CAPÍTULO 3
DESARROLLO E IMPLEMENTACIÓN

3.1 SISTEMA OPERATIVO

Antes de abordar el tema del funcionamiento e implementación de DNSSEC es necesario saber, en donde se implementa, si funciona sobre algún sistema operativo en específico, etc. Antes de seguir, definiré ¿Qué es un sistema operativo?

Un sistema operativo es el conjunto de programas que controlan y gestionan de una manera más eficiente, las acciones y recurso de un dispositivo u computadora. Proporcionando la base sobre la cual se pueden escribir programas de aplicación, estableciendo comunicación entre el usuario y la máquina.

Es por ello que la implementación y desarrollo han sido de varios años y aun en nuestros días no se llega a su total culminación.

DNSSEC son las medidas de seguridad que son implementadas en los servidores de nombre de dominio (DNS).

Pero recordemos que un Servidor de Nombre de Dominio, es aquel que administra la información de una parte del nombre de dominio llamada zona, dando origen a una base de datos, esta es distribuida, jerárquica y descentralizada, formando una parte fundamental para el funcionamiento en la internet.

Es por esta razón que tiene un crecimiento, mediante los diferentes niveles, propuesto por los creadores Jonathan B. Postell y Paul Mockapetris, especificado en los RFC's 882 y 883.

Para llevar a cabo el servicio del Servidor de Nombre de Dominio en 1980 Paul Vixie conocido en programas de sistemas UNIX diseña la arquitectura de un Software de licencia libre llamado **Berkeley Internet Name Domain (BIND)**, el más utilizado por el protocolo DNS, es conveniente mencionar que BIND se

puede implementar en cualquier sistema operativo, aunque para aquellos administradores que trabajan en plataformas Windows, Microsoft desarrollo sus propios DNS Server Windows.

Retomando lo anterior Paul Vixie trabajo y modifico el BIND hasta la versión 8 resolviendo las diferentes vulnerabilidades de seguridad con respecto a la primera versión liberada que fue la versión 4.3BSD, posteriormente deja el mantenimiento y distribución a la ISC (Internet Software Consortium) fundada en 1994, la cual es una organización que se encarga de la distribución y elaboración de software de código abierto, para el apoyo de la infraestructura de conectividad a la Internet.

"BIND proporciona una plataforma sólida y estable sobre la cual se pueden crear sistemas de computación distribuida con el conocimiento de que esos sistemas son totalmente compatibles con las normas publicadas DNS." [6]

Actualmente BIND se encuentra en la versión 9, esta contiene aspectos de seguridad que anteriormente no se contemplaban como son: DNS Security Extensions, IPV6, algunas mejoraras en los protocolos IXFR, DDNS etc.
Teniendo en cuenta que existe una variedad de sistemas operativos se debe de elegir uno que soporte el BIND para un mejor funcionamiento, para ello los estudios realizados comentan que depende de las necesidades y recursos con los que se cuenten influirá en la decisión.

Para la implementación de dicha investigación y las pruebas se decidió utilizar un sistema operativo tipo UNIX, llamado OpenBSD versión 4.6.

[6] Tomado y traducido de http://www.isc.org/software/bind

3.2 UNIX Y SUS DERIVADOS

UNIX es un Sistema operativo que controla los recursos de un equipo o dispositivo compartido, entre varios usuarios (multiusuario), es decir este acepta el acceso de más de un usuario al mismo tiempo, dando como principal objetivo el proporcionar una asignación ordenada y controlada de su uso del o los procesadores, la memoria y los dispositivos tanto de entrada como salida para los distintos usuarios y programas.

Estos sistemas operativos tienen la característica principal de ser multitarea, ya sea cooperativa (no existe prioridad en los procesos, el proceso decide cuando deja de utilizar los recursos) o de asignación de prioridad (el sistema puede intervenir en cualquier momento en la asignación de prioridades de los procesos), es decir, ejecutar más de un programa o proceso al mismo tiempo.

UNIX contiene un núcleo del sistema llamado kernel y diversos programas esenciales como son: los compiladores, editores, lenguajes de comandos, programas para copiado e impresión de archivos y programas desarrollados por el propio usuario formando así un sistema operativo sencillo, elegante y consistente.

UNIX fue creado por los Laboratorios Bell de AT & T y el MIT (Massachussets Institute of Technology) a finales de los 60’s, bajo el nombre de MULTICS. En la década de de los 70 fue retomado por Dennis Ritchie, quien desarrollando un compilador para el lenguaje C (Lenguaje en el que fue desarrollado UNIX), dado este suceso surgió lo que hoy es UNIX.

Gracias a su portabilidad, su ambiente y programación en un lenguaje de alto nivel, el código fuente fue difundido y utilizado por las universidades de Estados Unidos.

Una de las versiones más significativas y difundida fue la Universidad de California en Berkeley, la cual fue la base de diversas versiones.

A partir de la década de los 80 AT & T, comienza a comercializar UNIX.
Una de las funciones que se ejecutaba era
"controla el hardware y proporciona una interfaz de llamada al sistema para todos los programas" [7]

Estas interfaces permiten a los programas desarrollados por el usuario, participar en la creación y manejo de procesos, archivos y otros recursos.

Los programas realizan llamadas al sistema al colocar valores en los registros (a veces utilizan pilas), y en el momento de ejecutar las instrucciones de señalamientos, para alternar entre el modo usuario y el modo núcleo.

El shell de este sistema operativo, permite a los usuarios escribir comandos para su ejecución, permitiendo también redireccionar la entrada y salida. La base de UNIX se centra en tres aspectos fundamentales:

1. El modelo de memoria, el cual consta de un segmento de texto, de datos y de pilas de proceso.
2. El sistema de archivos que maneja, un sistema de orden jerárquico, parecido a un árbol
3. Las Entradas / Salidas se dan a través de llamadas al sistema.

Del UNIX original se han desprendido una amplia gama de sistemas operativos, los cuales, han adoptado comandos, base estructural y algunas otras ventajas que este sistema ofrece, permitiendo establecer una nueva gama de sistemas operativos que cubren con las diversas necesidades presentadas por los usuarios, algunos de estos sistemas operativos son Linux y OpenBSD.

[7] SISTEMAS OPERATIVOS MODERNOS

Linux es un software libre de código abierto. Surge del proyecto GNU, el cual inicia en 1983.

Este proyecto tiene el objetivo de crear un sistema operativo tipo UNIX. En 1991 se libera la primera versión de Linux, gracias a la programación ejercida por miles de voluntarios principalmente de la Universidad de Helsinki, y principalmente por las aportaciones realizadas por Linus Torvals, quien es el creador del Kernel que maneja Linux.

Actualmente existen diferentes soluciones algunas comerciales basadas en Linux, distribuidas alrededor del mundo, cada una de estas soluciones se enfoca a diversas necesidades presentadas por los usuarios.

Una de ellas es OpenBSD que en su primera versión surgió en 1995 desarrollado por voluntarios de distintas partes del mundo, que se unen para cumplir normas y regulaciones, correcciones del código seguridad y criptografía que integran en él para su funcionalidad es por ello que se actualiza mediante parches y suites de seguridad que se puede obtener de su página de una manera gratuita, si se requiere anexar algún otro componente se puede descargar de las páginas del fabricante tanto de Software como de Hardware.
Anteriormente OpenBSD contaba con BIND en la versión 4.3 y para realizar alguna actualización se realizaba mediante los parches, pero a partir de la versión 4.6 ya se cuenta con la BIND versión 9 solo se necesita realizar los parches que se sugiere en la página de OpenBSD.

“OpenBSD produce un sistema operativo LIBRE, multi-plataforma, se orientan principalmente en la portabilidad, estandarización, seguridad proactiva y criptografía integrada” [8] su logotipo se muestra en la figura 3.1.

[8]Tomado y traducido de http://www.openbsd.org/es/

Figura 3.1 Logotipo de OpenBSD 4.6

3.3 CARACTERÍSTICAS DEL BIND

BIND ofrece un Servidor de Nombre de Dominio a través del archivo Named. Named además de ser una biblioteca de resolución de sistemas de nombres de dominio, es un paquete de herramientas para monitorizar el correcto funcionamiento de todo el sistema (bind-utils).

Gracias a su arquitectura mejorada se ha conseguido una mejor portabilidad entre sistemas.

Como se vio anteriormente el servidor de nombre de dominio realiza una consulta en forma de árbol invertido donde parte de los Root Server, pregunta a diferentes servidores de nombre de dominio dependiendo del nivel hasta encontrar la ubicación de donde se localiza el nombre que apunta a una IP, para que pueda ser posible la consulta es necesario configurar el archivo *named.conf* en el Servidor de Nombre de Dominio (DNS), donde se configura las distintas zonas a su cargo así como los dominios que se almacenan y todo el conjunto forma la base de datos. Como se muestra en la siguiente figura 3.2.

```
// $OpenBSD: named-simple.conf,v 1.9 2008/08/29 11:47:49 jakob Exp $
//
// Example file for a simple named configuration, processing both
// recursive and authoritative queries using one cache.


// Update this list to include only the networks for which you want
// to execute recursive queries. The default setting allows all hosts
// on any IPv4 networks for which the system has an interface, and
// the IPv6 localhost address.
//
acl clients {
        localnets;
        ::1;
};

options {
        version "";     // remove this to allow version queries

        listen-on    { any; };
        listen-on-v6 { any; };

        empty-zones-enable yes;

        allow-recursion { clients; };
        dnssec-enable yes;

};

logging {
        category lame-servers { null; };
};
```

Figura 3.2 named.conf

```
# End of named.conf
// Standard zones
//
zone "." {
        type hint;
        file "etc/root.hint";
};

zone "localhost" {
        type master;
        file "standard/localhost";
        allow-transfer { localhost; };
};

zone "127.in-addr.arpa" {
        type master;
        file "standard/loopback";
        allow-transfer { localhost; };
};

zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.ip6.arpa" {
        type master;
        file "standard/loopback6.arpa";
        allow-transfer { localhost; };
};
```

Figura 3.2 named.conf

Como se vio con anterioridad uno de los servidores que configuraremos es de tipo maestro (Master) y esclavo-(Slave).

Por ejemplo para la configuración de zona maestra se configura de la siguiente manera:

zone "myzone.net" ---------- Nombre de la zona

{

type **master;**

file "master/myzone.net"; ---------- Archivo de configuración

}

Para la configuración en un esclavo es:

zone "otherzone.net" ---------- Nombre de la zona

{

type **slave;**

file "slave/otherzone.net"; ----------▶ Archivo de configuración
masters { 192.0.2.1; [...;] };
};

Donde cada zona es asociada a un archivo de de configuración donde contienen la información de la misma como se puede ver en la siguiente figura 3.3.

```
;  RR NS Name Servers
;  ____________________
;
        IN      NS      kate.nic.unam.mx.
        IN      NS      jack.nic.unam.mx.

;  RR MX Mail Exchangers
;  _____________________
;


;
;  RR A Name-to-Address Mapping
;  ____________________________

www             IN      A       132.248.120.137
;
;  RR CNAME Canonical Name (Alias)
;  _______________________________
;
ww2             IN      CNAME   www
```

Figura 3.3 Archivo de configuración

Donde se encuentran los RRs (*Resource Records) su formato es:*

[nombre] [TTL] [class] <type> <RDATA>

Donde:

- ***nombre**: Dominio o máquina a la cual se agrega un RR.*
- ***TTL**: Time to Live (tiempo de vida del registro).*
- ***Class**: tipo de red (IN, CHAOS).*
- ***Type**: Función del registro.*
- ***RDATA**: Datos del Registro.*

Los RRs (Resource Record) más conocidos son:

- **NS** (**N**ame **S**erver): Es el nombre de los servidores DNS tanto primarios como secundarios que se encuentran definidos dentro del archivo de zona es decir que disponen de la dirección y nombre para el dominio.
- **A** (**A**ddress)**:** Indica la dirección IP asociada al nombre host.
- **MX** (**M**ail E**x**changer)**:** Es el nombre del servidor encargado del correo en ese dominio así como la prioridad.
- **CNAME** (**C**anonical **Name**)**:** Indica cuál sea el nombre canónico de un alias.
- **PTR** (**P**oin**t**e**r**)**:** Host Name – Pointer Indica el dominio asociado a una dirección IP.
- **TXT** (**T**e**xt**)**:** – datos del Host arbitrariamente utilizados para las listas negras.
- **SOA** (**S**tart **of A**uthority) indica los datos de la autoridad para el domino o zona en cuestión, por lo que cada dominio deberá de existir.

Siendo el SOA el principal RRs porque contiene la siguiente información:

- ***Serial***: número de modificaciones que se realizaron a la tabla y debe de ser incrementada cada vez que se realicen cambios en los datos de la zona.de lo contrario el servidor no releerá la nueva información y el servidor secundario no se actualizara.

- ***Refresh*** es el intervalo de tiempo que cuenta desde la última vez que se realizo la actualización del archivo de zona en el Slave, es decir que desde la última actualización a partir tiene cierto tiempo para actualizar el Slave.

- ***Retry:*** es el tiempo que el secundario debe esperar para reintentar la actualización de la zona encaso de que falle la conexión al hacer el ***refresh.***

- ***Expire:*** tiempo después del cual si no se ha llegado a realizar la actualización se desecha la zona y el Slave deja de responder a actualizaciones y peticiones con respecto al dominio u zona.

- ***TTL***: Intervalo de tiempo asociado a cada uno de los RRs por cuánto tiempo guarda en cache el registro.

Como se muestra en la siguiente figura 3.4.

```
$TTL 2h
;
;  RR SOA Start of Authority
;  ------------------------
;
@       IN      SOA     kate.nic.unam.mx.       dns.unam.mx. (
                        2010081100              ; Serial  [yymmddss]
                        3600                    ; Refresh [secs]
                        1200                    ; Retry   [secs]
                        604800                  ; Expire  [secs]
                        7200 )                  ; TTL     [secs]
;
```

Figura 3.4 Configuración SOA

Pero como no es solo la implementación que se tiene que realizar a un servidor de nombre de dominio conforme el tiempo han surgido problemas ya que existen diferentes vulnerabilidades que conforme el tiempo ha surgido y que ocasionan dificultades para la administración de DNS.

3.4 IMPLEMENTACIÓN DE EXTENSIONES DE SEGURIDAD

Para la administración del servidor de nombre de dominio necesitamos que este, pueda permitir la manipulación de él remotamente para ello es necesario configurar las llaves de RNDC.

3.4.1 RNDC

BIND incluye la utilidad llamada rndc, la cual permite la administración de línea de comandos del demonio named desde el host local o desde un host remoto. Para ello BIND utiliza dos puertos en sus comunicaciones, el 53 TCP, que se usa para las transferencias y el 53 UDP, que se utiliza para las consultas. Es necesario no aplicar reglas de cortafuegos sobre estos puertos si queremos que el servicio DNS funcione de forma correcta. Mientras que la herramienta rndc utiliza el puerto 953 UDP para el control remoto.

Para prevenir el acceso no autorizado al demonio named, BIND utiliza un método de clave secreta compartida para otorgar privilegios a hosts.

Esto es que si se utiliza utiliza una clave idéntica, estas se obtiene con el comando ***rndc-congen*** y debe estar presente en los archivos de configuración como son:

- **/etc/named.conf**
- **/etc/rndc.conf**

En la siguiente figura 3.5 se muestras el resultado al aplicar el comando.

```
key "rndc-key" {
        algorithm hmac-md5;
        secret "D6CwiMmF474nHfhl/ZKBcw==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf

# Use with the following in named.conf, adjusting the allow list as needed:
# key "rndc-key" {
#       algorithm hmac-md5;
#       secret "D6CwiMmF474nHfhl/ZKBcw==";
# };
#
# controls {
#       inet 127.0.0.1 port 953
#               allow { 127.0.0.1; } keys { "rndc-key"; };
# };
# End of named.conf
kate#
```

Figura 3.5 Llaves de rndc

Pero para el caso práctico están colocados en diferentes rutas como son:

- /var/named/etc/named.conf como se muestras en la siguientes figura 3.6.

```
# pwd
/var/named/etc
# ls
named-dual.conf       named.conf
named-simple.conf     root.hint
#
```

Figura 3.6 ubicación del archivo named.conf

La declaración *controls* permite a *rndc* conectarse desde un host local, como se observa en la siguiente figura 3.7.

```
options {
        version "";     // remove this to allow version querie

        listen-on    { any; };
        listen-on-v6 { any; };

        empty-zones-enable yes;

        allow-recursion { clients; };
};

logging {
        category lame-servers { null; };
};

key "rndc-key" {
       algorithm hmac-md5;
       secret "Qg1BmBbUXRY0zeONkWpYxg==";
};

controls {
       inet 127.0.0.1 port 953
              allow { 127.0.0.1; } keys { "rndc-key"; };
};
```

Figura 3.7 declaración *controls* en el *named.conf*

En esta declaración le proporciona al documento named, que en el puerto 953 por TCP se encuentre a la escucha de la dirección loopback y que permita al comandos rndc provenientes del host local, autentificarse y realizar comunicación controlando al demonio named, si se proporciona la clave correcta.

- /etc/rndc.conf como se muestra en la siguiente imagen 3.8

```
# pwd
/etc
# cat rndc.conf
key "rndc-key" {
        algorithm hmac-md5;
        secret "Qg1BmBbUXRY0zeONkWpYxg==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf
#
```

Figura 3.8 rndc.conf

El valor <*key-name*> se relaciona con la declaración *key*, la cual está también en el archivo */etc/named.conf*

3.4.2 OPCIONES DE RNDC

El comando rndc tiene la siguiente forma:

rndc *<options> <command> <command-options>*

Están disponibles las siguientes opciones:

- **-c *<configuration-file>*** indica a rndc que use un archivo de configuración diferente a /etc/rndc.conf.
- **-p *<port-number>*** Especifica la utilización de un número de puerto diferente del predeterminado 953 para la conexión del comando rndc.
- **-s *<server>*** Indica a rndc que envié el comando a un servidor distinto al default-server especificado en su archivo de configuración.
- **-y *<key-name>*** Le permite especificar una clave distinta de la opción default-key en el archivo /etc/rndc.conf.

Y cuando se ejecuta rndc en una máquina local configurada de la forma correcta se pueden ejecutar los siguientes comandos:

- **halt** Detiene el servidor sin grabar los updates pendientes del servicio named.
- **querylog** Registra todas las peticiones de logueo hechas a este servidor de nombres.
- **refresh** Refresca la base de datos del servidor de nombres.
- **reload** Recarga los archivos de configuración de zona pero mantiene todas las respuestas precedentes situadas en caché. Esto le permite realizar cambios en los archivos de zona sin perder todas las resoluciones de nombres almacenadas.

  Si los cambios sólo afectaron una zona específica, vuelva a cargar una zona añadiendo el nombre de la zona después del comando reload.
- **stats** Descarga las estadísticas actuales de named al archivo /var/named/named.stats.
- **reconfig** relee la configuración y verificando toda el contenido actual.
- **stop** Graba los updates pendientes en archivo maestro y detiene al servidor, guardando todas las actualizaciones dinámicas y los datos de las transferencias de zona incremental (IXFR) antes de salir.

Para mayor información sobre estas opciones se puede consultar el manual de rndc.

Las llaves de rndc ayudan a los servidores de nombre de dominio a reconocer cuales son los servidores maestros o primarios ante los secundarios.

Para revisar que la sintaxis del archivo de configuración comprobando si hay errores del tipo de sintaxis o topográficos, se puede utilizar una herramienta llamada **named-checkconf** para comprobar servidor BIND DNS (nombre demonio), pero no puede comprobar si hay mal MX o una dirección asignada por nosotros. Sin embargo, esta es una herramienta excelente para solucionar problemas relacionados con el servidor DNS.

3.5 HERRAMIENTAS DE DIAGNÓSTICO

Para poder realizar, un monitoreo adecuado de los cambios y buen funcionamiento de los equipos, los administradores utilizan una serie de herramientas, que permiten consultar las operaciones realizadas por el equipo.

Algunas de estas herramientas, pueden ser descargadas de la red, dependiendo del sistema operativo que utilice el administrador, otras son creadas a partir de sus necesidades especificas.

Dentro de la amplia gama de opciones de herramientas, existen 3 herramientas básicas que son las más utilizadas por los administradores, PING, DIG y NSLOOKUP. A continuación se muestra una breve descripción y algunas opciones básicas de estas.

Nota: Las opciones de las herramientas dependen directamente del sistema operativo, en el cual se instalen y apliquen.

3.5.1 PING

Esta herramienta comprueba la conectividad a nivel red, de forma rápida la conexión entre 2 equipos. Su creador Mike Mususs, le asigno este nombre a esta herramienta, tomando en cuenta la opción de sonar para localizar un objeto.

Con ayuda de ping podremos determinar si el nivel de red funciona adecuadamente, así como los niveles físicos y de enlaces sobre los que descansa.

La forma básica de invocar a PING es:

ping <opcion> <ip>

Donde:

- **opción**: es el complemento de opciones a realizar, dentro de la herramienta. Algunas opciones son:

- **-c**: especifica el número de paquetes enviados.
- **-s**: tamaño del paquete enviado
- **-t:** especifica el tiempo de vida (tiempo de respuesta del paquete enviado)
- **ip**: dirección ip del equipo destinatario al que se le va a realizar el ping.

PING, trabaja bajo la funcionalidad "echo request" y "echo reply" del protocolo ICMP (Internet Control Message Protocol), donde, una entidad ICMP emisora envía un paquete pequeño, a un equipo destinatario, el cual es determinado por el usuario al momento de invocarlo, el ICMP del equipo destinatario, recibe el paquete y lo reenvía a la entidad emisora. Como se muestra en al siguiente figura 3.9.

```
# ping 132.248.10.1
PING 132.248.10.1 (132.248.10.1): 56 data bytes
64 bytes from 132.248.10.1: icmp_seq=0 ttl=62 time=0.768 ms
64 bytes from 132.248.10.1: icmp_seq=1 ttl=62 time=0.520 ms
64 bytes from 132.248.10.1: icmp_seq=2 ttl=62 time=0.557 ms
64 bytes from 132.248.10.1: icmp_seq=3 ttl=62 time=0.334 ms
64 bytes from 132.248.10.1: icmp_seq=4 ttl=62 time=0.608 ms
64 bytes from 132.248.10.1: icmp_seq=5 ttl=62 time=0.509 ms
64 bytes from 132.248.10.1: icmp_seq=6 ttl=62 time=0.411 ms
64 bytes from 132.248.10.1: icmp_seq=7 ttl=62 time=47.534 ms
--- 132.248.10.1 ping statistics ---
8 packets transmitted, 8 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 0.334/6.405/47.534/15.545 ms
#
```

Figura 3.9 Ejemplo de la herramienta ping

La salida del comando PING, nos muestra el nombre y la dirección IP del equipo al que se hace realiza el PING, además de la cantidad de datos que se envían en cada paquete (suelen ser por lo general 64 bytes), esta línea se repite en cada una de las respuestas a los paquetes enviados.
Esta herramienta se interrumpe con las teclas Ctrl. + C, al finalizar muestra una pequeña estadística basada en el número de paquetes enviados y recibido.

3.5.2 DIG (DOMAIN INFORMATION GROPER)

DIG, es una herramienta que realiza consultas de diversos tipos a un DNS. Esta herramienta, muestra las respuestas recibidas de acuerdo a la pregunta realizada. Es muy útil para detectar problemas en la configuración de los DNS debido a su claridad, flexibilidad y facilidad de uso.

La herramienta DIG tiene dos modos de invocarse: comando modo-simple línea para preguntas simples o múltiples, y el modo batch para la lectura de peticiones lookup de un archivo.

La forma básica de invocar a DIG es:

dig <servidor> <nombre> [tipo]

Donde:

- **Servidor:** es el nombre o la dirección IP del servidor a consultar.
- **Nombre:** es el nombre de dominio del record por el cual se quiere preguntar.
- **Tipo:** es el tipo del record por el que se consulta (ANY, NS, SOA, MX, etc.). De no indicarse un tipo especifico, dig asumirá el tipo A.

Algunas opciones básicas de DIG son:

- **@:** Especifica los servidores DNS que son utilizados en cada pregunta. Si no se provee un nombre especifico de Servidor, DIG intenta con cada servidor listado en el /etc/resolv.conf.

- **-h:** muestra la ayuda del comando.
- **-x:** hace consultas inversas, o sea, a partir de las direcciones IP determina nombres de dominio.
- **-f <filename>**: toma las consultas a partir de un fichero. Estas se definen una por línea y con la misma sintaxis que en la línea de comando.
- **-b <dirección>:** indica la dirección IP a partir de la cual se realizará la consulta dado el caso en que se tenga más de una interfaz de red configurada.

La sintaxis de esta herramienta en modo **–** simple línea es:

dig [servidor] [opciones] [nombre] [tipo] [clase] [opciones de consulta]

La sintaxis de esta herramienta en modo batch es:

dig [servidor-global] [opciones-d-global] dominio [servidor] [opciones] [q-opciones] [q-tipo] [q-clase] [dominio [servidor][opciones] [q-opciones] [q-tipo] [q-clase] [...]]

Las opciones de preguntas de dominio global controlan las búsquedas y despliegan los resultados de preguntas múltiples y afectan todas las preguntas.

Nota: Cada conjunto global de opciones de pregunta deben ser sobrescritas por cada conjunto de opciones de pregunta, por cada pregunta individual

En la salida de esta herramienta, hay una serie de campos identificados con una serie de letras. Estas letras indican la información específica, con respecto a la consulta realizada, no todas estas letras aparecen, en la salida arrojada por DIG.

Como se muestra en la siguiente figura 3.10.

```
jack# dig www.iingen.unam.mx

; <<>> DiG 9.4.2-P2 <<>> www.iingen.unam.mx
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37408
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;www.iingen.unam.mx.            IN      A

;; ANSWER SECTION:
www.iingen.unam.mx.     7200    IN      A       132.248.120.137

;; AUTHORITY SECTION:
iingen.unam.mx.         7200    IN      NS      kate.nic.unam.mx.
iingen.unam.mx.         7200    IN      NS      jack.nic.unam.mx.

;; Query time: 3 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Jun 29 17:21:32 2010
;; MSG SIZE  rcvd: 94

jack#
```

Figura 3.10 Ejemplo de la aplicación DIG

3.5.3 NSLOOKUP (*NAME SYSTEM LOOKUP*)

Es una herramienta que permite consultar un servidor de nombres para obtener información relacionada con el dominio o el host, así se diagnostica los eventuales problemas de configuración que pudieran haber surgido en el DNS. Basta con introducir el nombre de un dominio en la invitación de comando para detallar las características. De la misma manera, es posible solicitar información sobre un host indicando su nombre seguido del comando nslookup.

Nslookup tiene dos modos: interactivo y no interactivo.

- **Interactivo** este permite al usuario consultar servidores de nombres para obtener información sobre diversos huéspedes y dominios o para imprimir una lista de hosts en un dominio.

- **No interactivo** se utiliza para imprimir sólo el nombre y la solicitó información de un host o de dominio.

Su sintaxis es

nslookup[nombre | -] [-opciones] [servidor]

Para la forma interactiva

- **host [servidor]** Buscar información para el host utilizando el valor predeterminado actual servidor o utilizando el servidor.
- **Server** busca el servidor del dominio
- **lserver**: busca información acerca del dominio.
- **set [palabras] = valor** Este comando se utiliza para cambiar la información de estado que afecta a las búsquedas palabras como: **q= ns** busca la zona.

Nota: Las opciones de las herramientas dependen directamente del sistema operativo, en el cual se instalen y apliquen es recomendable revisar el manual.

Como por ejemplo la siguiente figura 3.11.

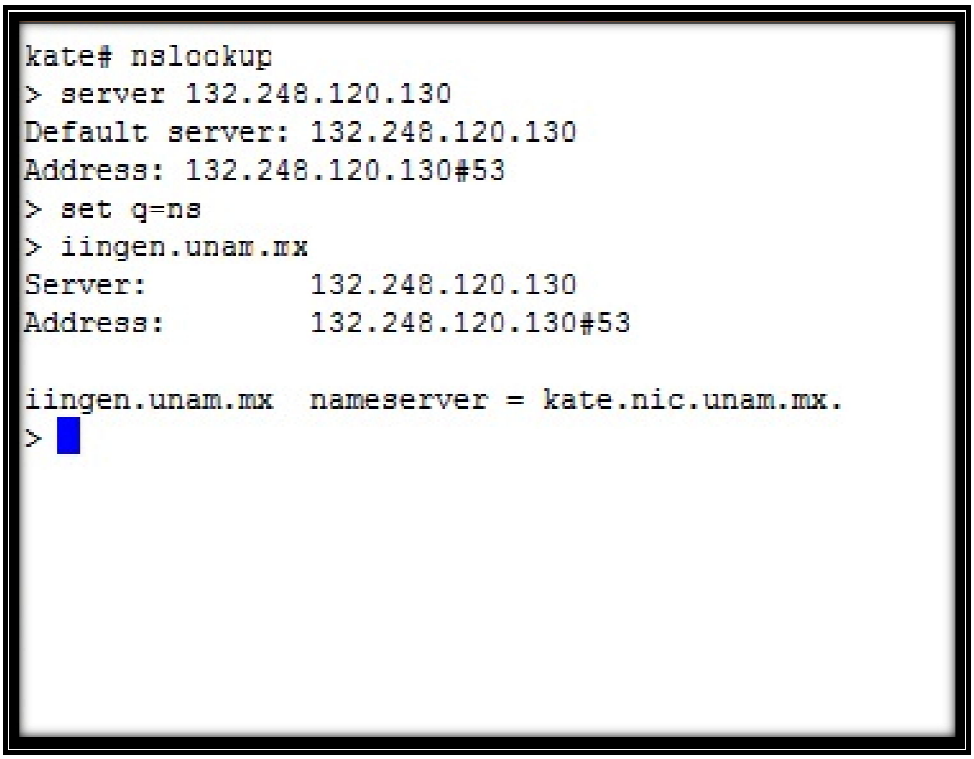


Figura 3.11 Ejemplo de Nslookup

3.6 TSIG

Para llevar a cabo la implementación de TSIG (Firma de transacción) es un protocolo que se define en el RFC 2845, se utiliza para proporcionar un medio de autenticación en la actualización de la base de datos en un DNS dinamico

TSIG utiliza para autentifica unas claves secretas mediante el one-way hashing para proporcionar medios seguro criptográficamente de identificación de cada punto final de una conexión, permitiendo realizar o responder a una actualización de DNS.

Aunque las consultas al DNS se pueden hacer de forma anónima, las actualizaciones de DNS deben ser autenticadas, puesto que al hacer cambios que duraderan en la estructura del sistema de nombres de Internet. El uso de

una clave compartida por el cliente que realiza la actualización y el servidor DNS garantiza la autenticidad de la solicitud de actualización.
Sin embargo, la solicitud de actualización se puede pasar sobre un canal inseguro. Una función unidireccional utilizada por one-way hashing se utiliza para prevenir los observadores malintencionados de aprendizaje de la clave secreta y usarla para hacer sus propias modificaciones.

Con la utilización de las funciones de hash, se permitirá asegurar que el que produce el mensaje es quien dice ser, y que el mensaje no ha sido alterado en un paso de la red.

Las actualizaciones de DNS, como consultas, que normalmente se transportan a través de UDP, ya que requiere menor sobrecarga que TCP. Sin embargo, los servidores DNS de apoyo tanto en las peticiones UDP y TCP.

“Para proporcionar la autenticación de clave secreta, se utiliza un nuevo tipo de RR cuya nemotécnica es TSIG y cuyo tipo de código es de 250. TSIG es un meta-RR y No debe ser almacenado en caché. TSIG RR´s se utilizan para la autenticación entre el DNS entidades que han establecido una clave secreta compartida. TSIG RR se ha calculado dinámicamente para cubrir una transacción de DNS.”[9]

3.6.1 GENERACIÓN DE LLAVES TSIG

Para la generación de llaves TSIG ocupa el algoritmo especificado en el RFC 1321 y 2104 “HMAC-MD5” para la aplicación de su operatividad, mediante la herramienta ***dnssec-keygen***.

Dnssec-keygen genera claves para DNSSEC especificado en el RFC 2535 y RFC 4034.A su vez también puede generar claves para su uso con TSIG (transacciones firmas), como definida en el RFC 2845.
Su sintaxis para TSIG es la siguiente.

[9] Tomado y traducido del rfc 2845

dnssec-keygen {-a algoritmo}{-b tamaño de la clave}{-n el tipo de nombre} nombre

A continuación explico las diferentes opciones:

- **- a algoritmo**: selecciona el algoritmo criptográfico a utilizar, para el caso de TSIG es obligatorio utilizar HMACMD5, eso especificado en el RFC 2845. Para el caso de DNSSEC se puede ocupar RSAMD5 (RSA) o RSASHA1, DSA, DH (Diffie-Hellman) o HMAC-MD5.
  Nota 1: que para DNSSEC se recomienda utilizar RSASHA -k.
- **-b el tamaño de la clave**: se especifica el número de bits en la clave. La elección del tamaño de la clave depende del algoritmo utilizado.
  RSAMD5 o RSASHA1 debe estar entre 512 y 2048 bits.
  DH(Diffie-Hellman) debe estar entre 128 y 4096 bits.
  DSA deben de ser entre 512 y 1024 bits y con exactitud de múltiplo de 64.
  HMAC-MD5 debe estar entre 1 y 512 bits.
- **-n tipo de nombre**: se especifica el tipo de propietario de la clave, es decir el valor de tipo de nombre o de ZONA (para una clave de zona DNSSEC(CLAVE / DNSKEY)), HOST o entidad (por una clave asociada con una gran cantidad (KEY)), usuario (por una clave asociada a unausuario (KEY)) u otros (DNSKEY). Estos valores son insensible.

Como se muestra en la siguiente figura 3.12.

```
kate# dnssec-keygen -a hmac-md5 -b 128 -n user rndc
Krndc.+157+30333
```

Figura 3.12 implementación de TSIG

En la imagen al momento de dar el comando nos proporciona dos llaves como son:

Krndc.+157+30333.key
Krndc.+157+30333.private

Como se muestra en la siguiente figura 3.13

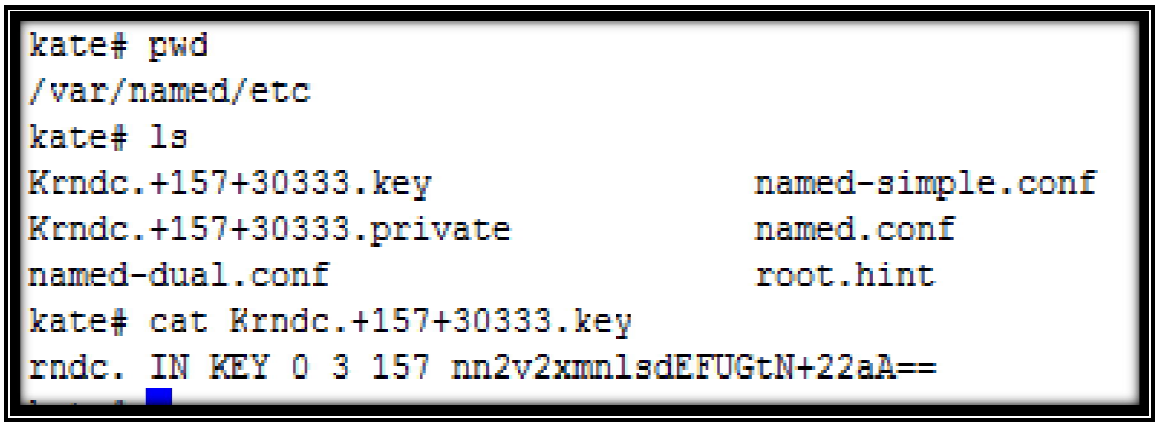


Figura 3.13 Llaves de TSIG obtenidas

La llave proporcionada un formato, el significado de cada campo se describe en el RFC 1035 como se muestra en la siguiente tabla 3.14.

Campo	Bytes	Descripción
NOMBRE	max 256	Nombre de clave, que debe ser único en el cliente y el servidor
TIPO	2	TSIG (250)
CLASE	2	CUALQUIER (255)
TTL	4	0 (desde que los registros TSIG no debe ser almacenado en caché)
RDLENGTH	2	Duración de la RDATA campo
RDATA	variable	Estructura que contiene la fecha y hora, el algoritmo de hash y datos

Tabla 3.14 Campos de registro de TSIG

Por los que rndc. IN KEY 0 3 157 nn2v2xmnlsdEFUGtN+22aA== siendo la última la llave que se anexara a el archivo named.conf del servidor primario o maestro como se muestra en la figura 3.15 y del secundario o esclavo como se observa en la figura 3.16.

```
//Pruebas dnssec

key kate.secreta.jack {
        algorithm hmac-md5;
        secret "nn2v2xmnlsdEFUGtN+22aA==";
};

server 132.248.120.129 {
        keys {kate.secreta.jack.;};
};

zone "iingen.unam.mx" {
        type master;
        file "master/iingenunam";
        allow-transfer {key kate.secreta.jack.;};
};

zone "120.248.132.in-addr.arpa" {
        type master;
        file "master/inverso248/120.132";
        allow-transfer {key kate.secreta.jack.;};
};
```

Figura 3.15 implementación de TSIG en el DNS maestro

```
//Pruebas dnssec

key kate.secreta.jack {
        algorithm hmac-md5;
        secret "nn2v2xmnlsdEFUGtN+22aA==";
};

sever 132.248.120.130 {
        keys {kate.secreta.jack.;};
};

zone "iingen.unam.mx" {
        type slave;
        file "slave/iingenunam";
        masters {132.248.120.130;};
};

zone "120.248.132.in-addr.arpa" {
        type slave;
        file "slave/120.132";
        masters {132.248.120.130;};
```

Figura 3.16 Implementación de TSIG en el DNS secundario o esclavo

3.7 GENERACIÓN DE LLAVES DNSSECBIS

En el anterior capítulo dnssecbis provee la integridad de datos, incluyendo mecanismos para verificación de negación de existencia.

Existen dos tipos de llaves llamadas:

- KSK solo es utilizada en la firma de ZSK, ya que el nodo padre valida la autenticidad de la llave KSK del hijo, esta llave se considera de mayor cantidad de bist
- ZSK es una llave que se utiliza para la firma de registros de la zona, es más sencillo porque su actualización es local.

Como se muestra en la siguiente figura 3.17

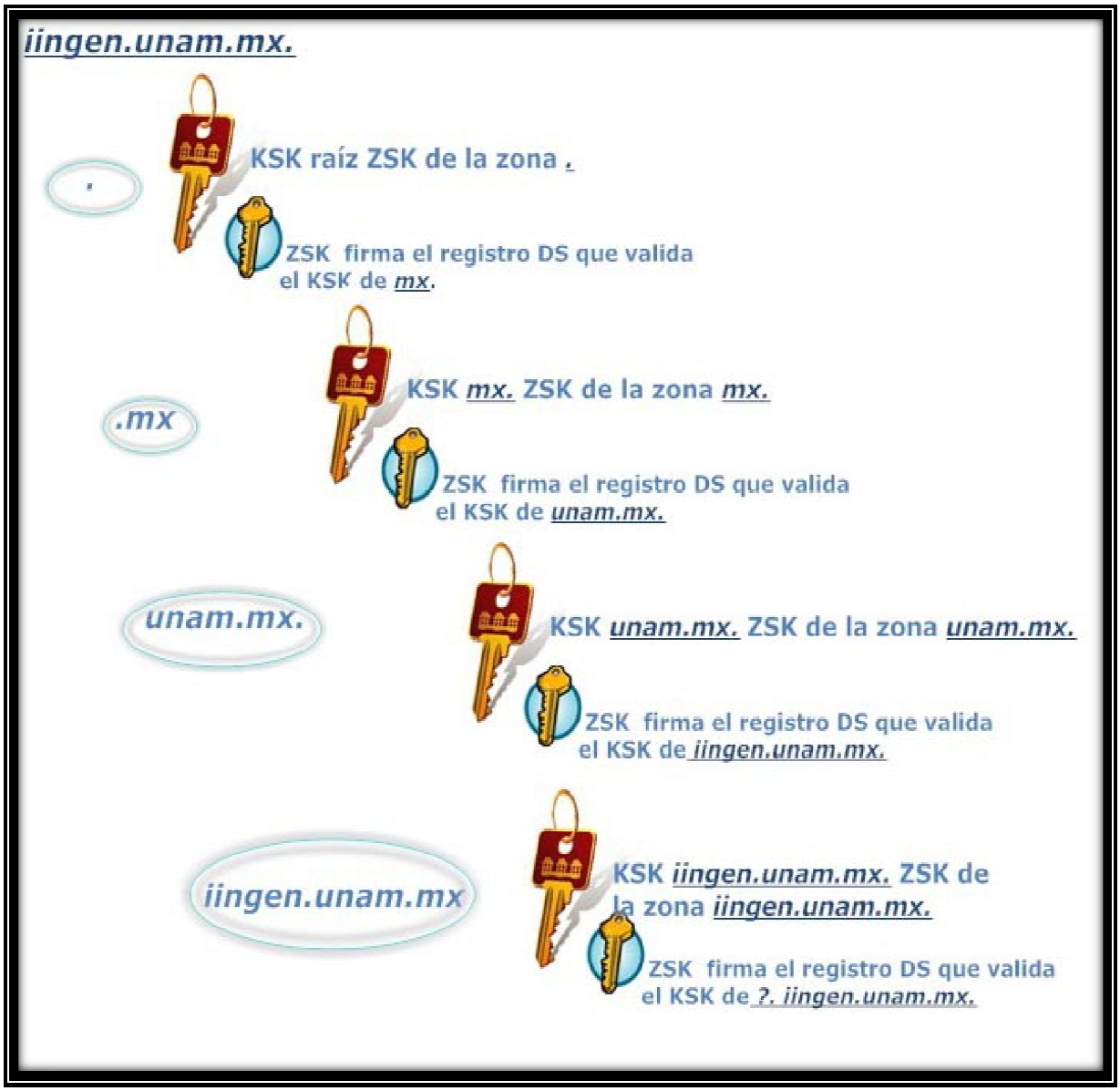


Figura 3.17 Estructura de firmas KSK y ZSK

Los servidores que resuelven mediante dnssec esta constituye la cadena de autenticación desde la raíz de la jerarquía hasta la zona.

Pero debido que aun no está completa la firma desde la raíz, se decidió realizar la firma desde unam.mx ya que se está realizando las pruebas en los servidores mx. y no se cuenta con su KSK.

Por esta razón realizaremos las pruebas en los servidores de la Universidad Nacional Autónoma de México (UNAM) creando así una isla de confianza. Para ello se llevara a cabo la firma DNSKEY, RRSIG y NEC.

3.7.1 DNSKEY

El RR de DNSSEY contiene la llave pública se utiliza para firmar, esta se obtiene con el siguiente comando:

dnssec-keygen –a RSASHA1 –b 512 –n Zone iingen.unam.mx para ZSK y
dnssec-keygen –a RSASHA1 –b 512 –f ksk –n Zone iingen.unam.mx para KSK

Como se menciona anteriormente se especifica que se utilizara el algoritmo RSASHA1 con 512 bits si ampliáramos mas bits el sistema se volvería muy lento por la generación de claves.
Donde contiene:

- Banderas del archivo donde indica el tipo de llave existen dos tipos 256 para ZSK y consultas como se muestra en la siguiente figura 3.18

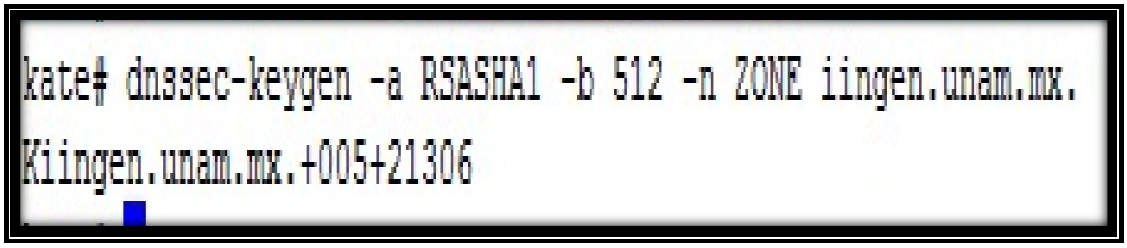


Figura 3.18 Generación de llave DNSKEY ZSK

Este entrega dos archivos que son
Kiingen.unam.mx.+005+21306.key figura 3.19.

Kiingen.unam.mx.+005+21306.private

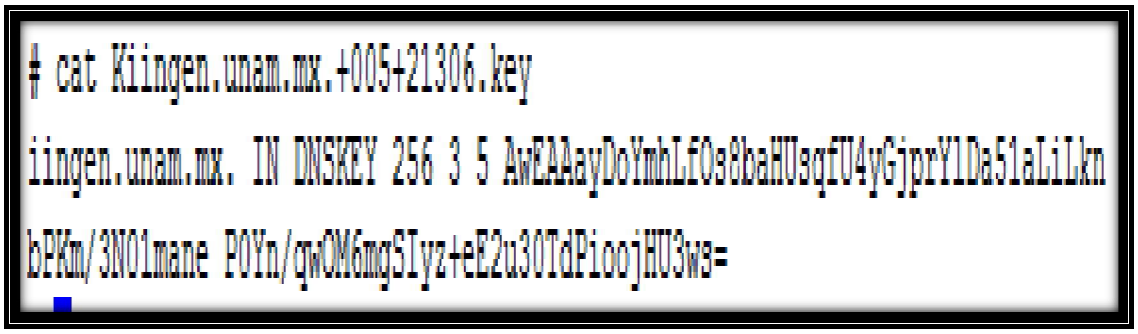


Figura 3.19 ZSK Kiingen.unam.mx.+005+21306.key

257 para KSK y consultas donde se entregan los siguientes archivos: Kiingen.unam.mx.+005+12604.key como se muestra en la siguiente figura 3.20 Kiingen.unam.mx.+005+12604.private

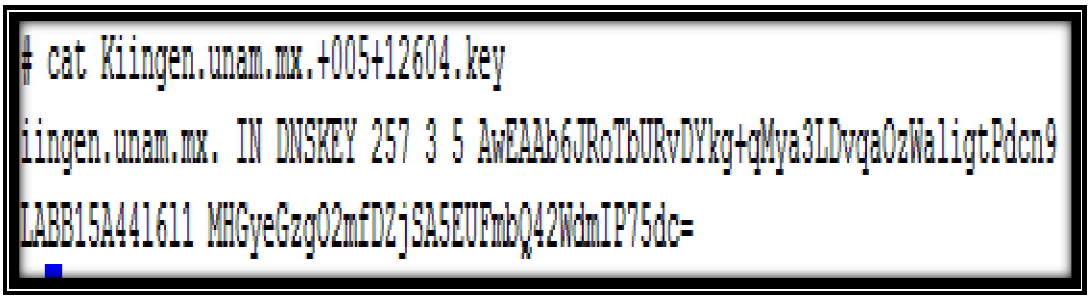


Figura 3.20 SKS Kiingen.unam.mx.+005+21306.key

Este entrega dos archivos que son:

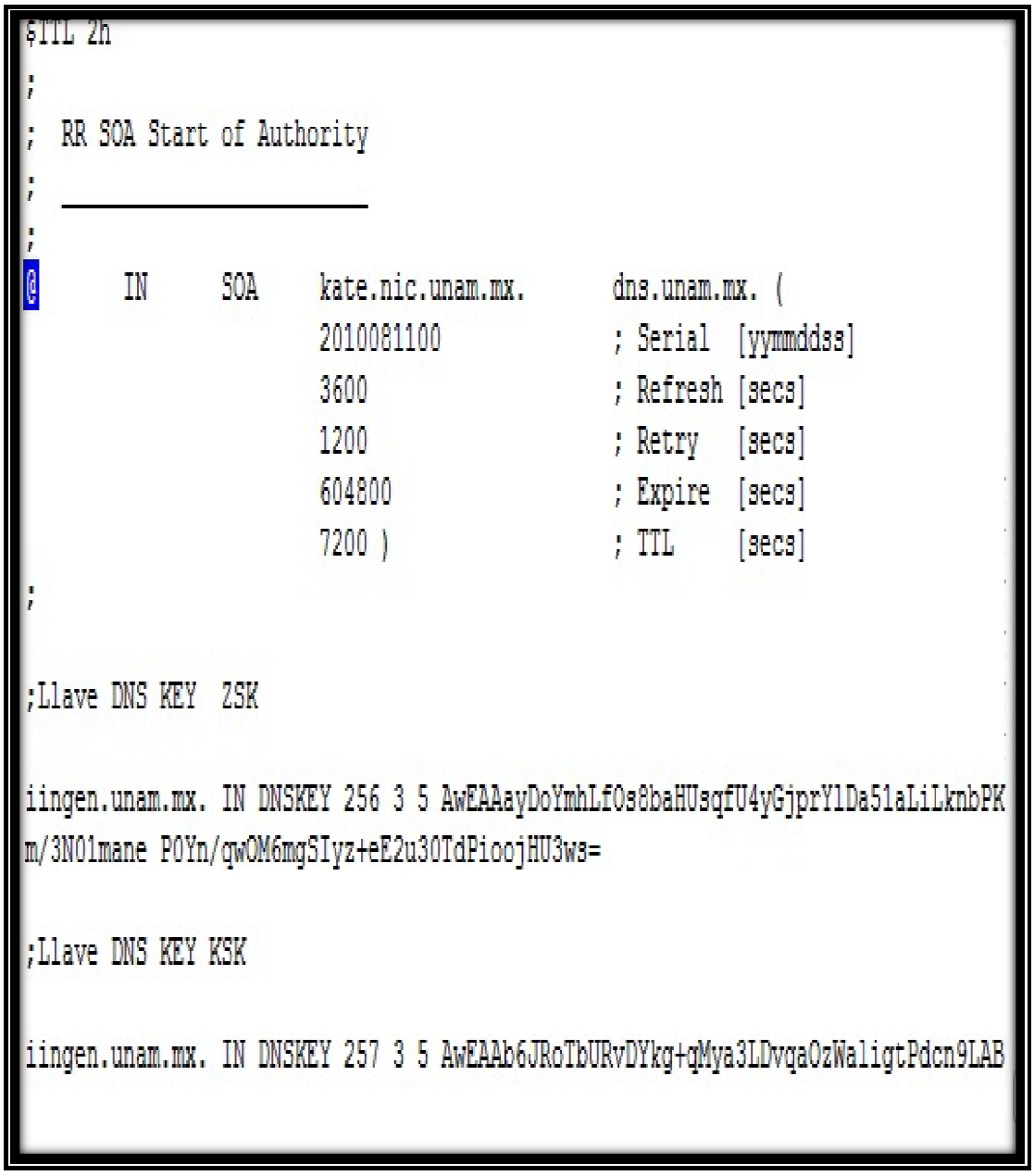
- El protocolo del archivo que tiene de valor 3
- El algoritmo del archivo donde se indica el tipo de algoritmo para generar la llave como se muestra en la siguiente tabla 3.21

Valor	Algoritmo	Status
0	Reservado	
1	RSA7MD5[RSAMD5]	No recomandado
2	Diffie-Herllman[DH]	
3	DSA/SHA-1 [DSA]	Opcional
4	Eliptic Curve [ECC]	
5	RSA/SHA-1[RSASHA1]	Mandatorio
252	Indirect [INDIRECT]	
253	Private[PRIVATEDNS]	Opcional
254	Private[PRIVATEOID]	Opcional
255	reservado	

Tabla 3.21 algoritmos para emplear

- La publicación de la llave del archivo en base 64

Esta será clocada en el archivo de configuración como se observa en la siguiente figura 3.22.



```
$TTL 2h
;
;  RR SOA Start of Authority
;  ------------------------
;
@       IN      SOA     kate.nic.unam.mx.       dns.unam.mx. (
                        2010081100              ; Serial  [yymmddss]
                        3600                    ; Refresh [secs]
                        1200                    ; Retry   [secs]
                        604800                  ; Expire  [secs]
                        7200 )                  ; TTL     [secs]
;

;Llave DNS KEY  ZSK

iingen.unam.mx. IN DNSKEY 256 3 5 AwEAAayDoYmhLfOs8baHUsqfU4yGjprY1Da51aLiLknbPK
m/3NO1mane POYn/qwOM6mgSIyz+eE2u30TdPiocjHU3ws=

;Llave DNS KEY KSK

iingen.unam.mx. IN DNSKEY 257 3 5 AwEAAb6JRoTbURvDYkg+qMya3LDvqaOzWaligtPdcn9LAB
```

Figura 3.22 Implementación de DNSKEY en archivo de resolución

Y al momento de consultar la base de datos con la herramienta dig nos muestra la firma que se realizó como se observa en la siguiente figura 3.23.

```
kate# dig @132.248.120.130 iingen.unam.mx dnskey

; <<>> DiG 9.4.2-P2 <<>> @132.248.120.130 iingen.unam.mx dnskey
; (1 server found)
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46905
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 0

;; QUESTION SECTION:
;iingen.unam.mx.                        IN      DNSKEY

;; ANSWER SECTION:
iingen.unam.mx.         7200    IN      DNSKEY  256 3 5 AwEAAayDoYmhLfOs8baHUsqf
U4yGjprYlDa51aLiLknbPKm/3N01mane P0Yn/qwOM6mgSIyz+eE2u30TdPioojHU3ws=
iingen.unam.mx.         7200    IN      DNSKEY  257 3 5 AwEAAb6JRoTbURvDYkg+qMya
3LDvqaOzWaligtPdcn9LABB15A441611 MHGyeGzgO2mfDZjSA5EUFmbQ42WdmIP75dc=

;; AUTHORITY SECTION:
iingen.unam.mx.         7200    IN      NS      kate.nic.unam.mx.
iingen.unam.mx.         7200    IN      NS      jack.nic.unam.mx.

;; Query time: 1 msec
;; SERVER: 132.248.120.130#53(132.248.120.130)
```

Figura 3.23.Consuta de DNSKEY

3.7.2 RRSIG

Es un archivo que contiene la firma digital que valida a otros registros o RR Mediante el siguiente comando Dnssec-sigzone –o zona-k llave zona ZSK.

Generando así un archivo, como se muestra en la siguiente figura 3.24.

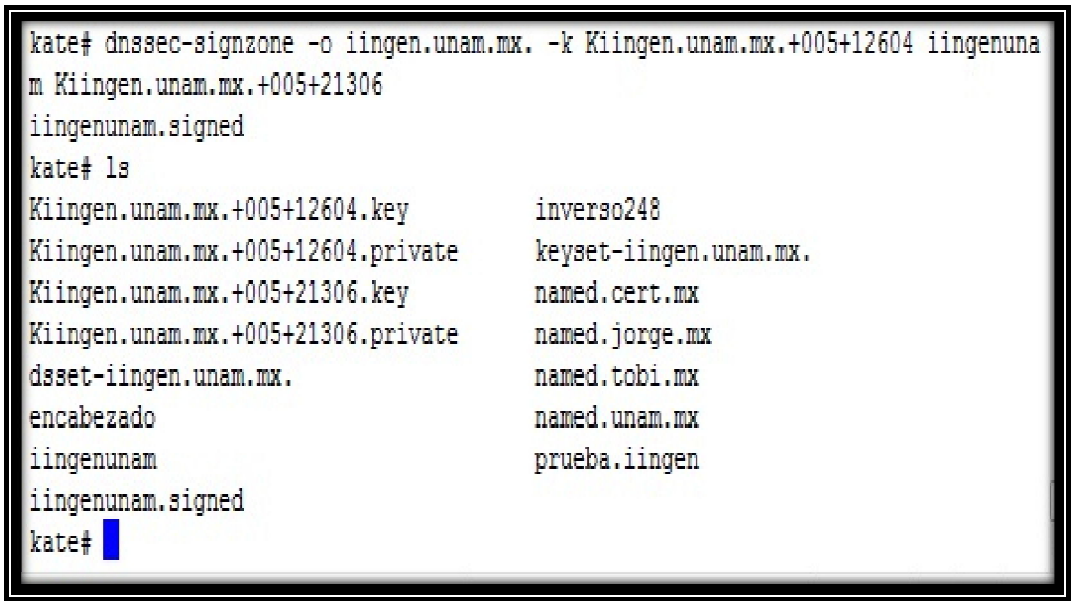


Figura 3.24.Implementación RRSIG

El nuevo archivo a consultar es iingen.unam, después de ejecutar el comando se creó el siguiente archivo iingenunam.signed donde firma todas los RR que se encuentran configurados como se observa en la siguiente figura 3.25.

```
; File written on Fri Aug 13 13:02:11 2010
; dnssec_signzone version 9.4.2-P2
iingen.unam.mx.         7200    IN SOA  kate.nic.unam.mx. dns.unam.mx. (
                                        2010081100 ; serial
                                        3600       ; refresh (1 hour)
                                        1200       ; retry (20 minutes)
                                        604800     ; expire (1 week)
                                        7200       ; minimum (2 hours)
                                        )
                        7200    RRSIG   SOA 5 3 7200 20100912170211 (
                                        20100813170211 21306 iingen.unam.mx.
                                        X9bBDUmYufjs9ImihAFm6Jgf3ncDASWDm9tZ
                                        sSEUMYF7K8m0mzWVqv5EfiYd793vqOP2/EOw
                                        1aOoCPliX6u6NQ== )
                        7200    NS      kate.nic.unam.mx.
                        7200    NS      jack.nic.unam.mx.
                        7200    RRSIG   NS 5 3 7200 20100912170211 (
                                        20100813170211 21306 iingen.unam.mx.
                                        lVIdmco8P0ppDUbOC+hnBN1nzaSFt7vsRUSl
                                        Xo/eIzjj4OZgGhJikHqVGjxjwzBn9ucgnbyn
                                        9rTo7/+FH/uipg== )
                        7200    NSEC    ww2.iingen.unam.mx. NS SOA RRSIG NSEC DNSKEY
                        7200    RRSIG   NSEC 5 3 7200 20100912170211 (
                                        20100813170211 21306 iingen.unam.mx.
                                        nF2LeC1+R4txo5BviP+vasF9We1oQA97FEN3
                                        iV5PoURuHoic9YbLqnFPejOzSE0ZWqUo+1Gz
                                        zVjqvMGly111rg== )
                        7200    DNSKEY  256 3 5 (
                                        AwEAAayDoYmhLfOs8baHUsqfU4yGjprY1Da5
                                        1aLiLknbPKm/3N01manePOYn/qwOM6mgSIyz
                                        +eE2u30TdPioojHU3ws=
                                        ) ; key id = 21306
                        7200    DNSKEY  257 3 5 (
                                        AwEAAb6JRoTbURvDYkg+qMya3LDvqaOzWali
                                        gtPdcn9LABB15A44161lMHGyeGzgO2mfDZjS
                                        A5EUFmbQ42WdmIP75dc=
                                        ) ; key id = 12604
                        7200    RRSIG   DNSKEY 5 3 7200 20100912170211 (
                                        20100813170211 21306 iingen.unam.mx.
iingenunam.signed 52%
```

Figura 3.25 archivo firmado por RRSIG