UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T

T.36

(07/97)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE T: TERMINALES PARA SERVICIOS DE TELEMÁTICA

Capacidades de seguridad para su utilización con terminales facsímil del grupo 3

Recomendación UIT-T T.36

(Anteriormente Recomendación del CCITT)

RECOMENDACIONES DE LA SERIE T DEL UIT-T

TERMINALES PARA SERVICIOS DE TELEMÁTICA

Para más información, véase la Lista de Recomendaciones del UIT-T.

RECOMENDACIÓN UIT-T T.36

CAPACIDADES DE SEGURIDAD PARA SU UTILIZACIÓN CON TERMINALES FACSÍMIL DEL GRUPO 3

Resumen

Esta Recomendación define las dos soluciones técnicas independientes que pueden utilizarse en el contexto de una transmisión facsímil segura. Las dos soluciones técnicas se basan en los algoritmos HKM/HFX40 y en el algoritmo RSA.

El anexo A contiene información relativa a los algoritmos HKM/HFX40.

El anexo B contiene información relativa al algoritmo RSA.

El anexo C describe la utilización del sistema HKM para proporcionar capacidades de gestión de claves de seguridad destinadas a terminales facsímil. La obtención de las capacidades se describe en términos de dos procedimientos principales:

– el procedimiento para registro unidireccional entre entidades X e Y (procREGxy); y

– el procedimiento para la transmisión segura de una clave secreta entre entidades X e Y (procSTKxy).

El anexo D trata los procedimientos para la utilización del sistema de cifrado de portadora HFX40 para obtener confidencialidad de mensajes en los terminales facsímil.

El anexo E describe el algoritmo de troceo HFX40-I, en términos de su utilización, a los cálculos necesarios y la información que ha de intercambiarse entre los terminales facsímil para conseguir la integridad de un mensaje facsímil transmitido como una alternativa seleccionada o previamente programada a la encriptación del mensaje.

Orígenes

La Recomendación UIT-T T.36 ha sido preparada por la Comisión de Estudio 8 (1997-2000) del UIT-T y fue aprobada por el procedimiento de la Resolución N.° 1 de la CMNT el 2 de julio de 1997.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución N.° 1 de la CMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

PROPIEDAD INTELECTUAL

UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT ha recibido/no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

ÍNDICE

Página

Recomendación T.36

CAPACIDADES DE SEGURIDAD PARA SU UTILIZACIÓN CON TERMINALES FACSÍMIL DEL GRUPO 3

(Ginebra, 1997)

1 Alcance

Esta Recomendación define dos soluciones técnicas independientes que pueden utilizarse en el contexto de una transmisión facsímil segura:

- una solución basada en los algoritmos HKM/HFX40, que se describen en el anexo A y en el anexo G/T.30;
- una solución basada en el algoritmo RSA descrito en el anexo B y en el anexo H/T.30.

2 Referencias normativas

Las siguientes Recomendaciones del UIT-T y otras referencias contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y otras referencias son objeto de revisiones por lo que se preconiza que los usuarios de esta Recomendación investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y otras referencias citadas a continuación. Se publica periódicamente una lista de las Recomendaciones UIT-T actualmente vigentes.

– Recomendación UIT-T T.30 (1996), Procedimientos de transmisión de documentos por facsímil por la red telefónica general conmutada.

3 Abreviaturas

Esta Recomendación utiliza las siguientes abreviaturas.

ASCII	Código de normalización norteamericano para intercambio de información (*american standard code for information interchange*)
B(n)	Valor de base (n)
ESH	Troceo simple encriptado y aleatorizado (24 cifras decimales) (*encrypted and scrambled plain hash)*
ESIM	Mensaje de integridad encriptado y aleatorizado. Un número de 12 cifras decimales (*encrypted scrambled integrity message*)
ESSK	Clave secreta encriptada y aleatorizada. Un número de 12 cifras decimales (*encrypted scrambled secret key*)
HKM	Algoritmo de encriptación HKM (*HKM encryption algorithm*)
HKMD+1	Encriptación doble que utiliza el algoritmo HKM (*double encryption using the HKM algorithm*)
IDx	Últimas seis cifras de la identificación del facsímil (número telefónico del facsímil) de X [*last six digits of the facsimile identification (facsimile telephone number) of X*]
IDy	Últimas seis cifras de la identificación del facsímil (número telefónico del facsímil) de Y [*last six digits of the facsimile identification (facsimile telephone number) of Y*]
IM	Mensaje de integridad utilizado para confirmar o denegar la integridad del mensaje recibido (12 cifras decimales) (*integrity message used to confirm or deny integrity of the received message*)
IMy	Mensaje de integridad generado por Y para confirmar o denegar la integridad del mensaje recibido. Un número decimal de 12 cifras (*integrity message generated by Y to confirm or deny integrity of the received message*)
mod n	Módulo aritmético que utiliza base n (*modulo arithmetic using base n*)

MPx — Primitiva mutua de X. Un número de 16 cifras decimales, sólo generable por X. MPx es producida por X utilizando el algoritmo HKM con primitivas formadas a partir de UINx, UCNx, IDx e IDy (*mutual primitive of X*)

MPy — Primitiva mutua de Y (*mutual primitive of Y*)

OT — Clave de un solo uso. Un número de 6 a 64 cifras decimales acordado por ambos usuarios (*one-time key*)

OTx — Clave de un solo uso empleada por primera vez por X en el registro de X' con Y (*one-time key as first used by X in X's registration with Y*)

OTy — Clave de un solo uso empleada por primera vez por Y, cuando Y inicia su registro con X para completar el registro mutuo, sea diferente o idéntica a Otx (*one-time key as first used by Y*)

PH — Troceo simple del mensaje (24 cifras decimales) (*plain hash of the message*)

P(n) — Valor de fase (n) [*phase value (n)*]

Primitiva — Número compuesto de 64 cifras formado a partir de UIN y UCN

ProcREGxy — Procedimiento de registro entre X e Y (*procedure for registration between X and Y*)

procSTKxy — Procedimiento para la transmisión segura de una clave secreta de X a Y (*procedure for the secure transmission of a secret key from X to Y*)

PRS — Secuencia seudoaleatoria (*pseudoRandom sequence*)

RCN — Número encriptado registrado. Un número de 16 cifras decimales (*registered crypt number*)

RNCn — Número aleatorio no secreto asociado con una SCn. Un número de 4 cifras decimales (*non-secret random number associated with an SCn*)

RNIM — Número aleatorio no secreto asociado con un IM. Un número de 4 cifras decimales (*non-secret random number associated with an IM*)

RNK — Número aleatorio no secreto utilizado para proporcionar variaciones de las primitivas generadas a partir de MPx cuando se encripta una SK. Un número de 4 cifras decimales (*non-secret random number used to provide variation of the primitives generated from MPx when encrypting an SK*)

RNSRn — Número aleatorio no secreto asociado con una SRn. Un numero de 4 cifras decimales (*non-secret random number associated with an SRn*)

RNSSn — Número aleatorio no secreto asociado con una SSn. Un número de 4 cifras decimales (*non-secret random number associated with an SSn*)

SCn — Clave secreta de petición de identificación, número n. Un número de 12 cifras decimales (*secret challenge key, number n*)

SH — Troceo simple aleatorizado (24 cifras decimales) (*scrambled plain hash*)

SK — Clave secreta que puede ser una SCn, SRn, SSn, etc. Un número de 12 cifras decimales (*secret key*)

SRn — Clave secreta de respuesta, número n. Un número de 12 cifras decimales (*secret response key, number n*)

SS — Clave secreta de sesión utilizada con el algoritmo de integridad HFX40-I (12 cifras decimales) (*secret session key*)

SSK — Clave secreta aleatorizada. Un número de 12 cifras decimales (*scrambled secret key*)

SSn — Clave secreta de sesión, número n que se utiliza con el sistema de cifrado y/o troceo. Un número de 12 cifras decimales (*secret session key, number n*)

SSx — Clave secreta de sesión generada por X para ser utilizada con el algoritmo de cifrado HFX40 (12 cifras decimales) (*secret session key generated by X*)

TKx — Clave de transferencia, una encriptación de MPx generado por X. Un número de 16 cifras decimales (*transfer key, an encryption of MPx generated by X*)

UCN — Número encriptado único, por ejemplo UCNx, UCNy. Un número de 16 cifras sólo conocido por el sistema (*unique crypt number*)

UIN	Número de identidad único, por ejemplo UINx, UINy. Un número de 48 cifras decimales sólo conocido por el sistema (*unique identity number*)
UIT-T	Unión Internacional de Telecomunicaciones – Sector de Normalización de las Telecomunicaciones
X	Nombre de una entidad
x	Sufijo que identifica propiedad o generación por X
XOR'd	Aplicación de la operación lógica O exclusivo (*exclusively OR'd*)
Y	Nombre de una segunda entidad
y	Sufijo que identifica propiedad o generación por Y

Anexo A

Procedimientos para la transmisión segura de documentos facsímil del grupo 3 utilizando los sistemas HKM y HFX

A.1 Introducción

A.1.1 Este anexo describe los procedimientos utilizados por los terminales para la transmisión de documentos facsímil del grupo 3 a fin de asegurar las comunicaciones utilizando los sistemas HKM y HFX.

A.1.2 La utilización de este anexo es opcional.

A.1.3 La corrección de errores definida en el anexo A/T.30 o el anexo C/T.30 (según convenga), es obligatoria.

A.2 Descripción del procedimiento de transmisión segura de documentos facsímil

A.2.1 Los sistemas HKM y HFX proporcionan las siguientes capacidades para las comunicaciones seguras de documentos entre entidades (terminales u operadores de terminales):

– autenticación mutua de entidades;

– establecimiento de claves secretas de sesión;

– confidencialidad de los documentos;

– confirmación de recibo;

– confirmación o denegación de la integridad de los documentos.

A.2.2 Funciones

La gestión de claves se efectúa mediante el sistema definido en el anexo C. Se definen dos procedimientos: el primero es el registro y el segundo la transmisión segura de una clave secreta. El registro establece secretos mutuos y permite que todas las transmisiones posteriores se efectúen con seguridad. En las transmisiones posteriores, el sistema HKM proporciona autenticación mutua, una clave secreta de sesión para la confidencialidad y la integridad de los documentos, la confirmación de recibo o la confirmación o denegación de la integridad de los documentos.

La confidencialidad de los documentos se obtiene utilizando el sistema de cifrado definido en el anexo D. El cifrado utiliza una clave de doce cifras decimales que sea aproximadamente equivalente a 40 bits.

La integridad del documento se obtiene mediante el sistema definido en el anexo E. Este anexo E define el algoritmo de troceo, incluidos los correspondientes cálculos y el intercambio de información.

Anexo B

Seguridad en el facsímil G3 basada en el algoritmo RSA

B.1 Preámbulo

(El preámbulo queda deliberadamente en blanco.)

B.2 Introducción

Este anexo especifica los mecanismos para ofrecer características de seguridad basadas en el mecanismo criptográfico RSA.

B.3 Referencias

– ISO/CEI 9796:1991, *Information technology – Security techniques – Digital Signature Scheme Giving Message Recovery*.

– RIVEST (R.L.), SHAMIR (A.), ADLEMAN (L.): A method for obtaining digital signatures and public-key cryptosystems, anexo A: RSA, *CACM (Communications of the ACM)*, vol. 21, N.° 2, pp. 120-126, 1978.

– 2° ISO/CEI CD 10118-3:1995.

– ISO/CEI JTC 1/SC 27 N1108:

  • SHA-1 (Secure Hash Algorithm), descrito en *Secure Hash Standard*, FIPS (Federal Information Processing Standard) PUB 180-1, abril 1995, algoritmo que procede del NIST (National Institute of Standardization) de Estados Unidos.

  • MD-5 (RFC 1321).

– ISO/CEI 9979:1991, *Information technology – Security techniques – Hash functions – Part 3: Dedicated hash functions – Procedures for the registration of cryptographic algorithms*.

B.4 Descripción técnica

El anexo H/T.30 contiene una descripción completa de esta solución.

Anexo C

Procedimiento de utilización del sistema de gestión de claves HKM para la transmisión segura de documentos por facsímil

C.1 Alcance

El propósito de este anexo es definir el sistema de gestión de claves HKM que será utilizado con terminales facsímil para permitir el intercambio seguro de claves.

El sistema de gestión de claves HKM está concebido para ser empleado con todo tipo de terminales facsímil especializados, pero es también aplicable a los sistemas facsímil con base informática.

Este anexo describe la utilización del algoritmo HKM en dos procedimientos principales, procREGxy y procSTKxy para proporcionar:

– autenticación mutua (descrita en C.5.3 y C.6);

– la utilización de un sistema de cifrado para proporcionar confidencialidad del mensaje (anexo D);

– la utilización de una función "troceo" (*hash*) para proporcionar integridad del mensaje (anexo E);

– el intercambio seguro de claves que pueden ser claves de interrogación, claves de respuesta para autenticación y claves de sesión para confidencialidad del mensaje o integridad del mensaje (descritas en C.5 y C.6).

Se emplea notación algebraica para asistir a la presentación de protocolos y procedimientos de gestión, como se indica en C.2.

El sistema HKM está basado en la utilización de 19 números primos. Estos mismos 19 números primos se utilizan también con el algoritmo de cifrado del mensaje que se describe en el anexo D y el algoritmo de troceo para la integridad de mensaje que se describe en el anexo E. No obstante, esta Recomendación no incluye el algoritmo de cifrado de mensaje ni el algoritmo de troceo para la integridad del mensaje.

En C.6 se dan ejemplos de cálculos que pueden utilizarse para verificar la aplicación de este anexo.

El sistema de gestión de claves HKM está amparado por los derechos de propiedad intelectual; sin embargo, el poseedor de esos derechos ha acordado seguir la norma de conducta de la TSB. La TSB puede facilitar más detalles al respecto.

C.2 Convenios

C.2.1 Generalidades

La notación algebraica detallada en C.2.2 se utiliza para describir protocolos y procedimientos de gestión de claves.

C.2.2 Símbolos

[ ]	encierra el mensaje
{ }	encierra el algoritmo
( )	encierra las primitivas
< >	encierra información que se almacena
> <	encierra información que se extrae de la memoria
&	fusión o modificación, por ejemplo UCNx con IDx, sin alterar su longitud
RCNx>>>>>>	RCNx enviado a Y
>>>>>>RCNx	RCNx recibido de X
HKM+1	encriptación que utiliza el algoritmo HKM
HKM–1	decriptación que utiliza el algoritmo HKM
HKMD+1	encriptación doble que utiliza el algoritmo HKM
HKMD–1	decriptación doble que utiliza el algoritmo HKM

C.3 Descripción del algoritmo HKM para su utilización con terminales facsímil

El algoritmo HKM utiliza números secretos específicos de terminal y otras variables específicas de usuario para formar primitivas que, junto con una clave de encriptación, proporcionan los números de entrada para cálculos que utilizan aritmética modular. Los números secretos UIN y UCN se almacenan con seguridad en el terminal facsímil en el proceso de fabricación o por encargo especial. No hay necesidad de interreferenciarlos con ninguna forma de número de serie.

Los módulos para la aritmética modular se extraen de un conjunto de 19 números primos especiales almacenados en el terminal.

La salida de los procesos aritméticos modulares son secuencias seudoaleatorias largas que se utilizan para encriptar un mensaje.

El algoritmo HKM se utiliza también en un modo irreversible, es decir, tiene lugar un proceso de encripción, pero no se puede efectuar el proceso inverso.

Las características precedentes forman la base criptográfica para los dos procedimientos, procREGxy y procSTKxy.

En C.6 figuran los detalles de la aritmética y de los números primos especiales.

C.4 Modo registro

C.4.1 Procedimiento para el registro entre entidades X e Y (procREGxy)

Para que X se registre con Y, X genera un número irreversible combinando criptográficamente UINx, y UCNx con IDx e IDy. El número de 16 cifras formado es MPx, que se utiliza para encriptar y transferir claves con seguridad como se explica en las subcláusulas siguientes.

En un medio seguro fuera de las transmisiones de registro, los usuarios acuerdan un OT. El usuario en X selecciona el modo registro e ingresa la identificación de facsímil (número telefónico de facsímil) de Y. IDx e IDy forman la base de las otras primitivas utilizadas por el algoritmo. El usuario en X ingresa también OTx.

X utiliza OTx con HKM+1 para encriptar MPx para formar TKx que se envía a Y. En Y, se ingresa OTx y se utiliza con HKM–1 para descriptar TKx y recuperar MPx.

Y no almacena MPx sino que lo encripta inmediatamente utilizando HKM+1 con primitivas formadas a partir de UINy y UCNy modificadas por IDx e IDy para formar RCNy. Y envía RCNy a X para ser almacenado. Y no tiene necesidad de almacenar RCNy, X lo devolverá abiertamente a Y cuando X inicia luego una transmisión segura a Y.

Los dos terminales implícitamente se autentican a sí mismos pues X es el único terminal que puede crear la MPx referido a Y, e Y es el único terminal que puede recuperar el MPx del RCNy.

procREGxy se puede mostrar utilizando la notación algebraica:

X	**Y**
> UINx, UCNx <	> UINy, UCNy <
MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy] TKx = (OTx){HKM+1}[MPx] TKx >>>>>>	
	>>>>>> TKx MPx = (OTx){HKM–1}[TKx] RCNy = (UINy, UCNy & IDx & IDy){HKM+1}[MPx] RCNy >>>>>>
>>>>>> RCNy <RCNy>	

En C.6.4 figura una descripción y ejemplos de todos los cálculos para procREGxy.

C.4.2 Procedimientos para el registro entre entidades Y y X (procREGyx)

Para completar el registro, Y lleva a cabo un procedimiento idéntico, procREGyx, con X, que establece MPy y RCNx. (El OTy acordado por los usuarios en Y y X puede ser igual o diferente al OTx usado durante procREGxy.)

procREGyx se puede representar utilizando la siguiente notación algebraica:

Y	**X**
>UINy, UCNy<	> UINx, UCNx <
MPy = (UINy, UCNy & IDy & IDx){HKM+1}[UCNy & IDy & IDx] TKy = (OTy){HKM+1}[MPy] TKy>>>>>>	
	>>>>>>TKy MPy = (OTy){HKM–1}[TKy] RCNx = (UINx, UCNx & IDy & IDx){HKM+1}[MPy] RCNx>>>>>>
>>>>>>RCNx <RCNx>	

C.4.3 Procedimiento de registro en una sola llamada

Los dos registros separados que utilizan procREGxy y procREGyx se pueden combinar en una sola llamada que se muestra a continuación utilizando la notación algebraica. En este ejemplo, X inicia la llamada.

X

> UINx, UCNx <

MPx = (UINx, UCNx & IDx & IDy) {HKM+1} [UCNx & IDx & IDy]
TKx = (OTx) {HKM+1} [MPx]
TKx >>>>>>

>>>>>> RCNy, TKy
< RCNy >
MPy = (OTy) {HKM–1} [TKy]
RCNx = (UINx, UCNx & IDy & IDx) {HKM+1} [MPy]
RCNx >>>>>>

Y

>UINy, UCNy<

>>>>>> TKx
MPx = (OTx) {HKM–1} [TKx]
RCNy = (UINy, UCNy & IDx & IDy) {HKM+1} [MPx]

MPy = (UINy, UCNy & IDy & IDx){HKM+1}[UCNy & IDy & IDx]
TKy = (OTy) {HKM+1} [MPy]
RCNy, Tky >>>>>>

>>>>>> RCNx
< RCNx >

C.4.4 Autenticación del registro

La autenticación del registro se incluye en los procedimientos para los registros entre entidades X e Y mediante la provisión de intercambios de petición de identificación/respuesta entre X e Y. La petición de identificación/respuesta utiliza procSTKxy y procSTKyx como se describe en C.5.1. A continuación se muestra un ejemplo en notación algebraica con el procedimiento para el registro en ambos sentidos en una sola llamada.

X

>UINx, UCNx<
MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy]
TKx = (OTx){HKM+1}{MPx}

<SC0x> by procSTKxy = ESSC0x
TKx, RNC0x, ESSC0x>>>>>>

>>>>>>RCNy, TKy, RNSR0y, ESSR0y, RNC0y, ESSC0y

<RCNy>

MPy = (OTy){HKM–1}[TKy]
RCNx = (UINx, UCNx & IDy & IDx){HKM+1}[MPy]

ESSR0y by procSTKyx = SR0y
Compare SR0y with <SC0y>

ESSC0y by STKyx = SC0y
SC0y = SR0x
SR0x by STKxy = ESSR0x

RCNx, RNSR0x, ESSR0x>>>>>>

Y

>UINy, UCNy<

>>>>>TKx, RNC0x, ESSC0x

MPx = (OTx){HKM–1}[TKx]
RCNy = (UINy, UCNy & IDx & IDy){HKM+1}[MPx]

MPy = (UINy, UCNy & IDy & IDx){HKM+1}[UCNy & IDy & IDx]
TKy = (OTy){HKM+1}[MPy]

ESSC0x by procSTKxy = SC0x
SC0x = SR0y
SR0y by procSTKyx = ESSR0y

<SC0y> by procSTKyx = ESSC0y

RCNy, TKy, RNSR0y, ESSR0y, RNC0y, ESSC0y>>>>>>

>>>>>>RCNx, RNR0x, ESSR0x

<RCNx>

ESSR0x by procSTKxy = SR0x
Compare SR0x with <SC0y>

Si la SC0x de petición de identificación es igual a la SR0y de respuesta y la SC0y de petición de identificación es igual a la SR0x de respuesta, se obtiene autenticación mutua de registro entre X e Y.

C.5 Modo seguro

Una vez que se ha establecido el registro de MPx y MPy, se utiliza el algoritmo HKM para proporcionar los medios de resguardar la comunicación de claves secretas entre X e Y. Las claves secretas pueden ser claves SC, SR o SS. El procedimiento, procSTKxy, utilizado se describe en las siguientes subcláusulas.

C.5.1 Procedimiento para la transmisión segura de SK de X a Y (procSTKxy)

SKx, una clave secreta comunicada con seguridad de X a Y, se aleatoriza y encripta utilizando HKMD+1 para formar ESSKx. Las primitivas utilizadas con HKMD+1 son generadas a partir de MPx y modificadas por RNKx. Este número aleatorio no secreto se envía libremente a Y junto con RCNy y ESSKx.

Y recupera MPx utilizando RCNy y desaleatoriza y descripta ESSKx por medio de HKMD–1 para recuperar SKx. Las primitivas utilizadas con HKMD–1 se derivan de MPx, modificada por RNKx como en X.

procSTKxy puede representarse utilizando la siguiente notación algebraica:

X
> UINx, UCNx, RCNy <

MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy]
ESSKx = (MPx & RNKx){HKMD+1}[SKx]
RCNy, RNKx, ESSKx >>>>>>

Y
> UINy, UCNy, RCNx <

>>>>>> RCNy, RNKx, ESSKx
MPx = (UINy, UCNy & IDx & IDy){HKM–1}[RCNy]
SKx = (MPx & RNKx){HKMD–1}[ESSKx]

En C.6.5 figura una descripción y ejemplos de todos los cálculos de procSTKxy.

C.5.2 Utilización de procSTKxy y procSTKyx en modo seguro

Una vez que X e Y hayan completado el registro, todas las futuras transmisiones se pueden hacer con seguridad. El algoritmo HKM se utiliza en el modo seguro para recrear MPx y MPy a fin de permitir la transferencia segura de claves secretas utilizando procSTKxy y procSTKyx.

C.5.3 Autenticación mutua de X e Y

En este contexto X ha iniciado la llamada enviando un mensaje seguro a Y.

En X
- X ingresa el número de teléfono facsímil de Y;
- X regenera MPx, previamente utilizada en el registro entre X e Y;
- X genera SC1x y RNC1x y almacena SC1x;
- X utiliza procSTKxy con MPx, RNC1x y SC1x para formar ESSC1x;
- X envía RCNy, RNC1x y ESSC1x a Y.

En Y
- Y descripta RCNy para formar MPx;
- Y utiliza procSTKxy con MPx, RNC1x y ESSC1x para recuperar SC1x;
- Y regenera MPy;
- Y utiliza SC1x como clave secreta de respuesta, SR1y, y regenera RNSR1y;
- Y utiliza procSTKyx con MPy, RNSR1y y SR1 para formar ESSR1y;
- Y genera SC1y y RNC1y y almacena SC1y;
- Y utiliza procSTKyx con MPy, RNC1y y SC1y para formar ESSC1y;
- Y envía RCNx, RNSR1y, ESSR1y, RNC1y y ESSC1y a X.

En X
– X descripta RCNx para formar MPy;
– X utiliza procSTKyx con MPy, RNSR1 y ESSR1y para formar SR1y;
– X compara SR1y con SC1x, si es igual Y es autenticada a X;
– X utiliza procSTKyx con MPy, RCN1y y ESSC1y para recuperar SC1y;
– X utiliza SC1y como clave secreta de respuesta, SR1x, y regenera RNSR1x;
– X utiliza procSTKxy con MPx, RNSR1x y SR1x para formar ESSR1x;
– X envía RNSR1x y ESSR1x a Y.

En Y
– Y utiliza procSTKxy con MPx, RNSR1y y ESSR1x para recuperar SR1x;
– Y compara SR1x con SC1y, si fuera igual, X es autenticado a Y.

En este punto X e Y han intercambiado RCNx y RCNy y completado el intercambio mutuo de petición de identificación respuesta. Si SC1x es igual a SR1y, y SC1y es igual a SR1x, se obtiene autenticación mutua.

Se puede representar el proceso de autenticación mutua de X e Y utilizando la siguiente notación algebraica:

X	**Y**
> UINx, UCNx, RCNy <	> UINy, UCNy, RCNx <
< SC1x > by procSTKxy = ESSC1x RCNy, RNC1x, ESSC1x >>>>>>	
	>>>>>>RCNy, RNC1x, ESSC1x < RCNy > ESSC1x by procSTKxy = SC1x SC1x = SR1y SR1y by procSTKyx = ESSR1y < SC1y > by procSTKyx = ESSC1y RCNx, RNSR1y, ESSR1y, RNC1y, ESSC1y >>>>>>
>>>>>> RCNx, RNSR1y, ESSR1y, RNC1y, ESSC1y < RCNx > ESSR1y by procSTKyx = SR1y Compare SR1y with < SC1x > ESSC1y by procSTKyx = SC1y SC1y = SR1x SR1x by procSTKxy = ESSR1x RNSR1x, ESSR1x >>>>>>	
	>>>>>> RNSR1x, ESSR1x ESSR1x by procSTKxy = SR1x Compare SR1x with < SC1y >

Si la puesta a prueba SC1x es igual a la respuesta SR1y, y la puesta a prueba SC1y es igual a la respuesta SR1x se obtiene autenticación mutua.

C.5.4 Establecimiento de claves secretas de sesión entre X e Y

En este contexto X ha iniciado una llamada enviando un mensaje seguro Y y la autenticación mutua se ha establecido satisfactoriamente como se indica en C.5.3.

En X
– X regenera MPx previamente utilizado en registro entre X e Y;
– X regenera SS1x y RNSS1x;
– X utiliza procSTKxy con MPx, RNSS1x y SS1x para formar ESSS1x;
– X envía RCNy, RNSS1x y ESSS1x a Y.

En Y
– Y descripta RCNy para recuperar MPx;
– Y utiliza procSTKxy con MPx, RNSS1x y ESSS1x para recuperar SS1x.

El proceso de establecimiento de clave secreta de sesión entre X e Y puede indicarse utilizando la siguiente notación algebraica:

X	**Y**
< RCNy >	< RCNx >
SS1x by procSTKxy = ESSS1x RCNy, RNSS1x, ESSS1x >>>>>>	
	>>>>>> RCNy, RNSS1x, ESSS1x ESSS1x by procSTKxy = SS1x

X e Y utilizan SS1x con el codificador de transporte, HFX40, para encriptar y descriptar el mensaje principal a fin de proporcionar confidencialidad (anexo D) y/o con el algoritmo de troceo, HFX40-I, para proporcionar integridad de mensaje (anexo E) durante la transmisión.

C.5.5 Confirmación de recibo

En este contexto X inició una llamada para enviar un mensaje seguro a Y, se estableció la autenticación mutua como se indica en C.5.3, se intercambió SS1x con seguridad como se indica en C.5.4, y el mensaje fue enviado. Al final del mensaje X envía SC2x a Y, que Y utiliza como SR2y si el mensaje se recibió intacto.

En X
- X regenera MPx, previamente utilizado en registro entre X e Y;
- X genera SC2x y RNC2x y almacena SC2x;
- X utiliza procSTKxy con MPx, RNC2x y SC2x para formar ESSC2x;
- X envía RCNy, RNC2x y ESSC2x a Y.

En Y
- Y descripta RCNy para formar MPx;
- Y utiliza procSTKxy con MPx, RNC1x y ESSC2x para recuperar SC2x;
- Y regenera MPy;
- Y utiliza SC2x como SR2y y genera RNSR2y;
- Y utiliza procSTKyx con MPy, RNSR2y y SR2y para formar ESSR2y;
- Y envía RCNx, RNSR2y y ESSR2y a X.

En X
- X descripta RCNx para MPy;
- X utiliza procSTKyx con MPy, RNSR2y y ESSR2y para recuperar SR2y;
- X compara SR2y con SC2x, si es igual Y tiene recepción confirmada con X.

El proceso de confirmación de recepción puede representarse utilizando la siguiente notación algebraica:

X	**Y**
> RCNy <	> RCNx <
< SC2x > by procSTKxy = ESSC2x RCNy, RNC2x, ESSC2x >>>>>>	
	>>>>>>RCNy, RNC2x, ESSC2x ESSC2x by procSTKxy = SC2x SC2x = SR2y SR2y by procSTKyx = ESSR2y RCNx, RNSR2y, ESSR2y >>>>>>
>>>>>> RCNx, RNSR2y, ESSR2y ESSR2y by procSTKyx = SR2y Compare SR2y with < SC2x >	

Si SR2y, procedente de Y, es igual a SC2x almacenado por X, X acepta la confirmación de recibo del mensaje por Y.

C.5.6 Confirmación o denegación de integridad

En este contexto, X inició una llamada para enviar un mensaje a Y de protección de integridad, se ha establecido la autenticación mutua satisfactoriamente como se indica en el C.5.3, se ha intercambiado SSx con seguridad como se indica en el C.5.4, y el mensaje ha pasado o no la prueba de Y de comprobación de integridad (anexo E).

Y genera IMy para confirmar o denegar la integridad del mensaje recibido. IMy es un número aleatorio de 12 cifras seleccionado a partir de las cifras 2 a 9. Una cifra se selecciona aleatoriamente para ser reemplazada por un "1" para indicar confirmación de integridad o por un "0" para indicar denegación.

En Y
- Y genera un número aleatorio de 12 cifras con las cifras 2 a 9;
- Y genera un número aleatorio entre 1 y 12 que será el puntero de reemplazo del número aleatorio anterior;
- Y cambia el dígito en el puntero de reemplazo por un "1" o un "0" para formar IMy;
- Y genera RNIMy;
- Y utiliza procSTKyx con MPy, RNIMy e IMy para formar ESIMy;
- Y envía RCNx, RNIMy y ESIMy a X.

En X

– X descripta RCNx para formar MPy;
– X utiliza procSTKyx con MPy, RNIMy y ESIMy para recuperar IMy;
– X comprueba si IMy contiene un "1" o un "0" para confirmación o rechazo de integridad.

El proceso de confirmación de integridad puede representarse utilizando la siguiente notación algebraica:

Si el mensaje de integridad secreto contiene un "1", la integridad es confirmada; si contiene un "0", la integridad es denegada.

Ejemplos de respuesta:

IMy = 257795**1**99982 Integridad confirmada.

IMy = 3**1**7736845378 Integridad confirmada.

IMy = 738543680892 Integridad denegada.

IMy = 4577452**0**4639 Integridad denegada.

C.6 El algoritmo HKM

C.6.1 Introducción

En C.6 se describe el algoritmo HKM en términos de los números que han de ser almacenados y las reglas para los cálculos que se efectúan utilizando estos números durante el procedimiento de registro, procREG, y el procedimiento de transmisión segura de una clave, procSTK. Las regla se explica mejor utilizando ejemplos numéricos. Pueden utilizarse cálculos que empleen valores de prueba para verificar la implementación.

C.6.2 Información almacenada

Todos los terminales están equipados con los mismos 19 números primos modulantes del sistema.

32603 32507 32183 32003 31847 31607 31583 31547 31259
31139 30803 30539 30467 30347 30323 30203 29879 29759 29663

Los primeros nueve números se utilizan con el algoritmo HKM para registro, autenticación y otras funciones de gestión de clave. Los 19 números se utilizan con el algoritmo de confidencialidad de mensaje, conforme al anexo D, y con el algoritmo de integridad del mensaje, según el anexo E.

C.6.3 Información almacenada con seguridad

Cada terminal está equipado, mediante un proceso adecuado, con dos números de cifras decimales generados al azar, que se almacenan con seguridad en el terminal. Estos son el UIN de 48 cifras y el UCN de 16 cifras. UIN y UCN se utilizan con otros números de identificación para formar las primitivas para el algoritmo HKM.

Ejemplos de prueba para X e Y son:

UINx = 345092978336094172898029844342879120988727823781
UCNx = 1333908734565521

UINy = 973557693837783148353709167436722873449819767357
UCNy = 7598247578649467

C.6.4 Modo registro

C.6.4.1 procREGxy utilizando notación algebraica

X	**Y**
>UINx, UCNx<	> UINy, UCNy <
MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy] TKx = (OTx){HKM+1}[MPx] TKx >>>>>>	
	>>>>>> TKx MPx = (OTx){HKM–1}[TKx] RCNy = (UINy, UCNy & IDx & IDy){HKM+1}[MPx] RCNy >>>>>>
>>>>>> RCNy <RCNy>	

Las subcláusulas siguientes utilizan valores de prueba para mostrar todos los cálculos utilizados en procREGxy:

C.6.4.2 Cálculos en X para obtener MPx

MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy]

C.6.4.2.1 Cálculos preliminares con las primitivas (UINx, UCNx & IDx & IDy)

UINx = 3450929783360941728980298443428791209887278237811333908734565521
UCNx = 1333908734565521
IDx = 642092
IDy = 538249

Se forma una primitiva de 64 cifras concatenando UINx y UCNx.

Primitiva = 3450929783360941728980298443428791209887278237811333908734565521

Esta primitiva se divide en dos números de 32 cifras, 9 valores de primitivas de fase (P(0) a P(8)) se obtienen del primer número y 9 valores de primitivas de base (B(0) a B(8)) se obtienen del segundo número. Los valores de fase se calculan dividiendo el primer número en 7 conjuntos de 4 cifras y 2 conjuntos de 2 cifras. Los valores de base se calculan exactamente del mismo modo a partir del segundo número.

Los valores de primitivas de fase se modifican por el agregado de productos incrementales de 101 y los valores de base se modifican por el agregado de productos incrementales de 79. La modificación por 101 y 79 se utiliza para asegurar que la modulación por los números primos se inicia lo más pronto posible.

Utilizando la primitiva = 3450929783360941728980298443428791209887278237811333908734565521 los valores de primitivas de fase y de base son los indicados a continuación:

P(0)	3450 + (0 * 101) = 3450	B(0)	9120 + (0 * 79) = 9120
P(1)	9297 + (1 * 101) = 9398	B(1)	9887 + (1 * 79) = 9966
P(2)	8336 + (2 * 101) = 8538	B(2)	2782 + (2 * 79) = 2940
P(3)	941 + (3 * 101) = 1244	B(3)	3781 + (3 * 79) = 4018
P(4)	7289 + (4 * 101) = 7693	B(4)	1333 + (4 * 79) = 1649
P(5)	8029 + (5 * 101) = 8534	B(5)	9087 + (5 * 79) = 9482
P(6)	8443 + (6 * 101) = 9049	B(6)	3456 + (6 * 79) = 3930
P(7)	42 + (7 * 101) = 749	B(7)	55 + (7 * 79) = 608
P(8)	87 + (8 * 101) = 895	B(8)	21 + (8 * 79) = 653

Los componentes de 6 cifras de IDx, 642092, y de IDy, 538249, se dividen cada uno en dos grupos de tres cifras (642, 092, 538 y 249) y se utilizan para modificar nuevamente P(0) a P(3) y B(0) a B(3):

P(0) = 3450 + 642 = 4092	B(0) = 9120 + 642 = 9762
P(1) = 9398 + 092 = 9490	B(1) = 9966 + 092 = 10058
P(2) = 8538 + 538 = 9076	B(2) = 2940 + 538 = 3478
P(3) = 1244 + 249 = 1493	B(3) = 4018 + 249 = 4267

C.6.4.2.2 Cálculos preliminares con el mensaje [UCNx & IDx & IDy]

IDx e IDy están concatenados para formar un número de 12 cifras que se agrega (mod 10) a UCNx para formar una UCNx modificada.

IDx concatenada con IDy = 642092538249
UCNx = 1333908734565521
UCNx modificada = 7753823016955521

C.6.4.2.3 Cálculos utilizando HKM+1

El algoritmo HKM+1 utiliza los primeros 9 números primos de C.6.2 como módulos para aplicar la aritmética modular utilizando las 9 primitivas de fase, P(0) a P(8), y las 9 primitivas de base, B(0) a B(8) calculadas en C.6.4.2.1 para generar un PRS de 16 entradas (mod 10). El PRS se agrega (mod 10) a la UCNx para formar MPx como se explica a continuación.

Por ejemplo, utilizando el primer conjunto de valores de fase y de base, P(0), B(0) y el primer número primo:

P(0) se multiplica por B(0)
4092 * 9762 = 39946104
39946104 (mod el primer número primo) = 39946104 (mod 32603) = 7429

7429 se utiliza entonces como nuevo valor de fase, P, y se multiplica por el valor de base, B(0)
7429 * 9762 = 72521898
72521898 (mod el primer número primo) = 72521898 (mod 32603) = 12826

Este proceso se lleva a cabo 16 veces en total (correspondiente al número de cifras en la UCNx modificada). Se repite también para los ocho conjuntos restantes de valores primos, de fase y de base.

Se agregan los resultados del primer cálculo para cada uno de los nueve "conjuntos" de números primos, de fase y de base. El resultado (mod 10) se agrega (mod 10) a la primera cifra de la UCNx modificada para formar la MPx. El proceso se repite para cada cifra de la UCNx modificada.

En el cuadro C.1 figuran los resultados de las operaciones anteriores para producir la MPx, 4314920574868366, a partir del UCNx modificado, 7753823016955521.

Cuadro C.1/T.36 – Cálculos en X para obtener MPx

Número primo B(n) P(n)	32603 9762 4092	32507 10058 9490	32183 3478 9076	32003 4267 1493	31847 1649 7693	31607 9482 8534	31583 3930 9049	31547 608 749	31259 653 895	Total	Última cifra	Número encriptado modificado	**Primitiva mutua**
	7429	9868	26988	2034	10651	5468	112	13734	21773	98057	7	7	**4**
	12826	8473	18636	6265	15802	12096	29581	21864	26183	151726	6	7	**3**
	11892	20587	31629	10250	6652	24076	27890	12025	30085	175086	6	5	**1**
	23024	26963	4168	20652	13780	22878	14690	23843	14853	164851	1	3	**4**
	27809	20460	13954	17825	16309	10355	29559	16471	8719	161461	1	8	**9**
	18880	17370	48	20147	14673	14768	4596	13969	4369	108820	0	2	**2**
	1801	14842	6029	7191	23904	11166	28387	7009	8388	108717	7	3	**0**
	8345	8692	17729	25123	22957	24169	9754	2627	7039	126435	5	0	**5**
	21596	12813	31017	21794	21857	19708	23041	19866	1394	173086	6	1	**7**
	9154	15406	31893	28283	23236	10672	2669	27574	3771	150658	8	6	**4**
	29128	25186	21236	11049	4223	17897	3614	13535	24261	150129	9	9	**8**
	16773	26244	31006	5664	21081	1371	22253	27060	25379	176831	1	5	**6**
	5760	5312	25818	6023	17492	9345	963	16493	5217	92423	3	5	**8**
	21548	19095	4434	1732	22773	14869	26213	27345	30729	168738	8	5	**3**
	29623	6154	5795	29754	5064	20638	24927	491	29018	151464	4	2	**6**
	23719	3604	8452	4417	6622	10579	24227	14605	5800	102025	5	1	**6**
MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy] = 4314920574868366													

C.6.4.3 Cálculos en X para obtener TKx

TKx = (OTx){HKM+1}[MPx]

MPx se encripta por medio del algoritmo HKM+1 para formar la TKx que utiliza OTx.

OTx = 71628582063812097215

OTx se amplía por concatenación para formar una primitiva de 64 cifras para el algoritmo HKM, los 9 valores de primitivas de fase y los 9 valores de primitivas de base se forman a partir de la primitiva de manera similar a la indicada en C.6.4.2.1. Sin embargo, no habrá nuevas modificaciones de P(0) a P(3) ni de B(0) a B(3).

Los resultados de los cálculos con los valores de prueba para formar P(0) a P(8) y B(0) a B(8) utilizando OTx son los siguientes:

Primitiva = 71628582063812097215 71628582063812097215 71628582063812097215 7162

P(0)	7162 + 0 * 101 = 7162	B(0)	1209 + 0 * 79 = 1209
P(1)	8582 + 1 * 101 = 8683	B(1)	7215 + 1 * 79 = 7294
P(2)	638 + 2 * 101 = 840	B(2)	7162 + 2 * 79 = 7320
P(3)	1209 + 3 * 101 = 1512	B(3)	8582 + 3 * 79 = 8819
P(4)	7215 + 4 * 101 = 7619	B(4)	638 + 4 * 79 = 954
P(5)	7162 + 5 * 101 = 7667	B(5)	1209 + 5 * 79 = 1604
P(6)	8582 + 6 * 101 = 9188	B(6)	7215 + 6 * 79 = 7689
P(7)	6 + 7 * 101 = 713	B(7)	71 + 7 * 79 = 624
P(8)	38 + 8 * 101 = 846	B(8)	62 + 8 * 79 = 694

Los valores de fase, P(0) a P(8), los valores de base, B(0) a B(8), y los 9 números primos se utilizan de la misma manera que en C.6.4.2.3. La MPx forma el mensaje y se agrega (mod 10) a la PRS y es así encriptada por el algoritmo HKM+1 para formar TKx. Los resultados de estas operaciones para producir TKx, 5371333066610533, a partir de MPx, 4314920574868366, figuran en el cuadro C.2.

Cuadro C.2/T.36 – Cálculos en X para obtener TKx

Número primo B(n) P(n)	32603 1209 7162	32507 7294 8683	32183 7320 840	32003 8819 1512	31847 954 7619	31607 1604 7667	31583 7689 9188	31547 624 713	31259 694 846	Total	Última cifra	Primitiva mutua	**Clase de transferencia**
	19063	10166	1847	21080	7410	2745	26944	3254	24462	116971	1	4	**5**
	29449	2337	3180	31096	30953	9607	19519	11488	2991	140620	0	3	**3**
	1365	12410	9291	1917	6993	17019	30758	7343	12660	99756	6	1	**7**
	20135	19052	7441	8439	15299	21635	4758	7717	2261	106737	7	4	**1**
	21377	30370	14484	16566	9320	29661	11148	20264	6184	159374	4	9	**3**
	23217	16082	12078	1859	5967	7709	710	25936	9213	102771	1	2	**3**
	30773	16852	4259	8985	23752	6899	26914	453	16986	135873	3	0	**3**
	4534	9521	22736	31290	16191	3546	9930	30296	3641	131685	5	5	**0**
	4302	11222	9227	16644	419	30131	15659	8051	26134	121789	9	7	**6**
	17241	642	21706	17678	17562	3021	7655	7851	6776	100132	2	4	**6**
	11052	1740	449	15669	2626	9813	20166	9239	13694	84448	8	8	**6**
	27241	13830	4014	27960	21138	31373	15427	23582	900	165465	5	6	**1**
	5339	6799	31584	28128	6501	3948	24038	14266	30679	151282	2	8	**0**
	32060	18731	24391	5579	23636	11192	4466	5730	3847	129632	2	3	**5**
	28176	29500	23019	12590	1068	30799	8353	10709	12803	157017	7	6	**3**
	27252	9167	21075	12803	31615	31462	17978	25999	7726	185077	7	6	**3**
TKx = (OTx){HKM+1}[MPx] = 5371333066610533													

C.6.4.4 Cálculo en Y para recuperar MPx por la decripción de TKx

MPx = (OTx){HKM–1}[TKx]
OTx = 71628582063812097215

Los mismos procesos que los utilizados para obtener TKx en C.6.4.3 se llevan a cabo utilizando OTx. Se genera la misma PRS (mod 10); sin embargo, para HKM–1 ésta se sustrae (mod 10) de la TKx que forma el "mensaje". El proceso de sustracción decripta al TKx para recuperar MPx.

Los resultados de los cálculos con los valores de prueba para formar P(0) a P(8) y B(0) a B(8) utilizando la OTx precedente son los siguientes:

Primitiva = 71628582063812097215 71628582063812097215 71628582063812097215 7162

P(0) 7162 + (0 * 101) = 7162
P(1) 8582 + (1 * 101) = 8683
P(2) 638 + (2 * 101) = 840
P(3) 1209 + (3 * 101) = 1512
P(4) 7215 + (4 * 101) = 7619
P(5) 7162 + (5 * 101) = 7667
P(6) 8582 + (6 * 101) = 9188
P(7) 6 + (7 * 101) = 713
P(8) 38 + (8 * 101) = 846

B(0) 1209 + (0 * 79) = 1209
B(1) 7215 + (1 * 79) = 7294
B(2) 7162 + (2 * 79) = 7320
B(3) 8582 + (3 * 79) = 8819
B(4) 638 + (4 * 79) = 954
B(5) 1209 + (5 * 79) = 1604
B(6) 7215 + (6 * 79) = 7689
B(7) 71 + (7 * 79) = 624
B(8) 62 + (8 * 79) = 694

Los valores de fase, P(0) a P(8), los valores de base B(0) a B(8), y los 9 números primos se utilizan del mismo modo que en C.6.4.3. La PRS idéntica se genera y sustrae (mod 10) de la TKx para recuperar MPx.

Los resultados de las operaciones anteriores para recuperar MPx, 4314920574868366, mediante TKx, 5371333066610533, se indican en el cuadro C.3.

Cuadro C.3/T.36 – Cálculos en Y para recuperar MPx mediante la decripción de TKx

Número primo B(n) P(n)	32603 1209 7162	32507 7294 8683	32183 7320 840	32003 8819 1512	31847 954 7619	31607 1604 7667	31583 7689 9188	31547 624 713	31259 694 846	Total	Última cifra	Clave de transferencia	**Primitiva mutua**
	19063	10166	1847	21080	7410	2745	26944	3254	24462	116971	1	5	**4**
	29449	2337	3180	31096	30953	9607	19519	11488	2991	140620	0	3	**3**
	1365	12410	9291	1917	6993	17019	30758	7343	12660	99756	6	7	**1**
	20135	19052	7441	8439	15299	21635	4758	7717	2261	106737	7	1	**4**
	21377	30370	14484	16566	9320	29661	11148	20264	6184	159374	4	3	**9**
	23217	16082	12078	1859	5967	7709	710	25936	9213	102771	1	3	**2**
	30773	16852	4259	8985	23752	6899	26914	453	16986	135873	3	3	**0**
	4534	9521	22736	31290	16191	3546	9930	30296	3641	131685	5	0	**5**
	4302	11222	9227	16644	419	30131	15659	8051	26134	121789	9	6	**7**
	17241	642	21706	17678	17562	3021	7655	7851	6776	100132	2	6	**4**
	11052	1740	449	15669	2626	9813	20166	9239	13694	84448	8	6	**8**
	27241	13830	4014	27960	21138	31373	15427	23582	900	165465	5	1	**6**
	5339	6799	31584	28128	6501	3948	24038	14266	30679	151282	2	0	**8**
	32060	18731	24391	5579	23636	11192	4466	5730	3847	129632	2	5	**3**
	28176	29500	23019	12590	1068	30799	8353	10709	12803	157017	7	3	**6**
	27252	9167	21075	12803	31615	31462	17978	25999	7726	185077	7	3	**6**
Primitiva mutua = (OTx){HKM–1}[TKx] = 4314920574868366													

C.6.4.5 Cálculos en Y para obtener RCNy

RCNy = (UINy, UCNy & IDx & IDy){HKM+1}[MPx]

La entidad Y sigue los mismos cálculos preliminares para (UINy, UCNy & IDx & IDy) como los que utiliza la entidad X para calcular MPx en C.6.4.2 pero utilizando UINy y UCNy para formar los valores de fase y de base. Se utilizan las identificaciones IDx e IDy para modificar P(0) a P(3) y B(0) a B(3) como anteriormente. Se utilizan los valores de fase y de base con los primeros 9 números primos de modulación del sistema conforme al algoritmo HKM+1 para crear una PRS (mod 10) la cual, agregada a MPx, forma RCNy, que puede ser comunicada abiertamente.

Los resultados de estos cálculos que utilizan valores de prueba son los siguientes:

Número de identidad único, UINy = 973557693837783148353709167436722873449819767357
Número de encriptación único, UCNy = 7598247578649467

Primitiva = 9735576938377831483537091674367228734498197673577598247578649467

P(0) 9735 + (0 * 101) = 9735
P(1) 5769 + (1 * 101) = 5870
P(2) 3837 + (2 * 101) = 4039
P(3) 7831 + (3 * 101) = 8134

B(0) 2873 + (0 * 79) = 2873
B(1) 4498 + (1 * 79) = 4577
B(2) 1976 + (2 * 79) = 2134
B(3) 7357 + (3 * 79) = 7594

P(4) 4835 + (4 * 101) = 5239
P(5) 3709 + (5 * 101) = 4214
P(6) 1674 + (6 * 101) = 2280
P(7) 36 + (7 * 101) = 743
P(8) 72 + (8 * 101) = 880

B(4) 7598 + (4 * 79) = 7914
B(5) 2475 + (5 * 79) = 2870
B(6) 7864 + (6 * 79) = 8338
B(7) 94 + (7 * 79) = 647
B(8) 67 + (8 * 79) = 699

P(0) a P(3), y B(0) a B(3) se modifican como anteriormente utilizando IDx e IDy
IDx = 642092 IDy = 538249

P(0) = 9735 + 642 = 10377
P(1) = 5870 + 092 = 5962
P(2) = 4039 + 538 = 4577
P(3) = 8134 + 249 = 8383

B(0) = 2873 + 642 = 3515
B(1) = 4577 + 092 = 4669
B(2) = 2134 + 538 = 2672
B(3) = 7594 + 249 = 7843

Los resultados de las operaciones anteriores para producir RCNy, 9865418902725854, mediante MPx, 43149205748688366, figuran en el cuadro C.4.

Cuadro C.4/T.36 – Cálculos en Y para obtener RCNy

Número primo B(n) P(n)	32603 3515 10377	32507 4669 5962	32183 2672 4577	32003 7843 8383	31847 7914 5239	31607 2870 4214	31583 8338 2280	31547 647 743	31259 699 880	Total	Última cifra	Primitiva mutua	**Número encriptado registrado**
	25001	10586	204	13707	28499	20306	29257	7516	21199	156275	5	4	**9**
	13430	15394	30160	5924	632	26519	29357	4614	1335	127365	5	3	**8**
	29909	1609	1288	25579	1669	31481	10416	19840	26654	148445	5	1	**6**
	18063	3304	30138	21293	23808	17664	26941	28398	782	170391	1	4	**5**
	13404	18058	6870	9345	9660	29659	15762	13152	15215	131125	5	9	**4**
	3725	22151	12330	5965	16440	3679	6693	23201	7225	101409	9	2	**1**
	19572	18252	22551	27112	11165	1992	30656	26222	17576	175098	8	0	**8**
	3250	17741	9696	11484	16232	27780	8509	24895	837	120424	4	5	**9**
	12700	4893	397	12570	21097	15746	12624	18095	22401	120523	3	7	**0**
	6993	25503	30928	17270	19684	24617	24356	3528	28799	181678	8	4	**2**
	30336	366	25855	11914	15499	9145	1638	11232	30964	136949	9	8	**7**
	19230	18490	19842	24745	16289	12340	13788	11294	12608	148626	6	6	**2**
	7431	23725	12423	8843	26337	15960	2224	19861	29213	146017	7	8	**5**
	4962	20676	13583	5148	24250	6657	4491	10438	7760	97965	5	3	**8**
	31428	22961	23535	19981	4478	14962	20103	2328	16433	156209	9	6	**5**
	10456	29330	32121	24295	25028	18634	7833	23507	14614	185818	8	6	**4**
Número encriptado registrado, RCNy = (UINy, UCNy & IDx & IDy) {HKM+1} [MPx] = 9865418902725854													

C.6.5 Modo seguro

C.6.5.1 Procedimiento procSTKxy que utiliza notación algebraica

Los siguientes puntos emplean valores de prueba para indicar todos los cálculos utilizados en procSTKxy que emplean HKMD para transferir la clave SKx, entre X e Y. La clave secreta puede ser específicamente una SCn, SRn, SSn, etc.

X

> UINx, UCNx, RCNy <

(UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy] = MPx
(MPx & RNKx){HKMD+1}[SKx] = ESSKx
RCNy, RNKx, ESSKx >>>>>>

Y

> UINy, UCNy <

>>>>>> RCNy, RNKx, ESSKx
MPx = (UINy, UCNy & IDx & IDy){HKM–1}[RCNy]
SKx = (MPx & RNKx){HKMD–1}[ESSKx]

C.6.5.2 Cálculos en X para recrear MPx

> UINx, UCNx, RCNy <
MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy]

Los cálculos y valores de prueba son idénticos a los indicados en el C.6.4.2, con los que se obtiene:

MPx = (UINx, UCNx & IDx & IDy){HKM+1}[UCNx & IDx & IDy] = 4314920574868366

C.6.5.3 Cálculos en X para formar ESSKx utilizando HKMD+1

ESSKx = (MPx & RNKx){HKMD+1}[SKx]

SKx está doblemente encriptada mediante la utilización de HKMD+1, siendo la primera encriptación un proceso de aleatorización basado en una PRS (mod 12) de 12 cifras generada mediante HKM para formar SSKx, y la segunda encriptación es la adición (mod 10) normal al "mensaje", SSKx, del la PRS (mod 10) generada mediante HKM.

C.6.5.3.1 Cálculos en X para formar SSKx

Se repite y concatena MPx para formar la primitiva de 64 cifras que se utiliza para calcular los valores de primitivas de fase y de base como se indica a continuación:

Primitiva = 4314920574868366 4314920574868366 4314920574868366 4314920574868366

P(0)	4314 + (0 * 101) = 4314	B(0)	4314 + (0 * 79) = 4314
P(1)	9205 + (1 * 101) = 9306	B(1)	9205 + (1 * 79) = 9284
P(2)	7486 + (2 * 101) = 7688	B(2)	7486 + (2 * 79) = 7644
P(3)	8366 + (3 * 101) = 8669	B(3)	8366 + (3 * 79) = 8603
P(4)	4314 + (4 * 101) = 4718	B(4)	4314 + (4 * 79) = 4630
P(5)	9205 + (5 * 101) = 9710	B(5)	9205 + (5 * 79) = 9600
P(6)	7486 + (6 * 101) = 8092	B(6)	7486 + (6 * 79) = 7960
P(7)	83 + (7 * 101) = 790	B(7)	83 + (7 * 79) = 636
P(8)	66 + (8 * 101) = 874	B(8)	66 + (8 * 79) = 698

RNKx, asociado con SKx, se divide en 2 pares de 2 cifras y el primer par se agrega a P(0) y el segundo par a P(1) para crear nuevos valores de P(0) y P(1); los nuevos valores para B(0) y B(1) se crean de idéntica manera.

RNKx = 3958

P(0) = 4314 + 39 = 4353	B(0) = 4314 + 39 = 4353
P(1) = 9306 + 58 = 9364	B(1) = 9284 + 58 = 9342

Los números primos y los valores de fase y de base se utilizan entonces con HKM para generar una PRS de 12 cifras, (mod 12) + 1 (es decir, se aplica el módulo 12 al número de la columna "Total" y se agrega 1 para producir la PRS). Los resultados de los cálculos figuran en el cuadro C.5.

Cuadro C.5/T.36 – Cálculos en X para formar la PRS del aleatorizador/desaleatorizador (mod 12) + 1

Número primo B(n) P(n)	32603 4353 4353	32507 9342 9364	32183 7644 7688	32003 8603 8669	31847 4630 4718	31607 9600 9710	31583 7960 8092	31547 636 790	31259 698 874	Total	**(mod12) + 1 PRS**
	6266	2151	914	12417	29145	6957	14583	29235	16131	117799	**8**
	19790	5316	2905	29440	5611	1609	13155	12277	6198	96301	**2**
	8744	23883	31733	578	23625	22184	16155	16063	12462	155427	**4**
	14931	19445	3781	12069	21152	30041	19407	26387	8474	155687	**12**
	16864	6074	1630	11875	4235	11332	7267	30675	6901	96853	**2**
	19639	18593	4899	7049	22145	27513	16847	13254	3012	132951	**4**
	3501	10905	19127	28865	15857	16708	702	6495	8023	110183	**12**
	14252	30079	31602	14318	10575	22882	29312	29710	4693	187423	**8**
	28050	7510	90	30210	13411	30157	19899	30454	24778	184559	**12**
	3415	8314	12117	267	23127	18687	7295	30433	8817	112472	**9**
	31130	10165	31857	24788	8396	25475	18646	17077	27502	195036	**1**
	10822	8483	18330	15175	20140	16641	13643	8804	3370	115408	**5**

La PRS del aleatorizador/desaleatorizador resultante se utiliza entonces como aleatorizador para aleatorizar la SKx. La primera cifra en el aleatorizador indica qué cifra en la SKx se debe intercambiar con la primera cifra en la SKx. La segunda cifra en el aleatorizador indica el intercambio de la segunda cifra de la SKx y así sucesivamente para las cifras restantes. Los resultados de este proceso de aleatorización para obtener una SSKx, 721647935700, mediante SKx, 309126704577, se indican a continuación:

PRS del aleatorizador	
8	**3**091267**0**4577 = SKx
2	0**0**9126734577
4	00**91**26734577
12	001**9**26734577
2	0**0**172**6**734579
4	021**7**0**6**734579
12	021607**7**34579
8	02160793**4**577
12	021607934577
9	02160793**75**74
1	**0**2160793574
5	7216**0**7935704
	721647935700 = SSKx

C.6.5.3.2 Cálculos en X para formar ESSKx

Los mismos cálculos que emplean HKM conforme a C.6.5.3.1 se utilizan para generar una PRS (mod 10) de 12 cifras, que se agrega (mod 10) a SSKx, para producir ESSKx.

Los resultados de los cálculos para producir ESSKx, 638378264968, mediante SSKx, 721647935700, figuran en el cuadro C.6.

Cuadro C.6/T.36 – Cálculos en X para formar ESSKx

Número primo B(n) P(n)	32603 4353 4353	32507 9342 9364	32183 7644 7688	32003 8603 8669	31847 4630 4718	31607 9600 9710	31583 7960 8092	31547 636 790	31259 698 874	Total	Última cifra	Clave secreta aleatorizada	**Clave secreta aleatorizada encriptada**
	6266	2151	914	12417	29145	6957	14583	29235	16131	117799	9	7	**6**
	19790	5316	2905	29440	5611	1609	13165	12277	6198	96301	1	2	**3**
	8744	23883	31733	578	23625	22184	16155	16063	12462	155427	7	1	**8**
	14931	19445	3781	12069	21152	30041	19407	26387	8474	155687	7	6	**3**
	16864	6074	1630	11875	4235	11332	7267	30675	6901	96853	3	4	**7**
	19639	18593	4899	7049	22145	27513	16847	13254	3012	132951	1	7	**8**
	3501	10905	19127	28865	15857	16708	702	6495	8023	110183	3	9	**2**
	14252	30079	31602	14318	10575	22882	29312	29710	4693	187423	3	3	**6**
	28050	7510	90	30210	13411	30157	19899	30454	24778	184559	9	5	**4**
	3415	8314	12117	267	23127	18687	7295	30433	8817	112472	2	7	**9**
	31130	10165	31857	24788	8396	25475	18646	17077	27502	195036	6	0	**6**
	10822	8483	18330	15175	20140	16641	13643	8804	3370	115408	8	0	**8**
SSKx = 638378264968 X envía RCNy, ESSKx y RNKx a Y.													

C.6.5.4 Cálculos en Y para recuperar SKx

Y
> UINy, UCNy <
>>>>>> RCNy, RNKx, ESSKx
MPx = (UINy, UCNy & IDx & IDy){HKM–1}[RCNy]
SKx = (MPx & RNKx){HKMD–1}[ESSKx]

C.6.5.4.1 Cálculos en Y para recuperar MPx mediante RCNy

MPx = (UINy, UCNy & IDx & IDy){HKM–1}[RCNy]

Los valores de prueba y cálculos son idénticos a los indicados en C.6.4.5, salvo que se utiliza el proceso de descriptación HKM–1 para recuperar MPx a partir de RCNy mediante la sustracción de la PRS de 16 cifras, cifra por cifra de la RCNy.

Los resultados de estos cálculos que utilizan valores de prueba son los siguientes:

UINy = 9735576938377831483537091674367228734498197673577598247578649467

UCNy = 7598247578649467

Primitiva = 9735576938377831483537091674367228734498197673577598247578649467

P(0)	9735 + 0 * 101 = 9735	B(0)	2873 + 0 * 79 = 2873
P(1)	5769 + 1 * 101 = 5870	B(1)	4498 + 1 * 79 = 4577
P(2)	3837 + 2 * 101 = 4039	B(2)	1976 + 2 * 79 = 2134
P(3)	7831 + 3 * 101 = 8134	B(3)	7357 + 3 * 79 = 7594
P(4)	4835 + 4 * 101 = 5239	B(4)	7598 + 4 * 79 = 7914
P(5)	3709 + 5 * 101 = 4214	B(5)	2475 + 5 * 79 = 2870
P(6)	1674 + 6 * 101 = 2280	B(6)	7864 + 6 * 79 = 8338
P(7)	36 + 7 * 101 = 743	B(7)	94 + 7 * 79 = 647
P(8)	72 + 8 * 101 = 880	B(8)	67 + 8 * 79 = 699

P(0) a P(3), y B(0) a B(3) se modifican como anteriormente utilizando IDx e IDy

IDx = 642092 IDy = 538249

P(0) = 9735 + 642 = 10377	B(0) = 2873 + 642 = 3515
P(1) = 5870 + 092 = 5962	B(1) = 4577 + 092 = 4669
P(2) = 4039 + 538 = 4577	B(2) = 2134 + 538 = 2672
P(3) = 8134 + 249 = 8383	B(3) = 7594 + 249 = 7843

En el cuadro C.7 se muestran los resultados de utilizar estos valores de primitiva de fase y de base con 9 números primos en el algoritmo HKM–1 para obtener MPx, 43149205748688366, a partir de RCNy, 9865418902725854.

Cuadro C.7/T.36 – Cálculos en Y para recuperar MPx a partir de RCNy

Número primo B(n) P(n)	32603 3515 10377	32507 4669 5962	32183 2672 4577	32003 7843 8383	31847 7914 5239	31607 2870 4214	31583 8338 2280	31547 647 743	31259 699 880	Total	Última cifra	Número encriptado registrado	**Primitiva mutua**
	25001	10586	204	13707	28499	20306	29257	7516	21199	156275	5	9	**4**
	13430	15394	30160	5924	632	26519	29357	4614	1335	127365	5	8	**3**
	29909	1609	1288	25579	1669	31481	10416	19840	26654	148445	5	6	**1**
	18063	3304	30138	21293	23808	17664	26941	28398	782	170391	1	5	**4**
	13404	18058	6870	9345	9660	29659	15762	13152	15215	131125	5	4	**9**
	3725	22151	12330	5965	16440	3679	6693	23201	7225	101409	9	1	**2**
	19572	18252	22551	27112	11165	1992	30656	26222	17576	175098	8	8	**0**
	3250	17741	9696	11484	16232	27780	8509	24895	837	120424	4	9	**5**
	12700	4893	397	12570	21097	15746	12624	18095	22401	120523	3	0	**7**
	6993	25503	30928	17270	19684	24617	24356	3528	28799	181678	8	2	**4**
	30336	366	25855	11914	15499	9145	1638	11232	30964	136949	9	7	**8**
	19230	18490	19842	24745	16289	12340	13788	11294	12608	148626	6	2	**6**
	7431	23725	12423	8843	26337	15960	2224	19861	29213	146017	7	5	**8**
	4962	20676	13583	5148	24250	6657	4491	10438	7760	97965	5	8	**3**
	31428	22961	23535	19981	4478	14962	20103	2328	16433	156209	9	5	**6**
	10456	29330	32121	24295	25028	18634	7833	23507	14614	185818	8	4	**6**
MPx = (UINy, UCNy & IDx & IDy){HKM–1}[RCNy] = 4314920574868366													

C.6.5.4.2 Cálculos en Y para recuperar SKx, mediante ESSKx, empleando HKMD–1

SKx = (MPx & RNKx){HKMD–1}[ESSKx]

La ESSKx se decripta doblemente mediante HKMD–1 en el orden inverso a la doble encriptación efectuada en X. La primera decriptación es la sustracción (mod 10) del "mensaje" (ESSKx) de la misma PRS (mod 10) que la utilizada por X. La segunda decriptación es un proceso de desaleatorización basada en la misma PRS de 12 cifras (mod 10) + 1 como fue utilizada por X.

C.6.5.4.2.1 Cálculo en Y para la primera decriptación de ESSKx, para recuperar SSKx

Los mismos cálculos que emplean HKM de acuerdo con C.6.5.3.2, se utilizan para generar una PRS (mod 10) de 12 cifras, que se sustrae (mod 10) de ESSKx, para producir SSKx.

La MPx, se repite y concatena para formar la primitiva de 64 cifras que se utiliza para calcular los valores de primitivas de fase y de base como se indica a continuación:

Primitiva = 4314920574868366 4314920574868366 4314920574868366 4314920574868366

P(0)	4314 + (0 * 101) = 4314	B(0)	4314 + (0 * 79) = 4314
P(1)	9205 + (1 * 101) = 9306	B(1)	9205 + (1 * 79) = 9284
P(2)	7486 + (2 * 101) = 7688	B(2)	7486 + (2 * 79) = 7644
P(3)	8366 + (3 * 101) = 8669	B(3)	8366 + (3 * 79) = 8603
P(4)	4314 + (4 * 101) = 4718	B(4)	4314 + (4 * 79) = 4630
P(5)	9205 + (5 * 101) = 9710	B(5)	9205 + (5 * 79) = 9600
P(6)	7486 + (6 * 101) = 8092	B(6)	7486 + (6 * 79) = 7960
P(7)	83 + (7 * 101) = 790	B(7)	83 + (7 * 79) = 636
P(8)	66 + (8 * 101) = 874	B(8)	66 + (8 * 79) = 698

RNKx, asociado con ESSKx, se divide en dos pares de dos cifras y el primer par se agrega a P(0) y el segundo par a P(1) para crear nuevos valores de P(0) y P(1); los nuevos valores para B(0) y B(1) se crean de manera idéntica.

RNKx = 3958

P(0) = 4314 + 39 = 4353	B(0) = 4314 + 39 = 4353
P(1) = 9306 + 58 = 9364	B(1) = 9284 + 58 = 9342

Los resultados de los cálculos para decriptar ESSKx, 638378264968, para formar SSKx, 721647935700, figuran en el cuadro C.8.

Cuadro C.8/T.36 – Cálculos en Y para formar SSKx mediante ESSKx

Número primo B(n) P(n)	32603 4353 4353	32507 9342 9364	32183 7644 7688	32003 8603 8669	31847 4630 4718	31607 9600 9710	31583 7960 8092	31547 636 790	31259 698 874	Total	Última cifra	Clave secreta aleatorizada encriptada	**Clave secreta aleatorizada**
	6266	2151	914	12417	29145	6957	14583	29235	16131	117799	9	6	**7**
	19790	5316	2905	29440	5611	1609	13165	12277	6198	96301	1	3	**2**
	8744	23883	31733	578	23625	22184	16155	16063	12462	155427	7	8	**1**
	14931	19445	3781	12069	21152	30041	19407	26387	8474	155687	7	3	**6**
	16864	6074	1630	11875	4235	11332	7267	30675	6901	96853	3	7	**4**
	19639	18593	4899	7049	22145	27513	16847	13254	3012	132951	1	8	**7**
	3501	10905	19127	28865	15857	16708	702	6495	8023	110183	3	2	**9**
	14252	30079	31602	14318	10575	22882	29312	29710	4693	187423	3	6	**3**
	28050	7510	90	30210	13411	30157	19899	30454	24778	184559	9	4	**5**
	3415	8314	12117	267	23127	18687	7295	30433	8817	112472	2	9	**7**
	31130	10165	31857	24788	8396	25475	18646	17077	27502	195036	6	6	**0**
	10822	8483	18330	15175	20140	16641	13643	8804	3370	115408	8	8	**0**
SSKx = 721647935700													

C.6.5.4.2.2 Cálculos en Y para obtener SKx mediante SSKx

Los mismos cálculos que se emplearon en X para producir la PRS (mod 12) + 1 (véase C.6.5.3.1) se utilizan en Y para producir la misma PRS del aleatorizador/desaleatorizador, 8 2 4 12 2 4 12 8 12 9 1 5. Esta PRS del aleatorizador/desaleatorizador se invierte y utiliza como PRS del desaleatorizador para desaleatorizar SSKx.

La PRS del desaleatorizador es 5 1 9 12 8 12 4 2 12 4 2 8.

La primera cifra en la PRS del desaleatorizador indica qué cifra de la SSKx debe intercambiarse con la duodécima cifra de la SSKx. La segunda cifra en la PRS del desaleatorizador indica el intercambio para la undécima cifra de la SSKx y así sucesivamente para las cifras restantes. Los pasos de este proceso de desaleatorización para producir SKx, 309126704577, a partir de SSKx, 721647935700, se indican a continuación:

PRS del desaleatorizador		
5	7216**4**793570**0** =	SSKx
1	**7**2160793570**4**	
9	021607935**7**74	
12	021607937574	
8	02160793**4**577	
12	021607**9**3457**7**	
4	0216**0**7**7**34579	
2	02**1**70**6**734579	
12	0017**7**26734579	
4	00**19**26734577	
2	0**0**9126734577	
8	**0**091267**3**4577	
	309126704577 =	**SKx**

Ambas entidades X e Y están ahora en posesión de la misma SKx.

C.6.6 Utilización del algoritmo HKM en modo seguro

El algoritmo HKM se utiliza en el modo seguro para recrear MPx y MPy en X e Y, empleando procSTKxy y procSTKyx para transferir claves secretas con seguridad a fin de proporcionar autenticación mutua, establecimiento de claves de sesión secretas para la confidencialidad y/o integridad del mensaje, confirmación de recepción y confirmación o denegación de integridad.

Los procedimientos para obtener estas capacidades se describen en C.5.

Anexo D

Procedimientos de utilización del sistema de cifrado HFX40 para proporcionar confidencialidad de mensaje para la transmisión segura de documentos por facsímil

D.1 Alcance

Este anexo define el sistema de cifrado HFX40 que se utilizará con terminales facsímil para proporcionar confidencialidad de mensaje. En D.3 figuran ejemplos de cálculos que utilizan valores de prueba. Estos cálculos se pueden utilizar para verificar la aplicación de este anexo.

El sistema de cifrado HFX40 está diseñado para ser utilizado con todos los tipos de terminales facsímil especializados pero también se aplica en sistemas facsímil con base informatizada.

El sistema HFX40 está basado en la utilización de 19 números primos. Estos mismos 19 números primos se utilizan también con el sistema de gestión de claves HKM que se define en el anexo C y con el sistema de troceo para la integridad del mensaje que se describe en el anexo E. Sin embargo, esta Recomendación no incluye el sistema de gestión de claves, ni el algoritmo de troceo para integridad del mensaje.

En D.3 figuran ejemplos de cálculos que pueden utilizarse para verificar la aplicación de este anexo.

El sistema HFX40 está amparado por los derechos de propiedad intelectual; sin embargo, el poseedor de esos derechos ha acordado seguir la norma de conducta de la TSB. La TSB puede facilitar más detalles.

D.2 Descripción del algoritmo HFX40 para utilización con terminales facsímil en modo seguro

El algoritmo HFX40 se utiliza para proporcionar confidencialidad de mensaje mediante el empleo de encripción. El algoritmo utiliza una clave secreta para proporcionar los números necesarios para efectuar cálculos aritméticos modulares. Los módulos para la aplicación de aritmética modular están proporcionados por una selección de 3 números primos obtenidos a partir de un conjunto de 19 números primos modulantes del sistema, almacenados en el terminal facsímil. Este conjunto de números primos es el mismo que el utilizado por el sistema de gestión de claves HKM (anexo C) y el sistema de integridad de mensaje HFX40-I (anexo E).

El resultado de los procedimientos aritméticos modulares son largas secuencias seudoaleatorias (PRS) que se utilizan para encriptar el mensaje facsímil comprimido. Sólo mediante la posesión de la clave de encriptación secreta es posible recrear el mensaje.

El algoritmo HFX40 utiliza una clave de 12 cifras decimales, equivalente a una longitud de clave de 40 bits aproximadamente.

Los procedimientos de gestión de claves se detallan en el anexo C.

Si se ha de efectuar registro mutuo entre las entidades X e Y, se puede utilizar el modo seguro para establecer la autenticación mutua de X e Y, después de lo cual se puede generar SSx en X y transferir con seguridad a Y. Si no se ha efectuado registro, se puede utilizar el modo anulación (véase la Recomendación T.30) para permitir a los usuarios en X e Y ingresar manualmente una clave de sesión secreta preacordada para formar SSx.

La entidad X utiliza SSx con el algoritmo HFX40 para generar tres PRS (mod 2) que se almacenan como entradas en tres tablas, referidas como tablas P, Q y R. El número de entradas en cada tabla es diferente. La tabla P se genera con 1021 entradas, la tabla Q con 1019 entradas y la tabla R con 1013 entradas.

Las últimas 4 entradas de cada tabla se utilizan para formar un multiplexor que modifica las entradas de la tabla cuando el mensaje está encriptado y las tablas abreviadas se utilizan para encriptar el mensaje. La tabla P tiene ahora 1017 entradas, la tabla Q 1015 entradas y la tabla R 1009 entradas.

El primer bit del mensaje facsímil comprimido se agrega (mod 2) a la adición (mod 2) de los bits en las primeras entradas de las tablas P, Q y R para formar el primer bit del mensaje encriptado. Una vez que cada bit de mensaje se haya procesado, se modifican las entradas de las tablas P, Q y R correspondientes conforme a las entradas en el multiplexor. La modificación en cada tabla es un intercambio de la entrada de la tabla con una entrada en el multiplexor que es determinada por las entradas en las otras dos tablas.

La entrada en la tabla P se intercambia con la entrada en el multiplexor determinada por las tablas Q y R.

La entrada en la tabla Q se intercambia con la entrada en el multiplexor determinada por las tablas R y P.

La entrada en la tabla R se intercambia con la entrada en el multiplexor determinada por las tablas P y Q.

Adviértase que el orden es significativo.

El segundo bit del mensaje principal se procesa de manera similar con los bits en las segundas entradas en las tablas. El nuevo multiplexor formado por el intercambio de las primeras entradas en las tablas se utiliza entonces para intercambio con las segundas entradas en las tablas.

El procedimiento anterior es seguido por cada uno de los bits de mensaje. Cuando se ha utilizado la entrada 1009 en la tabla R, la primera entrada (modificada) es la próxima que se ha de utilizar. El procedimiento continúa entonces con las nuevas entradas. De manera similar, las tablas P y Q "recomienzan" después de las entradas 1017 y 1015, respectivamente.

El mensaje encriptado se envía entonces a Y. Esta entidad utiliza el mismo SSx con el algoritmo HFX40 para generar tablas idénticas y modificar el multiplexor, sustrayendo la adición (mod 2) de las entradas en las tablas del mensaje encriptado para recuperar el mensaje facsímil comprimido original. El SSx utilizado por Y se transfiere con seguridad desde X en el modo seguro, o bien se forma a partir de una clave de sesión preacordada ingresada manualmente por el usuario en Y en el modo anulación.

D.3 Ejemplos de cálculos para el algoritmo HFX40

D.3.1 Introducción

Esta subcláusula describe el algoritmo HFX40 en términos de los números que han de almacenarse y las reglas que emplean estos números para generar las PRS utilizan para encriptar el mensaje facsímil comprimido. Los cálculos pueden utilizarse para verificar la implementación.

D.3.2 Información almacenada

Todos los terminales están equipados con los mismos 19 números primos de modulación del sistema.

32603 32507 32183 32003 31847 31607 31583 31547 31259
31139 30803 30539 30467 30347 30323 30203 29879 29759 29663

D.3.3 Selección de los números primos

Para cada llamada facsímil se utiliza una SSx diferente para seleccionar 3 números primos de los 19 números primos almacenados modulantes del sistema. SSx se divide en cuatro grupos de 3 cifras. Los primeros 2 grupos de tres cifras, g1 y g2, forman un tercer grupo g3 mediante la aplicación de una operación XOR (O exclusivo) lógica. Aplicando una XOR lógica a los segundos 2 grupos de 3 cifras, g4 y g5, se forma un sexto grupo, g6. Se agrega 1024 a los grupos g1 a g3 para formar P(0) a P(2) y a los grupos g4 a g6 para formar B(0) a B(2). Este procedimiento se muestra a continuación en el que se usa un ejemplo numérico.

SSx para este ejemplo es 149162536496.

g1 = 149	P(0) = 149 + 1024 = 1173
g2 = 162	P(1) = 162 + 1024 = 1186
g3 = g1 XOR g2 = 55	P(2) = 55 + 1024 = 1079
g4 = 536	B(0) = 536 + 1024 = 1560
g5 = 496	B(1) = 496 + 1024 = 1520
g6 = g4 XOR g5 = 1000	B(2) = 1000 + 1024 = 2024

Se forman entonces tres números de 8 cifras mediante la concatenación de los pares P(0) y B(0), P(1) y B(1), y P(2) y B(2) entre sí y la aplicación a cada número de aritmética modular base 19.

P(0) y B(0)	11731560 (modulo 19) = 10
P(1) y B(1)	11861520 (modulo 19) = 10
P(2) y B(2)	10792024 (modulo 19) = 5

En la lista de los 19 números primos modulantes del sistema que figuran en D.3.2, el primero de ellos, 32603, se designa número primo (0) y el último, 29663, número primo (18).

Empleando el primer valor, 10, obtenido de P(0) y B(0), el primer número primo, es decir número primo (0), 32603, se intercambia con el número primo (10), 30803.

Utilizando el segundo valor, 10, obtenido de P(1) y B(1), el segundo número primo, es decir número primo (1), 32507, se intercambia con el número primo (10), 32603.

Utilizando el tercer valor, 5, obtenido de P(2) y B(2), se cambia el tercer número primo, es decir número primo (2), 32183, con el número primo (5), 31607.

Este proceso se muestra a continuación:

Números primos modulantes del sistema no modificados	32603	32507	32183	32003	31847	31607	31583	31547	31259		
		31139	30803	30539	30467	30347	30323	30203	29879	29759	29663
10		**30803**	32507	32183	32003	31847	31607	31583	31547	31259	
		31139	**32603**	30539	30467	30347	30323	30203	29879	29759	29663
10		30803	**32603**	32183	32003	31847	31607	31583	31547	31259	
		31139	**32507**	30539	30467	30347	30323	30203	29879	29759	29663
5		30803	32603	**31607**	32003	31847	**32183**	31583	31547	31259	
		31139	32507	30539	30467	30347	30323	30203	29879	29759	29663

D.3.4 Cálculos que emplean el algoritmo HFX40 para generar 3 PRS

Los primeros tres números primos en la disposición final, 30803, 32603 y 31607, se utilizan como módulos para los cálculos efectuados con los 3 valores de primitivas de fase y 3 valores de primitivas de base obtenidos de SSx en D.3.3 para generar tres PRS (mod 2) que serán almacenados en las tablas P, Q y R. La tabla P tendrá 1021 entradas, la tabla Q, 1019 y la tabla R, 1013.

Los ejemplos de cálculos para el primer "conjunto", es decir número primo (0) = 30803, P de fase (0) = 1173 y B de base (0) = 1560, son los siguientes:

P(0) se multiplica por B(0):
1173 * 1560 = 1829880
1829880 [mod número primo (0)] = 1829880 (mod 30803) = 12503
12503 (mod 2) = 1

Se utiliza entonces 12503 como nuevo valor de fase y se multiplica por B(0):

12503 * 1560 = 19504680
19504680 (mod número primo (0)) = 19504680 (mod 30803) = 6381
6381 (mod 2) = 1

El proceso se repite para generar 1021 valores para llenar las entradas en la tabla P.

Se efectúan cálculos similares con el segundo y tercer "conjuntos" de fases, bases y números primos para generar entradas para las tablas Q y R. En el cuadro D.1 se muestran otros 16 conjuntos de cálculos utilizando el algoritmo HFX40.

Cuadro D.1/T.36 – Cálculos que emplean el algoritmo HFX40 para generar 3 PRS

Fase	B(0)	Nueva fase (mod 30803)	Tabla P (mod 2)	Fase	B(0)	Nueva fase (mod 32603)	Tabla Q (mod 2)	Fase	*B(0)	Nueva fase (mod 31607)	Tabla R (mod 2)
1173	1560	12503	1	1186	1520	9555	1	1079	2024	3013	1
12503	1560	6381	1	9555	1520	15265	1	3013	2024	29768	0
6381	1560	4991	1	15265	1520	22067	1	29768	2024	7490	0
4991	1560	23604	0	22067	1520	25956	0	7490	2024	20007	1
23604	1560	12655	1	25956	1520	3490	0	20007	2024	5601	1
12655	1560	27780	0	3490	1520	23114	0	5601	2024	21118	0
27880	1560	29767	1	23114	1520	19849	1	21118	2024	10168	0
29767	1560	16399	1	19849	1520	12705	1	10168	2024	3875	1
16399	1560	15950	0	12705	1520	10624	0	3875	2024	4464	0
15950	1560	23979	1	10624	1520	9995	1	4464	2024	27141	1
23979	1560	12398	0	9995	1520	32005	1	27141	2024	418	0
12398	1560	27399	1	32005	1520	3924	0	418	2024	24250	0
27399	1560	18679	1	3924	1520	30734	0	24250	2024	27936	0
18679	1560	30405	1	30734	1520	28184	0	27936	2024	29148	0
30405	1560	25983	1	28184	1520	31941	1	29148	2024	16890	0
25983	1560	27535	1	31941	1520	4453	1	16890	2024	18193	1
:	:	:	:	:	:	:	:	:	:	:	
:	:	:	:	:	:	:	:	:	:	:	

El cuadro D.2 muestra las tablas P, Q y R completas.

Las últimas 4 entradas de cada tabla se utilizan para formar las entradas iniciales en un multiplexor que se emplea con una "tabla de verdad" para modificar las entradas de las tablas P, Q y R cuando el mensaje es encriptado.

Multiplexor			**"Tabla de verdad"**	
P	**Q**	**R**		
1	1	1	0	0
1	1	1	0	1
1	1	1	1	0
0	1	1	1	1

Las tablas abreviadas – la tabla P tiene ahora 1017 entradas, la tabla Q, 1015 entradas y la tabla R, 1009 entradas – se utilizan para encriptar el mensaje como se explica a continuación.

D.3.5 Utilización de las tablas para encriptar el mensaje y del multiplexor para modificar las tablas

Se agrega el primer bit del mensaje principal (mod 2) a la adición (mod 2) de las primeras entradas en cada una de las tablas abreviadas, P, Q y R para formar el primer bit del mensaje encriptado. El segundo bit del mensaje principal se procesa del mismo modo con las segundas entradas en las tablas y así sucesivamente. Después de que cada bit de mensaje se haya procesado, se modifican las entradas de las tablas P, Q y R correspondientes basadas en las entradas del multiplexor. La modificación en cada tabla es un intercambio de la entrada de la tabla con una entrada en el multiplexor que se determina por las entradas en las otras dos tablas.

Cuadro D.2/T.36 – Tablas P, Q y R completas

<u>Tabla P (1021 entradas)</u> 1110101101011111000101000111001001011011000101001111110000000000100110111000011011000011101111 1101101100000111100001111011010111010110100100010011011010010110001000000101000010010111010101 1100001001010010011000010011100010010111100010111100110000000110101111111111001111010001111111 1000110110010111110101001001101101111100001111010111011011110100011000001011010100011010011111 0001011111010011111011000110011111010001010100100010100110011001100100011001100110011110110000 1011000111001110010001011110111011010000000100001010101010111011010010001000000011010011110100 0110000000000000100011001101110111010010000001101011100000000111010000100111001001011000111011 11 1010101001111111100111111001000011000100011100000101111100011011011011100100111100011101111100 0010111001100100101110111001010111001001101011000100101101000011000110000000000110001010111010 1111011101000111000000001010000101010000000011100111111111110111011000101110001101010100100011 0 10011110101010000000010101110000111100101010100001001011111111100000000110111 1110
<u>Tabla Q (1019 entradas)</u> 1110001101100011101000001100111011000011100111111111000001010111100001111111000101111110000111 1 1101011111100101101111111111001110110110100111100010101100010100110110110000011010010011010101 0 0111100001001111101011110010010001111101101011110100110111000100111011000000111011001101110001 1011101000101010110100110010011100001011110010000111110110110001000010011010111001000010110100 0000000000001010011110010001110010111110010000101110010111100110011111100111101101010000010101 0111110111011001110011011110111101001100100011000110101000101011001000101000110111100000000001 0111100101101011010110001110111100111101100010110011001100011101101010101001011110100011000010 0101101000111010010111100100011111110001110000111111100101011111100111000111100011100101110000 1001000010110011111111100001000101000101101100010100010101000100010011100111010010011001010000 0010011100101011001101001101010100101001110110100000010000100101000100101110111011011011000010 0001110001010111011110110011101100100111011101100001000111011000111010111 00 1111
<u>Tabla R (1013 entradas)</u> 1001100101000001101111100011010101110101000110100101001100110001101100011010111000001100001100 0111000111101000010100100110100111100000000101001100011011010011000001001001000010111101001101 1101001010110100111110101111011100000000110101111111101111100010111000000110011010100001011010 1110000101110101100101101011101010100111101001110110111000001111101111010101000111100000111111 1010001101101100111011011111111000100000011101101001010001010011100101100110001110000000101111 0110010010100000100111010000100000110010011001011101011011001100100011110100000000100001111000 1 0011101110101011000110001110110001100101100100110100010001001100010111001111011111001111011101 1100101110001101000001011100110010111000000000000100110101011011100111110111001001100001111101 1 1001111000111000110111001010101000100101101100011000111001101010000100001111000000110010110101 0 0000001001100000000111011111101110001111011111110001110101000101001001100011110110100110100110 10 1001001101101101100010101101011110000011110000110101011100010001101 00 1111

Después que se haya utilizado un conjunto de entradas en las tablas P, Q y R para encriptar un bit del mensaje:

– la entrada de la tabla P se intercambia con la entrada del multiplexor determinada por las entradas en las tablas Q y R;

– la entrada de la tabla Q se intercambia con la entrada del multiplexor determinada por las entradas en las tablas R y P;

– la entrada de la tabla R se intercambia con la entrada del multiplexor determinada por las entradas en las tablas P y Q;

Adviértase que el orden es significativo.

El siguiente ejemplo en el que se utilizan las primeras 7 entradas de las tablas P, Q y R muestra en detalle cómo se lleva a cabo el procedimiento.

Las primeras 7 entradas en las tablas P, Q y R iniciales son:

P	Q	R
1	1	1
1	1	0
1	1	0
0	0	1
1	0	1
0	0	0
1	1	0

Las entradas iniciales del multiplexor y la tabla de verdad son:

P	Q	R	Tabla de verdad	
1	1	1	0	0
1	1	1	0	1
1	1	1	1	0
0	1	1	1	1

Las primeras entradas en las tablas P, Q y R son 1, 1 y 1, respectivamente. La primera entrada en la tabla P se intercambia con las primeras entradas y las tablas Q y R, éstas son 1 y 1. De la "tabla de verdad", la primera entrada en la tabla P se intercambia con la entrada en la columna P del multiplexor que corresponde a (1, 1) en la "tabla de verdad", en este caso 0.

La primera entrada en Q se intercambia con la entrada en la columna Q del multiplexor que corresponde a (1, 1), en este caso 1.

De manera similar la primera entrada en R se intercambia con la entrada en la columna del multiplexor que corresponde a (1, 1), en este caso 1.

El resultado de esto es que las primeras entradas en las tablas P, Q y R, es decir 1, 1, 1, se transforma en 0, 1, 1, respectivamente. Estas entradas se utilizarán la próxima vez que se emplee la primera entrada en una tabla.

Las entradas del multiplexor después de estos intercambios son:

P	Q	R
1	1	1
1	1	1
1	1	1
1	1	1

Las nuevas entradas del multiplexor se utilizan ahora con las segundas entradas de las tablas P, Q y R exactamente de la misma manera. El resultado de esto es que las segundas entradas en las tablas P, Q y R, 1, 1, 0, se transforma en 1, 1, 1 y el multiplexor resulta:

P	Q	R
1	1	1
1	1	1
1	1	1
1	1	0

Conforme al procedimiento anterior para los primeros 5 bits de mensaje que utilizan las primeras 5 entradas de las tablas P, Q y R se obtienen los siguientes resultados:

	Tablas iniciales			Tablas después de la encripción de 5 bits de mensaje		
Entrada	P	Q	R	P	Q	R
1	1	1	1	0	1	1
2	1	1	0	1	1	1
3	1	1	0	1	1	0
4	0	0	1	1	1	1
5	1	0	1	0	1	1
6	0	0	0	0	0	0
7	1	1	0	1	1	0
:	:	:	:	:	:	:
:	:	:	:	:	:	:

Los valores del multiplexor utilizados para modificar las tablas luego de encriptar cada uno de los primeros 5 bits de mensaje son:

Después del bit número: (posición inicial)			1			2			3			4			5		
P	Q	R	P	Q	R	P	Q	R	P	Q	R	P	Q	R	P	Q	R
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	0	1
0	1	1	1	1	1	1	1	0	1	1	0	1	1	0	1	0	0

El procedimiento anterior se sigue para cada uno de los bits de mensaje. Luego que la tabla R utiliza la entrada número 1009, retorna a la entrada número 1 (al tope de la tabla) y continúa el procedimiento con las nuevas entradas de la tabla. De manera similar las tablas Q y P, después de los bits números 1015 y 1017, respectivamente, se "reinician" en sus nuevas primeras entradas. El procedimiento continúa hasta que se haya encriptado el mensaje completo.

El mensaje encriptado se envía entonces a Y. Esta entidad utiliza la misma SSx con el algoritmo HFX40 para generar tablas idénticas y el multiplexor de modificación, sustrayendo la adición (mod 2) de las entradas de las tablas del mensaje encriptado para recuperar el mensaje facsímil comprimido original y modificar las tablas utilizando el multiplexor.

Anexo E

Procedimientos de utilización del sistema de troceo HFX40-I para proporcionar integridad de mensaje para la transmisión segura de documentos por facsímil

E.1 Alcance

Esta Recomendación describe el algoritmo de troceo HFX40-I, en términos de su uso, los cálculos necesarios y de información que se han de intercambiar entre los terminales facsímil para proporcionar la integridad de un mensaje facsímil transmitido como una alternativa seleccionada o preprogramada a la encriptación del mensaje.

Para proporcionar integridad de mensaje, se utiliza una función troceo para hacer corresponder arbitrariamente largos mensajes con valores de longitud fija. La función troceo generada por el algoritmo HFX40-I transforma el mensaje facsímil comprimido en una secuencia de cifras decimales.

Una función troceo es criptográficamente fuerte si es difícil hallar cualquier mensaje que se correlacione con un determinado valor de troceo, o cualquier par de mensajes que se correlacionen con el mismo valor de troceo. Para protegerse de la posibilidad de que esto sea efectuado por un tercero, el algoritmo HFX40-I utiliza primitivas obtenidas de una clave secreta. El valor de troceo resultante del mensaje es también doblemente encriptado para evitar la posibilidad que un tercero revierta la función troceo para descubrir las primitivas originales.

El algoritmo de troceo HFX40-I se basa en la utilización de 19 números primos del sistema. Estos mismos 19 números primos se utilizan también con el sistema de gestión de claves HKM que se define en el anexo C y el sistema de cifrado de mensajes que se describe en el anexo E. Sin embargo, esta Recomendación no se ocupa del sistema de gestión de claves ni del cifrado de mensajes.

El intercambio seguro de claves secretas sigue los procedimientos detallados en el anexo C.

En E.3 y E.4 se incluyen ejemplos de cálculos que se pueden utilizar para verificar la aplicación del algoritmo.

El sistema HFX40-I está amparado por los derechos de propiedad intelectual; sin embargo, el poseedor de esos derechos ha acordado seguir la norma de conducta de la TSB. La TSB puede facilitar más detalles al respecto.

E.2 Utilización del sistema de troceo HFX40-I

Para enviar un mensaje en el que se protege su integridad, el usuario en X establece una comunicación con Y con quien se ha efectuado registro mutuo (véase el anexo C). Luego de una autenticación mutua satisfactoria de X e Y (véase el anexo C), X genera SSx que se envía con seguridad a Y, utilizando el procedimiento para la transferencia segura de una clave secreta (procSTKxy) definida en el anexo C.

La entidad X también utiliza SSx, para formar las primitivas para la función troceo, HFX40-I. Las primitivas son usadas por el algoritmo HFX40-I para los cálculos aritméticos modulares para operar en el mensaje facsímil comprimido byte por byte. Los módulos para la aritmética modular se extraen de un conjunto de 19 números primos modulantes del sistema almacenados en el terminal facsímil. El orden en el cual se utilizan los 19 números primos se deriva de SSx. Este conjunto de números primos es el mismo que el utilizado por el sistema de gestión de claves HKM (véase el anexo C) y el algoritmo del sistema de cifrado HFX40 (véase el anexo D).

La salida de una secuencia de cálculos modulares con un byte del mensaje se utiliza como entrada para los cálculos para el siguiente byte del mensaje, y así sucesivamente hasta que se haya procesado al mensaje completo. El PH resultante se encripta doblemente mediante la clave de sesión. La primera encriptación es una función de un solo uso que aleatoriza

las 24 cifras de PH para formar SH. La segunda encriptación utiliza una variante de un solo uso del cifrado HKM para formar ESH. La entidad X envía el mensaje facsímil comprimido y ESH a la entidad Y.

La entidad Y recupera SSx mediante el procedimiento procSTKxy y calcula el PH del mensaje facsímil comprimido recibido por medio del algoritmo HFX40-I. La entidad Y realiza las dos mismas encriptaciones de PH que X para formar ESH. Si este valor de ESH concuerda con el valor de ESH recibido de X, no sólo se confirma la integridad del mensaje sino también la autenticación del emisor. Cualquier discrepancia entre el ESH calculado y el ESH recibido es, evidentemente, pérdida de integridad.

Para proporcionar confirmación o denegación de integridad a X, Y genera IMy, que es un número aleatorio de 12 cifras seleccionado a partir de las cifras 2 a 9, en el cual una cifra es seleccionada al azar para ser reemplazada por un 1 para indicar confirmación de integridad o por un 0 para indicar negación. Y envía IMy en forma segura a X utilizando SSx y procSTKyx.

Ejemplos de respuestas:

IMy = 257795**1**99982 Integridad confirmada.
IMy = 3**1**7736845378 Integridad confirmada.
IMy = 73854368**0**892 Integridad denegada.
IMy = 4577452**0**4639 Integridad denegada.

E.3 El sistema de troceo HFX40-I para utilización con terminales facsímil

E.3.1 Introducción

En esta subcláusula se describe el sistema de troceo HFX40-I en términos de los números que serán almacenados y las reglas para los cálculos efectuados utilizando estos números para generar PH, SH y ESH. La regla se explica mejor utilizando ejemplos numéricos. Los ejemplos numéricos se utilizan también como valores de prueba que permiten verificar la implementación.

E.3.2 Información almacenada

Todos los terminales están equipados con los mismos 19 números primos modulantes del sistema (utilizados también en los anexos C y D).

32603 32507 32183 32003 31847 31607 31583 31547 31259
31139 30803 30539 30467 30347 30323 30203 29879 29759 29663

E.3.3 Reordenamiento de los números primos modulantes del sistema

La entidad X utiliza SSx para obtener tres valores de primitivas de fase iniciales, P(0) a P(2) y tres valores de primitivas de base, B(0) a B(2) que serán utilizados con los primeros tres números primos modulantes del sistema, 32603, 32507 y 32183 para generar 19 entradas en una PRS (mod 19).

SSx se divide en seis grupos superpuestos de tres cifras. El primer grupo se forma con las tres primeras cifras, el segundo con la segunda, tercera y cuarta cifras, y así sucesivamente. Al primer grupo de tres cifras se le aplica la operación lógica XOR (O exclusivo) con el segundo grupo, y a su resultado se le vuelve a aplicar la operación lógica XOR (O exclusivo) con el tercer grupo, y así sucesivamente. Se agregan dos unidades a cada grupo resultante para evitar un resultado de todos ceros. Los valores resultantes se utilizan entonces como valores iniciales de primitivas de fase y de base, P(0) a P(2) y B(0) a B(2). El cuadro E.1 muestra un ejemplo para SSx = 568702123345.

Cuadro E.1/T.36 – Generación de P(0) a P(2) y B(0) a B(2) para reordenar los números primos modulantes del sistema

SSx = 568702123345
Los seis grupos son: 568, 870, 021, 123, 334 y 345

Grupo			**Resultado**	+2	**Valor de fase/base**
568			568	570	P(0)
870	XOR	568	350	352	P(1)
021	XOR	350	331	333	P(2)
123	XOR	331	304	306	B(0)
334	XOR	304	126	128	B(1)
345	XOR	126	295	297	B(2)

Los valores de las primitivas de fase y de base se utilizan con los primeros tres números primos modulantes del sistema, 32603, 32507 y 32183, en el algoritmo HKM para generar los primeros 19 valores en una PRS (mod 19). En E.4 figura un ejemplo de la utilización de HKM con estos valores de pruebas.

La PRS (mod 19) generada es:

10, 14, 0, 12, 2, 14, 9, 6, 10, 1, 2, 9, 17, 6, 14, 5, 3, 15, 5

Los 19 números primos modulantes del sistema se transponen utilizando la PRS para determinar el orden en el cual se deben utilizar en el algoritmo de troceo HFX40-I. El primer valor en la PRS determina qué número primo "numerado" se intercambia con el número primo en la primera posición, el segundo valor, qué número primo "numerado" se intercambia con el número primo en la segunda posición, y así sucesivamente. En el cuadro E.2 se muestran las 19 fases de transposición frente a la PRS.

Cuadro E.2/T.36 – Transposición de los números primos modulantes del sistema, "números" 0 a 18, utilizando la PRS (mod 19)

Paso	PRS	…"Número" primo (0 a 18)																		
		0	1	2	3	4	5	6	7	8	9	**10**	11	12	13	14	15	16	17	18
1	10	**10**	1	2	3	4	5	6	7	8	9	**0**	11	12	13	14	15	16	17	18
2	14	10	**14**	2	3	4	5	6	7	8	9	0	11	12	13	**1**	15	16	17	18
3	0	**2**	14	**10**	3	4	5	6	7	8	9	0	11	12	13	1	15	16	17	18
4	12	2	14	10	**12**	4	5	6	7	8	9	0	11	**3**	13	1	15	16	17	18
5	2	2	14	**4**	12	**10**	5	6	7	8	9	0	11	3	13	1	15	16	17	18
6	14	2	14	4	12	10	**1**	6	7	8	9	0	11	3	13	**5**	15	16	17	18
7	9	2	14	4	12	10	1	**9**	7	8	**6**	0	11	3	13	5	15	16	17	18
8	6	2	14	4	12	10	1	**7**	**9**	8	6	0	11	3	13	5	15	16	17	18
9	10	2	14	4	12	10	1	7	9	**0**	6	**8**	11	3	13	5	15	16	17	18
10	1	2	**6**	4	12	10	1	7	9	0	**14**	8	11	3	13	5	15	16	17	18
11	2	2	6	**8**	12	10	1	7	9	0	14	**4**	11	3	13	5	15	16	17	18
12	9	2	6	8	12	10	1	7	9	0	**11**	4	**14**	3	13	5	15	16	17	18
13	17	2	6	8	12	10	1	7	9	0	11	4	14	**17**	13	5	15	16	**3**	18
14	6	2	6	8	12	10	1	**13**	9	0	11	4	14	17	**7**	5	15	16	3	18
15	14	2	6	8	12	10	1	13	9	0	11	4	14	17	7	**5**	15	16	3	18
16	5	2	6	8	12	10	**15**	13	9	0	11	4	14	17	7	5	**1**	16	3	18
17	3	2	6	8	**16**	10	15	13	9	0	11	4	14	17	7	5	1	**12**	3	18
18	15	2	6	8	16	10	15	13	9	0	11	4	14	17	7	5	**3**	12	**1**	18
19	5	2	6	8	16	10	**18**	13	9	0	11	4	14	17	7	5	3	12	1	**15**

En el proceso anterior, el primer paso intercambia el primer número primo con el número primo "número" 10. El segundo paso intercambia el segundo número primo con el número primo "número" 14. El paso final intercambia el número primo decimonoveno con el número primo "número" 5.

El orden final de los "números" primos modulantes del sistema es:

2, 6, 8, 16, 10, 18, 13, 9, 0, 11, 4, 14, 17, 7, 5, 3, 12, 1, 15

dando el orden final de los números primos modulantes del sistema:

32183, 31583, 31259, 29879, 30803, 29663, 30347, 31139, 32603
30539, 31847, 30323, 29759, 31547, 31607, 32003, 30467, 32507, 30203

Los primeros tres se utilizan con el algoritmo HFX40-I para generar PH y en las dos encripciones SH y ESH.

E.3.4 Cálculo de las primitivas que se utilizarán con el algoritmo HFX40-I

Se obtienen ocho primitivas de fase P(0) a P(7) a partir de SSx, como sigue.

SSx, 568702123345, se divide en ocho grupos superpuestos de 4 cifras. El primer grupo se forma con las primeras cuatro cifras, el segundo con las cifras tercera a sexta, y así sucesivamente. El primer grupo de cuatro cifras se le aplica la operación lógica XOR con el segundo, y a su resultado se le aplica la operación lógica XOR con el tercero, y así sucesivamente. Se agregan dos unidades a cada uno de los valores resultantes para evitar un resultado de todos ceros, y los valores resultantes se utilizan como valores iniciales para P(0) a P(7).

En el cuadro E.3 se muestra un ejemplo con SSx = 568702123345.

Cuadro E.3/T.36 – Cálculo de las primitivas que se utilizarán con el algoritmo HFX40-I

SSx = 568702123345
Los ocho grupos son: 5687, 8702, 7021, 0212, 2123, 1233, 2334 y 3345

Grupo			**Resultado**	**+2**	**Valor de fase/base**
5687				5689	P(0)
8702	XOR	5687	14281	14283	P(1)
7021	XOR	14281	11428	11430	P(2)
212	XOR	11428	11376	11378	P(3)
2123	XOR	11376	9275	9277	P(4)
1233	XOR	9275	8426	8428	P(5)
2334	XOR	8426	10740	10742	P(6)
3345	XOR	10740	9445	9447	P(7)

E.3.5 Cálculo de PH

Los valores de fase P(0) a P(7) se utilizan en rotación comenzando con P(1). Cuando se ha utilizado el valor inicial para P(1) en los cálculos con el primer byte del mensaje comprimido, se reemplaza con un nuevo valor generado por los cálculos detallados en los siguientes puntos. Este nuevo valor de P(1) se emplea la próxima vez que P(1) se utilice en los cálculos. Este mismo procedimiento continúa con P(2), P(3), etc. El procedimiento global, que a continuación se explica con mayor detalle, continúa hasta el final del mensaje.

Sea P(n) el valor de fase que se utiliza. P(n) se modifica para formar P′(n) agregando a P(n) el valor ASCII decimal del byte de mensaje corriente, b, y el valor Q (mod M), q, obtenido del byte anterior. Para el primer byte, q = 0.

$$P'(n) = P(n) + b + q$$

Q se obtiene de P′(n) multiplicado por (b + 1).

$$Q = P'(n) * (b + 1)$$

Q es modulado por uno de los 19 números primos reordenados modulantes del sistema utilizado en rotación, comenzando por el número primo (1) para formar los nuevos valores P(n) y q. Por ejemplo, para el primer byte del mensaje, Q (mod M) forma el valor q para el segundo byte y P(n) para el noveno byte. El orden de los números primos modulantes del sistema es el que se determina en E.3.3. Para el primer byte del mensaje, se utiliza el número primo (1), para el segundo byte el número primo (2), y así sucesivamente.

En el cuadro E.4 se muestran los cálculos para producir la operación troceo de un mensaje facsímil comprimido de 29 bytes.

PH se forma mediante la concatenación de las tres cifras menos significativas de los ocho valores de fase finales tomados en el orden P(0), P(1), P(2), etc. a P(7) para producir un número de 24 cifras. En este ejemplo, del cuadro E.4 se obtiene:

PH = 171 666 427 631 042 698 579 505

E.3.6 Primera encriptación (aleatorización) de PH para formar SH

Se utiliza SSx para obtener tres primitivas de fase, P(0) a P(2) y tres primitivas de base, B(0) a B(2). SSx se divide en seis grupos superpuestos de tres. Las primeras cifras 1-3 forman P(0), las cifras 3-5 forma P(1), las cifras 4-6 forman P(2), las cifras 7-9 forman B(0), las cifras 9-11 forman B(1) y las cifras 10-12 forman B(2).

Al primer grupo de tres cifras se le aplica la operación lógica XOR con el segundo, al resultado de la misma se le aplica la operación lógica XOR con el tercero, y así sucesivamente. Se añaden dos unidades a cada uno de los grupos de tres resultantes para evitar que se produzca un resultado de todos ceros. Los nuevos valores se combinan con los primeros seis grupos de tres cifras de PH para dar los nuevos valores de P(0) a P(2) y B(0) a B(2).

En el cuadro E.5 se muestra un cálculo tipo que utiliza los mismos valores del ejemplo anterior.

Cuadro E.4/T.36 – Cálculos de troceo para un mensaje facsímil comprimido de 29 bytes

b	y	P(n)	q	P′(n)	Q	M	Q(mod M)	Resultados de troceo
103	1	14283	0	14386	1496144	31583	11743	
121	2	11430	11743	23294	2841868	31259	28558	
2	3	11378	28558	39938	119814	29879	298	
0	4	9277	298	9575	9575	30803	9575	
34	5	8428	9575	18037	631295	29663	8372	
79	6	10742	8372	19193	1535440	30347	18090	
92	7	9447	18090	27629	2569497	31139	16099	
92	0	5689	16099	21880	2034840	32603	13454	
33	1	11743	13454	25230	857820	30539	2728	
33	2	28558	2728	31319	1064846	31847	13895	
33	3	298	13895	14226	483684	30323	28839	
10	4	9575	28839	38424	422664	29759	6038	
4	5	8372	6038	14414	72070	31547	8976	
238	6	18090	8976	27304	6525656	31607	14614	
161	7	16099	14614	30874	5001588	32003	9120	
141	0	13454	9120	22715	3225530	30467	26495	
24	1	2728	26495	29247	731175	32507	16021	
2	2	13895	16021	29918	89754	30203	29348	
3	3	28839	29348	58190	232760	32183	7479	
62	4	6038	7479	13579	855477	31583	2736	
149	5	8976	2736	11861	1779150	31259	28646	
66	6	14614	28646	43326	2902842	29879	4579	**579 [valor para P(6)]**
11	7	9120	4579	13710	164520	30803	10505	**505 [valor para P(7)]**
93	0	26495	10505	37093	3486742	29663	16171	**171 [valor para P(0)]**
39	1	16021	16171	32231	1289240	30347	14666	**666 [valor para P(1)]**
133	2	29348	14666	44147	5915698	31139	30427	**427 [valor para P(2)]**
19	3	7479	30427	37925	758500	32603	8631	**631 [valor para P(3)]**
124	4	2736	8631	11491	1436375	30539	1042	**042 [valor para P(4)]**
92	5	28646	1042	29780	2769540	31847	30698	**698 [valor para P(5)]**

Cuadro E.5/T.36 – Cálculo de P(0) a P(2) y B(0) a B(2) para la primera encriptación de PH

Clave de sesión = 568702123345

Valores P y B iniciales			+2			Grupos PH		Nuevos valores de P y B	
568			568	570	+	171	=	741	P(0)
870	XOR 568	=	350	352	+	666	=	1018	P(1)
021	XOR 350	=	331	333	+	427	=	760	P(2)
123	XOR 331	=	304	306	+	631	=	937	B(0)
334	XOR 304	=	126	128	+	042	=	170	B(1)
345	XOR 126	=	295	297	+	698	=	995	B(2)

Los valores de las primitivas de fase y de base, P(0) a P(2) y B(0) a B(2) precedentes, con el primero de los tres números primos modulantes del sistema, 32183, 31583 y 31259 procedentes de la secuencia reordenada obtenida en E.3.3, se utilizan en el algoritmo HKM de modo similar al indicado en E.4, para formar la PRS de 24 entradas (mod 24) siguiente:

4, 5, 4, 16, 2, 6, 16, 12, 24, 21, 6, 5, 11, 4, 8, 21, 22, 5, 24, 9, 3, 16, 19, 8

Esta PRS se utiliza para transponer las 24 cifras de PH para formar SH. El primer valor en la PRS determina la posición de la cifra en PH que se intercambia con la primera cifra de PH, el segundo valor determina la posición de la cifra que se intercambia con la segunda cifra, y así sucesivamente. Para la PRS anterior, el primer paso intercambia la posición 1 con la posición 4, el segundo paso intercambia la posición 2 con la 5, y así sucesivamente. Este proceso continúa hasta el paso final en que la posición 24 se intercambia con la posición 8. En el cuadro E.6 se muestran los 24 pasos de transposiciones frente a la PRS.

Cuadro E.6/T.36 – Transposición de PH para crear SH

Paso/posición	PRS	PH	171	666	427	631	042	698	579	505
1	4		**6**71	**1**66	427	631	042	698	579	505
2	5		66**1**	17**6**	427	631	042	698	579	505
3	4		66**1**	**1**76	427	631	042	698	579	505
4	16		661	**6**76	427	631	042	**1**98	579	505
5	2		6**7**1	6**6**6	427	631	042	198	579	505
6	6		671	66**6**	427	631	042	198	579	505
7	16		671	666	**1**27	631	042	**4**98	579	505
8	12		671	666	1**1**7	63**2**	042	498	579	505
9	24		671	666	11**5**	632	042	498	579	50**7**
10	21		671	666	115	**9**32	042	498	57**6**	507
11	6		671	66**3**	115	9**6**2	042	498	576	507
12	5		671	6**2**3	115	96**6**	042	498	576	507
13	11		671	623	115	9**0**6	**6**42	498	576	507
14	4		671	**4**23	115	906	6**6**2	498	576	507
15	8		671	423	1**2**5	906	66**1**	498	576	507
16	21		671	423	125	906	661	**6**98	57**4**	507
17	22		671	423	125	906	661	6**5**8	574	**9**07
18	5		671	4**8**3	125	906	661	65**2**	574	907
19	24		671	483	125	906	661	652	**7**74	90**5**
20	9		671	483	12**7**	906	661	652	7**5**4	905
21	3		67**4**	483	127	906	661	652	751	905
22	16		674	483	127	906	661	**9**52	751	**6**05
23	19		674	483	127	906	661	952	**0**51	67**5**
24	8		674	483	1**5**7	906	661	952	051	67**2**
		SH = 674 483 157 906 661 952 051 672								

E.3.7 Encriptación de SH para formar ESH

De la SSx se obtienen tres primitivas de fase, P(0) a P(2), y tres primitivas de base B(0) a B(2) en un proceso similar al indicado en E.3.6, salvo que los valores resultantes de la operación lógica XOR y la adición de 2 cálculos se combinan con los primeros seis grupos de tres cifras en SH para dar los nuevos valores que se utilizan como valores iniciales de P(0) a P(2) y B(0) a B(2).

En el cuadro E.7 se muestran los resultados de los cálculos empleando el valor SH obtenido en E.3.6.

Cuadro E.7/T.36 – Cálculo de las primitivas para la encriptación de EH para formar ESH

SH = 674 483 157 906 661 952 051 672

Valores P y B iniciales				**+2**			**Grupos PH**		**Nuevos valores de P y B**	
568				568	570	+	674	=	1244	P(0)
870	XOR	568	=	350	352	+	483	=	835	P(1)
021	XOR	350	=	331	333	+	157	=	490	P(2)
123	XOR	331	=	304	306	+	906	=	1212	B(0)
334	XOR	304	=	126	128	+	661	=	789	B(1)
345	XOR	126	=	295	297	+	952	=	1249	B(2)

Los valores de la primitiva de fase y de base, P(0) a P(2) y B(0) a B(2) precedentes, junto con los primeros tres números primos modulantes del sistema, 32183, 31583 y 31259, procedentes de las secuencias reordenada, obtenida en E.3.3, se utilizan en el algoritmo HKM de manera similar al ejemplo en E.4, para generar 24 entradas en una PRS (mod 10) y se agregan (mod 10) a SH para formar ESH.

SH: 674 483 157 906 661 952 051 672

(mod 10) PRS: 402 025 183 343 270 975 304 836

ESH: 076 408 230 249 831 827 355 408

E.4 Utilización del algoritmo HKM para producir una secuencia seudoaleatoria

E.4.1 Introducción

Esta subcláusula ofrece un ejemplo que utiliza valores de prueba para verificar la aplicación de la operación del algoritmo HKM que utiliza aritmética modular para generar una secuencia seudoaleatoria, PRS.

La aritmética modular utiliza valores de primitivas de fase y de base obtenidos de diversas maneras mediante un número secreto, tal como una clave de sesión, o una combinación de un número secreto y de otra información de identificación, junto con módulos extraídos de un conjunto universal de números primos modulantes del sistema para generar una PRS, a la que se le puede aplicar aún una operación modular para formar una PRS específica.

E.4.1.1 Cálculos que utilizan HKM para generar una PRS

Los valores de prueba empleados en este ejemplo son los utilizados en el E.3.3 para formar la PRS de 19 entradas (mod 19) para reordenar los números primos modulantes del sistema.

Las 3 primitivas de fase, P(0) a P(2) y las 3 primitivas de base B(0) a B(2), son:

P(0) = 570	B(0) = 306
P(1) = 352	B(1) = 128
P(2) = 333	B(2) = 297

Los 3 números primos modulantes del sistema son: 32603, 32507 y 32183

Utilizando el primer "conjunto" de valores de fase, de base y número primo P(0), B(0) y 32603:

P(0) se multiplica por B(0)

570 * 306 = 174420

174420 (mod el primer número primo) =174420 (mod 32603) =11405

Se utiliza entonces 11405 como el nuevo valor de fase y se lo multiplica por el valor de base, B(0)

11405 * 306 = 3489930

3489930 (mod el primer número primo) = 3489930 (mod 32603) = 1409

Este proceso se lleva a cabo un total de 19 veces (que corresponde a la cantidad de números primos que se han de reordenar). El proceso completo también se repite para los dos "conjuntos" restantes de valores de fase, base y número primo para generar secuencias similares.

Los resultados del primer cálculo de cada uno de los tres "conjuntos" de fase, base y número primo se adicionan para formar el primer valor en la columna total. Los resultados de cada uno de los otros cálculos se adicionan para formar las otras entradas en la columna total. A la secuencia resultante en este ejemplo se le aplica el módulo 19 para generar una PRS (mod 19). Se pueden utilizar otros números adecuados para modular la secuencia para aplicar la operación modular a la secuencia a fin de generar otras PRS específicas.

El conjunto completo de cálculos se muestra en el cuadro E.8.

Cuadro E.8/T.36 – Cálculos que utilizan HKM para generar una PRS de 19 entradas (mod 19)

Número primo modulante B(n) P(n)	32603 306 570	32507 128 352	32183 297 333	Total	**(mod 19) PRS**
	11405	12549	2352	26306	**10**
	1409	13429	22701	37539	**14**
	7315	28548	15950	51813	**0**
	21386	13360	6249	40995	**12**
	23516	19716	21522	64754	**2**
	23236	20609	19800	63645	**14**
	2762	4885	23294	30941	**9**
	30097	7647	31156	68900	**6**
	15636	3606	16811	36053	**10**
	24578	6470	4502	35550	**1**
	22178	15485	17591	55254	**2**
	5044	31660	10881	47585	**9**
	11123	21612	13357	46092	**17**
	12926	3241	8520	24687	**6**
	10393	24764	20166	55323	**14**
	17767	16613	3264	37644	**5**
	24604	13509	3918	42031	**3**
	30134	6281	5058	41473	**15**
	26958	23800	21808	72566	**5**

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie B	Medios de expresión: definiciones, símbolos, clasificación
Serie C	Estadísticas generales de telecomunicaciones
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	Transmisiones de señales radiofónicas, de televisión y de otras senales multimedios
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	Mantenimiento: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	**Terminales para servicios de telemática**
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	Redes de datos y comunicación entre sistemas abiertos
Serie Z	Lenguajes de programación