UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

SOFTWARE DE GENERACIÓN DE REPORTE DE RESILIENCIA DEL INTERNET A NIVEL DNS

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERA CIVIL EN COMPUTACIÓN

MAITE MANUELA GONZÁLEZ MENDOZA

PROFESOR GUÍA:
JAVIER BUSTOS JIMÉNEZ

MIEMBROS DE LA COMISIÓN:
ALEJANDRO HEVIA ANGULO
GONZALO NAVARRO BADINO

Este trabajo ha sido parcialmente financiado por NIC Chile Research Labs

SANTIAGO DE CHILE
ABRIL 2017

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE INGENIERA CIVIL EN COMPUTACIÓN
POR: MAITE MANUELA GONZÁLEZ MENDOZA
FECHA: ABRIL 2017
PROF. GUÍA: JAVIER BUSTOS JIMÉNEZ

SOFTWARE DE GENERACIÓN DE REPORTE DE RESILIENCIA DEL INTERNET A NIVEL DNS

El hecho de que Internet sea tan masivo en estos tiempos es debido principalmente al nacimiento del World Wide Web (WWW) en 1993. Internet logró llegar a muchos más usuarios debido a la facilidad de sólo tener que recordar un nombre para acceder al contenido. El primer paso para poder acceder al contenido web (entre otros contenidos de Internet) es traducir la dirección del servidor web al sistema de números IP que permite alcanzar el contenido. La traducción se hace utilizando el sistema DNS (*Domain Name System*), sistema jerárquico distribuido que, a partir de un nombre de dominio, entrega la dirección IP asociada, para así poder acceder al contenido del servicio solicitado.

El sistema DNS está basado en la buena fe de los participantes, por lo que es fácil de vulnerar si no se cumplen bien las recomendaciones de seguridad. Además, como es un sistema distribuido, tiene las fallas de un sistema de esa categoría, por lo que es necesario evaluar el estado actual para prevenir problemas y mejorar el sistema.

Esta memoria describe el desarrollo de un software de recopilación de datos de Internet chileno para chequear el estado de la implementación del sistema DNS en los dominios bajo el `.cl`, entregando un reporte con estadísticas de implementación de IPv6, DNSSE y comunicación mediante TCP, entre otros. El desarrollo de esta memoria nació con una idea desarrollada por AFNIC y ANSSI, que consiste en entregar un reporte para evaluar la resiliencia del Internet en los dominios franceses. Para este fin se estudiaron varios conceptos relacionados que se pueden encontrar en los RFCs asociados a DNS y DNSSEC, los que describen el correcto funcionamiento y buenas prácticas.

Luego de eso, se diseñó la arquitectura del software y se eligieron las herramientas a utilizar, con lo que se desarrolló la herramienta final que recopila datos, los analiza y finalmente genera un reporte. Como resultados de la implementación de este software y su utilización se obtuvieron datos del estado actual de Internet y del estado de cumplimiento de las recomendaciones. Sin embargo, dado que no existen resultados anteriores con los que compararlos, no se puede saber cómo ha evolucionado, pero con esta primera vez se establece un precedente, el cual va a ser útil para comparar y poder mejorar la resiliencia del Internet chileno a nivel DNS.

Agradecimientos

Durante este proceso de titulación recibí el apoyo y ayuda de varias personas, por eso quiero dar los siguientes agradecimientos:
A Javier Bustos Jiménez por su apoyo, motivación y guía durante todo el proceso de titulación.
A Hugo Salgado por toda la ayuda, los recursos y el conocimiento brindado para poder realizar una mejor herramienta.
A Alexandra Ibarra por toda la ayuda en la parte del diseño de los gráficos. A toda la gente de NICLabs por el compañerismo y el buen ambiente. A mi familia y amigos por el apoyo y el amor ♡.

Tabla de Contenido

Capítulo 1

Introducción

Internet, desde hace algunos años, se ha estado transformando en la base de los servicios, los negocios e incluso hasta de la vida social de la gente. Sin embargo, la mayoría de las personas no sabe cómo funciona. Sólo utilizan Internet ingresando, por ejemplo, un nombre del dominio en el navegador y esperando que éste les devuelva un resultado, sin notar todo lo que pasa tras las bambalinas del Internet, como por ejemplo, la importante tarea del Sistema de Nombres de Dominios o DNS.

1.1. ¿Qué es el Sistema de Nombres de Dominio?

El Sistema de Nombres de Dominios (o DNS, por su sigla en inglés) se compone de un conjunto de herramientas que ayuda a traducir los nombres de los dominios a direcciones IP de la forma 200.1.2.3 (IPv4) ó 2001:1234:5::6706 (IPv6). Estas direcciones son necesarias para saber en qué parte del Internet está el contenido buscado, ya que con sólo el nombre no hay forma de averiguar su ubicación. Pero, ¿Porqué es necesario el DNS?, ¿Porqué no simplemente se usan las direcciones IP?

En el principio Internet funcionaba sin DNS [10], pero el hecho de utilizar palabras (por ejemplo `www.uchile.cl`) en lugar de las direcciones IP, trajo consigo una mayor facilidad para los usuarios al momento de conectarse, y esta característica es uno de los motivos por los que se logró masificar el uso del Internet.

1.2. Organización del DNS

El protocolo DNS está organizado de manera jerárquica en forma de árbol (ver figura 1.1), es decir, si se escoge un dominio como `www.uchile.cl`, se puede ver que éste se divide en 3 partes separadas por un punto (`www`, `uchile` y `cl` en el ejemplo). Además existe una parte oculta que corresponde a un punto al final del dominio (`www.uchile.cl.`. El nombre

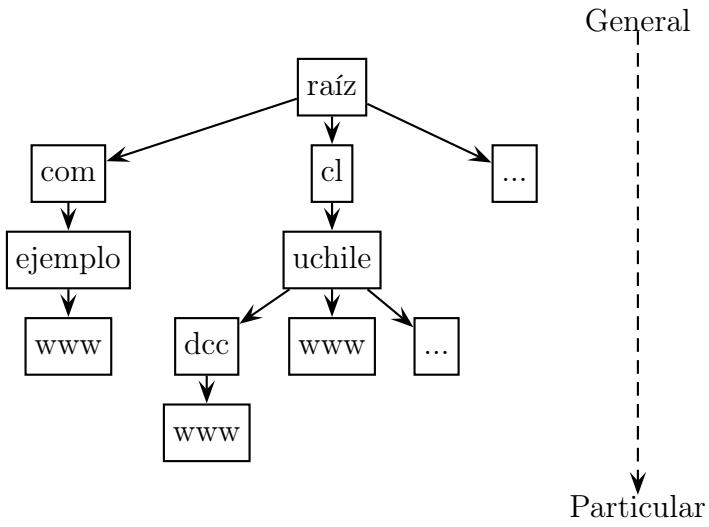


Figura 1.1: Diagrama de la organización del DNS

del dominio se ordena de lo más general a lo más particular de la siguiente forma: El punto a la derecha representa la raíz, que es el nodo más arriba en el árbol, luego si se avanza a la izquierda, hacie la parte más específica, se encuentran los llamados TLD o *Top Level Domains*, como el `.com`, `.org`, o el `.cl` en este caso. Bajo cada TLD se encuentran los dominios más específicos como `uchile.cl` en el caso del `.cl`. Estos dominios pueden tener bajo ellos otros sub-dominios como `www` en el caso de `www.ejemplo.com` o como `dcc` en el caso de `dcc.uchile.cl`.

1.3. Funcionamiento del DNS

Cuando se quiere encontrar la dirección de un dominio hay 2 partes que entran en juego: una parte son los servidores de nombres autoritativos y la otra los *resolvers*. Los *resolvers* son los servicios encargados de descubrir la dirección IP asociada un nombre de dominio, usando la información entregada por los servidores autoritativos, que son los que entregan información confiable para cada dominio sobre el que tienen autoridad.

Cuando un cliente quiere acceder al contenido de un dominio, envía una consulta al *resolver*, el que se encarga, de forma iterativa o recursiva, de traducir el nombre según corresponda (ver figura 1.1) partiendo por la raíz. El detalle paso a paso de la resolución del nombre `www.uchile.cl`(traducida y adaptada desde [8]) se muestra a continuación:

1. El cliente quiere saber la dirección de `www.uchile.cl`, la solicitud es enviada al resolver (que en general es el servidor de nombres que provee el ISP(*Internet Service Provider* o Proveedor de servicios de Internet. Ver figura 1.2). El resolver solicita el registro de tipo A, el cual representa una dirección IP. El servidor del ISP sabe que no es autoritativo para `uchile.cl`, por lo que no puede buscar en su propia zona. Lo que puede hacer es buscar en su caché de datos vistos recientemente donde podría estar el resultado

y entregarlo al cliente, si es así, la consulta concluye. Si no, el *resolver* lo busca en Internet como se explica a continuación.

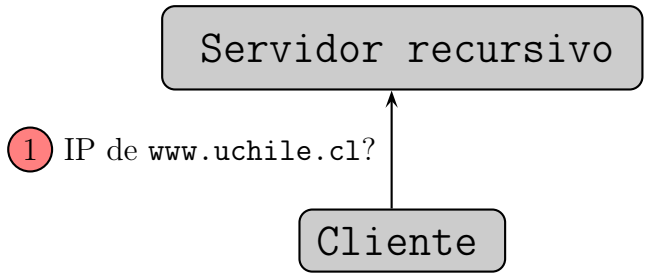


Figura 1.2: 1 Solicitud del cliente al ISP.

2 Todos los servidores de nombre recursivos están preconfigurados con una lista de 13 servidores raíz, una selección que se puede ver en la figura 1.3 El *resolver* elige una al azar y envía una solicitud del récord A de `www.uchile.cl`.

```
A.ROOT-SERVERS.NET.   IN  A  198.41.0.4
B.ROOT-SERVERS.NET.   IN  A  192.228.79.201
C.ROOT-SERVERS.NET.   IN  A  192.33.4.12
...
M.ROOT-SERVERS.NET.   IN  A  202.12.27.33
```

Figura 1.3: Lista de los servidores raíz preconfigurada.

3 El servidor raíz no sabe nada de `uchile.cl`, pero envía el camino a los servidores del CCTLD(*Country Code Top Level Domain*) responsable del dominio `.cl` (ver figura 1.5). Esta es la forma en que los servidores responden que no saben: "Ve a preguntarle a los que están en esta lista"(ver figura 1.4). Aunque técnicamente sólo se preguntó por los registros NS, los servidores raíz también entregan las direcciones IP de cada uno de los registros: Esto se conoce como "*glue records*" ("registros pegados" en español) y lo proveen para ahorrarle al cliente el tiempo de consultarlo.

```
/* Authority section */
CL.                                     IN   NS  A.NIC.CL
                                        IN   NS  B.NIC.CL
                                        IN   NS  C.NIC.CL.
                                        ...
                                        IN   NS  SNS-PB.ISC.ORG.
/* Additional section - glue records */
A.NIC.CL.                               IN   A   200.1.121.10
B.NIC.CL.                               IN   A   200.7.4.7
C.NIC.CL.                               IN   A   200.16.112.16
...
SNS-PB.ISC.ORG.                         IN   A   192.5.4.1
```

Figura 1.4: Respuesta de los servidores raíz cuando se consulta por `uchile.cl`.

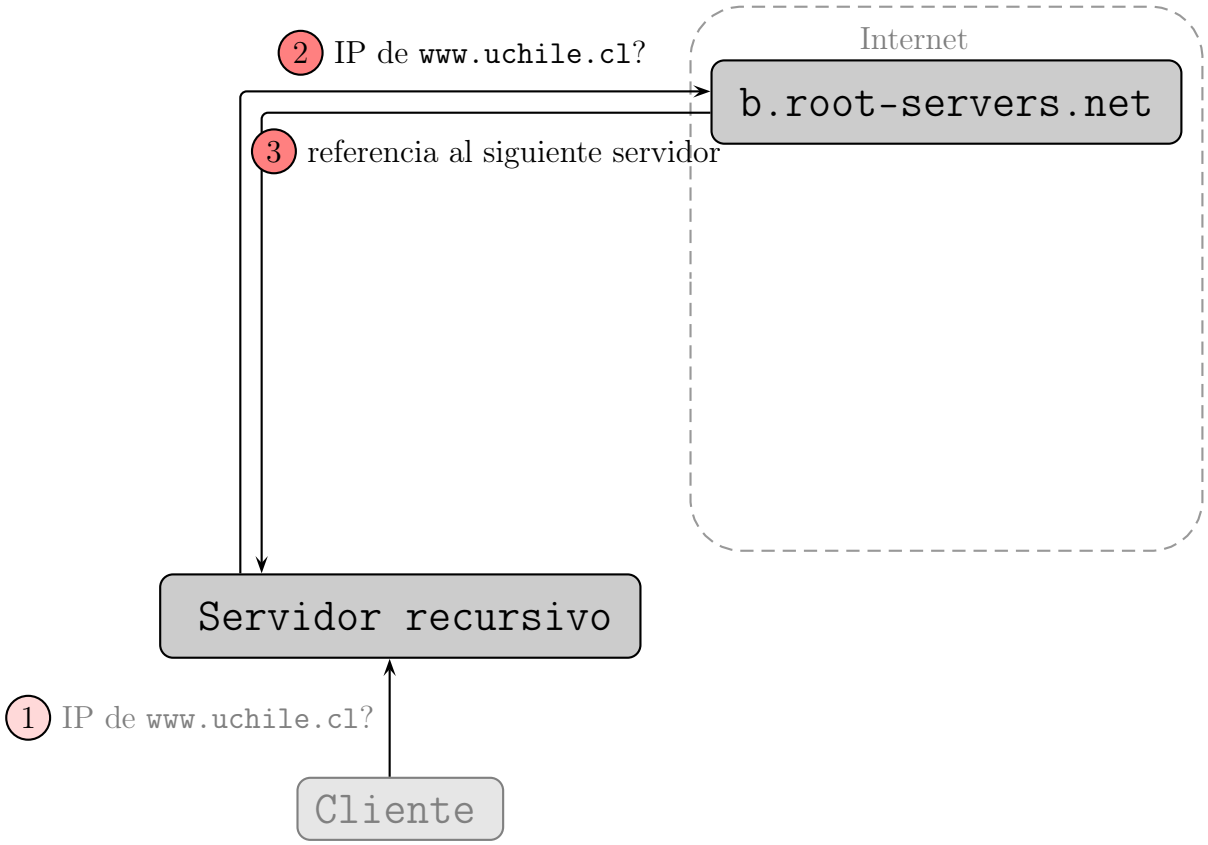


Figura 1.5: 2 Pregunta a servidores raíz.
3 Respuesta con referencias de `.cl`.

4 Con la referencia entregada por los servidores raíz, el resolver elige a uno de los servidores autoritativos al azar, por ejemplo `c.nic.cl` y le envía la misma consulta: "¿Cuál es el registro de tipo A para `www.uchile.cl`?" (ver este paso en la figura 1.7).

```
/* Authority section */
uchile.cl.                              IN  NS  ns4.uchile.cl.
                                        IN  NS  ns2.uchile.cl.
                                        IN  NS  ns1.uchile.cl.
/* Additional section - glue records */
ns4.uchile.cl                           IN  A   146.83.185.209
ns2.uchile.cl                           IN  A   200.89.70.70
ns1.uchile.cl                           IN  A   200.89.70.3
```

Figura 1.6: Respuesta de los CCTLD a la consulta para encontrar `www.uchile.cl`.

5 El servidor CCTLD no sabe cuál es la respuesta de esa consulta, pero sabe algo que puede acercarse. Tal como lo hicieron los servidores raíz, él envía una referencia a los servidores que probablemente saben la respuesta a la consulta (ver este paso en las figuras 1.7 y 1.6).

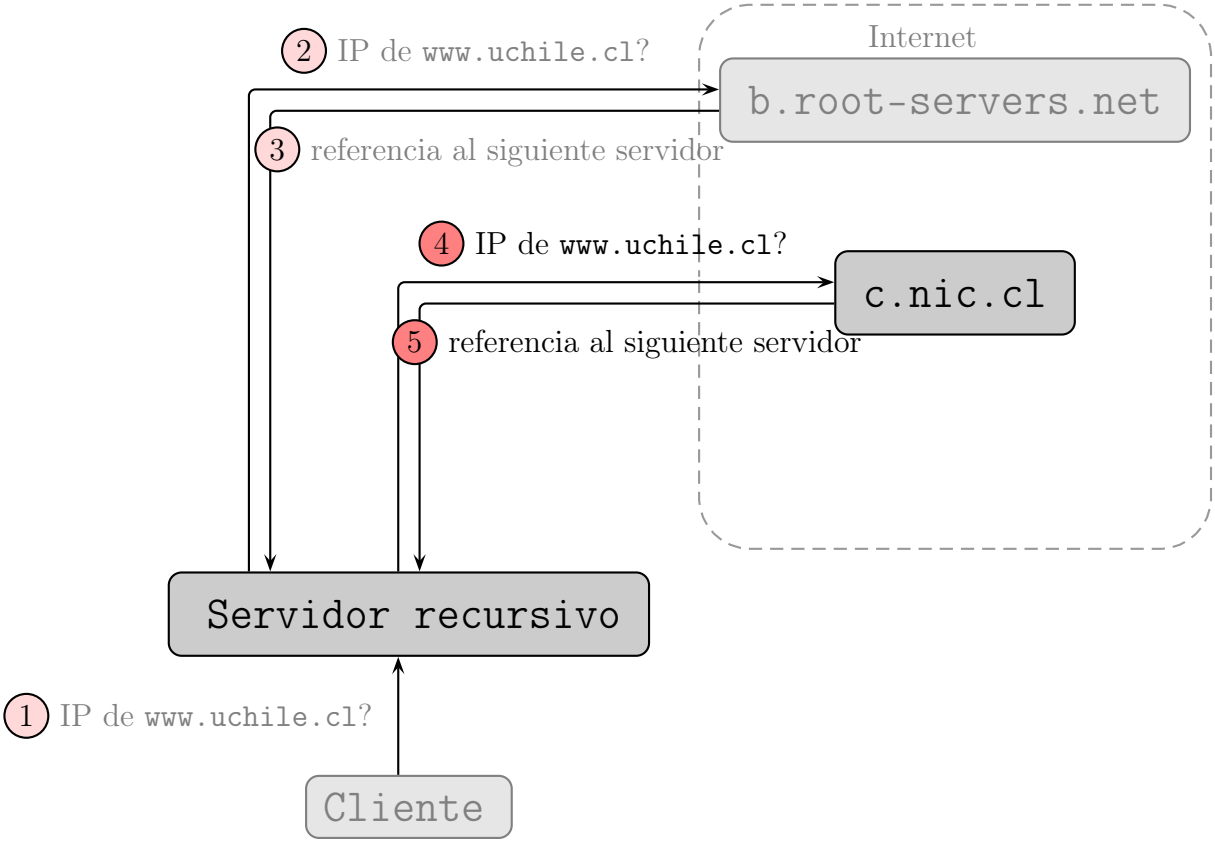


Figura 1.7: 4 Solicitud de `www.uchile.cl` al GTLD.
5 Respuesta con referencias a `uchile.cl`

6 Una vez más el servidor de nombres recursivo está siguiente una cadena de referencias para satisfacer al cliente, y elige al azar uno de los servidores de nombre recibidos y envía una tercera consulta (igual a las dos anteriores).

7 A diferencia de las otras respuestas, que sólo pasaban la responsabilidad a otro servidor de nombres, éste tiene la respuesta de la consulta enviada, y provee el registro de tipo A para `www.uchile.cl` (ver pasos 6 y 7 en la figura 1.8). Adicionalmente, indica que esta respuesta viene de un servidor autoritativo de `www.uchile.cl`, una fuente de confianza para ese dominio.

8 Ahora, con una respuesta en mano, el servidor recursivo del ISP entrega la respuesta al cliente, y satisface la consulta completa. Además el servidor recursivo guarda esta respuesta en su caché, en caso de que alguien le vuelva a realizar la misma consulta (ver paso 8 en la figura 1.8).

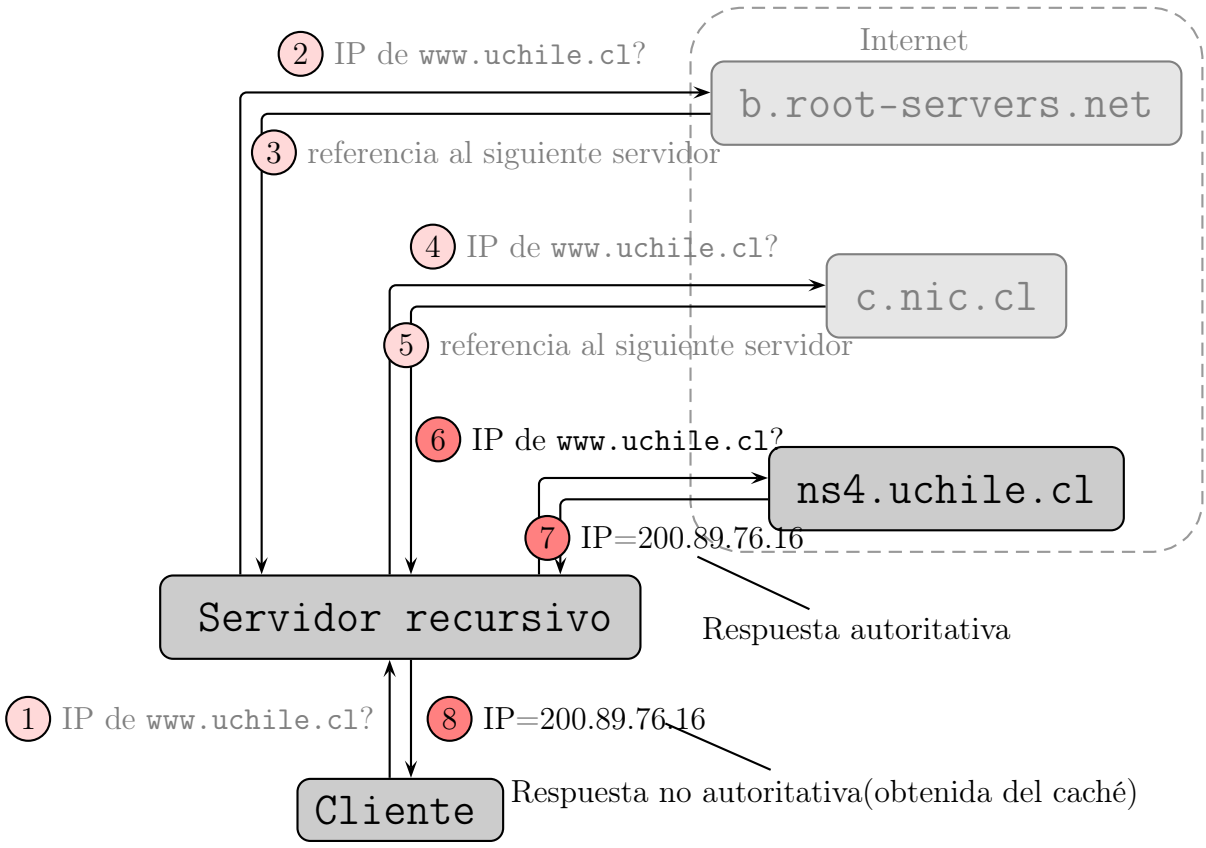


Figura 1.8: 6 Solicitud a `ns4.uchile.cl`.
7 Respuesta autoritativa del registro A de `www.uchile.cl`.
8 Respuesta no autoritativa del *resolver* al cliente.

Cabe notar que la respuesta al cliente no incluye el indicador de autoridad. Aunque él recibió la respuesta de un servidor autoritativo, no puede decirle al cliente que él es una fuente de autoridad para ese dominio, por lo que la respuesta es considerada no autoritativa.

1.4. Dominios bajo el `.cl`

En 1997 nació NIC Chile, organización encargada de administrar el registro de nombres del dominio `.cl` y de operar la tecnología que permite que estos nombres funcionen de manera eficiente y segura.

Desde que los dominios `.cl` existen, se ha notado un gran crecimiento con los años de la cantidad de dominios registrados bajo este TLD, llegando a más de 500 mil dominios en el año 2016 (ver figura 1.9).

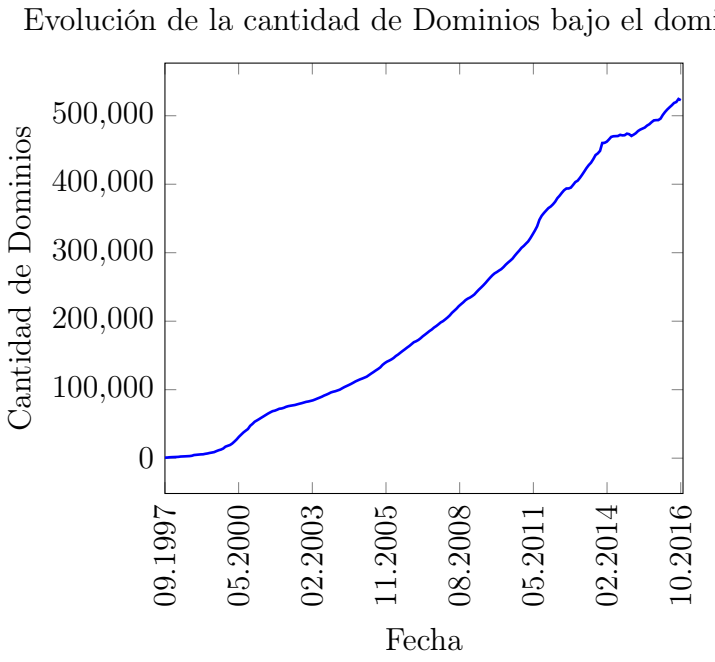


Figura 1.9: Crecimiento de la cantidad de dominios bajo el `.cl`

Además NIC Chile mantiene una red de servidores de nombre para el "*top level domain*" `.cl` en todo el mundo, con el fin de brindar un servicio robusto, estable, y con excelentes tiempos de respuesta. Contando con tres nubes *anycast*[1] bajo control directo de NIC Chile con 23 nodos repartidos en todo el mundo(ver figura 1.10). Además de contar con servicios de respaldo brindados por la *Internet Systems Consortium*(ISC), *Netnod* y *Packet Clearing House* (PCH).

[1]La tecnología *anycast* permite que la consulta por un nombre de dominio sea respondida por el servidor de nombres activo más çercano.ªl cliente, de acuerdo a un cálculo llevado a cabo en forma automática.

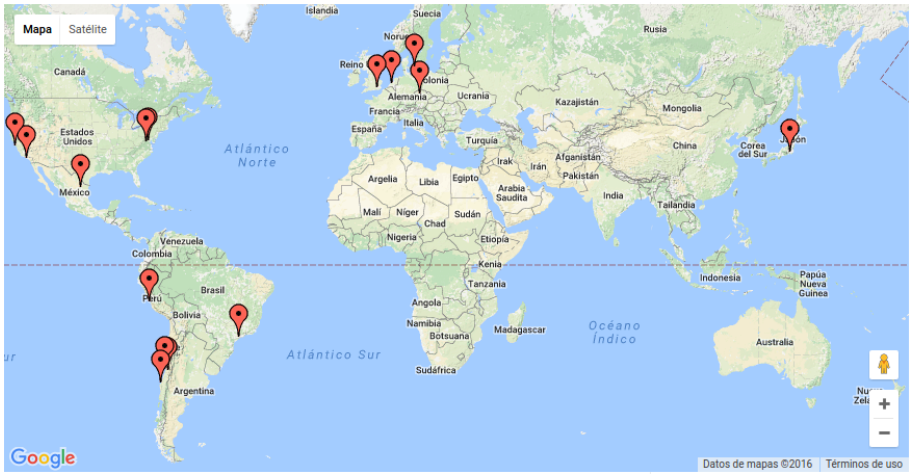


Figura 1.10: Mapa de *Nameservers* del `.cl` brindados por NIC Chile

1.5. Robustez del servicio DNS

Tener una buena configuración DNS para un dominio es algo muy importante que todo administrador DNS sabe. Sin embargo muchas veces se establece una configuración que cumple sólo lo básico para funcionar en situaciones normales.

El DNS está principalmente basado en la buena fe, pero eso llevó a usuarios maliciosos a querer apoderarse de cierta información, un ejemplo de los abusos que hacen a ciertas vulnerabilidades es el *DNS cache poisoning*, que corresponde a la redirección maliciosa de un nombre de dominio a una IP incorrecta. Peor aún es el fallo *Kaminsky*[8], conocido así por su descubridor Dan Kaminsky, que corresponde a la redirección de un servidor de nombre autoritativo malicioso, lo que infecta toda la zona y no sólo una dirección. Por esos y otros fallos de distinta naturaleza, como la caída de todos los servidores de nombre de un dominio dado en caso de terremoto u otra catástrofe, es necesario que los servidores DNS de los dominios se configuren no sólo para que funcionen sino para que también resistan fallos.

1.5.1. Recomendaciones para mejorar el sistema DNS

Gran parte de los problemas que se dan en las redes de comunicaciones son debido a que los dominios no tienen implementadas las recomendaciones dadas por las entidades del Internet, como IETF, ISOC, etc. y expertos. Para medir la resiliencia, se puede partir por averiguar si se están cumpliendo esos estándares, verificar dónde se está fallando, y hacer saber a la comunidad del Internet qué problemas deben solucionar.

Las recomendaciones de pueden separar en 3 grupos:

Mejora de dispersión de los servidores de nombre: En cuanto a la dispersión se puede mejorar contando con una cantidad mínima de direcciones IP por servidores y por

dominios, además se puede agregar robustez utilizando dispersión geográfica y además utilizando distintos protocolos de IP cono IPv4 e IPv6.

Implementación de DNSSEC para la zona: DNSSEC corresponde a extensiones de seguridad de DNS que permiten hacer el DNS más seguro, una implementación correcta y completa hará el sistema más robusto.

Implementación de recomendaciones de los servidores de nombre: Existen varias recomendaciones básicas para evitar ciertas vulnerabilidades, por ejemplo la utilización correcta de DNSSEC, permitir tráfico mediante protocolo TCP, prohibir la transferencia de la zona a usuarios no autorizados, y prohibir la recursividad para evitar el envenenamiento de caché o el fallo Kaminsky.

Descripciones más detalladas de estas recomendaciones se encuentran a lo largo del capítulo 2.

1.5.2. Justificación de este trabajo

Teniendo en cuenta que existen muchas recomendaciones para mejorar la robustez del sistema DNS y que no existe un buena forma de saber el estado actual para cualquier grupo de dominios, se debe implementar una herramienta que permita saber qué tan capaz es el Internet el Chile para resistir y recuperarse ante éstos y otros fallos relacionados con el DNS, lo que se conoce como resiliencia del Internet en términos de DNS, para un grupo determinado de dominios, como por ejemplo los dominios del `.cl`. Para eso, la herramienta recolectará los datos necesarios con tal de verificar el cumplimiento de los mecanismos de robustez recomendados.

Capítulo 2

Marco Teórico

La resiliencia se define como la capacidad de un sistema de soportar y recuperarse ante fallas de cierta índole. En el marco de este trabajo se entiende por resiliencia de un sistema DNS la propiedad de un sistema o servicio para soportar fallas relacionadas con la correcta traducción de un nombre de dominio a una dirección IP y así garantizar el correcto funcionamiento de todos los subsistemas asociados.

A continuación se describen algunos conceptos para ayudar a comprender mejor el problema y la solución propuesta más adelante.

2.1. RFCs: estándares y recomendaciones

Existe una comunidad internacional abierta llamada IETF o *Internet Engineering Task Force*, comunidad está compuesta por diseñadores de redes, operadores, vendedores e investigadores preocupados de la evolución de la arquitectura y el correcto funcionamiento del Internet. Una de las tareas del IETF es definir diferentes documentos llamados RFC o *Request For Comments*, que son los documentos que definen los estándares del Internet. Los RFC referenciados en este estudio se encuentran listados en la bibliografía y están a libre disposición en la página web del IETF.

2.2. Detalles del funcionamiento de DNS

Para comprender mejor cómo opera el DNS es necesario entender quiénes actúan y cuáles son sus roles.

Zona: Es lo que en general se entiende por "dominio": un conjunto de pares de nombres-direcciones IP, por ejemplo "{(`www.uhile.cl, 172.29.1.16`), (`dcc.uhile.cl, 192.80.24.4`), ... }".

Servidor de Nombres: Es un servidor con software específico que responde a consultas DNS tales como "¿Cuál es la IP de www.uchile.cl?". Los servidores pueden saber la respuesta directamente (si es un servidor autoritativo para la zona consultada), o pueden ir a Internet a consultar a otros servidores por la información solicitada (si es un servidor recursivo).

Servidor Autoritativo: Para cada zona tiene que existir una persona que mantenga los archivos con las asociaciones de nombre-IP (`www.uchile.cl` es 200.89.76.16 por ejemplo). Este archivo en general es administrado por un humano, y en la mayoría de los casos una máquina almacena este archivo, el cual es conocido como zona maestra. Cuando una zona tiene muchos servidores de nombre este archivo de zona debe ser transferido a cada uno de ellos.

***Resolver*:** Esta es la parte del sistema DNS tanto del cliente como de los servidores que realiza las consultas. El *resolver* es una pequeña librería compilada en cada programa que requiere servicios DNS y sabe cómo enviar consultas a otros servidores.

Servidor de nombres recursivo: Este es un servidor de nombres que está dispuesto a ir al Internet a averiguar la información solicitada de una zona de la cual no es autoritativo, como un servicio para sus clientes. No todos los servidores de nombre están configurados para ofrecer un servicio recursivo, pero en general los servidores de nombre que ofrecen los ISP (del inglés *Internet Service Provider* o Proveedor de Servicios de Internet) proveen un servicio recursivo a sus clientes.

Registros de recursos: Aunque hasta ahora se ha visto que el DNS sirve sólo para resolver consultas del tipo "Dame la IP del nombre `www.uchile.cl`", existen otros tipos de consultas que se le pueden realizar a un servidor de nombres, y esto destaca la noción de que el DNS es realmente una base de datos de "registros de recursos". El tipo más común es una dirección IPv4 (un registro de tipo "A"), pero existen otros registros como NS (del inglés *Nameserver*), MX (del inglés *Mail Exchanger*), SOA (*Start of Authority*), y muchos otros.

Delegación: Cuando un servidor de nombres no tiene los contenidos de una zona, pero sabe cómo encontrarlo, se dice que delega el servicio de esa zona a otro servidor de nombres.

2.2.1. Registros de Recursos

Los Registros de Recursos o RRs (por sus siglas del inglés "Resource Record") son la unidad básica de transferencia de información del DNS. Todos los RRs tiene el mismo formato superior mostrado en la figura 2.1, donde se ven seis campos claramente separados.

Donde:

NAME: El nombre del dueño o nodo al que pertenece este registro de recurso.

TYPE: Dos *bytes* que contienen uno de los códigos de los tipos de RR.

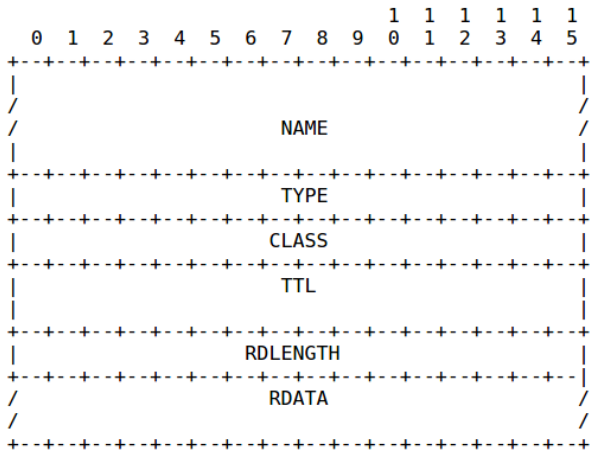


Figura 2.1: Formato de los recursos de registro

CLASS: Dos *bytes* que contienen uno de los códigos de las clases de RR.

TTL: Entero con signo de 32 bits de largo que especifica el intervalo de tiempo que el registro de recurso puede ser guardado en el caché antes de que la fuente de información tenga que volver a ser consultada. Cuando es cero, significa que el RR puede ser usado sólo por la transacción en progreso y no deberá ser guardado en el caché. También puede ser usado con valor cero para datos extremadamente volátiles.

RDLENGTH: Entero sin signo de 16 bits de largo que especifica la cantidad de octetos de el campo RDATA.

RDATA: Un *string* de octetos de largo variable que describe el recurso. El formato de esta información varía de acuerdo al tipo y a la clase del registro de recurso.

La clase del RR puede pertenecer a cualquiera de las 4 especificadas en el RFC1035 [11], ya sea la clase IN, CS, CH o HS. Sin embargo para los estudios realizados esto no es relevante y se asumirá que siempre se utiliza la clase IN que se refiere a Internet.

Además existen varios tipos de Registros y cada tipo provee información diferente utilizada para describir los elementos de la zona. A continuación se muestran algunos de los registros utilizados en el estudio con sus funciones.

A: Identifica una dirección IPv4 de 32 bits.

AAAA: Identifica una dirección IPv6 de 128 bits.

NS: Identifica un servidor de nombres autoritativo.

DNSKEY: Llave utilizada en DNSSEC.

NSEC: Probar la no existencia de un nombre con DNSSEC.

NSEC3: Probar la no existencia de un nombre (v3) con DNSSEC.

DS: Identifica la llave DNSKEY de una zona delegada.

RRSIG: Firma para un conjunto de registros asegurados con DNSSEC.

SOA: Especifica información autoritativa para la zona DNS.

2.3. Vulnerabilidades de DNS

En general, es recomendado mantener los servicios recursivos separados de los autoritativos [5], para evitar ataques de amplificación y de *cache poisoning*. Dicho ataque consiste en alterar la información entregada por el DNS mediante respuestas inválidas, pues un servidor recursivo averigua y memoriza las respuestas que recibe, las que podrían ser incorrectas.

2.3.1. Envenenamiento de Caché

El envenenamiento de caché o *Cache Poisoning* ocurre cuando un usuario malicioso logra inyectar información falsa al caché de un servidor recursivo, causando que este entregue esa información falsa a clientes que confían en él.

La forma de envenenar el caché no es tan simple como sólo enviar paquetes DNS aleatorios a un servidor de nombres. Un DNS acepta solamente respuestas a consultas pendientes; las respuestas inesperadas son ignoradas.

Lo que hacen los servidores de nombres para saber si están esperando un paquete de respuesta es revisar las siguientes condiciones:

- La respuesta llega por el mismo número de puerto UDP por el que fue enviada. Si no, es desechada.
- La sección *Question* que está duplicada en la respuesta es idéntica a la de la consulta pendiente.
- El número de identificación de la consulta (*Query ID*) coincide en la respuesta recibida.
- Las secciones *Authority* y *Additional* representan nombres que están en el mismo dominio que la consulta.

Si todas estas condiciones se satisfacen, un servidor de nombres acepta el paquete como respuesta genuina a la consulta, y utiliza su contenido. Esto incluye guardar las respuestas en el caché, además de la parte de autoridad y la adicional. Si un usuario malicioso logra predecir y generar un paquete DNS de respuesta que cumpla dichas condiciones, entonces puede engañar al receptor del paquete y entregar información maliciosa de forma transparente.

Lo que normalmente se hace para lograr envenenar el caché, es primero escoger una víctima

encontrando un servidor vulnerable[1] al envenenamiento (siendo víctimas también todos los usuarios que usen el servicio de ese servidor de nombres).

Luego hay que encontrar un objetivo, un dominio al cual suplantar para robar información. Lo que el usuario malicioso intenta con ésto es engañar a las víctimas para que visiten una página web maliciosa en lugar de la original, haciéndoles por ejemplo visitar la página `www.banco.cl`, la cual se resuelve a una dirección IP maliciosa.

Una vez que se tienen escogidas las víctimas (el servidor vulnerable y el objetivo al que se quiere atacar), sólo basta envenenar el caché, pero como los paquetes inesperados son simplemente ignorados, hay que buscar una forma de que el servidor espere los paquetes maliciosos que van a envenenar el caché. Lo que suena complicado en realidad es muy simple, sólo basta con que un cliente malicioso le envíe una consulta al servidor vulnerable por el dominio que fue escogido como objetivo (`www.banco.cl`), luego ese servidor como es recursivo comenzará a averiguar por sus medios la respuesta a dicha consulta, pero simultáneamente el servidor malicioso comenzará a bombardear con respuestas maliciosas probables, esperando que alguna de ellas cumpla con las condiciones mencionadas anteriormente, lo que podría ser de baja complejidad[2] si se predice correctamente el *Query ID* y el puerto mediante el cual fue enviada la solicitud, ya que el resto es conocido.

Esto no es *phishing*

Aunque el envenenamiento de caché tiene resultados similares al *phishing* (hacer creer a un usuario que un sitio malicioso es genuino), no son lo mismo.

Con *phishing*, una URL falsa es usada para referenciar a un servidor malicioso, pero la página web alojada en dicha dirección están disfrazada para verse muy parecido al sito original (utilizando el mismo HTML y CSS). Un usuario puede detectar este redireccionamiento incorrecto con sólo examinar la URL cuidadosamente.

Con el envenenamiento del cache DNS se logra romper la confianza que se tenía en el servicio DNS. La URL es idéntica a la original y el cliente no tiene cómo saber que está siendo direccionado a un servidor malicioso.

2.3.2. Fallo Kaminsky

Por lo que se ha visto, el envenenamiento de caché envenena sólo un registro, pero esto tiene menos impacto de lo esperado: "Es sólo un registro, y en general se requieren muchos

[1]Si un servidor de nombres no aleatoriza la elección del puerto a utilizar, ni aleatoriza la elección del ID a utilizar, se dice que es vulnerable.

[2]El campo del *Query ID* es un entero entre 0 y 65535. En muchos casos el número que se escoge para utilizar como identificador corresponde al de la consulta anterior más uno. Por ésto, "adivinar" el siguiente es tan simple como probar con cierta cantidad consultas con número de *Query ID* incrementados en uno hasta dar con el adecuado. Por otro lado, el puerto es un entero entre 0 y 65535. Predecir el puerto utilizado puede ser trivial si es que el servidor utiliza siempre el mismo puerto, lo que es común.

registros para operar correctamente, y es difícil *hackearlos* todos"[8].Dan Kaminsky encontró un enfoque que es mucho más efectivo que el anterior, basado en el mismo principio del envenenamiento del caché, pero con una impacto mucho más grande.

En lugar de envenenar un simple e insignificante registro de tipo A, Kaminsky descubrió que se puede ir un nivel más arriba y secuestrar la autoridad de los registros.

Antes de ejecutar el ataque es necesario que el usuario malicioso configure un servidor de nombres autoritativo para la zona de `banco.cl`, incluyendo todos los recursos necesarios: Registros A, MX, NS, etc.

Y una vez que se tiene el servidor de nombres funcionando se genera el envenenamiento: Un usuario malicioso pregunta por `textoalazar.banco.cl`. El servidor recursivo como no sabe la respuesta va a consultar a Internet. Siguiendo la lógica vista anteriormente le consulta primero al servidor raíz el cual al no saber guía al servidor recursivo a consultar al servidor de `.cl`, el servidor del `.cl` al no saber tampoco la respuesta guía al servidor recurivo a preguntar al servidor de nombres de `banco.cl`, pero simultáneamente el servidor malicioso envía una respuesta sobre con una dirección maliciosa del servidor de `banco.cl`, con lo que el servidor malicioso logra envenenar el cache del servidor recursivo, logrando envenenar por completo la zona de `banco.cl`.

Para ver más en detalle el funcionamiento del envenenamiento de caché y del Fallo Kaminsky ver [8].

2.3.3. No responder consultas

Cuando un servidor recibe una consulta tiene la opción de no responder si no conoce la respuesta. Sin embargo, ésto deja una puerta abierta a algún adversario que quiera entregar una respuesta maliciosa, por lo que es importante que un servidor siempre entregue respuestas, incluso si no conoce la respuesta a la consulta realizada. En este último caso se entrega una respuesta negativa.

2.4. DNSSEC: Extensiones de Seguridad de DNS

DNSSEC es una capa adicional a DNS que agrega mejoras de seguridad, siendo compatible con el sistema DNS. Las extensiones de seguridad están definidas en los RFC 4033[1], 4034[3] y 4035[2]. Para un correcto funcionamiento del DNS hay que considerar los aspectos descritos a continuación, tomando en cuenta siempre que el dueño de la zona es el encargado de entregar información verídica y vigente (entregar una respuesta más la prueba de autenticidad correspondiente) y el cliente es el encargado de verificar que la información entregada es correcta, siguiendo una cadena de confianza.DNSSEC se basa en una cadena de confianza, donde los servidores raíz son en quienes se confía siempre, ya que vienen incrustados con el software DNS y no suelen cambiar, y ellos mismos delegan esa confianza a sus hijos y así sucesivamente. Cuando se consulta por un dominio, es necesario verificar la existencia del

registro DS en cada paso de la cadena, ya que este registro cumple la función de probar que un dominio confía en su hijo. Por lo que para confiar en el dominio buscado hay que recorrer toda la cadena autenticando cada paso.

Una vez que se llega al dominio deseado existen dos tipos de respuestas básicas que se pueden tener ante la consulta. Esto se explica a continuación.

2.4.1. Prueba de existencia

Cuando un servidor de nombre recibe una consulta asociada a un dominio existente, con las extensiones de DNSSEC, la respuesta que el servidor entrega es la respuesta que el cliente busca, lo que corresponde al registro asociado a la consulta, además DNSSEC entrega un registro firmado asociada a la consulta llamado RRSIG. dicho registro se firma con una llave DNSKEY de la zona.

2.4.2. Negación de existencia

En el caso en que el servidor de nombres reciba una consulta preguntando por un dominio inexistente, con las extensiones de seguridad, la respuesta puede ser de 2 formas[3]: utilizando los registros NSEC o NSEC3.

La forma en que esto se prueba es primero ordenando la zona del dominio de forma la forma canónica definida en el RFC 4034 [3]. Luego por cada par de subdominios vecinos en la zona se crea una firma que prueba que no hay nada entremedio de esos dos dominios y se cierra el ciclo con una firma con el último y el primer dominio (para probar que no existe nada más después del último dominio). Luego, cuando se hace una consulta por un dominio inexistente, la respuesta debe contener las firmas que prueban que, en la posición donde debería estar el dominio, según el orden canónico, no hay nada. Y además entrega la firma que muestra que no existe un *wildcard*[4].

El comportamiento anterior es correcto para NSEC, pero para NSEC3 hay un pequeño cambio, donde en lugar de utilizar los nombres de los dominios en texto plano, se utiliza los nombres *hasheados*, para intentar evitar la enumeración de la zona o *zone walking*[5].

Para mayor información ver RFC 7129[9], donde se explica en más detalle el funcionamiento de la negación autenticada de existencia.

[3]Existen también los *wildcards* o comodines que, a cualquier dominio que cumpla la expresión regular le entregan una respuesta, aunque el nombre especificado no exista.

[4]El *wildcard* es el dominio comodín representado por el * como `*.dominio.cl` que, para cada dominio que calce en esa expresión regular, entrega el resultado asociado.

[5]Con NSEC3 más que evitar la enumeración de zona sólo lo dificulta porque ya existen formas de enumerar una zona con NSEC3.

2.4.3. EDNS

Para que un dominio soporte DNSSEC los servidores de nombre también deben soportarlo.

Para verificar si un servidor de nombre soporta DNSSEC, éste debe tener activado el *flag* del mecanismo EDNS en su versión 0 que es la actual.

2.4.4. Necesidad de TCP en servidores de nombre con DNSSEC

Con la llegada de DNSSEC se solucionaron varios problemas de seguridad del DNS, pero para poder mejorar ésto se tuvo que introducir una enorme sobrecarga que es el tamaño de los paquetes. Las respuesta DNSSEC firmadas son mucho más grandes y generalmente superan el tamaño por defecto UDP de 512 bytes, por lo que existe la necesidad de utilizar TCP para la transmisión de éstos paquetes.

Por otro lado, para muchos dominios es importante mantener su archivo de zona privado, principalmente para evitar dar información innecesaria que podría hacerlos blanco de algún ataque cibernético. Un mecanismo común de defensa, para evitar la transferencia de zona, es bloquear la transmisión mediante TCP (ya que antes de la llegada de DNSSEC, los únicos paquetes lo suficientemente grandes para necesitar el uso de TCP eran los de transferencia de zona).

Tomando en cuenta estos dos aspectos, existen muchos servidores que utilizan el mecanismo de bloqueo de transmisión mediante TCP, lo que hace a estos servidores incompatibles con el uso de DNSSEC.

2.5. Direcciones IPv4 e IPv6

En los RFCs 2460 [6] y 791 [12] se definen los protocolos de Internet en sus versiones IPv4 e IPv6 respectivamente.

Estos protocolos son los que se utilizan para enviar información entre distintos dispositivos conectados a la Internet. El protocolo IPv4 asigna una serie de cuatro números (entre 0 y 255 cada uno) a cada dispositivo, así dándoles una dirección que los identifica como únicos en la red. IPv4 sólo permite 4.294.967.296 direcciones. Tomando en cuenta que en al año 2015 se estimaron un total de 16,3 mil millones de dispositivos[6], este espacio resulta ser muy pequeño. Así nace IPv6, una nueva versión del protocolo que amplía el número de direcciones a 340 sextillones de direcciones, lo que es prácticamente ilimitado.

En términos de resiliencia, la diversidad tanto en cantidad como en tipo puede ser una ventaja, ya que si una IP no responde se puede utilizar otra. Por otro lado, diferenciar entre

[6]Ver mas detalle en *VNI Complete Forecast Highlights Tool* `http://www.cisco.com/c/m/en_us/solutions/service-provider/vni-forecast-highlights.html`.

las versiones (4 ó 6) es una ventaja, ya que si por algún motivo la comunicación a través de IPv4 falla, se tendría un respaldo con IPv6. Además, la recopilación de información del tipo de IP que se está usando sirve para analizar y estudiar los niveles de avance de este protocolo a nivel país.

2.6. Métricas a utilizar

2.6.1. Dispersión de servidores de nombre

Para que haya un mayor nivel de resiliencia, es necesario tener los servidores de nombre del dominio dispersos en diferentes niveles, ya sea cuantitativamente (alta cantidad de servidores de nombre), geográficamente (en diferentes países o regiones) o topológicamente (en diferentes sistemas autónomos).

Dispersión de cantidad de Servidores

La documentación definida en el RFC 1912 [4] indica que dos servidores de nombre es el mínimo recomendado para un buen funcionamiento. Sin embargo, a esta altura de Internet se considera que 2 es un número muy bajo, siendo 3 el mínimo recomendado (ver RFC 2181 [7]). El valor depende del uso que se le dará al dominio.

Dispersión Geográfica

Para la dispersión geográfica no hay un estándar establecido, pero en este estudio se considera que cada dominio debe tener servidores de nombre en al menos dos países diferentes para considerarse resiliente.

Dispersión a nivel de sistema autónomo

En cuanto a la dispersión de cantidad de sistemas autónomos en los que se deban encontrar los servidores de nombres no hay estándar ni recomendación, pero en los propósitos de este estudio, se considera que cada dominio debe tener sus servidores de nombres en al menos 2 sistemas autónomos diferentes para considerarse resiliente.

2.6.2. Resiliencia IP

En el año 2011 IANA, la entidad que supervisaba[7] la asignación global de direcciones IP entre otras tareas, entregó el último[8] bloque de direcciones IP de la versión 4, por lo que es importante saber cómo va evolucionando y si los usuarios se están adaptando a esta escasez utilizando el nuevo estándar IPv6.

Por otro lado, el hecho de tener direcciones IP en las 2 versiones puede añadir cierto grado de resiliencia, ya que da redundancia en caso de que alguno de los protocolos, ya sea en versión 4 o en versión 6, falle.

Dicho esto, como estándar de resiliencia en este estudio se tomará que cada dominio debe tener al menos un servidor con dirección en la versión de IPv4 y al menos un servidor con dirección en la versión IPv6 para considerarse resiliente en este aspecto.

2.6.3. Resiliencia DNSSEC del dominio

Para que un dominio se considere resiliente en cuando al estado de DNSSEC deberá cumplir los siguientes requisitos:

Verificación del dominio superior (DS)

Debe haber un registro del tipo DS asociado al dominio el cual se usa para verificar la llave.

Existencia DNSKEY

Debe tener al menos un registro de tipo DNSKEY, el cual será utilizado para verificar las firmas.

Prueba de existencia

Debe entregar ante cada consulta de un subdominio existente su firma asociada.

[7] IANA delegó definitivamente sus funciones a ICANN este año, ver reportaje "Histórica reunión de ICANN en Marrakech" de Patricio Poblete en la revista Bits no. 14 para mayor información en `https://www.dcc.uchile.cl/bits-de-ciencia`.

[8] Agotamiento de las direcciones IPv4: `https://www.nic.cl/anuncios/20110203-ipv4.html`.

Prueba de no existencia

Debe entregar una prueba de no existencia ante cada consulta de un subdominio no existente, esta prueba puede ser utilizando registros NSEC o NSEC3 siendo ambos resilientes.

Correcta verificación con RRSIGs

Cada uno de los requisitos anteriores deben ser verificados con su registro de firma RRSIG correspondiente.

2.6.4. Resiliencia de Servidores de nombre Autoritativos

Los requisitos que se nombran a continuación son requisitos sobre cada servidor de nombre, y no sobre cada dominio.

Permitir consultas mediante TCP

Debido a los explicado en la sección 2.4 en necesario que cada servidor permita consultas mediante TCP para un buen funcionamiento de DNSSEC, por lo tanto un servidor de nombre se considerará resiliente en este aspecto sólo si permite consultas a utilizando el protocolo TCP, tal como lo indican recomendaciones de CISCO y RIPENCC.

Tener habilitado EDNS

Además para un correcto funcionamiento de DNSSEC, el servidor debe tener habilitado EDNS, por lo que un servidor será resiliente en este aspecto sólo si tiene EDNS habilitado.

No permitir recursividad

Para evitar problemas como el envenenamiento de caché o el fallo Kaminsky, es necesario que los servidores de nombre autoritativos no permitan recursividad, y así el servidor será resiliente en este aspecto.

No permitir transferencia de la zona

Para evitar que usuarios maliciosos puedan obtener información sensible del dominio es necesario que los servidores autoritativos no permitan la transferencia de la zona para poder ser resilientes.

Siempre responder a las consultas

Es necesario que los servidores de nombre siempre contesten, aunque la respuesta no tenga contenido útil, para evitar que servidores maliciosos entreguen respuestas incorrectas. Para verificar esto, se realizará una consulta por un tipo poco utilizado como lo es el registro de tipo LOC. Si el servidor entrega una respuesta entonces se considerará resiliente.

Cabe destacar que para que un dominio sea resiliente con respecto a las características indicadas anteriormente, es necesario que todos los servidores de nombre que este dominio tenga registrado cumplan con lo indicado, es decir, que si sólo uno de sus servidores de nombre falla entonces el dominio no es resiliente.

Capítulo 3

Especificación del problema

A nivel de DNS existen varios puntos de falla asociados a distintas vulnerabilidades existentes. Para detectar si cierto dominio es susceptible a dichas fallas es necesario recolectar datos.

Para encontrar estos puntos de falla y ver el estado del DNS en Chile es necesario construir un software que sea capaz de recopilar datos de la resiliencia del Internet a nivel DNS, con el fin de generar un reporte sobre datos relevantes para así poder entregar esta información a la comunidad.

Algunos de los problemas asociados al DNS se muestran en la sección siguiente.

3.1. Problema existentes en el DNS

3.1.1. Baja dispersión de servidores de nombre

Cuando se habla de localización en términos de DNS hay varios niveles, por ejemplo la localización geográfica de un servidor de nombre, o la ubicación topológica de éste(redes IP o redes de sistemas autónomos). Tener una baja dispersión en alguno de estos niveles podría llevar a una caída completa del dominio, por lo que es importante mantener una alta dispersión tanto a nivel topológico como geográfico, sobre todo en un país sísmico como lo es Chile.

3.1.2. Isla de seguridad

Si bien un dominio puede estar cubierto por todo lo que es DNSSEC, es necesario que exista una cadena de confianza hasta un nodo raíz para recién poder confiar en que la información obtenida es verdadera, por lo que, para que cada dominio pueda contar con DNSSEC, también su dominio padre debe contar con DNSSEC.

En el caso del `.cl`, desde el año 2011 NIC Chile acogió la nueva tecnología DNSSEC y puede recibir las llaves criptográficas de los clientes que hayan activado DNSSEC en sus zonas bajo `.cl`, permitiendo a los dominios bajo el `.cl` el uso de estas extensiones de manera correcta.

3.1.3. Consultas sin respuesta

Como DNS está basado en la buena fe, existen usuarios maliciosos que podrían dar mal uso al hecho de que un servidor no entregue respuesta a ciertas consultas. Para evitar que alguien reemplace la respuesta (o la no respuesta) de un servidor por otra que no es verdadera, es necesario que el servidor tenga respuesta para cada una de las consultas que se le realizan, siendo una respuesta válida un error por ejemplo.

3.1.4. Envenenamiento de caché

Cuando un cliente quiere saber la dirección IP de un dominio manda una solicitud a su *resolver* con la consulta deseada y el *resolver* le devuelve una respuesta con la información que él resuelva(ver sección 1.3). Cuando un usuario malicioso quiere dirigir a los clientes que quieren acceder a un dominio a una dirección maliciosa lo que hacen es envenenar el caché con información falsa, en particular con direcciones IP maliciosas asociadas a dominios existentes o a sus servidores de nombre, lo que se conoce como Fallo Kaminsky, envenenando una zona completa y no solamente un dominio(Ver más detalles del envenenamiento del caché y del Fallo Kaminsky en la sección 2.3).

3.2. Desafíos

Desde inicios del año 2016, en NICLabs, se está desarrollando una herramienta llamada Observatorio, escrita en lenguaje *Python*, que permite realizar la obtención de datos relevantes para eventualmente generar un reporte, pero no fue probada con grandes volúmenes de datos. La aplicación fue hecha en módulos para cada distinta métrica, de forma que sea fácil de modificar, quitar o agregar alguno nuevo. Sin embargo, una de las falencias que posee es su secuencialidad, por lo que obtener todos los datos necesarios podría ser muy lento. Este trabajo de título se enmarca dentro de este proyecto.

Debido a la secuencialidad del software, la obtención de datos toma, para un dominio dado, un tiempo de entre 5 y 35 segundos, y realiza todas las consultas de manera lineal. Teniendo esto en cuenta, se plantearon los desafíos descritos a continuación.

3.2.1. Crear un sistema multi-hilos eficiente

Desde marzo del año 2016, la cantidad de dominios bajo el `.cl` llegó a una cifra de 500.000 dominios. Si se considera el tiempo que toma el software actual por cada dominio, entonces realizar la obtención de datos de forma lineal para todo el `.cl` tomaría alrededor de 28 días, según una estimación obtenida al extrapolar el resultado de un test realizado con 1600 dominios al azar.

La solución propuesta consiste en rediseñar la arquitectura del software y extenderlo para que funcione como un sistema paralelo real, de tal forma que realizar una análisis a muchos dominios sea resuelto en un tiempo del orden de horas en lugar de días.

Dado que el software anterior estaba escrito en `Python` y existe evidencia de que el *multithreading* de este lenguaje no es real, se migró el software a otro lenguaje que es capaz de efectuar consultas DNS, pero que además tiene un buen rendimiento usando múltiples hilos.

Con estas necesidades en mente, el lenguaje elegido es `GoLang`, que tiene un muy buen manejo de hilos y una muy buena librería de DNS versátil, que ofrece la posibilidad de generar consultas específicas muy fácilmente.

3.2.2. No sobrecargar

Cuando se quiere obtener información de los servidores autoritativos de los dominios, se tiene que realizar muchas consultas distintas, y dado que ahora las consultas se realizarán de manera más rápida, debido a la implementación de un sistema *multithreading*, podría haber una interferencia con el normal funcionamiento de los servidores. Por lo que se debió encontrar una buena forma de realizar estas consultas, para que los servidores no se sientan amenazados y no se produzca una denegación de servicios.

Además hay una serie de entidades a las que no le gusta recibir consultas no relacionadas a su uso normal, por lo que hay que revisar que las consultas a realizar no estén dentro de la lista negra previamente definida.

3.3. Trabajo Relacionado

DNSDelve es un software diseñado por AFNIC, que toma muchas de las métricas que se quieren medir en esta memoria, pero dejan de lado otras que se quieren incluir; como medir la capacidad de respuesta de los servidores de nombre ante una solicitud errónea, o verificar si es que se permiten consultas utilizando TCP. Además, este software sigue teniendo defectos, como por ejemplo la verificación no siempre correcta del uso de DNSSEC. Tomando en cuenta que fue modificado por última vez durante el año 2013, es presumible que ya no se corregirán estos errores.

Existen otras métricas para evaluar la resiliencia, como el uso de EDNS, o la capacidad

de permitir el uso de TCP en las consultas usando el test generado por DNS-OARC, que es un test de fácil uso mediante el comando `dig`. También, utilizando `dig` se pueden medir otras cosas, como la cantidad de servidores de nombre que tiene asociado un dominio, donde el recomendado es al menos 2.

Se puede averiguar también una ubicación tentativa del país en que se encuentran, averiguando a qué sistema autónomo pertenecen, para poder tener una idea de qué tan diversos son geográficamente hablando. Pero todos estos métodos nombrados hasta ahora son formas de conseguir los datos aislados, sin posibilidad de agregar los resultados fácilmente.

También existen herramientas para evaluar un único dominio a la vez, como Doctor DNS de NIC Chile, que entrega un diagnóstico de cada uno de los servidores de nombre asociados al dominio, o como *Simple Health Checker* de *MenAndMice*, que entrega un reporte del estado de DNS dado el nombre de dominio. Existe también una herramienta llamada *DNSSEC debugger* de *Verisign* que entrega un reporte del estado de validación de DNSSEC de un dominio, lo que también se podría hacer con *dig*, solicitando que se verifique DNSSEC con el *flag* `do` (*DNSSEC ok*)y esperando en la respuesta el *flag* `ad` (*Authenticated answer*).

Capítulo 4

Descripción de la Solución

Para poder mejorar la resiliencia del Internet chileno, es necesario conocer el estado actual, por lo que la solución propuesta consiste en recolectar datos asociados a un grupo de dominios y a sus servidores de nombre, para así poder obtener estadísticas sobre ellos para evaluar diferentes puntos de fallo.

Hay dos condiciones a las que está sujeto el diseño, las que definen la arquitectura del *software*: Lo primero es que se necesita rapidez al momento de recopilar datos (recopilar todos los datos en un tiempo del orden de horas en lugar de días) y lo segundo es que no se debe sobrecargar a los servidores externos ni a consultar a servidores prohibidos, ambos requerimientos producen un *trade-off*, que es la base de este trabajo de título.

4.1. Idea General de la Solución

La solución propuesta consiste en diseñar y desarrollar un *software*, adaptado a la realidad chilena, que sea capaz de generar un reporte con el detalle de la calidad y la resiliencia del Internet a nivel DNS de los dominios bajo el `.cl`. El *software* propuesto debe ser capaz de conseguir los datos en un tiempo no superior a 24 horas, teniendo cuidado de no sobrecargar de consultas a los servidores externos. Los módulos requeridos van a separarse en 3 capas: La primera capa es la de obtención y almacenamiento de Datos, la segunda es de análisis y agregación de datos, y la tercera parte es la de extracción de datos y su visualización.

4.1.1. Obtención y Almacenamiento de Datos

Dentro de la primera capa, debe haber una separación para cada grupo de dato que se quiere recolectar, siendo estos los siguientes:

Diversidad de Servidores Autoritativos

En esta etapa, se intenta averiguar qué tan dispersos se encuentran los servidores de nombre de un mismo dominio, tanto el cantidad, como en ubicación topológica y geográfica.

Cumplimiento de Recomendaciones

La idea aquí es verificar si los servidores DNS cumplen con ciertas recomendaciones como las mencionados en, sobre la utilización de protocolos como TCP y EDNS, y otras recomendaciones como responder aunque la consulta esté mal formada. La implementación de TCP hasta hace poco no estaba entre las recomendaciones, pero con la llegada de DNSSEC, las respuestas han aumentado de tamaño, lo cual hace necesario también comunicarse mediante este otro protocolo.

En cuanto a EDNS, que son las nuevas extensiones de DNSSEC hay muchos servidores que aun no las implementan por lo tanto se quiere verificar el uso o no uso de éstas.

Para el caso de las consultas incorrectas, existen servidores que no entregan un error cuando se solicita información de manera incorrecta, pese a que lo recomendado es enviar un error particular indicando que la consulta es incorrecta, en lugar de no responder o responder algo incorrecto.

En cuanto a los ataques asociados a DNS uno de los más comunes es el conocido como *DNS cache poisoning* o envenenamiento de caché DNS. El envenenamiento de caché DNS comienza cuando a un servidor de nombres se le hace una consulta solicitando recursividad. Si el servidor al que le consultan tiene habilitada la opción de realizar consultas recursivas, es muy probable que sea un servidor vulnerable con respecto a este ataque (ver más detalles en la sección 2.3), por lo tanto está también dentro de las recomendaciones no permitir recursividad en los servidores autoritativos de una zona.

Implementación y desarrollo de nuevas tecnologías

La no implementación de nuevas tecnologías puede ser un punto único de fallos. Por ejemplo la no implementación de DNSSEC o su implementación incorrecta, podría generar fallos de seguridad no garantizando que los datos entregados al cliente sean los que debería obtener. La no implementación de IPv6 por otro lado, podría disminuir la alta disponibilidad en el caso de que IPv4 tenga algún tipo de problema en algún nivel de la comunicación con el cliente por ejemplo. Por lo tanto es necesario recolectar los datos necesarios para verificar si dichas tecnologías están siendo implementadas correctamente o no.

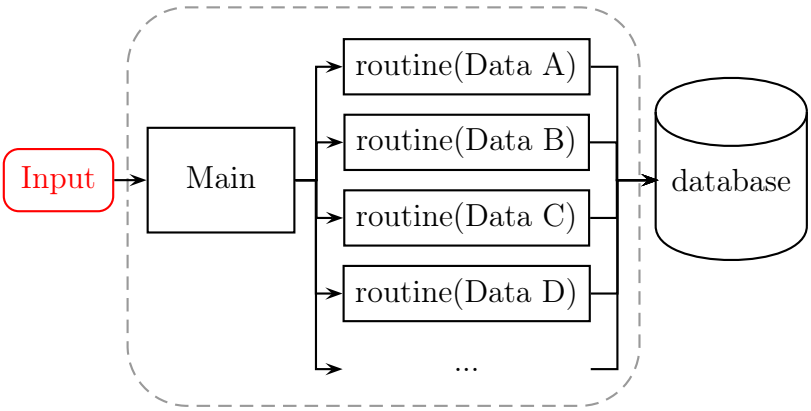


Figura 4.1: Diseño de la arquitectura del *software*.

4.1.2. Análisis y Agregación de Datos

La segunda capa, de análisis y agregación, toma los datos recolectados y genera estadísticas descriptivas sobre ellos de forma de tener una noción generalizada del estado actual de la resiliencia del Internet a nivel DNS.

4.1.3. Generación de Gráficas y del Reporte

La forma de este reporte es una página web que contiene una serie de gráficos mostrando el estado del Internet y una breve descripción y explicación de los datos.

4.2. Diseño

4.2.1. Arquitectura del *software* de recolección y análisis

La arquitectura diseñada se separa en 2 módulos. El primero es el de recolección de datos y el segundo corresponde al de análisis de los datos. Sin embargo, estos 2 módulos tienen un diseño muy parecido. En la figura 4.1 se puede ver el diseño que se separa en 4 partes: El flujo comienza por la entrada de los datos iniciales, que contiene los datos necesarios para que cada rutina se pueda ejecutar. Esta entrada de datos, es consumida por el *Main* o programa principal, que es el encargado de delegar tareas a múltiples rutinas. Las rutinas son pequeños programas que dado un nombre de dominio saben recopilar todos los datos que se explicaron anteriormente en la sección 3.

La forma en que el *Main* delega es generando rutinas que consumen los datos de un canal que contiene los nombres de los dominios, canal llenado previamente por el programa principal a partir del *input* de datos. Una vez que la rutina termina la tarea que se le asigna, guarda los resultados en la base de datos.

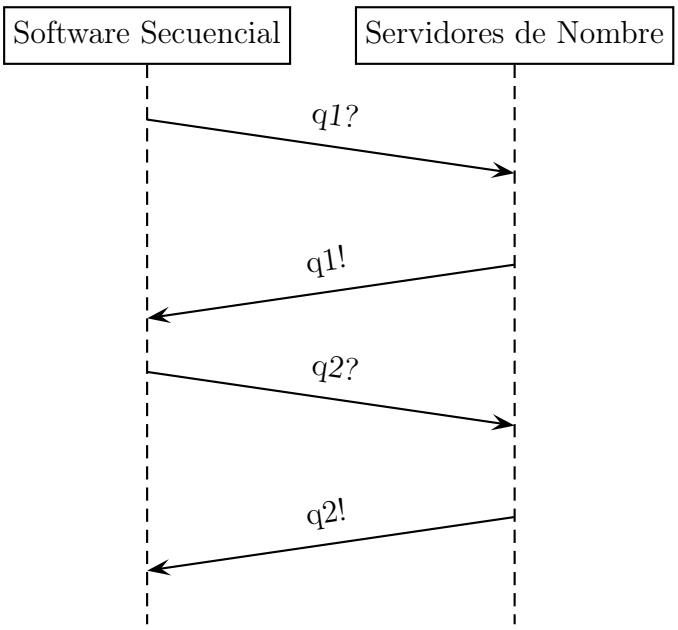


Figura 4.2: Ejemplo de consultas síncronas en un *software* secuencial.

Para el caso de la recolección de datos, la entrada de datos es una lista con los nombres de dominios a consultar, y en el caso del análisis de los datos el *input* corresponde a una consulta a la base de datos que entrega los datos a analizar.

La arquitectura se decidió basándose en el principio de que la tarea que realiza cada rutina corresponde a una serie de consultas síncronas o bloqueantes que, por su naturaleza, tienen un "tiempo muerto" de procesamiento, que ocurre cuando esperan la respuesta a alguna consulta realizada, por lo que tiene mucho sentido realizar múltiples consultas a la vez. En el esquema de la figura 4.2 se ve cómo un *software* secuencial ejecuta tareas bloqueantes, siendo mucho más lento que en un *software* multi-hilos. Como se ve en la figura 4.3, en el software multihilos, el tiempo es mucho menor debido a que se mandan múltiples consultas al mismo tiempo, por lo que el tiempo de espera debiese ser igual al tiempo de la tarea que se demora más.

Cabe destacar que debido a las limitaciones de procesamiento de los computadores, no se pueden ejecutar en paralelo todas las tareas que se requieran al mismo tiempo. Debido a esto, la solución que se diseñó toma un poco de las dos soluciones, basándose en el patrón de diseño *Single Program Multiple Data* (SPMD) o "Un programa, múltiples datos", secuencializando tareas relacionadas a un mismo dominio, pero lanzándolas en hilos diferentes hasta un límite de N hilos. Cuando el límite de los n hilos es alcanzado, se espera hasta que una de las tareas termine y luego se ejecuta la misma tarea con los datos del siguiente dominio en la cola.

Complejidad del algoritmo

Dada el diseño anterior y una configuración en donde la entrada es de m dominios y n es la cantidad de rutinas concurrentes, tomando en cuenta que el *Main* realiza al inicio una

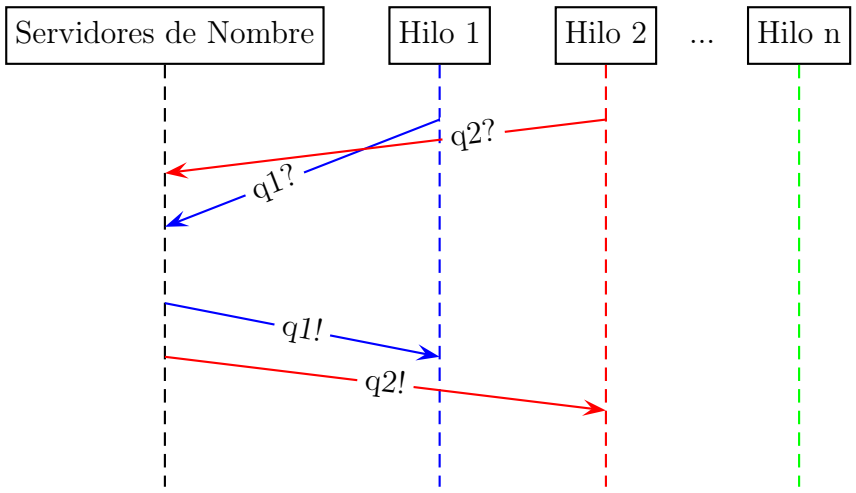


Figura 4.3: Ejemplo de consultas síncronas en un *software* multi-hilos.

serie de tareas de inicialización que toman tiempo constante A y asumiendo que cada rutina se demora un tiempo (más o menos) constante B se puede decir que la complejidad del algoritmo desarrollado es de:

$$Tiempo = \underbrace{A}_{\text{inicialización}} + \overbrace{\frac{m}{n}}^{\text{cantidad de rutinas a lineales}} * \underbrace{B}_{\text{Tiempo por cada rutina}}$$

El tiempo de inicialización es despreciable comparado con el tiempo restante, por lo que se puede decir que el tiempo disminuiría inversamente proporcional a la cantidad de rutinas. Sin embargo, cabe destacar que un alto número de procesos simultáneos genera una sobrecarga adicional de procesamiento por razones de *scheduling*, por lo que la disminución no será inversamente proporcional. Resultados del comportamiento real del *software* generado se pueden ver en la sección 5.

4.2.2. Diseño de la base de datos

Una vez que se obtiene toda la información necesaria, hay que guardar esa información para su posterior procesamiento. La forma adecuada es utilizar una base de datos, la que debe tener la capacidad de poder ser llenada de forma parcial, ya que la generación de estos datos se hace por partes, debido a su arquitectura modular. Por lo tanto el diseño de la base de datos es también realizado de forma modular, por lo que se tiene las siguientes tablas:

1. La tabla `Runs` almacena la información asociada a la iteración en que se ejecuta el *software*.

2. La tabla `Domain` guarda la información obtenida del dominio además de datos posteriormente analizados en relación a DNSSEC.

3. La tabla `DomainIp` guarda una dirección IP de un dominio (notar que un dominio puede tener muchas direcciones IP).

4. La tabla `Nameserver` guarda información de un servidor de nombre autoritativo asociado a su dominio, información como habilitación de recursión, habilitación para responder consultas con DNSSEC, entre otras cosas.

5. La tabla `NameserverIP` guarda información de la diversidad y tipos de direcciones IP de los servidores de nombre.

6. La tabla `Dnskey`, que almacena información de cada llave, como su estado de validez por ejemplo.

Para mayor detalle ver anexo 7.1.

4.3. Descripción detallada de la solución

Para evaluar si los dominios son susceptibles a las fallas descritas en el capítulo 3 hay que recolectar datos para su posterior análisis. Las métricas a utilizar están indicadas en la sección 2.6. Para poder utilizar estas métricas los datos a recolectar se muestran a continuación.

4.3.1. Datos a recolectar por cada dominio

Chequeo de existencia del registro SOA: Se realiza una consulta solicitando el registro SOA del dominio. Si dicho registro no existe, no se sigue analizando el dominio.

Obtención de direcciones IP: Se obtienen las direcciones IPv4 e IPv6 asociadas al dominio.

Validación DNSSEC: Se obtienen todos los registros necesarios para verificar si DNSSEC está implementado correctamente[1].

Verificación DNSSEC

Los datos a recolectar para verificar la correctitud de la implementación de DNSSEC se separan en dos tipos de respuesta.

La primera corresponde a cuando la consulta por un dominio entrega una respuesta positiva, con lo que hay que verificar que la respuesta venga con la firma RRSIG correspondiente y que la firma sea genuina verificando con los mecanismos definidos en los RFC 4033 [1], 4034 [3] y 4035 [2]. El segundo tipo de respuesta es una respuesta de no existencia, respuesta que

[1]Se explica más adelante en 4.3.1.

se obtiene cuando se pregunta por un dominio que no existe, donde hay que realizar una verificación de los registros NSEC o NSEC3 que verifican que realmente no existe dicho dominio, además hay que verificar siguiendo la cadena de confianza mediante los RRSIG correspondientes que dichos registros NSEC/NSEC3 son correctos según los mecanismos definidos en los RFC 4033 [1], 4034 [3] y 4035 [2].

Una descripción detallada de como funciona la prueba y negación de existencia (NSEC y NSEC3) se encuentra en la sección 2.4.

4.3.2. Datos a recolectar por Servidor de Nombres Autoritativo asociado al Dominio

Para cada dominio, hay uno o más servidores de nombre autoritativos, que son los que tienen autoridad para responder las consultas dirigidas a ese dominio. por eso, para cada servidor de nombre asociado al dominio, se obtienen los siguientes datos:

Obtención de direcciones IP: Obtener las direcciones IPv4 e IPv6 asociadas al servidor.

Recursividad: Si la respuesta a una consulta solicitando recursividad contiene el *flag* RA (*recursion allowed*) entonces este permite la recursividad.

EDNS: EDNS permite una correcta utilización de DNSSEC, por lo que también se quiere saber si está habilitado o no.

TCP: Verificar si el servidor permite consultas mediante TCP, para el correcto funcionamiento de DNSSEC.

Transferencia de zona: Verificar factibilidad de Transferencia de zona.

Respuesta ante cualquier consulta: Se envía una consulta por un dominio inexistente de un tipo raro (en este caso del tipo LOC) y se espera una respuesta negativa. En caso de no haber respuesta o de que se reciba un *timeout* se considera que no hay respuesta.

No consultar a ciertos servidores

Como uno de los requisitos del *software* era necesario dejar fuera las consultas asociadas a servidores que se encuentran en la "Lista Negra" por lo que antes de realizar cualquier consulta, hay que verificar si es que la consulta está dirigida a alguno de los servidores de la lista, en caso de que la respuesta fuera afirmativa, la consulta simplemente no se realiza y se continúa con la siguiente tarea.

La lista de los servidores viene en un formato particular: es una lista con redes de la forma a.b.c.d/x (ver más detalles en el Anexo 7.4), por lo que al hacer una consulta a un servidor del que se conoce el nombre, lo primero que hay que hacer es obtener su IP, luego verificar que la IP no esté contenida en ninguna de esas redes y en caso de que no esté, entonces

continuar haciendo la consulta requerida.

4.3.3. Análisis de los datos

Una vez que los datos están recolectados hay que analizarlos, el análisis se separa en 2 partes: Dispersión de *Nameservers* y sus vulnerabilidades, y verificación de DNSSEC.

Nameservers: Propiedades y dispersión

Para analizar la dispersión de los servidores de nombre se recopila información asociada a su ubicación, sistema autónomo al que pertenece y las direcciones IP asociadas a éste para evaluar qué tan dispersos están los servidores en esas 3 capas. Las direcciones IP se obtienen fácilmente, primero haciendo una consulta de los registros NS asociados al dominio obteniendo así los nombres de los servidores de nombre. Luego, para cada servidor de nombre se realizan consultas para obtener los registros A y AAAA para así obtener las direcciones IP tanto en versión 4 como en versión 6 de cada servidor de nombre. A continuación con las direcciones IP en mano, se realizan consultas a una base de datos abierta llamada GeoLite, de MaxMind[2], la cual se conecta usando la librería GeoIP para GoLang, de donde se obtiene información como el país o la ciudad[3] y el sistema autónomo al que pertenece dicha dirección.

DNSSEC: verificando la validez

Para verificar la validez de DNSSEC hay que verificar varios pasos:

1. Primero se verifica la existencia de un registro `DS` asociado al Dominio.

2. Luego se verifica la existencia de alguna llave `DNSKEY` para el dominio dado. Esta llave es la llave pública que se utiliza para verificar los registros firmados `RRSIG` bajo ese dominio.

3. teniendo ahora las `DNSKEY` y los registros `DS` se procede a verificar la llaves con los registros `DS`.

4. Finalmente se evalúa la no existencia de algún subdominio y su prueba de no existencia con registros `NSEC` o `NSEC3` (ver más detalles en la sección 2.4.2).

Todo esto siempre verificando en cadena las firmas `RRSIG` asociadas a cada set de registros antes mencionados.

[2]Este proyecto incluye datos de GeoLite creados por MaxMind, disponible desde `http://www.maxmind.com`

[3]En el software se utiliza solamente la información de país, ya que la información geográfica obtenida corresponde a información registrada al momento inscribir una IP, por lo que la ciudad registrada suele ser menos confiable que la información de país, aunque ninguna de las dos es 100 % confiable.

4.3.4. Generación de gráficos y reporte

Finalmente, para poder mostrar los resultados obtenidos y analizados se diseñó una página web, donde se muestran diferentes gráficos estáticos y dinámicos asociados a cada métrica disponible, para mostrar la dispersión y el cumplimiento de los estándares en cada aspecto. Algunos resultados se encuentran en la sección de Anexos, o también se puede revisar el reporte en `http://viz.niclabs.cl/ir/`.

Los datos utilizados se encuentran en formato CSV generados en la etapa de análisis de datos donde se lee y se procesa la base de datos previamente poblada en la etapa de recolección.

Capítulo 5

Resultados

Como resultados del desarrollo de esta memoria se obtuvo un *software* de recopilación y análisis de datos, que genera datos en formato CSV que se muestran en un reporte web de resiliencia del Internet a nivel DNS. A continuación se muestra la configuración utilizada para ejecutar el *software* desarrollado, los datos de entrada utilizados y los resultados obtenidos.

5.1. Datos de entrada utilizados

Para poder recopilar los datos es necesario saber de quién son los datos que se quieren. Por eso se necesita como entrada inicial una lista con los dominios a consultar.

Para la obtención de los datos se utilizó el archivo de zona del `.cl` que fue entregado por NIC Chile para fines únicos de la realización de este estudio. El archivo de zona (ver ejemplo en anexo 7.2) viene en el formato establecido en el RFC 1034[10] por lo que fue necesario realizar cierto procesamiento, de forma de obtener una lista con los dominios en el formato apto para poder ejecutar el software (ver anexos 7.3). El archivo de zona utilizado es de Enero 2017, fecha con la que se contaba con un total de 398.000 dominios. Para la evaluación de la *performance* del software se utilizó un subconjunto de 1603 de estos dominios.

Por otro lado, considerando que el software incluye un sistema que evita consultas a redes prohibidas, la entrada para este sistema es un archivo de texto que contiene una lista con todas las redes a las que no se debe enviar consultas (para revisar formato ver anexo 7.4).

Además, para poder realizar una comparación con el software DNSDelve, se realizó una ejecución con los datos del archivo de zona de LAC (estudio que ya se tenía realizado con dicho *software*).

5.2. Rendimiento

Para evaluar el rendimiento del software con distinta cantidad de rutinas se utilizó un archivo de entrada con 1603 dominio ejecutando el *script* en una máquina con las siguientes especificaciones:

- Servidor Dell PowerEdge R430 con sistema operativo *Linux*
- RAM: 64 GB.
- Controller: Perc H730P.
- Network: 4 Port Gigabit Ethernet.
- CPU: 2 x Intel Xeon E5-2650, 10 cores (40 virtual CPU).
- Storage: 2 x SAS, 600 GB, 10KRPM.

Además, la versión de GO utilizada es la versión 1.6.3 y de postgreSQL 9.2.15.

En la figura 5.1 se muestran resultados del promedio de tiempo de 20 ejecuciones del mismo proceso en la misma máquina con diferente cantidad de rutinas, donde se puede apreciar que a mayor cantidad de rutinas se obtiene un mejor rendimiento con lo que se valida el hecho de que separar las tareas en más hilos diferentes fue una buena decisión.

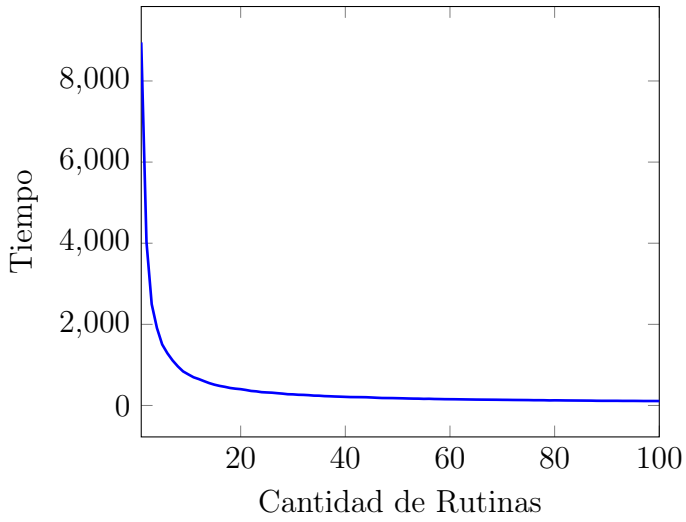


Figura 5.1: Tiempo de demora software con diferente cantidad de rutinas.

Cabe destacar que la disminución de tiempo no es lineal, lo que presumiblemente se debe a la sobrecarga que se genera con el *scheduling* al utilizar un alto número de hilos.

Cómo métrica de *performance* se utilizó la aceleración, que se define como la tasa de cambio de velocidad de el proceso con i hilos con respecto al tiempo del proceso realizado de

manera lineal, esto es:

$$Aceleración[\text{i}] = \frac{Tiempo\ lineal}{Tiempo\ con\ i\ hilos}$$

En la figura 5.2 se muestra la aceleración que se obtiene a utilizar una mayor cantidad de rutinas, donde se confirma que con una mayor cantidad de rutinas se obtiene un mucho mejor resultado.

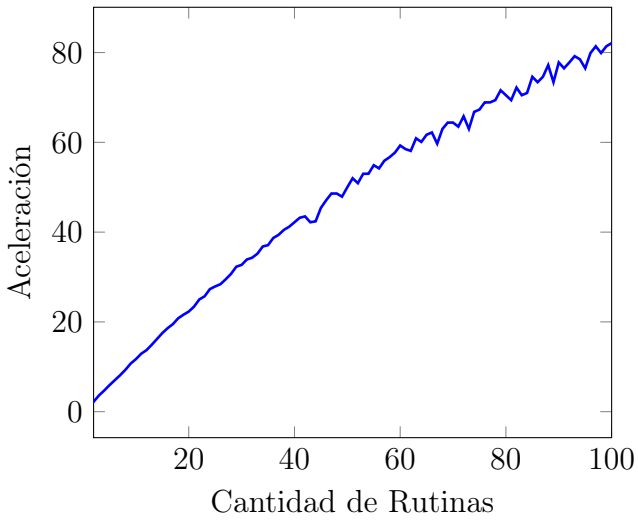


Figura 5.2: Aceleración de algoritmo al aumentar la cantidad de rutinas.

5.3. Comparación con el software existente

En un análisis preliminar, se pudieron obtener los siguientes resultados en comparación a ejecuciones anteriores realizadas con el software DNSDelve (ver referencia en la sección 3.3), donde se ejecutó sobre la zona de LAC (30124 dominios) con 100 hilos en ambos *softwares* y se obtuvieron los siguientes resultados:

	DNSDelve	Observatorio
Tiempo[min]	118	41
Tamaño BD[mb]	~ 20	~ 15.5

Tabla 5.1: Comparación de DNSDelve con el software desarrollado.

Donde se puede observar que el software implementado es casi un 300 % más rápido que el *software* existente. Además su base de datos ocupa menos espacio.

5.4. Estudio de resiliencia

De acuerdo a las métricas previamente definidas en la sección 2.6 se muestran los resultados obtenidos para el conjunto de los dominios bajo el `.cl`.

5.4.1. Dispersión de Servidores de Nombre

En cuanto a la dispersión de los servidores de nombre hay varios ítems a analizar.

Lo primero que se va a analizar es la cantidad de servidores de nombre que tiene asociado cada dominio. Como se vio anteriormente en la sección 2.6.1 la recomendación de un mínimo de 3 *nameservers* por dominio. Mirando la tabla 5.2 se puede ver que sólo un un 34 % de los dominios cumple con la recomendación de tener 3 o más servidores de nombre.

Cabe notar que hay 216.448 dominios que tienen sólo 2 servidores de nombre, lo que cumple la antigua recomendación, pero no la actual. Para ver más detalles de la cantidad de servidores de nombre ver el anexo 7.5 ó la página web `http://viz.niclabs.cl/ir/#g3`.

Cantidad de Servidores	Cantidad de Dominios	Porcentaje
< 3	224.225	65,97
≥ 3	115.675	34,03

Tabla 5.2: Cantidad de dominios que cumplen recomendación de más de dos servidores de nombre.

Ahora, si se hace un análisis de la dispersión geográfica, se puede ver en la tabla 5.3[1], la gran mayoría de los dominios no tienen dispersión geográfica de *nameservers* con respecto al país, habiendo sólo 83.574 dominios cuyos servidores de nombre se ubican en 2 o más países, lo que corresponde a casi un 25 % de los dominios.

Cantidad de países	Cantidad de Dominios	Porcentaje
< 2	251449	75,05
≥ 2	83.574	24,94

Tabla 5.3: Cantidad de dominios que cumplen recomendación de tener servidores en más de dos países.

Centrándose ahora en una dispersión a nivel de sistema autónomo, se puede ver en la tabla 5.4 sólo un 33,3 % de los dominios cumple con la recomendación de tener sus servidores de nombre autoritativos en 2 ó más sistemas autónomos. Para un mayor detalle de la distribución de la cantidad de sistemas autónomos ver anexo 7.5 o ver `http://viz.niclabs.cl/ir/#g4`.

[1]Ver detalle en la página web `http://viz.niclabs.cl/ir/#g5` o en el anexo 7.5.

Cantidad de sistemas autónomos	Cantidad de Dominios	Porcentaje
< 2	223.446	66,69
≥ 2	111.577	33,30

Tabla 5.4: Cantidad de dominios que cumplen recomendación de tener servidores en más de dos sistemas autónomos.

Ahora, si se ve la tabla 5.5, que agrupa los datos, y verifica la cantidad de dominios que cumplen con estas 3 recomendaciones, es decir, que tienen 2 o más servidores de nombre que se encuentran en 2 o más países organizados en 2 o más sistemas autónomos, se tiene que solo un 16,6 % de los dominios es resiliente con respecto a la dispersión, para ver un mayor detalle de cómo se distribuyen la cantidad de servidores, países y sistemas autónomos, se puede ver el gráfico en el anexo 7.5 o en la página `http://viz.niclabs.cl/ir/#g1` para ver el gráfico dinámico que muestra desagregadamente los datos.

Cumplimiento de recomendaciones	Cantidad de Dominios	Porcentaje
No cumple	279.389	83,39
Cumple	55.634	16,61

Tabla 5.5: Cantidad de dominios que cumplen las 3 recomendaciones de dispersión.

5.4.2. Hallazgos

Revisando sistemáticamente los datos y los errores encontrados al momento de realizar la ejecución se pudo notar que ciertos errores eran producidos por configuraciones erróneas de los dominios. En la tabla 5.6 se ven los distintos tipos de errores:

- El error "Sin registro NS" se refiere a los dominios que no contienen un registro NS asociado (recordar que un dominio sin un registro NS no se puede acceder, ya que el *nameserver* es quién dice dónde encontrar el contenido de dicho dominio).
- El error "Timeout" se refiere a los dominios cuyos NS no fueron encontrados por error de *timeout*, lo que presumiblemente es dado que no existe dicho récord para ese dominio, pero no es algo que se puede asegurar, ya que también podría ser por otros motivos, como saturación de la red por ejemplo.
- Y por último se ve un campo "Otros" que contiene errores no identificados.

Tipo de Error	Cantidad
Sin registro NS	46340
Timeout	11147
Otros	363

Tabla 5.6: Errores encontrados al recolectar datos.

Un hallazgo interesante al mirar estos errores fue el que en el error "Sin registro NS" es

un error que no debería aparecer, ya que por obligación un dominio, al incorporarse a la zona del dominio superior (en este caso la zona de `.cl` que fue la zona que se utilizó para realizar estos experimentos), debe tener un registro NS asociado. Por otro lado, al realizar una consulta mediante la herramienta *dig* a uno de estos dominios (`dig dominio.cl NS`) se obtiene el mismo resultado (no hay registro NS asociado), sin embargo al modificar un poco esta consulta, especificando el servidor al cual consultarle (y no dejando este valor por defecto) se obtiene un resultado diferente cuando se especifica a este servidor como uno de los servidores oficiales del NIC (`a.nic.cl, b.nic.cl, c.nic.cl`).

Este error, que no es poco común en la zona del `.cl` se conoce como *Lame Delegation*, que corresponde a un tipo de error que ocurre cuando un servidor de nombres es designado como autoritativo para un dominio para el cual no tiene información autoritativa. En el caso del `.cl`, al menos 46340 dominios caen en este caso, lo que corresponde a un 11,6 % de los dominios.

5.4.3. IPv4 y IPv6

Como se mencionó en el capítulo 2 la versión de IP utilizada puede dar un nivel extra de resiliencia en el Internet, por eso en esta parte se muestran los resultados de la dispersión en cuando a la versión de IP utilizada por los servidores de nombre de los dominios.

En la tabla 5.7 se puede apreciar que sólo un 13 % de los dominios cumple con tener al menos un servidor de nombre con dirección IPv4 y al menos un servidor con dirección en IPv6, de hecho, los dominios que no cumplen esto son dominios que tienen sólo dirección en IPv4. No existe ningún dominio que tenga servidores de nombre sólo con IPv6. Para ver un mayor detalle de cómo se distribuyen las versiones de IP de los servidores se puede ver el anexo 7.5 o ver la página web `http://viz.niclabs.cl/ir/#g9`.

Familia de IP	Cantidad de Dominios	Porcentaje
Sólo IPv4	291.029	86,87
Ipv4 e IPv6	43.994	13,13

Tabla 5.7: Cantidad de dominios que tienen servidores con direcciones IP en las dos versiones.

5.4.4. Estado de implementación de DNSSEC

Con respecto a DNSSEC lo único que se puede decir es que bajo los dominios del `.cl` es prácticamente inexistente, siendo sólo 78 dominios los que lo tienen implementado correctamente y 200 los que lo implementan con alguna falla.

Categoría	Cantidad de Dominios	Porcentaje
Sin DNSSEC	398.350	99,93
Con DNSSEC Fallando	200	0,05
Con DNSSEC Funcionando	78	0,02

Tabla 5.8: Cantidad de dominios que tienen DNSSEC.

Haciendo un análisis sobre los errores de esos 200 dominios que tienen mal implementado DNSSEC se puede ver en la tabla 5.9 que el error más común, es tener problemas con el registro DS, ya sea de parte de su validación o quizás su no existencia, lo que provoca una isla de seguridad. El segundo error típico consiste en no negar correctamente la existencia de los subdominios existentes, y por último el error menos frecuente es el de validación de las llaves DNSKEY, errores que se pueden deber a firmados incorrectos, o a expiración del contenido.

Falla	Cantidad de Dominios
Negación de existencia	134
Validación de llaves	46
Validación de DS	177

Tabla 5.9: Cantidad de dominios que tienen distintos errores en DNSSEC.

A modo de resumen, se presenta la tabla 5.10, donde se puede ver fácilmente que no hay ninguna categoría de recomendación que tenga un cumplimiento superior al 35 %, lo que no da mucha esperanza de resiliencia para los dominios bajo el `.cl`. Sin embargo, sin tener datos previos para comparar no se puede decir mucho más sobre eso.

5.4.5. Resumen de cumplimiento de recomendaciones de los dominios

Recomendación	Cumple	No cumple
Mínimo 3 servidores	34,03 %	65,97 %
Servidores en un mínimo de 2 Países	24,94 %	75,05 %
Servidores en un mínimo de 2 ASNs	33,30 %	66,69 %
Servidores con 3 recomendaciones de dispersión	16,61 %	83.39 %
Servidores con IPv4 e IPv6	13,13 %	86,87 %
Tener DNSSEC funcionando correctamente	0,02 %	99,98 %

Tabla 5.10: Nivel de cumplimiento de recomendaciones por dominio.

5.4.6. Cumplimiento de Recomendaciones de los servidores de nombre

Si es que hay un nivel alto de resiliencia en materia de DNS, es cuando se analizan los servidores de nombre, donde para cada recomendación hay un cumplimiento por sobre el 95 %, que aunque no es perfecto, está mucho más cerca de la perfección que todos los resultados anteriores. El detalle del cumplimiento se puede ver en la tabla 5.11 y en el gráfico del anexo 7.5 o en la página web `http://viz.niclabs.cl/ir/#g11`.

Recomendación	Cumplimiento	Porcentaje
Rechazar recursividad	825.913	98,40
Tener EDNS0 Habilitado	808.533	96,33
Tener TCP habilitado	797.403	95,00
Rechazar transferencia de zona	799.190	99,22
Responder a consultas LOC	839.286	99,99

Tabla 5.11: Nivel de cumplimiento de recomendaciones por servidor de nombres.

Capítulo 6

Conclusiones

El diseño del software implementado demostró obtener buenos resultados principalmente gracias a la utilización de patrones de diseño y de un lenguaje adecuado. Sin embargo, trabajar con un gran volumen de datos, trajo consigo otros problemas, que en este caso estuvieron relacionados, por un lado con el manejo del acceso a la base de datos y por otro lado con el manejo de las múltiples conexiones de red necesitadas, que abren muchos *sockets* y pueden ser difíciles de manejar.

Con respecto a los resultados obtenidos del estudio de resiliencia, el hecho de realizar una exploración manual sobre los datos fue de gran ayuda para encontrar irregularidades o problemas que a simple vista no son evidentes, como por ejemplo los *Lame Delegation*.

Con respecto al estado de resiliencia del Internet de los dominios del `.cl` se puede separar el estado en 3 partes: Dispersión, DNSSEC y estado de IP. En cuanto a la dispersión se puede decir que los dominios evaluados no son resilientes en materia de dispersión ya que sólo un 16,61 % de los dominios cumplía con los 3 requisitos mínimos de dispersión. Sin embargo, al no tener datos anteriores para comparar, no se puede decir nada sobre si se ha mejorado o no con respecto al estado de resiliencia del Internet a nivel DNS como país, por otro lado, como se pretende seguir realizando estos estudios de manera periódica, se está logrando establecer precedentes para futuras comparaciones del estado del Internet chileno.

Viéndolo desde otro punto de vista, dado que existen recomendaciones para las distintas categorías evaluadas, es útil saber dónde se está fallando, para así poder ir mejorando, por ejemplo, el hecho de verificar que muchos de los dominios sufren de *Lame Delegation* o que gran parte de los dominios que implementan DNSSEC no lo implementan correctamente es un indicio para alertar a la comunidad sobre la parte del DNS a la que se tiene que dar más prioridad.

Con respecto a la implementación de IPv6, se pudo detectar que, al parecer, no es de gran importancia para los administradores de los servidores de nombres implementar este nuevo protocolo, a pesar de que es sabido que las direcciones IPv4 están en proceso de agotamiento total.

En términos de seguridad, muy sorpresivamente, sólo un 0,02 % de los dominios tiene correctamente implementado DNSSEC, a pesar de las vulnerabilidades que estas extensiones de seguridad evitan, no parece haber mucha preocupación de parte de los administradores de los dominios en este ámbito.

A diferencia de los resultados anteriores, con respecto al cumplimiento de las recomendaciones de los servidores de nombre, se obtuvo que más de un 95 % de los servidores cumple correctamente con todas las recomendaciones, lo que corresponde a la parte más positiva de este estudio.

6.1. Trabajo futuro

Obtener datos y analizarlos es sólo el primer paso. Luego de poder realizar periódicamente estos estudios viene una etapa de análisis agregado de comparación, para poder evaluar así la evolución de los dominios y poder ir mejorando las falencia detectadas en cada etapa. Además, para poder comparar de manera más simple, sería muy útil tener una herramienta de visualización de la evolución de los cambios con respecto al estado de la resiliencia de Internet.

Además, como Internet evoluciona continuamente, se pretende ir también evolucionando las métricas utilizadas para poder estar acorde al día y que esta siga siendo una herramienta vigente.

Con respecto a las métricas utilizadas, sería útil poder mejorar algunas como la de dispersión geográfica, utilizando algún medio que otorgue información más localizada que simplemente el país, para así obtener resultados más certeros. Por el lado de la dispersión topológica podría ser útil analizar datos de bloques de IP, esto es, verificar qué tan lejanas son las IP para ver así qué tan resiliente es, ya que en general IPs muy parecidas suelen estar en el mismo lugar, o lugares cercanos geográficamente hablando.

En términos de crecimiento, y tomando en cuenta que la cantidad de datos encontrados para el dominio `.cl` es baja con respecto a otros TLD's como el de Brasil por ejemplo, sería un buen desafío intentar analizar un mayor volumen de datos incluyendo datos de otros países o incluso de toda latinoamérica.

Bibliografía

[1] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS Security Introduction and Requirements. RFC 4033, Internet Engineering Task Force, March 2005.

[2] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Protocol Modifications for the DNS Security Extensions. RFC 4035, Internet Engineering Task Force, March 2005.

[3] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. Resource Records for the DNS Security Extensions. RFC 4034, Internet Engineering Task Force, March 2005.

[4] D. Barr. Common DNS Operational and Configuration Errors. RFC 1912, Internet Engineering Task Force, February 1996.

[5] J. Damas and F. Neves. Preventing Use of Recursive Nameservers in Reflector Attacks. RFC 5358, Internet Engineering Task Force, October 2008.

[6] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, Internet Engineering Task Force, December 1998.

[7] R. Elz and R. Bush. Clarifications to the DNS Specification. RFC 2181, Internet Engineering Task Force, July 1997.

[8] Steve Friedl. An illustrated guide to the kaminsky dns vulnerability. Technical report, August 2008.

[9] R. Gieben and W. Mekking. Authenticated Denial of Existence in the DNS. RFC 7129, Internet Engineering Task Force, February 2014.

[10] P.V. Mockapetris. Domain names - concepts and facilities. RFC 1034, Internet Engineering Task Force, November 1987.

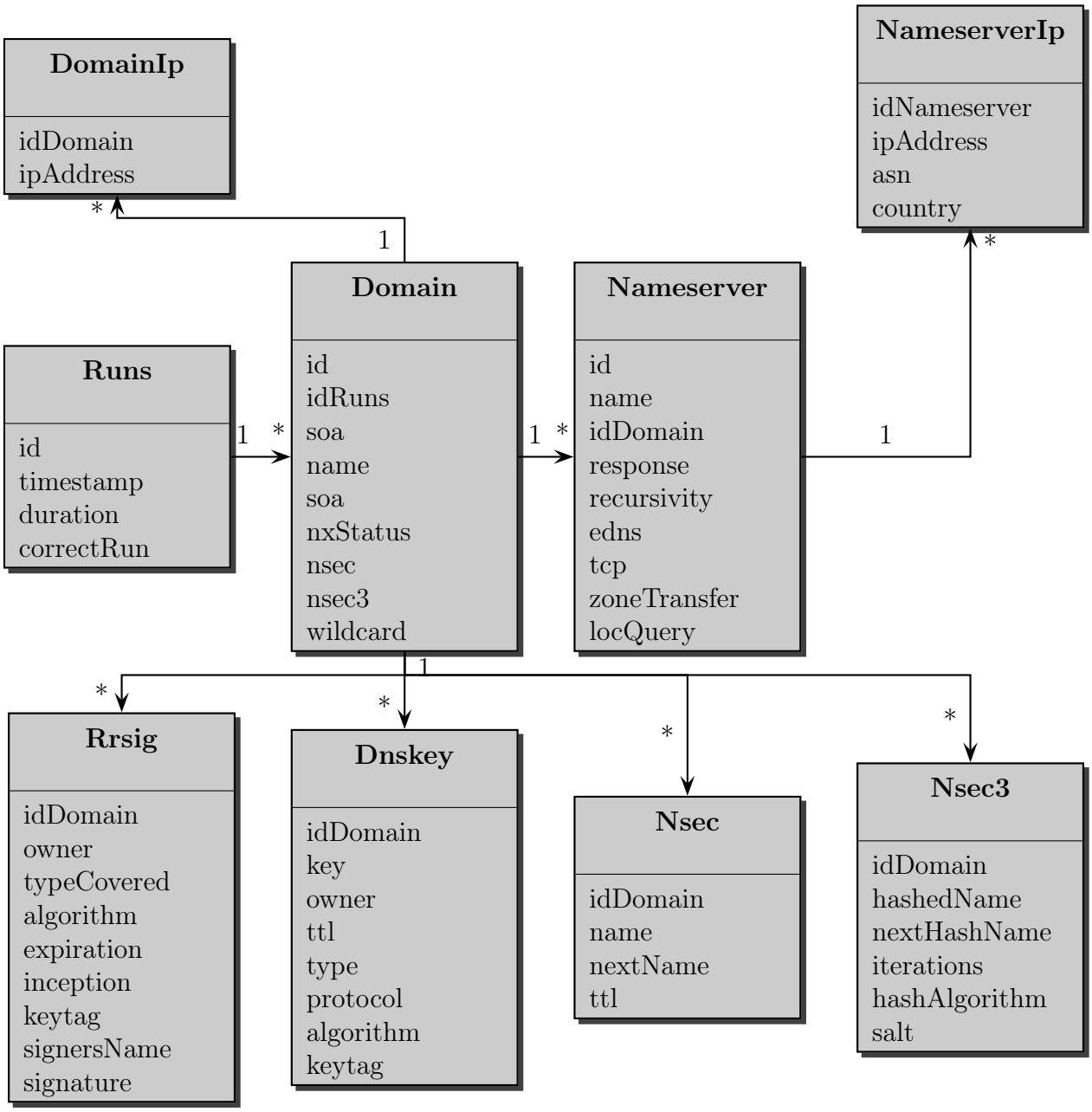
[11] P.V. Mockapetris. Domain names - implementation and specification. RFC 1035, Internet Engineering Task Force, November 1987.

[12] J. Postel. Internet Protocol. RFC 0791, Internet Engineering Task Force, September 1981.

Capítulo 7

Anexos

7.1. Diseño de la Base de Datos



7.2. Ejemplo de archivo de zona

```
$ORIGIN example.com.     ; designates the start of this zone file
   in the namespace
$TTL 1h                  ; default expiration time of all resource
    records without their own TTL value
example.com.  IN  SOA   ns.example.com. username.example.com. (
   2007120710 1d 2h 4w 1h )
example.com.  IN  NS    ns                    ; ns.example.com is
   a nameserver for example.com
example.com.  IN  NS    ns.somewhere.example. ; ns.somewhere.
   example is a backup nameserver for example.com
example.com.  IN  MX    10 mail.example.com.  ; mail.example.com
   is the mailserver for example.com
@             IN  MX    20 mail2.example.com. ; equivalent to
   above line, "@" represents zone origin
@             IN  MX    50 mail3              ; equivalent to
   above line, but using a relative host name
example.com.  IN  A     192.0.2.1             ; IPv4 address for
   example.com
              IN  AAAA  2001:db8:10::1        ; IPv6 address for
                 example.com
ns            IN  A     192.0.2.2             ; IPv4 address for
   ns.example.com
              IN  AAAA  2001:db8:10::2        ; IPv6 address for
                 ns.example.com
www           IN  CNAME example.com.          ; www.example.com is
    an alias for example.com
wwwtest       IN  CNAME www                   ; wwwtest.example.
   com is another alias for www.example.com
mail          IN  A     192.0.2.3             ; IPv4 address for
   mail.example.com
mail2         IN  A     192.0.2.4             ; IPv4 address for
   mail2.example.com
mail3         IN  A     192.0.2.5             ; IPv4 address for
   mail3.example.com
```

7.3. Ejemplo archivo de entrada

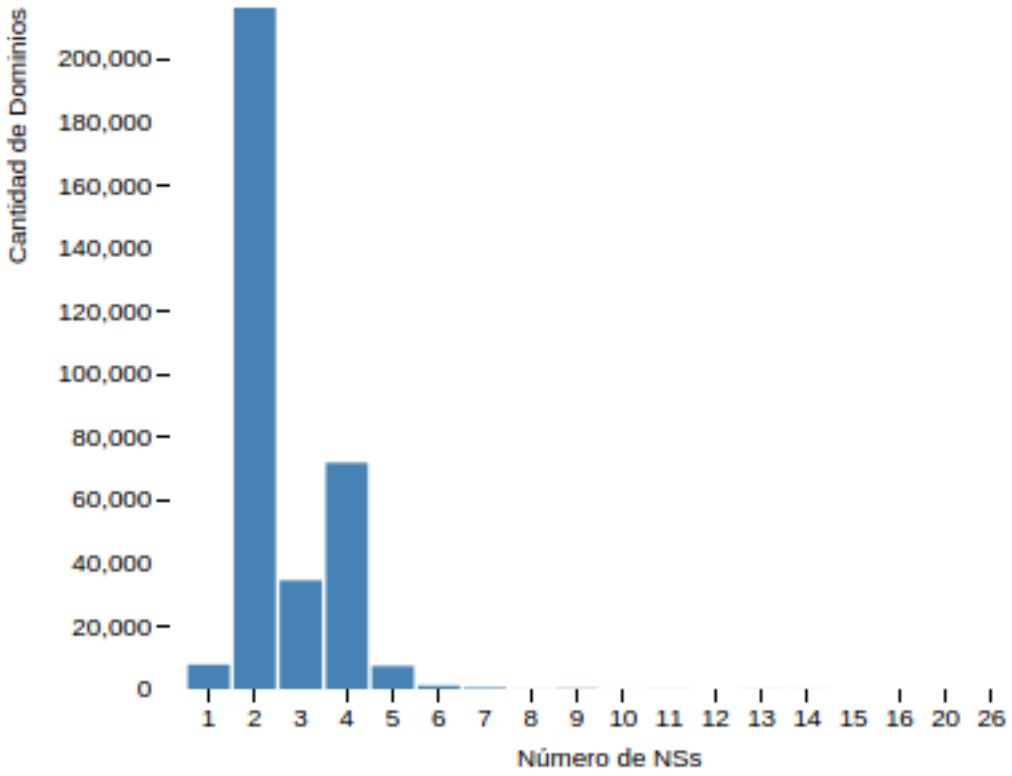
```
australvaldivia.cl
elvacanudo.cl
properati.cl
doite.cl
wieslaw.cl
mercadolibre.cl
dbsbeautystore.cl
elandino.cl
chilecompra.cl
misegundavivienda.cl
nikoncenter.cl
...
```

7.4. Ejemplo de archivo de lista con redes prohibidas

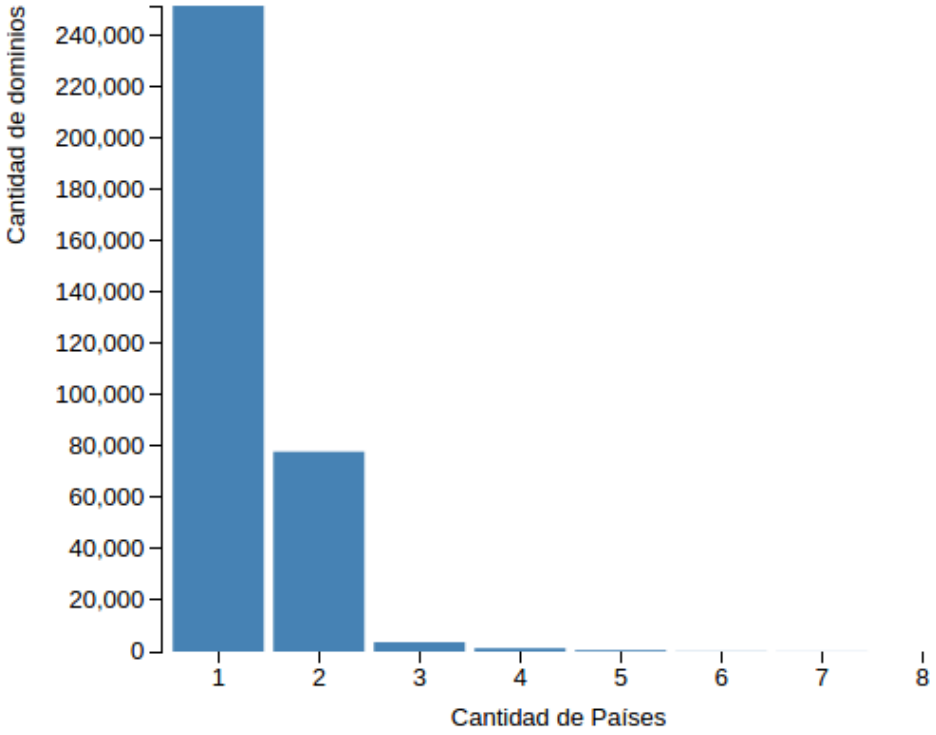
```
# ------------------------------------------
#                                          :
#       OARC's DNS Don't-Probe List        :
#                                          :
#       Generated for name                 :
#       at date                            :
#                                          :
# ------------------------------------------

# This list contains IP addresses and address
# ranges that should not be probed with DNS
# messages.  It is described in more detail
# at: http://dontprobe.dns-oarc.net/

# The use of this list is subject to the
# restrictions described in the access agreement at:
# http://dontprobe.dns-oarc.net/agreement.html

0.0.0.0/8
3.0.0.0/8
4.79.129.1/32
6.0.0.0/7
10.0.0.0/7
...
```

7.5. Resultados

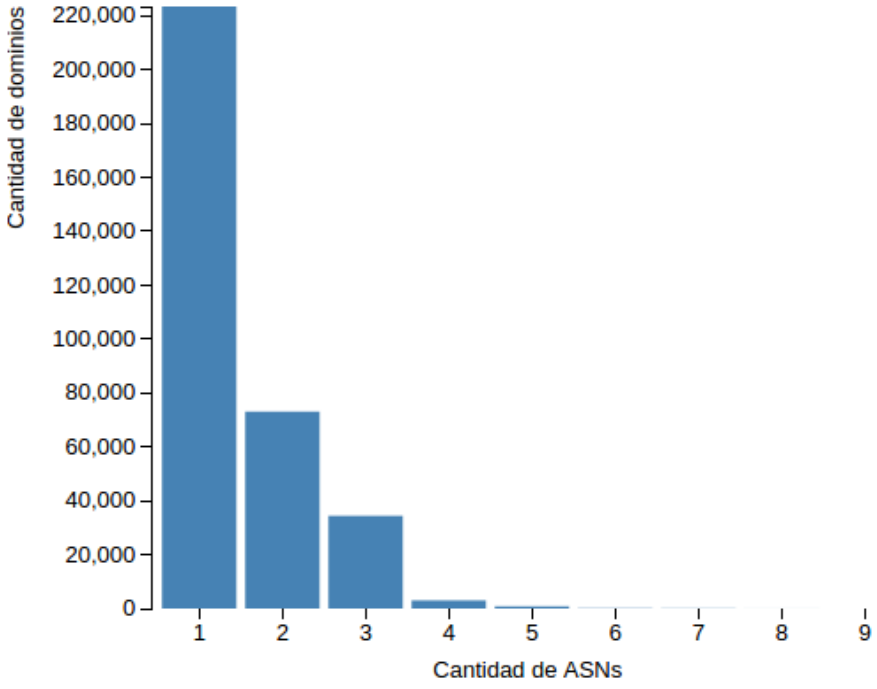
Cantidad de servidores de nombre versus cantidad de dominios



Cantidad de países de los servidores de nombre versus cantidad de dominios



Cantidad de sistemas autónomos de los servidores de nombre versus cantidad de dominios



Torta anidada de dispersión de servidores de nombre por dominio

El gráfico de torta anidado de la figura 7.1[1] muestra de forma dinámica la cantidad de dominios que cumplen diferentes condiciones. En la leyenda se encuentran 9 grupos de colores diferentes, para 3 de las métricas de dispersión: cantidad de servidores de nombre(círculo interior en tonos rojos), cantidad de sistemas autónomos(círculo medio en tonos azules) y cantidad de países(círculo exterior en tonos amarillos). Para cada métrica se hicieron 3 grupos asociados a la cantidad, ya sea 1, 2 y 3 ó más para cada una de las categorías.

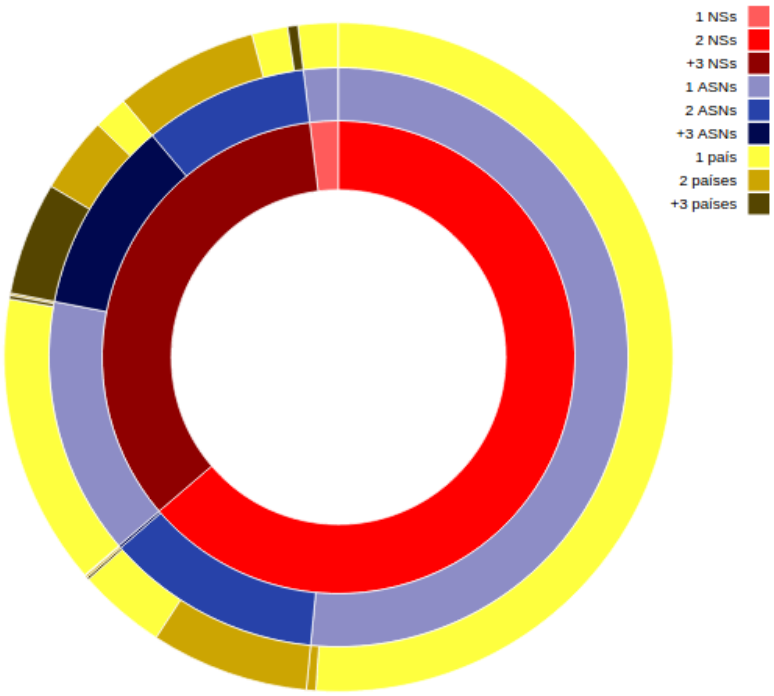
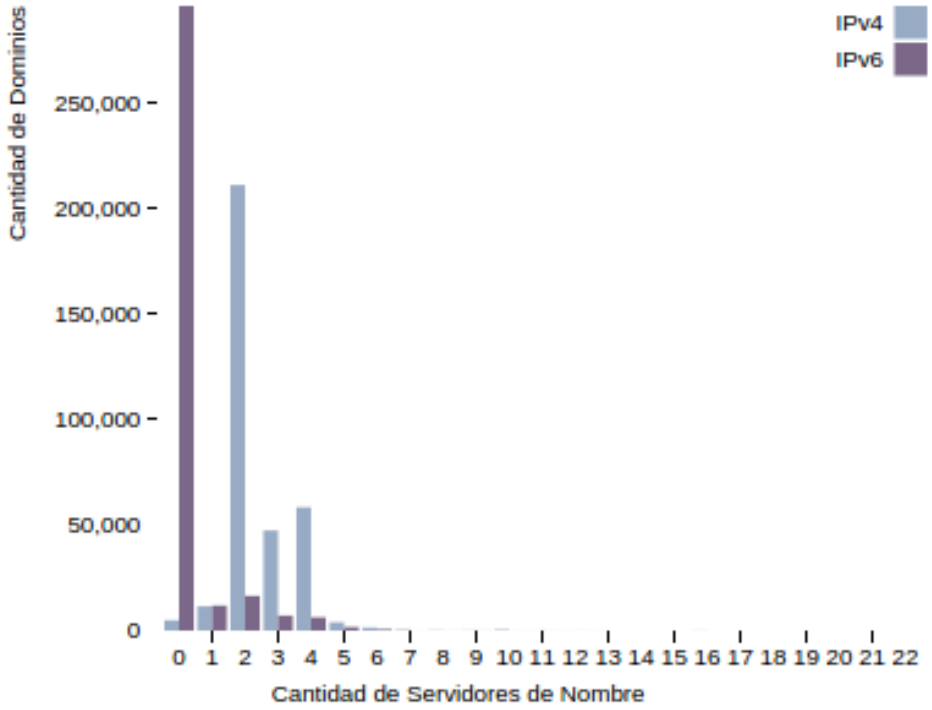


Figura 7.1: Torta anidada de dispersión de servidores de nombre por dominio

[1]ver detalle en la página web `http://viz.niclabs.cl/ir/#g1`

Cantidad de servidores de nombre por versión de IP versus cantidad de dominios



Cantidad de servidores de nombre que cumplen con las recomendaciones