UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN
TECNOLOGÍAS DE LA INFORMACIÓN

Biblioteca
UNIVERSITAS Miguel Hernández

DNS Cache Viewer: Análisis y monitorización de la caché DNS

TRABAJO FIN DE GRADO

Febrero - 2026

AUTOR: Joan Amorós Ramírez
DIRECTOR/ES: Óscar Martínez Bonastre
José Ramón García Valdés

Biblioteca
UNIVERSITAS Miguel Hernández

RESUMEN

Este Trabajo de Fin de Grado consiste en el diseño e implementación de un visor de caché DNS (DNS Cache Viewer) orientado al análisis y monitorización del funcionamiento interno de la caché DNS en servidores BIND9.

En este contexto, el principal problema abordado es la ausencia de herramientas accesibles y visuales que permitan inspeccionar de forma sencilla el contenido de la caché DNS y analizar métricas clave como el tiempo de vida (TTL) de los registros, el comportamiento temporal de la caché, así como las tasas de aciertos (hits) y fallos (misses) en la resolución de dominios.

Para ello, el visor desarrollado permite extraer y procesar información de la caché DNS mediante el volcado de ficheros internos de BIND, identificando los dominios almacenados, los tipos de registros, el TTL restante y otros parámetros relevantes.

Asimismo, la aplicación ha sido implementada principalmente en Python, empleando bibliotecas para el procesamiento de texto, análisis de datos y generación de gráficos, y se apoya en scripts Bash para la automatización del volcado periódico de la caché y la ejecución de experimentos controlados.

El desarrollo del proyecto ha seguido una metodología incremental, combinando el diseño del sistema de adquisición de datos, la implementación del visor y la validación experimental mediante escenarios con distintos valores de carga de consultas DNS.

Como conclusión, el DNS Cache Viewer constituye una herramienta útil para comprender el funcionamiento interno de la caché DNS y evaluar el impacto de los parámetros de configuración sobre el rendimiento del sistema, contribuyendo a la optimización de infraestructuras DNS y a la mejora de la eficiencia de los sistemas de resolución de nombres.

Biblioteca
UNIVERSITAS Miguel Hernández

ABSTRACT

This Final Degree Project consists of the design and implementation of a DNS cache viewer aimed at analysing and monitoring the internal functioning of the DNS cache on BIND9 servers.

In this context, the main problem addressed is the lack of accessible and visual tools that allow for easy inspection of the DNS cache content and analysis of key metrics such as the time-to-live (TTL) of records, the temporal behaviour of the cache, as well as the hit and miss rates in domain resolution.

To this end, the viewer allows information to be extracted and processed from the DNS cache by dumping internal BIND files, identifying stored domains, record types, remaining TTL, and other relevant parameters.

Furthermore, the application has been implemented mainly in Python, using libraries for text processing, data analysis and graph generation, and relies on Bash scripts for the automation of periodic cache dumps and the execution of controlled experiments.

The project was developed using an incremental methodology, combining the design of the data acquisition system, the implementation of the viewer, and experimental validation using scenarios with different DNS query load values.

In conclusion, the DNS Cache Viewer is a useful tool for understanding the internal workings of the DNS cache and evaluating the impact of configuration parameters on system performance, contributing to the optimisation of DNS infrastructures and improving the efficiency of name resolution systems.

Biblioteca
UNIVERSITAS Miguel Hernández

AGRADECIMIENTOS

Me gustaría expresar mi agradecimiento a todas las personas que han formado parte de mi trayectoria académica durante la carrera y en mi experiencia Erasmus. Profesores, compañeros y compañeras me han acompañado, enseñado y motivado, contribuyendo a que cada paso en este camino haya sido significativo y enriquecedor.

Quiero dar las gracias a mi familia, cuyo apoyo incondicional ha sido fundamental. Su confianza, paciencia y ánimo me han permitido superar obstáculos y seguir adelante incluso en los momentos más complicados. Este trabajo refleja también el esfuerzo por no rendirme ante las asignaturas más difíciles, cuya superación ha sido clave para llegar hasta aquí.

A mis amigos y amigas de siempre, gracias por estar ahí, por la motivación, las risas y los momentos de desconexión que han hecho más llevadero este proceso. Su compañía ha sido un pilar imprescindible durante estos años.

Finalmente, deseo expresar mi profundo agradecimiento a mis tutores, Óscar y José Ramón, por su dedicación, orientación y consejos durante la realización de este proyecto. Su guía ha sido esencial para poder abordar este Trabajo Fin de Grado y alcanzar los objetivos propuestos.

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

ÍNDICE GENERAL

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Biblioteca UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 1
INTRODUCCIÓN

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

1.1. ENTORNO DE APLICACIÓN

El sistema de nombre de dominios (DNS, Domain Name System) es un protocolo fundamental que posibilita el funcionamiento de Internet tal y como lo conocemos. Su función básica consiste en traducir nombres de dominio legibles (como www.google.com) a direcciones IP numéricas (por ejemplo: 172.217.19.14) que los dispositivos de red utilizan para comunicarse. Esta transformación es imprescindible para la navegación y la mayoría de servicios conectados a Internet, ya que sin el DNS sería necesario recordar direcciones IP en lugar de nombres de páginas web **[1]**.

Cuando un usuario introduce un nombre de dominio en el navegador web, este debe ser traducido a una dirección IP válida para los equipos de red, puesto que las comunicaciones en Internet se realizan exclusivamente mediante direcciones numéricas. Este proceso, también conocido como **resolución DNS**, permite localizar el servidor que aloja el recurso solicitado para poder establecer la comunicación correspondiente.

Desde el punto de vista del usuario, la resolución del sistema de nombre de dominios se realiza de forma transparente. Para ello, el navegador web inicia la consulta sin requerir ninguna acción adicional por parte del usuario y todo el intercambio de información necesario tiene lugar en segundo plano de forma inmediata. Internamente, esta operación implica la interacción entre varios componentes de la infraestructura del DNS, diseñada como una arquitectura jerárquica y distribuida, en la que múltiples servidores cooperan para resolver los nombres de dominio de una forma eficiente y escalable **[2]**.

Históricamente, el sistema DNS fue desarrollado en 1983 por Paul Mockapetris como respuesta a la necesidad de escalar el sistema de mapeo de nombres a direcciones de la ARPANET, el cual dependía originalmente de un único archivo de texto centralizado llamado hosts.txt que ya no resultaba manejable **[3]**. Esta evolución permitió pasar de una tabla simple a la arquitectura jerárquica y distribuida que conocemos hoy.

En la carga de un sitio web suelen intervenir cuatro tipos principales de servidores DNS:

- **Servidor DNS recursivo:** se encarga principalmente de recibir las consultas realizadas por los equipos cliente (normalmente desde aplicaciones como navegadores web) y actúa como intermediario entre el cliente y el resto de la jerarquía del DNS, realizando las consultas necesarias en nombre del usuario.

- **Servidores raíz:** forman el nivel superior de la jerarquía DNS. Indican al servidor recursivo cuáles son los servidores de dominios de nivel superior que deben consultarse a continuación, pero no proporcionan de manera directa las direcciones IP finales.
- **Servidores de dominios de nivel superior (TLD, Top Level Domain):** gestionan dominios de nivel general o geográfico (.com, .es, .org…). Por ejemplo, en el caso de google.com, el servidor TLD correspondiente sería el del dominio .com, que orienta al servidor recursivo hacia el servidor autoritativo adecuado.
- **Servidor DNS autoritativo:** contiene la información definitiva de un dominio concreto. Si recibe una consulta válida, devuelve la dirección IP que se encuentra asociada al nombre del dominio solicitado al servidor recursivo, y este la transmite al cliente original.

Gracias a este modelo el sistema DNS es altamente escalable y tolerante a fallos, y al mismo tiempo permite la incorporación de mecanismos como la caché DNS, fundamental para mejorar el rendimiento en la resolución de nombres de dominio.

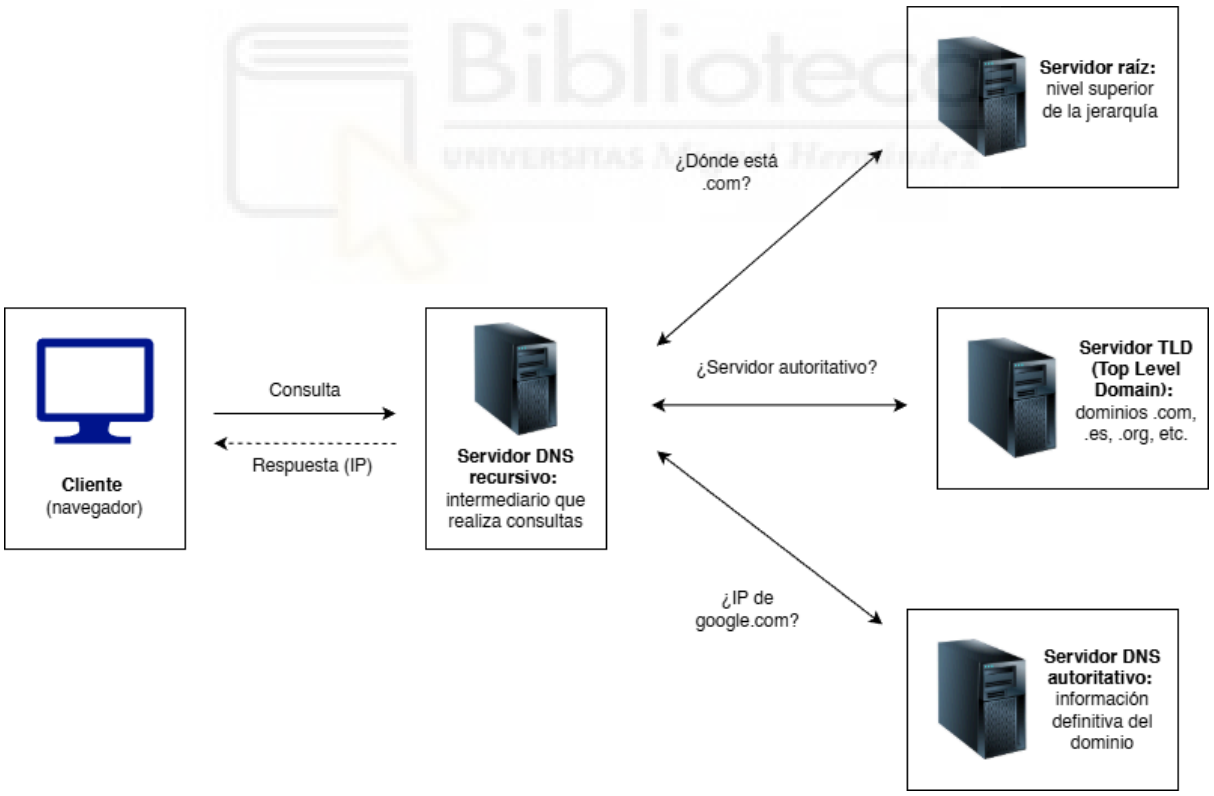


Figura 1: Esquema del proceso de resolución de una consulta DNS

1.1.1. BIND9 como servidor DNS recursivo

Una de las implementaciones de servidores DNS más utilizadas es **BIND9** (Berkeley Internet Name Domain), desarrollado y mantenido por el Internet Systems Consortium (ISC) **[4]**. BIND9 es un software de código abierto ampliamente extendido en sistemas Unix y Linux, y puede actuar tanto como servidor autoritativo o como recursivo. Esto lo convierte en una herramienta versátil y adecuada para entornos reales de producción, así como para contextos educativos y de investigación.

En este trabajo de fin de grado, BIND9 se emplea como **servidor recursivo** dentro de un entorno educativo y experimental, desplegado sobre una máquina virtual con sistema operativo Linux. Esta configuración permite trabajar en un entorno controlado para facilitar el análisis con detalle del comportamiento interno del servidor DNS sin interferir con sistemas en producción. Asimismo, la virtualización aporta flexibilidad para modificar la configuración del servidor, generar carga de consultas DNS y observar el impacto de dichas consultas sobre la caché.

Uno de los componentes esenciales de BIND9 es su mecanismo de caché DNS **[5]**, mediante el cual el servidor almacena temporalmente las respuestas a consultas previamente resueltas junto con su correspondiente valor de tiempo de vida (TTL, Time to Live). Con este mecanismo, las consultas repetidas se pueden resolver de forma más rápida, reduciendo la latencia de respuesta, el tráfico de red y la carga sobre servidores externos. No obstante, el acceso a información detallada sobre el estado de la caché como aciertos (hits), fallos (misses) o el TTL restante de los registros no resulta inmediato ni fácil de interpretar con las herramientas estándar **[6]**.

1.2. JUSTIFICACIÓN DEL PROYECTO

El funcionamiento interno de la caché DNS suele ser poco transparente para usuarios y administradores, a pesar de ser un componente crítico para el rendimiento del sistema DNS. Aunque BIND9 es un servidor robusto y ampliamente utilizado en entornos reales de producción, la información que proporciona sobre el estado de la caché en tiempo real es limitada y, en muchos casos, difícil de interpretar sin un conocimiento profundo del sistema.

BIND9 ofrece ciertos mecanismos para obtener información estadística, volcar el contenido de la caché o analizar archivos de registro. Sin embargo, estas herramientas (explicadas en el capítulo 2) no están orientadas a la visualización directa ni al análisis dinámico del comportamiento de la caché, ya que normalmente generan información en bruto, distribuida en ficheros de texto y sin una estructura pensada para su monitorización en tiempo real. Todo esto dificulta la obtención de métricas clave como el número de aciertos (hits) y fallos (misses) o el tiempo de vida restante (TTL) de los registros almacenados **[7]**.

La ausencia de una visión clara y accesible del estado de la caché DNS complica la comprensión de cómo se comporta el servidor ante cargas reales de consultas, así como la evaluación del impacto que tienen factores como el patrón de acceso, la repetición de consultas o la expiración de registros. Esta falta de visibilidad limita tanto el aprendizaje en entornos educativos como la capacidad de análisis y optimización en tareas de administración de sistemas.

En este contexto, surge la necesidad de desarrollar una herramienta específica (un visor) que permita visualizar y analizar el comportamiento real de la caché DNS, utilizando información obtenida directamente del servidor BIND9. Dicha herramienta tiene un doble enfoque: por un lado, educativo, al facilitar la comprensión del funcionamiento interno de la caché DNS y de conceptos como el TTL o la tasa de aciertos; y por otro, administrativo, al proporcionar métricas útiles para el diagnóstico, la evaluación del rendimiento y la optimización del servicio DNS.

Por lo tanto, este proyecto se justifica como una contribución orientada a mejorar la visibilidad del funcionamiento interno de la caché DNS, cubriendo una carencia existente en las herramientas estándar y proporcionando un soporte útil tanto para el aprendizaje como para el análisis técnico del sistema.

1.2.1. Motivación personal

La elección de este proyecto surge tanto de un interés académico como de una motivación personal relacionada con mi perfil técnico y mi forma de abordar los problemas. A lo largo de mi formación en ingeniería informática, he desarrollado una clara inclinación por el análisis de datos, la programación y la comprensión de sistemas complejos, así como por el estudio de su comportamiento interno y su optimización. El DNS es un componente esencial dentro de la infraestructura de Internet, pero su funcionamiento interno suele ser poco visible para el usuario final. Esta opacidad me llevó a plantear la necesidad de contar con herramientas que permitan observar, medir y entender lo que sucede realmente en la caché DNS.

Además, la realización de este proyecto combina dos aspectos que me resultan especialmente motivadores: por un lado, el trabajo con sistemas reales y su análisis en profundidad; y por otro, la creación de soluciones prácticas que puedan ser utilizadas en entornos educativos o administrativos. La idea de desarrollar un visor que facilite la interpretación de métricas y que permita visualizar la evolución de la caché en tiempo real encaja con mi interés por transformar información compleja en resultados comprensibles y útiles.

En resumen, este proyecto no solo responde a una necesidad técnica y académica, sino que también refleja mi interés por entender cómo funcionan internamente los sistemas y por desarrollar herramientas que permitan monitorizar y optimizar su rendimiento de forma clara y accesible.

1.3. OBJETIVOS

1.3.1. Objetivo general

El objetivo principal de este proyecto es desarrollar un **visor de estadísticas** de la caché DNS, capaz de extraer y mostrar información relevante obtenida directamente del servidor BIND9. El sistema ha de permitir comprender el comportamiento real de la caché y facilitar su monitorización en tiempo real, contribuyendo tanto a la formación educativa como a la gestión y optimización de servicios DNS en entornos reales.

1.3.2. Objetivos específicos

Para alcanzar el objetivo general, se plantean los siguientes objetivos particulares:

- **Analizar el funcionamiento interno de la caché DNS en BIND9** y determinar los puntos de extracción de información relevantes para su monitorización.
- **Desarrollar scripts en Python** que permitan extraer, procesar y normalizar datos de la caché del servidor BIND9 como el estado de los registros, el TTL restante y estadísticas de consultas.
- **Implementar mecanismos de registro y seguimiento de consultas DNS**, distinguiendo entre aciertos (hits) y fallos (misses) en la caché.
- **Diseñar e implementar una interfaz gráfica** que presente de forma clara y visual las métricas principales, permitiendo su consulta en tiempo real y facilitando la interpretación de los datos.
- **Validar el visor mediante pruebas reales**, generando cargas de consultas DNS y comprobando el comportamiento de la caché, así como la consistencia y utilidad de la información mostrada.
- **Documentar y justificar el proceso de desarrollo**, incluyendo el análisis de resultados, posibles limitaciones y futuras líneas de mejora.

1.4. LÍMITES DEL PROYECTO

Para mantener un alcance manejable y garantizar la viabilidad del proyecto, se establecen los siguientes límites:

- **Tipos de registros DNS:** el análisis se centrará exclusivamente en **registros de tipo A**, que son los más habituales y permiten establecer la correspondencia entre nombres de dominio y direcciones IPv4. Esta limitación se justifica por la necesidad de simplificar el alcance del proyecto y focalizar el desarrollo en el comportamiento básico de la caché. Los registros de tipo AAAA (IPv6), CNAME, MX, TXT u otros tipos de registros pueden presentar características y estructuras adicionales que aumentarían la complejidad del análisis sin aportar un valor significativo al objetivo principal.
- **Ámbito de la caché:** solo se considerará la **caché local del servidor BIND9** utilizada en el entorno experimental. No se analizarán ni se tendrán en cuenta otras capas de caché que puedan existir en la red (como la caché del propio sistema operativo, la caché del navegador o la caché de proveedores de red intermedios). Esto permite aislar el comportamiento de BIND9 y obtener resultados reproducibles y comparables en un entorno controlado.
- **Modificaciones al protocolo:** no se realizarán cambios en el protocolo DNS ni en su funcionamiento estándar. El proyecto se centrará en analizar y visualizar el comportamiento de la caché mediante herramientas y mecanismos nativos de BIND9, así como mediante técnicas de monitorización y registro de consultas. Este enfoque garantiza que los resultados sean aplicables a entornos reales y no dependan de modificaciones experimentales que podrían alterar el comportamiento natural del servidor.
- **Alcance temporal y de carga:** las pruebas y validaciones se realizarán en un entorno controlado, con cargas de consultas generadas de forma deliberada. No se pretende evaluar el rendimiento de BIND9 bajo condiciones de tráfico masivo a nivel de producción ni realizar una evaluación exhaustiva de escalabilidad, ya que esto requeriría recursos y un entorno de pruebas específico fuera del alcance de este trabajo.
- **Alcance de la herramienta:** el visor desarrollado se orientará principalmente a la monitorización y visualización de métricas, no a la gestión completa del servidor DNS. No se contempla la implementación de funciones avanzadas como configuración automática, balanceo de carga o políticas de cacheo, ya que el objetivo principal es el análisis y la comprensión del comportamiento de la caché.

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 2 ANTECEDENTES Y ESTADO DE LA CUESTIÓN

Biblioteca
UNIVERSITAS Miguel Hernández

2.1. SITUACIÓN ACTUAL DEL ENTORNO

2.1.1. Funcionamiento general de la caché en un servidor DNS autoritativo - recursivo

Los servidores DNS recursivos (como BIND) implementan un mecanismo de caché con la finalidad de evitar resolver repetidamente los mismos nombres de dominio. Cuando un cliente solicita la resolución de un dominio, el servidor recursivo consulta la jerarquía del DNS solo si no dispone de una respuesta válida almacenada en la caché. De este modo, la caché permite reutilizar respuestas obtenidas previamente, reduciendo así tanto la latencia de las consultas como la carga sobre los servidores autoritativos y la infraestructura de red en general.

La caché DNS almacena las respuestas completas a las consultas, incluyendo los registros solicitados y el valor del TTL (Time To Live), que es un parámetro que indica el tiempo (en segundos) que la caché conserva la información de un dominio antes de requerir una actualización. Mientras el TTL sea mayor que cero, el servidor recursivo puede responder directamente desde su caché sin tener que reenviar la consulta a otros servidores DNS **[8]**.

Por otra parte, el valor del TTL se decrementa de manera progresiva con el paso del tiempo desde que la respuesta es almacenada en caché. Cuando el TTL alcanza el valor cero, la entrada se considera caducada y debe eliminarse de la caché. A partir de ese momento, si el dominio vuelve a ser consultado, el servidor DNS deberá iniciar de nuevo el proceso completo de resolución para obtener una respuesta actualizada desde los servidores autoritativos.

2.1.2. Información expuesta por BIND9 sobre su caché y métricas asociadas

BIND9 proporciona varios mecanismos para obtener información sobre su funcionamiento interno, incluyendo datos relacionados con la actividad del servidor DNS y el uso de su caché. Sin embargo, dicha información está orientada principalmente a tareas de administración y diagnóstico general, y no a la visualización detallada o al análisis dinámico del contenido de la caché.

Entre la información disponible se encuentran estadísticas globales sobre el número total de consultas procesadas, respuestas servidas desde caché, consultas enviadas a servidores externos y otros indicadores de rendimiento (explicados en el apartado 2.2). Estas métricas permiten evaluar el comportamiento general del

servidor y detectar posibles problemas de carga o configuración, pero no ofrecen una visión detallada del comportamiento de la caché a nivel de cada dominio.

Asimismo, BIND9 permite acceder al contenido de la caché mediante volcados puntuales que reflejan su estado en un instante concreto. Sin embargo, esta información se presenta en formato textual y sin una estructura pensada para su análisis automático o visualización en tiempo real. En particular, no se proporciona de forma directa información como el TTL restante de cada entrada, la identificación de hits o misses por dominio, ni un seguimiento continuo de la evolución de la caché durante la ejecución del servidor.

En consecuencia, aunque BIND9 expone información suficiente para la administración básica del DNS, existe una falta de visibilidad sobre el comportamiento interno de la caché en tiempo real. Esto dificulta tanto el aprendizaje en detalle del funcionamiento del sistema como el análisis del impacto que tienen los patrones de consulta sobre el rendimiento del servidor.

2.1.3. Restricciones en la obtención de información de la caché DNS en BIND9

A pesar de que BIND9 ofrece mecanismos para consultar el estado del servidor y su caché, existen limitaciones significativas en cuanto a la accesibilidad y utilidad de la información expuesta. No existe una API nativa ni un comando específico que permita consultar de forma directa y estructurada el contenido de la caché.

Además, la información que sí se encuentra disponible se encuentra dispersa en diferentes ficheros y formatos, generalmente en forma de texto plano. Por ejemplo, el volcado de la caché o las estadísticas se generan como archivos estáticos que deben ser interpretados manualmente o mediante procesamiento posterior. Esta forma de presentación resulta poco práctica para el análisis dinámico, ya que requiere procesos adicionales de extracción, limpieza y estructuración de datos antes de poder obtener métricas significativas.

Por último, las métricas que se pueden obtener de forma estándar en BIND9 son de carácter global, es decir, están orientadas a ofrecer estadísticas generales del servidor (por ejemplo, total de hits y misses o tasa de consultas) y no proporcionan información de cada dominio. Esto impide conocer, por ejemplo, qué dominios concretos se encuentran en caché, cuál es su TTL restante o cuántas veces han sido consultados de manera individual. Esta carencia limita tanto la capacidad de análisis técnico como el uso educativo, ya que no permite observar el comportamiento real de la caché ante patrones de consulta específicos.

2.2. HERRAMIENTAS DISPONIBLES

Existen diversas herramientas que permiten interactuar con los servidores DNS y obtener información sobre su funcionamiento. En el caso de BIND9, estas herramientas están orientadas principalmente a tareas de administración, diagnóstico y pruebas de rendimiento, y no a la visualización detallada del estado interno de la caché DNS en tiempo real.

Entre las herramientas más utilizadas se encuentran **dig**, **rndc** y **DNSperf**, las cuales proporcionan distintos niveles de acceso a la información del servidor DNS. No obstante, ninguna de ellas permite observar de forma directa y continua métricas clave de la caché como el TTL restante por dominio, los aciertos (hits) y fallos (misses) individuales, o la evolución dinámica del contenido de la caché. En los siguientes subapartados se describen estas herramientas, su funcionamiento y sus principales limitaciones en el contexto de este proyecto.

2.2.1. dig

El comando **dig** (Domain Information Groper) es una de las herramientas más utilizadas para realizar consultas DNS manuales. Permite enviar peticiones directas a servidores DNS y obtener información detallada sobre las respuestas recibidas, incluyendo los registros devueltos, los valores de TTL y los servidores que han intervenido en la resolución **[9]**.

```
; <<>> DiG 9.18.33 <<>> @127.0.0.1 google.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49758
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 02b813247c56925b0100000069779cb309a42eb08887ee4c (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             110     IN      A       142.250.184.14

;; Query time: 15 msec
;; SERVER: 127.0.0.1#53(127.0.0.1) (UDP)
;; WHEN: Mon Jan 26 17:56:19 CET 2026
;; MSG SIZE  rcvd: 83
```

Figura 2: Volcado de caché DNS en BIND9

Esta herramienta resulta especialmente útil para realizar pruebas individuales, verificar configuraciones del DNS, comprobar la correcta resolución de dominios o analizar respuestas específicas del servidor. Gracias a su flexibilidad, dig permite especificar el tipo de registro consultado y el servidor DNS al que se envía la petición.

No obstante, dig está diseñado para **consultas puntuales** (realizadas de forma manual o mediante scripts) y no para la monitorización continua del estado de un servidor DNS. Aunque se puede observar el TTL que devuelve una respuesta concreta, esta información corresponde únicamente al resultado de una consulta aislada y no refleja el estado interno de la caché del servidor ni el TTL restante almacenado en ella. Además, dig no permite distinguir si una respuesta ha sido servida desde caché o resuelta mediante una consulta externa, ni ofrece métricas agregadas sobre el comportamiento de la caché. Por estas razones, aunque dig es una herramienta fundamental para pruebas y validación, no resulta tan adecuada para analizar el comportamiento dinámico de la caché DNS en tiempo real.

2.2.2. rndc

El comando **rndc** (Remote Name Daemon Control) es la herramienta principal para la administración y control de servidores BIND9 en ejecución. Permite enviar órdenes al daemon (demonio) named para gestionar su funcionamiento, cargar configuraciones, obtener estadísticas y acceder a información interna del servidor.

Entre las funcionalidades más relevantes para el análisis de la caché DNS se encuentran los comandos rndc dumpdb -cache y rndc stats. El primero genera un volcado del contenido de la caché DNS en un fichero de texto, mostrando las entradas almacenadas en un instante concreto. El segundo produce estadísticas globales del servidor, como el número total de consultas atendidas, respuestas servidas desde caché y otros contadores de rendimiento **[10]**.

```
    ;
    ; Start view _default
    ;
    ;
    ; Cache dump of view '_default' (cache _default)
    ;
    ; using a 0 second stale ttl
    $DATE 20260126165646

; answer
google.com.             83      A       142.250.184.14
```

Figura 3: Respuesta del servidor de un dominio concreto

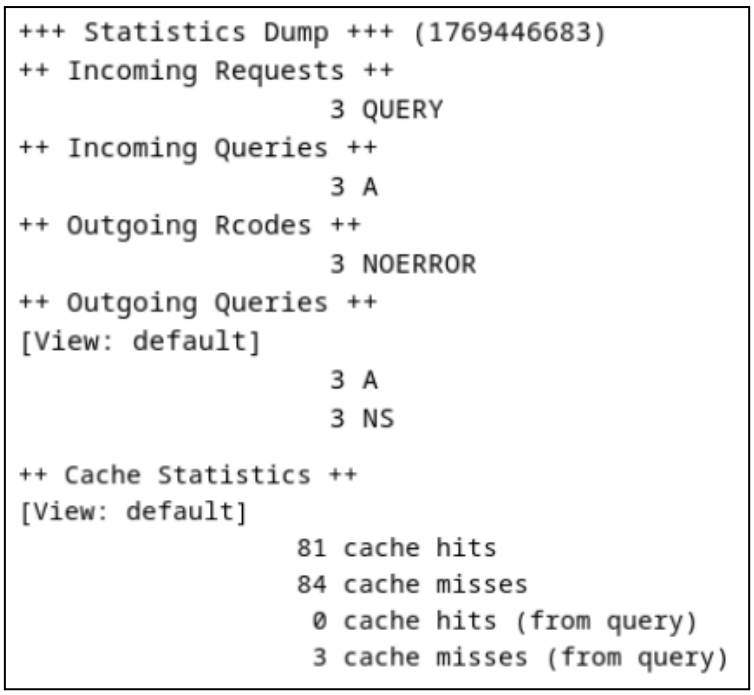


Figura 4: Contenido del archivo de estadísticas internas de la caché DNS

A pesar de su utilidad, estas opciones presentan importantes limitaciones. El volcado de la caché generado por dumpdb -cache es estático, lo que significa que no permite observar la evolución de la caché en tiempo real ni realizar un seguimiento continuo de los TTL. Además, la información se presenta en formato textual, sin una estructura orientada a su análisis automático o visualización directa.

Por otro lado, las estadísticas obtenidas mediante rndc stats son globales, y no permiten identificar métricas por dominio, como el número de hits o misses individuales, ni conocer el TTL restante de cada entrada específica. En consecuencia, aunque rndc proporciona acceso a información interna relevante, su uso está más orientado a la administración general del servidor que al análisis detallado del comportamiento de la caché DNS.

2.2.3. DNSperf

DNSperf es una herramienta diseñada para realizar pruebas de rendimiento sobre servidores DNS. Permite generar grandes volúmenes de consultas DNS de forma controlada midiendo métricas como la latencia, la tasa de respuestas por segundo y la capacidad del servidor para manejar cargas elevadas **[11]**.

Esta herramienta es especialmente útil en entornos de prueba y evaluación de rendimiento, ya que permite simular escenarios de carga realista y analizar cómo responde el servidor ante miles o millones de peticiones DNS. Gracias a DNSperf, es posible identificar cuellos de botella, evaluar configuraciones y comparar el rendimiento entre distintas implementaciones o ajustes del servidor.

```
root@vbox:~# dnsperf -s 127.0.0.1 -d queries.txt -l 10 -q 100
DNS Performance Testing Tool
Version 2.14.0

[Status] Command line: dnsperf -s 127.0.0.1 -d queries.txt -l 10 -q 100
[Status] Sending queries (to 127.0.0.1:53)
[Status] Started at: Mon Jan 26 18:09:12 2026
[Status] Stopping after 10.000000 seconds

Statistics:

  Queries sent:         462469
  Queries completed:    462447 (100.00%)
  Queries lost:         22 (0.00%)

  Response codes:       NOERROR 462447 (100.00%)
  Average packet size:  request 32, response 116
  Run time (s):         10.031679
  Queries per second:   46098.664042

  Average Latency (s):  0.001915 (min 0.000162, max 0.115010)
  Latency StdDev (s):   0.001500
```

Figura 5: Estadísticas de DNSperf

Sin embargo, DNSperf no está diseñada para analizar el contenido interno de la caché DNS. Aunque puede ayudar a inferir indirectamente el uso de la caché a partir de métricas globales de rendimiento, no proporciona información explícita sobre qué dominios se encuentran almacenados, cuál es su TTL restante o si una consulta concreta ha sido servida desde caché o no. Tampoco permite visualizar métricas de hits y misses de forma individualizada.

Por lo tanto, DNSperf resulta adecuada para estudios de rendimiento global, pero no para la monitorización ni la visualización detallada del estado interno de la caché DNS.

2.2.4. Limitaciones comunes a las herramientas existentes

A pesar de sus diferencias funcionales, las herramientas analizadas comparten una serie de limitaciones comunes en lo que respecta a la monitorización detallada de la caché DNS. Ninguna de ellas permite obtener una lista completa y actualizada de los dominios almacenados en la caché, ni visualizar el TTL restante de cada entrada de forma continua.

Asimismo, no es posible identificar de manera directa los aciertos (hits) y fallos (misses) de caché por dominio, ni realizar un seguimiento temporal de cómo evoluciona la caché durante la ejecución del servidor. La información disponible suele ser global, estática o dispersa en múltiples archivos, lo que dificulta su análisis y comprensión.

Estas limitaciones justifican la necesidad de desarrollar una herramienta específica que permita visualizar el comportamiento real de la caché DNS en tiempo real, utilizando información obtenida directamente del servidor BIND9. En este contexto, el visor propuesto en este trabajo de fin de grado se plantea como una solución complementaria a las herramientas existentes, orientada tanto al aprendizaje como al análisis técnico y la optimización del servicio DNS.

Biblioteca
UNIVERSITAS Miguel Hernández

2.3. VALORACIÓN

Tras el análisis de las limitaciones actuales de BIND y de las principales herramientas disponibles para su uso y monitorización se ha realizado una comparación conjunta que permite evaluar el grado de cobertura funcional de cada solución. Esta valoración pone de manifiesto las carencias existentes en cuanto al acceso detallado y en tiempo real a la caché DNS, y sirve como base para justificar el sistema propuesto en este trabajo.

Característica	dig	rndc	DNSperf	Sistema propuesto
Consulta manual de dominios	Sí	No	No	Sí
Monitorización continua	No	No	Parcial (carga)	Sí
Acceso directo a la caché DNS	No	Sí (volcado estático)	No	Sí
Visualización de la lista de dominios en caché	No	No	No	Sí
TTL restante por dominio	Sí	No	No	Sí
Estadísticas de hits/misses globales	No	Sí	No	Sí
Estadísticas de hits/misses por dominio	No	No	No	Sí
Orientación al análisis de caché	No	Limitada	No	Sí
Facilidad de interpretación de datos	Alta	Baja	Media	Alta
Visualización estructurada	No	No	No	Sí

Tabla 1: Comparación de características entre las herramientas existentes y el sistema propuesto

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 3
HIPÓTESIS DE TRABAJO

Biblioteca
UNIVERSITAS Miguel Hernández

3.1. HIPÓTESIS PRINCIPAL

En rasgos generales, el trabajo consiste en desarrollar una herramienta externa de visualización (visor) que al conectarse a un servidor BIND9, pueda leer y procesar dinámicamente la información de su caché DNS (dominio, TTL restante y estadísticas asociadas a hits y misses dentro de la caché) para presentar métricas relevantes en tiempo real.

Se ha elegido Python como lenguaje principal para el desarrollo del visor ya que es ampliamente utilizado en proyectos de análisis de datos y visualización, así como en su ecosistema de bibliotecas científicas que facilitan la captura, transformación y representación gráfica de datos. Python ofrece bibliotecas como **pandas** para la manipulación y análisis de datos, **matplotlib** para visualización estadística y **Tkinter** para la construcción de interfaces de usuario nativas, lo que lo convierte en una opción adecuada para un prototipo de monitorización en tiempo real.

La idea subyacente es que, mediante un visor desarrollado en Python, sea posible no solo observar la caché actual de BIND9 sino comprender patrones de uso y rendimiento. Entre los beneficios esperados de la herramienta destacan:

- Comprender el comportamiento real de la caché DNS, aparte de las estadísticas globales proporcionadas por herramientas como rndc stats.
- Detectar patrones de uso y posibles problemas de configuración, mediante la identificación de dominios que generan un alto número de misses o que presentan TTL reducidos repetidamente.
- Facilitar el aprendizaje y la educación, permitiendo a estudiantes y administradores visualizar la evolución de la caché y su relación con las consultas DNS recibidas.

Esta hipótesis guía tanto el diseño de la herramienta como la evaluación posterior de su efectividad frente a las alternativas existentes.

3.2. RESUMEN DEL FUNCIONAMIENTO DE BIND9

Para contextualizar el diseño del visor, es importante comprender cómo funciona BIND9 tanto como servidor recursivo como, en menor medida, servidor autoritativo, y cómo gestiona su caché interna.

3.2.1. BIND9 como servidor recursivo y autoritativo

BIND9 es una de las implementaciones más populares y utilizadas de servidores DNS, capaz de operar en distintos modos de funcionamiento. Puede actuar como **servidor autoritativo**, en el que responde directamente a consultas sobre zonas de las cuales tiene autoridad, y como **servidor recursivo**, que resuelve consultas para clientes solicitantes realizando consultas hacia la jerarquía de servidores DNS hasta obtener una respuesta válida.

En el modo recursivo, BIND9 recibe consultas de clientes y, si no tiene la respuesta en su propia caché, consulta a servidores externos (por ejemplo, servidores raíz y de TLD) para obtener la respuesta más actual. El proceso de resolución termina cuando se obtiene una respuesta que puede devolverse al cliente, ya sea de forma positiva, negativa o errónea.

3.2.2. Caché de registros DNS y TTL

Una parte esencial del comportamiento recursivo de BIND9 es la caché DNS, que almacena temporalmente las respuestas que han sido resueltas con éxito para poder utilizarlas en futuras consultas sin reiniciar el proceso completo de resolución. A cada entrada en la caché se le asocia un valor de Time To Live (TTL), que corresponde al tiempo máximo en segundos durante el cual ese dato puede permanecer válido antes de ser considerado como expirado **[12]**. Este valor puede definirse en el archivo de zona del dominio DNS y es interpretado por BIND9 como el límite superior de permanencia en la caché; cuando ese tiempo expira, la entrada es eliminada y la próxima consulta para ese dominio implica un nuevo proceso de resolución completo.

3.2.3. Comportamiento básico de la caché

El comportamiento de la caché **[13]** puede describirse mediante dos conceptos fundamentales:

- **Cache hit:** se produce cuando una consulta DNS coincide con una entrada existente en la caché y cuyo TTL aún no ha expirado. En este caso, el servidor devuelve la respuesta directamente y no necesita consultar servidores externos, reduciendo así la latencia y la carga de red.
- **Cache miss:** ocurre cuando no existe una entrada válida en la caché para la consulta recibida (porque nunca se ha almacenado o porque la entrada ha caducado). En esta situación, BIND9 debe iniciar el proceso completo de resolución recursiva para obtener la respuesta desde servidores autoritativos antes de devolverla y almacenarla en la caché para futuras referencias.

La explotación (parcial) de estos comportamientos se realiza principalmente en el proyecto mediante la recolección de información con el comando rndc dumpdb -cache.

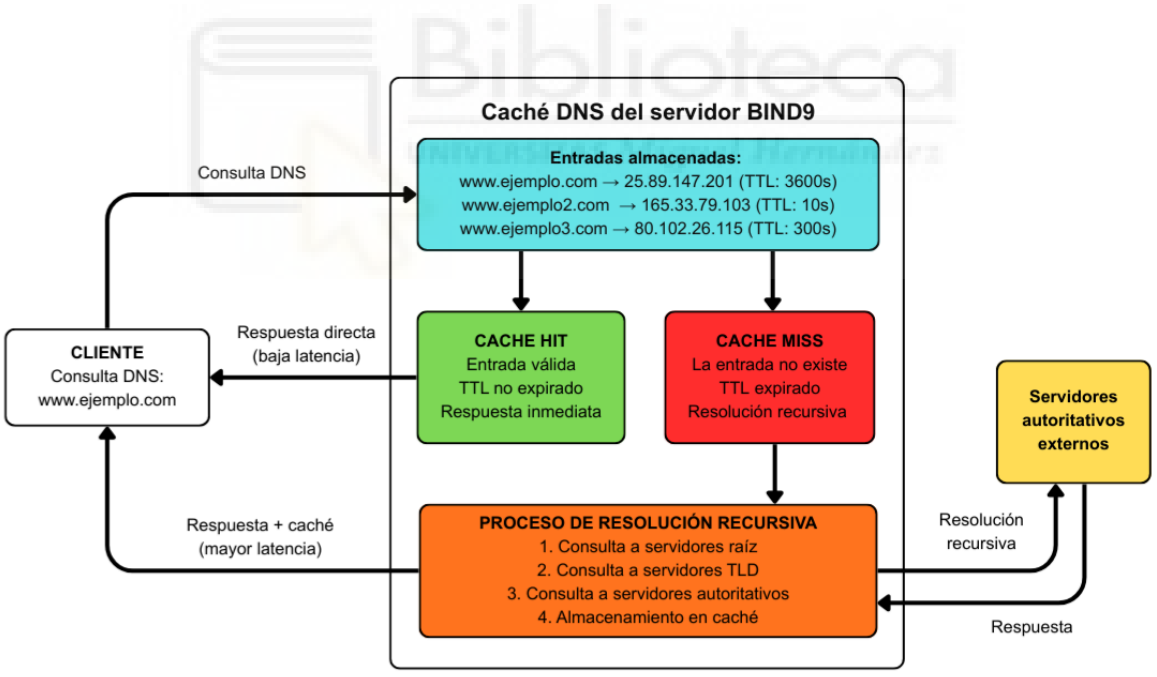


Figura 6: Comportamiento de la caché DNS

3.3. HERRAMIENTAS Y BIBLIOTECAS PYTHON

3.3.1. Tkinter y ttk

Es la biblioteca estándar de Python para crear interfaces gráficas de usuario (GUIs) **[14]**. Proporciona elementos visuales básicos como ventanas, botones, menús y paneles que permiten al usuario interactuar con la aplicación de forma intuitiva. Por otro lado, las extensiones ttk (themed tkinter) añaden controles con estilo moderno que facilitan la creación de interfaces más atractivas y funcionales.

Figura 7: Logo de Tkinter

Biblioteca
UNIVERSITAS Miguel Hernández

3.3.2. pandas

La biblioteca pandas **[15]** proporciona estructuras de datos flexibles y eficientes (dataframes) que son ideales para almacenar y manipular grandes volúmenes de información extraída de la caché, como listas de dominios, tiempos de vida restantes y contadores de hits y misses. Permite filtrar, agrupar y resumir estos datos para análisis posteriores.

Figura 8: Logo de Pandas

3.3.3. matplotlib

Matplotlib **[16]** es una biblioteca ampliamente utilizada para crear gráficos estáticos y dinámicos en Python. Proporciona una API versátil para generar diagramas de líneas, barras o distribuciones, entre otras.

Figura 9: Logo de Matplotlib

3.3.4. Otras bibliotecas de visualización

Otras bibliotecas como Plotly **[17]** o Bokeh **[18]** permiten crear gráficos interactivos que facilitan la exploración de datos, aunque por su enfoque web o más complejo se pueden integrar en etapas posteriores de evolución del visor, particularmente si se quiere una herramienta más visual o navegable.

3.3.5. Módulos auxiliares

Existen módulos estándar como os y datetime, que resultan útiles para el manejo de ficheros (por ejemplo para la lectura de volcados de caché) y marcas temporales que permiten coordinar actualizaciones periódicas de la interfaz y escalado de mediciones temporales.

3.4. MODELO ANALÍTICO DE LA CACHÉ

El comportamiento de la caché DNS puede analizarse no solo desde un punto de vista práctico o experimental, sino también mediante modelos analíticos que permiten describir de forma matemática la relación entre las consultas recibidas, el tiempo de vida de los registros y la probabilidad de obtener hits o misses en la caché.

3.4.1. Conceptos básicos del modelo

El modelo parte de la consideración de un único registro DNS almacenado en la caché de un servidor recursivo. Dicho registro tiene asociado un valor de tiempo de vida **T (TTL)**, durante el cual la entrada permanece válida en la caché tras ser obtenida desde un servidor autoritativo. Mientras el TTL no haya expirado, cualquier consulta adicional al mismo dominio puede resolverse directamente desde la caché, produciendo un cache hit. En cambio, una consulta realizada cuando el registro no se encuentra en caché o cuando su TTL ha expirado da lugar a un cache miss, lo que obliga al servidor a realizar de nuevo el proceso de resolución.

Un elemento fundamental del modelo es el número de consultas que se producen durante el intervalo de validez del TTL. Sea **N(T)** la variable aleatoria que representa el número de consultas adicionales que llegan al servidor para un determinado dominio durante un intervalo de tiempo. El valor esperado de esta variable, **E[N(T)]**, juega un papel central en la definición de las probabilidades de hit y miss.

Cada ciclo completo de caché puede interpretarse como una secuencia formada por un miss inicial (que provoca la inserción del registro en caché) seguido de múltiples hits, hasta que el TTL expira y el ciclo vuelve a comenzar **[19]**.

3.4.2. Definición de métricas de hit y miss

A partir de la variable N(T), el modelo define las siguientes métricas globales de rendimiento de la caché:

$$H\ (T)\ =\ \frac{E\ [N\ (T)]}{E\ [N\ (T)] + 1}$$

$$M\ (T)\ =\ \frac{1}{E\ [N\ (T)] + 1}$$

donde H(T) representa la probabilidad de que una consulta sea resuelta mediante un cache hit, y M(T) representa la probabilidad de que una consulta produzca un cache miss.

Las expresiones anteriores reflejan una idea intuitiva: por cada ciclo de caché existe exactamente un miss (el que provoca la carga del registro) y un número esperado de hits igual a E[N(T)]. Por lo tanto, el rendimiento de la caché depende directamente del número de consultas que se produzcan durante el tiempo de vida del registro.

Cuando el patrón de llegada de consultas se modela como un proceso de Poisson con tasa λ, el número esperado de consultas durante el intervalo T es:

$$E\ [N\ (T)]\ =\ \lambda T$$

Si sustituimos este valor este valor en las expresiones anteriores se obtienen las métricas simplificadas:

$$H_1\ (T)\ =\ \frac{\lambda T}{\lambda T + 1}$$

$$M_1\ (T)\ =\ \frac{1}{\lambda T + 1}$$

Estas fórmulas permiten analizar de forma directa cómo influyen el TTL y la frecuencia de consultas en el comportamiento de la caché DNS. A mayor valor de λ (consultas más frecuentes) o mayor TTL, la probabilidad de hit aumenta y la de miss disminuye.

3.4.3. Renovación del TTL y tiempos entre consultas

Un aspecto relevante del modelo es el concepto de **renovación del TTL**. Cada vez que se produce un cache miss, el servidor DNS obtiene de nuevo el registro desde un servidor autoritativo y reinicia el valor del TTL asociado. A partir de ese instante, el contador del TTL comienza a decrecer de forma continua hasta alcanzar el valor cero, momento en el que la entrada deja de ser válida.

El modelo también introduce el concepto de inter-arrival time, es decir, el tiempo transcurrido entre dos consultas consecutivas al mismo dominio. Si los tiempos entre consultas son pequeños en relación con el TTL, es probable que múltiples consultas se resuelvan como hits. Por el contrario, si los tiempos entre consultas son grandes, el TTL puede expirar entre dos consultas consecutivas, aumentando la tasa de misses.

Este comportamiento explica por qué dominios muy populares suelen presentar tasas de hit elevadas, mientras que dominios consultados de forma esporádica tienden a generar misses con mayor frecuencia, incluso aunque el TTL configurado sea relativamente alto.

3.4.4. Interpretación del modelo y representación conceptual

Desde un punto de vista conceptual, el comportamiento descrito puede representarse como una secuencia temporal de eventos para un mismo dominio:

- **Primera consulta** → cache miss → el registro se inserta en la caché y se establece el TTL
- **Consultas sucesivas dentro del TTL** → cache hits
- **Expiración del TTL** → se elimina el registro
- **Nueva consulta tras la expiración** → nuevo cache miss

Este patrón se repite de forma cíclica y constituye la base del modelo analítico. Una representación gráfica de esta secuencia (mostrando la evolución del TTL y la alternancia entre hits y misses) resulta especialmente útil para comprender el funcionamiento interno de la caché.

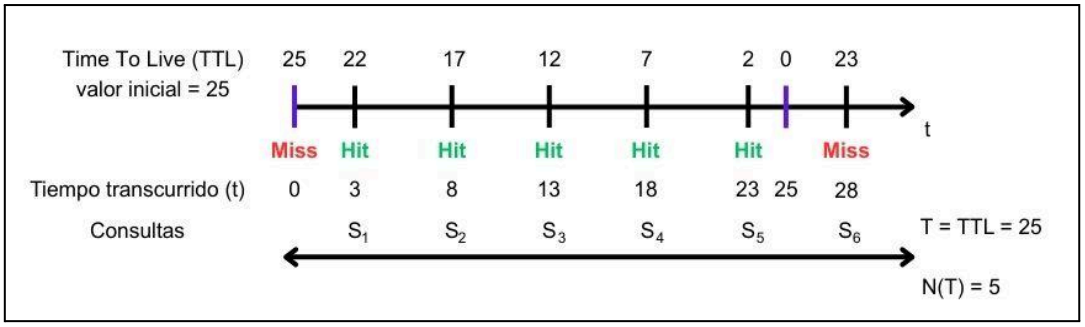


Figura 10: Cronograma de la evolución del TTL

3.4.5. Relación con el visor propuesto y datos reales de BIND9

El modelo analítico descrito proporciona una base teórica sólida para interpretar los datos observados en un servidor DNS real. En particular, las métricas obtenidas mediante el visor desarrollado en este proyecto (como el número de hits, misses, el TTL restante de cada dominio y la frecuencia de consultas) permiten contrastar las predicciones del modelo con el comportamiento real de la caché de BIND9.

De esta manera es posible comparar la tasa de hits observada con los valores teóricos $H(T)$ y $H_1(T)$, analizar cómo influyen distintos valores de TTL en el rendimiento de la caché y estudiar el efecto de patrones de consulta reales frente a los supuestos ideales del modelo (por ejemplo, llegadas Poisson).

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 4
METODOLOGÍA Y RESULTADOS

Biblioteca
UNIVERSITAS Miguel Hernández

4.1. PLANIFICACIÓN DEL PROYECTO

La planificación del proyecto se ha estructurado con el fin de desarrollar de manera progresiva y controlada un visor de estadísticas de la caché DNS basado en BIND9. Para ello, se ha seguido una metodología incremental, en la que cada fase se apoya en los resultados de la anterior, permitiendo así validar el funcionamiento del sistema antes de avanzar a etapas más complejas.

Esta metodología ha permitido dividir el proyecto en tareas independientes, facilitar la detección temprana de errores y asegurar la trazabilidad entre los requisitos definidos y la implementación final del sistema.

4.1.1. Visión general y entornos de ejecución

El sistema desarrollado en este proyecto se apoya en una arquitectura distribuida basada en tres entornos de ejecución diferenciados, cada uno con un rol específico dentro del flujo global de trabajo. Esta separación permite mejorar la organización del proyecto, facilitar las pruebas y garantizar la reproducibilidad de los experimentos.

- **Entorno de control:** corresponde a un equipo de sobremesa compacto (denominado cubo), que actúa como sistema principal de gestión y coordinación. En este entorno se realizan tareas de supervisión general, acceso remoto a la máquina virtual y gestión del flujo de datos entre los distintos componentes del sistema. El cubo cumple con las siguientes especificaciones:

Hardware	
Modelo	CHUWI HeroBox
Fabricante	CHUWI Innovation And Technology Co., Ltd.
Procesador	Intel® Celeron® J4125 (4 núcleos)
Memoria RAM	8 GiB
Gráficos	Intel® UHD Graphics 600
Capacidad de almacenamiento	256,1 GB

Tabla 2: Especificaciones hardware del entorno de control

Software	
Sistema operativo	Debian GNU/Linux 13, 64 bits
Kernel (núcleo)	Linux 6.12.63+deb13-amd64
Entorno de escritorio	GNOME 48
Versión de firmware	GB3B 1.02

Tabla 3: Especificaciones software del entorno de control

- **Entorno de servicio y prueba:** está constituido por una máquina virtual creada mediante Oracle VirtualBox. En esta máquina se ha instalado el servidor de BIND9 y las herramientas necesarias para generar carga DNS y recopilar información sobre el comportamiento de la caché. Las características principales de este entorno son las siguientes:

Nombre del dispositivo	vbox
Modelo de hardware	innotek GmbH VirtualBox
Procesador	12th Gen Intel® Core™ i5-1235U (virtualizado)
Capacidad de disco	21,5 GB
Sistema operativo	Debian GNU/Linux 12 (bookworm), 64 bits
Entorno gráfico	GNOME 43.9

Tabla 4: Características de la máquina virtual

- **Entorno de análisis:** corresponde a los equipos locales del usuario (portátil y PC de sobremesa), desde los cuales se ejecuta el visor gráfico desarrollado en Python. Se procesan los datos extraídos desde la máquina virtual y se representan de forma visual para su estudio.

La separación en tres entornos independientes permite aislar las tareas de control, generación de tráfico y análisis, evitando interferencias entre ellas y mejorando la reproducibilidad de los experimentos.

Además, esta arquitectura distribuida refleja un escenario realista de despliegue, en el que el servidor DNS se encuentra en un sistema remoto y el análisis se realiza desde un cliente externo.

4.1.2. Fases del proyecto

El desarrollo del proyecto se ha organizado de las siguientes fases:

1) **Configuración del entorno de servicio:** instalación y configuración del servidor BIND9 y de la herramienta DNSperf en la máquina virtual, y verificación del funcionamiento correcto del servidor DNS recursivo.
2) **Exploración de mecanismos de extracción de datos:** búsqueda de opciones disponibles en BIND9 para la obtención de estadísticas y volcados de caché.
3) **Desarrollo de scripts de extracción:** implementación de programas en Python para procesar la información generada por BIND9:
    a) extraer_datos_stats.py: procesamiento de estadísticas globales
    b) extraer_datos_dumpdb.py: análisis del contenido de la caché volcada.
4) **Desarrollo del controlador de experimentos:** creación de una interfaz (en el cubo) para la ejecución de pruebas de carga DNS mediante DNSperf (dnsperf_gui.py), permitiendo automatizar experimentos y generar datos reproducibles.
5) **Desarrollo del visor de análisis:** implementación del visor gráfico para la visualización de métricas como hits, misses y TTLs, así como su evolución temporal.
6) **Integración y pruebas:** integración de todos los componentes del sistema y realización de pruebas completas para validar la coherencia entre los datos generados por BIND9 y los resultados mostrados en el visor.

4.2. CAPTURA DE REQUISITOS

4.2.1. Requisitos funcionales

Los **requisitos funcionales (RF)** definen las funcionalidades esenciales del sistema y describen el comportamiento que debe presentar ante determinadas acciones o entradas. Por otro lado, especifican de forma precisa qué debe hacer el sistema, cómo debe responder ante una solicitud concreta y qué resultados espera obtener el usuario al interactuar con él. Estos requisitos permiten delimitar las capacidades básicas del programa y constituyen la base sobre la que se desarrolla e implementa la aplicación. Para cada requisito funcional se detalla su nombre y una descripción, indicando de manera clara la funcionalidad que debe proporcionar el sistema. Los RF de este desarrollo son los siguientes:

1. **Mostrar lista de dominios en la caché:** el sistema debe ser capaz de obtener y mostrar (de manera estructurada) el conjunto de dominios almacenados en la caché DNS en un momento dado. Esto permite al usuario conocer qué entradas se encuentran actualmente cacheadas y constituye la base para el análisis posterior de su comportamiento.
2. **Visualizar TTL restante:** para cada dominio presente en la caché, el visor debe mostrar el valor del TTL restante asociado a cada entrada. Con esta información se analiza la caducidad de los registros y se evalúa el impacto del tiempo de vida en el rendimiento de la caché.
3. **Indicar hits y misses por dominio:** el sistema ha de proporcionar información sobre el número de hits y misses asociados a cada dominio analizado. Estos datos permiten evaluar la efectividad de la caché DNS y entender el patrón de acceso a los distintos dominios.
4. **Filtros y ordenación de datos:** el visor debe permitir aplicar filtros y criterios de ordenación sobre los datos mostrados (nombre de dominio, TTL, número de hits y misses). Esta funcionalidad facilita el análisis y la exploración de grandes volúmenes de información por parte del usuario.
5. **Opción de exportar datos a CSV o Excel:** se debe ofrecer la posibilidad de exportar los datos recopilados y visualizados a formatos estándar como CSV o Excel. De este modo, el usuario puede realizar análisis adicionales externos, generar informes o conservar los resultados para estudios posteriores.
6. **Control remoto de DNSperf desde la interfaz gráfica:** el sistema debe permitir el control remoto de la herramienta DNSperf a través de la interfaz gráfica del cubo, posibilitando su ejecución, configuración y parada sin necesidad de acceder manualmente al entorno de línea de comandos.
7. **Generación de snapshots de caché periódicos:** se han de generar capturas periódicas (snapshots) del estado de la caché DNS permitiendo

almacenar la evolución de la caché a lo largo del tiempo, lo que sirve como base para el análisis comparativo.

8. **Cálculo de hits y misses mediante comparativa de snapshots:** a partir de los snapshots generados, el sistema debe calcular los valores de hits y misses comparando el contenido de la caché entre diferentes instantes temporales. Este enfoque permite estimar el comportamiento de la caché sin depender únicamente de contadores internos del servidor DNS.

4.2.2. Requisitos no funcionales

Los **requisitos no funcionales (RNF)** recogen aquellas condiciones, restricciones o características adicionales que aunque no formen parte directamente de la funcionalidad principal del sistema, influyen en su calidad, rendimiento o usabilidad. Este tipo de requisitos no especifica acciones concretas, sino aspectos relacionados con cómo debe funcionar el sistema. Mientras que los requisitos funcionales determinan las operaciones que el sistema debe realizar, los requisitos no funcionales describen criterios como eficiencia, facilidad de uso, fiabilidad, portabilidad o aspectos técnicos y estéticos. Los RNF del sistema son:

1. **Ejecución en un entorno seguro:** el proyecto se ha de ejecutar en un entorno aislado (como una máquina virtual) garantizando que las pruebas y experimentos realizados no afectan a entornos de producción ni comprometen la estabilidad de servicios reales.
2. **Rendimiento mínimo aceptable:** el sistema debe ofrecer un rendimiento adecuado, de manera que la actualización y visualización de los datos se realice en un tiempo razonable, permitiendo una interacción fluida con la herramienta.
3. **Comunicación fiable por SSH:** la comunicación entre la interfaz gráfica y el entorno remoto debe realizarse de forma segura y fiable mediante SSH, asegurando la correcta ejecución de comandos y la integridad de los datos intercambiados.
4. **Sincronización correcta entre DNSperf y la captura de datos:** se ha de garantizar una correcta sincronización temporal entre la ejecución de DNSperf y la captura de información de la caché, evitando inconsistencias o desajustes en los datos analizados.
5. **Interfaz intuitiva y fácil de usar:** la interfaz gráfica debe ser clara e intuitiva, permitiendo que el usuario pueda acceder a las distintas funcionalidades del sistema sin necesidad de conocimientos avanzados.
6. **Soporte para múltiples dumps históricos:** el sistema debe permitir el almacenamiento y la gestión de múltiples dumps o snapshots históricos, de forma que el usuario pueda consultar y comparar datos de diferentes ejecuciones o periodos temporales.

4.3. DISEÑO DEL SISTEMA DEL VISOR DE CACHÉ DNS

Este apartado describe con detalle la arquitectura del sistema propuesto, el flujo completo de ejecución, la estructura de los archivos desarrollados y la lógica de análisis de la caché DNS.

Además del entorno local, el sistema se distribuye en dos entornos diferenciados: el cubo local y la máquina virtual con BIND9. Ambos entornos se coordinan mediante conexiones SSH seguras para la generación, transferencia y posterior análisis de datos.

4.3.1. Visión general del sistema y flujo global

El sistema completo se compone de tres fases principales:

a) **Configuración y ejecución de la simulación de carga DNS**, realizada desde el cubo local mediante la herramienta DNSperf.
b) **Captura y exportación de estadísticas y dumps de caché**, ejecutada en la máquina virtual donde se aloja el servidor BIND9.
c) **Análisis y visualización de resultados**, llevada a cabo en el equipo local del usuario mediante el visor desarrollado en Python.

El flujo general es el siguiente:

1. El usuario inicia la aplicación **DNSperf_gui.py** en el cubo local.
2. Se selecciona la dirección IP del servidor DNS, el archivo de consultas y los parámetros del experimento.
3. La aplicación establece una conexión SSH con la máquina virtual y ejecuta de forma remota el generador de carga **DNSperf**, junto con el script de exportación en tiempo real **real_time_export.sh**.
4. Durante la ejecución de la prueba, la máquina virtual genera periódicamente estadísticas del servidor DNS (named.stats) y volcados de caché (named_dump.db). Estos archivos son procesados automáticamente y convertidos a CSV, que se transfieren al entorno local.
5. Al finalizar la simulación, el usuario analiza en el entorno local los archivos CSV mediante la herramienta del visor.

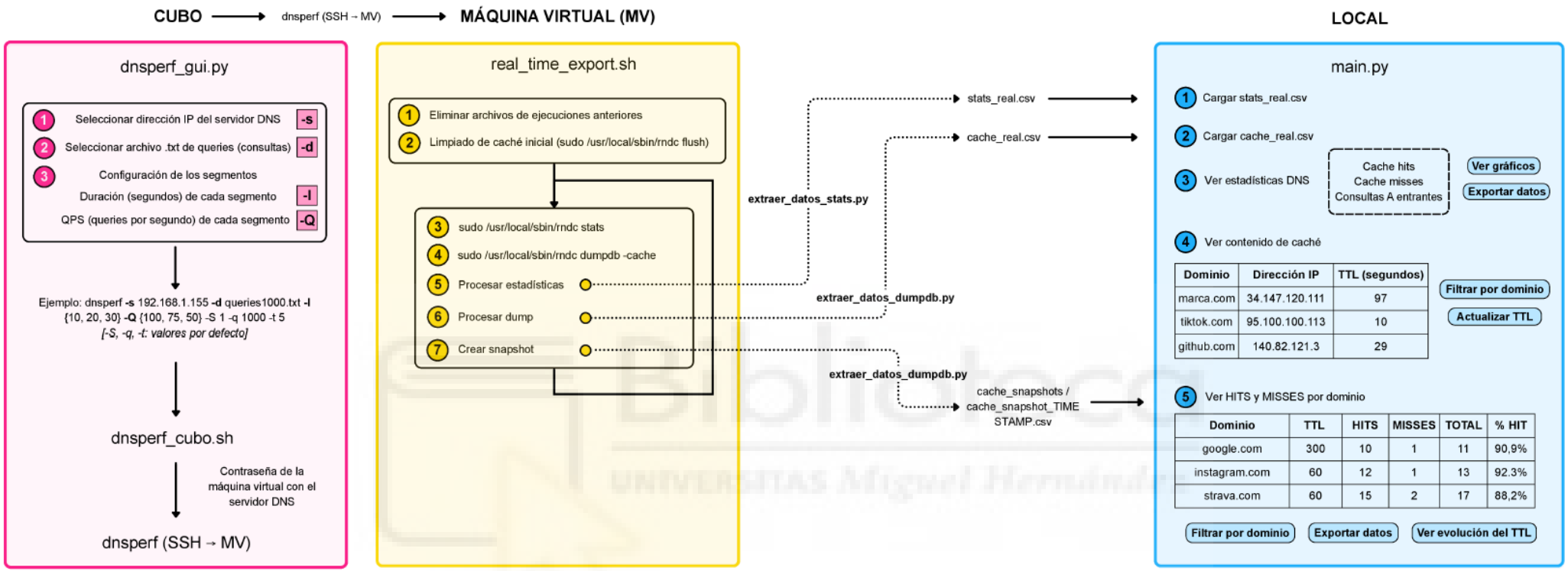


Figura 11: Esquema general del diseño del sistema

4.3.2. Estructura modular del sistema

El sistema se organiza en tres módulos funcionales principales, siguiendo un enfoque modular que facilita su mantenimiento y extensión futura:

- **Módulo de control:** encargado de configurar y ejecutar DNSperf de forma remota. Incluye la interfaz gráfica para definir los parámetros del experimento y coordinar la ejecución de los scripts remotos mediante SSH.
- **Módulo de captura y exportación de datos:** responsable de generar dumps de caché y estadísticas, y convertirlos a formatos estructurados.
- **Módulo de análisis y visualización:** implementa la interfaz gráfica para la exploración y análisis de los datos recopilados.

La separación en módulos permite desacoplar las responsabilidades del sistema y facilita la reutilización de componentes en otros entornos.

4.3.3. Ejecución del programa

El usuario inicia la aplicación **dnsperf_gui.py** desde el cubo local y configura los parámetros del experimento. A partir de estos parámetros, la aplicación genera dinámicamente el comando DNSperf correspondiente y lo ejecuta en la máquina virtual mediante SSH.

Durante la ejecución, se controla la duración del experimento y se sincroniza con la captura de datos de la caché.

Una vez finalizada la simulación el usuario abre el visor de análisis, que permite:

- **Visualizar estadísticas globales del servidor DNS** (número total de consultas, hits y misses)
- **Inspeccionar el contenido de la caché DNS** (dominios y su TTL restante)
- **Analizar los hits y misses por dominio y su evolución temporal** (eficiencia de la caché y patrones de acceso a dominios concretos)

4.3.4. Diseño de la lógica de análisis de caché

Se define un criterio operacional basado en la captura periódica (snapshot) de la caché DNS:

- **Hit:** un dominio aparece en el snapshot N y también en el snapshot N+1 con su TTL vigente.
- **Miss:** un dominio aparece en el snapshot N+1 pero no estaba presente en el snapshot N, o su TTL ha expirado entre ambos snapshots.

Este criterio permite inferir el comportamiento de la caché sin modificar el código interno de BIND9. Para garantizar la consistencia del análisis, los snapshots se realizan cada segundo, utilizando una herramienta automatizada que extrae el contenido de la caché y registra los valores de TTL asociados a cada dominio.

4.3.5. Visor estático vs análisis dinámico

El visor desarrollado combina distintos modos de análisis según el tipo de información:

- **Análisis estático de estadísticas DNS:** las estadísticas globales del servidor se muestran de forma estática, basadas en archivos named.stats.
- **Análisis dinámico del contenido de la caché:** la lista de dominios y sus TTL se actualiza periódicamente, permitiendo observar la evolución temporal de la caché.
- **Análisis estático de hits y misses por dominio:** los valores de hits y misses se calculan a partir de snapshots y se presentan de forma agregada tras la ejecución del experimento.

Esta combinación permite estudiar tanto el estado puntual como la evolución temporal del comportamiento de la caché DNS.

4.3.6. Estrategia de sincronización entre entornos

La sincronización entre el cubo y la máquina virtual se realiza mediante SSH y scripts temporizados.

El controlador **dnsperf_gui.py** coordina la ejecución de DNSperf con la captura periódica de dumps y estadísticas, garantizando que los snapshots de caché se corresponden temporalmente con la carga DNS generada.

4.4. IMPLEMENTACIÓN

4.4.1. Implementación del flujo de control y captura (cubo y máquina virtual)

El controlador de experimentos implementa una **interfaz gráfica** que permite configurar y ejecutar simulaciones de carga DNS de forma controlada y reproducible.

Dicho controlador incluye las siguientes funcionalidades principales:

- Gestión de configuración persistente.
- Configuración segmentada del experimento, permitiendo definir múltiples **segmentos** o fases consecutivas, cada una con parámetros de carga específicos (por ejemplo, número de consultas por segundo y duración), lo que facilita la simulación de escenarios dinámicos y variaciones progresivas de carga.
- Construcción dinámica del comando **DNSperf** a partir de los parámetros definidos por el usuario.
- Ejecución remota sincronizada mediante **SSH**.
- Validación de parámetros y manejo de errores de conexión o ejecución.

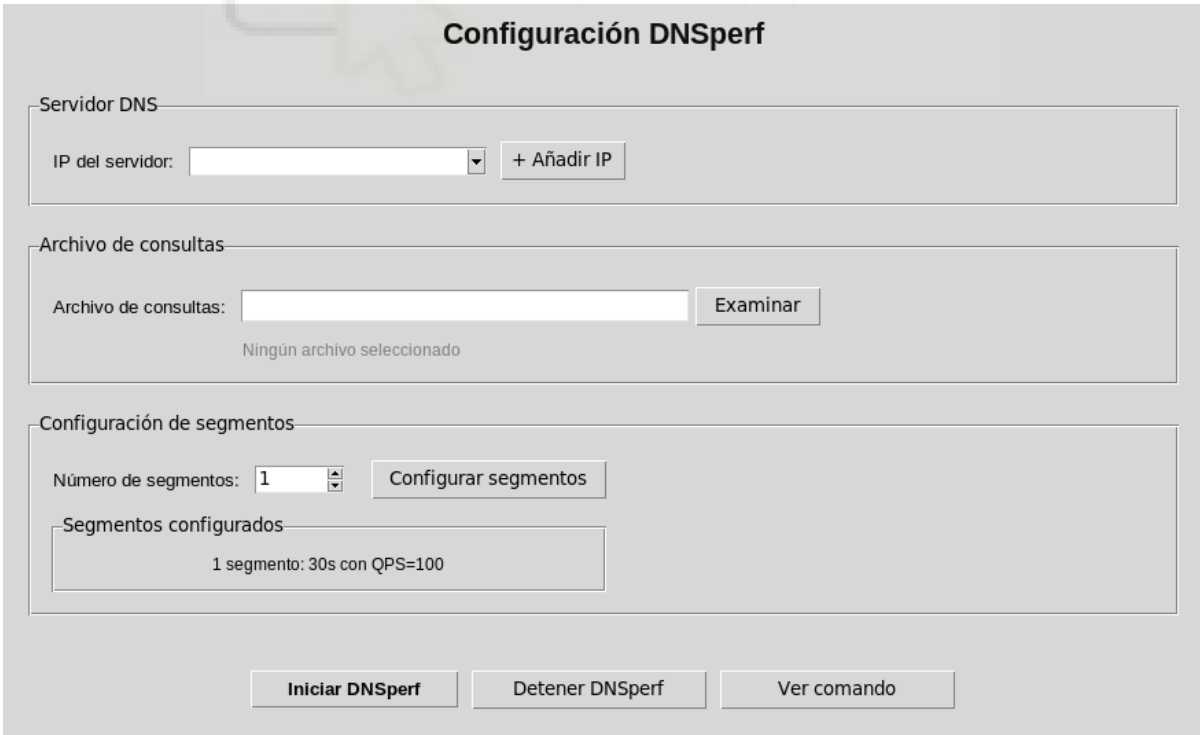


Figura 12: Interfaz de configuración DNSperf

Por otro lado, el script DNSperf_cubo.sh actúa como ejecutor local de comandos DNSperf, facilitando la automatización del experimento y encapsulando la lógica de conexión remota.

El script real_time_export.sh de la máquina virtual implementa la captura periódica de estadísticas y dumps de caché durante la ejecución del experimento. Se encarga de ejecutar comandos rndc stats y rndc dumpdb -cache, y de organizar los archivos generados. A lo largo de la prueba, se procesan los datos con los siguientes programas:

- **extraer_datos_dumpdb.py:** implementa un parser para los archivos named_dump.db, extrayendo dominios y TTLs.
- **extraer_datos_stats.py:** procesa incrementalmente los archivos named.stats y extrae métricas globales del servidor DNS.

4.4.2. Implementación del visor de análisis

El visor se ha implementado en Python con una arquitectura modular:

```
visor_dns/
│
├── utils/
│   ├── __pycache__/
│   ├── __init__.py
│   └── colors.py
│
├── viewer/
│   ├── __pycache__/
│   ├── __init__.py
│   ├── cache_view.py
│   ├── dns_viewer.py
│   ├── hits_misses_view.py
│   ├── stats_view.py
│   ├── tempCodeRunnerFile.py
│   └── ui_setup.py
│
└── main.py
```

Tabla 5: Esquema de archivos del visor

La interfaz se ha desarrollado utilizando tkinter y ttk, proporcionando ventanas, tablas interactivas y controles de filtrado.

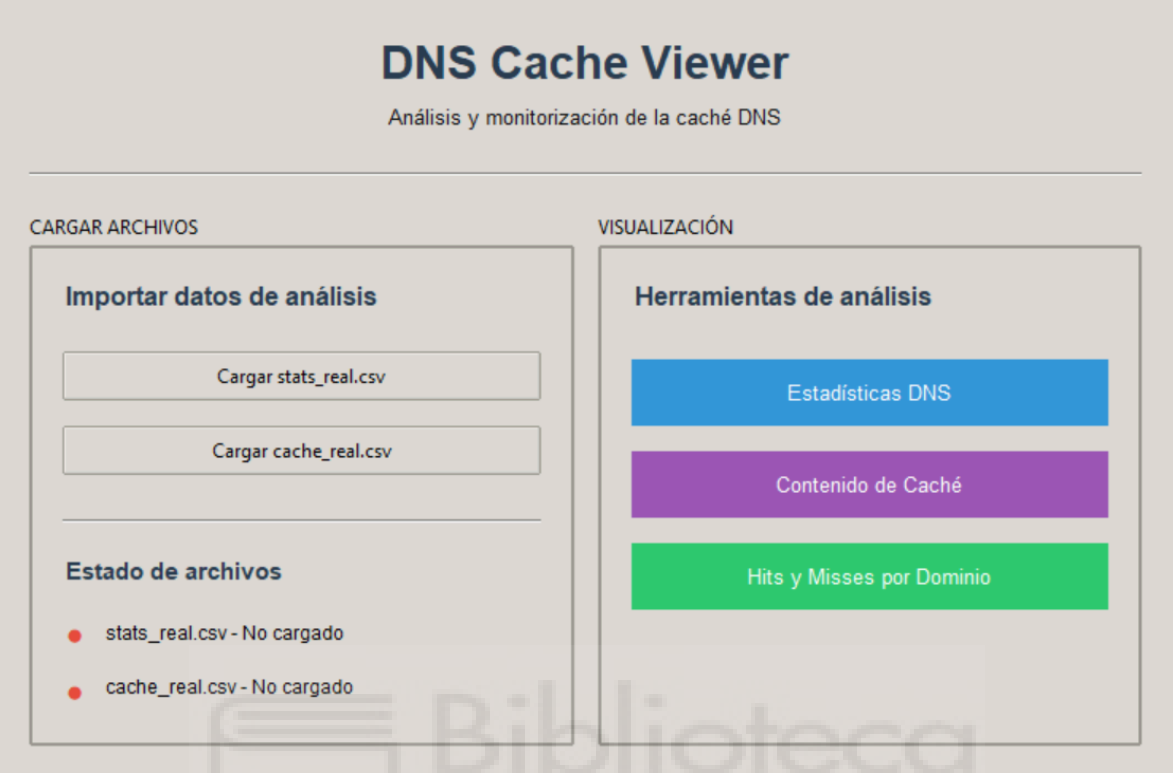
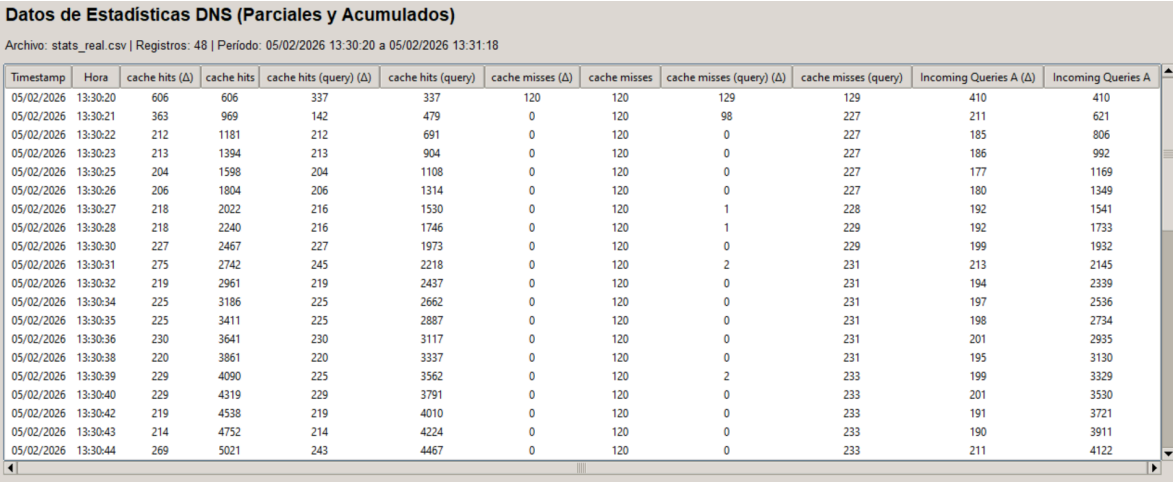


Figura 13: Interfaz del visor DNS Cache Viewer

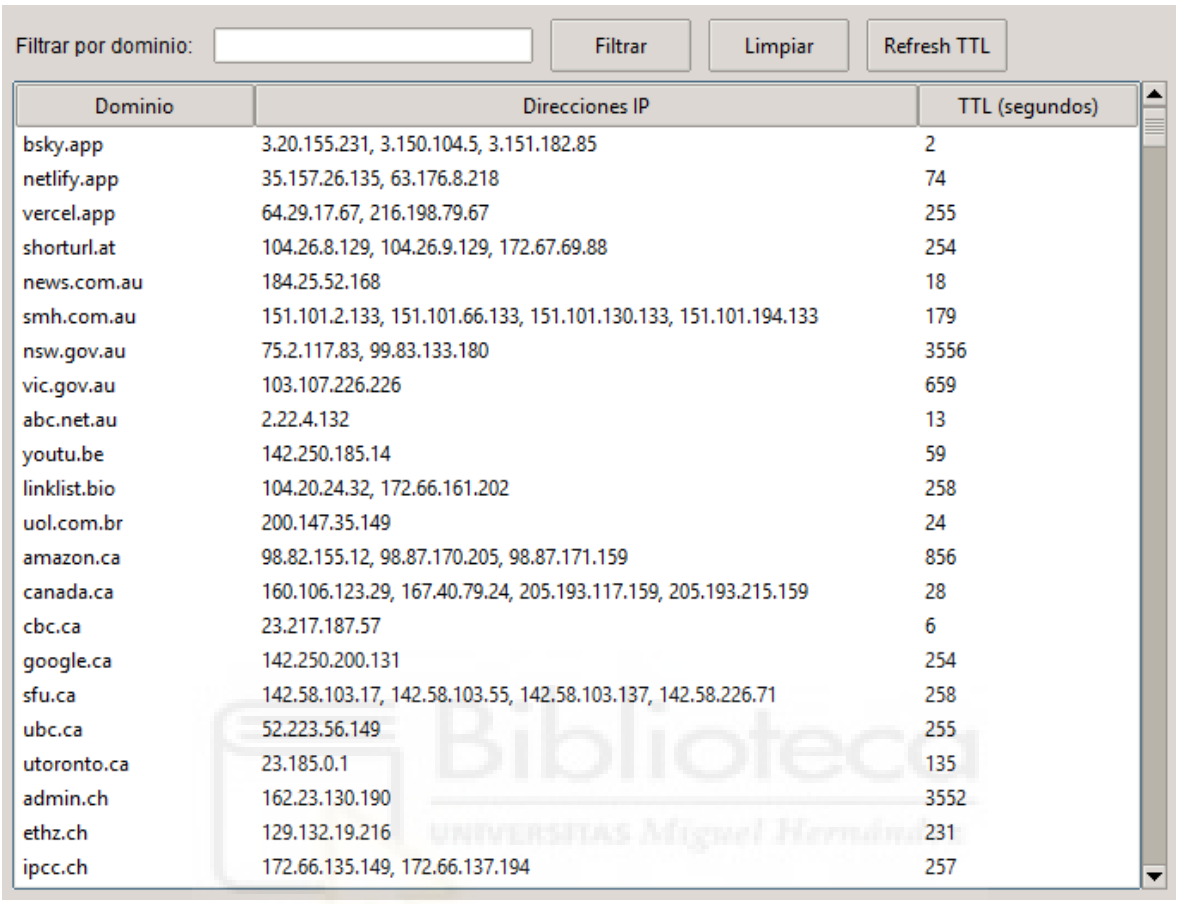
Por un lado, se muestran estadísticas globales extraídas de named.stats, incluyendo número de consultas, hits y misses globales.



Timestamp	Hora	cache hits (Δ)	cache hits	cache hits (query) (Δ)	cache hits (query)	cache misses (Δ)	cache misses	cache misses (query) (Δ)	cache misses (query)	Incoming Queries A (Δ)	Incoming Queries A
05/02/2026	13:30:20	606	606	337	337	120	120	129	129	410	410
05/02/2026	13:30:21	363	969	142	479	0	120	98	227	211	621
05/02/2026	13:30:22	212	1181	212	691	0	120	0	227	185	806
05/02/2026	13:30:23	213	1394	213	904	0	120	0	227	186	992
05/02/2026	13:30:25	204	1598	204	1108	0	120	0	227	177	1169
05/02/2026	13:30:26	206	1804	206	1314	0	120	0	227	180	1349
05/02/2026	13:30:27	218	2022	216	1530	0	120	1	228	192	1541
05/02/2026	13:30:28	218	2240	216	1746	0	120	1	229	192	1733
05/02/2026	13:30:30	227	2467	227	1973	0	120	0	229	199	1932
05/02/2026	13:30:31	275	2742	245	2218	0	120	2	231	213	2145
05/02/2026	13:30:32	219	2961	219	2437	0	120	0	231	194	2339
05/02/2026	13:30:34	225	3186	225	2662	0	120	0	231	197	2536
05/02/2026	13:30:35	225	3411	225	2887	0	120	0	231	198	2734
05/02/2026	13:30:36	230	3641	230	3117	0	120	0	231	201	2935
05/02/2026	13:30:38	220	3861	220	3337	0	120	0	231	195	3130
05/02/2026	13:30:39	229	4090	225	3562	0	120	2	233	199	3329
05/02/2026	13:30:40	229	4319	229	3791	0	120	0	233	201	3530
05/02/2026	13:30:42	219	4538	219	4010	0	120	0	233	191	3721
05/02/2026	13:30:43	214	4752	214	4224	0	120	0	233	190	3911
05/02/2026	13:30:44	269	5021	243	4467	0	120	0	233	211	4122

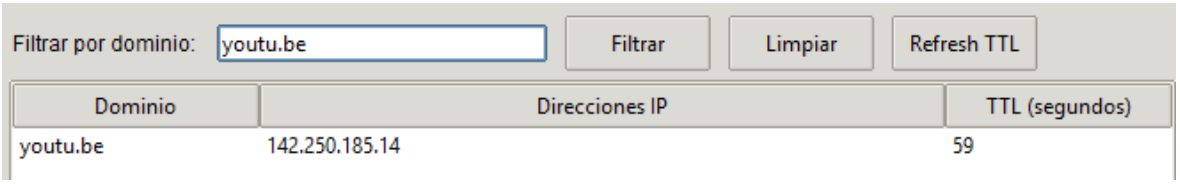
Figura 14: Datos de estadísticas DNS

Por otra parte, se presentan tablas con los dominios en caché y su TTL restante, permitiendo el filtrado.



Dominio	Direcciones IP	TTL (segundos)
bsky.app	3.20.155.231, 3.150.104.5, 3.151.182.85	2
netlify.app	35.157.26.135, 63.176.8.218	74
vercel.app	64.29.17.67, 216.198.79.67	255
shorturl.at	104.26.8.129, 104.26.9.129, 172.67.69.88	254
news.com.au	184.25.52.168	18
smh.com.au	151.101.2.133, 151.101.66.133, 151.101.130.133, 151.101.194.133	179
nsw.gov.au	75.2.117.83, 99.83.133.180	3556
vic.gov.au	103.107.226.226	659
abc.net.au	2.22.4.132	13
youtu.be	142.250.185.14	59
linklist.bio	104.20.24.32, 172.66.161.202	258
uol.com.br	200.147.35.149	24
amazon.ca	98.82.155.12, 98.87.170.205, 98.87.171.159	856
canada.ca	160.106.123.29, 167.40.79.24, 205.193.117.159, 205.193.215.159	28
cbc.ca	23.217.187.57	6
google.ca	142.250.200.131	254
sfu.ca	142.58.103.17, 142.58.103.55, 142.58.103.137, 142.58.226.71	258
ubc.ca	52.223.56.149	255
utoronto.ca	23.185.0.1	135
admin.ch	162.23.130.190	3552
ethz.ch	129.132.19.216	231
ipcc.ch	172.66.135.149, 172.66.137.194	257

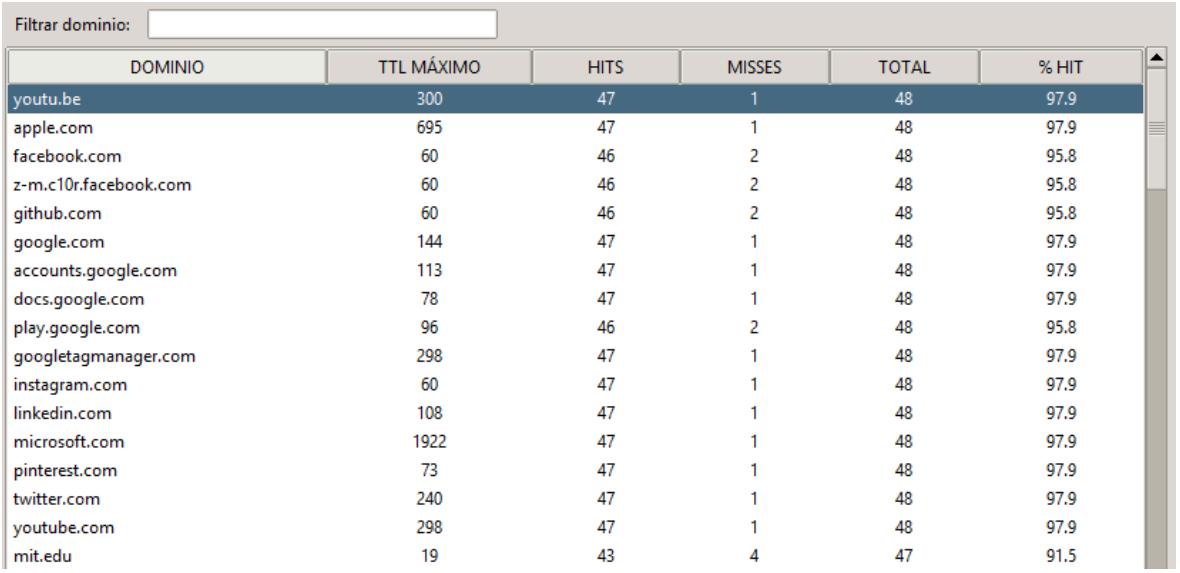
Figura 15: Contenido de caché por dominios



Dominio	Direcciones IP	TTL (segundos)
youtu.be	142.250.185.14	59

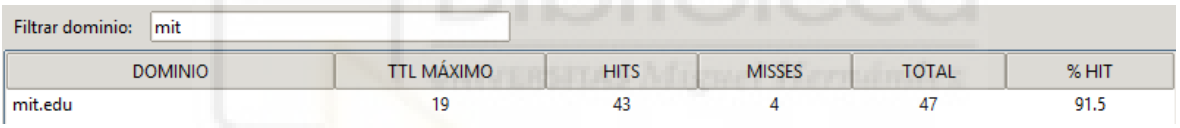
Figura 16: Filtrado de dominio en el contenido de caché del visor

Por último, el visor calcula el número de hits y misses por dominio comparando snapshots consecutivos.



DOMINIO	TTL MÁXIMO	HITS	MISSES	TOTAL	% HIT
youtu.be	300	47	1	48	97.9
apple.com	695	47	1	48	97.9
facebook.com	60	46	2	48	95.8
z-m.c10r.facebook.com	60	46	2	48	95.8
github.com	60	46	2	48	95.8
google.com	144	47	1	48	97.9
accounts.google.com	113	47	1	48	97.9
docs.google.com	78	47	1	48	97.9
play.google.com	96	46	2	48	95.8
googletagmanager.com	298	47	1	48	97.9
instagram.com	60	47	1	48	97.9
linkedin.com	108	47	1	48	97.9
microsoft.com	1922	47	1	48	97.9
pinterest.com	73	47	1	48	97.9
twitter.com	240	47	1	48	97.9
youtube.com	298	47	1	48	97.9
mit.edu	19	43	4	47	91.5

Figura 17: Lista de dominios con estadísticas relevantes de hits y misses



DOMINIO	TTL MÁXIMO	HITS	MISSES	TOTAL	% HIT
mit.edu	19	43	4	47	91.5

Figura 18: Filtrado de dominio dentro del menú de hits y misses por dominio

4.5. PRUEBAS Y RESULTADOS

4.5.1. Estrategia de pruebas

Para evaluar el funcionamiento del visor de caché DNS se ha diseñado una estrategia de pruebas basada en tres ejes:

- **Pruebas funcionales:** se ha verificado que la aplicación capture correctamente los snapshots de caché, procese los datos en formato CSV y calculase hits y misses por dominio.
- **Pruebas de rendimiento:** se ha medido la eficiencia del sistema en términos de latencia de captura, procesamiento de archivos y generación de gráficos bajo diferentes cargas de consultas.
- **Pruebas de validez:** se han comparado las métricas obtenidas por el visor con los contadores internos de BIND9 para validar la coherencia de los resultados.

Con esto, se ha permitido garantizar que el visor cumpliera tanto los requisitos funcionales como los de precisión y eficiencia, y proporcionara información confiable sobre el comportamiento de la caché DNS.

Biblioteca
UNIVERSITAS Miguel Hernández

4.5.2. Resultados experimentales del visor

Se ha realizado un experimento de 45 segundos utilizando un archivo con 500 nombres de dominio (queries500.txt), y con tres segmentos de carga diferenciada:

Segmento	Duración	QPS (consultas por segundo)
1	10 segundos	150
2	15 segundos	200
3	20 segundos	125

Tabla 6: Especificaciones de la prueba DNSperf

Durante la simulación se han generado **gráficos de evolución temporal** de las métricas principales, mostrando tanto **valores parciales**, calculados de forma independiente para cada segmento del experimento, como **valores acumulados**, obtenidos a partir de la suma de todos los valores parciales.

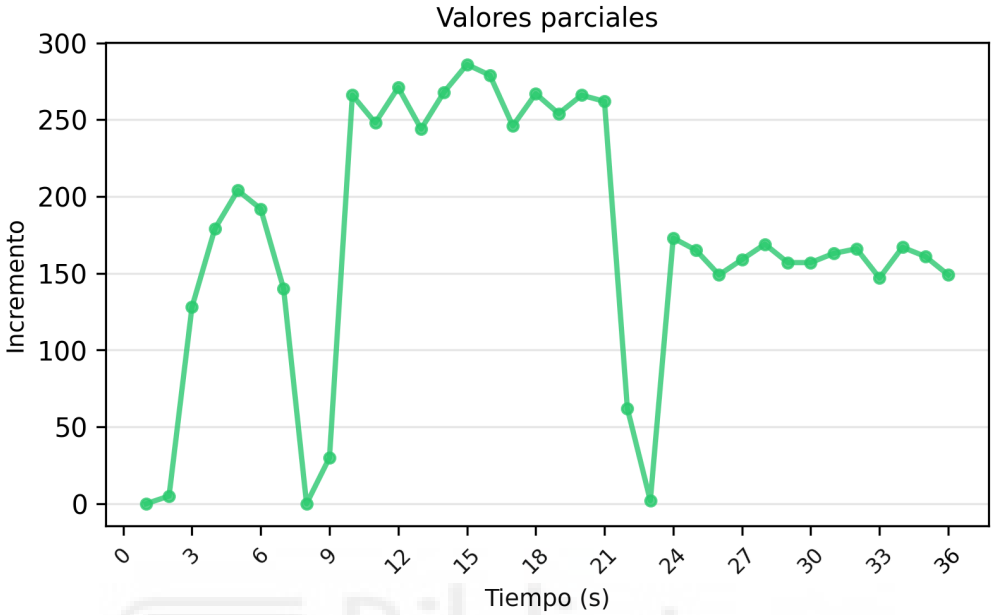


Figura 19: Valores parciales de cache hits (from query)

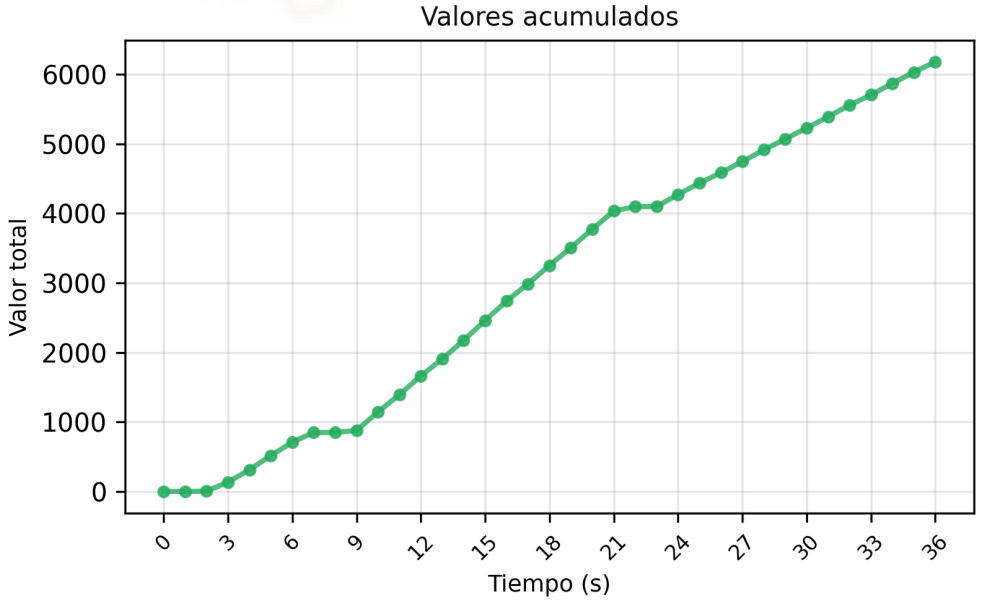


Figura 20: Valores acumulados de cache hits (from query)

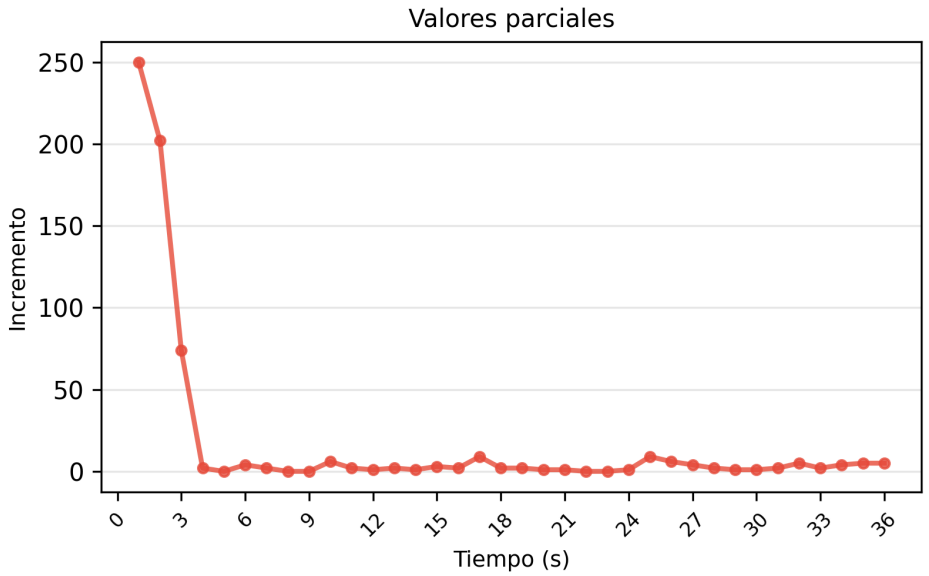


Figura 21: Valores parciales de cache misses (from query)

Figura 22: Valores acumulados de cache misses (from query)

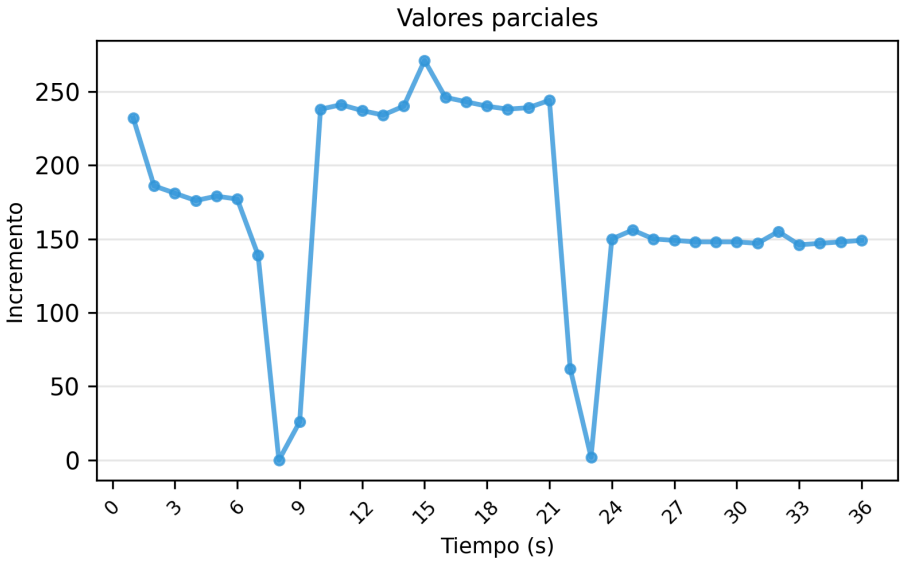


Figura 23: Valores parciales de Incoming Queries A

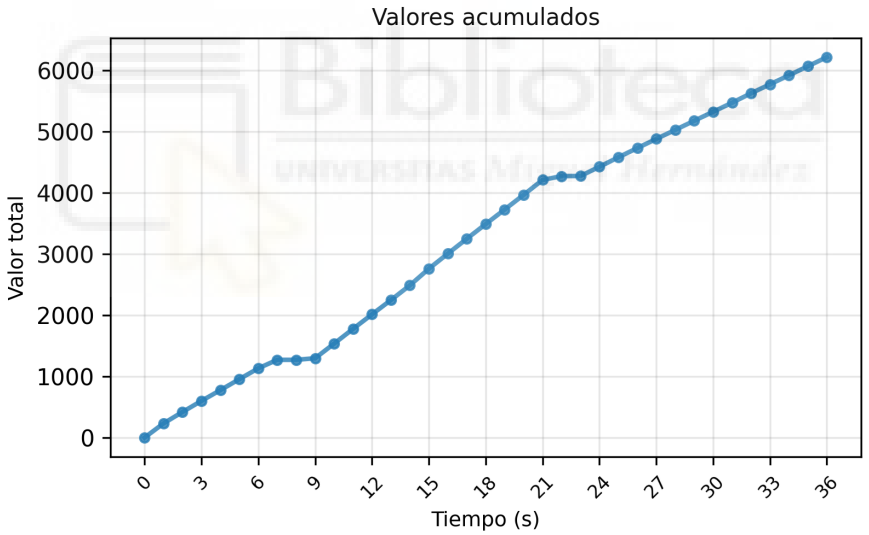


Figura 24: Valores acumulados de Incoming Queries A

Tiempo (s):	0	1	2	3	4	5	6	7	8	9	10	11	12	13
TTL:	18	16	15	14	13	11	10	9	7	6	5	3	3	2

Figura 25: Evolución del TTL de cada dominio a lo largo de la simulación

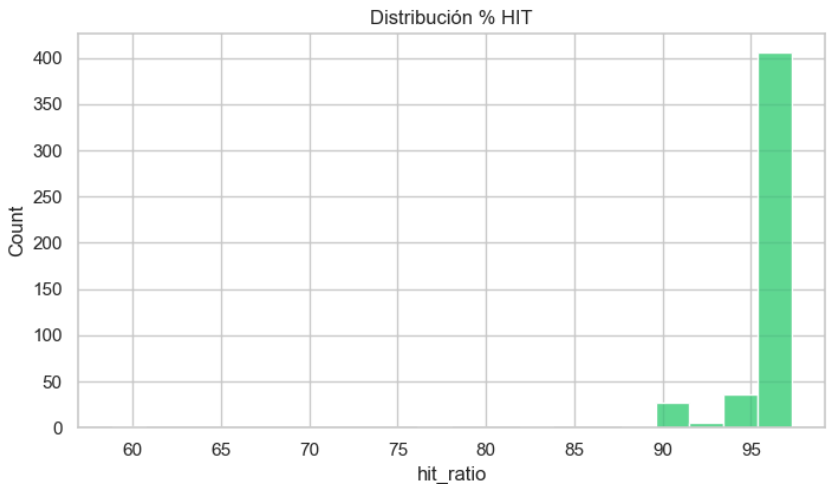


Figura 26: Número de dominios clasificados por su tasa de hits

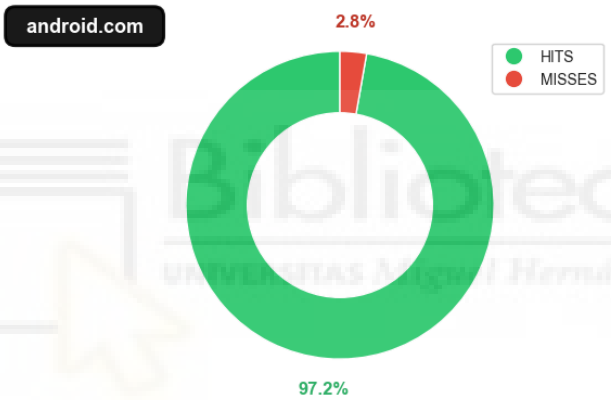


Figura 27: Gráfico circular de tasa de hits y misses de un dominio con TTL = 300

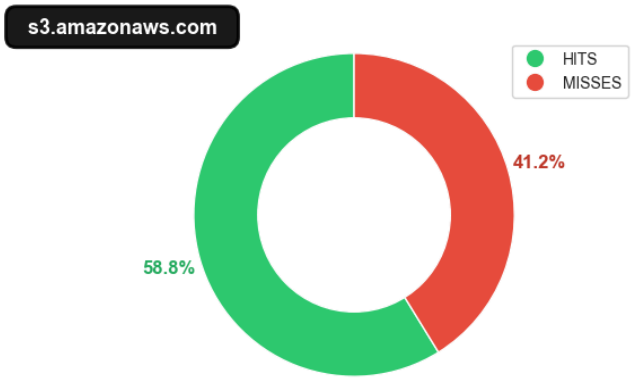


Figura 28: Gráfico circular de tasa de hits y misses de un dominio con TTL = 4

El **análisis de estos resultados** muestra que la mayoría de dominios presentan un alto porcentaje de hits y un porcentaje bajo de misses, lo que indica que la caché está utilizando eficientemente los registros previamente almacenados. Los aumentos temporales de misses coinciden con la aparición de dominios recién consultados o con TTL expirado, lo que confirma el correcto funcionamiento de la lógica de expiración de BIND9.

El comportamiento observado por segmentos de carga indica lo siguiente:

- Durante el segundo segmento, con mayor QPS (200 consultas por segundo), se produce un ligero incremento de misses, evidenciando que la caché empieza a atender consultas de dominios no presentes.
- Dominios con TTL largo mantienen tasas de hit elevadas de manera constante.
- Dominios con TTL corto muestran mayor variabilidad en los hits, lo que refleja la expiración más frecuente de sus registros.

Este análisis permite comprender cómo los parámetros de TTL y la carga de consultas afectan la eficiencia de la caché y proporciona datos concretos para optimizar políticas de almacenamiento en BIND9.

También se ha realizado un **ajuste logarítmico** para modelar la relación entre el TTL y la tasa de hits, siendo la variable independiente el TTL en segundos y la variable dependiente el porcentaje de tasa de hits.

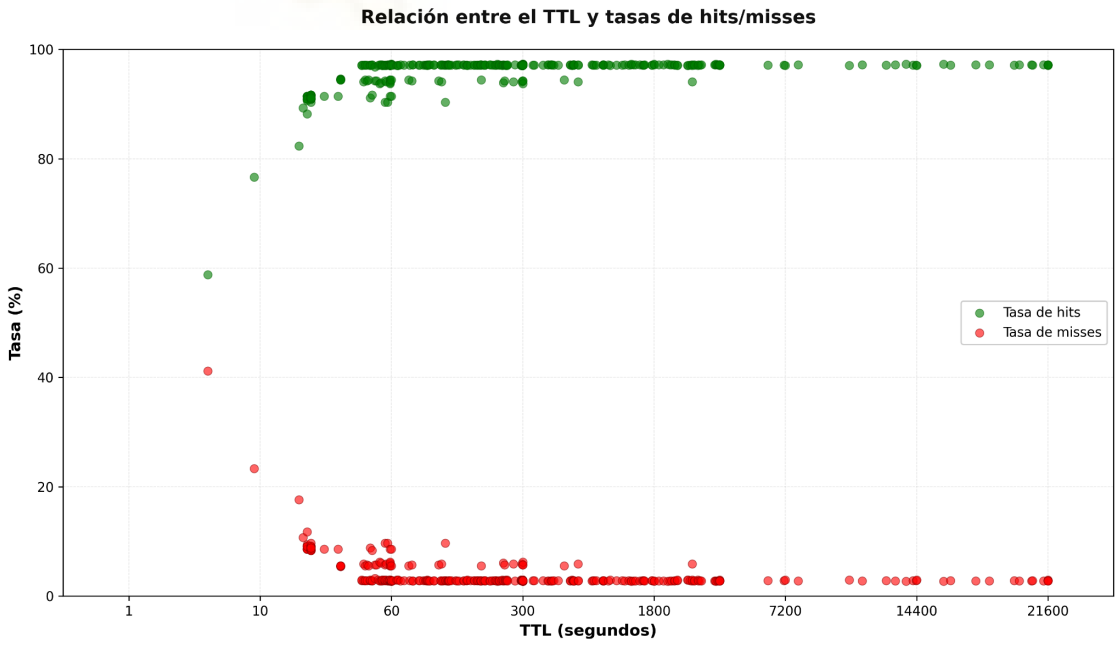


Figura 29: Relación entre el TTL y tasas de hits/misses

Los resultados muestran un incremento rápido de la tasa de hits para TTL bajos, que se estabiliza para TTL más largos, indicando rendimientos marginales decrecientes: aumentar el TTL más allá de un cierto valor no genera mejoras significativas en la probabilidad de hit.

De manera complementaria se ha analizado la tasa de misses, observándose la relación inversa: a medida que los hits aumentan los misses disminuyen, confirmando la consistencia interna del modelo de caché.

4.5.3. Comparación entre métricas internas de BIND9 y métricas del visor

Por otra parte, se han comparado las métricas calculadas por el visor con los contadores internos de BIND9 obtenidos del archivo named.stats. Las métricas principales incluyen:

- **Cache hits:** número total de entradas servidas desde la caché, incluyendo tanto las consultas externas como las generadas internamente por BIND9.
- **Cache hits (from query):** número de hits asociados exclusivamente a consultas DNS reales originadas por clientes externos.
- **Cache misses:** número total de misses producidos, ya sea por expiración del TTL o por ausencia de la entrada en caché, incluyendo consultas internas de BIND9.
- **Cache misses (from query):** número de misses asociados únicamente a consultas reales de clientes.
- **Incoming Queries A:** número total de consultas DNS de tipo A recibidas por el servidor.

Se ha observado que los valores no coinciden exactamente, pero esto se debe a que cada métrica mide un fenómeno diferente:

- **BIND9:** contabiliza eventos globales, incluyendo consultas internas, reintentos y resoluciones parciales.
- **Visor:** calcula hits y misses a nivel de dominio mediante la comparación entre snapshots consecutivos, proporcionando una visión más detallada de eventos significativos para el análisis de la caché.

Por lo tanto, los resultados del visor y de BIND9 son complementarios, ofreciendo dos perspectivas: una global del resolver y otra enfocada al comportamiento por dominio. Esto confirma la validez de los datos generados por el visor y su utilidad para estudios experimentales.

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 5
CONCLUSIONES Y TRABAJO FUTURO

Biblioteca
UNIVERSITAS Miguel Hernández

5.1. CONCLUSIONES

Se ha demostrado que el sistema permite visualizar y analizar de manera efectiva el comportamiento dinámico de la caché DNS en servidores BIND9. Mediante la captura periódica de snapshots y estadísticas del servidor, el visor ha permitido identificar los dominios almacenados en caché, seguir la evolución del TTL y detectar eventos de cache hits y cache misses.

Este enfoque ha facilitado la comprensión de conceptos teóricos como la renovación del TTL, los intervalos entre consultas y el impacto de la carga sobre la eficiencia de la caché, conectando la teoría con resultados experimentales obtenidos en un entorno realista.

5.1.1. Limitaciones encontradas

Durante el desarrollo del proyecto se identificaron diversas **limitaciones técnicas y metodológicas**. En primer lugar, el acceso a los dumps de caché y estadísticas requirió permisos elevados y configuraciones específicas de BIND, lo que obligó a trabajar en entornos controlados y limitó su aplicación directa en producción.

Además, los valores de TTL dependían de los servidores autoritativos externos, restringiendo el control sobre el comportamiento de la caché. Aunque se exploró la modificación del TTL en BIND, esta requería cambios profundos en el código y podía alterar el funcionamiento estándar del protocolo DNS.

El **método basado en snapshots** ha permitido inferir hits y misses sin modificar el núcleo del servidor, pero no proporciona información exacta a nivel de consulta individual, pudiendo introducir pequeñas imprecisiones en escenarios de alta carga. Por último, la generación periódica de dumps producía grandes volúmenes de datos, lo que podía afectar al rendimiento y requirió optimización en escenarios a gran escala.

5.2. POSIBLES DESARROLLOS FUTUROS

Una primera línea de trabajo futuro consistiría en **ampliar el visor** para soportar otros tipos de registros DNS (como AAAA, MX, CNAME, NS o TXT) permitiendo analizar escenarios más realistas y heterogéneos.

Asimismo, el sistema podría **integrarse con herramientas de monitorización y administración de redes**, de manera que el análisis de la caché DNS forme parte de un sistema global de supervisión de infraestructuras.

Otra posible extensión es la **modificación dinámica del comportamiento de la caché**, incluyendo políticas de reemplazo, tamaño de caché o estrategias de expiración, con el objetivo de estudiar su impacto en el rendimiento del servidor DNS.

También se plantea la **modificación dinámica del TTL** de los registros almacenados en caché, lo que permitiría simular políticas personalizadas y analizar el impacto del TTL en el tráfico DNS y la latencia.

Finalmente, como línea avanzada, el visor podría integrarse con **modelos analíticos o técnicas de aprendizaje automático** para predecir el ratio de hits, optimizar automáticamente los parámetros de caché y detectar patrones anómalos en las consultas DNS, conectando este trabajo con áreas de inteligencia artificial y optimización de redes.

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

CAPÍTULO 6
BIBLIOGRAFÍA

Biblioteca
UNIVERSITAS Miguel Hernández

[1] DNS
https://www.kaspersky.es/resource-center/definitions/dns
Consultado en enero de 2026

[2] ¿Qué es DNS? | Cómo funciona
https://www.cloudflare.com/es-es/learning/dns/what-is-dns/
Consultado en enero de 2026

[3] Origins of the Domain Name System
O. M. Bonastre y A. Veà - https://ieeexplore.ieee.org/document/8700196
Consultado en diciembre de 2025

[4] BIND
https://www.isc.org/bind/
Consultado en diciembre de 2025

[5] DNS and BIND, 5th ed.
Liu, C. - O'Reilly Media (2016)
Consultado en enero de 2026

[6] BIND9 Administrator Reference Manual - Internet Systems Consortium (2023)
https://bind9.readthedocs.io/
Consultado en noviembre de 2025

[7] How to get the DNS server statistics?
https://access.redhat.com/solutions/477073
Consultado en diciembre de 2025

[8] ¿Qué es el almacenamiento en caché de DNS?
https://www.akamai.com/es/glossary/what-is-dns-caching
Consultado en enero de 2026

[9] Dig (comando)
https://es.wikipedia.org/wiki/Dig_(comando)
Consultado en noviembre de 2025

[10] Rndc man page
https://manpages.debian.org/jessie/bind9utils/rndc.8.en.html
Consultado en enero de 2026

[11] DNSperf man page
https://linux.die.net/man/1/DNSperf
Consultado en octubre de 2025

[12] ¿Qué es el tiempo de vida (TTL)? | Definición de TTL
https://www.cloudflare.com/es-es/learning/cdn/glossary/time-to-live-ttl/
Consultado en enero de 2026

[13] Cache definition
https://www.techtarget.com/searchstorage/definition/cache
Consultado en diciembre de 2025

[14] Tkinter — Python documentation
https://docs.python.org/es/3/library/tkinter.html
Consultado en diciembre de 2025

[15] Pandas
https://pandas.pydata.org/
Consultado en diciembre de 2025

[16] Matplotlib
https://matplotlib.org/
Consultado en diciembre de 2025

[17] Plotly
https://plotly.com/
Consultado en enero de 2026

[18] Bokeh
https://bokeh.org/
Consultado en enero de 2026

[19] Analysis of DNS Cache Effects on Query Distribution
Zheng Wang - https://pmc.ncbi.nlm.nih.gov/articles/PMC3874940/ (2013)
Consultado en octubre de 2025

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

ANEXOS

Biblioteca
UNIVERSITAS Miguel Hernández

Biblioteca
UNIVERSITAS Miguel Hernández

Este anexo describe el proceso de instalación, compilación y configuración del servidor DNS BIND9, así como la definición de zonas DNS utilizadas en el desarrollo del sistema DNS Cache Viewer. La instalación se ha realizado a partir del código fuente oficial de ISC [https://downloads.isc.org/isc/bind9/9.18.33/].

A.1. COMANDOS PRINCIPALES DE GESTIÓN

Para la administración del servidor DNS se utilizan los siguientes comandos:

```shell
# Obtener privilegios de superusuario
su

# Ejecutar BIND9 en primer plano con nivel de depuración 1
sudo /usr/local/sbin/named -f -d 1 -c /usr/local/etc/named.conf

# Control remoto del servidor mediante RNDC
sudo /usr/local/sbin/rndc <comando>
# Ejemplos: dumpdb, reload, stop, stats, status, flush, reconfig
```

A.2. ARCHIVOS Y DIRECTORIOS RELEVANTES

- `/usr/local/sbin/named` – Demonio del servidor DNS
- `/usr/local/sbin/rndc` – Herramienta de control remoto
- `/usr/local/etc/named.conf` – Configuración principal
- `/usr/local/etc/named.conf.options` – Opciones globales del servidor
- `/usr/local/etc/named.conf.local` – Zonas locales
- `/usr/local/etc/named.conf.default-zones` – Zonas estándar
- `/usr/local/etc/rndc.key` – Clave RNDC
- `/usr/local/etc/named.ca` – Servidores raíz
- `/usr/local/etc/named_dump.db` – Volcado de caché
- `/usr/local/etc/named.stats` – Estadísticas de BIND
- `/var/named/` – Archivos de zona

Archivos de zona configurados:

- `/var/named/db.joanamoros23.home` – Zona directa
- `/var/named/db.192` – Zona inversa

Otros archivos del sistema:

- `/usr/local/etc/bind.keys`
- `/usr/local/etc/managed-keys.bind`
- `/usr/local/etc/named.run`

```
usr/
└── local/
    ├── sbin/
    │   ├── named
    │   └── rndc
    └── etc/
        ├── named.conf
        ├── named.conf.options
        ├── named.conf.local
        ├── named.conf.default-zones
        ├── rndc.key
        ├── named.ca
        ├── named_dump.db
        ├── named.stats
        ├── bind.keys
        ├── managed-keys.bind
        └── named.run

var/
└── named/
    ├── db.joanamoros23.home
    └── db.192
```

Biblioteca UNIVERSITAS Miguel Hernández

A.3. PROCESO DE INSTALACIÓN DE BIND9

A.3.1. Instalación de dependencias

Antes de compilar BIND9 desde el código fuente, se instalaron las dependencias necesarias:

```shell
sudo apt update
sudo apt install build-essential libssl-dev libcap-dev libxml2-dev
libuv1-dev libnghttp2-dev pkg-config make -y
```

A.3.2. Descarga y compilación del código fuente

Se descargó la versión 9.18.33 del repositorio oficial de ISC:

```shell
cd ~
wget https://downloads.isc.org/isc/bind9/9.18.33/bind-9.18.33.tar.xz
tar -xf bind-9.18.33.tar.xz
cd bind-9.18.33
./configure --prefix=/usr/local --sysconfdir=/usr/local/etc
sudo make -j$(nproc)
sudo make install
```

A.3.3. Configuración de bibliotecas compartidas

Para evitar errores de carga de bibliotecas:

```shell
echo "/usr/local/lib" | sudo tee /etc/ld.so.conf.d/bind9.conf
sudo ldconfig
```

A.3.4. Generación de la clave RNDC

```shell
sudo /usr/local/sbin/rndc-confgen -a -c /usr/local/etc/rndc.key
wrote key file "/usr/local/etc/rndc.key"
```

A.3.5. Descarga del archivo de servidores raíz

```shell
sudo curl -o /usr/local/etc/named.ca
https://www.internic.net/domain/named.root
```

Biblioteca
UNIVERSITAS Miguel Hernández

A.4. CONFIGURACIÓN DEL SERVIDOR DNS EN BIND9

A.4.1. Archivo principal (named.conf)

```shell
zone "." IN {
    type hint;
    file "/usr/local/etc/named.ca";
};

include "/usr/local/etc/named.conf.options";
include "/usr/local/etc/named.conf.local";
include "/usr/local/etc/named.conf.default-zones";
```

A.4.2. Opciones globales (named.conf.options)

```shell
options {
    directory "/media/sf_tfg_shared";
    statistics-file "/media/sf_tfg_shared/named_stats/named.stats";
    dump-file "/media/sf_tfg_shared/named_dumps/named_dump.db";
    listen-on { any; };
    allow-query { any; };
    allow-query-cache { any; };
    recursion yes;
    dnssec-validation no;

    forwarders {
        8.8.8.8;
        8.8.4.4;
    };
};
```

Biblioteca UNIVERSITAS Miguel Hernández

A.4.3. Zonas locales (named.conf.local)

```
Shell
zone "joanamoros23.home" {
    type master;
    file "/var/named/db.joanamoros23.home";
};

zone "1.168.192.in-addr.arpa" {
    type master;
    file "/var/named/db.192";
};
```

A.4.4. Zonas estándar (named.conf.default-zones)

```
Shell
zone "localhost" {
    type master;
    file "db.local";
};

zone "127.in-addr.arpa" {
    type master;
    file "db.127";
};

zone "0.in-addr.arpa" {
    type master;
    file "db.0";
};

zone "255.in-addr.arpa" {
    type master;
    file "db.255";
};
```

Biblioteca
UNIVERSITAS Miguel Hernández

A.5. ARCHIVOS DE ZONA DNS

A.5.1. Zona directa (db.joanamoros23.home)

```
Shell
$TTL    86400
@       IN      SOA     joanamoros23.home. root.joanamoros23.home. (
                              3         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                          86400 )       ; Negative Cache TTL
;
@       IN      NS      ns.joanamoros23.home.
ns      IN      A       192.168.1.155   ; Dirección del servidor DNS
(BIND)
@       IN      A       192.168.1.155  ; Dirección del dominio
principal
pc      IN      A       192.168.1.50    ; Dirección del portátil
```

Biblioteca
UNIVERSITAS Miguel Hernández

A.5.2. Zona inversa (db.192)

```
Shell
$TTL    86400
@       IN      SOA     joanamoros23.home. root.joanamoros23.home. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                          86400 )       ; Negative Cache TTL
;
@       IN      NS      ns.joanamoros23.home.
155     IN      PTR     joanamoros23.home.   ; Resolución inversa del
servidor DNS
50      IN      PTR     pc.joanamoros23.home.   ; Resolución inversa
del portátil
```

A.6. OTROS ARCHIVOS DEL SISTEMA

- **/usr/local/etc/bind.keys:** contiene las claves de confianza DNSSEC.
- **/usr/local/etc/rndc.key:** Clave secreta para la autenticación de RNDC.
- **/usr/local/etc/named_dump.db:** Archivo de volcado de la caché DNS.
- **/usr/local/etc/managed-keys.bind:** Claves DNSSEC gestionadas automáticamente.
- **/usr/local/etc/managed-keys.bind.jnl:** Journal de las claves gestionadas.
- **/usr/local/etc/named.run:** Archivo de estado de ejecución del servidor.
- **/usr/local/etc/named.stats:** Estadísticas de funcionamiento del servidor DNS.

A.7. VERIFICACIÓN DE ZONAS

Para validar la sintaxis de los archivos de zona se utilizaron los siguientes comandos:

```
Shell
root@vbox:/var/named# sudo named-checkzone joanamoros23.home
/var/named/db.joanamoros23.home
zone joanamoros23.home/IN: loaded serial 3
OK

root@vbox:/var/named# sudo named-checkzone 1.168.192.in-addr.arpa
/var/named/db.192
zone 1.168.192.in-addr.arpa/IN: loaded serial 2
OK
```