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

PREDICCIÓN DE TRÁFICO DNS PARA DETECCIÓN DE EVENTOS ANÓMALOS

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS, MENCIÓN COMPUTACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

MARTÍN ESTEBAN PANZA PERALTA

PROFESOR GUÍA:
JAVIER BUSTOS JIMÉNEZ

MIEMBROS DE LA COMISIÓN:
ALEJANDRO HEVIA ANGULO
LUIS MATEU BRULÉ
GONZALO ACUÑA LEIVA

Este trabajo ha sido parcialmente financiado por NIC Chile Research Labs

SANTIAGO DE CHILE
2020

Resumen

El sistema de nombres de dominio (DNS) es, hoy en día, un elemento fundamental para el funcionamiento de todo Internet. Posee la tarea de traducir entre nombres de dominio y direcciones IP. Si se perdiese esta funcionalidad, no seríamos capaces de encontrar fácilmente, por ejemplo, los servicios web que frecuentamos y de los que dependemos (puesto que los humanos en general recuerdan mejor nombres que números). Es por esto que resulta de vital importancia asegurar el correcto funcionamiento de este sistema.

Ante la existencia de eventos maliciosos que amenazan las tareas del sistema DNS, la detección automática de anomalías toma especial relevancia. En la medida que se perciba a tiempo un evento malicioso, se podrán tomar medidas rápidas y eficientes que mitiguen o anulen el daño. La mayoría de los sistemas de detección se basan en aprendizaje supervisado. Esto es, buscar nuevas instancias de las amenazas que ya se conocen. Por otro lado, el aprendizaje no-supervisado se fundamenta en la búsqueda de patrones y cambios en el estado del sistema. Lo que permite también, en teoría, encontrar instancias de eventos de los que no se tiene previo conocimiento.

Este trabajo se centra en el enfoque no-supervisado para desarrollar un sistema de detección de eventos anómalos en tráfico DNS, mediante el uso de técnicas de *Machine Learning*. Estas herramientas se entrenan para aprender lo que corresponde a un comportamiento normal del sistema, haciendo posible predecir qué es lo que debería ocurrir en el futuro bajo un contexto normal. Observar una gran diferencia entre los valores reales y lo esperado por el modelo, señalaría la presencia de una anomalía.

Así, aprovechando los patrones periódicos de comportamiento humano presentes en DNS, se entrena un modelo de red neuronal recurrente para predecir tráfico DNS. Aplicando una heurística sobre los errores obtenidos, el objetivo de este trabajo es evaluar dicha técnica como detector de eventos anómalos, con el propósito de aumentar la seguridad en un servidor DNS particular, correspondiente al *dominio de nivel superior de código de país* (*ccTLD*) de Chile: *.cl*. Para esto, se recopiló el tráfico perteneciente a este servidor, ubicado en Santiago, Chile, durante un mes completo de su funcionamiento normal en el año 2017. Más precisamente, todas las consultas recibidas y respuestas enviadas durante aquel periodo.

El sistema propuesto fue validado a través experimentos donde fue sometido a un conjunto de ataques emulados sobre tráfico DNS real, evaluando su capacidad de detección. La agregación de la información en múltiples series de tiempo, en conjunto con las predicciones de comportamiento normal de estas agregaciones permitieron detectar los ataques emulados con alto *recall* (exhaustividad), de acuerdo a un *threshold* (umbral) determinado. No obstante, el procedimiento permitió revelar además, anomalías no-etiquetadas presentes en los datos que corresponden a eventos de carácter malicioso.

A nuestra familia, Agus, Mariana, Martín, Tere y Andrés.

Agradecimientos

A quienes, sin importar la tarea que esté haciendo o al desafío al que me enfrente, me entregan su apoyo, cariño y ánimo en forma incondicional. Tengo la enorme fortuna de que todas las personas que componen mi vida personal, corresponden a gente maravillosa. Suelo asombrarme de la sabiduría, bondad, fidelidad y valentía de aquellos y aquellas en mis relaciones cercanas, a quienes admiro más que a ningún otro.

A mis amigos y amigas de Asado's, los Welis y las Polillas, de Rayo, de Panzer, the Goons, de Zippedi, y a quienes no puedo agrupar y que de igual manera son indispensables para mí.

Un especial agradecimiento a mis amigos y amigas de NICLabs, con los que compartí durante la realización de este trabajo. Especialmente a Diego por su inmenso apoyo, colaboración y amistad. *"Lo importante es que nos damos cuenta."*

A Javier, por darme cada una de las varias oportunidades que formaron el inicio de mi carrera profesional de la mejor manera.

Y a toda mi increíble familia. Mis queridos tíos y tías, primos y primas, sobrinos y sobrinas, y a mis abuelos y abuelas. Particularmente, a mi abuelo Daniel, de quien heredé la pasión por la ingeniería, la ciencia y la computación; a quien admiro profundamente por su vasto conocimiento y generosidad.

El agradecimiento más especial es a mis hermanos, Mariana y Agustín, esenciales en mi vida; y a Tere y Andrés, las mejores personas que conozco, además de ser mis padres. Lo que recibo de ustedes va más allá de cualquier otra cosa.

Tabla de Contenido

Índice de Ilustraciones

Índice de Tablas

Glosario

CCTLD (***Country Code Top Level Domain***) Dominio asignado específicamente para un país, identificado por su código de país de dos caracteres. Ejemplos: *.cl*, *.us*, *.nz*. 8

curvas ROC (***Receiver Operating Characteristic Curve***) Gráfico entre la tasa de verdaderos positivos versus la tasa de falsos positivos que busca diagnosticar el resultado de una clasificación binaria de acuerdo a la variación de un parámetro. 46

DNS (***Domain Name System***) Sistema de nombres de dominio, realiza la traducción de direcciones IP a nombres de dominio. 1

DNSSEC (***Domain Name System Security Extensions***) Extensiones añadidas al sistema DNS para asegurar mayor seguridad, ante el descubrimiento de vulnerabilidades en la implementación original. 11

DoS (***Denial of Service***) Tipo de ataque malicioso que busca causar la denegación del servicio de algún objetivo, típicamente sobrecargándolo mediante un envío continuo de información. 16

IP (***Internet Protocol***) Protocolo que identifica únicamente a cada miembro de la red que compone a Internet. 1

LSTM (***Long Short-Term Memory***) Tipo de red neuronal recurrente que busca mantener información de largo plazo en su aprendizaje. 18

MAE (***Mean Absolute Error***) Medida de error entre el promedio absoluto de dos series de tiempo. 33

MAPE (***Mean Absolute Percentage Error***) Medida porcentual de error entre el promedio absoluto de dos series de tiempo. 33

NXDOMAIN Código de respuesta utilizado por DNS para especificar la no-existencia de un dominio. 57

RFC (***Request For Comments***) Documentos publicados específicamente para describir o

discutir aspectos referentes a la implementación de Internet, bajo la organización IETF. 8

RR (***Resource Records***) Unidad elemental de información que maneja el sistema DNS. Poseen tipos específicos, que determinan la clase de datos que conllevan. 9

Serie de Tiempo Secuencia de valores numéricos ordenados cronológicamente. 18

TCP (***Transmission Control Protocol***) Protocolo para el transporte de datos a través de Internet. Establece una conexión entre emisor y receptor, por lo que asegura entrega, orden y chequeo de errores en el envío de datos. 17

TTL (***Time To Live***) Número designado para un paquete que se transmite a través de Internet, el cual se va decrementando en cada nodo que lo procesa, tanto con el transcurrir del tiempo desde su origen como con el número de traspasos entre nodos. Al disminuir a 0, el paquete es descartado o enviado de vuelta a su origen. 5

UDP (***User Datagram Protocol***) Protocolo para el transporte de datos a través de Internet. Carece de conexión entre emisor y receptor, por lo que no asegura entrega, orden o unicidad de los datos. Utilizado especialmente en aplicaciones en la que la retransmisión no es fundamental, como DNS. 17

Capítulo 1

Introducción

Internet, como un medio instantáneo de acceso a una masiva cantidad de información, es innegablemente un agente fundamental para el desarrollo de la vida humana y de su cultura. Desde servicios críticos como los sistemas bancarios y financieros, hasta servicios de entretenimiento como videojuegos, muchos dependen estrechamente del funcionamiento de este medio. Inclusive, la vida social de muchas personas se ha vuelto dependiente de las comunicaciones que Internet nos permite. Especialmente en los últimos años, donde el acceso a Internet desde dispositivos móviles ha aumentado considerablemente el número de usuarios en Internet. Siendo éste el principal medio de conexión a la red desde el año 2016 [4].

En efecto, Internet ha experimentado un crecimiento considerable en las últimas décadas, alcanzando un número de 4536 millones de usuarios activos en el año 2019 [2], un 5 % más que respecto al año anterior. Esto corresponde a un aumento de más de 200 millones de personas, valor que se incrementa de forma casi constante cada año. Esto se traduce en más del doble de usuarios con respecto a hace 10 años. Asimismo, se superó a partir del año 2017 el 50 % de porcentaje de la población mundial total, tal como lo muestra el gráfico de la Figura 1.1.

DNS es un sistema que ha crecido de forma muy similar a como lo ha hecho Internet. Esto se debe principalmente a que corresponde a un sistema cuyo funcionamiento es crucial para Internet mismo, puesto que, al estar encargado de la traducción de nombres de dominio a direcciones IP, es quien nos permite acceder a cualquier sitio (o contenido visible) de acuerdo a su nombre de dominio; tales como *registrocivil.cl*, *uchile.cl*, *google.cl*. Ya que éstos son nombres que los humanos manejamos bien - en contraposición a las direcciones IP, por las cuales se identifican los computadores que se comunican a través de la red -, DNS se encarga de encontrar el sitio al cual queremos acceder. En caso de fallar, se tendría que buscar una forma alternativa de encontrar la IP respectiva y comunicársela tanto a nuestro dispositivo, como a los nodos en la ruta hasta aquel *host*. Esto resulta inverosímil en una red del tamaño como la de Internet. Así, la infraestructura actual se apoya fuertemente en DNS.

El sistema DNS almacena de forma distribuida todos los dominios públicamente existentes. Solo los registrados en las autoridades de *dominios de nivel superior* (ver Sección 1.2.4) han alcanzado los 354.7 millones este año [8], aumentando a un ritmo aproximado de 4.4 % cada año. Es decir, se registraron alrededor de 14.9 millones de dominios nuevos bajo estas

autoridades en el último año. Lo que concuerda con el crecimiento que ha experimentado Internet, donde la cantidad de sitios web ha alcanzado los 1729 millones. En consecuencia, existe un enorme flujo de datos y cantidad de información que se maneja en DNS, a través de todo Internet.

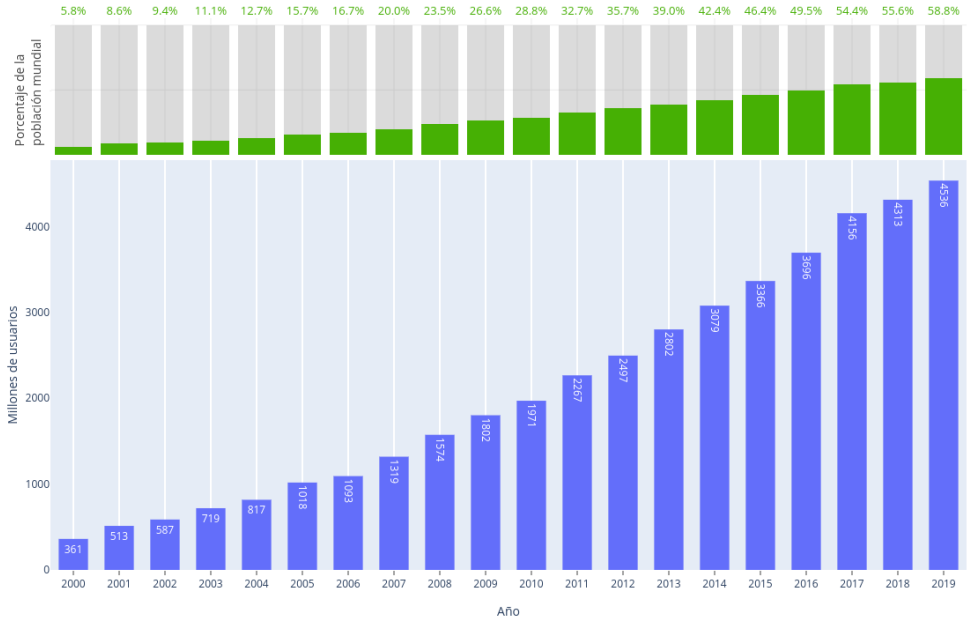


Figura 1.1: Evolución del número de usuarios en Internet por cada año durante los últimos 20 años. Para cada año se muestra en azul el número de usuarios total (en millones), y en verde se muestra el porcentaje de la población mundial que aquel número de usuarios en Internet representa.

1.1. Motivación

A lo largo de la existencia de DNS, han surgido diversos ataques en contra de las vulnerabilidades de su infraestructura, ataques con objetivos tales como el robo de autoridad (sustracción de información, ejecución de acciones no autorizadas), la negación del servicio o la distribución de información falsa. Considerando la importancia que el funcionamiento de DNS tiene para Internet, es vital protegerlo contra estos eventos. No obstante, esta no es una tarea simple. Las debilidades que han sido descubiertas, se han solucionado al costo de añadir una alta complejidad a la implementación del sistema. Lo que ha causado, a su vez, el origen de otros tipos de problemas que, asimismo, han debido detectarse y tratarse en un proceso de perfeccionamiento, como se detalla más adelante en la Subsección 1.2.7.

Más aún, la masiva cantidad de datos que se mueve a través del tráfico generado por DNS dificulta enormemente este proceso de detección y tratamiento de eventos maliciosos. En efecto, se hace imposible inspeccionar de manera individual cada paquete enviado, o a un individuo específico con este propósito. Por ejemplo, en el tráfico percibido por un *dominio de nivel superior* de alto uso, donde pueden llegar a recibirse miles de consultas por cada segundo.

Frente al surgimiento de nuevos eventos de los cuales no se tiene conocimiento, resulta muy difícil asegurar la completa seguridad del sistema. Inclusive, instancias de ataques para los cuales se tiene un conocimiento previo pueden ser modificadas o simplemente aumentadas en magnitud, dejando inútiles a las medidas de protección tomadas anteriormente. Por ejemplo, haciendo uso de redes de *bots*, con posibilidad de infectar varios computadores para que actúen involuntariamente como causantes de un ataque.

Actualmente, la mayoría de sistemas de detección que abordan este problema se basan en aprendizaje supervisado; es decir, utilizan conocimiento previo para reconocer nuevas instancias de un evento en particular. Estos sistemas son incapaces de detectar eventos desconocidos o lo suficientemente modificados. Más aún, en caso de reconocerse un nuevo evento, es necesario el diseño de una solución, en conjunto con la implementación o la actualización debida; lo que puede significar un largo periodo de vulnerabilidad.

Por otro lado, utilizando aprendizaje no-supervisado, sí es posible detectar eventos desconocidos, bajo la premisa de que cualquier evento anómalo generará algún cambio en algún punto del estado del sistema. Lo que no requiere conocimiento a priori. Esto motiva el presente trabajo de Tesis, donde se pretende utilizar aprendizaje no-supervisado para detectar eventos maliciosos en grandes volúmenes de tráfico de DNS de forma automática.

Un sistema que cumple con este objetivo sería de gran importancia para operadores DNS, los cuales deben ofrecer confiabilidad en sus servicios. Detecciones oportunas de amenazas que puedan comprometer el funcionamiento del sistema les permitiría la toma rápida de medidas de protección.

1.2. DNS

En esta sección se describe el sistema de nombres de dominios (por sus siglas en inglés: *Domain Name System* ó DNS) y se explica su funcionamiento, con el objetivo de dar un contexto necesario para entender la hipótesis y el trabajo realizado que se detallan más adelante en este documento. No obstante, tomando en cuenta lo complejo que es el sistema DNS [29, 11], se presentan a continuación únicamente los aspectos que se consideran relevantes para esta Tesis en particular.

Las direcciones IPv4 (en adelante IP) son las que identifican únicamente a cada *endpoint* en una red. Es decir, a cada punto de conexión visible. Cada dirección está compuesta por 4 bytes representados por números del 0 a 255 concatenados por puntos, de la forma: *192.0.7.23*. Esta dirección es la que nos permite acceder a los recursos ubicados en un miembro específico de la red, sea este un computador, servidor, router o cualquier tipo de elemento conectado capaz de comunicarse. Sin embargo, el llevar cuenta de las direcciones IP que determinan a cada miembro se transforma en un problema cuando la cantidad de miembros en la red es elevada. Este es el problema que se observa en la vasta red que compone a internet, donde la cantidad de direcciones IP es inmensa, correspondiente a la enorme cantidad de usuarios, tal como se expuso y discutió al inicio de este capítulo.

El sistema DNS resuelve este problema al permitirnos asociar a las direcciones IP nombres más fáciles de recordar para los humanos. Naturalmente, nos resulta más simple recordar si-

tios como “Google”, “Banco Estado” o “CNN” que sus respectivas direcciones IP. Una analogía correspondería a los números de teléfonos o celulares que guardamos en nuestro dispositivo, a los cuales asignamos un nombre con significado que nuestro dispositivo almacena, ahorrándonos el trabajo de memorizar aquel número extenso y sin significado. En adelante, es nuestro dispositivo el que se encarga de la traducción cuando le comandamos comunicarnos con un contacto de acuerdo a su nombre, haciendo que escasamente tengamos que siquiera ver aquel número otra vez. En el caso de DNS, los nombres asignados son llamados *dominios*, y los números corresponden a direcciones IP. Sin embargo, en DNS, esta información que permite traducir en ambos sentidos, se encuentra distribuida a lo largo de toda la red. En efecto, DNS funciona como una base de datos distribuida.

La razón de esta configuración, es debido a que responde al problema del alto volumen de datos que se maneja. Naturalmente, la traducción requiere de una tabla que permita el mapeo de cada dominio con su dirección IP asociada. Originalmente, cuando Internet era una pequeña y joven red llamada “ARPANET,” esta tabla era copiada y almacenada en cada una de las máquinas dentro de la red, siendo actualizada y anunciada por los administradores semanalmente. Ante la explosión del número de usuarios, mantener la consistencia y almacenar el creciente espacio que requería esta tabla se volvió inviable. Además del trabajo que esto significaba para los administradores en términos de carga de tráfico de red y verificación de errores, tales como la duplicación de nombres.

De esta forma, fue implementado un modelo de datos jerárquico con estructura de árbol invertido, cuya información parcial pertinente se encontrase de manera particular en cada uno de los miembros de la red, con tal de que se tenga acceso a toda la información de forma remota desde cualquier lugar de la red.

1.2.1. Estructura de Datos

DNS funciona como una base de datos con estructura de árbol invertido. Donde cada nodo corresponde a un dominio, el cual contiene información sobre el host, o grupo de hosts asociados a ese dominio. Todos los nodos poseen un nombre con largo máximo de 63 caracteres. Dos nodos hermanos, es decir provenientes del mismo nodo superior, deben poseer un nombre distinto. Sin embargo, el *nombre de dominio* correspondiente a un nodo, no esta solamente dado por el nombre del nodo, sino que por la concatenación de este nombre con los nombres de todos sus antecesores, separados por puntos. En el ejemplo de la Figura 1.2, el nombre de dominio para el nodo destacado en rojo corresponde a *dcc.uchile.cl*. De esta forma, cada nombre de dominio existente en la base de datos es único. Incluso si existen un dominio *dcc.uchile.cl* y un dominio *dcc.uc.cl* de forma simultánea.

Naturalmente, el árbol tiene una raíz. Este nodo tiene por nombre el *string vacío*. En consecuencia, todos los nombres de dominio terminan en ‘.’, que es en general omitido. A este nombre completo, por ejemplo *dcc.uchile.cl.*, se le llama *fully qualified domain*, pues determina inequívocamente la posición del nodo en el árbol. Se dice también que un dominio es *subdominio* de otro si el primero es un sucesor del segundo dentro del árbol.

Si junto al hecho de que cada dominio es único agregamos que cada dominio contiene, aparte de la información relativa a su host, la información referente al nodo raíz y a sus nodos hijos, resulta posible acceder a la información de cualquier nodo del árbol desde cualquier

otro nodo. Es el mismo nombre de dominio consultado el que contiene lo necesario para realizar la resolución de su dirección IP.

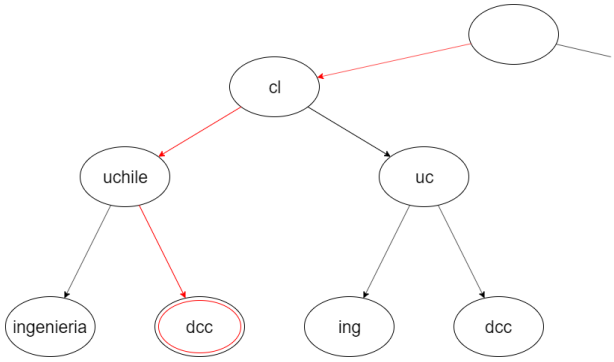


Figura 1.2: Estructura jerárquica de los dominios en DNS. En rojo se destaca el ejemplo del nombre de dominio *dcc.uchile.cl.*, el cual comienza desde la raíz y sigue la concatenación de dominios *cl* y *uchile*, finalizando en el nodo *dcc*, denotado por un doble óvalo.

1.2.2. Resolución

En DNS se entiende por resolución al proceso de traducción de nombre de dominio a dirección IP, incluyendo la búsqueda a través del árbol que este conlleva. El proceso de resolución funciona mediante una implementación cliente/servidor. Al cliente, quien consulta a los servidores por la información, se le menciona *resolver*. El resolver comienza la búsqueda en el nodo raíz, quien le entrega la información de a cuál de sus hijos debe ir a consultarle para continuar el camino al nodo que busca. Por ejemplo, si deseamos encontrar el nodo de *dcc.uchile.cl*, la raíz le dará al resolver la información de *cl*, quien de igual manera, al ser consultado, lo referirá a *uchile*, finalmente entregando la dirección de *dcc*, el cual posee la dirección IP para acceder al servicio que se busca. Este ejemplo está ilustrado en la Figura 1.3.

Sin embargo, debemos notar ciertos inconvenientes en este proceso. Lo primero que se debe observar es que la cantidad de carga que recibe el nodo raíz en este escenario es tremenda, donde cada consulta hecha por cada usuario empieza en la raíz. Es por esto que más información, aparte de la dirección del nodo raíz, es almacenada en los servidores intermedios. La búsqueda no necesariamente comenzará en la raíz, en general se comienza consultando nodos dentro de la misma zona DNS. Concepto que se detalla más adelante en la sección de Delegación. Junto a esto, no siempre es necesario realizar a otros servidores una consulta que ya se realizó hace poco tiempo. Las respuestas obtenidas pueden ser almacenadas localmente de forma temporal por los resolvers. Esto se llama *caching*, lo que representa una muy importante optimización que alivia el tráfico del sistema enormemente. Sin embargo, esto provoca el problema de mantener información errónea en caso de que la información en el servidor consultado cambie o sea eliminada. Para esto se da un tiempo determinado al caching de la respuesta, llamado *Time-To-Live* (TTL), antes de volver a consultar al servidor respectivo para actualizar la información.

Lo segundo que notamos es que es el cliente quien está realizando todas las consultas a cada nodo dentro del árbol. Tampoco esto es siempre de esta manera. La responsabilidad

de realizar la resolución también puede ser encomendada al servidor al que se consulta, pidiéndole que, en caso de no tener la información necesaria para la resolución, sea él quien realice otra consulta al siguiente servidor correspondiente. A esto se le conoce como *resolución recursiva*, representada en la Figura 1.3, específicamente en el número 1, donde el resolver que consulta solo espera una respuesta sin realizar más trabajo. Diferente a lo realizado por el resolver en los números 2, 4 y 6, donde al recibir una referencia al siguiente servidor, realiza una nueva consulta iterativamente hasta obtener la respuesta buscada. Esta última es la *resolución iterativa*.

Es el tipo de consulta el que determina el tipo de resolución que se usará. Es decir, para encomendar una resolución recursiva, esta debe solicitarse. Sin embargo esta puede ser denegada. Por ejemplo, un servidor de mucha carga no puede tardar mucho ni asignar muchos recursos a las consultas que recibe. De la misma forma, un cliente simple, como un usuario hogareño, no esperaría realizar toda la labor por sí solo. Por otro lado, la ventaja radica en que existen resolvers, con los recursos necesarios y especialmente destinados a esta función, que están dispuestos a atender resoluciones recursivas para un cliente. Por ejemplo, los resolvers que ofrece una ISP (proveedora de servicio de Internet). Esto compone los tres tipos de agentes que podemos encontrar en el proceso de resolución, y que en la literatura se pueden encontrar con las siguientes denominaciones:

1. Servidores Autoritativos: son aquellos que tienen la información necesaria para responder dentro de su zona DNS.
2. Resolvers Recursivos: atienden consultas recursivas del cliente. Realizan consultas a otros servidores o utiliza la información de su cache para entregar una respuesta a su cliente.
3. Stub-Resolver: Un Resolver básico que consulta a un Resolver Recursivo por la resolución de algún dominio. Posee tiempos de espera máximos y rutinas de selección de servidores en base a tiempos de espera.

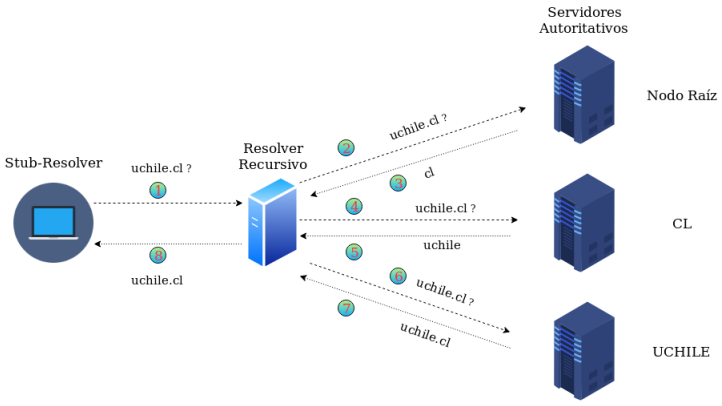


Figura 1.3: Diagrama de resolución en DNS. Se ilustra aquí el proceso de resolución ordenado secuencialmente (desde 1 hasta 8), que procede cuando un cliente (*Stub-Resolver*) envía una consulta a su *resolver recursivo* por el dominio *uchile.cl* **(1)**. El resolver realiza la resolución iterativa al consultar primero al *servidor autoritativo* del nodo raíz **(2)** y recibir la referencia al *servidor autoritativo* del subdominio siguiente: *cl* **(3)**. Repitiendo este procedimiento con los servidores autoritativos de *cl* y *uchile* **(4-5** y **6-7** respectivamente**)**, hasta encontrar la dirección buscada y entregarle una respuesta al cliente **(8)**.

1.2.3. Delegación

La delegación introduce el concepto de *zona* al sistema DNS. La idea de esta función es crear unidades autónomas compuestas de un grupo de dominios adyacentes dentro del árbol, con el objetivo de distribuir la carga del trabajo de resolución entre estas unidades. Es decir, siempre existirá una autoridad responsable de responder a las consultas dentro de una misma zona. La delegación consiste en asignar la responsabilidad de un subdominio a una nueva autoridad, lo que implica la creación de una nueva zona DNS. Naturalmente, de acuerdo a la estructura de árbol, una zona siempre tiene un dominio raíz.

En términos de implementación, esto significa que los servidores correspondientes a un dominio y sus subdominios en una misma zona poseen, en primer lugar, la información necesaria para responder consultas por nombres de dominio pertenecientes a aquella zona en la que se encuentran; y en segundo lugar, referencias a los subdominios a los que se delegó responsabilidad, pertenecientes a otra zona DNS, para así permitir la resolución por consultas de nombres de dominio más profundos dentro del árbol.

En el ejemplo de la Figura 1.4, el dominio *cl* es una zona por sí solo, ya que delegó la responsabilidad de la administración de los subdominios *uchile* y *uc* a otras autoridades: Universidad de Chile y Universidad Católica respectivamente. Así, en caso de que *cl* reciba una consulta por uno de sus subdominios, él solo entrega la referencia a la zona de la autoridad correspondiente. En caso de que la consulta sea *dcc.uchile.cl*, se referencia a *uchile*, cuyo servidor DNS está dispuesto por la Universidad de Chile. Por su parte, *dcc* pertenece al Departamento de Ciencias de la Computación (DCC) de la Universidad de Chile. La Universidad de Chile delegó, asimismo, el subdominio *dcc* a este departamento para que lo administre de forma independiente, iniciando una zona DNS. Por otro lado, *repositorio* es también un subdominio de *uchile*, que dirige al sitio de repositorio académico de la Universidad de Chile. Este subdominio no fue delegado a otro organismo, por lo que la misma Universidad de Chile lo administra, perteneciendo a la misma zona.

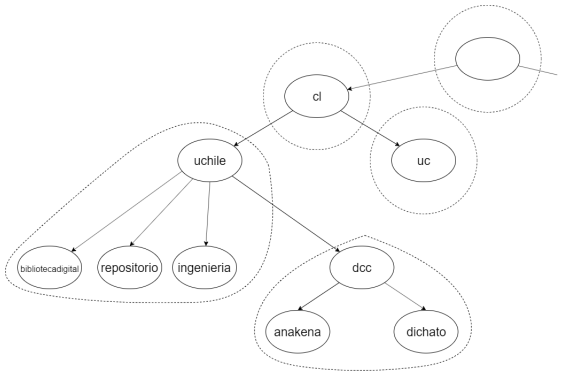


Figura 1.4: Zonas DNS en la estructura jerárquica de dominios. En esta figura, los dominios englobados juntos corresponden a una misma zona DNS. Es decir, se encuentran bajo la misma administración, delegando a otras zonas en caso de existir un subdominio bajo una administración ajena. En el ejemplo, Universidad de Chile administra *bibliotecadigital.uchile.cl.*, *ingenieria.uchile.cl*, *uchile.cl* y *repositorio.uchile.cl*. Mientras que el dominio *dcc.uchile.cl* es delegado a una administración diferente, que corresponde en particular al Departamento de Ciencias de la Computación (DCC), traduciéndose en una nueva zona DNS.

1.2.4. Dominios de Nivel Superior

El administrador del nodo raíz, a la fecha, es la organización sin fines de lucro ICANN (Internet Corporation for Assigned Names and Numbers). Sus subdominios directos son llamados *Dominios de Nivel Superior*. En sus inicios, de acuerdo al RFC 1591 [41], estos correspondían a un grupo de estricta selección por parte de los administradores del dominio raíz. Se clasificaban en los *dominios de nivel superior genéricos* (*gTLD*) y los *dominios de nivel superior de código de país* (*CCTLD*).

Los *gTLD* eran *.com, .org, .net, .int, .edu, .gov* y *mil*, los cuales tenían cada uno un propósito específico. Por ejemplo, *.com* fue destinado a organizaciones con fines comerciales. Por su parte, *.edu* sería asignado a instituciones educacionales. En esta época, los dominios estaban estrictamente ligados a Estados Unidos, como los casos de *.gov* y *.mil* que pertenecían particularmente al gobierno y ejército de este país respectivamente.

En cuanto a los *ccTLD*, un dominio fue asignado a cada país reconocido por las Naciones Unidas, utilizando el código de dos letras que les corresponde de acuerdo a lo especificado en ISO 3166-1. De esta forma, cada país está encargado de administrar su dominio respectivo de forma completamente independiente. En efecto, algunos países como Nueva Zelanda (*.nz*) o Argentina (*.ar*) también establecen una división organizacional de sus subdominios; por ejemplo *.com.ar* o *.org.nz*. Asimismo Estados Unidos (*.us*), entre otros, establece subdominios para sus estados, existiendo *.ca.us* para California o *.hi.us* para Hawaii. Finalmente, otros dominios de código de país, como el de Chile (*.cl*), no realizan este tipo de separación y asignan subdominios directamente.

Mediante esta convención, la idea era reconocer fácilmente el propósito del dominio y/o su país; información que se encuentra contenida en el sufijo del nombre de dominio por ser un nodo superior. No obstante, con el tiempo esta organización se fue disipando. Circunstancialmente, *.com* se convirtió en el dominio más utilizado. Especialmente a partir del "Dotcom-Boom" ocurrido a fines del siglo XX. Hoy, el 46.7 % de los sitios web en el mundo utiliza un dominio *.com* [5]. El siguiente dominio más utilizado es *.org* con 5.2 %. Por lo demás, no solo compañías con fines comerciales utilizan el dominio *.com*, sino también organizaciones sin fines de lucro o establecimientos educacionales.

Por otro lado, al establecer este esquema de organización en el RFC 1591 [41], se mencionó que "es muy poco probable que se creen nuevos TLDs." No obstante, nuevos *TLD* han sido aprobados desde entonces, lo que ha causado a su vez varias situaciones controversiales. Como la negación a la creación del dominio *.islam*, o la creación del dominio *.sucks*. De hecho, por un tiempo, un nuevo tipo de dominios de nivel superior fue establecido por ICANN: los *dominios de nivel superior patrocinados* (*sTLD*), abiertos a representantes de alguna comunidad. Como *.travel* para agencias de viaje, aerolíneas u hoteleras, o *.museum* para museos. Sin embargo, en la actualidad ICANN no sigue reconociendo esta categoría de dominios de nivel superior y los considera dentro de los *gTLD*.

Finalmente, un dominio de nivel superior particular llamado *.arpa*, es considerado un *dominio de nivel superior de infraestructura*, y contiene la información relevante para realizar las búsquedas DNS inversas.

1.2.5. Registros DNS

Las zonas DNS están definidas por un *archivo de zona* (RFC 1035 [34]). En este archivo se encuentran todas las especificaciones que permiten administrar la zona DNS a cada uno de los hosts responsables de esta tarea. Estos hosts pueden ser uno o más servidores, que en conjunto, deben ser capaces de entregar una respuesta definitiva ante una consulta por un dominio considerado dentro de su zona, o de referir a los servidores correspondientes en caso de tratarse de un subdominio ajeno. Este archivo de zona determina desde la información que se entregará como respuesta, hasta la organización de estos servidores.

Lo primero que se especifica en el archivo de forma obligatoria es la variable *TTL* correspondiente a la zona, indicando al resto de los servidores DNS cuánto tiempo pueden guardar en sus *cache* la información que reciben. Luego existe una variable llamada *ORIGIN*, la cual define el nombre de dominio del cual se origina la zona. El resto de las especificaciones del archivo corresponde a los *resource records* (*RR*), o traducidos, *registros DNS*.

Los *registros DNS* determinan el tipo de información que intercambia DNS. Existe una lista determinada de *resource records*, en la cual cada registro posee un código y especifica un tipo particular de información. Los más importantes son:

- A: Este registro contiene la transformación directa entre una dirección IPv4 y un nombre de dominio.
- AAAA: Es el equivalente del registro A para IPv6.
- NS: Define a los servidores DNS autoritativos para una zona o dominio.
- MX: Define a los servidores de servicios e-mail para una zona o dominio.
- SOA: Contiene la información básica de una zona DNS. Tales como el servidor DNS autoritativo principal, tiempos de actualización y tiempos de expiración.

Todos los registros anteriores fueron definidos en RFC 1035 [34], a excepción de *AAAA*, el cual es definido posteriormente en RFC 3596 [45].

De esta forma, siempre que se pida información a un servidor DNS, se añade el código del tipo de *registro DNS* junto con el *payload*. El servidor DNS responderá buscando en los *resource records* que posee en su *archivo de zona* para entregar una respuesta.

Respecto a los *registros DNS* contenidos en el *archivo de zona*, antes que cualquier otro, se debe tener un registro de tipo *SOA*, que otorga la información general sobre la zona misma. Este es obligatorio y debe ser único en el archivo. Luego se tendrán obligatoriamente registros *NS*, los cuales listarán a los servidores autoritativos pertenecientes o no a la zona. Asimismo, se tendrán registros *MX* para los servicios de e-mail. Esto es opcional y pueden, de la misma forma, referenciar servidores locales o ajenos a la zona. Finalmente, los registros *A* y *AAAA* definen las direcciones IPv4 e IPv6 para los hosts de la zona respectivamente. Éstos son accesibles públicamente y son opcionales.

1.2.6. Balanceo de Carga

El balanceo de carga es una funcionalidad de carácter fundamental en DNS. El balanceo de carga es una estrategia para distribuir el trabajo percibido por un servicio DNS, aumentando su capacidad. Esto tiene por objetivo ofrecer mayor resiliencia y reducir los tiempos de latencia en las respuestas. Para llevar esto a cabo existen dos mecanismos comunes.

Lo primero que se utiliza para este fin, son los registros *NS* mencionados en la subsección anterior. Usando estos registros, se designan múltiples servidores que operan en diferentes IP, capaces de responder las consultas por los nombres de dominio dentro de la zona DNS.

El segundo mecanismo consiste en utilizar múltiples servidores asociados a una misma dirección IP. A esto se le conoce como *IP anycast*. Se utiliza enrutamiento para seleccionar el destino más conveniente en base a medidas de latencia. Por ejemplo, mediante el protocolo *BGP*. Uno de los beneficios que esto permite, es una eficiente distribución geográfica de los servidores.

En la Figura 1.5 se ilustran estas dos configuraciones, donde un dominio puede ser atendido por múltiples IPs distintas. A su vez, estas IP pueden dirigir a diferentes servidores según conveniencia, usando *IP anycast*.

La mayoría de las zonas DNS de mayor jerarquía, como el nodo raíz y los dominios de nivel superior, utilizan estas configuraciones. También resolvers públicos de alto tráfico como OpenDNS [6].

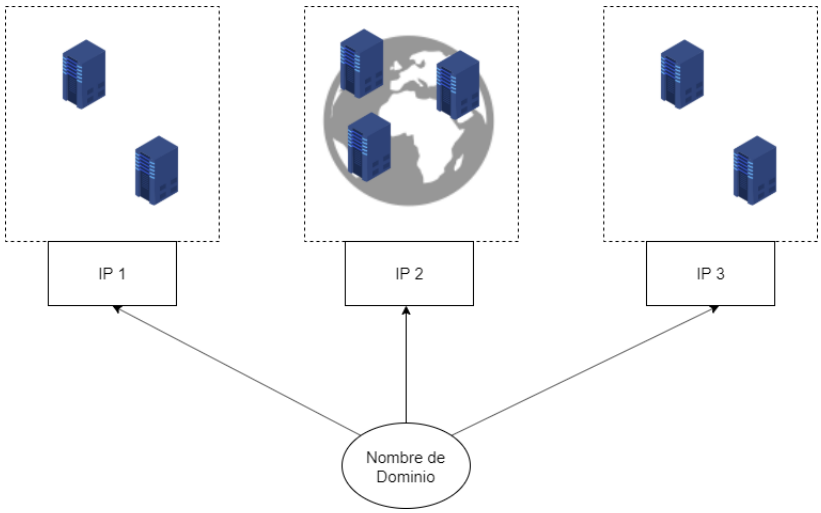


Figura 1.5: Balanceo de carga utilizando IPs múltiples e IP anycast. En esta figura se muestran ambas técnicas de balanceo de carga, en la que tres diferentes direcciones IP pueden atender a un mismo dominio utilizando registros del tipo NS (*IP múltiple*). A su vez, cada una de estas IP puede implementar independientemente *IP anycast* para distribuir la carga a través de múltiples servidores, inclusive repartidos alrededor del mundo (IP 2). Esta es la configuración que usa el nodo Raíz con sus trece diferentes direcciones IP.

El caso más importante es el del nodo raíz, puesto que corresponde al nodo que más carga recibe en todo el sistema. Este implementa ambas técnicas mencionadas anteriormente. Al

día de hoy, existen 13 servidores con direcciones IP propias que responden por esta zona DNS. De hecho, cada uno de ellos es administrado por operadores independientes en acuerdo con ICANN, como VeriSign Inc. o NASA. Cada uno de estos 13 servidores implementa, a su vez, *IP anycast*, por lo que existen, en la práctica, cientos de servidores distribuidos en todo el mundo atendiendo por el dominio raíz. La razón particular para que sean 13 direcciones IP, es que todas estas direcciones IP se encuentran contenidas en cada uno de los paquetes DNS. Si se aumentase este número, se incrementaría en consecuencia el tamaño de todos los paquetes transmitidos, añadiendo mayor carga al tráfico de red.

1.2.7. Seguridad en DNS

A través de la historia de DNS han surgido razones para realizar modificaciones en la implementación del sistema, especialmente relacionadas con temas de seguridad. El espíritu de estas modificaciones siempre fue de ser compatibles con lo anterior, sin transgredir los principios originales de la implementación. La manera en que se llevaron a cabo estos cambios fue a través de extensiones a lo que ya se tenía previamente, añadiendo registros DNS. Esto causó que el sistema se volviese mucho más complejo. Más aún, las extensiones que se debieron realizar fueron más grandes de lo que se pensaba que serían inicialmente. En efecto, con los nuevos registros DNS implementados, los paquetes requerían un tamaño mayor. A pesar de que esto es posible de solucionar mediante fragmentación en la capa de red, atenta contra la concepción original de DNS, además de facilitar ataques de amplificación.

Ya se discutían varias brechas de seguridad (interceptación de paquetes, servidores no confiables, re-direccionamiento intencionado) cuando se plantearon nuevas extensiones de seguridad en el RFC 2065 [18] en 1997, las cuales más tarde se establecerían en un conjunto de especificaciones bajo el nombre de DNSSEC (Domain Name System Security Extensions). Principalmente se sabía que era posible inyectar información falsa en el proceso de resolución de las consultas DNS, por lo que se necesitaba asegurar la autenticación de quienes solicitaban y recibían información. Lo que inicialmente se pensaba sería una solución pequeña fue escalando a medida que se descubrían nuevas falencias. Por ejemplo, para validar la no existencia de un nombre de dominio, se debía entregar una respuesta explícita autenticada de esto. La respuesta entregaba información sobre el dominio más parecido encontrado, lo que originó a su vez el *zone walking*, que consiste en la realización de múltiples consultas con el único objetivo de descubrir información referente a una zona DNS. De esta forma, DNSSEC no se implantó hasta el año 2010, trece años después de su primera versión.

El hecho que causó un apresuramiento en la implementación de esta extensión fue la publicación de un estudio hecho por Dan Kaminsky en 2008, que demostraba cómo, adivinando números de secuencia, era posible contaminar el caché de los resolvers con información errónea. A este ataque se le llamó Kaminsky Cache Poisoning.

1.2.8. DNSSEC

DNSSEC consiste en una serie de registros DNS nuevos que aseguran la autenticidad de las respuestas entregadas por los servidores. Se utiliza *criptografía asimétrica* para verificar, mediante llaves públicas y privadas, la fuente de la información de acuerdo a dos aspectos. Primero, que quien envía la información es efectivamente el administrador de la zona con-

sultada. Y segundo, que este se encuentre dentro de la cadena de confianza que se establece desde el nodo raíz hasta su zona DNS. Los nuevos *RRs* (resource records) más importantes en esta extensión son:

- DNSKEY: Contiene una llave pública. Existen dos posibles: Zone Signing Key (ZSK) y Key Signing Key (KSK).
- DS: Contiene una llave KSK transformada mediante una función de Hash, enviada por nodos hijos a nodos padre para establecer una cadena de confianza.
- NSEC/NSEC3: Respuesta explícita de no-existencia de un registro DNS. Se diferencian en que NSEC3 aplica una *función de Hash* a la información con el fin de evitar *zone walking*.
- RRSIG: Corresponde a una firma que certifica que un conjunto de registros DNS pertenecen a un servidor autoritativo.

La extensión se añade al proceso de resolución explicado en la Subsección 1.2.2 de la siguiente forma. Un resolver obtiene un certificado *RRSIG* junto a la información que buscaba originalmente. Este certificado fue creado por la zona utilizando una llave privada. Un *DNSKEY* con la llave pública es solicitado también. Esta corresponde a la llave *ZSK*, que permite validar que el certificado corresponde auténticamente a la zona DNS que consultó el resolver, y que nadie se ha hecho pasar por la autoridad de esa zona. Ahora, también debemos saber si esta zona DNS es confiable, por lo que se debe verificar que pertenece a la cadena de confianza desde el nodo raíz. Esto se asegura haciendo que las llaves *ZSK* sean, a su vez, firmadas por otra llave privada generada por la zona: la llave *KSK*. Una llave pública *KSK* es solicitada por el resolver también mediante un registro *DNSKEY*. La zona que crea esta llave pública, utiliza además una función de *hash* para obtener una transformación que es enviada al nodo padre a través de un registro *DS*. De esta forma, el resolver confirma con el nodo padre que aquel es su subdominio auténtico al solicitar el registro *DS*. Esto se repite de nodo en nodo hasta la raíz, recorriendo la cadena de confianza establecida. Proceso ilustrado en la Figura 1.6.

Por motivos de seguridad es conveniente generar nuevas llaves periódicamente. Esto es mucho más simple de realizar para las llaves *ZSK* que para las llaves *KSK*. En consecuencia, las llaves *KSK* son renovadas con mucha menor frecuencia, y se debe esperar un tiempo a que la información sea borrada de los *cache*, puesto que los resolvers la actualizan también con menor frecuencia. En el caso del nodo raíz, esto se realiza alrededor de una vez por año y en una ceremonia. Esta ceremonia consiste de una cantidad fija de personas expertas y de confianza que se reúne en una facilidad resguardada físicamente, en la que los participantes actúan como testigos de la generación de una nueva llave *KSK* para la raíz, y que asegurará la cadena de confianza para los servidores de nombres de todo Internet. Esta llave se genera y almacena utilizando una máquina de encriptación HSM (Hardware Security Module). Al finalizar el proceso, la máquina usada para la ceremonia anterior es completamente destruida.

Esta ceremonia consiste en la participación de una cantidad exacta de personas expertas participantes con turnos fijos para entrar de manera solitaria a una habitación a realizar labores con hardware específico y guardando apropiadamente registros del proceso.

Si bien DNSSEC fue establecido hace casi una década, su instauración no ha sido precisa-

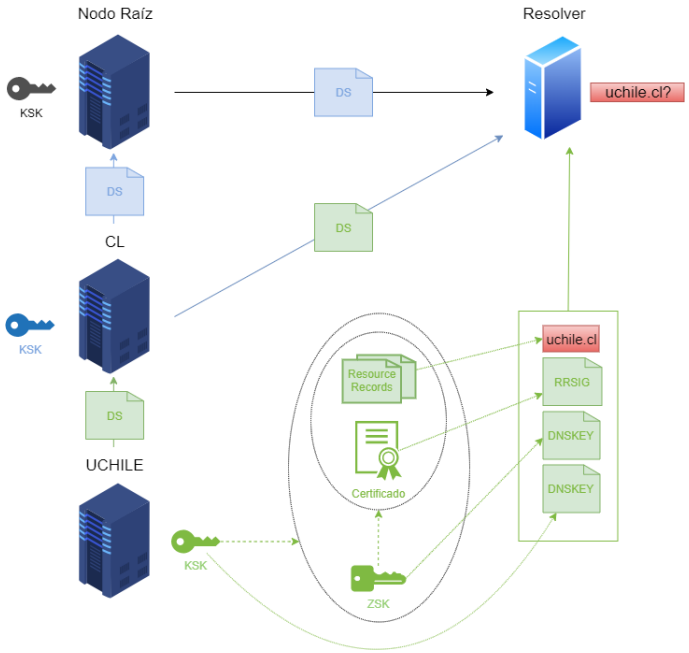


Figura 1.6: Diagrama de resolución en DNSSEC sin almacenamiento en cache previo. En esta extensión de seguridad para DNS, junto al dominio consultado se reciben también registros *RRSIG* y *DNSKEY*. Uno de los registros *DNSKEY* corresponde a la llave *ZSK* que firma el certificado recibido como registro *RRSIG* por parte del servidor, acreditando que es autoritativo para la zona. El otro registro *DNSKEY* corresponde a la llave *KSK* de la zona, que a su vez firma la llave *ZSK*. Este registro permite confirmar con su nodo padre (que posee aquella información en forma previa), que la zona pertenece a la cadena confianza desde el nodo raíz.

mente rápida, especialmente en los niveles más bajos del esquema jerárquico. Inclusive a la fecha de hoy existen algunos *ccTLD* que no han implementado DNSSEC. En la Figura 1.7 se muestra un mapa global con el nivel de implantación de DNSSEC en cada uno de los *cctld* en los años 2019 y 2014.

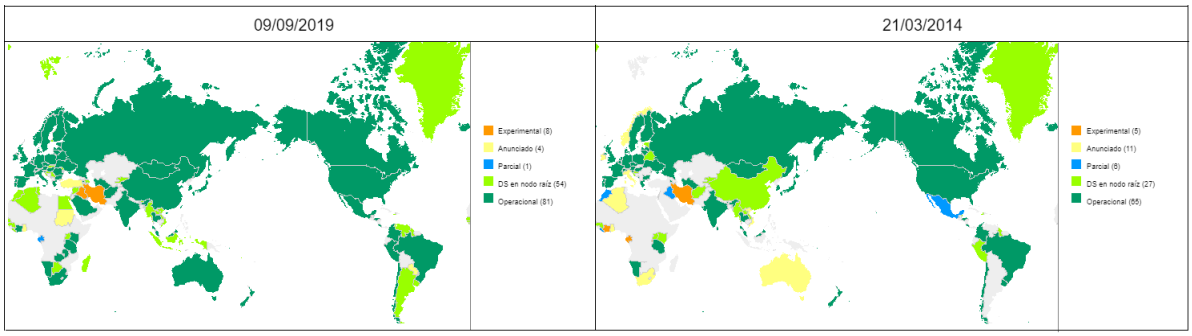


Figura 1.7: Estado de implementación de DNSSEC en dominios ccTLD [1]. El mapa de la izquierda muestra el estado de implementación (Experimental, Anunciado, Parcial, DS en nodo raíz u Operacional) de DNSSEC en cada país en el año 2019. Por su parte, el mapa derecho muestra aquella situación a la fecha de Marzo 2014.

1.3. Descripción de la Tesis

El presente trabajo pretende detectar eventos anómalos en tráfico DNS, mediante la experimentación con técnicas de *Machine Learning* para la predicción del tráfico. Diferencias apreciables entre lo esperado y lo predicho pueden indicar la ocurrencia de algún evento anormal. Con esto se busca detectar ataques a DNS, además de cualquier evento que pueda comprometer el funcionamiento del sistema; no solo para instancias de las que se tiene conocimiento previo, sino también para eventos maliciosos desconocidos que puedan manifestarse. Esto sería posible dado que el método a utilizar consiste en el uso de aprendizaje no-supervisado. No obstante, esto requiere una descripción correcta del sistema en base a agregaciones que permita tanto el procesamiento apto dado el amplio volumen de datos, como la persistencia de la información relevante para la detección. Para esto, se experimenta con datos reales de un servidor DNS, correspondiente al ccTLD de Chile: *.cl*.

1.3.1. Hipótesis

La hipótesis de este trabajo corresponde a que "es posible utilizar predicción de tráfico DNS para detectar eventos anómalos en él." Dado que esta técnica utiliza aprendizaje no-supervisado, se deriva una nueva hipótesis: "es posible utilizar aprendizaje no-supervisado para detectar eventos tanto conocidos como desconocidos." Esto último bajo el supuesto de que cualquier evento anómalo generará algún cambio en algún punto del flujo de tráfico.

1.3.2. Objetivos

Se listan a continuación el objetivo general de la presente Tesis, junto a los objetivos específicos que representan resultados independientes y que contribuyen a cumplir con el objetivo general.

Objetivo General

El objetivo general de este trabajo es detectar eventos anómalos en tráfico DNS utilizando predicción del tráfico mismo.

Objetivos Específicos

- Realizar una correcta descripción del comportamiento del sistema, en base a agregaciones de datos, que permita reducir el volumen de datos sin perder información necesaria para cumplir el objetivo general.
- Construir un modelo predictivo capaz de realizar predicciones del tráfico DNS con bajo error de acuerdo a las métricas de evaluación a utilizar.
- Emular ataques en el set de datos para evaluar el procedimiento de acuerdo al índice de detección de estas instancias de ataques.
- Definir una heurística que determine la ocurrencia de anomalías de acuerdo a los resultados obtenidos por el modelo predictivo.

1.3.3. Metodología

Utilizando metodología experimental se buscará comprobar la hipótesis anteriormente descrita.

Los datos con los que se llevará a cabo la experimentación corresponden a un mes completo de datos reales provenientes de uno de los servidores autoritativos del dominio *.cl*.

Se transformará el set de datos a múltiples series temporales en base a agregaciones del tráfico de DNS, como parte del pre-procesamiento necesario para suministrar la entrada del modelo predictivo.

Usando tres de las cuatro semanas de tráfico DNS se entrenará un modelo predictivo de *Machine Learning*, para que realice una predicción de la semana siguiente. Comparando con los valores reales de la cuarta semana del set de datos se evaluará cuantitativamente la predicción del modelo.

Realizando alteraciones al set de datos original mediante la emulación de diferentes ataques DNS en él, se comparará la predicción realizada del modelo predictivo, con los valores del nuevo set de datos con ataques.

Aplicando una heurística respecto a las diferencias encontradas, se determinarán las anomalías detectadas.

Se analizarán las detecciones encontradas a modo de evaluación del procedimiento utilizado.

1.3.4. Contribución

La principal contribución de esta Tesis es la evaluación de un procedimiento basado en aprendizaje no-supervisado para la detección de anomalías en tráfico DNS. Este procedimiento serviría como punto de partida para la implementación de un sistema detector de eventos maliciosos.

Por otro lado, la creación de una herramienta de emulación de ataques DNS, junto con el nuevo set de datos con ataques incluidos para su detección, permitirá realizar nuevos trabajos de investigación que lo utilicen.

Al realizar este trabajo con datos reales y de alta importancia al provenir de un ccTLD, se espera no solo que los resultados se adecúen al tráfico de este dominio, lo que por sí solo representa una contribución a la seguridad de los servidores DNS de Chile; sino que se adecúe también a otros dominios con características similares, tales como otros ccTLD.

Capítulo 2

Trabajo Relacionado

El área de la ciber-seguridad ha tomado especial relevancia en los últimos años. En particular, en el área de la investigación [24, 25]. Contenido en esto, el problema de la detección automática de eventos en tráfico de red ha sido abordado desde diversas perspectivas y metodologías en la literatura. A pesar de que este trabajo se centra en el tipo específico de tráfico de DNS, resulta conveniente analizar también los trabajos relacionados con otros tipos de tráfico de red. Esto es debido a que tales estudios proliferan a partir de tráfico general de red, y a que en general, se tienen muchas similitudes con el tráfico de DNS utilizado en este trabajo. Por ejemplo, la existencia de ataques comunes a diferentes tipos de tráfico, incluyendo DNS, como lo son el ataque de *denegación de servicio* (DoS) y el ataque de *denegación de servicio distribuido* (DDoS), o el *escaneo de puerto* (Port Scanning), los cuales interesa detectar. Adicionalmente, la fuente de la información, el volumen de datos, el muestreo y la agregación de datos, las características de los usuarios, entre otros, son factores que añaden un importante grado de singularidad a los trabajos de investigación relacionados que se debe tomar en cuenta, dificultando reconocerles como solución única a este amplio problema.

2.1. Ataques

Existen estudios que analizan ciertos ataques de manera particular, y proponen metodologías para detectarlos adecuadamente. Tales como DoS [20], DDoS [26, 33], los que corresponden a ataques que pueden a afectar tanto a servicios de red en general, como a servicios de DNS. Otros ataques específicos en contra del sistema de DNS son Domain Fluxing [49], Botnet Domains y Malware [47], o Kaminsky Cache Poisoning [35]. Estos trabajos otorgan importantes consideraciones sobre qué aspectos y porción de los datos deben ser tomados en cuenta al implementar nuestro sistema de detección.

2.2. Sistemas de Detección

Existen también trabajos que implementan sistemas de detección automática de ataques, intrusos o eventos anómalos en general. Vale mencionar sistemas que han subsistido y mejorado por muchos años hasta ahora como lo son SNORT [44], o BRO [40], los cuales realizan

detección automática de intrusos en la red, mediante el establecimiento y verificación de reglas determinadas que deben cumplirse para encontrar una ocurrencia.

Por su parte, los sistemas que se basan en modelos estadísticos o modelos de *Machine Learning*, cuyo procedimiento utiliza información previa, pueden ser separados en dos categorías:

1. Sistemas basados en aprendizaje supervisado: son aquellos que requieren un conocimiento previo de lo que se busca identificar. Se les entrena con varios ejemplos, indicando cuál es la resolución correcta para cada uno.
2. Sistemas basados en aprendizaje no-supervisado: son aquellos que no requieren un conocimiento previo, es decir, no saben a priori qué es lo que se busca identificar.

2.2.1. Basados en Aprendizaje Supervisado

La mayoría de las soluciones de detección automática de eventos en tráfico de red utilizan un procedimiento supervisado, entrenando los modelos con instancias de eventos previamente clasificados para reconocerlos en ocurrencias futuras en el tráfico.

En [14], se construyó un sistema capaz de variar el algoritmo de clasificación de acuerdo a la cantidad de tráfico de red entrante, con el objetivo de clasificar ataques de tipo DDoS, extrayendo de los paquetes percibidos información de protocolos IP, TCP, UDP y SSL.

El trabajo [36] aplica técnicas de Machine Learning a redes definidas por software (SDN), con tal de clasificar si el tráfico recibido por parte de cada IP corresponde a un posible ataque. De esta forma, si se predice que una IP produce actividad maliciosa, ésta será bloqueada.

Un trabajo que ocupa tráfico de DNS para detectar redes de bot [12], verifica diferentes reglas basadas tanto en características del *header* de los paquetes, como en el contenido de la respuesta. Verifica también la respuesta del servidor y compara con otros paquetes en datos históricos para identificar a un nuevo origen como miembro de una *botnet*. Sin embargo, el aspecto no-supervisado que utilizan es la detección de anomalías, mediante la comparación de la variación de la entropía contra un *threshold* no explicitado.

2.2.2. Basados en Aprendizaje No-Supervisado

Algunos trabajos de investigación han abordado el problema utilizando métodos de aprendizaje no-supervisado, con el propósito de reconocer anomalías que comprenden no sólo amenazas conocidas, sino también desconocidas.

Casas [16] implementa UNIDS, un sistema basado completamente en aprendizaje no-supervisado, con el objetivo de encontrar intrusos en tráfico de red. Este se basa en algoritmos de detección de cambio, *clustering* y filtración mediante rankeo y comparación con threshold. UNIDS se compara mediante su aplicación en diferentes sets de datos de fuente abierta con buenos resultados, con respecto a ciertos algoritmos con los que se comparó. Sin embargo, varios de estos muy típicos sets de datos son ya antiguos (KDD99(1999), NSL-KDD(2009)), aunque se sigan utilizando en publicaciones recientes.

El trabajo [13], utiliza una primera etapa de aprendizaje no-supervisado mediante una red neuronal del tipo *Spike Neural Network*, con el objetivo de identificar ataques en el set de datos NSL-KDD. Luego añade aprendizaje supervisado al intentar clasificar estos ataques detectados. La comparación de los resultados también se realiza contra otros reconocidos algoritmos.

Este es el enfoque que se adopta en la presente Tesis. Con la diferencia de que se aplicará una herramienta de predicción como método de aprendizaje no-supervisado, con un enfoque más específico en tráfico de tipo DNS, utilizando datos reales de un *ccTLD*.

2.3. Trabajos en ccTLDs

InternetNZ, actuales administradores del ccTLD *.nz* de Nueva Zelanda, mostraron en su blog el uso del modelo predictivo *Prophet* [46] para el análisis de tendencias en consultas DNS por el dominio *.nz* [42]. También buscaron mediante inspección visual, anomalías dentro de su set de datos.

Cercano a los objetivos de este trabajo, la gente de KU Leuven, utilizando datos pertenecientes al ccTLD *.be* de Bélgica, detallaron la implementación de su sistema detector de anomalías: QLAD [43]. Ellos intentaron encontrar eventos anómalos en una Serie de Tiempo multidimensional, correspondiente a particiones de sus *logs* de DNS. Un procedimiento similar es realizado en esta Tesis. No obstante, con la importante diferencia de que KU Leuven analiza datos no etiquetados, mientras que en esta Tesis se añadieron ataques simulados con el propósito de evaluar el algoritmo de detección.

Adicionalmente, en la implementación de los trabajos relacionados se utilizan modelos estadísticos basados en regresiones y variación de entropía respectivamente, en contraposición al método de Machine Learning explorado en este trabajo. La diferencia entre ambos enfoques usados para detección de anomalías recae principalmente en el objetivo de esta detección. Los métodos estadísticos se enfocan en inferir información acerca de las relaciones que existen entre variables presentes en los datos, mientras que la complejidad en las operaciones de las soluciones basadas en Machine Learning dificultan esta tarea al actuar como *cajas negras*, en virtud de realizar mejores predicciones a partir de los datos.

2.4. Método Predictivo

La predicción ha sido también estudiada en la literatura como un importante método para soluciones autónomas de sistemas computacionales [28]. En particular, el uso de redes neuronales recurrentes para este fin ha sido utilizado en detección de anomalías con resultados prometedores. El tipo más popular de estas redes es *LSTM* (Long Short-Term Memory), la cual se utiliza también en este trabajo. Algunos de los escenarios donde este modelo ha realizado detección de anomalías satisfactoriamente son: en datos provenientes de una nave espacial de la NASA [27], señales de electrocardiografía (ECG) [17] y lecturas de sensores de motores [32].

2.4.1. Patrones Humanos

El entendimiento del comportamiento humano en las ciencias de la computación es un tema emergente que se ha beneficiado significativamente de la proliferación de dispositivos móviles que frecuentemente reportan actualizaciones de estado y localización [15].

Los trabajos en el estado del arte que abordan este tema a través de datos de redes aprovechan especialmente la periodicidad presente en el tráfico. Esta alta periodicidad, y en consecuencia, baja entropía, es mayormente atribuida al impacto de la regularidad de los patrones humanos [22, 37] en el estado de la red [48].

Estos patrones resultan convenientes para el método predictivo, debido a que la periodicidad es identificable por los modelos.

2.4.2. Series de Tiempo

El análisis de series temporales se ha vuelto, por sí mismo, un tópico popular en investigación recientemente. Especialmente por su aplicabilidad en temas muy recurridos, como finanzas; y debido a que conceptos como *similarity* y *summarization* tienen diferentes visiones dependiendo del problema. Sobre eso, las técnicas comunes de Machine Learning han recibido varias adaptaciones [21] a esta configuración de datos, donde en general, cada problema es abordado con un procedimiento original dependiendo de sus condiciones.

2.5. Trabajo Previo

Este trabajo surge como una implementación experimental parcial de un sistema de detección teórico propuesto en un trabajo de mi autoría, que discute las etapas necesarias para llevar a cabo en el objetivo de detectar eventos anómalos en tráfico DNS [38].

Por otro lado, este trabajo usa directamente las conclusiones obtenidas de nuestro trabajo previo junto a miembros de NICLabs [31], donde se demuestra la factibilidad de predecir tráfico DNS usando el modelo LSTM, en el cual se obtuvieron resultados satisfactorios de acuerdo a diferentes medidas de evaluación para series temporales. Esto se demostró mediante la comparación con otros modelos de predicción estadísticos, discutiendo las diferencias en el comportamiento demostrado por los modelos respectos a los resultados obtenidos durante la experimentación sobre tráfico DNS del mismo origen al utilizado en el presente trabajo.

Finalmente, la existencia de patrones humanos presentes en nuestros datos de DNS se encuentra analizada a fondo en otro trabajo de mi autoría, utilizando datos provenientes de la misma fuente [39]. En él, se utilizan métodos de *clustering* y *reglas de asociación* para extraer estos patrones, hallando grupos y relaciones de acuerdo a la actividad de los usuarios.

2.6. Resumen

En vista de lo expuesto anteriormente, siendo el problema de detección de anomalías en tráfico de red un tema muy popular de manera reciente, falta investigación en torno al tráfico

específico de DNS. En efecto, la mayoría de la investigación trabaja con tráfico de red en general. Ya sea con capturas de tráfico privado, o utilizando set de datos abiertos masificados.

Por otro lado, a pesar de que recientemente se ha comenzado a observar trabajos basados en aprendizaje no-supervisado, continúan siendo la minoría en comparación a algoritmos de clasificación o validación de reglas, especialmente en trabajos relacionados a tráfico DNS.

La tabla 2.1 ordena los trabajos relacionados discutidos en detección automática de anomalías en base a soluciones de Machine Learning.

Trabajos de Detección de Anomalías	Tráfico de Red General	Tráfico DNS
Aprendizaje Supervisado	[14, 36]	[12]
Aprendizaje No-Supervisado	[16, 13]	

Tabla 2.1: Comparación de trabajo relacionado sobre detección de eventos anómalos en tráfico de red basados en Machine Learning, de acuerdo al enfoque utilizado por los modelos de detección y al tipo de tráfico de red sobre el que se desempeñan.

Este trabajo utiliza datos reales pertenecientes a un ccTLD, aplicando un método de predicción de Machine Learning no antes probado en el contexto de DNS. Conjuntamente, los datos se presentan como series temporales, una estructura de datos con muchos aspectos abiertos dentro de la investigación. Todo esto contribuye a la singularidad del presente trabajo de Tesis respecto del resto de trabajos relacionados.

Capítulo 3

Descripción de Datos

En este capítulo se presenta una descripción de los datos que serán utilizados para la experimentación a realizar posteriormente en este trabajo. Junto a características básicas como tamaño y periodicidad, se detalla la configuración del servidor DNS del cual se obtuvieron, y el proceso de creación de un nuevo set de datos a partir de la modificación del original en base a emulación de ataques y se muestran patrones interesantes en los datos, convenientes para el desempeño de los modelos predictivos.

El servidor DNS del cual provienen los datos utilizados en esta Tesis, corresponde a un servidor autoritativo del ccTLD de Chile *.cl*. Estos datos corresponden a tráfico real obtenido a partir de auténticas consultas de usuarios a este dominio. El administrador oficial actual del dominio es NIC Chile, quien facilitó los datos para este trabajo. El servidor registra todos los paquetes DNS enviados y recibidos por él cada 10 minutos en un archivo log.

Los datos contemplan un mes completo de funcionamiento de este servidor, contenidos entre los días 2 de Octubre de 2017 y 1 de Noviembre del mismo año. El servidor recibe en promedio aproximadamente 5000 consultas por minuto.

3.1. Configuración del servidor DNS

NIC Chile hace uso de *IP Anycast* para generar *nubes* de servidores respondiendo por una misma IP, de acuerdo a lo detallado en la Sección 1.2.6. De esta forma, NIC Chile distribuye geográficamente los servidores pertenecientes a una misma nube para balancear la carga efectivamente al atender consultadas realizadas desde distintos continentes. Actualmente poseen tres *nubes anycast* que agrupan alrededor de 20 servidores autoritativos DNS.

Recientemente, se está realizando un esfuerzo por compartir estas *nubes anycast*, regionalmente, con otros *ccTLD* de Latinoamérica y el Caribe [3].

En la Figura 3.1 se observan los servidores administrados de manera independiente por NIC Chile para el dominio *.cl*.

El servidor del que se obtuvieron los datos para este trabajo corresponde a uno de los

servidores incluido en una *nube anycast*, el cual se encuentra localizado en Santiago, Chile.

Figura 3.1: Mapa de Servidores DNS administrados por NIC Chile [7].

3.2. Emulación de Ataques

Con el objetivo de realizar una apropiada evaluación de la metodología posteriormente descrita, se llevó a cabo una emulación de ataques DNS, en colaboración con gente de NICLabs. Estos ataques serían incluidos dentro del tráfico real, con el propósito de que el sistema de detección fuese capaz de detectarlos, buscando validar la hipótesis de este trabajo. Como producto de esta emulación e inserción de ataques, resulta un set de datos etiquetado de ataques en tráfico DNS, el cual puede también ser usado para fines de investigación en trabajos futuros.

Diferentes ataques fueron emulados en el set de datos, los cuales se explican en las subsecciones siguientes. La herramienta implementada permitió generar múltiples instancias para cada uno, variando sus tiempos de duración y magnitudes.

Se buscó realizar las simulaciones de la forma más apegada a la realidad posible. En efecto, dado que los datos se tienen como captura de paquetes en formato *.pcap*, lo que se hizo fue generar cada uno de los paquetes que componen a un determinado ataque. Los que fueron insertados en los archivos originales, entre consultas y respuestas DNS auténticas, de acuerdo al tiempo correspondiente.

Para cumplir con esto, a los paquetes generados se les otorgó valores para los campos de Ethernet en base a lo contenido dentro de los paquetes originales del set de datos. Por otra parte, de acuerdo a lo observado en las distintas granularidades de las respuestas que el servidor estudiado realizaba, como se muestra en la Figura 3.2, se estimó la capacidad máxima que tenía el servidor. En adición, la Tabla 3.1 indica el promedio y desviación estándar de los tiempos de respuesta del servidor dentro del set de datos para los diferentes códigos de respuesta. Con estas estimaciones, se determinó la frecuencia y el rango de tiempo para los timestamps de envío y respuesta de los nuevos paquetes generados. En específico, se utilizaron números aleatorios dentro de una distribución Gaussiana con promedio y desviación estándar

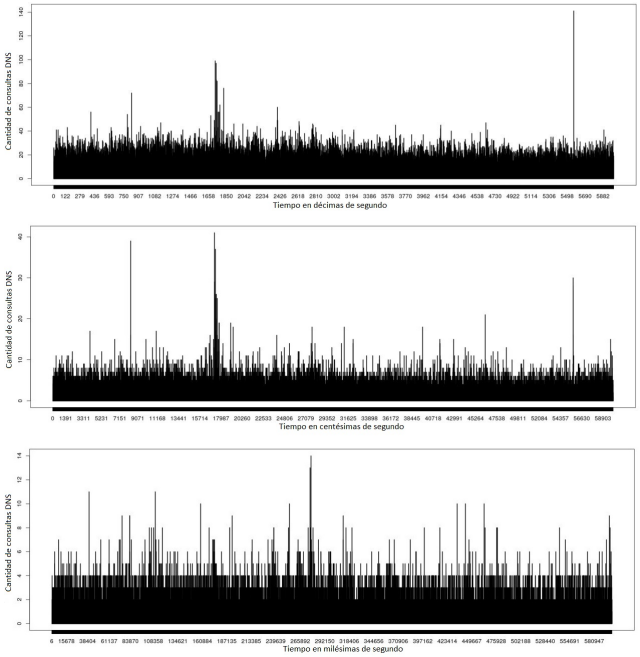


Figura 3.2: Consultas DNS según granularidad de tiempo. Cada gráfico muestra una cantidad de consultas en una agregación de tiempo diferente para el servidor DNS observado. El gráfico superior muestra cantidad de consultas por décima de segundo, el central en centésimas de segundo y el último en milésimas de segundo.

siguiendo los valores indicados en la Tabla 3.1 dependiendo del código de respuesta que se debe utilizar. Con el fin de añadir mayor realismo a la emulación.

Dependiendo del ataque específico que se emularía, se asignarían direcciones IP y puertos válidos, tanto para el emisor y el receptor de los paquetes. Considerando también los ataques que se basarían en botnets, usando múltiples direcciones IP diferentes para realizar ataques distribuidos. O los que requerirían *spoofing* en su IP de origen.

En las subsecciones posteriores se explican en detalle, los tres ataques que fueron emulados para la posterior experimentación: *Random Subdomain*, *UDP Flood* y *DNS Amplification*. Estos fueron escogidos de acuerdo a la factibilidad de su emulación; y junto a la factibilidad de ser recibidos a nivel de servidor de ccTLD, tal como el origen de los datos utilizados en el presente trabajo.

RCODE	Promedio [s]	Desviación estándar
0 (NOERROR)	0.00032	0.018
1 (FORMERR)	0.00010	0.000019
3 (NXDOMAIN)	0.00023	0.000036
5 (REFUSED)	0.00019	0.000027

Tabla 3.1: Estadísticas en tiempo de respuesta según RCODE, para todo el set de datos provenientes del servidor DNS utilizado.

3.2.1. Random Subdomain

El ataque *Random Subdomain*, también conocido como *DNS Water Torture* o *Non-sense Name*, es un tipo de *DDoS (Ataque de Denegación de Servicio Distribuido)*. Tiene como objetivo al servidor autoritativo de un dominio en particular, con tal de causar una falla o deficiencia. Sin embargo, debe realizarse de forma distribuida debido a que afecta también a los servidores recursivos intermedios en el proceso de resolución [23], ya que las consultas generadas no levantan coincidencias en los *cache* de los resolvers, dado su contenido.

El atacante, desde múltiples fuentes, envía consultas por subdominios generados aleatoriamente, mas terminados en un dominio válido correspondiente al de la víctima. De esta forma, se asegura a nivel del protocolo DNS que la respuesta llegue a un servidor autoritativo para ese dominio. Los subdominios aleatorios causan que el servidor deba realizar una búsqueda distinta por el subdominio dentro de sus *resource records*. El servidor almacenará inútilmente en su *cache*, la no-existencia de cada uno de estos subdominios. Por otro lado, en la gran mayoría de los casos, estos subdominios generados causarán una búsqueda infructuosa, lo que es costoso para el servidor. Esto se ilustra en la Figura 3.3.

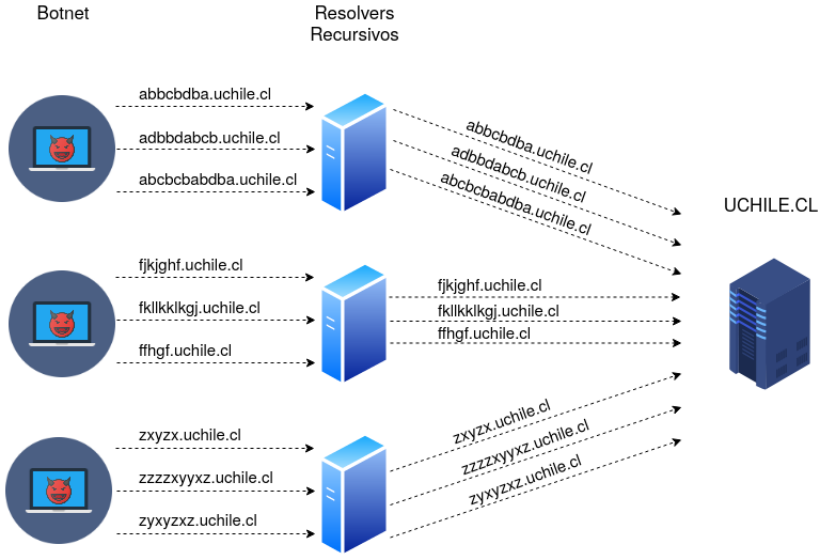


Figura 3.3: Ejemplo de generación aleatoria de consultas distribuidamente a través de una botnet para un ataque Random Subdomain al dominio *uchile.cl*. Esto provoca alta carga, dada la cantidad de búsquedas infructuosas y almacenamiento inútil en caché, en el servidor atacado.

A pesar de que existe trabajo relacionado que ofrece soluciones a este ataque [19], estos necesitan establecer una manera de determinar si un subdominio fue generado aleatoriamente o no, evitando el tener que generar una respuesta. Esto puede ser de igual forma costoso, sobretodo si se utilizan algoritmos un poco más sofisticados de generación aleatoria de subdominios, con tal de eludir esta clasificación. En la práctica, lo que se hace es suspender la resolución de la zona DNS atacada temporalmente, o bloquear la IP de los atacantes, afectando también a otros clientes o servicios. Por otro lado, algunas organizaciones utilizan ocasionalmente subdominios generados aleatoriamente para sus propios fines, los cuales desechan luego de su momentáneo propósito.

3.2.2. UDP Flood

El ataque de UDP Flood (inundación UDP), utiliza características específicas del protocolo UDP para generar un eficaz ataque de tipo DoS. El protocolo UDP, al estar orientado a aplicaciones donde la confiabilidad de las transmisiones no es crítica, carece de conexión y sesión, a diferencia del protocolo TCP. Junto a esto, el protocolo UDP no define un formato de paquetes, haciendo posible generar paquetes de gran volumen rellenos con basura. De esta forma, es posible enviar altas cantidades de tráfico a un mismo destino de forma directa, con bajo costo de generación. Se puede, incluso, utilizar paquetes duplicados.

Como DNS se basa en el protocolo UDP, escuchando en el puerto 53, los servidores de nombres siempre se encuentran expuestos a este ataque. Donde la recepción de esta enorme cantidad de paquetes, afecta el rendimiento del servidor. Aun logrando descartar los paquetes maliciosos, se percibe un alto consumo de ancho de banda. Limitar las respuestas generadas afectaría inevitablemente al tráfico real.

Más aún, la IP de origen de los paquetes es fácilmente alterada (Spoofing), anonimizando al atacante o red de atacantes, y desviando las respuestas entregadas por el servidor. Esto dificulta, de igual manera, cualquier intento de bloquear el servicio a la procedencia de los paquetes.

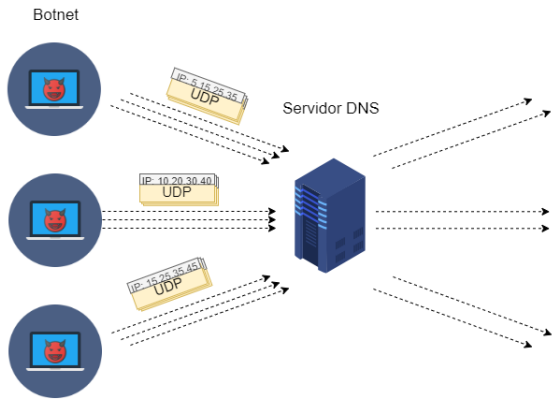


Figura 3.4: Ejemplo de ataque UDP Flood a un servidor DNS mediante uso de paquetes UDP con uso de Spoofing en la IP de destino, para dirigir las respuestas a otras direcciones (5.15.25.35, 10.20.30.40 y 15.25.35.45 en el ejemplo), con tal de anonimizar y facilitar la generación del ataque.

Las estrategias que se utilizan para mitigar este ataque, son mediante inspección de los paquetes, observando su contenido. Ya que estos generalmente contienen datos sin significado y se repiten, se crean clasificadores sobre el tráfico UDP percibido, con el objetivo de bloquear posibles fuentes del ataque. Sin embargo, esto sigue resultando muy costoso para el servidor atacado.

En otros escenarios fuera de DNS, es común realizar este ataque enviando paquetes tipo UDP con destino a múltiples puertos, obligando al objetivo a analizar los puertos disponibles, en virtud de la generación de un paquete de tipo ICMP, indicando la no-disponibilidad de un servicio en el puerto consultado.

3.2.3. DNS Amplification

La amplificación de DNS consiste en el uso del protocolo para generar paquetes de gran tamaño a partir de consultas DNS de menor tamaño. Es un tipo de ataque de *Denegación de Servicio Reflectante (RDOS)*, pues fuerza a resolvers recursivos o servidores autoritativos a participar inconscientemente del ataque a un servidor en específico.

La IP de destino es modificada en los paquetes DNS enviados por el o los atacantes, reemplazándola con la IP del servidor víctima. De esta forma, cuando la consulta sea recibida por un resolver, éste entregará la respuesta a aquel servidor.

Este ataque busca que cada consulta enviada produzca una respuesta de mayor tamaño. En consecuencia, para el servidor víctima será mucho más costoso procesar los paquetes que lo que cuesta al atacante su generación. Una forma efectiva de llevar esto a cabo, es utilizar el tipo de registro *ANY*. La consulta DNS de este tipo tiene por respuesta el conjunto completo de *resource records* pertenecientes a una zona DNS. Esto puede causar una amplificación de paquetes de 40 bytes hasta un máximo de 4096 bytes. El proceso se ve ilustrado en la Figura 3.5.

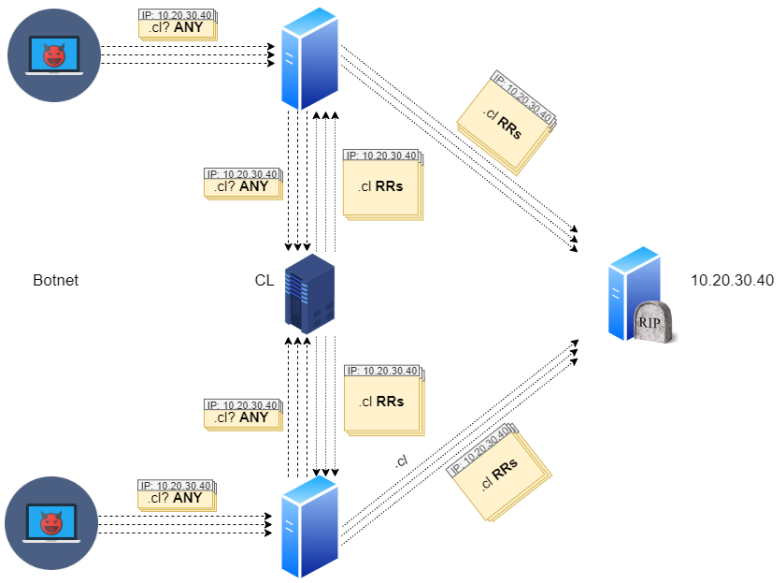


Figura 3.5: Ejemplo de uso de botnet para realizar un ataque DNS Amplification al servidor con dirección IP 10.20.30.40 mediante uso de consultas tipo ANY al dominio *.cl*. Las consultas provocan unas respuestas de tamaño mucho mayor, provocando tráfico de mayor volumen dirigido hacia el servidor atacado.

Si bien el tipo de registro *ANY* pasó a ser opcional de forma reciente debido precisamente a esta causa [10], existen otros tipos de consultas posibles que producen respuestas de mayor tamaño. Tales como las extensiones DNSSEC (ver Sección 1.2.8), o EDNS, que permite una ampliación del tamaño en varios campos del paquete. Con estas extensiones es incluso posible amplificar simples consultas de tipo *A* en un radio de 16.37 %. Para consultas *ANY* este aumento se aproxima a 32.77 % [30].

3.3. Patrones de Comportamiento Humano

El tráfico DNS está determinado por la actividad de los usuarios en Internet. Es debido a esto que es posible encontrar patrones de comportamiento humano en esta actividad. En especial al considerar que el objetivo principal de DNS es permitirnos identificar servicios de Internet con nombres entendibles para los humanos, donde el contenido de la consulta nos dice mucho de la actividad que está realizando el usuario.

A modo de ejemplificación, en la Figura 3.6 se muestra la cantidad total de consultas realizadas a nuestro servidor DNS durante la primera semana de datos. Es fácil identificar visualmente aquí un patrón común de comportamiento humano, que corresponde a un aumento de actividad durante el día, y contrariamente, un descenso de actividad de noche cuando la mayoría de la gente descansa. En efecto, cada uno de los 'peaks' de consultas presentes en el gráfico corresponde a un día de la semana. En adición, también es posible distinguir los días que corresponden a fin de semana, denotados por una menor actividad en comparación a los días de la semana. Dicho comportamiento también es observado y se menciona en los trabajos relacionados a ccTLD de otro países [43, 42].

Este es un ejemplo de patrón periódico que resulta conveniente para el modelo predictivo, ayudándole a reconocer lo que es el comportamiento normal del sistema, con el fin de realizar una más precisa predicción del tráfico. Lo que se necesita para lograr el objetivo de este trabajo. En efecto, en caso de ocurrir alguna anomalía, esta debería resaltar por sobre los patrones observados en una actividad regular, facilitando su detección.

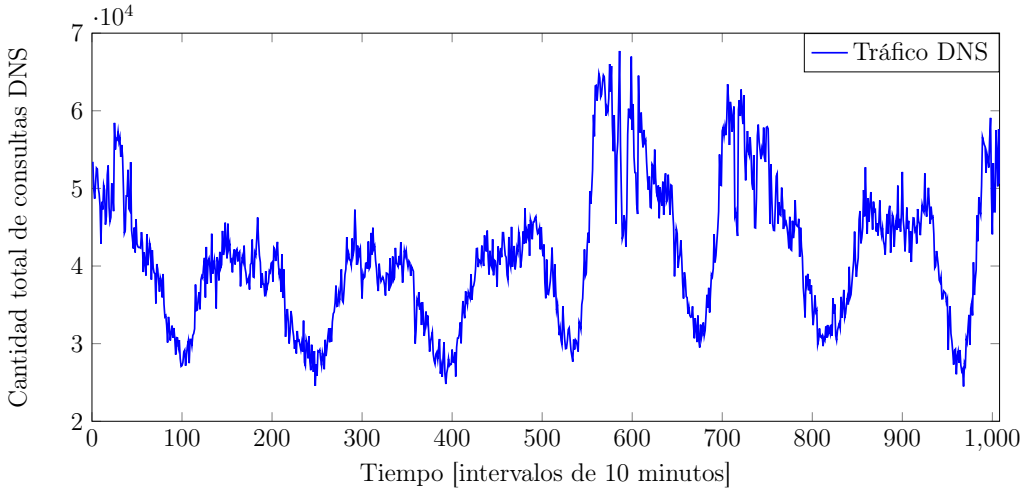


Figura 3.6: Serie de tiempo con cantidad de consultas recibidas cada 10 minutos por el servidor DNS observado en un intervalo de una semana. Se observa en esta figura el patrón periódico dado por la actividad de los usuarios durante el día y durante la noche a lo largo de la semana.

Capítulo 4

Modelo y Evaluación

En este capítulo se mencionan todas las herramientas y métricas utilizadas para llevar a cabo la experimentación de este trabajo y obtener los indicadores de validación y comparación de resultados. Junto a un marco teórico y explicación del funcionamiento de cada técnica.

4.1. Redes Neuronales

Una red neuronal artificial consiste en un modelo computacional inspirado en la propia red de neuronas del cerebro humano, y que tiene por objetivo resolver los problemas de la misma manera en que resuelve aquella red. De esta forma, este sistema se compone de conexiones entre unidades neuronales simples, en que los enlaces incrementan o inhiben el estado de activación de las neuronas adyacentes, determinando la propagación de las señales a través de las redes y enlaces. Estos sistemas están diseñados para reconocer patrones en los datos que procesa.

Cada neurona en particular posee un conjunto de entradas, análogo a las dendritas en su versión biológica. Otorgando a este vector de entradas un vector de pesos sinápticos, que se puede definir como la intensidad o importancia de la interacción entre neuronas mediante sus enlaces, se alimenta una función de activación que proporciona el estado de activación de la neurona. Con este estado se determina una salida, que afecta los enlaces y salida de la red completa, análogamente al axón de las neuronas biológicas. Así, la red también puede verse como un conjunto de capas. Definiendo como capa de entrada a la conformada por neuronas que reciben datos o señales procedentes del exterior de la red. Asimismo, la capa de salida contendría a las neuronas que entregan las señales al exterior, funcionando como la capa que proporciona la respuesta de la red a los estímulos sobre capa de entrada. La capa oculta es quien no recibe ni suministra información fuera de la red, es el procesamiento interno.

Las redes neuronales tienen la capacidad de aprender y formarse a sí mismas, por lo que corresponden a un campo muy importante dentro de la Inteligencia Artificial. En adición, al haber realizado ya el aprendizaje, pueden congelar sus pesos, suspendiéndose y almacenando la información, la cual puede ser transferida a otros modelos, o modificada mediante nuevos estímulos.

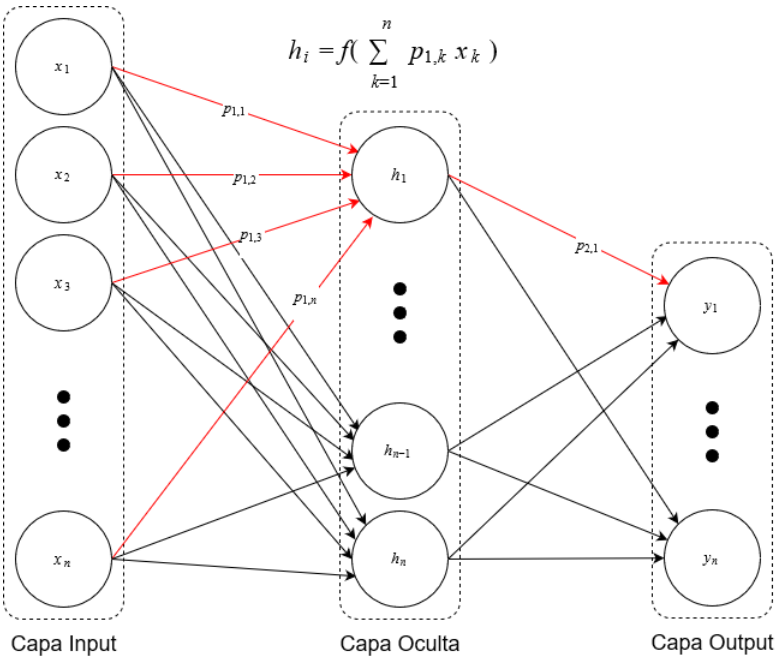


Figura 4.1: Esquema de red neuronal con 3 capas completamente conectadas. Con $p_{\mathrm{i},j}$ indicando el peso asociado a cada conexión, y $f(x)$ correspondiente a la función de activación. Las neuronas de las capas Input, Oculta y Output se ven representadas por x_i, h_i y y_i respectivamente. Se denotan en rojo todas las conexiones que influyen en la activación de la neurona de la capa Ouptut y_1.

Función de Activación

La función de activación determina si es que una neurona transmitirá una señal a la siguiente capa en la red neuronal, y con qué intensidad lo hará. La función es elegida dependiendo del contexto, existiendo funciones convenientes y/o populares para problemas particulares. Sin embargo, esta función debe ser diferenciable y, regularmente, no-lineal. Esto responde a que se debe cumplir el proceso de *backpropagation*, basado en derivadas parciales con respecto a la función de costos, donde una función lineal causaría una derivada constante, causando una variación de pesos independiente del input. Algunas funciones ampliamente utilizadas son:

1. Sigmoide:
$$sigm(x) = \frac{1}{1+\mathrm{e}^{-x}}, \qquad Rec_{sigm} = (0,1) \tag{4.1}$$

2. Tangente Hiperbólica:
$$tanh(x) = \frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}}, \qquad Rec_{tanh} = (-1,1) \tag{4.2}$$

3. ReLu:
$$relu(x) = \begin{cases} 0 & \text{para} \quad x < 0 \\ x & \text{para} \quad x \geq 0 \end{cases}, \qquad Rec_{relu} = [0,\infty) \tag{4.3}$$

Backpropagation

Si bien existen alternativas recientes, la familia de algoritmos que utiliza el descenso del gradiente llamada backpropagation, es la más comúnmente utilizada para actualizar los pesos de las unidades neuronales. En ellos se calcula un costo asociado a cada par de input/output que recibe la red durante el periodo de entrenamiento, conforme a una *función de costo* C.

La actualización del peso de cada neurona en particular se encuentra definida por

$$\Delta p = \alpha \frac{\partial C}{\partial p} \tag{4.4}$$

Donde α corresponde al valor *taza de aprendizaje*, acotado entre 0 y 1.

Es aquí donde toma gran relevancia la selección de la función de activación, puesto que al calcular la derivada para el peso de una neurona, esta encadena las funciones de activación del resto de las neuronas en el camino desde el output hasta el objetivo. Pudiendo caer en lo que se conoce como *desvanecimiento de gradiente* y *gradiente explosivo*.

Por último, el *optimizador* se encarga de los resultados obtenidos por backpropagation para agilizar y/o mejorar el proceso de entrenamiento. Por ejemplo, utilizando el promedio de gradientes, para evitar realizar actualizaciones opuestas que se cancelen mayormente entre ellas; o adaptando el valor de α para realizar actualizaciones de mayor o menor magnitud. Uno de los algoritmos optimizadores más populares es *Adam*, el cual mezcla ambas de estas funcionalidades.

Dropout

Las capas de dropout pueden ser agregadas con el fin de evitar el sobre-entrenamiento (overfitting). Esto es, ajustarse en demasía a los datos de entrenamiento, perdiendo capacidad de generalización. Estas capas literalmente eliminan un conjunto aleatorio de activaciones, dándoles un valor 0 en el paso siguiente. El objetivo de esto es obligar a la red a ser redundante, es decir, que entregue una buena respuesta a pesar de perder algunas de sus activaciones. Esta capa solo se utiliza durante el entrenamiento, no durante la prueba.

Redes Recurrentes

Un tipo particular de redes neuronales, corresponde a las redes neuronales recurrentes (RNN). Esta clase de red busca mantener una memoria interna en su procesamiento, mediante la inclusión de *loops* (ciclos) dentro de sus conexiones. Esto permite integrar en forma directa la información recibida por otros input previos al actualmente procesado.

En la Figura 4.2 se muestra un esquema de una red neuronal recurrente, en una forma básica de incorporación de output previo en el modelo. Aquí, se clona el output resultante de la capa oculta en nuevas neuronas, cuya salida se re-incorpora nuevamente como entrada a la capa oculta. Creando ciclos de información.

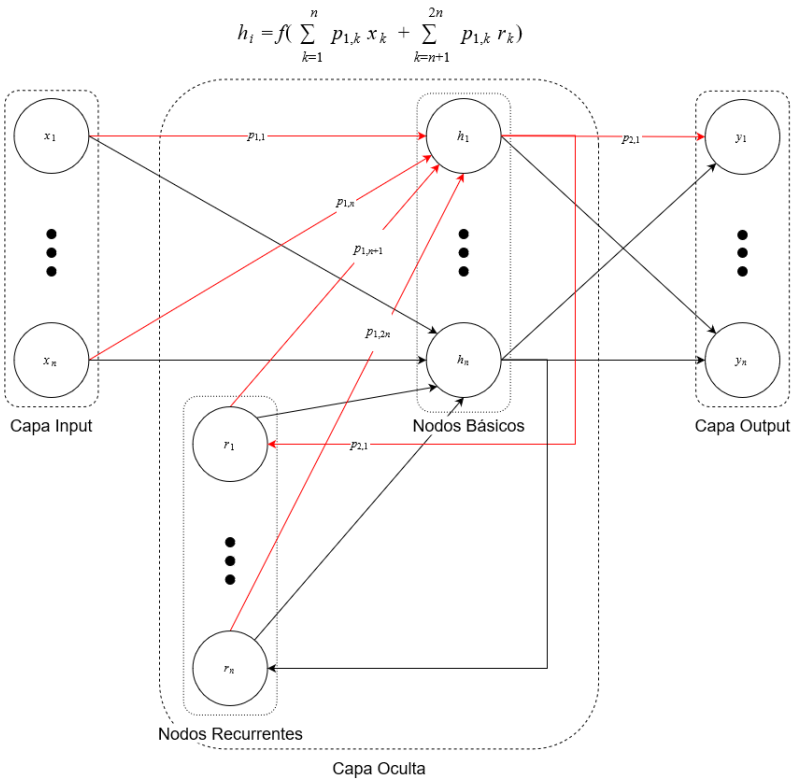


Figura 4.2: Esquema de red neuronal con capa oculta recurrente respecto al último output generado. Con $p_{i,j}$ indicando el peso asociado a cada conexión, y $f(x)$ correspondiente a la función de activación. Las neuronas de las capas Input y Output se ven representadas por x_i y y_i respectivamente. Las neuronas pertenecientes a la capa oculta se denotan por h_i y r_i; donde estas últimas corresponden a las originadas por la copia del último output de la capa.

Este tipo de redes han resultado ser muy exitosas en aplicaciones como en NLP (Procesamiento de Lenguaje Natural), procesamiento de audio, y en lo que concierne a este trabajo: predicción y reconocimiento de patrones en series temporales.

Existen varias formas de implementar redes neuronales recurrentes, un tipo particular de estas implementaciones es la red LSTM, la cual se detalla en la siguiente sección y corresponde a la utilizada para la experimentación posterior en este documento.

4.2. LSTM

Long Short-Term Memory (LSTM) es un tipo de red neuronal recurrente. Esta red nace con el propósito de solucionar un importante problema para cierto tipo de aplicaciones: el uso de memoria de largo plazo. Observando nuevamente la Figura 4.2, es posible notar que la red será capaz de incorporar de buena forma la información recopilada en el último ejemplo. Sin embargo, si es relevante la información de dos periodos atrás, sería conveniente tener una nueva copia del penúltimo ejemplo en el sistema. De esta forma, cada vez se vuelve más costoso añadir información más lejana en el tiempo. La red LSTM busca mantener la información de largo plazo añadiendo un nuevo flujo de información a través de las neuronas, capaz de

fluir fácilmente, pero siendo modificada de forma controlada por compuertas especialmente diseñadas con este fin. A este flujo se le llama *estado de largo plazo*.

En la Figura 4.3 se presenta un esquema con la configuración interna de una neurona individual del tipo LSTM. En ella se muestran las interacciones entre los diferentes flujos de información que entran y salen de la neurona (L, X y R), con las operaciones y funciones de activación involucradas a través de las tres diferentes compuertas que caracterizan a las neuronas LSTM: compuerta *forget* o de olvido, compuerta *input* y compuerta *output*.

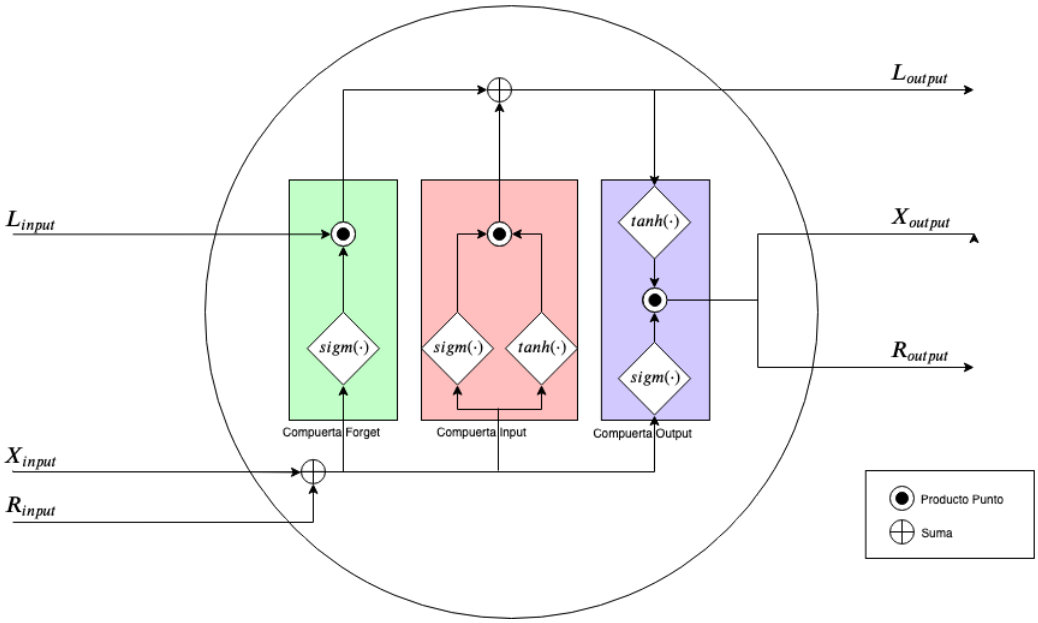


Figura 4.3: Esquema de unidad individual de neurona tipo LSTM. El flujo descrito por L corresponde al estado de largo plazo. El input recibido por la neurona se denota por X. R es la parte recurrente o de corto plazo. Los círculos corresponden a operaciones y los rombos a funciones de activación. Finalmente, en colores se diferencian las compuertas *forget*, *input* y *output*.

La primera compuerta que afecta el estado de largo plazo L, corresponde a la compuerta *forget*, cuyo objetivo es remover información de largo plazo. Esto se logra mediante la función sigmoide (4.1), que arroja valores acotados entre 0 y 1, a partir de los otros input de la neurona. Operando con el producto, L es disminuido o no afectado. Lo que se expresa como:

$$f_t = sigm(X + R) \cdot L$$

$$X = \sum_{\mathrm{i}=1}^{n} p_{c,\mathrm{i}}\ x_{\mathrm{i}}, \qquad R = \sum_{\mathrm{i}=n+1}^{2n} p_{c,\mathrm{i}}\ r_{\mathrm{i}}, \qquad L = f_{t-1} + \mathrm{i}_{t-1} \tag{4.5}$$

donde f_t e i_t son el valor para las compuertas forget e input respectivamente, en el paso de tiempo t. Los pesos de las conexiones que entran a la capa c se denotan por $p_{c,\mathrm{i}}$ y n es el número de conexiones de input que entran. x_{i} refiere a las neuronas de input i y r_{i} a la neurona de recurrencia i.

La compuerta input define qué es agregado y almacenado en la información de largo plazo. Igualmente regulando mediante una función sigmoide el flujo de información que circula hacia el estado de largo plazo, el cual proviene de otra función de activación, típicamente tangente hiperbólica (4.2). De acuerdo a la siguiente ecuación:

$$\mathrm{i}_t = sigm(X + R) \cdot tanh(X + R) \tag{4.6}$$

Ambos resultados son sumados, lo que resulta en el estado de largo plazo que recibirá la neurona en la siguiente iteración del entrenamiento. Por otro lado, este valor es considerado en la compuerta output, luego de una función de activación; también usualmente la función *tanh*. Lo que se multiplica con el resto del input de la neurona nuevamente, controlado por la función *sigm*. Esto constituye finalmente el output de la neurona y el input para la parte recurrente de corto plazo:

$$o_t = tanh(f_t + \mathrm{i}_t) \cdot sigm(X + R) \tag{4.7}$$

con o_t representando el valor para la compuerta output.

4.3. Evaluación

Con el objetivo de comparar los resultados obtenidos por el modelo de predicción propuesto, es necesario establecer métricas a considerar durante el proceso de evaluación. Las métricas utilizadas se dividen en dos grupos: error y distancia. Las cuales contemplan visiones diferentes respecto a la comparación de series temporales.

4.3.1. Error

En primer lugar, es necesario evaluar los resultados de predicción respecto a la diferencia entre los valores reales y predichos, por los cuales las siguientes métricas son consideradas:

1. **Mean Absolute Error** (MAE): Promedio de la diferencia vertical entre cada punto real con su valor predicho.

$$MAE = \sum_{\mathrm{i}=1}^{n} \frac{|r_\mathrm{i} - p_\mathrm{i}|}{n} \tag{4.8}$$

2. **Mean Absolute Percentage Error** (MAPE): Promedio del porcentaje de error vertical entre cada punto real con su valor predicho.

$$MAPE = \frac{\sum_{\mathrm{i}=1}^{n} \frac{|r_\mathrm{i} - p_\mathrm{i}|}{|r_\mathrm{i}|}}{n} \tag{4.9}$$

4.3.2. Distancia

En adición a los errores de predicción, también es importante tomar en cuenta otros factores al realizar una evaluación de los modelos de predicción, principalmente debido a que los errores de predicción mencionados anteriormente no consideran algunas importantes características de una serie de tiempo al momento de compararlas. Tal como el hecho de que los puntos de una serie de tiempo poseen un orden lógico, dado por el tiempo. En este sentido, otro método de comparación es necesario, con el objetivo de capturar la similitud entre la forma denotada por los datos reales y las curvas de predicción, o la distribución de los datos.

Las siguientes métricas de similitud, comúnmente usadas por trabajos de clustering en series de tiempo, son consideradas:

1. **Edit Distance for Real Sequences** (EDR): Compara dos series de tiempo en términos de cuántas operaciones de edición (insertar, borrar, reemplazar) deben ser realizadas para transformar una curva en la otra. La distancia entre dos puntos es reducida a 0 o 1, donde si la distancia entre dos puntos r_{i} y p_j es menor que un ε dado, entonces ambos puntos son considerados iguales.
2. **Dynamic Time Warping** (DTW): Es un algoritmo de alineamiento que busca alinear dos secuencias deformando el eje temporal iterativamente hasta encontrar un 'match' óptimo entre ambas secuencias, lo que minimiza la suma de las diferencias absolutas para cada par de índices 'matcheados'. Un 'match' aceptable debe cumplir con las siguientes condiciones:
   - Cada índice de la primera secuencia debe estar 'matcheado' con uno o más índices de la otra secuencia y vice versa.
   - El primer índice de la primera secuencia debe ser 'matcheado' con el primer índice de la otra secuencia.
   - El último índice de la primera secuencia debe ser 'matcheado' con el último índice de la otra secuencia
   - El 'mapeo' de los índices de la primera a la segunda secuencia deben crecer monotónicamente, y vice versa.

4.4. Implementación

El modelo de red neuronal fue implementado con la siguiente configuración e hiperparámetros:

Input:

El modelo recibe como input una semana de datos recopilados por cada 10 minutos en una serie de tiempo multi-dimensional, lo que corresponde a 1008 valores. Esta serie de tiempo multi-dimensional contiene a cada una de las series de tiempo generadas a partir de diferentes agregaciones en el tráfico DNS. Estas corresponden a 19 en total y se detallan en el siguiente Capítulo 5 de Predicción. La razón por la que estas series de tiempo se entregan al modelo como una única serie de tiempo multidimensional, a diferencia de realizar predicciones

sobre cada una de ellas en forma independiente, es para permitir a la red el uso de posibles dependencias entre estas variables.

Capa Recurrente:

Una capa oculta recurrente de 150 neuronas tipo LSTM.

Capa Dropout:

Capa de dropout del 30 %.

Entrenamiento:

El modelo es alimentado con *batches* de tamaño 24 y entrenado por 10 épocas usando el optimizador Adam, con MAE como función de costo. Además, se utilizó una taza de aprendizaje $\alpha = 0{,}01$.

Output:

El output del modelo es el valor predicho para los siguientes 10 minutos de tráfico para cada una de las dimensiones de la serie de tiempo multi-dimensional recibidas como *input*, correspondiente a un arreglo de 19 valores.

Retro-alimentación:

Al realizar las predicciones para cada valor de las series temporales en forma secuencial, cada predicción realizada actualiza los pesos sinápticos del modelo en forma previa a la siguiente predicción. Esto incorpora información reciente al sistema mediante el cálculo de la función de costo entre la predicción resultante y el valor real observado, aplicando el procedimiento de *backpropagation* sobre las conexiones de la red neuronal. Como se mencionó anteriormente, es parte de la implementación del modelo el cómo y en qué magnitud esta nueva información olvida o añade inteligencia en la red. Por otro lado, naturalmente se utiliza el valor real observado como input para la siguiente predicción secuencial, y no el valor predicho. Una ilustración de esta implementación se muestra en la Figura 4.4.

La razón de esta retro-alimentación secuencial recae en que, ante una implementación real del sistema propuesto, se esperaría que el sistema sea adaptativo a través del tiempo respecto de los cambios en el comportamiento observado en el tráfico del servidor.

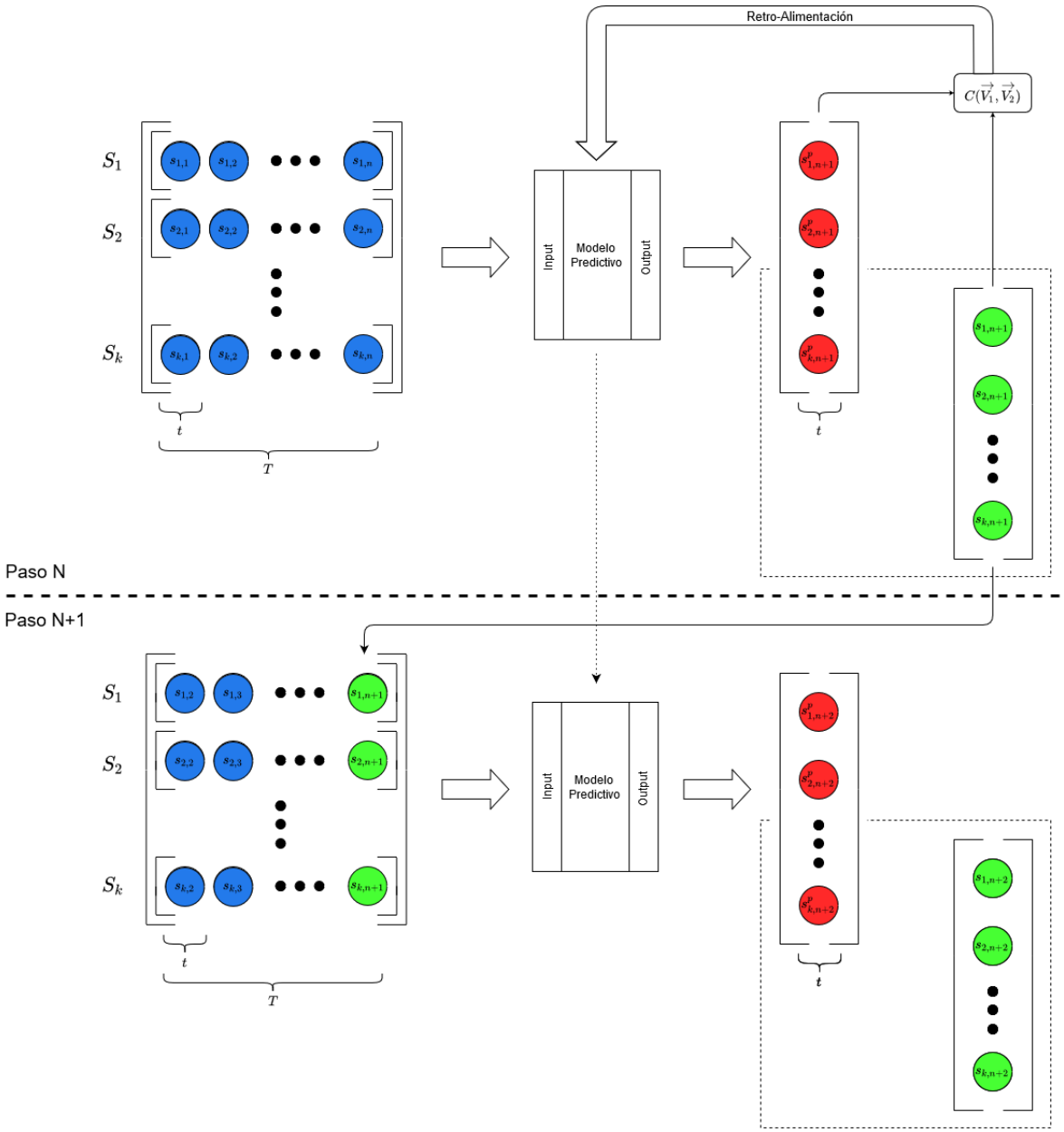


Figura 4.4: Implementación del modelo predictivo para dos pasos consecutivos N y $N+1$. Las k diferentes series de tiempo, de tamaño n cada una, componen una matriz de tamaño $k \times n$ que recibe el modelo predictivo como *input*. Los elementos de estas series de tiempo se representan en el esquema por $s_{\mathrm{i},j}$, donde i es la serie de tiempo correspondiente y j representa el valor en la secuencia por un intervalo de tiempo t. El tiempo total en la matriz es $T = t \cdot n$. El *output* entregado es un vector de tamaño k con la predicción para el instante $n+1$ en cada serie temporal, representado por $s^p_{\mathrm{i},n+1}$. Existirá además un vector $s_{\mathrm{i},n+1}$ con los valores reales. La función de costo C será utilizada para realizar *backpropagation* y retro-alimentar con la nueva información al modelo, de forma previa al siguiente paso $N+1$, donde se calculará el nuevo vector predicho $s^p_{\mathrm{i},n+2}$, utilizando los instantes $[2, n]$ del *input* en el paso N, junto al nuevo vector observado $s_{\mathrm{i},n+1}$. La configuración usada estipula $k = 19$, $n = 1008$, $t = 10[minuto]$, $T = 1[semana]$, $C = MAE$.

Capítulo 5

Predicción de Tráfico DNS

En este capítulo se detalla la transformación realizada a los datos, con el propósito de entregarlos al modelo predictivo. Luego, se describe el experimento realizado utilizando el modelo. Finalmente, se exponen y discuten los resultados de forma visual y respecto a las métricas de evaluación detalladas en el Capítulo 4

5.1. Pre-Procesamiento de Datos

Considerando el volumen de los datos, es necesario resumir la información, de manera que sea factible de procesarla en tiempo real, y evitando perder información relevante para la posterior detección de ataques. Para esto, se armaron múltiples series de tiempo con información específica y particular del tráfico DNS. Además, se realizó una estandarización apropiada para la experimentación.

5.1.1. Agregación

Todos los paquetes DNS fueron agregados en intervalos de 10 minutos, con el fin de generar series de tiempo con diferente información, la cual toma en cuenta tanto los ataques que fueron emulados, como aspectos utilizados en trabajo relacionado. De tal forma de mantener la información relevante para hacer posible la detección de eventos anómalos en el tráfico, en un rápido procesamiento de los datos.

En efecto, para el mes completo de datos, se armaron series de tiempo de largo 4427, donde cada punto representa la información contenida en 10 minutos de tiempo. En total, se generaron 19 series de tiempo, las cuales comprendían la siguiente información:

- Número de consultas y respuestas al y desde el servidor con los tipos de registros DNS:
  - A
  - AAAA
  - NS
  - MX
  - ANY

- Número de consultas al servidor bajo el protocolo UDP.
- Número de consultas y respuestas al y desde el servidor en total, es decir, considerando la suma de los ítems anteriores.
- Número de nombres de dominio diferentes consultados.
- Número de IPs diferentes bajo una máscara de red de 32, 28, 24, 20 y 16.

5.1.2. Estandarización

Dado que las redes neuronales son sensibles a la escala de los datos, fue necesario realizar una estandarización de los datos. Sin embargo, tomando en consideración que, debido a que los ataques emulados causarán una variación abrupta en algunas series de tiempo, no es conveniente normalizar a un rango de valores fijo como lo hacen las normalizaciones que utilizan mínimo y máximo. De esta forma, se decidió efectuar una estandarización de acuerdo al Z-Score de cada punto, manteniendo la distribución en un rango reducido, lo que permite la detección de outliers fuera del set de entrenamiento.

El puntaje Z para un punto determinado está dado por la siguiente fórmula:

$$Z = \frac{x - \mu}{\sigma} \tag{5.1}$$

donde μ corresponde al promedio y σ a la desviación estándar del set de datos.

5.2. Experimentación

Luego del pre-procesamiento, las 19 series de tiempo estandarizadas resultantes fueron entregadas al modelo como una única serie de tiempo de 19 dimensiones. Esto con el propósito de hacer posible al modelo predictivo el encontrar correlaciones presentes entre las series de tiempo en forma conjunta. Para un input de 1 semana de información, se predecía el valor para los siguientes 10 minutos. De esta forma, el output consistía en un punto de 19 dimensiones. En otras palabras, se realizó una predicción para cada una de las series de tiempo generadas, de forma no independiente.

Para las series de tiempo de largo 4427, lo que corresponde con corta diferencia a 4 semanas, se estableció una separación en set de entrenamiento, set de validación y set de prueba. Tres semanas y media (2915 registros) fueron utilizadas para el entrenamiento del modelo. Media semana (504 registros) constituyeron la validación. La semana restante (1008 registros) se usó para prueba. La distribución en porcentaje es respectivamente 70 %, 10 % y 20 % en forma aproximada.

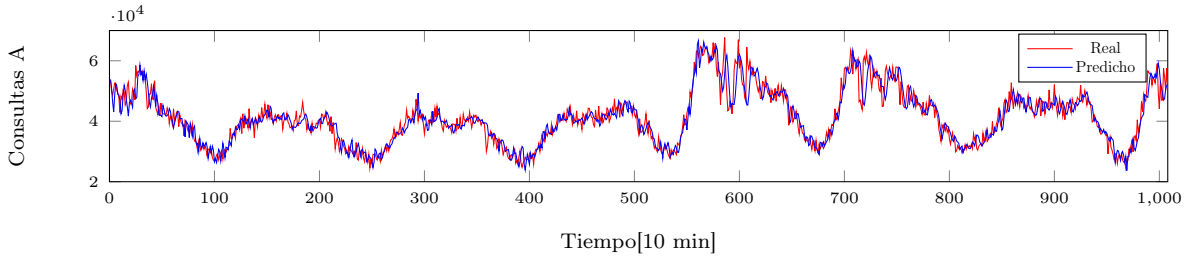
Inicialmente, el modelo es entrenado con los valores incluidos en el set de entrenamiento repetidamente, validando el aprendizaje al medir el desempeño de las predicciones con los ejemplos contenidos en el set de validación. Finalmente, en el set de prueba, se añadió un aspecto secuencial. Cada predicción realizada por el modelo dentro de la semana de prueba retro-alimenta al modelo, incorporando la nueva información (actualizando los pesos de las neuronas), de forma previa a la siguiente predicción.

Tomando esto último en cuenta, se realizó una evaluación del modelo respecto de los resultados obtenidos utilizando el set de datos sin modificaciones. A modo de validar su competencia como modelo predictivo y como herramienta de detección de eventos anómalos.

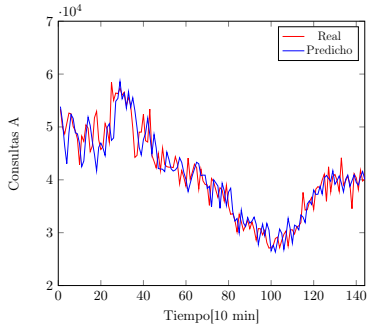
Los ataques serían incorporados posteriormente en la siguiente sección, sólo en la semana de prueba. Así, el modelo habrá sido entrenado con un funcionamiento normal del sistema, para luego analizar su desempeño al trabajar sobre los datos que contienen a los ataques emulados.

5.3. Resultados

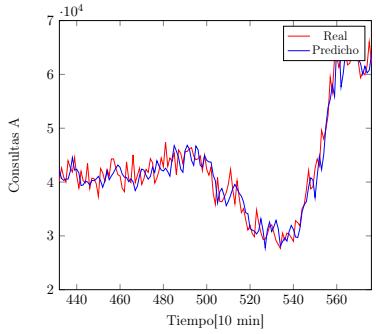
En la figuras 5.1, 5.2 y 5.3 se muestra una comparación visual entre la serie de tiempo de tráfico DNS predicha por los modelos en la semana de prueba, junto a los valores reales observados. En la parte superior de cada figura se observan las curvas para la semana completa. Para una observación más aguda, se muestran también en la parte inferior de cada figura, los días 1, 4 y 7 de la misma semana, en el mismo orden.



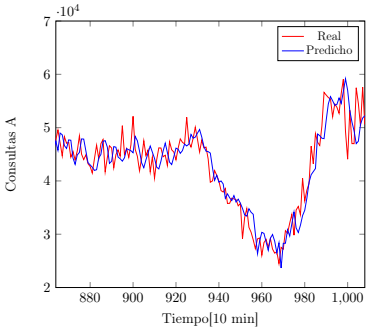
(a) Set de prueba: una semana completa



(b) Primer Día



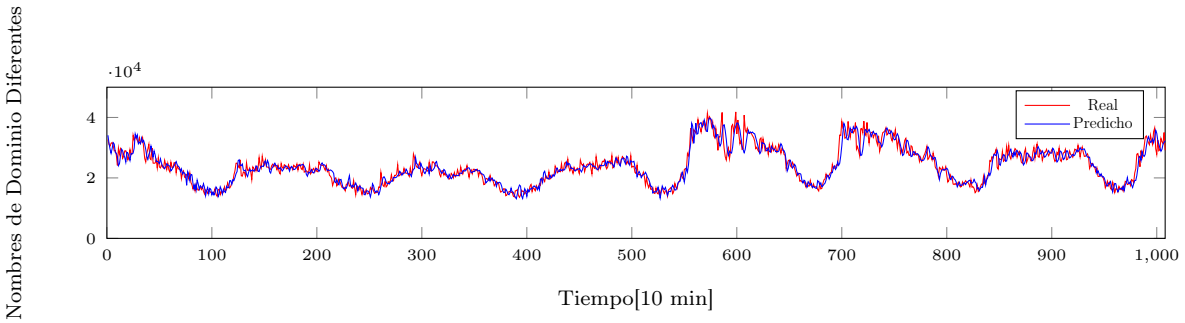
(c) Cuarto Día



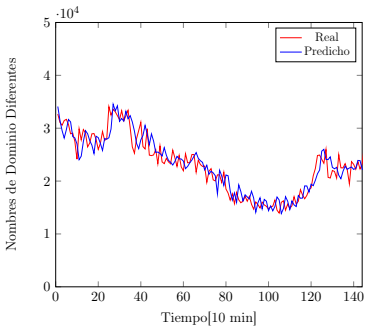
(d) Séptimo Día

Figura 5.1: Resultados de predicción en serie de tiempo de cantidad de consultas A por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.

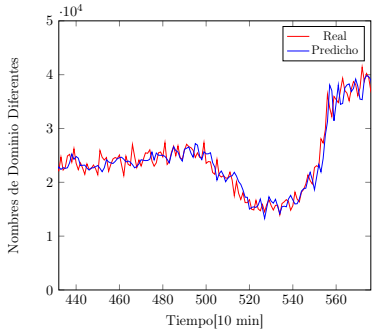
La Figura 5.1 denota la serie de tiempo de consultas tipo A realizadas al servidor. Mientras que la Figura 5.2 corresponde a los resultados para la serie de tiempo de número de nombres de dominio diferentes consultados. Finalmente, la Figura 5.3 muestra los resultados de la serie de tiempo de consultas tipo NS.



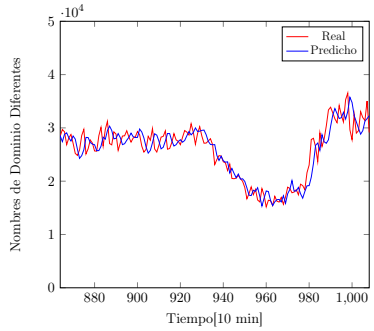
(a) Set de prueba: una semana completa



(b) Primer Día



(c) Cuarto Día

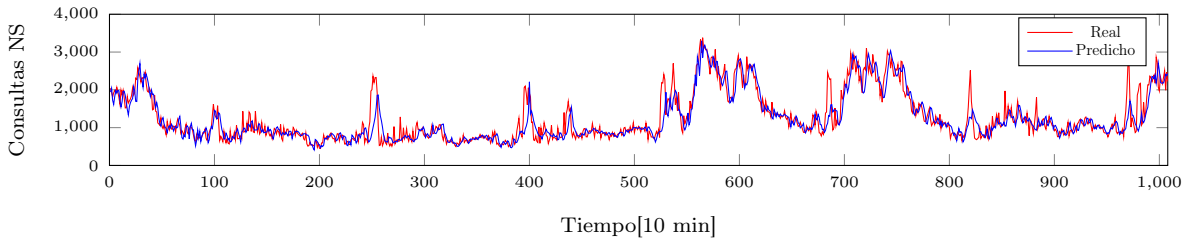


(d) Séptimo Día

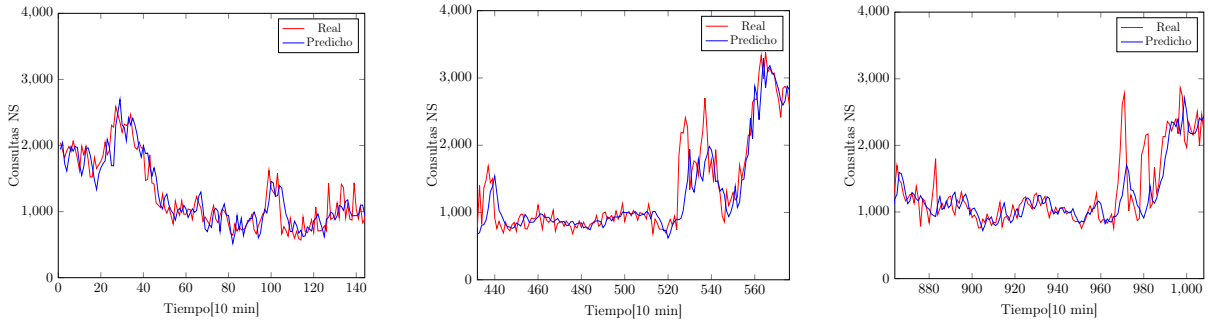
Figura 5.2: Resultados de predicción en serie de tiempo de cantidad de dominios diferentes consultados por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.

Más figuras visuales sobre series de tiempo con otras agregaciones de datos se dejan apartadas en el Anexo.

En la tabla 5.1 se presentan los valores obtenidos mediante cada medida de evaluación, para cada predicción realizada, en todas las series de tiempo generadas. Con el objetivo de establecer una medida de comparación y validar al modelo predictivo.



(a) Set de prueba: una semana completa



(b) Primer Día (c) Cuarto Día (d) Séptimo Día

Figura 5.3: Resultados de predicción en serie de tiempo de cantidad de consultas NS por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.

Serie de Tiempo	MAPE	DTW	EDR ($\varepsilon = 5$)
Consultas tipo A	0.057	1620959.7	1004
Respuestas tipo A	0.058	1603647.7	1005
Consultas tipo AAAA	0.064	579494.5	993
Respuestas tipo AAAA	0.064	580744	997
Consultas tipo NS	0.167	128257.9	931
Respuestas tipo NS	0.167	128422.4	924
Consultas tipo MX	0.137	817576.7	990
Respuestas tipo MX	0.135	805516.7	995
Consultas tipo ANY	0.503	26239.9	699
Respuestas tipo ANY	0.517	26000.8	710
Número de nombres de dominio diferentes consultados	0.066	1060634.1	1002
Número de subredes diferentes agrupadas con máscara /32	0.026	81496.1	923
Número de subredes diferentes agrupadas con máscara /28	0.026	63923.1	899
Número de subredes diferentes agrupadas con máscara /24	0.027	56998.5	887
Número de subredes diferentes agrupadas con máscara /20	0.027	44174.2	858
Número de subredes diferentes agrupadas con máscara /16	0.027	21395.8	731
Respuestas NXDOMAIN	0.089	311474.2	981
Respuestas NOERROR	0.06	2547774.3	1007
Consultas bajo protocolo UDP	0.061	2765654	1004

Tabla 5.1: Evaluación de predicciones para cada serie de tiempo, en base a resultados obtenidos de acuerdo a las medidas de evaluación MAPE, DTW y EDR (con el valor 5 para ε), detalladas en la sección 4.3.

5.4. Discusión

De acuerdo a los resultados dispuestos en la Sección 5.3, se presenta en esta sección un análisis y se obtienen conclusiones acerca de los valores obtenidos.

Como puede observarse de manera visual en las figuras 5.1, 5.2 y 5.3, las predicciones capturan el comportamiento del tráfico en forma general. Es decir, demuestra errores ante la variabilidad de los datos, mas se apega a los patrones periódicos. Esto es importante especialmente a la hora de enfrentarse a outliers, los cuales se ven presentes en algunas figuras, y que no son tomados en cuenta por las predicciones. A diferencia de los cambios abruptos que sí están determinados por un factor habitual, los cuales son tomados en cuenta por el modelo, ajustándose las predicciones siguientes a tal variación.

Esta característica significa una herramienta a la hora de detectar anomalías. Siendo posible identificar, a través de las predicciones realizadas por la red neuronal recurrente, outliers en el tráfico, denotados por cambios abruptos anormales. Lo que se estudiará en el capítulo siguiente.

Adicionalmente, al considerar las medidas de evaluación dispuestas en la Tabla 5.1, se observa que las mediciones conllevan un bajo error porcentual. Por ejemplo, la predicción de la serie de tiempo de consultas de tipo A produjo un error porcentual de 5.8 %. Esta serie de tiempo manifiesta un promedio de 41305.7 consultas por cada 10 minutos. Es decir, se promedia un error de alrededor de 2400 consultas. Lo que resulta lo suficientemente aceptable si tomamos en cuenta que los ataques emulados consideran una mayor cantidad de consultas para cumplir su propósito. Más aún en el resto de agregaciones por tipo de registro, que son menos frecuentes que el tipo A.

Lo mismo ocurre con el número de subredes diferentes. Por ejemplo, bajo la máscara /32, cuyo promedio en la serie de tiempo es 4445.6. Con un MAPE de 0.026, se erra en promedio en 115. Esto permitiría, en forma discutible, detectar la actividad de redes de botnets con IP lo suficientemente diferentes. Pero la probabilidad de detección aumenta al agrupar por máscara de subred /16. Cuya serie de tiempo y errores porcentuales son respectivamente 1207.7 y 0.028. Estimando un error en promedio de 33 subredes.

No obstante, para las series de tiempo de registros con un menor volumen de consultas, como NS o MX, el error aumenta. Esto es especialmente drástico en las series de tiempo para tipos de registro ANY, donde las consultas son tan poco comunes (promedio de 88 consultas al servidor en la semana de prueba), que resulta difícil predecir el comportamiento, además de perder el factor periódico demostrado por otras series de tiempo. En este sentido, una variación en la actividad causará un muy alto error en los resultados de las predicciones.

Se concluye que las predicciones son satisfactorias para cumplir con el objetivo de detectar anomalías en el tráfico, lo que se verificará en el siguiente capítulo, utilizando los ataques emulados.

Capítulo 6

Detección de Anomalías

En este capítulo se plantea una heurística de detección para la detección de anomalías a partir de dos series de tiempo: una predicha y otra real. Se detalla así, la inclusión de los ataques emulados descritos en el Capítulo 3. Se realiza luego, una nueva predicción de la semana de prueba sobre el set modificado, aplicando la heurística para la detección de esta anomalía. Finalmente se exponen los resultados, acompañados de una apropiada discusión sobre estos.

6.1. Heurística de Detección

Como se ha mencionado anteriormente, lo que será identificado como anomalía, se determinará en base a los errores que presente la predicción del tráfico respecto a los valores reales observados en la serie de tiempo. En consecuencia, es necesario establecer una metodología que compare ambas series de tiempo. A continuación se detalla dicha metodología, la cual utilizará Weighted Moving Average (WMA) para determinar un promedio horizontal en porciones de cada serie, Error Porcentual para encontrar una diferencia vertical entre cada valor predicho con su símil real y un Threshold para decidir si un valor se considera anómalo.

6.1.1. Weighted Moving Average

Tomando en cuenta que una anomalía podría considerar un tiempo mayor a la agregación propuesta, es necesario tomar en cuenta una ventana de valores dentro de la serie de tiempo, de forma previa al establecimiento de un error puntual obtenido por el modelo predictivo. Calcular el promedio de los valores pertenecientes a esta ventana, a través de la serie de tiempo, corresponde al método Moving Average (Promedio Móvil). Por otro lado, ya que las anomalías pueden también tener una corta duración, resulta útil otorgarle mayor importancia a los puntos más recientes. Esto es en lo que consiste la variación Weighted Moving Average (Promedio Móvil con Pesos), cuya fórmula para un instante t de una serie de tiempo corresponde a:

$$WMA(y_t) = \frac{\sum_{\mathrm{i}=1}^{V} w_{\mathrm{i}}\ y_{t+\mathrm{i}-V}}{\sum_{\mathrm{i}=1}^{V} w_{\mathrm{i}}} \tag{6.1}$$

donde w_{i} es el peso correspondiente a la posición i de la ventana de valores, cuyo tamaño es V. El elemento y_t es el valor en el eje Y observado en una serie de tiempo en el instante t.

De esta forma, tanto para el tráfico original como para el predicho, será calculado un Weighted Moving Average. El cual entregará una serie de tiempo de las mismas dimensiones, pero con un promedio ponderado con los valores últimos valores en cada instante.

El tamaño de ventana utilizado será de 3 puntos ($V = 3$), lo que corresponde a 30 minutos de tráfico. La distribución de pesos será creciente de la forma $w_{\mathrm{i}} = 2^{\mathrm{i}-1}$.

6.1.2. Error Porcentual

Utilizando los valores de cada de cada serie de tiempo obtenidos mediante el cálculo de Weighted Moving Average, se procederá a obtener una medida de error en cada unidad de tiempo. Esto determinará un error vertical entre la predicción y la realidad, correspondiente al *Error Porcentual*, definido por la siguiente fórmula:

$$Err(r_t, p_t) = 100 \cdot \frac{|r_t - p_t|}{\frac{(r_t\ +\ p_t)}{2}} \tag{6.2}$$

donde el r_t es el valor real en el instante t y p_t es el valor predicho por el modelo predictivo en el instante t.

Así, para cada par de series de tiempo predicha y real, se tendrá una serie de errores porcentuales, que nos permitirán reconocer un instante como anómalo, o como normal.

6.1.3. Threshold

Por último, establecer un *threshold* sobre el error porcentual en un punto dado, clasificará a dicho punto en anómalo o normal. En otras palabras, dado un valor Th, todos los puntos de una serie de tiempo que presenten un error mayor a Th serán reconocidos como anómalos.

Podemos resumir, a fin de cuentas, a la heurística de detección como una condición expresada de la forma siguiente:

$$Err(WMA(r_t), WMA(p_t)) > Th \tag{6.3}$$

la cual señalará a un punto como anómalo en caso de cumplirse.

No obstante, este valor del threshold no será definido, sino que será variable para la experimentación descrita en la sección a continuación.

6.2. Experimentación

En esta sección se describe a las instancias de ataques emulados y cómo fueron incluidas en el set de prueba de los datos. Además, se detalla la forma en que estas instancias fueron etiquetadas. Posterior a esto, se realizó una nueva predicción sobre las series de tiempo modificadas.

6.2.1. Inclusión de Anomalías

Distintas instancias de los ataques emulados fueron incluidas en la semana de prueba, procurando abarcar variabilidad, con el propósito de añadir robustez a la experimentación con el método de detección propuesto. Así, los ataques se diversificaron en términos de duración y de cantidad de miembros ejecutando el ataque (emulando una botnet). Además, se distribuyeron de forma dispersa a lo largo de la semana de prueba del set de datos.

En total, se realizaron 15 ataques, 3 de cada tipo. Los detalles de cada emulación se encuentran a continuación, en la Tabla 6.1.

No.	Fecha (DD-MM HH:mm)	Tipo de Ataque	Número de Bots	Duración [s]
1	27-10 13:46	UDP Flood	80	100
2	28-10 5:37	Random Subdomain	10	260
3	28-10 20:53	DNS Amplification	1	200
4	29-10 12:05	DNS Amplification	25	60
5	30-10 12:11	Random Subdomain	1	300
6	31-10 6:53	DNS Amplification	30	220
7	31-10 23:44	UDP Flood	40	200
8	1-11 16:50	UDP Flood	1	350
9	2-11 8:28	Random Subdomain	60	180

Tabla 6.1: Descripción de ataques insertados en el set de prueba, variando tipo de ataque, fuentes de origen, duración y ubicación en el tiempo.

6.2.2. Etiquetado de Anomalías

De acuerdo a lo que fue descrito en la Subsección 6.1.1, se está calculando un promedio ponderado respecto a los valores pertenecientes a una ventana de tamaño 3. En consecuencia, una anomalía influye en al menos tres diferentes errores porcentuales. En el caso de los ataques 2 y 8 de la Tabla 6.1, estos influyen en cuatro puntos diferentes, dada la agregación realizada de acuerdo a intervalos de 10 minutos.

Tomando en cuenta lo anterior, se etiquetó como anomalía a todos los errores cuya ventana de cálculo de WMA contiene a uno de los ataques insertados. Sin embargo, se hizo una diferenciación entre las anomalías que caen en la última posición de la ventana del promedio, y las que corresponden a posiciones anteriores de la ventana. A las del primer tipo se les asignó un valor *2*, mientras que a las segundas mencionadas se les etiquetará por *1*. Podemos definir a las etiquetas *2* como *Anomalía Tipo A*, y a las etiquetas *1* como *Anomalía Tipo B*. Un ejemplo de esto se ilustra en la Figura 6.1.

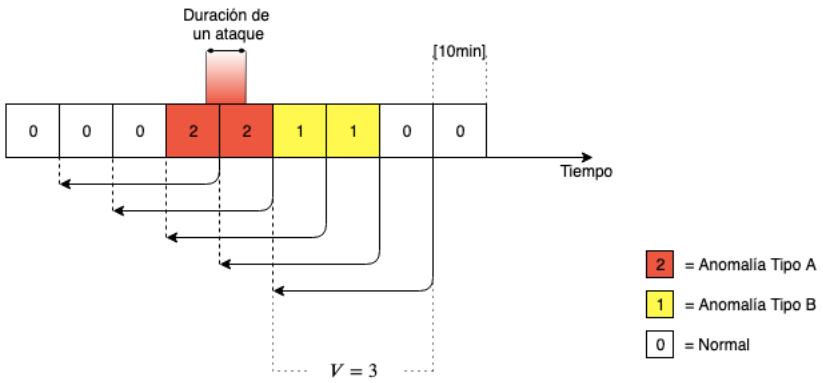


Figura 6.1: Ejemplo de etiquetado de ataques incluidos en el set de prueba. En rojo se muestra el etiquetado de anomalías definidas como Tipo A (ataque emulado cuya incidencia ocurre en la posición más reciente de la ventana de tiempo utilizada). En amarillo se muestra el etiquetado de anomalías definidas como Tipo B (ataques emulados cuya incidencia pertenece a la ventana de tiempo, pero previamente a la última posición). Se mantienen en blanco los intervalos sin incidencia de ataques.

Con esta metodología, 29 de 1008 registros de la semana de prueba fueron etiquetados como anomalías en total. De las cuales, 12 corresponden a anomalía A, y 17 a anomalía B.

6.2.3. Predicción de la Semana de Prueba

Con un nuevo set de datos de prueba, alterado por los ataques emulados incluidos, se realizó una nueva predicción sobre esta semana. Esto consideró la misma metodología explicada en el Capítulo 5 de Predicción de Tráfico DNS. El mismo modelo predictivo descrito en la Sección 4.4, ya entrenado con los set de prueba y validación, se utilizó para obtener las nuevas predicciones. El único aspecto en el que los resultados son diferentes, es de acuerdo al re-entrenamiento aplicado por cada nuevo valor real percibido, en consecuencia con la metodología descrita en la Sección 5.2.

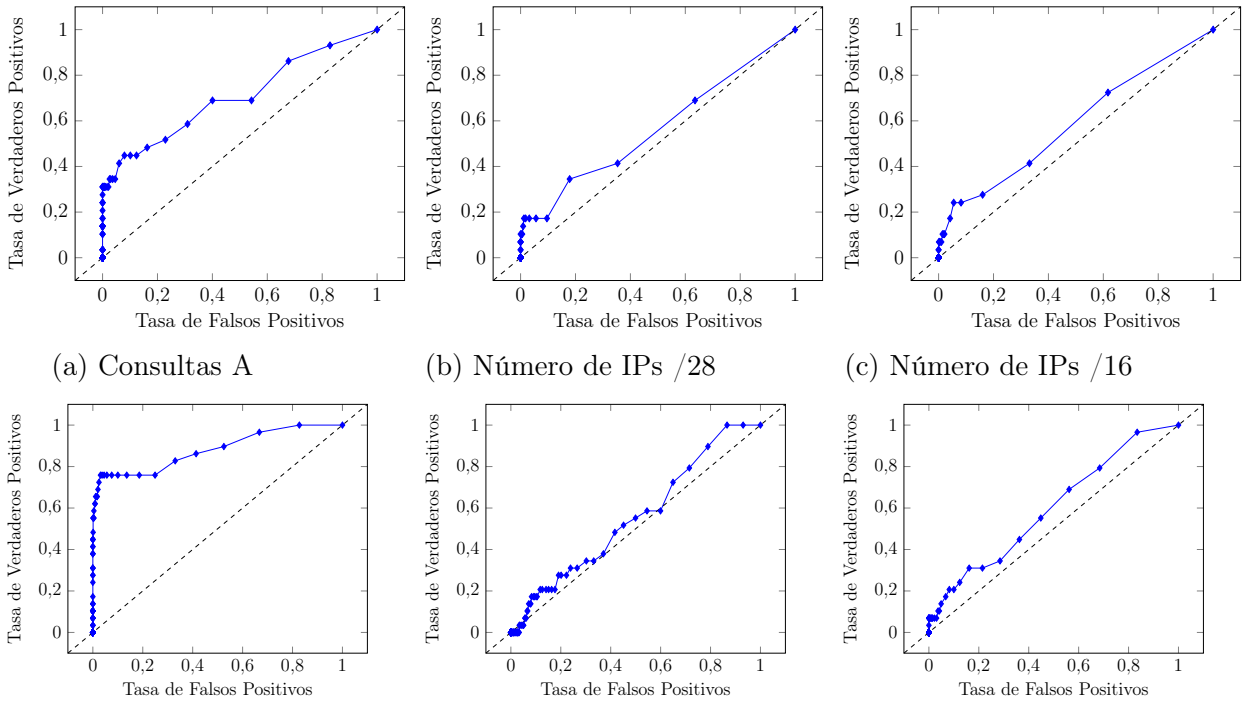
De este modo, se obtuvieron nuevamente dos series de tiempo para cada una de las agregaciones de datos descritas en la Subsección 5.1.1: una predicha y otra real. Aplicando la heurística, descrita anteriormente en este capítulo, sobre estas series de tiempo, se obtuvieron los resultados expuestos a continuación en la sección de Resultados.

6.3. Resultados

Con el objetivo de visualizar el desempeño de las detecciones, se presentan a continuación gráficos de curvas ROC, los cuales relacionan tasa de verdaderos positivos (o *recall*) con tasa de falsos positivos para múltiples valores del threshold Th. Resulta apropiado utilizar estas métricas, debido a que para este problema en particular, es crítico detectar los ataques que podrían comprometer el funcionamiento del servidor. Aunque esto signifique generar ciertas falsas alarmas, lo cual es comparativamente menos dañino que lo anterior.

La Figura 6.2, muestra las curvas ROC correspondientes a la aplicación de la heurística de detección, con diferentes valores del threshold, sobre las predicciones de las series de tiempo

respecto a algunas de las agregaciones. En ellas se consideran a ambos tipos de anomalías (de tipo A y de tipo B) como verdaderos positivos. Más figuras de este tipo se dejan apartadas en el Anexo.



(a) Consultas A (b) Número de IPs /28 (c) Número de IPs /16

(d) Paquetes UDP (e) Consultas NS (f) Nombres de Dominio

Figura 6.2: Curvas ROC de detección de eventos para múltiples agregaciones: consultas tipo A, número de IPs con máscara /28, número de IPs con máscara /16, paquetes UDP, consultas NS, nombres de dominio diferentes consultados

Naturalmente, algunas de las agregaciones no constituyen una buena característica para las detecciones, debido a que no se ven afectadas por los ataques emulados. Como es el caso de las consultas de tipo NS (Figura 6.2e), cuya curva ROC se encuentra contigua a la diagonal. Esto se traduce en un aumento similar en la cantidad de falsos positivos y verdaderos positivos a medida que disminuye el valor del threshold.

Las agregaciones por máscaras de IP detectan tempranamente algunos ataques, correspondientes a los que poseen un mayor número de miembros en su botnet.

Por otro lado, ya que todos los ataques emulados generan un aumento de paquetes UDP, es posible detectar las anomalías de forma más pronta en valores altos del threshold, a un bajo costo de falsos positivos. Tal como se observa en la Figura 6.2d.

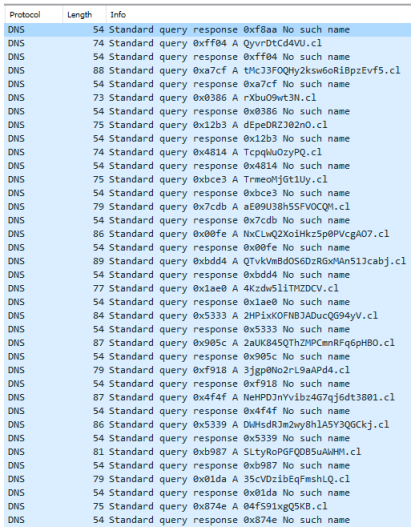
No obstante lo anterior, existe aún un margen de mejora, que es otorgado al concentrarse en los aspectos que se ven afectados por cada ataque en forma individual, como se manifiesta en la sección siguiente.

6.3.1. Anomalías Etiquetadas

En esta sección se presentan los resultados obtenidos al realizar una diferenciación para cada ataque de manera individual. Esto es, considerar en cada escenario como anomalía solo a las instancias del ataque en cuestión, desestimando a los otros tipos de ataque.

Random Subdomain

La naturaleza de este ataque afecta a la cantidad de consultas tipo A, el número de dominios diferentes consultados y a la cantidad de respuestas NXDOMAIN generadas. Ejemplificando esto último, una captura de paquetes mediante Wireshark [9] muestra el rastro que deja una de las instancias emuladas de este ataque, en la Figura 6.3.

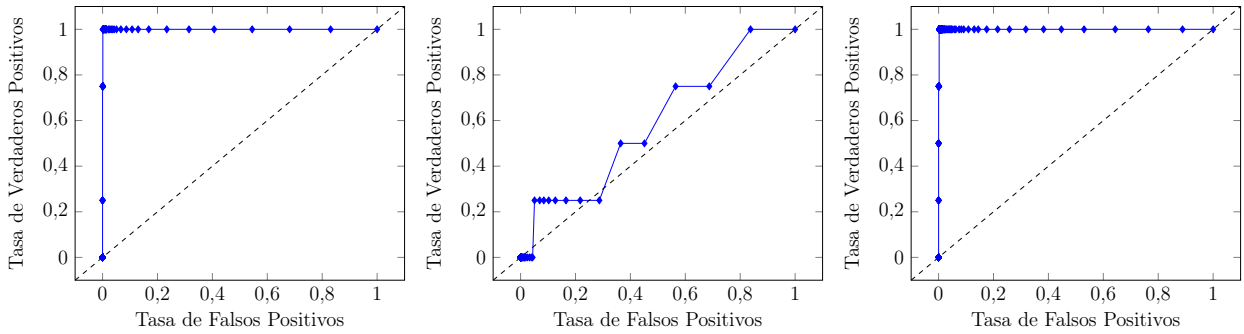


Protocol	Length	Info
DNS	54	Standard query response 0xf8aa No such name
DNS	74	Standard query 0xff04 A QyvrDtCd4VU.cl
DNS	54	Standard query response 0xff04 No such name
DNS	88	Standard query 0xa7cf A tNcJ3FOQHy2ksw6oRiBpzEvf5.cl
DNS	54	Standard query response 0xa7cf No such name
DNS	73	Standard query 0x0386 A rXbuO9wt3N.cl
DNS	54	Standard query response 0x0386 No such name
DNS	75	Standard query 0x12b3 A dEpeDRZJ02nO.cl
DNS	54	Standard query response 0x12b3 No such name
DNS	74	Standard query 0x4814 A TcpqWuOzyPQ.cl
DNS	54	Standard query response 0x4814 No such name
DNS	75	Standard query 0xbce3 A TrmeoMjGt1Uy.cl
DNS	54	Standard query response 0xbce3 No such name
DNS	79	Standard query 0x7cdb A aE09U38h5SFVOCQM.cl
DNS	54	Standard query response 0x7cdb No such name
DNS	86	Standard query 0x00fe A NxCLwQ2XoiHkz5p0PVcgAO7.cl
DNS	54	Standard query response 0x00fe No such name
DNS	89	Standard query 0xbdd4 A QTvkVmBdOS6DzRGxMAn51Jcabj.cl
DNS	54	Standard query response 0xbdd4 No such name
DNS	77	Standard query 0x1ae0 A 4Kzdw5liTMZDCV.cl
DNS	54	Standard query response 0x1ae0 No such name
DNS	84	Standard query 0x5333 A 2HPixKOFNBJADucQG94yV.cl
DNS	54	Standard query response 0x5333 No such name
DNS	87	Standard query 0x905c A 2aUK845QThZMPCmnRFq6pHBO.cl
DNS	54	Standard query response 0x905c No such name
DNS	79	Standard query 0xf918 A 3jgp0No2rL9aAPd4.cl
DNS	54	Standard query response 0xf918 No such name
DNS	87	Standard query 0x4f4f A NeHPDJnYvibz4G7qj6dt3801.cl
DNS	54	Standard query response 0x4f4f No such name
DNS	86	Standard query 0x5339 A DWHsdRJm2wy8hlA5Y3QGCkj.cl
DNS	54	Standard query response 0x5339 No such name
DNS	81	Standard query 0xb987 A SLtyRoPGFQDB5uAWHM.cl
DNS	54	Standard query response 0xb987 No such name
DNS	79	Standard query 0x01da A 35cVDzibEqFmshLQ.cl
DNS	54	Standard query response 0x01da No such name
DNS	75	Standard query 0x874e A 04fS91xgQ5KB.cl
DNS	54	Standard query response 0x874e No such name

Figura 6.3: Captura de paquetes en instancia de ataque emulado Random Subdomain. Las consultas se realizan por un dominio aleatoriamente generado finalizado en *.cl*. Las respuestas indican la no existencia del dominio.

La Figura 6.4 presenta las curvas ROC de las agregaciones mencionadas. La primera fila considera solo anomalías A, mientras que la segunda fila presenta ambos tipos de anomalías: A y B. En ellas se observa que la detección resulta mucho más confiable tanto para la cantidad de consultas tipo A, como para las respuestas NXDOMAIN. Especialmente en esta última, donde para ambos tipos de anomalías, el filtro es casi ideal a partir de un cierto threshold (85.4 %), donde se encuentran 9 anomalías de sin falsos positivos. Para las dos restantes se requiere disminuirlo a 45.29 %, incluyendo 4 falsos positivos.

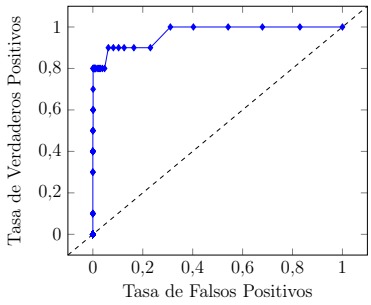
A diferencia de lo que ocurre con las consultas tipo A, las cuales detectan las anomalías A también de forma casi ideal para un threshold de 49 %. Mas la detección es más costosa en términos de falsos positivos al tomar en cuenta las anomalías B; como se puede observar en la Subfigura 6.4e. Caso contrario es el de la agregación por nombres de dominio diferentes, el cual no representa una buena característica para la detección, según señalan las curvas. Esto se debe a que el aumento en la cantidad de nombres de dominio diferentes no es significativo, en comparación a la amplia cantidad de dominios diferentes consultado por el resto de los usuarios regulares.



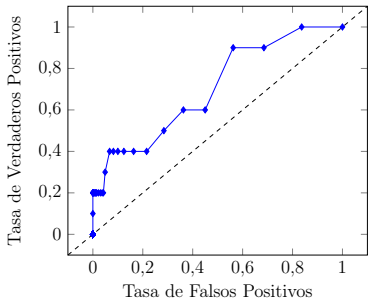
(a) Consultas A (anomalías A)

(b) Nombres de Dominio (anomalías A)

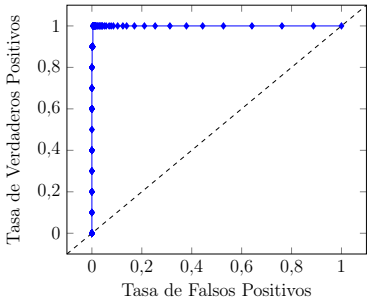
(c) Respuestas NXDOMAIN (anomalías A)



(d) Consultas A (anomalías A y B)

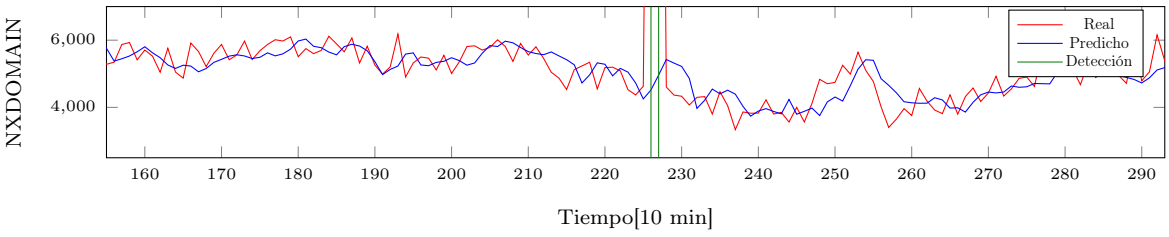


(e) Nombres de Dominio (anomalías A y B)

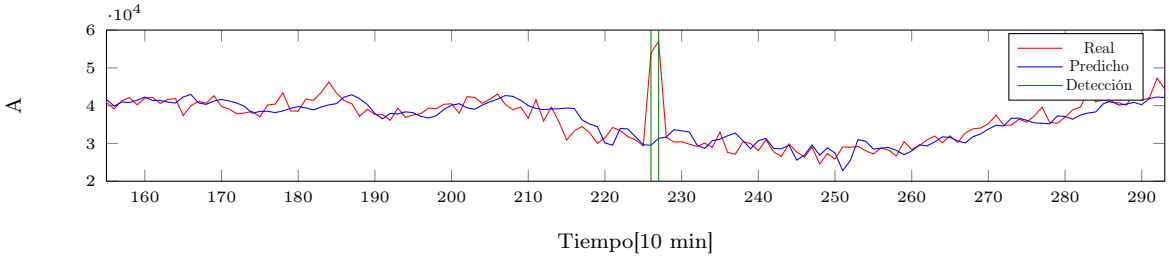


(f) Respuestas NXDOMAIN (anomalías A y B)

Figura 6.4: Curvas ROC de detección de eventos en ataque Random Subdomain para múltiples agregaciones: consultas tipo A, nombres de dominio diferentes consultados y respuestas NXDOMAIN. Considerando solo anomalías A en un caso, y tanto A como B en el otro.



(a) Respuestas NXDOMAIN



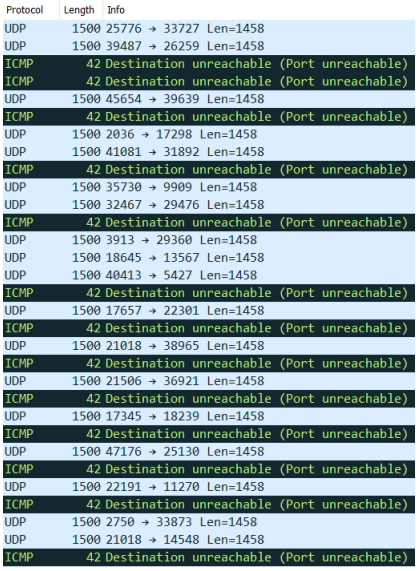
(b) Consultas A

Figura 6.5: Detección de ataque Random Subdomain sobre tráfico agregado. La instancia corresponde al ataque número 2 (ver Tabla 6.1). Comparación de tráfico real y predicho para las agregaciones: Consultas A y respuestas NXDOMAIN.

Finalmente, la Figura 6.5 muestra en la serie de tiempo, la incidencia de uno de los ataques emulados, a modo de observación de la información sobre la que actúa la heurística de detección. Los paquetes recibidos por el servidor

UDP Flood

El ataque UDP Flood funciona ajeno al protocolo de DNS, por lo que las series de tiempo que se ven afectadas por estas instancias, son las IP de origen (si no existe *Spoofing*, como en este caso), y el número de paquetes UDP enviados al servidor. La Figura 6.6 ilustra cómo son recibidos y respondidos los paquetes del ataque emulado, por parte del servidor.



Protocol	Length	Info
UDP	1500	25776 → 33727 Len=1458
UDP	1500	39487 → 26259 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	45654 → 39639 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	2036 → 17298 Len=1458
UDP	1500	41081 → 31892 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	35730 → 9909 Len=1458
UDP	1500	32467 → 29476 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	3913 → 29360 Len=1458
UDP	1500	18645 → 13567 Len=1458
UDP	1500	40413 → 5427 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	17657 → 22301 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	21018 → 38965 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	21506 → 36921 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	17345 → 18239 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	47176 → 25130 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	22191 → 11270 Len=1458
ICMP	42	Destination unreachable (Port unreachable)
UDP	1500	2750 → 33873 Len=1458
UDP	1500	21018 → 14548 Len=1458
ICMP	42	Destination unreachable (Port unreachable)

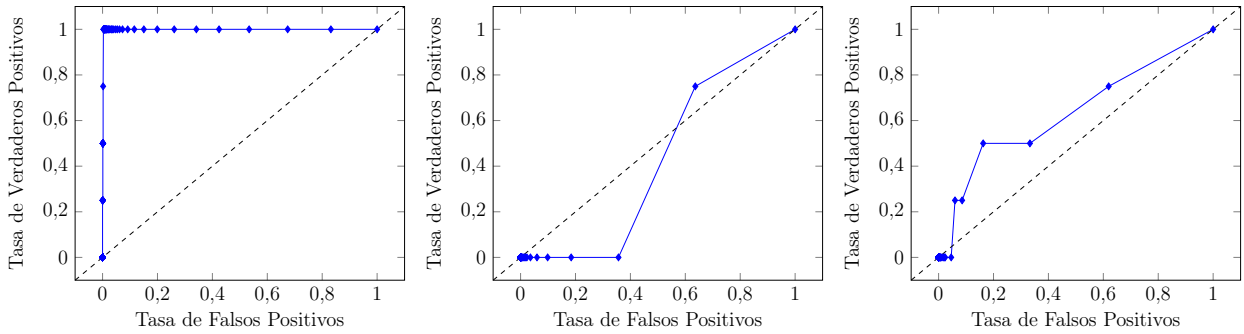
Figura 6.6: Captura de paquetes en instancia de ataque emulado UDP Flood. Se reciben varias instancias de paquetes tipo UDP, a las que se responde con un paquete generado del tipo ICMP indicando la no-disponibilidad del puerto consultado.

Nuevamente, se presentan en la Figura 6.7, las curvas ROC respecto a las detecciones de los ataques emulados. Separando por tipos de ataque y para las agregaciones relevantes para este tipo de ataque en particular: cantidad de paquetes UDP recibidos por el servidor y número de IPs agrupadas tanto por la máscara /28 como por la máscara /16.

Las curvas muestran que utilizando la agregación por cantidad de paquetes UDP es simple reconocer a las anomalías del tipo A. Sin embargo, se comienza a caer en un aumento de falsos positivos al intentar encontrar el tipo de anomalías B. Esto se ve en buena parte afectado por el resto de ataques emulados, los cuales también generan un aumento de paquetes UDP. En este caso, para un threshold de 42.7 % se habrán detectado las instancias del ataque UDP Flood de tipo A. Para un threshold de 26 % se tendrán el resto de las anomalías de UDP Flood, pero a un costo de 40 % de tasa de falsos positivos.

Por otro lado, las agrupaciones por subred no demuestran ser un buen clasificador, en vista de la alta cantidad de IPs distintas que consultan al servidor en cada unidad de tiempo.

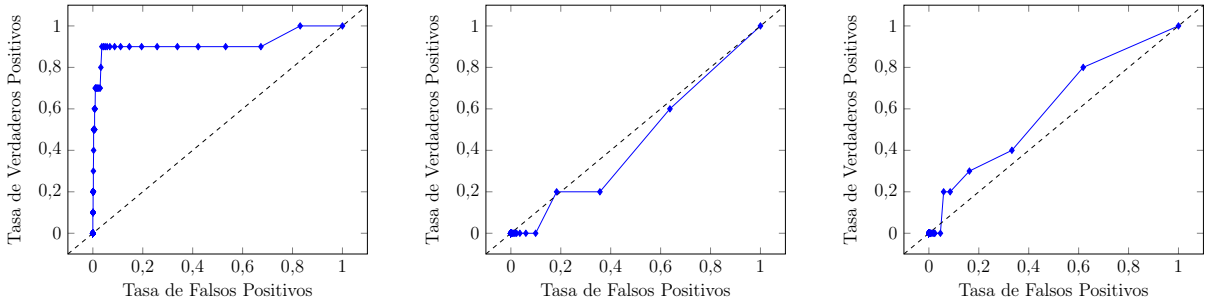
Por último, la Figura 6.8 muestra un ejemplo visual de una de las detecciones mediante la agregación por cantidad de paquetes UDP.



(a) Paquetes UDP (anomalías A)

(b) Número de IPs /28 (anomalías A)

(c) Número de IPs /16 (anomalías A)



(d) Paquetes UDP (anomalías A y B)

(e) Número de IPs /28 (anomalías A y B)

(f) Número de IPs /16 (anomalías A y B)

Figura 6.7: Curvas ROC de detección de eventos en ataque UDP Flood para múltiples agregaciones: paquetes UDP, número de IPs con máscara /28 y número de IPs con máscara /16. Considerando solo anomalías A en un caso, y tanto A como B en el otro.

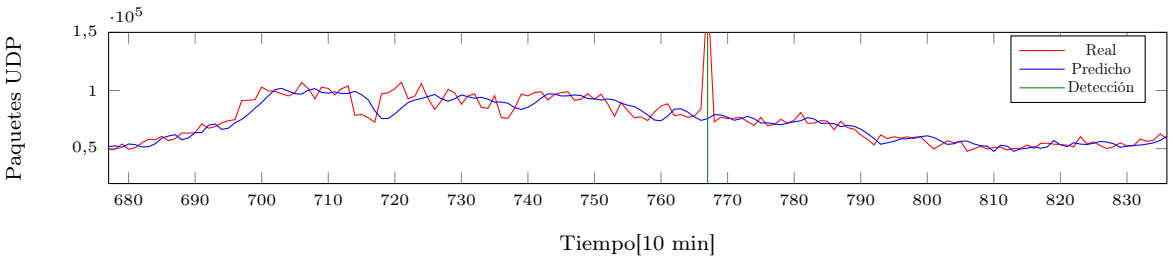
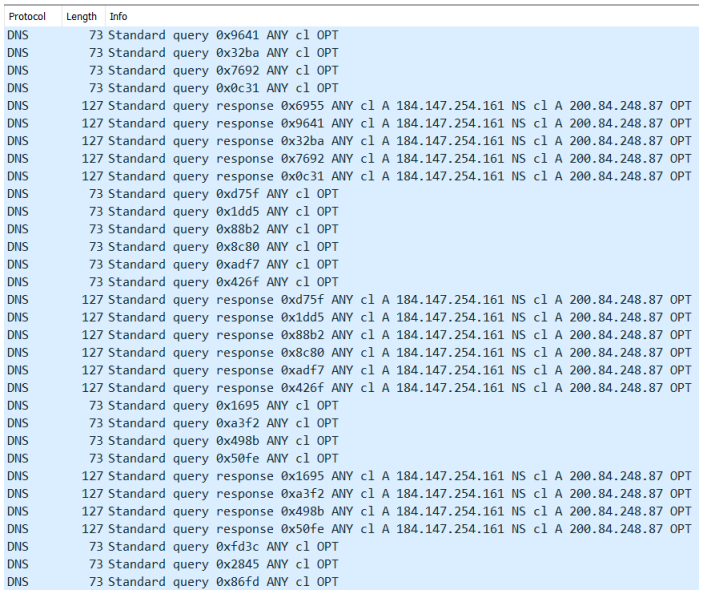


Figura 6.8: Detección de ataque UDP Flood sobre tráfico agregado. La instancia corresponde al ataque número 7. Comparación de tráfico real y predicho para la agregación de paquetes UDP.

DNS Amplification

El ataque DNS Amplification afecta principalmente en la cantidad de consultas de tipo ANY. Además podría identificarse en agrupaciones de IP al ejecutarse como botnet. En la implementación de la emulación de este ataque realizada en este trabajo, la IP de origen sí presenta Spoofing. Una captura de los paquetes emulados se observa en 6.9.



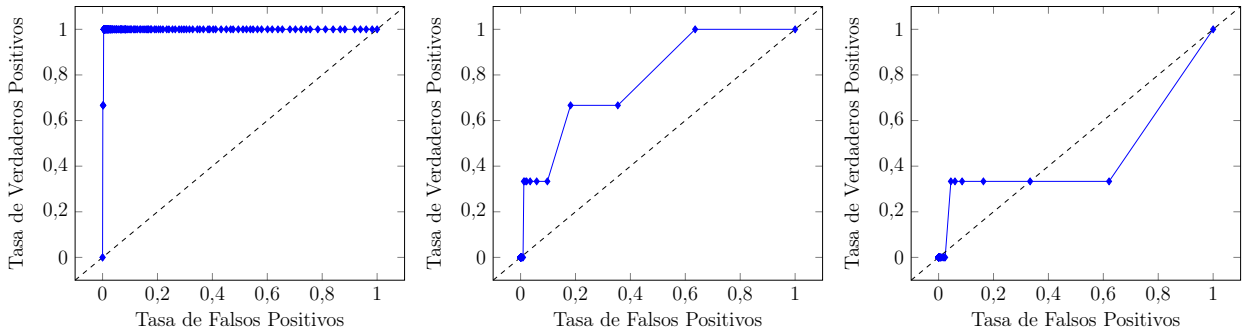
Protocol	Length	Info
DNS	73	Standard query 0x9641 ANY cl OPT
DNS	73	Standard query 0x32ba ANY cl OPT
DNS	73	Standard query 0x7692 ANY cl OPT
DNS	73	Standard query 0x0c31 ANY cl OPT
DNS	127	Standard query response 0x6955 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x9641 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x32ba ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x7692 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x0c31 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	73	Standard query 0xd75f ANY cl OPT
DNS	73	Standard query 0x1dd5 ANY cl OPT
DNS	73	Standard query 0x88b2 ANY cl OPT
DNS	73	Standard query 0x8c80 ANY cl OPT
DNS	73	Standard query 0xadf7 ANY cl OPT
DNS	73	Standard query 0x426f ANY cl OPT
DNS	127	Standard query response 0xd75f ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x1dd5 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x88b2 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x8c80 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0xadf7 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x426f ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	73	Standard query 0x1695 ANY cl OPT
DNS	73	Standard query 0xa3f2 ANY cl OPT
DNS	73	Standard query 0x498b ANY cl OPT
DNS	73	Standard query 0x50fe ANY cl OPT
DNS	127	Standard query response 0x1695 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0xa3f2 ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x498b ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	127	Standard query response 0x50fe ANY cl A 184.147.254.161 NS cl A 200.84.248.87 OPT
DNS	73	Standard query 0xfd3c ANY cl OPT
DNS	73	Standard query 0x2845 ANY cl OPT
DNS	73	Standard query 0x86fd ANY cl OPT

Figura 6.9: Captura de paquetes en instancia de ataque emulado DNS Amplification. Consultas reiteradas por el tipo de registro ANY, provocando una respuesta de múltiples registros contenidos en un paquete de mayor tamaño (amplificado).

Así, las curvas ROC de la Figura 6.10 muestran las detecciones de las instancias de la emulación de este tipo de ataque, respecto a dichas agregaciones. La cantidad de consultas ANY que se realizan al servidor normalmente es baja. Es por esto que resulta fácil identificar a los ataques en aquella agregación de los datos, como se demuestra en las figuras 6.10a y 6.10e, las cuales realizan una clasificación ideal, pues los errores en los puntos anómalos superan el 180 %.

La agregación por IP con máscara /28 demuestra un mejor resultado que para el ataque anterior de UDP Flood. Sin embargo, no se puede asegurar que sea un mejor clasificador, la diferencia recae netamente en la implementación de los ataques y en que agregación cayeron las IP a las que se asignó cada botnet, además de el número de IP diferentes que consultaron al servidor cuando se integró cada instancia de ataque.

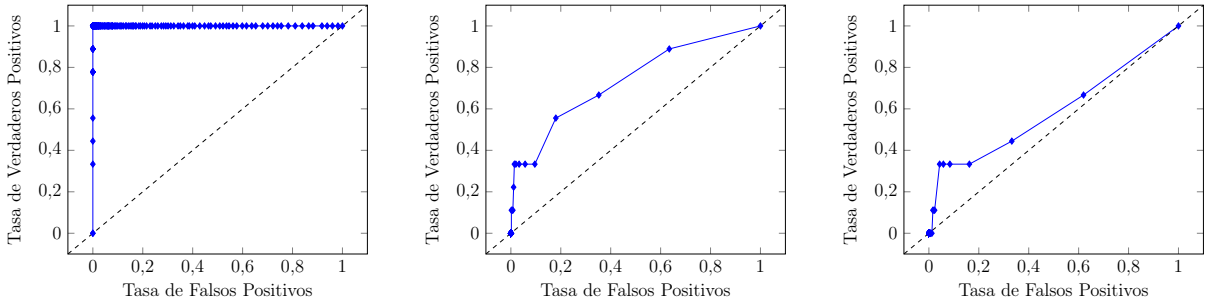
El ejemplo de detección, correspondiente al ataque número 6 de acuerdo a la Tabla 6.1 se encuentra en la Figura 6.11. Aquí se observa la naja cantidad de consultas tipo ANY entre las que se detecta el ataque de amplificación.



(a) Consultas ANY (anomalías A)

(b) Número de IPs /28 (anomalías A)

(c) Número de IPs /16 (anomalías A)



(d) Consultas ANY (anomalías A y B)

(e) Número de IPs /28 (anomalías A y B)

(f) Número de IPs /16 (anomalías A y B)

Figura 6.10: Curvas ROC de detección de eventos en ataque Random Subdomain para múltiples agregaciones: Consultas tipo A, nombres de dominio diferentes consultados y respuestas NXDOMAIN. Considerando solo anomalías A en un caso, y tanto A como B en el otro.

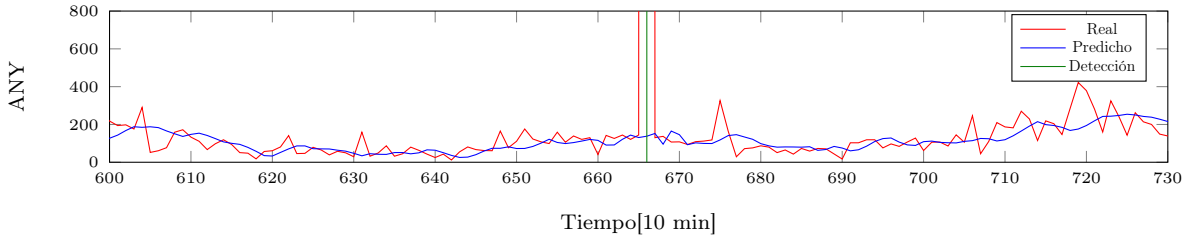


Figura 6.11: Detección de ataque DNS Amplification sobre tráfico agregado. La instancia corresponde al ataque número 6. Comparación de tráfico real y predicho para la agregación de consultas ANY.

6.3.2. Anomalías No-Etiquetadas

Si bien se utilizaron los ataques emulados para validar el procedimiento de detección que se está aplicando, los falsos positivos que se generaron no necesariamente son errores. Nada asegura que no haya eventos maliciosos existentes entre los datos originales. Es por esta razón que en la sección presente se procede a analizar también a algunas de las instancias que arrojaron un alto error porcentual bajo el procedimiento realizado hasta ahora.

Evento 1

Dos de los puntos que obtuvieron mayor error porcentual en la serie de tiempo de consultas de tipo MX se observan en la Figura 6.12. Los cuales demostraron ambos un error de más de un 60 % con respecto a la predicción realizada por el modelo predictivo. Durante este intervalo de tiempo se observa un anormal aumento en la cantidad de consultas asociadas a servicios de e-mail.

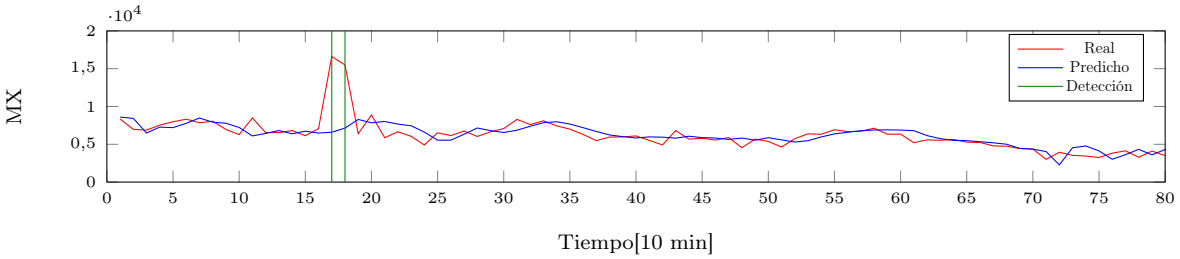


Figura 6.12: Detección de anomalía no-etiquetada en serie de tiempo de consultas MX.

Inspeccionando en detalle las consultas contenidas en este periodo, se observó que la mayoría de las consultas (cerca de 3000 durante los 20 minutos, correspondiente casi al 10 % del total de consultas MX) provenían de una única IP. Las consultas consisten de dominios válidos, tal como se observa en la Figura 6.13. Repitiendo frecuentemente consultas, este comportamiento se produjo a lo largo de los 20 minutos de forma ininterrumpida.

Más aún, 17628 consultas dentro del intervalo de 20 minutos provienen de un mismo grupo de IPs pertenecientes a una misma clase C. Esto corresponde a más del 50 % del total. El contenido de las consultas es similar al anteriormente expuesto.

Todo esto indica a una red organizada realizando *spam* a correos con algún objetivo en específico.

Protocol	Length	Info	Time
DNS	86	Standard query 0x1e62 MX catransporte.cl OPT	0.698030
DNS	83	Standard query 0x92cf MX workforce.cl OPT	0.967762
DNS	78	Standard query 0x0e46 MX opus.cl OPT	1.730634
DNS	84	Standard query 0x522d MX egdo-chile.cl OPT	1.798138
DNS	86	Standard query 0x306d MX catransporte.cl OPT	1.882550
DNS	79	Standard query 0xfb0a MX aysen.cl OPT	2.586610
DNS	83	Standard query 0x929e MX workforce.cl OPT	2.669591
DNS	86	Standard query 0x5154 MX gsiseguridad.cl OPT	3.354380
DNS	78	Standard query 0x56bc MX opus.cl OPT	3.528382
DNS	86	Standard query 0x609a MX evonchrismar.cl OPT	3.531468
DNS	84	Standard query 0xb879 MX egdo-chile.cl OPT	3.745408
DNS	86	Standard query 0xa1f3 MX catransporte.cl OPT	3.764794
DNS	84	Standard query 0x5361 MX egdo-chile.cl OPT	3.779542
DNS	86	Standard query 0x4cc7 MX gsiseguridad.cl OPT	4.101031
DNS	86	Standard query 0x3d0f MX lostulipanes.cl OPT	4.204422
DNS	94	Standard query 0x2aec MX fundacionsolidaridad.cl OPT	4.237285
DNS	83	Standard query 0xf81f MX workforce.cl OPT	4.514544
DNS	83	Standard query 0x0047 MX workforce.cl OPT	4.527265
DNS	81	Standard query 0x1f45 MX elecman.cl OPT	4.695767
DNS	84	Standard query 0xcfb8 MX grupomacro.cl OPT	4.781454
DNS	86	Standard query 0xd782 MX catransporte.cl OPT	5.475089
DNS	86	Standard query 0xa83b MX lostulipanes.cl OPT	5.679118
DNS	86	Standard query 0x7c69 MX catransporte.cl OPT	5.751933
DNS	94	Standard query 0x8527 MX fundacionsolidaridad.cl OPT	5.809582
DNS	86	Standard query 0x9033 MX evonchrismar.cl OPT	6.675529
DNS	86	Standard query 0x5699 MX lostulipanes.cl OPT	6.909045
DNS	79	Standard query 0x1c09 MX aysen.cl OPT	7.137737
DNS	83	Standard query 0x56fd MX workforce.cl OPT	7.642768
DNS	84	Standard query 0x886f MX egdo-chile.cl OPT	7.823815
DNS	84	Standard query 0x6926 MX egdo-chile.cl OPT	8.018207
DNS	78	Standard query 0xaa67 MX opus.cl OPT	8.047171
DNS	78	Standard query 0x1b45 MX opus.cl OPT	8.056581
DNS	86	Standard query 0x3fec MX evonchrismar.cl OPT	8.655763
DNS	86	Standard query 0x74b2 MX lostulipanes.cl OPT	8.673170
DNS	83	Standard query 0x2c50 MX workforce.cl OPT	9.769907
DNS	78	Standard query 0xcfb3 MX opus.cl OPT	10.738531
DNS	86	Standard query 0x1675 MX oldmackayans.cl OPT	10.740897

Figura 6.13: Captura de paquetes en anomalía no-etiquetada número 1. Múltiples consultas por registros MX provenientes de un grupo de IPs, indicando *spam* de correos electrónicos.

Evento 2

En la serie de tiempo de consultas de tipo ANY también existieron puntos con un alto error (mayor a 100 %), dado por un alto aumento en este tipo de consultas que no resultan ser tan comunes. Por ejemplo, el evento observado en la Figura 6.14 registra un *peak* de 760 consultas ANY en el intervalo de 10 minutos implicado. De estas consultas, 711 poseen una IP de origen perteneciente a una misma clase C. Lo cual es el 93.55 % del total.

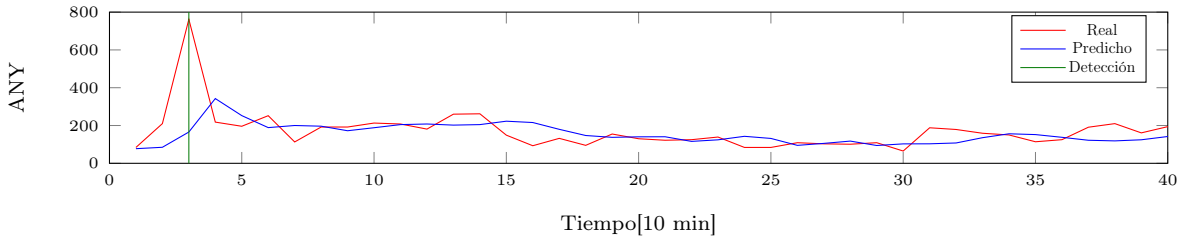


Figura 6.14: Detección de anomalía no-etiquetada en serie de tiempo de consultas ANY.

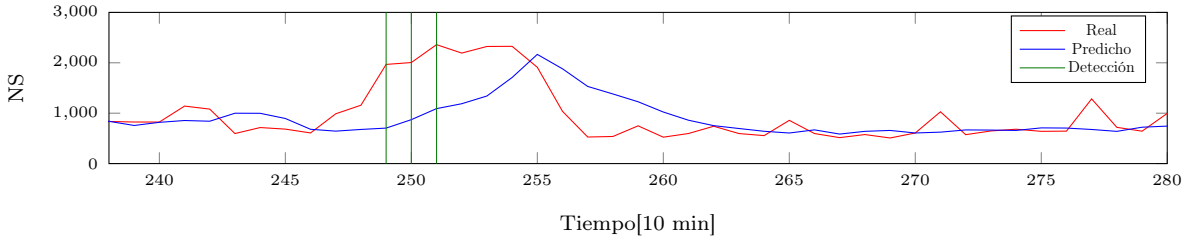
Los dominios consultados son válidos. No obstante, estos no poseen ninguna relación entre unos y otros. Lo que se ilustra en la captura de paquetes en la Figura 6.15. Esto puede indicar la presencia de un ataque de amplificación dirigido a algún servicio ajeno.

Protocol	Length	Info	Time
DNS	69	Standard query 0x2338 ANY magasa.cl	420.956211
DNS	148	Standard query response 0x2338 ANY magasa.cl NS ns2.telmexchile.cl NS ns.telmexchile.cl A 200.27.2.2 A 200.27.2.7	420.956408
DNS	79	Standard query 0x94fb ANY autopistacentral.cl	421.652000
DNS	257	Standard query response 0x94fb ANY autopistacentral.cl NS nsgtd.autopistacentral.cl NS nstie.autopistacentral.cl NS secundario…	421.652252
DNS	81	Standard query 0xdb6f ANY colegiosdiaconales.cl	422.894783
DNS	188	Standard query response 0xdb6f ANY colegiosdiaconales.cl NS ns1.colegiosdiaconales.cl NS secundario.nic.cl A 190.151.104.41 A …	422.895003
DNS	69	Standard query 0x2123 ANY kupfer.cl	424.800233
DNS	175	Standard query response 0x2123 ANY kupfer.cl NS secundario.nic.cl NS ns.kupfer.cl A 200.68.0.121 A 200.7.5.7 AAAA 2001:1398:27…	424.800473
DNS	78	Standard query 0x600b ANY investigaciones.cl	425.014604
DNS	146	Standard query response 0x600b ANY investigaciones.cl NS ns2.investigaciones.cl NS ns1.investigaciones.cl A 201.217.242.244 A …	425.014740
DNS	73	Standard query 0x8035 ANY lauruguaya.cl	426.952899
DNS	128	Standard query response 0x8035 ANY lauruguaya.cl NS ns13.domaincontrol.com NS ns14.domaincontrol.com	426.953048
DNS	79	Standard query 0xbb63 ANY itala.cl OPT	427.626483
DNS	669	Standard query response 0xbb63 ANY itala.cl NS ns.skberge.cl NS ns2.skberge.cl NSEC3 RRSIG NSEC3 RRSIG A 200.68.23.154 A 200.5…	427.626683
DNS	66	Standard query 0x956c ANY hts.cl	428.882610
DNS	116	Standard query response 0x956c ANY hts.cl NS ns2.gtdinternet.com NS ns.gtdinternet.com	428.882813
DNS	66	Standard query 0xc802 ANY mph.cl	429.336534
DNS	140	Standard query response 0xc802 ANY mph.cl NS ns2.dospc.cl NS ns1.dospc.cl A 131.72.237.166 A 131.72.237.166	429.336720
DNS	70	Standard query 0xa40b ANY elglobo.cl	430.364419
DNS	210	Standard query response 0xa40b ANY elglobo.cl NS ns-1386.awsdns-45.org NS ns-251.awsdns-31.com NS ns-948.awsdns-54.net NS ns-1…	430.364629
DNS	76	Standard query 0xf600 ANY montajesnieto.cl	431.261980
DNS	216	Standard query response 0xf600 ANY montajesnieto.cl NS ns-335.awsdns-41.com NS ns-1579.awsdns-05.co.uk NS ns-747.awsdns-29.net…	431.262145
DNS	72	Standard query 0xa5f2 ANY southwind.cl	432.421118
DNS	220	Standard query response 0xa5f2 ANY southwind.cl NS dns1.hosting.cl NS dns3.hosting.cl NS dns2.hosting.cl NS dns4.hosting.cl A …	432.421316
DNS	68	Standard query 0x5310 ANY tecas.cl	432.709061
DNS	142	Standard query response 0x5310 ANY tecas.cl NS dns2.hmm.cl NS dns1.hmm.cl A 200.35.156.207 A 200.35.156.207	432.709254
DNS	68	Standard query 0xef70 ANY fosis.cl	433.334221
DNS	250	Standard query response 0xef70 ANY fosis.cl NS secundario.nic.cl NS ns.interior.cl NS ns2.interior.cl NS ns.fosis.cl A 163.247…	433.334456
DNS	71	Standard query 0x8b40 ANY esquerre.cl	434.329648
DNS	141	Standard query response 0x8b40 ANY esquerre.cl NS dns1.esquerre.cl NS dns2.esquerre.cl A 200.73.113.157 A 200.73.113.157	434.329888
DNS	66	Standard query 0x5c23 ANY cpf.cl	434.537297
DNS	219	Standard query response 0x5c23 ANY cpf.cl NS secundario.nic.cl NS ns40.lxh.cl NS ns30.lxh.cl NS ns20.lxh.cl NS ns10.lxh.cl A 2…	434.537536

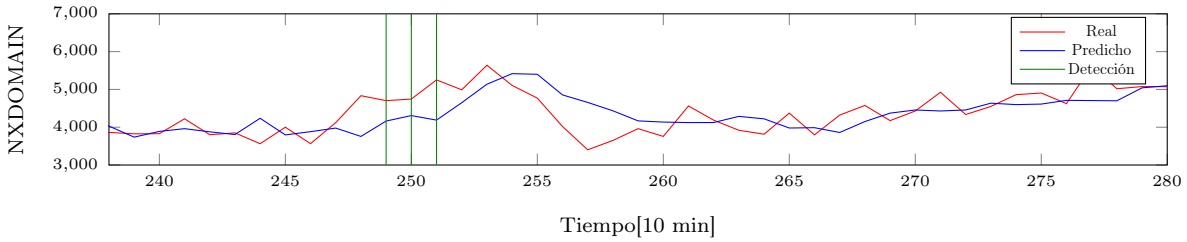
Figura 6.15: Captura de paquetes en anomalía no-etiquetada número 2. Aumento abrupto en consultas por registro ANY, indicando un ataque de amplificación llevado a cabo por un grupo reducido de IPs.

Evento 3

Las consultas tipo NS presentan instancias con errores que alcanzan el 80 %. El ejemplo del evento anómalo de la Figura 6.16 presenta 3 puntos con errores mayores al 70 %. En ellos, se observa un aumento en la cantidad de consultas tipo NS, acompañado de un aumento en la cantidad de respuestas del servidor de tipo NXDOMAIN.



(a) Consultas NS



(b) Respuestas NXDOMAIN

Figura 6.16: Detección de anomalía no-etiquetada en serie de tiempo de consultas NS.

En este caso, los dominios consultados que causan este error en las predicciones sí tie-

nen relación entre ellos. Como puede observarse en la Figura 6.17. Estas se relacionan con '888casino', que corresponde a un grupo de marcas de entretenimiento en línea. Ocurre lo mismo para una marca de soluciones clínicas dentro del intervalo de tiempo involucrado. Estas consultas calzan en una clase B de IP.

Protocol	Length	Info
DNS	140	Standard query response 0xf732 No such name NS 888casinomoneycenterssucks.cl SOA a.nic.cl
DNS	77	Standard query 0xc9d8 NS failu888casino.cl
DNS	128	Standard query response 0xc9d8 No such name NS failu888casino.cl SOA a.nic.cl
DNS	102	Standard query 0x4e90 NS 888casino-united-states-of-americasucks.cl
DNS	153	Standard query response 0x4e90 No such name NS 888casino-united-states-of-americasucks.cl SOA a.nic.cl
DNS	80	Standard query 0xd0ba NS 888casinocomplain.cl
DNS	131	Standard query response 0xd0ba No such name NS 888casinocomplain.cl SOA a.nic.cl
DNS	81	Standard query 0x3b35 NS we-hate-888-casino.cl
DNS	132	Standard query response 0x3b35 No such name NS we-hate-888-casino.cl SOA a.nic.cl
DNS	89	Standard query 0x994a NS 888casinosadvertisingsucks.cl
DNS	140	Standard query response 0x994a No such name NS 888casinosadvertisingsucks.cl SOA a.nic.cl
DNS	81	Standard query 0x6b04 NS ihatethe888-casino.cl
DNS	132	Standard query response 0x6b04 No such name NS ihatethe888-casino.cl SOA a.nic.cl
DNS	87	Standard query 0xfb86 NS 888casinohateshomeowners.cl
DNS	138	Standard query response 0xfb86 No such name NS 888casinohateshomeowners.cl SOA a.nic.cl
DNS	73	Standard query 0x339d NS 888cas5ino.cl
DNS	124	Standard query response 0x339d No such name NS 888cas5ino.cl SOA a.nic.cl
DNS	80	Standard query 0x8695 NS 888casinosucksass.cl
DNS	131	Standard query response 0x8695 No such name NS 888casinosucksass.cl SOA a.nic.cl
DNS	86	Standard query 0x859d NS ihate888casinosolutions.cl
DNS	137	Standard query response 0x859d No such name NS ihate888casinosolutions.cl SOA a.nic.cl
DNS	82	Standard query 0x30a8 NS ihate888-casinobank.cl
DNS	133	Standard query response 0x30a8 No such name NS ihate888-casinobank.cl SOA a.nic.cl
DNS	79	Standard query 0xc8d5 NS ti888casinosucks.cl
DNS	130	Standard query response 0xc8d5 No such name NS ti888casinosucks.cl SOA a.nic.cl
DNS	84	Standard query 0x1ee8 NS ihatethose888-casinos.cl
DNS	135	Standard query response 0x1ee8 No such name NS ihatethose888-casinos.cl SOA a.nic.cl
DNS	85	Standard query 0xa9e7 NS 888casinoscrewedmeover.cl
DNS	136	Standard query response 0xa9e7 No such name NS 888casinoscrewedmeover.cl SOA a.nic.cl
DNS	77	Standard query 0x6b67 NS 888casinosucks.cl
DNS	128	Standard query response 0x6b67 No such name NS 888casinosucks.cl SOA a.nic.cl
DNS	74	Standard query 0xd7fe NS 7888-casino.cl

Figura 6.17: Captura de paquetes en anomalía no-etiquetada número 3. Múltiples consultas por registros NS relacionadas a la marca de entretenimiento '888casino'.

6.4. Discusión

La detección de los ataques emulados fue satisfactoria gracias a la agregación de datos realizada. Esto se traduce en una correcta descripción del sistema al tomar en cuenta los tipos de ataques que se utilizaron. De esta forma, nuevas instancias de estos ataques causarán un cambio en las series de tiempo propuestas, facilitando su detección. Lo que fue mostrado mediante las consultas de tipo ANY para el ataque DNS Amplification, o las respuestas NXDOMAIN para el ataque de Random Subdomain.

Por su parte, la elección del threshold se dejó abierta para experimentación. Los resultados indican que cada serie de tiempo utilizada presenta distintos valores de threshold para los que resulta conveniente reconocer a los eventos como anómalos. Esto debido a la diferencia en la naturaleza de su comportamiento y resultado de las predicciones. Un evento cuya predicción presente un alto error en una de las series de tiempo no necesariamente será un error alto al llevarlo a una comparación con otra serie de tiempo. Un ejemplo son las consultas de tipo ANY, más escasas, las cuales generan mayor error ante cambios más pequeños que, por ejemplo, consultas tipo A. La elección de los threshold es un problema que requiere de mayor atención ante una eventual implementación de este sistema.

No obstante, al guiarse únicamente por los resultados obtenidos respecto a la emulación de ataques, se pierde el objetivo inicial de este trabajo. Resulta relativamente simple construir un detector de eventos para los cuales se tiene conocimiento. Específicamente sabiendo en

cuáles aspectos del tráfico se generarán alteraciones. El problema es que esto no asegura que nuevos eventos puedan afectar al sistema.

Es por esto que se realizaron también agregaciones que no tenían relación con los ataques emulados. Una descripción amplia del sistema, capaz de abarcar la mayor cantidad de aspectos diferentes en el tráfico será la más adecuada para este último propósito. Sumando a esto el que se utilizó un procedimiento netamente no supervisado, se puede argumentar que el sistema propuesto es capaz de detectar también eventos de los cuales no se tenga previo conocimiento. A esta afirmación es a lo que añaden gran valor las detecciones de eventos no-etiquetados, expuestos en la Sección 6.3.2. El procedimiento realizado permitió detectar eventos anómalos contenidos en el set de datos de los cuales no se tenía conocimiento.

Sin embargo, esto no quiere decir que la solución se acerque siquiera al óptimo, eso es lo que jamás se podrá asegurar ante la búsqueda de eventos desconocidos. Mas el desempeño del sistema se considera apto para cumplir con el objetivo de este trabajo.

Asimismo, los resultados aquí expuestos adquieren mérito al establecer una comparación con el estado del arte, de acuerdo a lo descrito en el Capítulo 2 de Trabajo Relacionado. En primer lugar, a pesar de que los valores de precisión y tasa de falsos positivos son fácilmente determinados por un sistema basado en aprendizaje supervisado respecto de la detección de anomalías previamente establecidas, como lo es en [12] bajo el contexto de DNS, la comparación resulta difícil dada la variable del threshold que se expone para análisis en el presente trabajo. No obstante, la ventaja del enfoque no-supervisado para un problema de estas características se observa en la mayor cantidad y diversidad de eventos anómalos que se abordan en esta Tesis, en comparación con el acotado problema de dicho trabajo. Lo que se ve acentuado por el hallazgo de eventos no previamente establecidos para el estudio.

Por otro lado, ante la comparación con otros trabajos bajo el mismo enfoque de Machine Learning bajo aprendizaje no-supervisado pero en el contexto de tráfico de red general, aun cuando agregaciones sobre máscaras de IP fueron cruciales para la detección de los eventos anómalos como ataques de tipo DoS o escaneos de puerto o red en [16] (/8, /16 y /24), este tipo de agregación no resultó ser determinante en la detección de eventos y de botnet en el trabajo presente (/16, /20, /24, /28, /32). En particular, las detecciones recabadas en esta Tesis se produjeron de manera más efectiva principalmente en agregaciones sobre aspectos inherentes a DNS, como en los tipos de registros o el código de respuesta a las consultas. Sobre esta diferencia, debe ser influyente el número de usuarios y subredes diferentes que se manejan en ambos escenarios. Los set de datos públicos como MAWI y KDD99 provienen de redes privadas pertenecientes a universidades e instituciones militares respectivamente. Por otra parte, los datos del trabajo presente corresponden a un servidor que atiende conexiones en todo el mundo, principalmente en Chile.

Probablemente la comparación más importante recae sobre los resultados que se obtuvieron en datos de otros ccTLD, como lo que concierne a los sistemas Prophet [42] (*.nz*) y QLAD [43] (*.be*), al corresponder al escenario y objetivos más similares al presente trabajo. Satisfactoriamente, se obtienen conclusiones en común. En particular, una anomalía presente en el tráfico destacada en [42] coincide con uno de los eventos no-etiquetados hallado en esta Tesis. Específicamente, al Evento 3 en la Subsección 6.3.2, donde se observa un aumento en la cantidad de consultas de tipo NS por dominios no-existentes. La inspección de dicha

ocurrencia es visual y no se profundiza en su causa, mas es reconocida como anomalía, de igual forma que por el procedimiento aquí aplicado.

De la misma manera, las anomalías halladas por el sistema QLAD [43], conceptualmente muy similar al aquí propuesto, resultaron también análogas a las expuestas por este trabajo. A pesar de que en dicho trabajo no entregaron una explicación de cómo clasificaron cada evento anómalo que detectaron, los autores afirmaron detectar *"Spam sender"* (análogo a Evento 1), *"Reflection Attack"* (análogo a DNS Amplification), *"DoS"* (el que pudo ser tanto UDP Flood como Random Subdomain), *"Domain enumeration"* (posiblemente análogo al Evento 3), y finalmente una instancia de *"phishing"*, evento que no fue observado en esta Tesis.

Sin embargo, algunos aspectos importantes difieren entre ambos sistemas. En primer lugar, el estudio de QLAD se realizó en dos días de tráfico de datos, realizando detecciones en ventanas de 10 minutos. El presente trabajo, por su parte, considera todo un mes de información (1 semana para pruebas), con una ventana de detección de 30 minutos. Esto podría explicar la detección de eventos más específicos como lo es *phishing*. Mas la prioridad sobre eventos que afectan al servidor del ccTLD y la consistencia de los resultados en el tiempo, justifican no considerar esto particularmente como una debilidad. En segundo lugar, en el presente trabajo se aborda una dificultad que se reitera en el trabajo del dominio belga: *"Since manually identifying all true anomalies in the traffic is infeasible, it is impossible to determine which anomalies are missed by the detectors. Hence, we cannot compute metrics like recall."* - Dado que identificar manualmente todas las anomalías en el tráfico es inviable, es imposible determinar qué anomalías no son encontradas por los detectores. Por lo tanto, no podemos calcular métricas como *recall* -. La emulación de anomalías etiquetadas en el set de datos permitió calcular esta medida dado un *threshold* en esta Tesis. Finalmente, en tercer lugar, la disimilitud entre el enfoque de Machine Learning y el modelo estadístico utilizados en ambos trabajos manifiesta algunas implicancias. La red LSTM es capaz de almacenar deliberadamente información a largo plazo, mientras que la solución de variación de entropía en QLAD se restringe a la ventana de tiempo sobre la que actúa. Además, una notoria ventaja respecto a aquel trabajo se desprende de otra dificultad planteada explícitamente en el trabajo de *.be*, que corresponde a la correlación entre las variables dadas por las múltiples series temporales, la cual no consideraron al realizar análisis sobre ellas de forma independiente. En esta Tesis se aborda este aspecto en la Sección 4.4, que corresponde en general a una propiedad importante de los modelos de Machine Learning.

Capítulo 7

Conclusión

A través de los capítulos de este trabajo se detalló la teoría y la implementación de un sistema predictor capaz de detectar anomalías, particularmente en tráfico de tipo DNS. Se plantearon una hipótesis y objetivos, y se analizó el trabajo relacionado pertinente. Los datos reales utilizados para la experimentación fueron descritos en detalle y se modificaron de acuerdo a ciertos ataques que se utilizaron para validar al sistema respecto a sus objetivos. La experimentación incluyó un modelo predictivo y herramientas de evaluación con una apropiada explicación. Luego se estableció una heurística para clasificar porciones del tráfico como anómalas o normales. Con un parámetro utilizado como variable para la experimentación, se desempeñó la detección de anomalías en el tráfico. Lo que arrojó resultados satisfactorios respecto de los eventos añadidos a propósito para validar al sistema. Sin embargo, esta aplicación arrojó detecciones de eventos que se encontraban ya en el set de datos original.

Volviendo sobre la hipótesis planteada en el primer capítulo de esta Tesis: "es posible utilizar predicción de tráfico DNS para detectar eventos anómalos en él". De acuerdo a los experimentos en los que se detectaron las instancias incluidas de ataques emulados, se concluye la factibilidad de usar predicción de tráfico DNS para detectar eventos anómalos en él. Respecto a "es posible utilizar aprendizaje no-supervisado para detectar eventos tanto conocidos como desconocidos", la detección de eventos que no estaban previamente etiquetados permite aseverar esta propuesta.

No obstante, esta conclusión requiere de algunos aspectos primordiales. En primer lugar, se debe realizar una correcta descripción del sistema, capaz de cubrir los posibles lugares donde un evento malicioso podría dejar rastro. En segundo lugar, se debe tratar con la posible existencia de falsos positivos. Ser muy permisivo puede generar una alta cantidad de *falsas alarmas*. Sin embargo, en este problema resulta más dañino no reconocer un ataque real a pensar que un evento normal es malicioso.

Del Capítulo 5 sobre la predicción de tráfico DNS, se confirma del trabajo relacionado, los buenos resultados del uso de redes neuronales recurrentes para predecir tráfico del tipo DNS. Gracias a la acción de patrones periódicos presentes en el tráfico, marcados por el comportamiento de los usuarios en Internet.

Del Capítulo 6 sobre la detección de anomalías, se valida el procedimiento utilizado con el fin de detectar ataques respecto a los ataques emulados añadidos al set de datos. Sin embargo, lo que otorga valor adicional al sistema es la detección de eventos anómalos no etiquetados presentes en el set de datos original, como productos de la aplicación del procedimiento en busca de los ataques etiquetados. Esto mediante el uso de un sistema basado en aprendizaje no supervisado, cuyo principal objetivo es preparar no sólo contra instancias conocidas de ataques, sino que cualquier instancia que genere un cambio, en el sistema. Incluyendo ataques de los cuales podría no tenerse conocimiento previo.

Por último, de la discusión realizada respecto a la comparación con trabajos del estado del arte, se concluye que el trabajo presente contribuye conocimiento al abordar problemas explícitamente mencionado por los autores, como la obtención de una medida de evaluación y la correlación entre variables. En adición, la similitud entre los eventos hallados en trabajos similares indica que los resultados obtenidos en la presente Tesis están a la altura de los trabajos del estado del arte. Más aún, se concluye la presencia y detección de estos eventos en tráfico DNS del *ccTLD* de Chile, incluyéndolo a las observaciones conjuntas de otros dominios internacionales.

7.1. Herramienta de Emulación y Set de Datos

Del trabajo realizado en la presente Tesis, se desprenden principalmente dos productos de utilidad. El primero es la herramienta de emulación de ataques mediante la creación e inserción de paquetes en archivos de captura tipo *.pcap*, de acuerdo a un conjunto de parámetros que incluyen tiempos de respuesta del servidor, número de orígenes realizando el ataque en forma coordinada, entre otros. Esta herramienta fue desarrollada en Python 3 junto a colegas del laboratorio NICLabs, para su uso en otros trabajos de investigación.

El segundo es la creación del set de datos en base agregaciones en los datos. Los datos originales pueden contener información sensible que amerita no liberarlos. Sin embargo, esta información agregada estará disponible, mediante solicitud, en el laboratorio NICLabs. Ambos, set de datos normal y el alterado con instancias de ataques emulados.

7.2. Trabajo Futuro

El siguiente paso en el desarrollo de este sistema corresponde a la clasificación automática de los eventos. Fue posible observar en la Subsección 6.3.1, cómo cada ataque afecta aspectos particulares en el tráfico, con los cuales fue posible detectarlos. Estas características resultan un posible buen input para una posible clasificación automática de las instancias. Habiendo reconocido las anomalías en el tráfico, se puede decir a que tipo de ataque o evento corresponde. Esto debe considerar, sin embargo, que algunos eventos pueden no tener información previa, por lo que debe también considerarse una clase de *evento desconocido*, para realizar un posterior análisis de la nueva instancia.

Respecto a lo realizado en este mismo documento. La elección definitiva del threshold es un problema que no solo requiere de diferenciación respecto a cada porción agregada del tráfico, sino que requiere actualización en caso de llevar este sistema a una implementación

en línea sobre tráfico real. Esto sí se encuentra considerado por el modelo predictivo, por otro lado.

Finalmente, realizar nuevas agregaciones, para seguir haciendo más robusta a la descripción del sistema respecto al captar cambios causados por nuevos ataques, posee un amplio margen de mejor. El cual puede considerar a los mismos ataques presentes en el set de datos detectados por el procedimiento. Así como su uso en distintos set de datos de tráfico DNS.

Bibliografía

[1] Dnssec deploy maps. https://www.internetsociety.org/deploy360/dnssec/maps/. Accessed: 2019-010-08.

[2] Internet world stats, users and population statistics. https://www.internetworldstats.com/stats.htm. Accessed: 2019-11-05.

[3] Lactld: Nube anycast de latinoamérica y el caribe. https://anycast.lactld.org/. Accessed: 2019-12-02.

[4] Mobile vs. desktop usage (latest 2019 data). https://www.broadbandsearch.net/blog/mobile-desktop-internet-usage-statistics. Accessed: 2019-11-06.

[5] Most popular top-level domains worldwide as of june 2019. https://www.statista.com/statistics/265677/number-of-internet-top-level-domains-worldwide/. Accessed: 2019-11-06.

[6] Opendns. https://www.opendns.com/about/global-dns-infrastructure/. Accessed: 2019-09-30.

[7] Red de servidores de nombre para .cl. https://www.nic.cl/estadisticas/mapaDNS.html. Accessed: 2019-12-02.

[8] Total number of websites. https://www.internetlivestats.com/total-number-of-websites/. Accessed: 2019-11-05.

[9] Wireshrak. https://www.wireshark.org/. Accessed: 2020-01-020.

[10] J. Abley. Providing minimal-sized responses to dns queries that have qtype=any. RFC 8482, RFC Editor, January 2019.

[11] Ron Aitchison. *Pro DNS and BIND 10*. Apress, USA, 2nd edition, 2013.

[12] Kamal Alieyan, Ammar Almomani, Mohammed Anbar, Mohammad Alauthman, Rosni Abdullah, and B. B. Gupta. Dns rule-based schema to botnet detection. *Enterprise Information Systems*, 0(0):1–20, 2019.

[13] Ammar Almomani, Mohammad Alauthman, Firas Albalas, O Dorgham, and Atef Obeidat. An online intrusion detection system to cloud computing based on neucube algo-

rithms. *International Journal of Cloud Applications and Computing (IJCAC)*, 8(2):96–112, 2018.

[14] Amjad Alsirhani, Srinivas Sampalli, and Peter Bodorik. Ddos attack detection system: Utilizing classification algorithms with apache spark. In *New Technologies, Mobility and Security (NTMS), 2018 9th IFIP International Conference on*, pages 1–7. IEEE, 2018.

[15] Nicola Bui, Matteo Cesana, S Amir Hosseini, Qi Liao, Ilaria Malanchini, and Joerg Widmer. A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques. *IEEE Communications Surveys & Tutorials*, 19(3):1790–1821, 2017.

[16] Pedro Casas, Johan Mazel, and Philippe Owezarski. Unsupervised network intrusion detection systems: Detecting the unknown without knowledge. *Computer Communications*, 35(7):772–783, 2012.

[17] S. Chauhan and L. Vig. Anomaly detection in ecg time signals via deep long short-term memory networks. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–7, Oct 2015.

[18] D. Eastlake and C. Kaufman. Domain name system security extensions. RFC 2065, RFC Editor, January 1997.

[19] Shir Landau Feibish, Yehuda Afek, Anat Bremler-Barr, Edith Cohen, and Michal Shagam. Mitigating dns random subdomain ddos attacks by distinct heavy hitters sketches. In *Proceedings of the Fifth ACM/IEEE Workshop on Hot Topics in Web Systems and Technologies*, HotWeb '17, pages 8:1–8:6, New York, NY, USA, 2017. ACM.

[20] Laura Feinstein, Dan Schnackenberg, Ravindra Balupari, and Darrell Kindred. Statistical approaches to ddos attack detection and response. In *null*, page 303. IEEE, 2003.

[21] Tak-chung Fu. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181, 2011.

[22] Marta C Gonzalez, Cesar A Hidalgo, and Albert-Laszlo Barabasi. Understanding individual human mobility patterns. *nature*, 453(7196):779, 2008.

[23] H. Griffioen and C. Doerr. Taxonomy and adversarial strategies of random subdomain attacks. In *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5, June 2019.

[24] B B Gupta. *[CFC] Computer and Cyber Security: Principles, Algorithm, Applications and Perspectives*. 10 2017.

[25] B B Gupta, Dharma Agrawal, and Shingo Yamaguchi. *Handbook of Research on Modern Cryptographic Solutions for Computer and Cyber Security*. 05 2016.

[26] BB Gupta and Omkar P Badve. Taxonomy of dos and ddos attacks and desirable defense mechanism in a cloud computing environment. *Neural Computing and Applications*,

28(12):3655–3682, 2017.

[27] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 387–395, New York, NY, USA, 2018. ACM.

[28] Ali Imran Jehangiri, Ramin Yahyapour, Edwin Yaqub, and Philipp Wieder. Distributed predictive performance anomaly detection for virtualised platforms. *International Journal of High Performance Computing and Networking*, 11(4):279–290, 2018.

[29] Cricket Liu and Paul Albitz. *DNS and BIND (5th Edition)*. O'Reilly Media, Inc., 2006.

[30] Douglas C. MacFarland, Craig A. Shue, and Andrew J. Kalafut. Characterizing optimal dns amplification attacks and effective mitigation. In Jelena Mirkovic and Yong Liu, editors, *Passive and Active Measurement*, pages 15–27, Cham, 2015. Springer International Publishing.

[31] Diego Madariaga, Martín Panza, and Javier Bustos-Jiménez. Dns traffic forecasting using deep neural networks. In Éric Renault, Paul Mühlethaler, and Selma Boumerdassi, editors, *Machine Learning for Networking*, pages 181–192, Cham, 2019. Springer International Publishing.

[32] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks foranomaly detection in time series. *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, 2015.

[33] Jelena Mirkovic and Peter Reiher. A taxonomy of ddos attack and ddos defense mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53, 2004.

[34] P. Mockapetris. Domain names - implementation and specification. RFC 1035, RFC Editor, November 1987.

[35] Yasuo Musashi, Masaya Kumagai, Shinichiro Kubota, and Kenichi Sugitani. Detection of kaminsky dns cache poisoning attack. In *Intelligent Networks and Intelligent Systems (ICINIS), 2011 4th International Conference on*, pages 121–124. IEEE, 2011.

[36] S. Nanda, F. Zafari, C. DeCusatis, E. Wedaa, and B. Yang. Predicting network attack patterns in sdn using machine learning approach. In *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 167–172, Nov 2016.

[37] Eduardo Mucelli Rezende Oliveira, Aline Carneiro Viana, Carlos Sarraute, Jorge Brea, and Ignacio Alvarez-Hamelin. On the regularity of human mobility. *Pervasive and Mobile Computing*, 33:73–90, 2016.

[38] Martín Panza. Detecting abnormal dns events and attacks in real-time. In Sandra Céspedes and Javier Bustos-Jiménez, editors, *Proceedings of the IV School on Systems*

and Networks, SSN. CEUR Workshop Proceedings, 2018.

[39] Martín Panza, Diego Madariaga, and Javier Bustos-Jiménez. Revealing user behavior by analyzing dns traffic. In Éric Renault, Paul Mühlethaler, and Selma Boumerdassi, editors, *Machine Learning for Networking*. Springer International Publishing, 2020.

[40] Vern Paxson. Bro: a system for detecting network intruders in real-time. *Computer networks*, 31(23-24):2435–2463, 1999.

[41] J. Postel. Domain name system structure and delegation. RFC 1591, RFC Editor, March 1994.

[42] Jing Qiao. .nz dns traffic: Trend and anomalies. `https://blog.nzrs.net.nz/nz-dns-traffic-trend-and-anomalies/`. 2018-10-10.

[43] Pieter Robberechts, Maarten Bosteels, Jesse Davis, and Wannes Meert. Query log analysis: Detecting anomalies in dns traffic at a tld resolver. *In: Monreale A. et al. (eds) ECML PKDD 2018 Workshops. ECML PKDD 2018. Communications in Computer and Information Science, vol 967.*, 2015.

[44] Martin Roesch et al. Snort: Lightweight intrusion detection for networks. In *Lisa*, volume 99, pages 229–238, 1999.

[45] V. Ksinant S. Thomson, C. Huitema and M. Souissi. Dns extensions to support ip version 6. RFC 3596, RFC Editor, October 2003.

[46] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

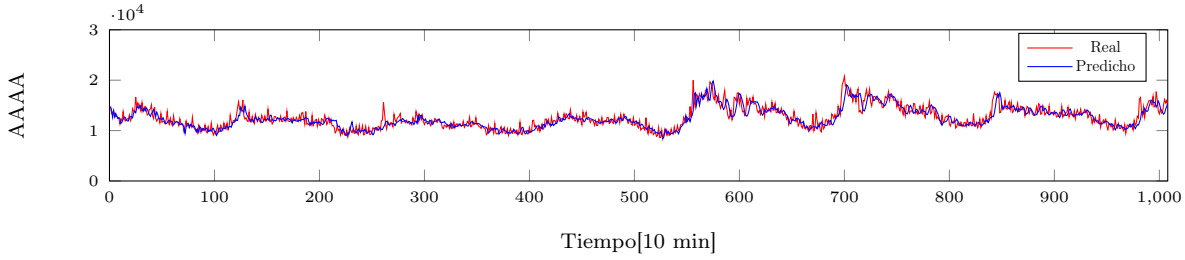
[47] Matthew Thomas and Aziz Mohaisen. Kindred domains: detecting and clustering botnet domains using dns traffic. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 707–712. ACM, 2014.

[48] Huandong Wang, Fengli Xu, Yong Li, Pengyu Zhang, and Depeng Jin. Understanding mobile traffic patterns of large scale cellular towers in urban environment. In *Proceedings of the 2015 Internet Measurement Conference*, pages 225–238. ACM, 2015.

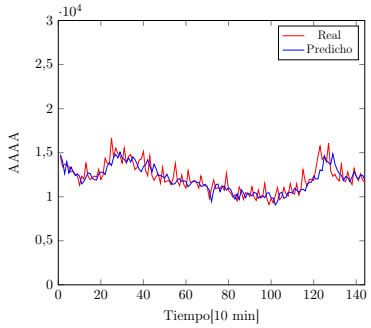
[49] Sandeep Yadav, Ashwath Kumar Krishna Reddy, AL Narasimha Reddy, and Supranamaya Ranjan. Detecting algorithmically generated domain-flux attacks with dns traffic analysis. *IEEE/Acm Transactions on Networking*, 20(5):1663–1677, 2012.

Anexo A

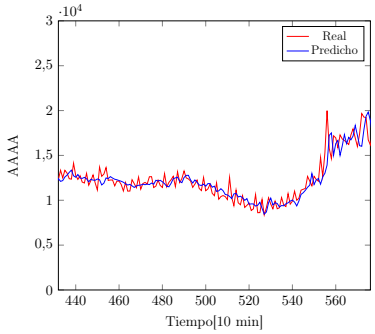
Predicciones de Series de Tiempo Normales



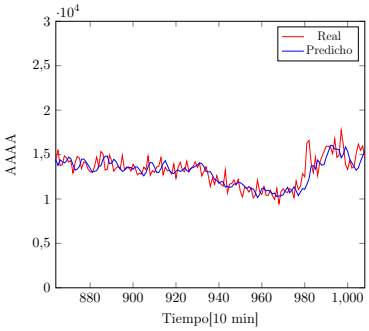
(a) Set de prueba: una semana completa



(b) Primer Día

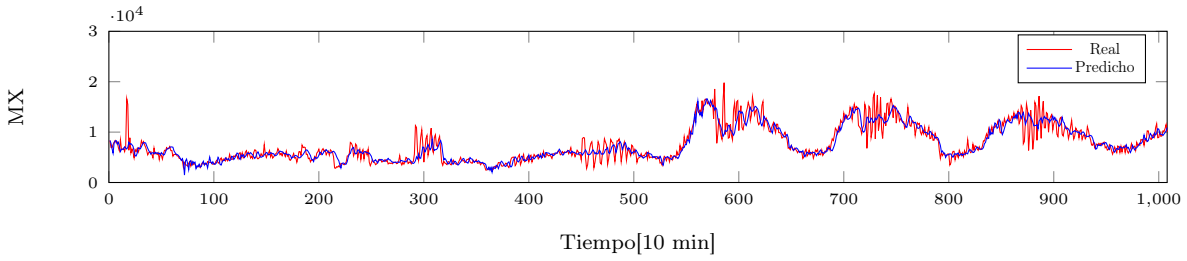


(c) Cuarto Día

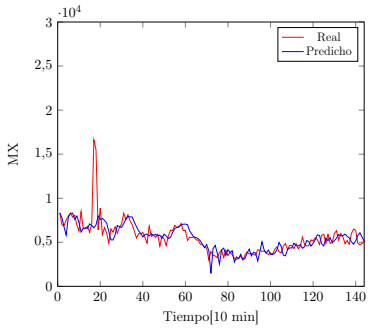


(d) Séptimo Día

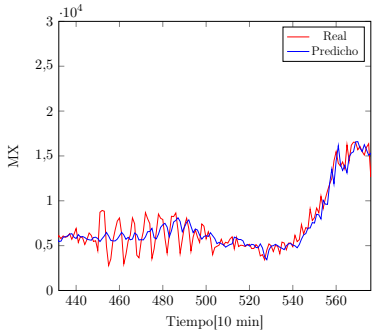
Figura A1: Resultados de predicción en serie de tiempo de cantidad de consultas AAAA por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.



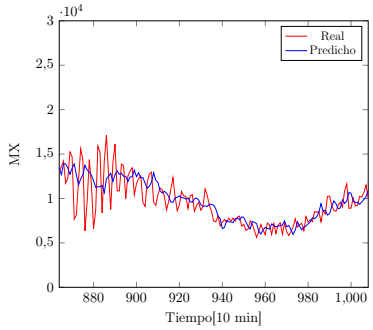
(a) Set de prueba: una semana completa



(b) Primer Día

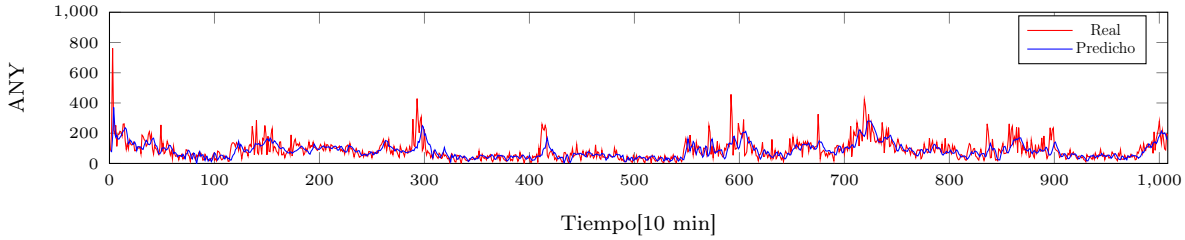


(c) Cuarto Día

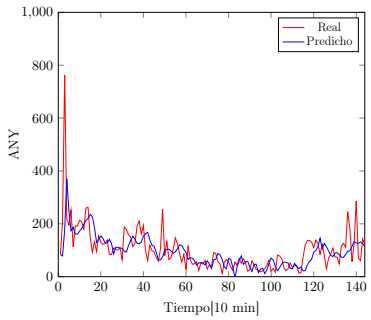


(d) Séptimo Día

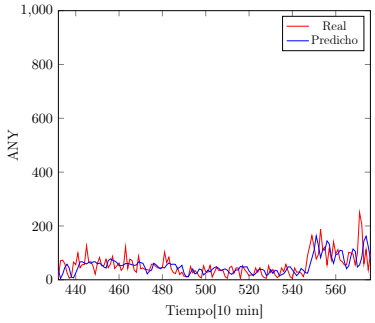
Figura A2: Resultados de predicción en serie de tiempo de cantidad de consultas MX por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.



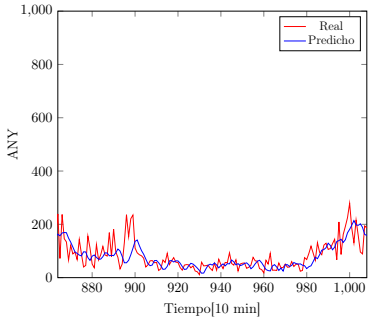
(a) Set de prueba: una semana completa



(b) Primer Día



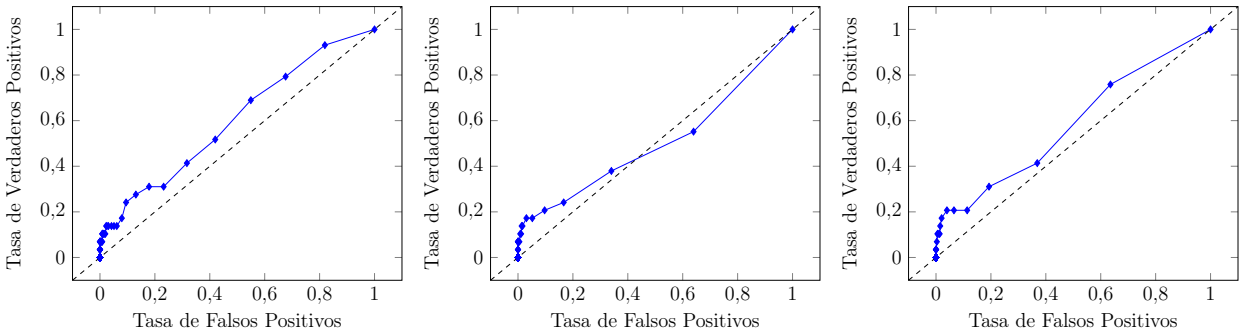
(c) Cuarto Día



(d) Séptimo Día

Figura A3: Resultados de predicción en serie de tiempo de cantidad de consultas ANY por unidad de tiempo de 10 minutos. En la parte superior se muestra el resultado obtenido para la semana completa. A continuación se muestran tres de aquellos días individualmente para una mejor apreciación.

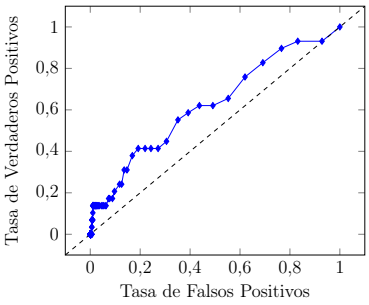
Curvas ROC



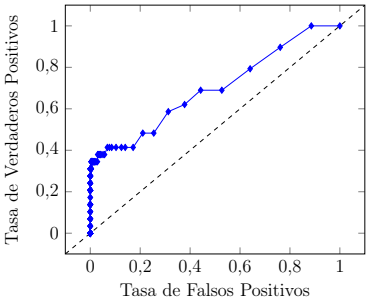
(a) Consultas AAAA

(b) Número de IPs /24

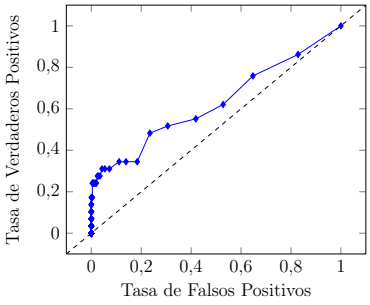
(c) Número de IPs /20



(d) Consultas MX



(e) Respuestas NXDOMAIN



(f) Respuestas NOERROR

Figura A4: Curvas ROC de detección de eventos para otras agregaciones: consultas tipo AAAA, número de IPs con máscara /24, número de IPs con máscara /20, consultas MX, respuestas NXDOMAIN, respuestas NOERROR