvil puede ser rastreado [26].

Existe una preocupación todavía mayor con la autentificación del 802.11 en el modo de infraestructura: la autentificación es de un solo sentido, es decir, se provee un mecanismo para que el PA autentifique a las terminales pero no tiene ningún mecanismo para que las terminales autentiquen a la red, es decir, al PA. Esto significa que un nodo impostor puede hacerse pasar por un PA y establecer comunicación con la terminal. Dado que la terminal móvil no puede saber si se está comunicando con un PA auténtico, el nodo impostor tiene acceso a todo lo que la terminal le envía.

Una vez que el PA ha otorgado acceso a una terminal móvil con alguna de estas modalidades, para asegurar la privacidad, es decir, para intercambiar paquetes de datos entre el PA y la terminal móvil (TM) de manera cifrada, el IEEE 802.11 en la capa MAC define a **WEP**, una capacidad opcional.

- Para el cifrado, WEP utiliza una llave secreta que se comparte entre una TM y un PA. Todos los datos enviados y recibidos entre ambas entidades debe ser cifrado utilizando la llave compartida. El estándar no específica cómo se genera la llave compartida, pero

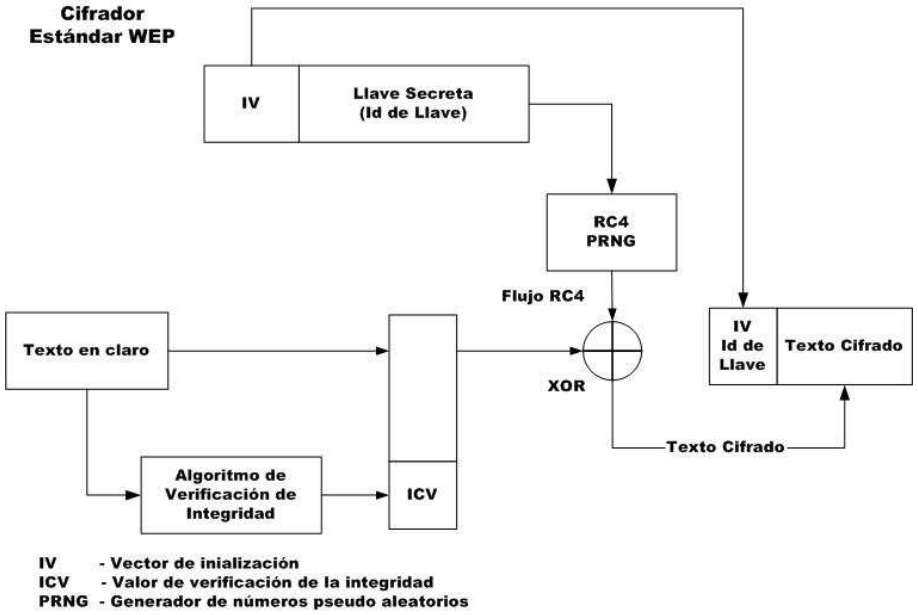


Figura 3.3: Cifrador Estándar WEP

permite un arreglo que contiene una llave única por cada TM. Sin embargo, en la práctica, una llave única es compartida entre todas las TMs y PAs en un sistema [29].

La figura 3.3 muestra cada uno de los pasos del proceso de cifrado de WEP. El cifrado de datos se realiza con una llave secreta de 40 bits (en el 802.11) o una más fuerte de 128 bits (en el 802.11b) y un generador de números pseudo aleatorios (PRNG) de RC4. Al texto en claro se le aplican dos procesos: uno que cifra el texto y otro que lo protege de modificaciones no autorizadas mientras es transmitido. Como parte del proceso de cifrado, WEP prepara una semilla al concatenar la llave secreta con un vector de inicialización (IV) de 24 bits generado de manera aleatoria. El IV alarga la vida de la llave secreta porque la estación puede cambiar el IV en cada trama transmitida. WEP da como entrada el IV al PRNG que produce un flujo de llave igual a la longitud de la trama más un valor de verificación de la integridad (ICV) de 32 bits. Antes de que la transmisión se lleve a cabo, WEP combina todos los flujos con la operación XOR aplicada a cada bit, lo que produce el texto cifrado.

El ICV es un código de verificación que la estación receptora recalcula eventualmente y lo compara con el enviado por la estación remitente para determinar si la transmisión de datos fue alterada. Si la estación receptora calcula un ICV que no corresponde al encontrado en la trama, entonces la estación receptora puede rechazar la trama o marcar al usuario para una posterior auditoria de sus acciones.

WEP únicamente cifra datos entre dos estaciones 802.11. Una vez que las tramas entran al lado no inalámbrico, por ejemplo el acceso entre dos APs, WEP no aplica.

El WEP del 802.11 es una mala solución de seguridad por muchas razones. WEP utiliza el cifrador de flujo RC4 de RSA para cifrar los paquetes de datos en modo sincronizado, es decir, los dos generadores de llaves en los dos puntos en comunicación deben permanecer sincronizados para que funcione de una manera correcta. La pérdida de un sólo bit del flujo de datos cifrado causa la pérdida de todos los datos que siguen al bit perdido. Dado que la pérdida de datos es una situación frecuente en un medio inalámbrico, no es recomendable el uso de un cifrador de este tipo. A pesar de ello, en lugar de seleccionar un cifrador de bloque como AES o DES más apto para un medio inalámbrico, 802.11 trata de resolver el problema de sincronización del cifrador de flujo cambiando las llaves en cada paquete como se muestra en la figura 3.3, de esta manera cada paquete puede ser cifrado/descifrado de manera independiente de los paquetes previos.

El algoritmo RC4 utiliza una llave secreta previamente compartida de 40 bits y un vector de inicialización de 24 bits. Se ha probado que este tamaño de llave es inseguro, desafortunadamente parece ser que el tamaño de llave no es en sí la principal causa de la falla de WEP pues aunque se han hecho versiones con tamaños de llave mucho mayores la falla persiste. De hecho, los ataques a WEP están basados en el diseño del sistema en sí: Fluhrer, Mantin, y Shamir encontraron una falla en el algoritmo de planificación de llaves del RC4 que hace a ciertas llaves de RC4 fundamentalmente débiles. El diseño del ataque se basa en un escucha pasivo que colecciona un número suficiente de paquetes cifrados con llaves débiles para recuperar de esta manera la llave secreta de WEP [2, 26].

En [20] se habla de estos servicios de seguridad y su relación con la calidad de servicio (QoS). Relativamente se ha hecho muy poco para considerar dentro de un mismo diseño a ambos servicios, el de seguridad y el de calidad de servicio en el contexto de las redes inalámbricas y el caso de las redes 802.11 y su capa MAC no es la excepción. Este es otro punto que debe tomarse en cuenta ya que la elección de los mecanismos de seguridad impactan directamente la efectividad de los servicios de calidad y viceversa.

3.3. WAP

En las últimas décadas dos tecnologías han revolucionado el mundo de la información: Internet y los dispositivos móviles. En la intersección de ambas se encuentra WAP (*Wireless Application Protocol*), uno de los protocolos que con mayor fuerza han impulsado y facilitado la navegación por Internet para dispositivos que siendo móviles, típicamente se conectan en red de manera inalámbrica [9, 36].

Como podría preverse y debido a la creciente cantidad de transacciones comerciales que se realizan diariamente en las redes inalámbricas, la seguridad electrónica ha adquirido un papel fundamental para tales medios de comunicación. Dentro de este contexto, el protocolo WAP ofrece que el intercambio seguro de información por Internet –intercambio que incluye el envío de datos altamente confidenciales tales como números de tarjetas de crédito u otras transacciones financieras complejas– sea una práctica común para sus usuarios.

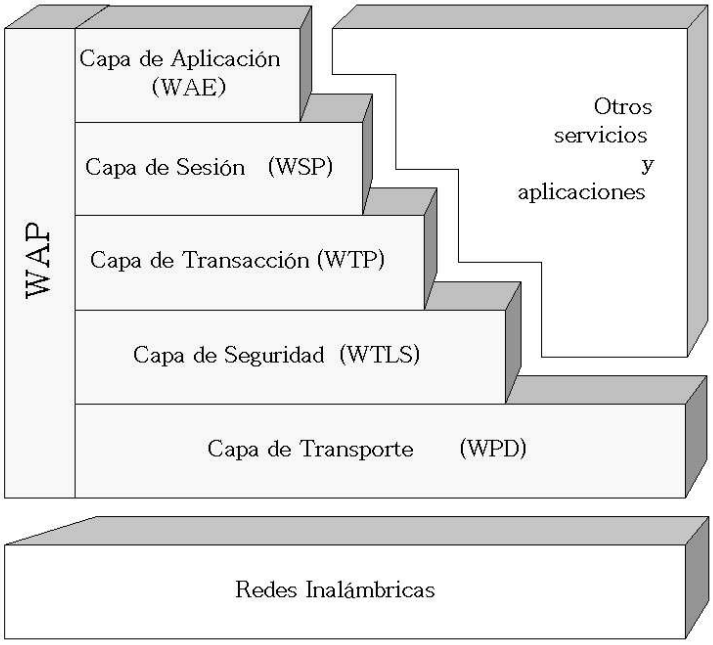


Figura 3.4: Pila de protocolos de WAP

En 1997, debido a la creciente demanda de servicios móviles y a una difícil interacción entre usuarios de diferentes compañías celulares, las principales empresas relacionadas con la telefonía móvil se organizaron para generar y estandarizar un protocolo para dispositivos móviles que fuera independiente de la plataforma y sistema operativo, dicho protocolo tomó el nombre de WAP (Wireless Application Protocol). La organización, conocida como *el foro WAP*, publicó la primera versión de WAP en 1998; en el 2002 se publicó la versión más reciente del protocolo, la versión 2.0.

WAP provee servicios orientados a datos (no de voz) sobre redes inalámbricas WAN/MAN y ha sido diseñado para funcionar sobre redes que toleran un ancho de banda restringido y una latencia relativamente alta.

Uno de los objetivos de WAP es disponer de Internet en los dispositivos móviles, por lo tanto, es natural que se adoptara un enfoque orientado a Internet. WAP está construido por una arquitectura protocolaria, una pila de varios niveles, que deriva y hereda la mayoría de sus características del modelo de referencia ISO OSI (ISO7498). La pila está dividida en cinco niveles: **Capa de aplicación** (WAE - Wireless Application Environment), **Capa de sesión** (WSP - Wireless Session Protocol), **Capa de transacción** (WTP - Wireless Transaction Protocol), **Capa de seguridad** (WTLS - Wireless Transport Layer Security), y **Capa de transporte** (WDP - Wireless Datagram Protocol). La pila de protocolos de WAP se muestra en la figura 3.4.

WAE suministra un entorno de aplicación para el desarrollo y la ejecución de aplica-

ciones y servicios portátiles principalmente mediante WML (Wireless Markup Language) y mediante WMLScript. WSP es una versión binaria de HTTP que define cómo los mensajes deben ser formateados y transmitidos. El control de los mensajes se lleva al cabo por el protocolo WTP. WDP proporciona un protocolo de transmisión sin conexión con soporte para segmentación y ensamblaje de mensajes, su interfaz se adapta a las características de las diferentes redes inalámbricas.

Debajo de la capa de sesión se encuentra la capa WTLS que es la solución al tema de la seguridad planteado en el foro WAP y provee a las capas de nivel superior de WAP con una interfaz de servicio de transporte segura que preserva la interfaz de servicio de transporte por debajo de ella. Adicionalmente, WTLS provee una interfaz para administrar (es decir, crear y terminar) conexiones seguras.

3.3.1. WTLS

En el caso de WAP, los servicios de seguridad son proporcionados por la capa WTLS, que es el protocolo donde se definen los procedimientos y herramientas criptográficas a ser utilizadas para crear, mantener y terminar conexiones de manera segura. Dado que las redes inalámbricas toleran un ancho de banda restringido con una latencia relativamente alta, la relación seguridad contra tiempo de procesamiento y transmisión se vuelve un punto altamente crítico que debe ser tomado en cuenta cuidadosamente por los diseñadores de realizaciones del protocolo WTLS.

En la etapa de negociación, WTLS admite el empleo de únicamente dos sistemas criptográficos de llave pública: RSA y CCE. Los métodos de llave pública son muy poderosos pero implican un costo computacional elevado, siendo por ello que se considera a esta etapa como la más costosa dentro del proceso de seguridad. Diversas fuentes [23, 36] afirman que CCE ofrece el mismo nivel de seguridad que RSA al precio de tamaños de llave aproximadamente diez veces menores, lo que implicaría un proceso de negociación potencialmente más *económico*. A pesar de ello, la mayoría de las implementaciones de WTLS han optado por RSA debido, entre otras razones, a su mayor difusión en el mercado informático y de redes alámbricas.

Varios trabajos reportan el desempeño del protocolo WTLS en diversas plataformas. Por ejemplo, Herwono y Liebhardt en [13, 14] presentan una evaluación de una simulación hecha de los protocolos de Registro y Negociación de WTLS. En ese trabajo se analizan los niveles de seguridad con diferentes tamaños de llave y de mensajes, encontrándose un mejor comportamiento con CCE. Por otro lado, Levi y Savas en [23] hacen un estudio analítico del desempeño de los dos sistemas criptográficos de llave pública elegibles para ser incluidos en realizaciones del protocolo WTLS. En dicho trabajo se consideran dos tipos de protocolos de negociación: autentificación por parte del servidor y autentificación mutua. En ambos casos se concluye que CCE debería brindar un mejor desempeño que RSA. Sin embargo ninguna de las investigaciones mencionadas [23, 13, 14] incluye una implementación real del protocolo WTLS que permita corroborar fehacientemente sus conclusiones y/o predicciones.

WTLS es similar en espíritu y en arquitectura al protocolo *TLS*, que es el estándar del IETF (Internet Engineering Task Force) para una navegación en Internet segura, y que es el sucesor del protocolo de seguridad de Internet *SSL* 3.0 (Secure Socket Layer). WTLS incorpora algunas características nuevas a TLS (por ejemplo el soporte de datagramas, un proceso de negociación optimado, la actualización dinámica de llaves, etc.), además está diseñado para funcionar sobre redes que toleran un ancho de banda restringido y una latencia relativamente alta. WTLS es un nivel del protocolo WAP y está proyectado para funcionar tanto orientado a la conexión como con protocolos de transporte de datagramas tales como *UDP* (User Datagram Protocol), o con WDP en la ausencia de UDP.

El objetivo principal de WTLS es el de proporcionar a las aplicaciones *Privacidad*, *Integridad* y el servicio de *Autentificación*. WTLS garantiza estas propiedades utilizando el mismo esquema de criptografía que TLS.

WTLS, como se ha dicho antes, funciona también con protocolos de transporte de datagramas como UDP o WDP. Tales protocolos se caracterizan por el hecho de que los datos viajan de manera completamente independiente entre ellos y además pueden perderse, llegar en desorden o llegar duplicados. Puede ocurrir entonces que en la fase de negociación entre un cliente y un servidor no sea posible tener un buen resultado si, por ejemplo, la petición de conexión segura inicial del cliente no llega nunca al servidor o si la aceptación de un certificado de parte de una de las dos entidades no llega nunca a la otra. Para poder soportar los datagramas se introducen entonces en WTLS una serie de mecanismos que hacen frente a tales eventualidades.

El protocolo WTLS tiene una arquitectura cliente servidor, las conexiones aseguradas con WTLS son siempre iniciadas por el cliente, que debe ser visto como el dispositivo móvil. Conceptualmente es útil pensar en el componente WTLS de WAP como una máquina de estados. En particular para superar los problemas mencionados anteriormente el WTLS se basa en esta máquina de estados vista de forma asimétrica (esto es, una para el cliente y otra para al servidor), la intención de las dos máquinas es permitir la sincronización de los datos que las dos entidades intercambian en una conexión segura.

La capa WTLS, a su vez, está compuesta de dos niveles lógicos como se muestra en la figura 3.5: un nivel inferior que contiene al llamado protocolo de **Registro** y uno superior sobre el cual se encuentran reunidos otros tres protocolos: **protocolo de Alerta, protocolo de Especificación de Cambio de Cifrado, protocolo de Negociación**. La finalidad de esta división es administrar de manera separada y sincronizada las fases relativas a una realización completa de las sesiones seguras [11].

El *protocolo de aplicación* es la interfaz para las capas superiores. El *protocolo de especificación de cambio de cifrado* indica que a partir del momento en que este protocolo es llamado se utilizaran los métodos de cifrado acordados para codificar los mensajes. El *protocolo de alerta* mantiene los avisos relativos a la ocurrencia de problemas eventuales que se tengan en la fase de gestión de una sesión de seguridad estableciendo según el tipo de evento ocurrido tres niveles de aviso *fatal*, *crítico* y de *advertencia*.

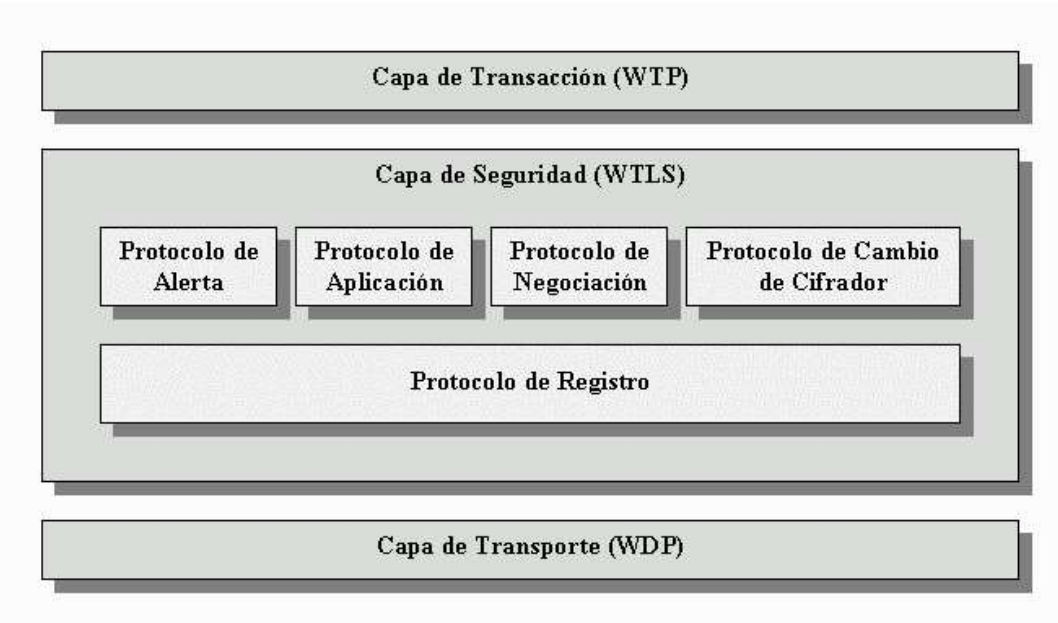


Figura 3.5: Los niveles del WTLS

El *protocolo de registro* administra la fragmentación de los mensajes provenientes de los niveles superiores, es allí donde se cifran las tramas utilizando algoritmos de cifrado de bloques. Además, en este nivel se hace uso del MAC (*Código de Autentificación de Mensaje*) para la verificación de la integridad del mensaje enviado.

En el *protocolo de negociación* se producen los parámetros criptográficos de una sesión segura y es allí donde la autentificación de las partes involucradas en la comunicación se lleva a cabo.

3.4. Protocolo de Negociación de WTLS

En el protocolo de Negociación se acuerdan los parámetros criptográficos para establecer o reiniciar una conexión segura entre un cliente WAP y una pasarela WAP (servidor WAP). Cuando un cliente y un servidor inician una comunicación, ellos deciden que versión del protocolo usarán, seleccionan los algoritmos criptográficos, y utilizan técnicas de criptografía de llave pública para autenticarse mutuamente y generar, finalmente, la llave de sesión compartida [9, 32]. Esta llave de sesión secreta será posteriormente utilizada por el protocolo de registro para cifrar, con algún algoritmo de llave simétrica, la información.

El protocolo de Negociación de WTLS es equivalente al de SSL pero existen diferentes variantes de tal protocolo, por ejemplo, una *negociación completa*, una *negociación de conexión continuada*, una *negociación óptima*, etc. Grosso modo la figura 3.6 representa los diversos pasos que ocurren en el protocolo de Negociación completo.

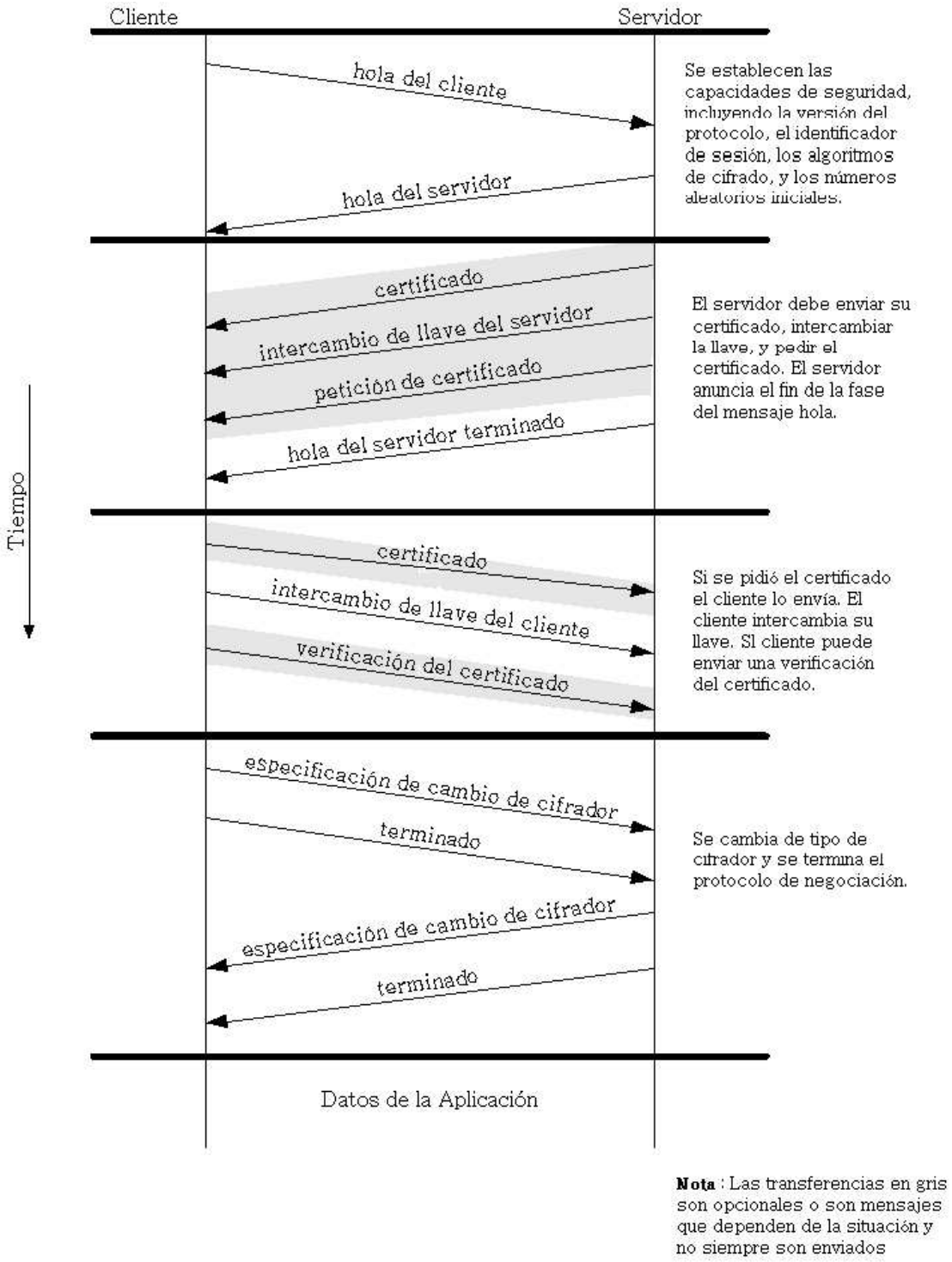


Figura 3.6: Protocolo de Negociación Completo

Dependiendo de las opciones tomadas en el protocolo de negociación se pueden discernir tres clases de implementaciones de WTLS definidas en la especificación de WAP, éstas son:

WTLS Clase 1: Únicamente brinda privacidad e integridad de datos mediante un intercambio de llaves anónimo sin autentificación.

WTLS Clase 2: Brinda privacidad e integridad de datos además de autentificación WAP a nivel del servidor. Aquí, la autentificación del servidor se basa en certificados. La llave del servidor puede ser anónima o autenticada, la llave del cliente es anónima.

WTLS Clase 3: Brinda privacidad e integridad de datos además de autentificación WAP tanto del servidor como del cliente. Aquí, la autentificación del servidor y el cliente se basa en Certificados. Tanto la llave del cliente como del servidor puede ser anónima o autenticada.

Los algoritmos criptográficos aceptados en el estándar incluyen, para los *algoritmos de intercambio de llaves de sesión*, a ECDH, RSA y Diffie-Hellman. Los esquemas de *firma digital* utilizados son RSA y ECDSA. Los algoritmos de *cifrado de bloques* que funcionan en modo CBC (*Cipher Block Chiang*) incluyen DES, Triple DES, RC5, e IDEA. Para las funciones *hash* se contempla MD5 y SHA-1.

3.4.1. Autentificación en WTLS

Dentro de los protocolos de seguridad, el de autentificación es el inicio de una comunicación segura. Como se analizó en el capítulo anterior, por autentificación se entiende cualquier método que permite comprobar de manera segura alguna característica sobre un objeto. Dicha característica puede ser su origen, su integridad, su identidad, etc. En la autenticación de mensaje, se busca garantizar la procedencia de un mensaje conocido, de forma que se pueda asegurar que no es una falsificación.

Para los dispositivos móviles dentro de redes celulares con acceso a Internet, el protocolo WTLS brinda una alternativa de autentificación en la clase 2 y clase 3. La especificación de WTLS trabaja con certificados X.509 para ser compatible con las aplicaciones existentes en la infraestructura de Internet y con un formato de certificado propio para las características de los dispositivos móviles, los certificados WTLS.

Los **certificados WTLS** están basados en los certificados X.509 vistos en la sección 2.4.2. A diferencia de tales certificados, los certificados WTLS han optimizado el número de campos a los estrictamente necesarios para proveer la información requerida para autenticar a las entidades.

Los dispositivos móviles, es decir, asistentes digitales, teléfonos celulares, etc., deben de ser capaces de generar y procesar certificados de tamaño al menos de 700 bytes; y si

aceptan autentificación del servidor con certificados X.509 deben de ser capaces de procesar certificados de tamaño al menos 1000 bytes y certificados de ACs de al menos 2000 bytes [8].

Existen diferentes perfiles definidos por WAP que se basan en los perfiles de certificados del grupo de trabajo PKIX del IETF [8]. Los perfiles que define son para certificados de usuario, servidor y de AC:

- Un **certificado WTLS de servidor** WAP es un certificado que autentica la identidad de un sitio WAP a los micro-navegadores de los dispositivos móviles. Cuando un cliente (usualmente el micro-navegador de un usuario) quiere enviar información confidencial al servidor WAP, el cliente accede al certificado digital del servidor. El certificado, que contiene la llave pública del servidor, está firmado por una AC reconocida y sirve para autenticar la identidad del servidor.
- Un **certificado AC** es un certificado que identifica a una autoridad Certificadora. Este tipo de certificados son idénticos a los demás certificados digitales excepto que están firmados por la propia autoridad certificadora. Los certificados AC se usan para determinar cuando confiar en los certificados expedidos por la AC.
- Los **certificados de usuario** tienen el fin de autenticar al cliente; el perfil del certificado se almacena en el cliente WAP.

Cuando un certificado de servidor se presenta a un cliente, el cliente utiliza el certificado de la AC para determinar si confía o no en el certificado del servidor. Si el certificado del servidor es válido, la sesión WTLS continúa. Si el certificado del servidor no es válido, el certificado del servidor es rechazado y la sesión WTLS se detiene.

Los campos de los certificados para cualquiera de estos perfiles son:

- Número de Serie del Certificado. Las ACs no deben de utilizar números de serie mayores a 8 bytes.
- Identificador del algoritmo de firma. Los únicos algoritmos definidos para este perfil son SHA1 con RSA y ECDSA.
- Nombre del Emisor. Se deben reconocer el nombre del país, el nombre de la organización, el componente del dominio, entre otros atributos.
- Nombre del Usuario Se deben distinguir todos los mismos atributos del nombre del emisor.
- Llave Pública del Usuario Las únicas llaves definidas para usarse en esta especificación son de RSA de 1024 o mayores y CE de 160 bits o mayores.
- Extensiones de los Certificados En el estándar definido en [8] se especifican tales extensiones y si su uso es crítico o no.

Capítulo 4

Diseño e Implementación del Prototipo

Algo que atañe a los dispositivos móviles es la relación seguridad contra tiempo de procesamiento y transmisión, por ello se deben considerar las mejores alternativas criptográficas y tecnológicas para llevar al cabo el diseño e implementación de los servicios de seguridad en las redes inalámbricas, y en el caso de los mecanismos de autentificación esta no es una excepción.

Con el propósito de establecer una sesión segura el protocolo de Negociación del WTLS (*Wireless Transport Layer Security*) admite el uso de únicamente dos criptosistemas de llave pública: RSA y Criptosistemas de Curvas Elípticas (CCE). Los algoritmos para el intercambio de llaves de sesión permitidos por el estándar WTLS incluyen ECDH, RSA y Diffie-Hellman. En esta ocasión se ha desarrollado un prototipo que simula la funcionalidad del protocolo de Negociacón de WTLS para evaluar el desempeño de cada uno de estos criptosistemas. El protocolo de Negociación de WTLS, como se ve en el capítulo anterior, interactúa con la capa WDP que es el protocolo de transmisión de los mensajes y quien hace llegar la información al protocolo WTLS; en esta simulación, la implementación del protocolo WTLS esta basada en la contraparte alámbrica, TCP/IP. El cliente y el servidor se comunican utilizando tal protocolo de transporte mediante sockets.

En este capítulo se describe cada fase del diseño del prototipo implementado. En la primera sección se detalla la arquitectura de los dos sistemas realizados, donde se incluyen los diagramas de contexto de los sistemas y los diagramas de flujo de los procesos principales.

4.1. Arquitectura del Sistema

Para la elaboración del prototipo del protocolo de Negociación de WTLS se diseñaron dos sistemas: el sistema *ClienteWTLS* y el sistema *ServidorWTLS*. El modelo de cada uno

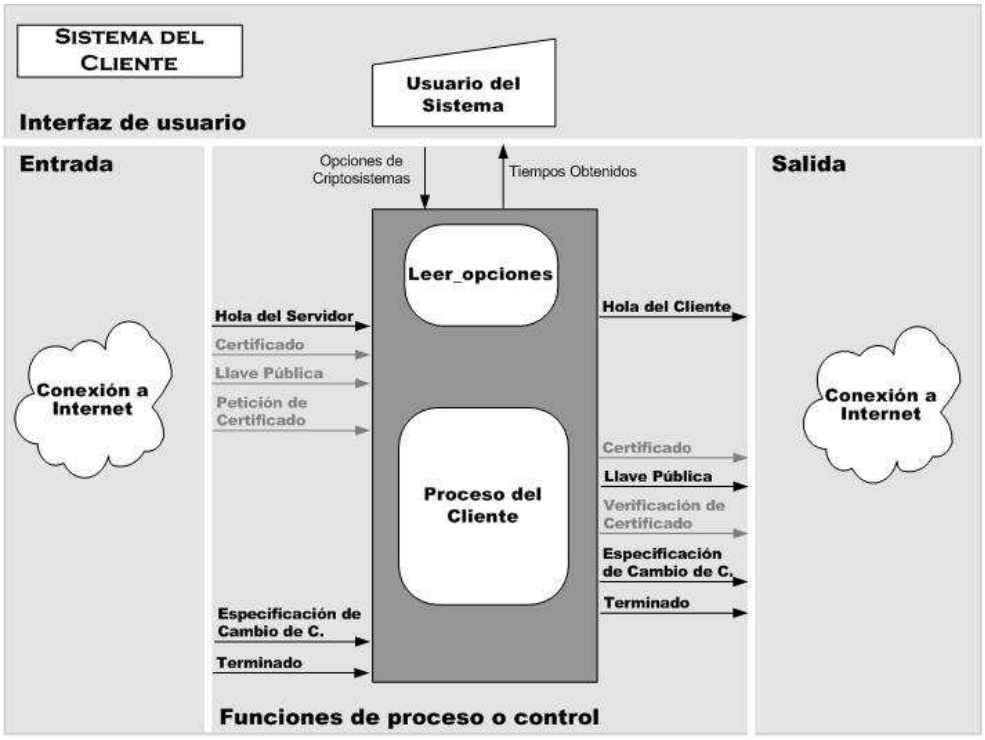


Figura 4.1: Diagrama de contexto de la arquitectura del sistema Cliente

de los sistemas se desarrolló utilizando una *plantilla de arquitectura* donde se asignan los elementos del sistema a cada región: interfaz de usuario, entrada del sistema, salida del sistema, y las funciones de proceso o control. Este tipo de arquitectura sigue el modelo *entrada-proceso-salida*, y muestra la relación que existe entre los componentes del sistema [30].

El diagrama de contexto de la arquitectura del sistema *ClienteWTLS* se muestra en la figura 4.1 y esquematiza los siguientes elementos:

- La *función de control*, que se ha dividido en 2 procesos principales que se comunican entre sí, el proceso *1.1 leer_opciones* y el proceso *1.2 cliente*.
- La *entrada* al sistema es a través de un puerto de red, la información proveniente del servidor WAP que se espera obtener son los mensajes involucrados en el proceso de Negociación tales como el mensaje *hola del servidor*, la *llave del servidor* o el *certificado*, etc.
- De igual manera, la *salida* del cliente se dirige al mismo puerto de comunicaciones, la información enviada al servidor WAP son los mensajes *hola del cliente* que inicia la negociación; dependiendo del tipo de protocolo WTLS ejecutado se enviará la *llave del cliente* o el *certificado* del cliente; etc.

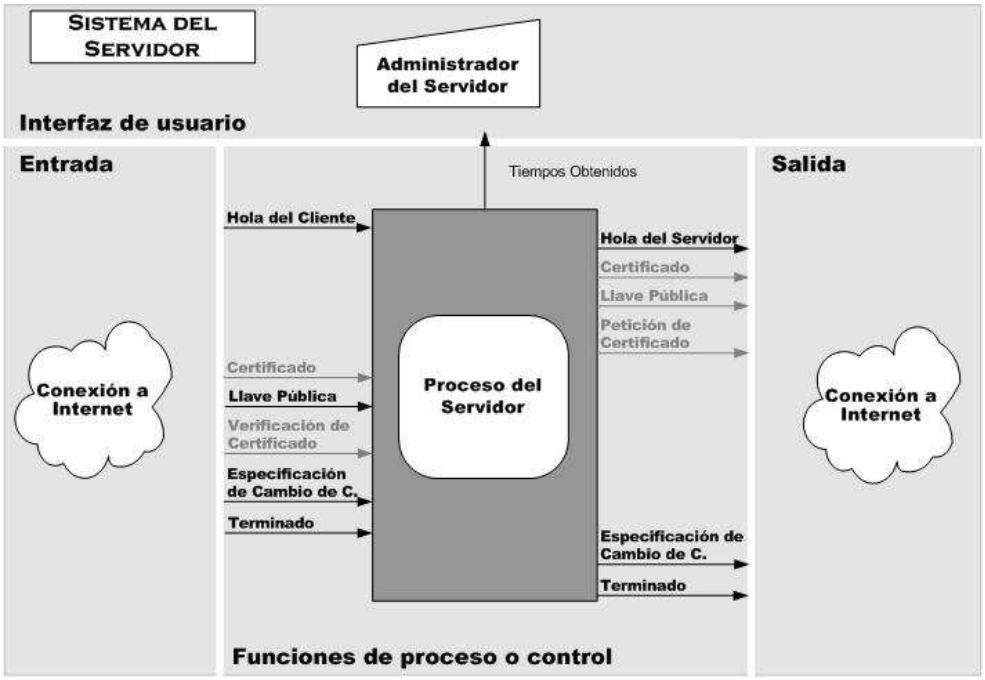


Figura 4.2: Diagrama de contexto de la Arquitectura del Sistema Servidor

El proceso *cliente* será el encargado de dar respuesta y generar los mensajes de entrada y salida. La información opcional se indica con color gris y la información obligatoria con color negro.

- La *interfaz de usuario* se controla por el proceso *leer_opciones*, se despliega al usuario las opciones disponibles del sistema y una vez finalizado el protocolo de negociación, muestra los tiempos de ejecución obtenidos.

La figura 4.2 muestra el diagrama de contexto de la arquitectura del sistema *Servidor WTLS*, los elementos de este diagrama son:

- La *función de control*. Está compuesta únicamente por un proceso *2.1 servidor* que es el encargado de recibir las peticiones de los clientes, procesarlas y darles respuesta.
- La *entrada* al sistema. Ocurre a través de un puerto de red, la información proveniente son los mensajes que envían los clientes: el mensaje *hola del cliente*, la *llave del cliente* o el *certificado* del cliente, etc.
- La *salida* del servidor. Vincula al puerto de comunicaciones de la red, la información enviada al cliente WAP son los mensajes *hola del servidor* que dan respuesta a la petición de inicio de negociación; el mensaje que contiene a la *llave del servidor* o el mensaje que contiene al *certificado*, etc.

Del mismo modo que en el sistema *Ciente WTLS*, la información opcional se indica con color gris y la información obligatoria con color negro.

- La *interfaz de usuario* sólo despliega al administrador del servidor los tiempos de ejecución obtenidos.

En el siguiente apartado se describe, con ayuda de diagramas de flujo, a los procesos involucrados a las funciones de control de ambos sistemas.

4.1.1. Especificación de los Componentes

De los diagramas de contexto se extraen los procesos que componen a los sistemas, a continuación se describe a detalle la función de cada uno de ellos.

1.1 Proceso leer_opciones. Es el proceso encargado de mostrar, al usuario del prototipo, la información de los algoritmos criptográficos que están disponibles tanto para firma digital como para protocolo de generación de llave de sesión. Así mismo, este proceso se encarga de obtener la elección del usuario y comunicarla al proceso *cliente*. Una vez terminado el proceso de generación de llave de sesión, el proceso *cliente* comunica al usuario los tiempos de ejecución obtenidos.

1.2 Proceso Cliente

El primer paso en el proceso del cliente es generar el mensaje *Hola del Cliente* con las especificaciones dadas por el usuario, tal como se muestra en el diagrama de flujo de la figura 4.3. Los parámetros que se requieren para la generación del mensaje inicial por parte del usuario son: la *definición del algoritmo de firma y de acuerdo de llave* y las *opciones de los parámetros a utilizarse* tomados del estándar WTLS [9]. Una vez generado el mensaje *Hola del Cliente* se produce una versión compacta que será enviada al servidor, donde sólo se encuentran los datos básicos del mensaje.

Después de enviar el mensaje solicitando el inicio de comunicación segura con el servidor, el cliente entra a un estado de espera que termina cuando se reciba el mensaje *HolaDelServidor* que confirme la petición.

Dependiendo de la clase de implementación de WTLS se pueden tener los siguientes casos:

Clase 1. El cliente genera en este momento, el par de llaves para la sesión actual, la *pública* y la *privada*. Todas estas variables se generan a partir de los parámetros criptográficos indicados por el cliente y confirmados por el servidor. El cliente, después de haber generado el par de llaves, entra a un estado latente donde espera recibir la llave pública del servidor, una vez recibida, envía su propia llave pública.

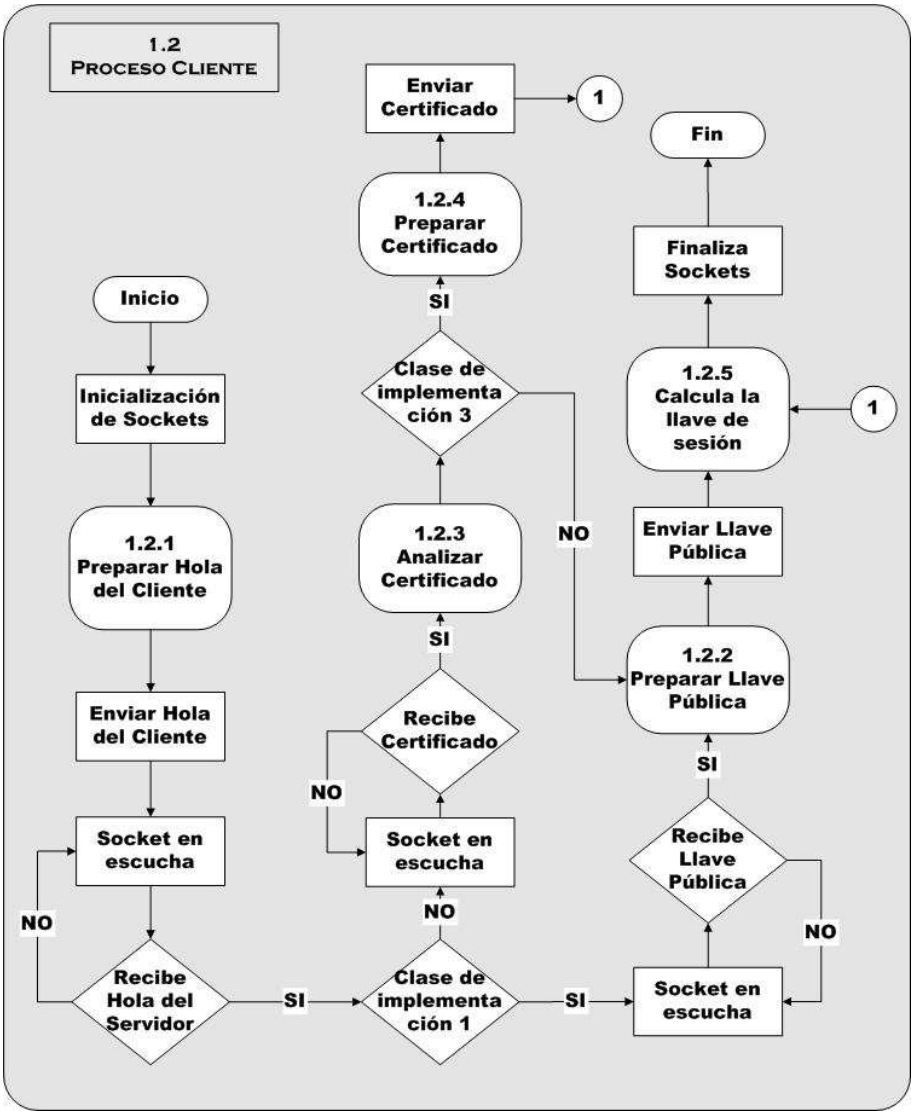


Figura 4.3: Diagrama de Flujo del Proceso del Cliente

Clase 2. El cliente espera el certificado del servidor, una vez recibido verifica la autenticidad del certificado y extrae la llave pública del servidor. Prepara el par de llaves propias para la sesión actual, la *pública* y la *privada*, generadas a partir de los parámetros criptográficos indicados por el cliente y confirmados por el servidor. El cliente, en este momento envía su llave pública al servidor.

Clase 3. Al igual que la clase 2, el cliente espera el certificado del servidor, una vez recibido verifica la autenticidad del certificado y extrae la llave pública del servidor. Pero esta vez envía el certificado del cliente y obtiene su llave pública de este certificado, al cual corresponde a los parámetros criptográficos sugeridos por el usuario y confirmados por el servidor. Recupera la llave privada que hace juego con la llave pública ubicada en el certificado del repositorio privado.

Cuando ambas llaves públicas han llegado a su destino el cliente genera la llave de sesión a ser utilizado por otro protocolo de WTLS en una etapa posterior de cifrado.

2.1 Proceso Servidor

El servidor inicia en un estado de espera, como indica el diagrama de flujo para este proceso de la figura 4.4. El cliente desencadena la comunicación al enviar el mensaje básico *Hola del Cliente*. Con tal información el servidor será capaz de reconstruir el mensaje hola original del cliente, entonces verifica que la propuesta del cliente para iniciar la generación de llaves y los parámetros sugeridos sean válidos, de ser así, forma un *Hola del Servidor* confirmando la petición y lo envia en un buffer al cliente.

Una vez enviado el mensaje *Hola del Servidor* puede pasar una de tres opciones:

Clase 1. El servidor genera el par de llaves para esa sesión, la *pública* y la *privada*. Todas estas variables se forman a partir de los parámetros criptográficos indicados por el cliente. En ese momento el servidor envía su llave pública y espera la llave pública del cliente.

Clase 2. El servidor obtiene del repositorio el certificado que corresponde a los parámetros indicados por el cliente, lo envía y extrae de él la llave pública que utilizará en esa sesión. La siguiente acción es recuperar del repositorio la llave privada que corresponde a la obtenida del certificado. El servidor entra a un estado de espera del cual sale al recibir la llave pública del cliente.

Clase 3. El servidor obtiene del repositorio el certificado que corresponde a los parámetros indicados por el cliente, lo envía y extrae de él la llave pública que utiliza en esa sesión. La siguiente acción es recuperar del repositorio la llave privada que corresponde a la obtenida del certificado. El servidor entra a un estado de espera de donde espera recibir el certificado del cliente, verifica la autenticidad del certificado y extrae la llave pública del cliente.

Cuando el servidor ha obtenido las llaves pública y privada propias y ha recibido, de manera directa o por medio del certificado, la llave pública del cliente, el servidor

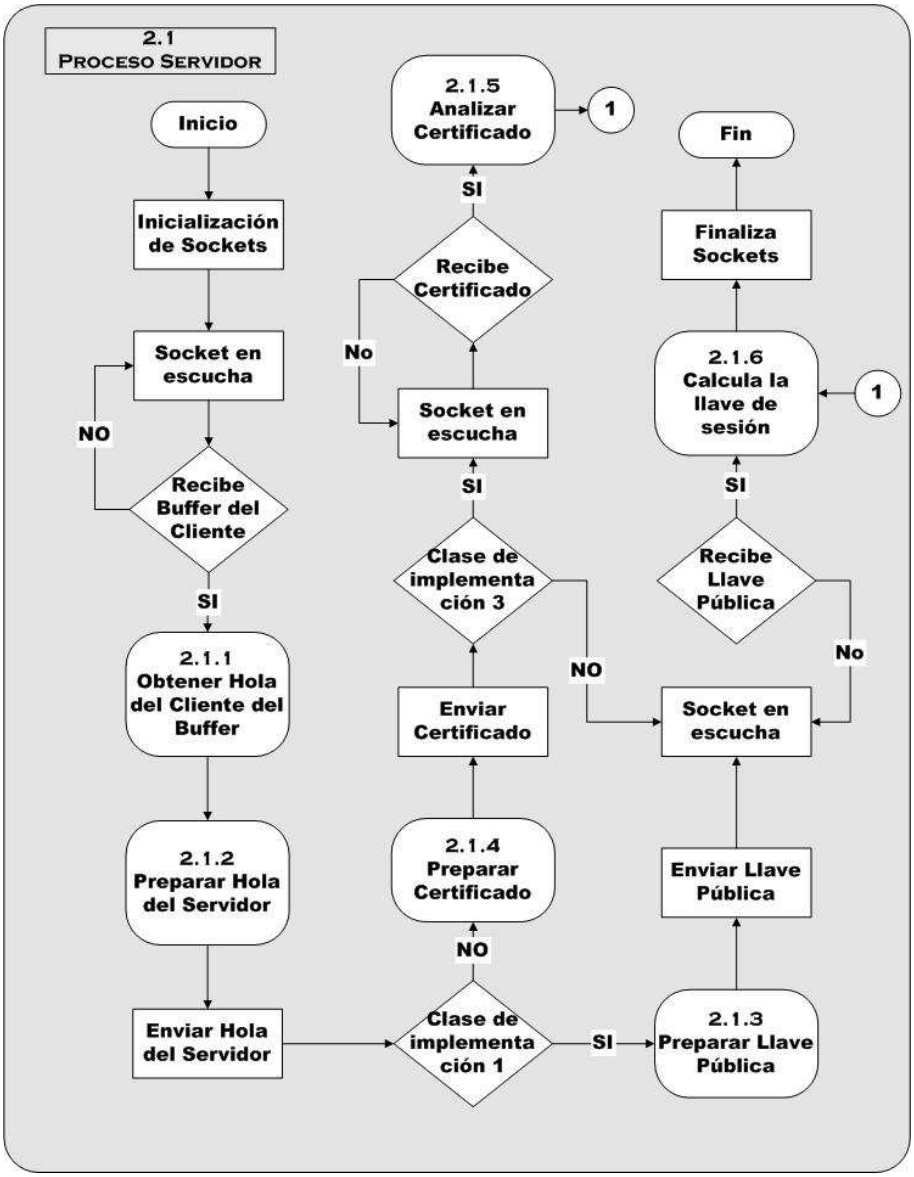


Figura 4.4: Diagrama de Flujo del Proceso del Servidor

genera la llave de sesión que será utilizada en la etapa de cifrado por el protocolo de Registro de WTLS.

Los procesos descritos en esta sección están conformados por distintos sub-procesos, tal como se muestra en los diagramas de flujo correspondientes. Pero, antes de detallar su funcionamiento, se enumeran las estructuras de datos necesarias para la implementación de los sistemas.

4.2. Especificación de las Estructuras de Datos

Para llevar al cabo los procesos especificados en la sección anterior, es necesario definir las estructuras de datos que representan los mensajes involucrados en el protocolo de negociación.

El mensaje *Hola del Cliente* corresponde a la estructura de datos *ClientHello* y el mensaje *Hola del Cliente* a la estructura *ServerHello* de la figura 4.5, allí se muestra el desglose de las estructuras de datos utilizadas en ambos casos.

Dentro de la estructura *KeyExchangeIds* se especifican los parámetros del sistema de generación de llave de sesión y el tipo de autentificación a utilizarse. Si damos seguimiento a esta estructura, iniciando en la figura 4.5 y continuando en la figura 4.6 se observa que *KeyEschangeSuite* indica el tipo de combinaciones posibles:

```
{ NULLKES, ECDH_anon, RSA_anon, ECDH_oneway, RSA_oneway, ECDH_twoway, RSA_twoway }
```

Los parámetros del criptosistema a emplearse están depositados en *ParameterSet*. En el caso de curvas elípticas estos parámetros toman la forma especificada en la figura 4.7 donde se incluye la curva a ser utilizada, el campo, el punto base, el orden y el cofactor. Para el caso de RSA estos parámetros se especifican en la figura 4.8. Es necesario definir la llave pública, la llave privada y el módulo dentro de la estructura *RSAParameters*.

Para la firma digital, los esquemas contemplados son:

```
{ ANONYMOUS, ECDSA_SHA1, RSA_SHA1 }
```

contenidos en *SignatureAlgorithm*, que a su vez forma parte de la estructura *SignatureInfo*. Otros componentes son la autoridad certificadora y el tamaño del certificado, este último indicará qué tamaño de buffer esperan recibir tanto el cliente como el servidor en el mensaje *Certificado*.

Para los mensajes *Llave Pública* del cliente y del servidor la estructura incluye:

- La especificación de los parámetros, con etiqueta 2 en la figura 4.6.
- El identificador del algoritmo de intercambio de llave especificado en 1 dentro de la figura 4.6.

- Los parámetros del criptosistema de llave pública a utilizarse especificado en 4.7 o 4.8.

Todas las estructuras de datos descritas en esta sección se especifican en el estándar WTLS ([9]).

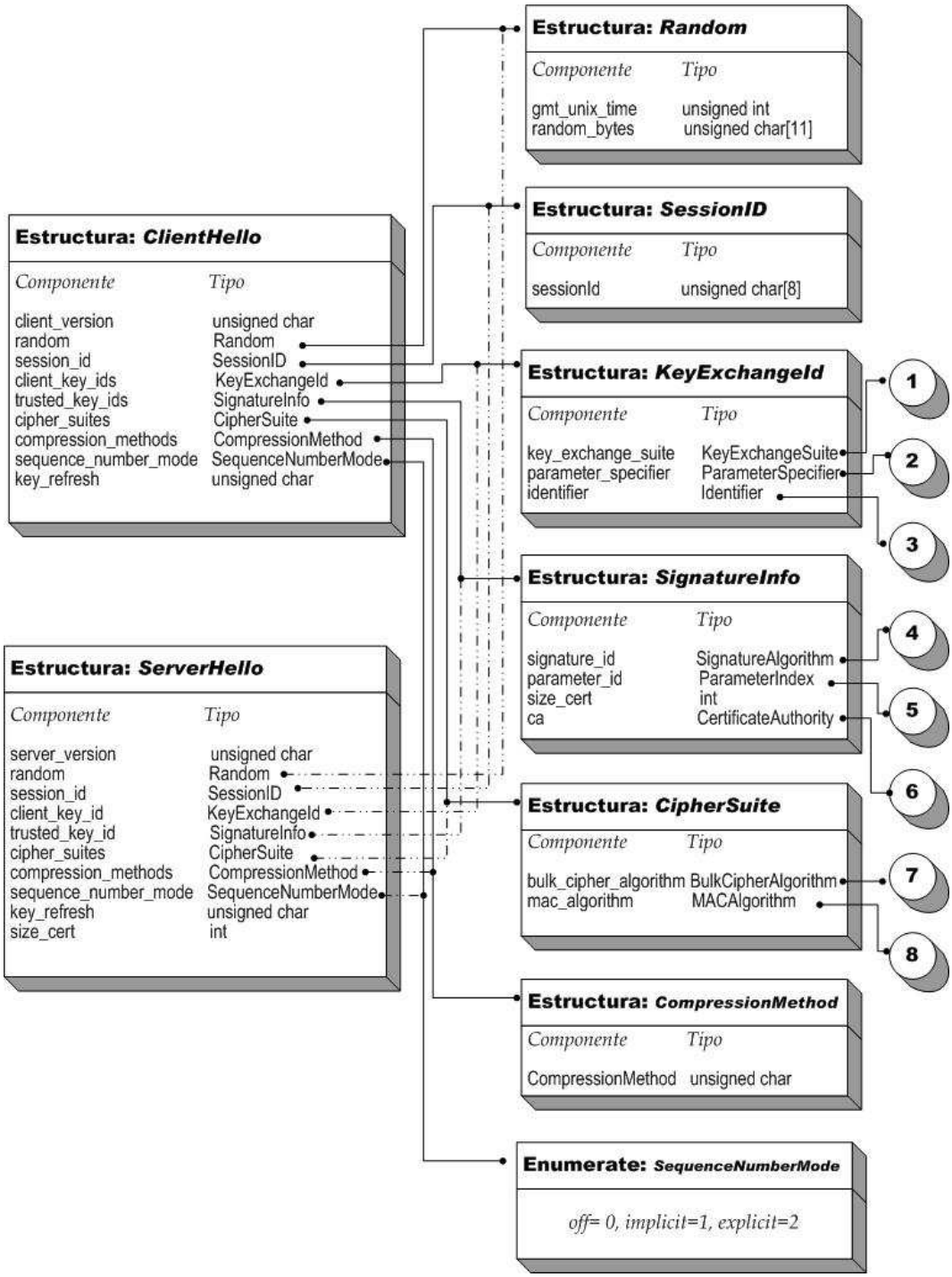


Figura 4.5: Estructuras de datos *HelloClient*, *HelloServer* y sus componentes

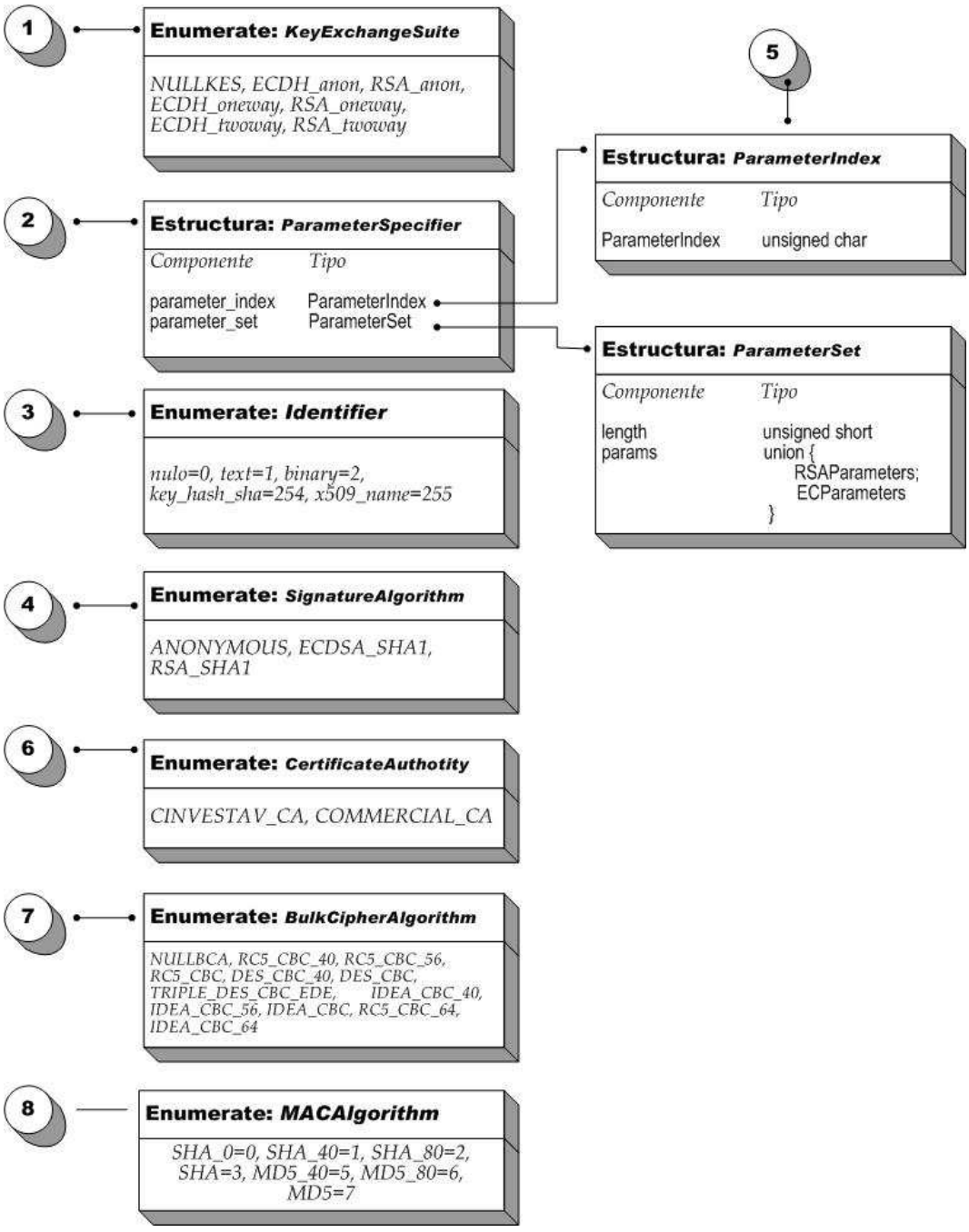


Figura 4.6: Estructuras de Datos Auxiliares

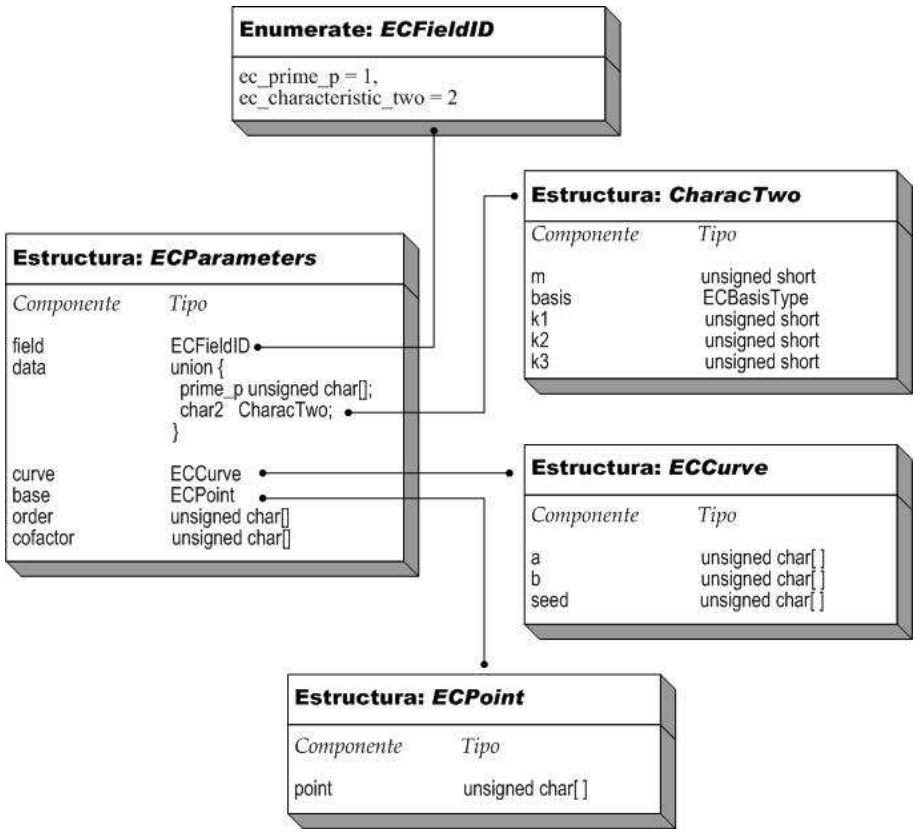


Figura 4.7: Estructuras de Datos de las Curvas Elípticas

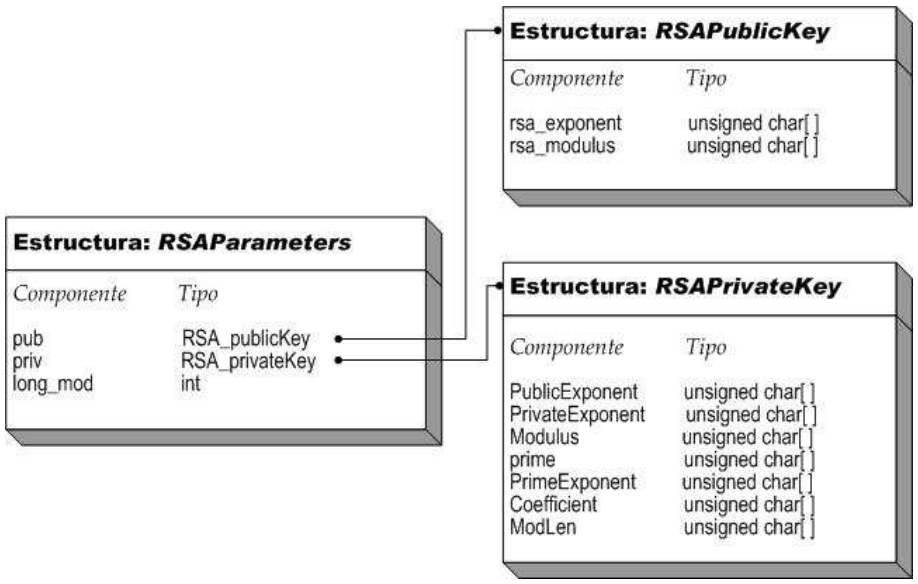


Figura 4.8: Estructuras de Datos de RSA

4.3. Especificación de los Algoritmos

Una vez especificadas las estructuras de datos más relevantes, se da una breve reseña de los subprocesos contenidos en los procesos *cliente* y *servidor*.

- El proceso *1.2.1 Preparar Hola del Cliente* toma como parámetro de entrada las opciones dadas por el cliente e inicializa los campos de la estructura *ClientHello* como *client_version, session_id, random*, etc. En el caso del campo key_exchange_suite se tiene lo siguiente:

```
switch(key_exchange_suite)
      case ECDH_anon:
      case ECDH_oneway:
            leer los parámetros de la curva del archivo;
            el tamaño del certificado del servidor es 0;
            se asigna como AC a CINVESTAV_CA;
            break;
      case RSA_anon:
      case RSA_oneway:
            se asignan los parámetros necesarios para RSA;
            el tamaño del certificado del servidor es 0;
            se asigna como AC a CINVESTAV_CA;
            break;
      case ECDH_twoway:
      case RSA_twoway:
            se abre el certificado correspondiente;
            se asigna como AC a CINVESTAV_CA;
            break;
      case NULLKES:
      default:
            //  no habrá generación de secreto
      }
```

- El proceso *2.1.1 Preparar Hola del Cliente del Buffer*

  El servidor, al recibir una versión compacta del mensaje *Hola del Cliente* debe, a partir de la información recibida, llenar los campos de la estructura *CientHello*. Ya que la versión compacta indica el algoritmo de intercambio de llave y el índice de los parámetros correspondientes al estándar, puede a su vez recuperar la información de su repositorio, para generar una copia idéntica del dicho mensaje.

- El proceso *2.1.2 Preparar Hola del Servidor* A partir del mensaje *Hola del Cliente* el servidor verifica que los parámetros para iniciar una sesión segura indicados por el cliente sean correctos. Una vez verificada esta condición, se genera el mensaje *Hola del Servidor* completando la estructura *ServerHello*.

- El proceso *1.2.2 (2.1.3) Preparar Llave Pública*. Para generar la llave pública y privada necesarias para el acuerdo de la llave de sesión, tanto el cliente como el servidor realizan lo siguiente:

```
switch(key_exchange_suite)
      case ECDH:
            iniciar la memoria necesaria para almacenar las llaves;
            generar la llave pública y privada para curvas elípticas;
            break;
      case RSA:
            iniciar la memoria necesaria para almacenar las llaves;
            generar la llave pública y privada para RSA
            break;
      case NULLKES:
      default:
            //  no habrá generación de secreto
      }
```

- El proceso *1.2.3 (2.1.5) Analizar Certificado* El análisis de certificados está basado en la figura 2.2. El análisis inicialmente extrae los siguientes campos:

  1. La firma digital del certificado.
  2. El identificador del algoritmo con el que fue firmado.
  3. El identificador de la autoridad certificadora que lo firmó

  En base a esa información se realiza la verificación del certificado con el algoritmo indicado (RSA o ECDSA) y con la llave pública de la autoridad certificadora correspondiente. Si es aceptado el certificado, se extrae del certificado TBS lo siguiente:

  1. La llave pública del usuario.
  2. El identificador del algoritmo de generación de llave de sesión (RSA o ECDH).

- El proceso *1.2.4 (2.1.4) Preparar Certificado*

  En base al sistema de llave pública especificado en *ClientHello* se recupera el certificado del repositorio, se asigna la información a la estructura *SignatureInfo* y se extrae la llave pública propia del certificado.

- El proceso *1.2.1 (2.1.6) Calcular Llave de sesión*

  Para generar la llave de sesión, dependiendo del algoritmo elegido se realiza lo siguiente:

```
switch(key_exchange_suite)
      case ECDH:
            generar llave de sesión para curvas elípticas;
            break;
      case RSA:
            generar llave de sesión para curvas RSA;
            break;
      case NULLKES:
      default:
            //  no habrá generación de secreto
      }
```

4.4. Detalles de Implementación

A continuación se describen las condiciones de implementación:

- Nuestro prototipo del protocolo de Negociación está implementado en ANSI C siguiendo las indicaciones del estándar WTLS [9].
- Para las operaciones criptográficas se cuenta con el trabajo realizado en la Universidad de Oregon State, donde se desarrolló un conjunto de herramientas criptográficas, englobadas en una biblioteca conocida como RCT. Entre los algoritmos incluidos en la biblioteca se encuentran implementaciones de criptografía de llave pública con RSA y Curvas Elípticas, AES y DES para la criptografía de llave simétrica, y la familia SHA para las funciones *hash* [6].
- Para la comunicación se utilizaron sockets de Windows (Winsocks). El canal es simétrico, es decir, se asume que el tiempo de enlace es el mismo en las dos entidades.

En el apéndice A se detalla uno de los certificados utilizados en las pruebas.

Capítulo 5

Análisis y Evaluación del Desempeño

Las redes inalámbricas, como se ha mencionado anteriormente, toleran un ancho de banda restringido y una latencia relativamente alta por lo que el tiempo de procesamiento, el tamaño de los mensajes a transmitir son puntos clave para desarrollar una implementación eficiente del protocolo de negociación de WTLS.

WTLS recomienda utilizar, en la etapa de negociación, dos sistemas criptográficos de llave pública: RSA y CCE. CCE ofrece el mismo nivel de seguridad que RSA pero con tamaños de llave aproximadamente diez veces menores [25], lo que implica un proceso de negociación más "económico".

Existen diferentes artículos que han realizado comparaciones de desempeño entre el criptosistema RSA y el criptosistema de curvas elípticas dentro del protocolo de negociación de WTLS. En el artículo de Levi y Savas se presenta, por ejemplo, un modelo de desempeño analítico para las operaciones de llave pública dentro del protocolo WTLS donde se encontró que en general ECC es mejor opción que RSA para todos los niveles de seguridad en WTLS [23]. Por otra parte en el artículo presentado en [18, 19] se presentan resultados preeliminares a los mostrados en este trabajo, que incluye la evaluación del protocolo de negociación de WTLS clase 1.

En este capítulo se incluye un estudio comparativo de la clase 1 y 3 del protocolo de negociación de WTLS. En la primera sección se despliegan los resultados obtenidos para RSA y curvas elípticas y en la segunda sección se hace un análisis de ellos.

5.1. Pruebas Realizadas

La tabla 5.1 muestra los diferentes niveles de seguridad dados por tamaños de llaves comparables de RSA y CCE sugeridos por el estándar [9]. Esta equivalencia de seguridad se basa en la dificultad del problema matemático que origina a ambos criptosistemas. El mejor algoritmo conocido para resolver el problema de logaritmo discreto con curvas elípticas

(PLDCE) toma tiempo exponencial mientras que el mejor algoritmo conocido para resolver el problema matemático en el que se fundamenta RSA toma tiempo subexpononcial [25].

Nivel de Seguridad	CCE	RSA
1	160P,163K,163R	1024
2	224P,233K,233R	2048

Tabla 5.1: Nivel de Seguridad Criptográfica de CCE y RSA

Como puede observarse en la tabla 5.1, el nivel de seguridad ofrecido con una llave RSA de 1024 bits es comparable al nivel ofrecido por CCE con las curvas 160P,163K,163R; asimismo, las curvas 224P,233K,233R exhiben un nivel de seguridad comparable con una llave RSA de 2048 bits.

La plataforma de pruebas para el cliente y el servidor es una computadora Pentium II a 750Mhz. El canal es simétrico, es decir, se asume que el tiempo de enlace es el mismo en las dos entidades.

5.2. Resultados Obtenidos

A continuación se muestra en cada apartado los resultados obtenidos para la clase 1 la clase 3 del protocolo de negociación de WTLS. Para este estudio se han considerado todas las curvas especificadas por el estándar WTLS, esto es, las curvas elípticas siguientes [27]:

- las curvas de Koblitz: 163K y 233K
- las curvas sobre $GF(P)$: 160P y 224P
- las curvas pseudo-aleatorias sobre $GF(2^p)$: 163R, y 233R

cuya definición se encuentra en el Apéndice B.

5.2.1. WTLS Clase 1

- CCE En esta sección se muestran los resultados obtenidos para el sistema criptográfico de curvas elípticas utilizado en el protocolo de Negociación de WTLS. Se ha tomado en cuenta el tiempo de ejecución como el factor a evaluar. El protocolo de negociación ejecutado es completo sin Certificados, es decir, WTLS clase 1. Esto incluye a los protocolos RSA y ECDH para la generación de la llave de sesión.

Nivel de Seguridad	Curva	Tiempo de ejecución
1	160P	4.56 ms
1	163K	6.59 ms
1	163R	13.28 ms
2	224P	7.21 ms
2	233K	9.36 ms
2	233R	18.56 ms

Tabla 5.2: Tiempos de ejecución obtenidos para CCE

La tabla 5.2 muestra los tiempos de ejecución obtenidos para las diferentes curvas especificadas en el estándar WTLS obtenidos por el cliente. Para obtener los tiempos se ha tomado en cuenta la duración completa del protocolo de Negociación, ejecutado centenas de veces para después obtener la media aritmética de los tiempos alcanzados.

- RSA

  En esta sección se muestran los tiempos obtenidos como resultado de la ejecución del protocolo WTLS con el sistema criptográfico RSA. Los tamaños de llave evaluados son los recomendados por el estándar: 1024 bits y 2048 bits. Se recalca que el protocolo de negociación evaluado corresponde al WTLS clase 1, esto es, completo sin certificados.

  La tabla 5.3 muestra los tiempos obtenidos para las diferentes llaves, con su nivel de seguridad correspondiente.

Nivel de Seguridad	Tamaño Llave	Tiempo de ejecución
1	1024	18.48 ms
2	2048	82.45 ms

Tabla 5.3: Tiempos de ejecución obtenidos para RSA

5.2.2. WTLS Clase 3

Aunque el prototipo acepta diferentes combinaciones entre los protocolos de generación de llaves y de firma, esto es, se pueden ejecutar una firma RSA con una llave contenida para ECDH, se contemplan, para este estudio los casos RSA tanto para firma como para generación de llave (caso RSA) y el protocolo ECDSA para firma en combinación con ECDH para generación de llave (caso CCE).

- CCE

Nivel de Seguridad	Curva	Tiempo de ejecución
1	160P	11.26 ms
1	163K	12.44 ms
1	163R	19.30 ms
2	224P	13.22 ms
2	233K	16.21 ms
2	233R	23.45 ms

Tabla 5.4: Tiempos de ejecución obtenidos para CCE

Los resultados obtenidos para el sistema criptográfico de curvas elípticas ECDSA-ECDH utilizado en el protocolo de Negociación de WTLS se muestran en está sección. Para este estudio, como se mencionó al inicio, se han considerado todas las curvas especificadas por el estándar WTLS, esto es, las curvas conocidas como 160P, 163K, 163R, 224P, 233K, y 233R [9]. Se evalúa el tiempo de ejecución del protocolo de negociación ejecutado con autenticación por ambas partes, es decir, WTLS clase 3.

La tabla 5.4 muestra los tiempos de ejecución obtenidos. Estos tiempos se obtuvieron tomando en cuenta la duración completa del protocolo de Negociación desde el punto de vista del cliente, ejecutado centenas de veces para después obtener la media aritmética de los tiempos alcanzados.

- RSA

A continuación se muestran los tiempos obtenidos como resultado de la ejecución del protocolo WTLS con el sistema criptográfico RSA en ambos protocolos, el de firma y de generación de llave de sesión. Los tamaños de llave evaluados son 1024 bits y 2048 bits. El protocolo de negociación implementado corresponde al WTLS clase 3, esto es, con intercambio de certificados del cliente y del servidor.

La tabla 5.5 muestra los tiempos obtenidos para las diferentes llaves, con su nivel de seguridad correspondiente.

Nivel de Seguridad	Tamaño Llave	Tiempo de ejecución
1	1024	32.08 ms
2	2048	86.35 ms

Tabla 5.5: Tiempos de ejecución obtenidos para RSA

Mensaje	Protocolo	Tamaño
Hola del Cliente	CCE 160P	23 bytes
	CCE 163K	23 bytes
	CCE 163R	23 bytes
	RSA 1024	23 bytes
	RSA 2048	23 bytes
Hola del Servidor	CCE 160P	18 bytes
	CCE 163K	18 bytes
	CCE 163R	18 bytes
	RSA 1024	18 bytes
	RSA 2048	18 bytes
Llave Pública	CCE 160P	20 bytes
	CCE 163K	21 bytes
	CCE 163R	21 bytes
	RSA 1024	128 bytes
	RSA 2048	256 bytes
Certificado	CCE 160P	509 bytes
	CCE 163K	510 bytes
	CCE 163R	510 bytes
	RSA 1024	617 bytes
	RSA 2048	745 bytes

Tabla 5.6: Tamaños de mensajes intercambiados

5.3. Análisis de Resultados

Uno de los aspectos que se deben de tomar en cuenta cuando se trabaja en un medio con restricciones tales como una latencia alta y ancho de banda relativamente bajo es la cantidad y tamaño de los mensajes intercambiados. Como se puede observar en la tabla 5.6 los mensajes *Hola del Cliente* y *Hola del Servidor* tienen el mismo tamaño debido a la optimización de estos mensajes para su tránsito. Sin embargo, en los mensajes donde se intercambian la llave y el certificado la diferencia de tamaño se hace visible, tal como se muestra en las figuras 5.1 y 5.2.

- WTLS Clase 1 En las figuras 5.3 y 5.4 tenemos los gráficos donde se muestran los tiempos de ejecución para ambos sistemas criptográficos, CCE y RSA, en los diferentes niveles de seguridad. En ambos casos el tiempo medido es el de la duración total del protocolo WTLS en el servidor. Dado que tanto el cliente como el servidor ejecutan en la misma plataforma, los tiempos de ejecución del cliente no varían significantemente con los obtenidos en el servidor. Debido a ello sólo se tomó en cuenta en este estudio el tiempo de ejecución desde que el cliente inicia la comunicación hasta el momento que se ha generado la llave de sesión.

  Como se puede apreciar, el mejor tiempo obtenido, para el nivel de seguridad 1, fue

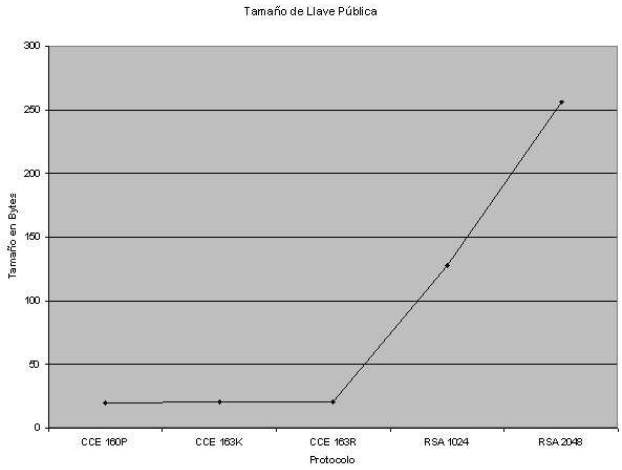


Figura 5.1: Tamaño de la Llave intercambiada

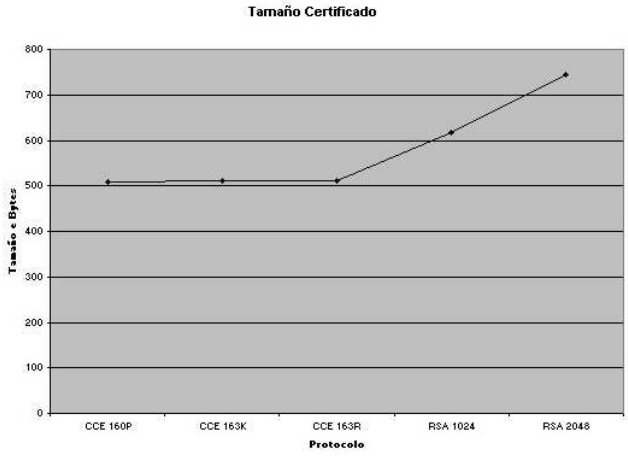


Figura 5.2: Tamaño del certificado intercambiado

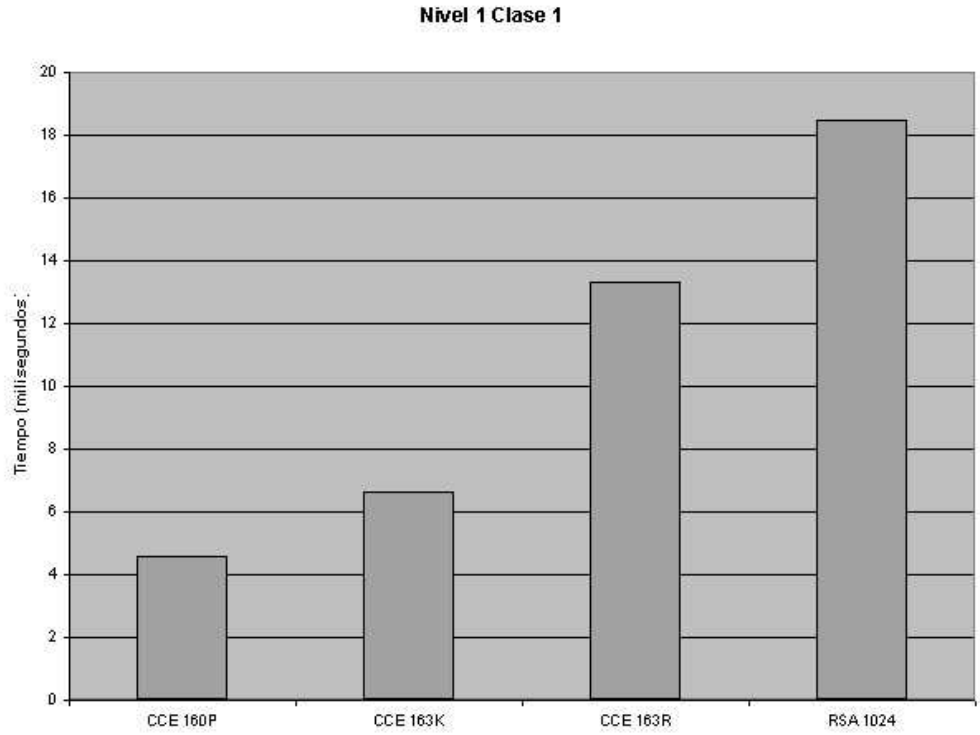


Figura 5.3: Gráfico comparativo de los tiempos de ejecución en el nivel 1 de seguridad

alcanzado con la curva 160P de CCE, mientras que el peor caso se obtiene con RSA. Esta misma situación se repite para el nivel de seguridad 2, pero, como se observa en la figura 5.4 la diferencia en tiempos es aún más pronunciada.

- WTLS Clase 3

  En las figuras 5.5 y 5.6 tenemos los gráficos donde se muestran los tiempos de ejecución para ambos sistemas criptográficos, CCE y RSA, en los diferentes niveles de seguridad. En ambos casos el tiempo medido es el de la duración total del protocolo WTLS en el servidor. Dado que tanto el cliente como el servidor ejecutan en la misma plataforma, los tiempos de ejecución del cliente no varían significantemente con los obtenidos en el servidor. Debido a ello sólo se tomó en cuenta en este estudio el tiempo de ejecución en el cliente.

  Como se puede apreciar, el mejor tiempo obtenido, para el nivel de seguridad 1, fue alcanzado con la curva 160P de CCE, mientras que el peor caso se obtiene con RSA. Esta misma situación se repite para el nivel de seguridad 2.

Para el protocolo WTLS clase 3 las diferencias de tiempo de ejecución entre RSA y ECC son muy marcadas debido a que la generación de llaves de RSA es un proceso tardado.

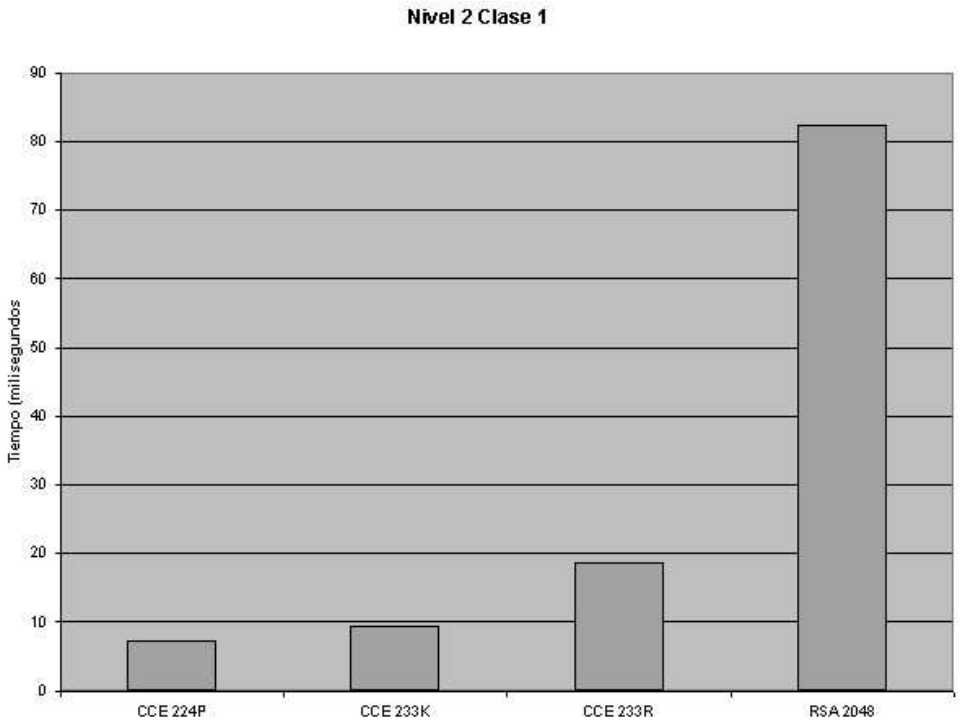


Figura 5.4: Gráfico comparativo de los tiempos de ejecución en el nivel 2 de seguridad

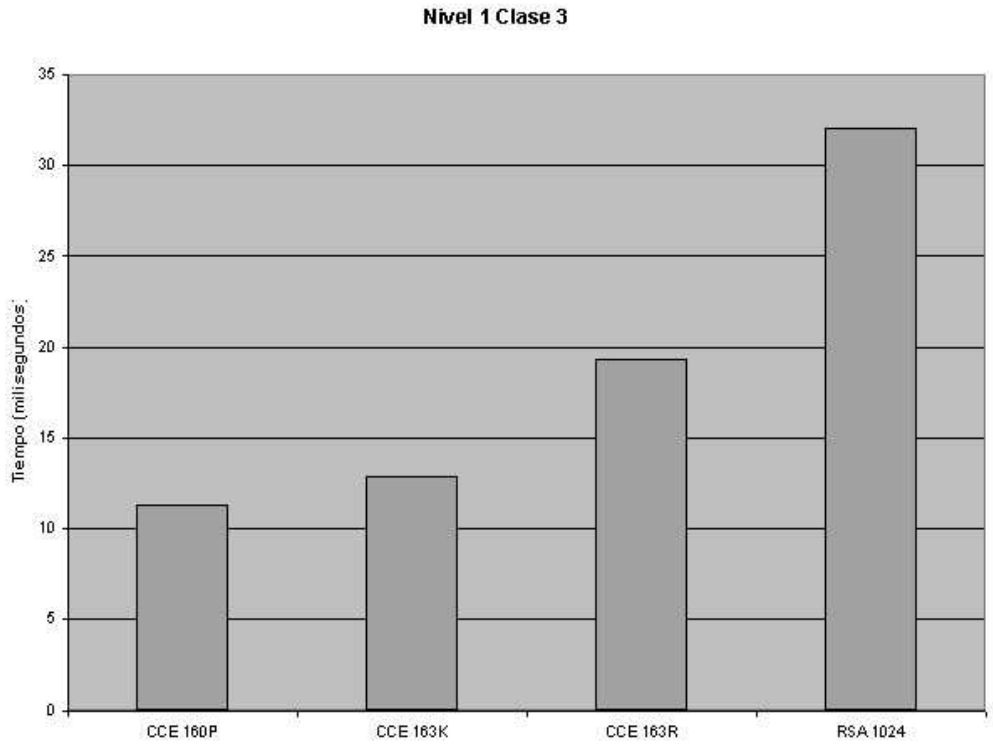


Figura 5.5: Gráfico comparativo de los tiempos de ejecución en el nivel 1 de seguridad

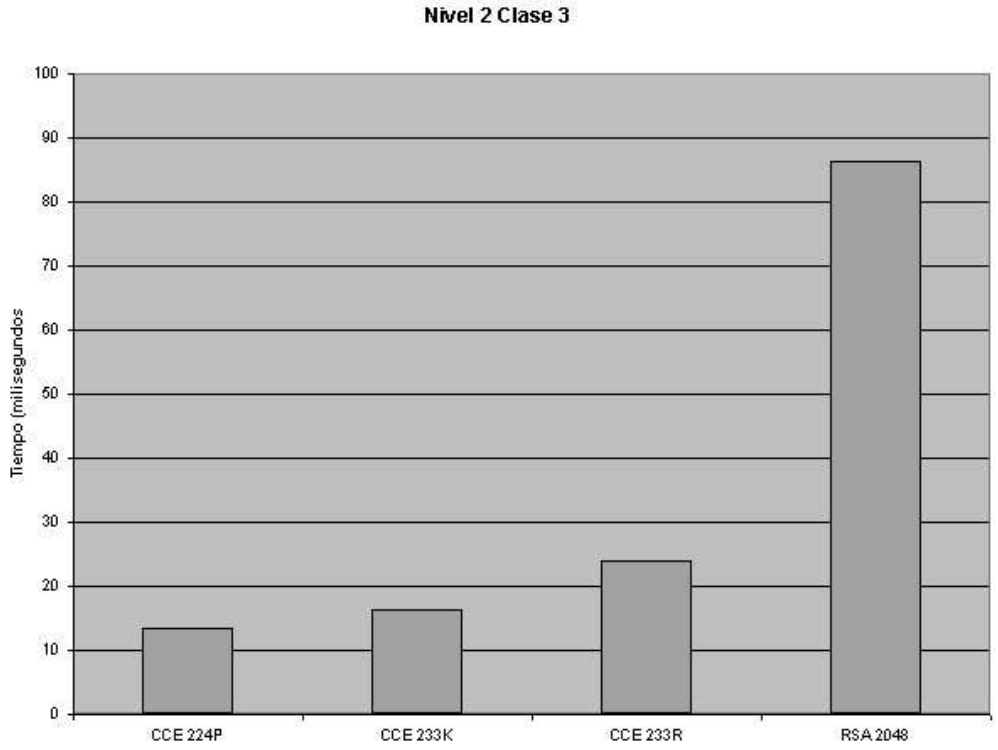


Figura 5.6: Gráfico comparativo de los tiempos de ejecución en el nivel 2 de seguridad

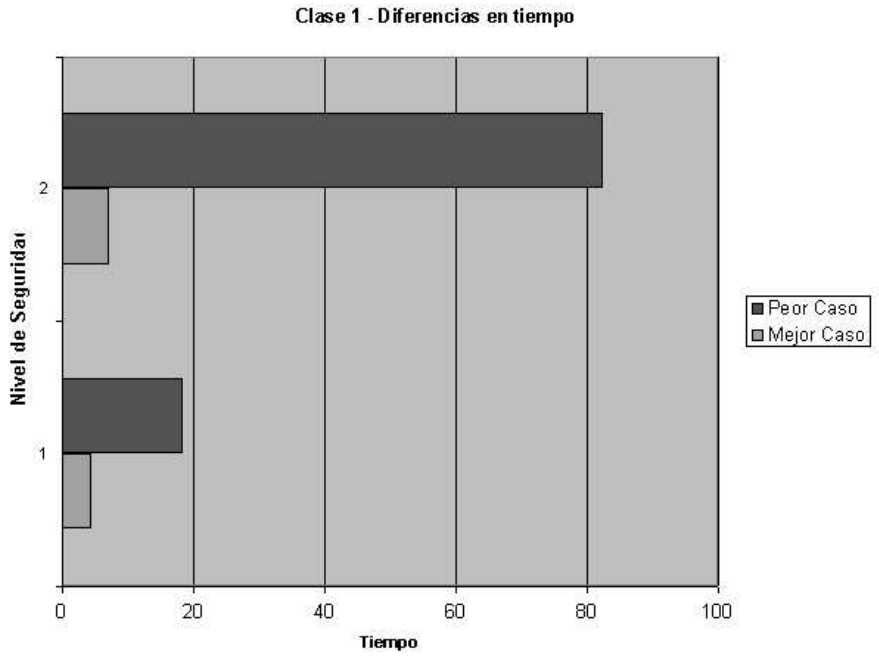


Figura 5.7: Gráfico comparativo WTLS Clase 1

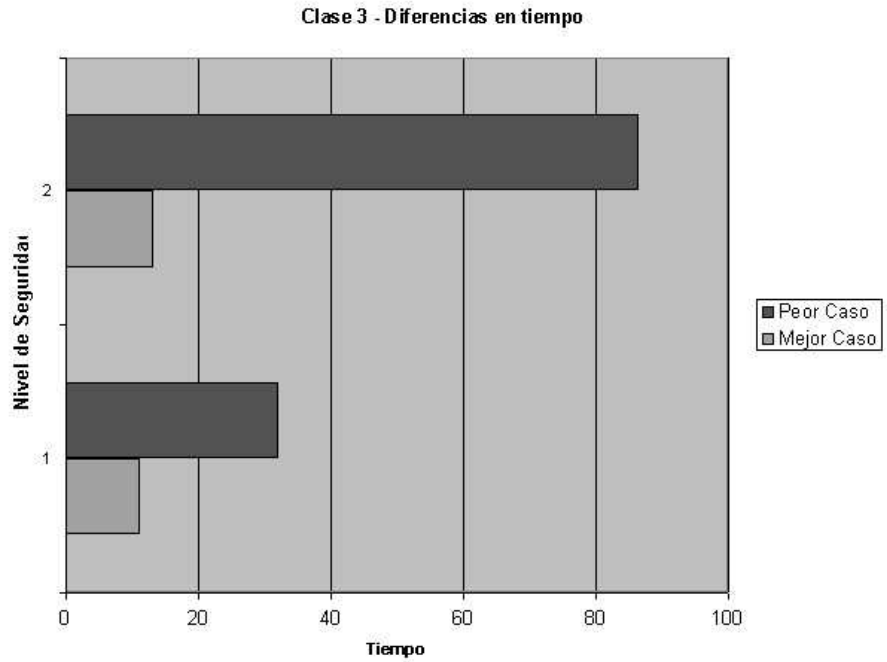


Figura 5.8: Gráfico comparativo WTLS Clase 3

En el trabajo analítico realizado en [23] encontraron, para el protocolo de Negociación de WTLS clase 3, diferencias de tiempos entre ECC y RSA hasta de 4 veces mayores, comparando ECC 160P y RSA 1024 para el nivel 1. Para el nivel 2 encontraron una diferencia de un poco más de 11 veces al comparar ECC 224P contra RSA de 2048. En este trabajo, al implementar el protocolo y ejecutarlo se encuentra, para el protocolo de Negociación de WTLS clase 3, diferencias de tiempos entre ECC y RSA hasta de 2.4 veces mayores, comparando ECC 160P y RSA 1024 para el nivel 1. Para el nivel 2 encontraron una diferencia de aproximadamente 5 veces al comparar ECC 224P contra RSA de 2048.

Capítulo 6

Conclusiones

La movilidad ha originado nuevos desafíos a la seguridad de los sistemas de información tradicional. El intercambio de datos confidenciales es común en las redes inalámbricas donde el canal de comunicación es inherentemente inseguro y puede ser atacado de manera pasiva comprometiendo la confidencialidad de los datos. Otro de los puntos débiles de la seguridad en las redes inalámbricas es el acceso no autorizado a sus recursos donde un intruso puede violar la confidencialidad y la integridad del tráfico de la red al enviar, recibir, alterar o falsificar mensajes. Este último ataque, que se considera activo, puede llevarse al cabo utilizando un dispositivo inalámbrico comprometido con acceso libre a la red. Se ha destacado que la mejor protección contra el acceso no autorizado es la implementación de mecanismos de autenticación que aseguren el acceso de usuarios identificados a la red.

En las redes WLAN y celulares, los dispositivos móviles como asistentes digitales y celulares tienen un poder de cómputo y memoria limitados. Aunado a ello, en las redes celulares el ancho de banda restringido es otra limitación importante, cada mensaje enviado incrementa el costo de la comunicación. Es por ello que para implementar los mecanismos de autenticación se debe minimizar el intercambio de mensajes cuando se establece una sesión segura. De igual manera, las implementaciones de seguridad no deben hacer cálculos extensivos del lado del cliente. Por lo tanto no sólo basta con la puesta en marcha de servicios de seguridad en estas redes, sino que se requiere reducir la complejidad computacional de las implementaciones de tales servicios.

Este trabajo aporta un estudio del estado del arte de la autenticación desde diferentes perspectivas: la *teoría* que abarca las bases de la autenticación, es decir, una guía de cómo la criptografía de llave pública se utiliza en la autenticación; un *caso de estudio teórico* para las redes 802.11, y un *caso de estudio práctico* para las redes celulares en el caso específico del protocolo WTLS de WAP, aportando, para este caso, un prototipo que sirve de plataforma de pruebas para la comparación de diferentes protocolos criptográficos y un estudio comparativo de los criptosistemas sugeridos por el estándar.

Con lo que respecta a las redes 802.11 se encontró que aún quedan varios aspectos por definir con respecto al tema de autenticación. Pero el aspecto que más preocupa es que, en

el modo de infraestructura, sólo se realiza autenticación a las terminales, pero en contraste, las terminales no tienen ningún mecanismo para que autentiquen la red a la que se están comunicando, es decir, al PA. Esto implica que un nodo impostor puede hacerse pasar por un PA y establecer comunicación con la terminal teniendo de esta manera acceso a todo lo que la terminal le envía.

En el caso del protocolo WTLS se cuenta con un estándar que específica como evitar el problema anterior mediante la implementación del protocolo de Negociación clase 3 de WTLS. Sin embargo, es en está etapa donde puede hacerse palpable al usuario de un dispositivo móvil una disminución en la velocidad de las transacciones de comercio electrónico si se hace una mala elección del criptosistema de llave pública utilizado.

La mayoría de las implementaciones de WTLS han optado por RSA debido, entre otras razones, a su mayor difusión en el mercado informático y de redes alámbricas. En diversos estudios teóricos se ha argüido que el esquema de curvas elípticas ofrece niveles de seguridad comparables a los de RSA con tamaños de llave aproximadamente diez veces menores. La complejidad de ejecución en los algoritmos de llave pública está directamente relacionada con el tamaño de las llaves, por lo que CCE es en principio una mejor opción que RSA.

En este trabajo se ha realizado una implementación práctica del protocolo WTLS, que en esta etapa inicial, fue ejecutada en la misma plataforma. La evidencia experimental encontrada hasta el momento muestra resultados favorables para el protocolo WTLS cuando utiliza el sistema de criptografía de curvas elípticas. Es así que los resultados experimentales presentados en este trabajo permiten confirmar que efectivamente CCE es la opción criptográfica más eficiente para la implementación de WTLS, con tiempos de ejecución 2 y 5 veces más rápidos que los correspondientes para RSA de 1024 y 2048 bits en la clase de implementación 3, respectivamente. Esta diferencia de complejidad de ejecución entre criptosistemas de curvas elípticas y RSA tiene repercusiones en el desempeño del protocolo de seguridad lo que a su vez implica un detrimento en la calidad final de los servicios ofrecidos al usuario.

En todo caso, brindar servicios de seguridad en especial de autenticación de mala calidad, ya sea por un mal diseño o al hacer elecciones desfavorables en su implementación puede traer consecuencias perjudiciales a los usuarios de las redes inalámbricas.

Este trabajo deja entrever diferentes aspectos que pueden explotarse como trabajo a futuro, entre ellos se pueden mencionar:

- ¿Cúales serán los resultados en el desempeño al mudar la parte de cliente a una plataforma móvil?
- ¿Se obtendrán los mismos resultados al implementar un esquema de autenticación similar al de WTLS en las redes 802.11 o son necesarias otras consideraciones?
- ¿Cuál es la relación de los servicios de seguridad con la calidad de servicio?

Apéndice A

Este es un ejemplo de un certificado versión 3 de 510 bytes. El certificado contiene la siguiente información:

1. La versión del certificado es 3.
2. El número de serie del certificado es 1 (01 hex).
3. El certificado esta firmado con ECDSA y con el hash SHA-1
4. El nombre distinguido del emisor del certificado es OU=cinvestav; O=edu; C=MX.
5. El nombre distinguido del usuario es OU=cinvestav; O=org; C=MX; y CN=Servidor WAP.
6. El certificado fue expedido el 19 de Septiembre de 2003 y expirará el 19 de Septiembre de 2004
7. El certificado contiene una llave pública de CE de 163 bits con parámetros de curva default
8. La huella digital

```
0000 30 82 01 fa  506: SEQUENCE
0004 30 82 01 ba  442: . SEQUENCE    tbscertificate
0008 a0 03          3: . . [0]
0010 02 01          1: . . . INTEGER 2
                     : 02
0013 02 01          1: . . INTEGER 1
                     : 01
0016 30 09          9: . . SEQUENCE
0018 06 07          7: . . . OID 1.2.840.10045.4.1: ecdsa-with-SHA1
                     : 2a 86 48 ce 3d 04 03
0027 30 2f         47: . . SEQUENCE
0029 31 0b         11: . . . SET
0031 30 09          9: . . . . SEQUENCE
```

```
0033 06 03          3: . . . . . OID 2.5.4.6: C
                     : 55 04 06
0038 13 02          2: . . . . . PrintableString  'MX'
                     : 4d 58
0042 31 0c         12: . . . SET
0044 30 0a         10: . . . . SEQUENCE
0046 06 03          3: . . . . . OID 2.5.4.10: O
                     : 55 04 0a


0051 13 03          3: . . . . . PrintableString  'edu'
                     : 65 64 75
0056 31 12         18: . . . SET
0058 30 10         16: . . . . SEQUENCE
0060 06 03          3: . . . . . OID 2.5.4.11: OU
                     : 55 04 0b
0065 13 09          9: . . . . . PrintableString  'cinvestav'
                     : 63 69 6e 76 65 73 74 61 76
0076 30 1e         30: . . SEQUENCE
0078 17 0d         13: . . . UTCTime  '030919000000Z'
                     : 30 33 30 39 31 39 30 30 30 30 30 30 5a
0093 17 0d         13: . . . UTCTime  '040919000000Z'
                     : 30 34 30 39 31 39 30 30 30 30 30 30 5a
0108 30 46         70: . . SEQUENCE
0110 31 0b         11: . . . SET
0112 30 09          9: . . . . SEQUENCE
0114 06 03          3: . . . . . OID 2.5.4.6: C
                     : 55 04 06
0119 13 02          2: . . . . . PrintableString  'MX'
                     : 4d 58
0123 31 0c         12: . . . SET
0125 30 0a         10: . . . . SEQUENCE
0127 06 03          3: . . . . . OID 2.5.4.10: O
                     : 55 04 0a
0132 13 03          3: . . . . . PrintableString  'edu'
                     : 65 64 75
0137 31 12         18: . . . SET
0139 30 10         16: . . . . SEQUENCE
0141 06 03          3: . . . . . OID 2.5.4.11: OU
                     : 55 04 0b
0146 13 09          9: . . . . . PrintableString  'cinvestav'
                     : 63 69 6e 76 65 73 74 61 76
0157 31 15         21: . . . SET
```

```
0159 30 13         19: . . . . SEQUENCE
0161 06 03          3: . . . . . OID 2.5.4.3: CN
                     : 55 04 03
0166 13 0c         12: . . . . . PrintableString 'Servidor WAP'
                     : 53 65 72 76 69 64 6F 72  20 57 41 50
0180 30 82 d9     217: . . SEQUENCE
0183 30 82 0b      11: . . . SEQUENCE
0186 06 07          7: . . . . OID 1.2.840.10045.4.1: id-ecPublicKey
                     : 2a 86 48 ce 3d 04 01
0195 05 00          0: . . . NULL
0197 03 c9        201: . . . BIT STRING
   : 04 06 3b 30 92 01 02 56 27 43 e3 bd ce 2d 26 a0
   : e6 97 fa 02 da 87 03 c4 5a 32 a1 9b c9 65 1e 23
   : b4 54 ad e6 1a 12 09 e3 ab b6 49 cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc cc
   : cc cc cc cc cc cc cc cc cc
0400 a3 81 2f      47: . . [3]
0403 30 2d         45: . . . SEQUENCE
0405 30 1b         27: . . . . SEQUENCE
0407 06 03          3: . . . . . OID 2.5.29.35: authorityKeyIdentifier
                     : 55 1d 23
0412 04 14         20: . . . . . OCTET STRING
                     : 30 12 80 10 0e 6b 3a bf 04 ea 04 c3 0e 6b 3a bf
                     : 04 ea 04 c3
0434 30 0e         14: . . . . SEQUENCE
0436 06 03          3: . . . . . OID 2.5.29.15: keyUsage
                     : 55 1d 0f
0441 01 01          1: . . . . . TRUE
                        01
0444 04 04          4: . . . . . OCTET STRING
                     : 03 02 07 80
0450 30 09          9: . SEQUENCE
0452 06 07          7: . . OID 1.2.840.10045.4.1: ecdsa-with-SHA1
                     : 2a 86 48 ce 3d 04 01
0461 03 2f         47: . BIT STRING  (0 unused bits)
```

```
: 30 2c 02 14 a0 66 c1 76 33 99 13 51 8d 93 64 2f
: ca 13 73 de 79 1a 7d 33 02 14 5d 90 f6 ce 92 4a
: bf 29 11 24 80 28 a6 5a 8e 73 b6 76 02 68 23
```

A continuación se presenta el dump hexadecimal, que es el certificado en sí:

```
30 82 01 fa     30 82 01 ba     a0 03 02 01     02 02 01 01
30 09 06 07     2a 86 48 ce     3d 04 03 30     2f 31 0b 30
09 06 03 55     04 06 13 02     4d 58 31 0c     30 0a 06 03
55 04 0a 13     03 65 64 75     31 12 30 10     06 03 55 04
0b 13 09 63     69 6e 76 65     73 74 61 76     30 1e 17 0d
30 33 30 39     31 39 30 30     30 30 30 30     5a 17 0d 30
34 30 39 31     39 30 30 30     30 30 30 5a     30 46 31 0b
30 09 06 03     55 04 06 13     02 4d 58 31     0c 30 0a 06
03 55 04 0a     13 03 65 64     75 31 12 30     10 06 03 55
04 0b 13 09     63 69 6e 76     65 73 74 61     76 31 15 30
13 06 03 55     04 03 13 0C     53 65 72 76     69 64 6F 72
20 57 41 50     30 82 d9 30     82 0b 06 07     2a 86 48 ce
3d 04 01 05     00 03 c9 04     06 3b 30 92     01 02 56 27
43 e3 bd ce     2d 26 a0 e6     97 fa 02 da     87 03 c4 5a
32 a1 9b c9     65 1e 23 b4     54 ad e6 1a     12 09 e3 ab
b6 49 cc cc     cc cc cc cc      cc cc cc cc      cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
cc cc cc cc     cc cc cc cc     cc cc cc cc     cc cc cc cc
a3 81 2f 30     2d 30 1b 06     03 55 1d 23     04 14 30 12
80 10 0e 6b     3a bf 04 ea     04 c3 0e 6b     3a bf 04 ea
04 c3 30 0e     06 03 55 1d     0f 01 01 01     04 04 03 02
07 80 30 09     06 07 2a 86     48 ce 3d 04     01 03 2f 30
2c 02 14 a0     66 c1 76 33     99 13 51 8d     93 64 2f ca
13 73 de 79     1a 7d 33 02     14 5d 90 f6     ce 92 4a bf
29 11 24 80     28 a6 5a 8e     73 b6 76 02     68 23
```

Apéndice B

Curvas Elípticas Utilizadas

Número Asignado	3
Básica	No
Tamaño del Campo	163
Polinomio Irreducible	$x^{163} + x^7 + x^6 + x^3 + 1$
Curva Elíptica E	$y^2 + xy = x^3 + ax^2 + b$; sobre $GF(2^{163})$
Parámetro a	01
Parámetro b	01
Punto generador G	02 FE13C053 7BBC11AC AA07D793 DE4E6D5E 5C94EEE8, 02 89070FB0 5D38FF58 321F2E80 0536D538 CC DAA3D9
Orden de G	04 00000000 00000000 00020108 A2E0CC0D 99F8A5EF
Cofactor K	02

Tabla 6.1: Curva $163K$

Número Asignado	5
Básica	Si
Tamaño del Campo	163
Polinomio Irreducible	$x^{163} + x^8 + x^2 + x + 1$
Curva Elíptica E	$y^2 + xy = x^3 + ax^2 + b$; sobre $GF(2^{163})$
Semilla	D2C0FB15 760860DE F1EEF4D6 96E67687 56151754
Parámetro a	07 2546B543 5234A422 E0789675 F432C894 35DE5242
Parámetro b	00 C9517D06 D5240D3C FF38C74B 20B6CD4D 6F9DD4D9
Punto generador G	07 AF699895 46103D79 329FCC3D 74880F33 BBE803CB, 01 EC23211B 5966ADEA 1D3F87F7 EA5848AE F0B7CA9F ($\widetilde{y}p = 01$)
Orden de G	04 00000000 00000000 0001E60F C8821CC7 4DAEAFC1
Cofactor K	02

Tabla 6.2: Curva $163R$

Número Asignado	10
Básica	No
Tamaño del Campo	233
Polinomio Irreducible	$x^{233} + x^{74} + 1$
Curva Elíptica E	$y^2 + xy = x^3 + ax^2 + b$; sobre $GF(2^{233})$
Parámetro a	0000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
Parámetro b	0000 00000000 00000000 00000000 00000000 00000000 00000000 00000001
Punto generador G	0172 32BA853A 7E731AF1 29F22FF4 149563A4 19C26BF5 0A4C9D6E EFAD6126, 01DB 537DECE8 19B7F70F 555A67C4 27A8CD9B F18AEB9B 56E0C110 56FAE6A3 ($\widetilde{y}p = 00$)
Orden de G	80 00000000 00000000 00000000 00069D5B B915BCD4 6EFB1AD5 F173ABDF
Cofactor K	04

Tabla 6.3: Curva $233K$

Número Asignado	11
Básica	No
Tamaño del Campo	233
Polinomio Irreducible	$x^{233} + x^{74} + 1$
Curva Elíptica E	$y^2 + xy = x^3 + ax^2 + b$; sobre $GF(2^{233})$
Semilla	74D59FF0 7F6B413D 0EA14B34 4B20A2DB 049B50C3
Parámetro a	0000 00000000 00000000 00000000 00000000 00000000 00000000 00000001
Parámetro b	0066 647EDE6C 332C7F8C 0923BB58 213B333B 20E9CE42 81FE115F 7D8F90AD
Punto generador G	00FA C9DFCBAC 8313BB21 39F1BB75 5FEF65BC 391F8B36 F8F8EB73 71FD558B, 0100 6A08A419 03350678 E58528BE BF8A0BEF F867A7CA 36716F7E 01F81052 ($\widetilde{y}p = 01$)
Orden de G	0100 00000000 00000000 00000000 0013E974 E72F8A69 22031D26 03CFE0D7
Cofactor K	02

Tabla 6.4: Curva $233R$

Número Asignado	12
Básica	No
Tamaño del Campo	224
Curva Elíptica E	$y^2 = x^3 + ax + b$; sobre $GF(p)$
Primo p	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 00000000 00000001
Semilla	BD713447 99D5C7FC DC45B59F A3B9AB8F 6A948BC5
Parámetro a	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFE
Parámetro b	B4050A85 0C04B3AB F5413256 5044B0B7 D7BFD8BA 270B3943 2355FFB4
Punto generador G	B70E0CBD 6BB4BF7F 321390B9 4A03C1D3 56C21122 343280D6 115C1D21, BD376388 B5F723FB 4C22DFE6 CD4375A0 5A074764 44D58199 85007E34 ($\widetilde{y}p = 00$)
Orden de G	FFFFFFFF FFFFFFFF FFFFFFFF FFFF16A2 E0B8F03E 13DD2945 5C5C2A3D
Cofactor K	01

Tabla 6.5: Curva $224P$

Número Asignado	7
Básica	Sí
Tamaño del Campo	160
Curva Elíptica E	$y^2 = x^3 + ax + b$; sobre $GF(p)$
Primo p	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 7FFFFFFF
Semilla	S = 1053CDE4 2C14D696 E6768756 1517533B F3F83345;
	r = 2DA6C4D7 0B90FF91 2E725E25 E90AF631 C18F0D2F
Parámetro a	FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 7FFFFFFC
Parámetro b	1C97BEFC 54BD7A8B 65ACF89F 81D4D4AD C565FA45
Punto generador G	4A96B568 8EF57328 46646989 68C38BB9 13CBFC82, 23A62855 3168947D 59DCC912 04235137 7AC5FB32 ($\widetilde{y}p = 00$)
Orden de G	01 00000000 00000000 0001F4C8 F927AED3 CA752257
Cofactor K	01

Tabla 6.6: Curva $160P$

Bibliografía

[1] Macphee A. *Understanding Digital Certificates and Wireless Transport Layer Security (WTLS).* Entrust, 2001. Disponible en `http://www.entrust.com/resourcecenter/whitepapers.htm`.

[2] Menezes A., van Oorschot P., and Vanstone S. *Handbook of Applied Cryptography.* CRC Press, New York, quinta edición edition, 2001.

[3] Zarba C. *Model Checking the Needham-Schroeder Protocol.* 1998. Disponible en `http://rodin.Stanford.EDU/case-studies/security`.

[4] Gollmann D. *Computer Security.* John Wiley & Sons, Chichester,New York, 1999.

[5] Martínez E. Comienza el boom del internet inalámbrico. *Revista RED*, 2001. Volumen de Enero 2001.

[6] Savas E., Rodríguez F., Koc C., and et al. *RCT, RSA and ECC Toolkit.* ISL Group, Oregon State University, 2002.

[7] WAP Forum. *WAP white paper.*

[8] WAP Forum. $WAP-211-WAPCert$, *WAP Certificate and CRL Profiles.* 2001. versión 22-May-2001.

[9] WAP Forum. $WAP-261-WTLS-20010406-a$, *WTLS Specification.* 2001. Versión 06-Abr-2001, disponible en `http://www.wapforum.com`.

[10] F. García. *Certificados X.509.* Disponible en: `http://www.alu.ua.es/f/`.

[11] GSMWORLD. *WAP: Il futuro nel telefonino.* 2002. Disponible en `http://www.gsmworld.com/wap`.

[12] Krawczyky H. Sigma: the 'sign-and-mac' approach to authenticated diffie-hellman and its use in the ike protocols. *Advances in Cryptology - Crypto 2003, Lecture Notes in Computer Science , Volume 2729 / 2003*, pages 400 – 425, 2003.

[13] Herwono I. and Liebhardt I. Performance evaluation of the wap security protocols. In *Proceedings of the 10th Aachen Symposium on Signal Theory*, pages 95 – 100, Achen, Germany, 2001.

[14] Herwono I. and Liebhardt I. Performance of wtls and its impact on an m-commerce transaction. In *Proceedings of the Third International Conference on Information and Communications Security, ICICS 2001*, pages 167 – 171, Xi'an, China, 2001. Disponible en `http://www.comnets.rwth-aachen.de`.

[15] Daemen J. and Rijmen V. *The Design of Rijndael*. Springer-Verlag, Germany, 2002.

[16] Bassham L., Housley R., and Polk W. *Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and CRL Profile*. PKIX Working Group, 2001.

[17] Lucena L. *Criptografía y Seguridad en Computadores*. 2001. Tercera Edición, Versión 1.00.

[18] Reyes L. and Rodríguez F. Desempeño del protocolo de negociación de wtls para curvas elípticas. In *Congreso de Ingeniería Eléctrica CINVESTAV-IPN*, Ciudad de México, 2003.

[19] Reyes L. and Rodríguez F. Estudio comparativo de los sistemas criptográficos de clave pública de wtls. In *Segundo Congreso Iberoamericano de Seguridad Informática*, Ciudad de México, 2003.

[20] Reyes L. and Rodríguez F. Quality of security service for wireless lans. In *Congreso de Ingeniería Eléctrica CINVESTAV-IPN*, Ciudad de México, 2003.

[21] RSA Laboratories. *Página Oficial*. Disponible en `http://www.rsasecurity.com/rsalabs/`.

[22] RSA Laboratories. *PKCS #1 v2.1: RSA Cryptography Standard*. 2002. Disponible en `http://www.rsasecurity.com/rsalabs/`.

[23] Savas E. Levi A. Performance evaluation of public-key cryptosystem operations in wtls protocol. In *The 8th IEEE Symposium on Computers and Communications ISCC 2003*, Kemer-Antalya, Turkey, 2003.

[24] Blum M. and Goldwasser S. An efficient probabilistic public key encryption scheme which hides all partial information. *Advances in Cryptology - Crypto'84, Lecture Notes in Computer Science vol.196*, pages 289–299, 1985.

[25] Brown M. and et al. Cheung D. *PGP in Constrained Wireless Devices*.

[26] Gast M. *Wireless LAN Security: A Short History*. 2002. Disponible en: `http://www.oreillynet.com/`.

[27] U.S. DEPARTMENT OF COMMERCE / NIST. Digital signature standard (dss). *FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION FIPS PUB 186-2*, 2000.

[28] Housley R., Ford W., Polk W., and Solo D. *Internet X.509 Public Key Infrastructure, Certificate and CRL Profile*. Network Working Group.

[29] Nichols R. and Lekkas P. *Wireless Security: Models, Threats, and Solutions.* McGraw-Hill Telecom, 2002.

[30] Pressman R. *Ingniería del Software, un Enfoque Práctico*. McGraw-Hill, 1993. Tercera Edición.

[31] Zuccherato R. and Adams C. *Using Elliptic Curve Diffie-Hellman in the SPKM GSS-API*. 1999.

[32] K. Spen. *WAP Security*. 1999. Wireless Application Forum. Disponible en `http://www.wapforum.com`.

[33] STARWIM. *Enabling Secure Mobile Internet*. Giesecke & Devrient GmbH, 2001. Reporte Técnico. Disponible en `http://www.gieseckedevrient.com`.

[34] Diffie W. and Hellman M. New directions in cryptography. *IEEE Trans. Information Theory*, pages 644–654, 1976. IT-22(6).November 1976.

[35] Trappe W. and Washington L. *Introduction to Cryptography with Coding Theory*. Prentice-Hall, Upper Saddle River, New Jersey, 2002.

[36] Certicom Whitepaper. *The Elliptic Curve Digital Signature Algorithm (ECDSA)*. 2003. Disponible en: `http:/www.certicom.com/resources/w_papers/w_papers.html`.

Índice alfabético