Sistema de intercambio de archivos P2P
P2P File Sharing System

Trabajo de Fin de Máster
Curso 2023–2024

Autor
Sergio García Sánchez

Director
Adrián Riesco Rodríguez

Máster en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

Sistema de intercambio de archivos P2P
P2P File Sharing System

Trabajo de Fin de Máster en Ingeniería Informática
Departamento de XXXXXXXXXXXXX

Autor
Sergio García Sánchez

Director
Adrián Riesco Rodríguez

Convocatoria: *Febrero/Junio/Septiembre 2024*
Calificación: *Nota*

Máster en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

DIA de MES de AÑO

Dedicatoria

A Pedro Pablo y Marco Antonio, por crear TeXiS
e iluminar nuestro camino

Agradecimientos

A Guillermo, por el tiempo empleado en hacer estas plantillas. A Adrián, Enrique y Nacho, por sus comentarios para mejorar lo que hicimos. Y a Narciso, a quien no le ha hecho falta el Anillo Único para coordinarnos a todos.

Resumen

Sistema de intercambio de archivos P2P

Un resumen en castellano de media página, incluyendo el título en castellano. A continuación, se escribirá una lista de no más de 10 palabras clave.

Palabras clave

Máximo 10 palabras clave separadas por comas

Abstract

P2P File Sharing System

An abstract in English, half a page long, including the title in English. Below, a list with no more than 10 keywords.

Keywords

10 keywords max., separated by commas.

Índice

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Contexto

Las redes Peer-to-Peer (P2P) representan un modelo de comunicación descentralizado en el que todos los nodos participan de forma equitativa, actuando tanto como clientes como servidores. Este modelo de comunicación, ofrece una alternativa al modelo cliente-servidor tradicional, transformando la forma en que compartimos información. Una de las áreas donde su impacto es más notable es en el intercambio de archivos, gracias a su capacidad para distribuir grandes cantidades de datos sin depender de servidores centralizados ha supuesto una revolución Schollmeier (2001).

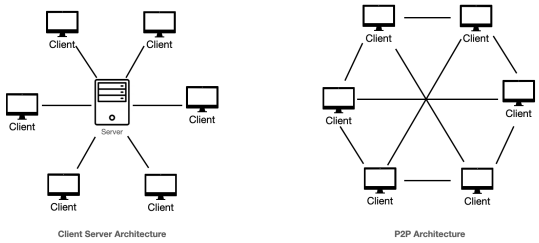

Figura 1.1: Comparación entre arquitectura cliente-servidor y arquitectura P2P

Las redes P2P han tenido un gran impacto en la democratización del acceso a los recursos. Al eliminar la necesidad de servidores centralizados, estas redes permiten a los usuarios compartir directamente recursos como archivos, ancho de banda o recursos computacionales. Esto ha impulsado aplicaciones en dominios como el intercambio de contenido multimedia o la computación distribuida.

Sin embargo, este modelo también plantea algunos retos. Al no tener un punto central de control, se crea la necesidad de desarrollar algoritmos eficientes para la localización de recursos, gestión del tráfico en la red y seguridad en las comunicaciones.

En la actualidad, las redes P2P son fundamentales en aplicaciones que requieren escalabilidad y descentralización. Como servicios de streaming o distribución de videojuegos hasta sistemas basados en blockchain como Bitcoin, estas tecnologías suponen un cambio hacia infraestructuras más descentralizadas.

1.2. Motivación

Las redes P2P han cambiado la manera en que compartimos información, posicionándose como una alternativa eficiente y escalable al modelo cliente-servidor. A pesar de su relevancia y aplicaciones actuales en campos como el intercambio de archivos, el streaming de vídeo y las plataformas blockchain, el diseño e implementación de estas redes desde un nivel práctico sigue siendo un reto para muchos desarrolladores.

Este trabajo no busca desarrollar una red masiva con millones de usuarios, más bien se enfoca en la creación de una red P2P funcional desde cero, centrándose en los conceptos fundamentales y su implementación práctica.

Con ello, se espera facilitar el entendimiento de este modelo descentralizado y motivar el desarrollo de nuevas aplicaciones basadas en redes P2P.

1.3. Objetivos

El objetivo principal de este trabajo es diseñar e implementar una red P2P funcional para el intercambio de archivos de manera distribuida, integrando tecnologías como UPnP para la gestión automática de puertos y un sistema centralizado para la localización y conexión de nodos. Este desarrollo se centrará en la creación de un cliente con interfaz gráfica que sirva como base para todas las funcionalidades de la red.

Para alcanzar este objetivo general, se plantean los siguientes objetivos específicos:

- **Desarrollar un cliente con interfaz gráfica:** Diseñar e implementar una aplicación gráfica que permita a los usuarios compartir y descargar archivos de manera sencilla e intuitiva. Este cliente será el núcleo sobre el cual se integrarán las funcionalidades de la red P2P.

- **Desarrollar un sistema de registro y gestión de nodos:** Desarrollar un sistema centralizado que permita identificar y hacer un seguimiento de los peers conectados a la red, permitiendo su interacción y comunicación.

- **Diseñar e implementar la comunicación directa entre nodos:** Crear un protocolo basado en TCP que permita a los peers establecer la conexión para el intercambio de archivos.

- **Integrar soporte para UPnP:** Automatizar la configuración de puertos en los routers mediante el uso de UPnP, permitiendo una configuración sencilla y accesible para los usuarios.

- **Implementar un sistema de intercambio de archivos distribuido:** Diseñar un modelo que permita a los peers compartir y descargar archivos fragmentados desde múltiples nodos, optimizando el uso de los recursos de la red.

- **Documentar el diseño, implementación y evaluación del sistema:** Elaborar una documentación técnica detallada del desarrollo de la red P2P, incluyendo los algoritmos y tecnologías utilizadas.

1.4. Plan de trabajo

Durante el desarrollo del proyecto se ha seguido un plan de trabajo detallado con el objetivo de estructurar las tareas y garantizar el cumplimiento de los objetivos. La planificación ha permitido organizar de manera eficiente las distintas fases del desarrollo, estableciendo prioridades y tiempos. El plan de trabajo se ha dividido en los siguientes puntos:

1. **Estudio de las tecnologías a utilizar:** Se llevó a cabo un análisis detallado de las tecnologías necesarias para el desarrollo del proyecto, incluyendo protocolos de comunicación como TCP, sistemas de apertura de puertos con UPnP y arquitecturas P2P para el intercambio de archivos. Este estudio incluyó la búsqueda de documentación técnica, recursos en línea y herramientas que facilitaran el desarrollo.

2. **Desarrollo del cliente con interfaz gráfica:** Se inició el diseño e implementación en java del cliente gráfico como base del proyecto. Este cliente incluye funcionalidades esenciales para compartir y descargar archivos, sirviendo como núcleo sobre el cual se desarrollaron las demás características de la red. Durante esta fase, se probaron aspectos básicos de la interacción con el sistema de archivos local.

3. **Implementación del sistema de registro de nodos:** Se diseñó e implementó un sistema centralizado para gestionar los nodos conectados a la red. Este sistema permite registrar los peers activos y facilitar su descubrimiento entre ellos. Además, se integró con el cliente para garantizar que los usuarios pudieran conectarse a otros peers de manera sencilla.

4. **Diseño e implementación de la comunicación entre nodos:** Se desarrolló un protocolo basado en TCP para la conexión directa entre peers. Esta etapa incluyó las pruebas para garantizar la fiabilidad y estabilidad de las conexiones.

5. **Integración del soporte para UPnP:** Se implementó soporte para la configuración automática de puertos mediante UPnP, lo que permitió facilitar la conexión entre nodos en redes con configuraciones complejas, eliminando la necesidad de ajustes manuales por parte del usuario.

6. **Evaluación y pruebas del sistema:** Se llevaron a cabo pruebas funcionales para verificar el funcionamiento de cada componente de la red P2P, estas pruebas incluyeron casos con múltiples nodos conectados para garantizar la robustez del sistema.

7. **Escritura y revisión de la memoria:** Durante las fases de implementación y pruebas, se elaboró la memoria del proyecto, documentando las decisiones tomadas y los resultados obtenidos.

8. **Cierre del proyecto:** Una vez terminadas todas las fases, se realizó una revisión final de la aplicación y de la memoria, realizando las correcciones necesarias para su entrega final. Se verificó que todos los objetivos se hubieran cumplido antes del cierre del proyecto.

En definitiva, un plan de trabajo bien estructurado ha permitido avanzar de manera eficiente y cumplir con los objetivos establecidos dentro del tiempo límite.

1.5. Organización de la memoria

Capítulo 2

Fundamentos téoricos

Este capítulo aborda los fundamentos teóricos necesarios para comprender las redes Peer-to-Peer (P2P), comenzando con su definición y evolución histórica. También, se analizan los tipos de arquitecturas P2P más comunes y los distintos protocolos que permiten su funcionamiento, como TCP y UPnP.

2.1. Introducción a las redes P2P

Las redes P2P son sistemas de comunicación que se caracterizan por su arquitectura descentralizada, donde todos los nodos de la red desempeñan roles equivalentes. Esto significa que los nodos actúan tanto como clientes, solicitando recursos, como servidores, compartiendo datos con otros nodos. Según Schollmeier (2001), una red P2P se define por su capacidad para que los nodos compartan recursos directamente entre ellos, sin depender de una entidad central que coordine o gestione las interacciones. Este enfoque fomenta la resiliencia y la escalabilidad, ya que la red no depende de un único punto de fallo.

En contraste, el modelo cliente-servidor tradicional se basa en un servidor central que almacena los datos y responde a las solicitudes de los clientes. Este modelo tiene algunas limitaciones importantes, como posibles cuellos de botella y la vulnerabilidad a fallos del servidor centralizado Coulouris et al. (2011). Por el contrario, las redes P2P distribuyen la carga de trabajo entre todos los nodos, aprovechando la capacidad colectiva de la red para manejar grandes volúmenes de datos de manera eficiente.

Primera generación de redes P2P El desarrollo moderno de las redes P2P comenzó en 1999 con la aparición de Napster, una plataforma diseñada para compartir música. Napster marcó un hito al permitir que los usuarios intercambiaran archivos directamente, popularizando el concepto de redes P2P Oram (2001). Su arquitectura introdujo el uso de servidores para indexar recursos, una tecnología que facilitaba la búsqueda y transferencia de archivos. Sin embargo, su dependencia de un único punto de control lo convirtió en un objetivo legal vulnerable. El cierre de Napster en 2001 marcó el final de esta primera etapa y el inicio de redes P2P más descentralizadas.

Segunda generación de redes P2P La segunda generación de redes P2P, representada por plataformas como Gnutella y eDonkey, eliminó los servidores centrales y adoptó

modelos completamente distribuidos. En Gnutella, por ejemplo, cada nodo se conectaba directamente con otros nodos vecinos, formando una red en malla. Aunque esta descentralización mejoró la resiliencia de la red, también introdujo grandes retos técnicos. Entre los principales desafíos se encontraban la búsqueda eficiente de recursos en una red distribuida y la gestión del tráfico generado por los distintos nodos Risson y Moors (2006).

Por su parte, eDonkey introdujo mejoras en la forma de localizar archivos, empleando servidores auxiliares para agilizar las búsquedas, pero manteniendo la transferencia directa entre nodos. Estas plataformas demostraron que las redes P2P podían escalar y manejar grandes cantidades de datos, pero también dejaron claro la necesidad de mejorar su funcionamiento.

BitTorrent y la fragmentación de archivos En 2001, BitTorrent marcó un punto de inflexión en la evolución de las redes P2P al introducir un nuevo enfoque para el intercambio de archivos. En lugar de que un nodo descargara un archivo completo de otro, BitTorrent dividía los archivos en fragmentos más pequeños. Esto permitía que un usuario descargara diferentes fragmentos de un archivo desde múltiples nodos simultáneamente, aumentando la eficiencia del ancho de banda disponible Cohen (2003). Además, los nodos comenzaban a compartir los fragmentos descargados con otros usuarios antes de completar la descarga, lo que aumentaba la colaboración entre los nodos.

Esta innovación convirtió a BitTorrent en una tecnología líder en el intercambio de archivos y popularizó el uso de las redes P2P entre los usuarios, demostrando su capacidad para manejar grandes volúmenes de datos de manera eficiente.

Relevancia actual de las redes P2P En la actualidad, las redes P2P van más alla del ámbito del intercambio de archivos y se han adaptado a una amplia gama de aplicaciones. Por ejemplo, en el streaming de contenido, tecnologías como WebRTC aprovechan la conectividad directa entre nodos para mejorar la latencia y reducir la carga en servidores centrales. Además, plataformas basadas en blockchain, como Bitcoin, utilizan redes P2P para garantizar la descentralización no solo en la transferencia de datos, sino también en la gestión financiera. Según Nakamoto (2008), la arquitectura P2P es fundamental para el funcionamiento de Bitcoin, ya que elimina la necesidad de intermediarios y permite un sistema financiero más transparente y seguro.

2.2. Arquitecturas de redes P2P

La arquitectura de una red P2P define cómo los nodos se organizan para compartir recursos y comunicarse entre ellos. Este aspecto estructural tiene un gran impacto en la eficiencia, escalabilidad y resiliencia de la red. Existen tres tipos principales de arquitecturas en redes P2P: centralizadas, descentralizadas e híbridas Schollmeier (2001); Risson y Moors (2006).

2.2.1. Arquitecturas centralizadas

Las arquitecturas centralizadas en redes P2P se caracterizan por la presencia de un servidor central que actúa como coordinador de las interacciones entre los nodos. Este servidor tiene la responsabilidad de gestionar tareas clave como la indexación de los recursos

disponibles, el descubrimiento de nodos y la resolución de las peticiones de los usuarios. Los nodos individuales, por su parte, se conectan al servidor para registrar su disponibilidad y consultar la ubicación de los recursos que desean descargar Schollmeier (2001).

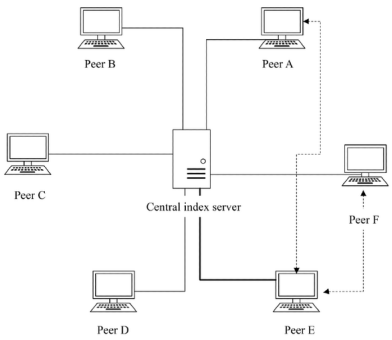

Figura 2.1: Ejemplo de arquitectura P2P centralizada

Un ejemplo representativo de este modelo es Napster, una de las primeras aplicaciones de intercambio de archivos en redes P2P. En Napster, los usuarios podían buscar canciones mediante el servidor central, que les proporcionaba una lista de nodos disponibles que poseían el archivo solicitado. Aunque las transferencias de datos se realizaban directamente entre los nodos, toda la coordinación dependía del servidor central. Este diseño permitió a Napster ofrecer un servicio rápido y eficiente para la localización de archivos Oram (2001).

Sin embargo, la dependencia de un único punto de control introdujo limitaciones significativas en las redes centralizadas. La principal desventaja es la vulnerabilidad ante fallos del servidor, ya que su caída puede interrumpir por completo el funcionamiento de la red. Además, esta arquitectura plantea retos relacionados con la escalabilidad. A medida que el número de nodos y peticiones aumenta, el servidor central puede convertirse en un cuello de botella, afectando negativamente al rendimiento general del sistema.

Otro problema importante de las arquitecturas centralizadas es su exposición a riesgos legales y regulatorios. En el caso de Napster, las demandas por infracción de derechos de autor llevaron al cierre de sus servidores en 2001, lo que dejó a los usuarios sin acceso a la red. Este incidente manifesto la fragilidad de este modelo frente a las presiones externas, fomentando el desarrollo de alternativas más descentralizadas.

A pesar de sus limitaciones, las arquitecturas centralizadas siguen siendo útiles en ciertos escenarios donde la eficiencia y el control son prioritarios. Por ejemplo, en redes pequeñas o sistemas donde la localización rápida de recursos es más importante que la resiliencia, un enfoque centralizado puede resultar ventajoso. En estos casos, el servidor actúa como un mediador confiable, garantizando la consistencia y facilitando la gestión de la red.

2.2.2. Arquitecturas descentralizadas

En las redes P2P descentralizadas, todos los nodos participan con el mismo rol, asumiendo tanto funciones de cliente como de servidor. A diferencia de las arquitecturas centralizadas, estas redes eliminan el servidor único como punto de coordinación, distribuyendo las responsabilidades entre los nodos participantes. Este diseño aumenta significativamente la resiliencia de la red, ya que no depende de un único punto de fallo para su funcionamiento.

Una de las características distintivas de las arquitecturas descentralizadas es su capaci-

dad para formar redes en malla. Cada nodo establece conexiones con otros nodos vecinos, permitiendo la transferencia directa de datos sin intermediarios. Este enfoque mejora la redundancia, ya que incluso si varios nodos fallan, la red puede seguir operando al redirigir las conexiones a través de otros caminos disponibles. Sin embargo, esta descentralización también introduce nuevos problemas, como la necesidad de algoritmos eficientes para la localización de recursos y la gestión del tráfico de red.

Dentro de las arquitecturas descentralizadas, podemos diferenciar dos tipos principales: **estructuradas** y **no estructuradas**, cada una con características, ventajas e inconvenientes.

Redes no estructuradas En las redes no estructuradas, los nodos se conectan de manera arbitraria, sin una organización predefinida en la topología de la red. Estas redes se basan en conexiones dinámicas entre nodos vecinos, lo que permite la transferencia directa de datos sin intermediarios. Este diseño mejora la redundancia, la localización de recursos es más compleja. En estas redes, las búsquedas suelen realizarse mediante técnicas como *flooding* o *broadcast*, donde un nodo envía una solicitud a todos sus vecinos, y estos la retransmiten a los suyos. Aunque estas técnicas son efectivas en redes pequeñas, generan un gran volumen de tráfico en redes más grandes, afectando significativamente el rendimiento.

La simplicidad de las redes no estructuradas las hace ideales para aplicaciones donde los recursos se distribuyen de manera uniforme y la actividad de los nodos es impredecible. Sin embargo, no garantizan que un recurso disponible sea localizado con éxito, especialmente en redes heterogéneas y de gran escala. Las primeras versiones de Gnutella son un ejemplo clásico de esta arquitectura, donde cada nodo actuaba de forma completamente autónoma sin una estructura definida.

Redes estructuradas Por otro lado, las redes estructuradas imponen una organización específica sobre la topología de la red, lo que las hace más eficientes para la localización de recursos. Estas redes utilizan algoritmos distribuidos para asignar identificadores únicos a nodos y recursos, creando un espacio lógico donde la ubicación de cualquier recurso puede ser calculada de manera predecible. Las tablas hash distribuidas (DHTs) son un ejemplo común de este enfoque.

En las redes estructuradas, como Chord, cada nodo y recurso se asigna a una posición dentro de un anillo lógico basado en una función hash. Esto permite realizar búsquedas con una complejidad logarítmica, en lugar de depender de técnicas de *flooding*. Esta organización mejora la escalabilidad y la eficiencia de las redes, haciéndolas ideales para aplicaciones donde se requiere una alta predictibilidad en la localización de datos.

Aunque las redes estructuradas ofrecen claras ventajas en términos de eficiencia, también tienen algunas limitaciones. Su implementación es más compleja y requiere que los nodos sigan estrictamente las reglas del protocolo para mantener la integridad de la red. Además, son más vulnerables a nodos maliciosos que podrían intentar alterar la topología de la red o la asignación de recursos.

Comparación y aplicaciones actuales Tanto las redes no estructuradas como las estructuradas tienen ventajas y desventajas que las hacen adecuadas para diferentes aplicaciones. Las redes no estructuradas son mejores en su flexibilidad y simplicidad, lo que las hace útiles en aplicaciones donde no se requiere una localización precisa de recursos. Por otro lado, las redes estructuradas son preferidas en sistemas donde la eficiencia y

la predictibilidad son críticas, como en aplicaciones de almacenamiento distribuido como BitTorrent.

En la actualidad, muchas arquitecturas descentralizadas han evolucionado hacia modelos híbridos, combinando elementos de ambos enfoques. Por ejemplo, el uso de supernodos en redes no estructuradas mejora la eficiencia al centralizar parcialmente ciertas funciones, mientras que las DHTs han sido integradas en sistemas más flexibles para manejar fallos de nodos o entornos más dinámicos.

2.2.3. Arquitecturas híbridas

Las arquitecturas híbridas combinan elementos de las arquitecturas centralizadas y descentralizadas, logrando un equilibrio entre eficiencia, escalabilidad y resiliencia. Este enfoque es común en redes que requieren una coordinación inicial centralizada para algunas tareas, mientras mantienen la descentralización en la transferencia de datos. Este diseño aprovecha la simplicidad de las centralizadas y la flexibilidad de las descentralizadas.

Las redes híbridas suelen tener un servidor central o un conjunto de nodos especializados que gestionan funciones como la indexación de recursos, la localización de nodos o la autenticación de usuarios. Sin embargo, la transferencia de datos se hace de manera descentralizada, lo que permite que la red continúe funcionando incluso si los nodos centralizados fallan.

BitTorrent es el ejemplo más popular de una arquitectura híbrida que combina características de ambos tipos de redes. En su implementación clásica, utiliza trackers, servidores centralizados que ayudan en la localización inicial de recursos. Estos trackers proporcionan una lista de peers (nodos) que poseen partes del archivo solicitado, facilitando el inicio de las transferencias.

El diseño híbrido de BitTorrent se basa en dos componentes:

- **Tracker centralizado:** Los trackers centralizados actúan como mediadores para localizar recursos. Aunque no participan directamente en la transferencia de datos, son esenciales para la localización de los peers.
- **Transferencia descentralizada:** Una vez que el tracker proporciona la información sobre los peers, la transferencia de datos se realiza directamente entre los nodos, utilizando una estructura completamente descentralizada.

La combinación de estos dos elementos permite a BitTorrent equilibrar eficiencia y escalabilidad. Sin embargo, la dependencia de trackers introduce vulnerabilidades, como puntos únicos de fallo. Para mitigar esto, BitTorrent ha evolucionado hacia modelos más descentralizados mediante el uso de Tablas Hash Distribuidas (DHT), que eliminan la necesidad de trackers centrales, aumentando la robustez de la red.

2.3. Protocolos de comunicación en redes P2P

2.3.1. Protocolos de transporte: TCP y UDP

Los protocolos TCP (Transmission Control Protocol) y UDP (User Datagram Protocol) son los más usados para la comunicación de nodos en redes P2P. Aunque ambos son

esenciales para la transmisión de datos entre nodos, tienen diferencias fundamentales que los hacen más adecuados para algunos tipos aplicaciones específicas. TCP prioriza la fiabilidad y la entrega ordenada de los datos, mientras que UDP, es más rápido y ligero, por lo que es mejor para aplicaciones que necesitan menor latencia. En las siguientes secciones, se analizarán las características de ambos protocolos.

2.3.1.1. Protocolo TCP

El Transmission Control Protocol (TCP) es uno de los pilares fundamentales en las comunicaciones de redes Peer-to-Peer (P2P). Este protocolo de transporte confiable garantiza la entrega íntegra y ordenada de los datos entre dos nodos, lo que lo convierte en una herramienta clave para aplicaciones que priorizan la fiabilidad. TCP opera en un modelo orientado a la conexión, lo que implica que antes de transferir datos, se debe establecer una conexión entre los nodos implicados. Estas características hacen que TCP sea especialmente útil en redes P2P, incluso en entornos con condiciones de red variables.

Establecimiento de conexión: Three-Way Handshake

El proceso de inicialización en TCP, conocido como Three-Way Handshake, permite establecer una conexión confiable entre dos nodos. Este procedimiento consta de tres pasos principales:

1. SYN: El nodo que inicia la conexión envía un paquete de sincronización (SYN) al nodo receptor, indicando su intención de establecer comunicación.

2. SYN-ACK: El receptor responde con un paquete de sincronización y acuse de recibo (SYN-ACK), confirmando la recepción del mensaje inicial y mostrando su disposición para continuar.

3. ACK: Finalmente, el nodo iniciador envía un paquete de acuse de recibo (ACK), completando el proceso y estableciendo la conexión.

Este mecanismo permite a ambos nodos acordar parámetros fundamentales, como los números de secuencia iniciales y las ventanas de transmisión. Estos parámetros son esenciales para garantizar la sincronización durante la transmisión y la integridad de los datos **?**.

Mecanismos de fiabilidad

TCP incluye varios mecanismos diseñados para garantizar que los datos lleguen correctamente al destino, incluso en presencia de fallos en la red:

- Números de secuencia: Cada byte transmitido está asociado a un número de secuencia único, que permite identificar el orden correcto de los datos y detectar duplicados.

- Acuse de recibo (ACK): El receptor confirma la recepción de los datos enviando un mensaje ACK al emisor, indicando el próximo byte esperado. Este sistema asegura que los segmentos recibidos sean correctamente identificados.

- Retransmisión de paquetes: Si el emisor no recibe un acuse de recibo dentro de un tiempo específico calculado dinámicamente (Retransmission Timeout, RTO), el paquete se retransmite **?**.

- Checksum: Cada segmento contiene un valor checksum que valida la integridad de los datos. Si el checksum recibido no coincide, el segmento se descarta **?**.

Estos mecanismos garantizan la fiabilidad y precisión en la transmisión, una característica esencial para aplicaciones de intercambio de archivos en redes P2P.

Control de flujo y congestión

Para gestionar la transferencia de datos de manera eficiente, TCP utiliza algoritmos avanzados de control de flujo y congestión:

- Ventana deslizante (Sliding Window Protocol): Este sistema permite al emisor transmitir múltiples segmentos sin esperar confirmación individual para cada uno, optimizando el uso del ancho de banda disponible.

- Control de congestión: TCP adapta dinámicamente su tasa de transmisión según las condiciones de la red. Los principales algoritmos utilizados incluyen:

  - Slow Start: Inicia con un tamaño de ventana reducido, que aumenta exponencialmente hasta que se detectan signos de congestión.
  - Congestion Avoidance: Tras alcanzar la capacidad de la red, el crecimiento del tamaño de la ventana es lineal.
  - Fast Retransmit: Cuando se detectan tres ACK duplicados consecutivos, TCP asume que un paquete se perdió y lo retransmite inmediatamente, sin esperar al tiempo de retransmisión (RTO) **?**.

Estos mecanismos permiten que TCP maximice el rendimiento, evitando la saturación de la red y garantizando una transferencia fluida.

Desafíos de TCP en redes P2P

A pesar de sus ventajas, TCP enfrenta ciertos desafíos en el contexto de redes P2P:

1. Overhead en la gestión de conexiones: Cada conexión TCP requiere recursos significativos, como memoria y capacidad de procesamiento. En redes P2P, donde los nodos deben establecer y mantener múltiples conexiones simultáneamente, este overhead puede limitar el rendimiento Cohen (2003).

2. Latencia: Los mecanismos de retransmisión y control de flujo pueden introducir retrasos, especialmente en redes congestionadas o de alta latencia.

3. Escalabilidad: La necesidad de gestionar múltiples conexiones persistentes puede convertirse en un cuello de botella en sistemas con miles de nodos.

Relevancia de TCP en redes P2P

A pesar de estos desafíos, TCP sigue siendo el protocolo dominante para aplicaciones P2P que priorizan la fiabilidad y la integridad de los datos. Sistemas como BitTorrent utilizan TCP para transferir fragmentos de archivos entre nodos, asegurando que cada parte se reciba de manera correcta antes de ensamblar el archivo completo. Su capacidad para operar en redes heterogéneas y manejar errores lo convierte en una herramienta esencial en el diseño de redes P2P modernas Cohen (2003).

2.3.1.2. Protocolo UDP

2.3.2. Configuración y gestión de conectividad en redes P2P

2.3.2.1. Problemas de NAT y firewalls

2.3.2.2. Configuración automática mediante UPnP

2.3.2.3. STUN y TURN como soluciones para NAT traversal

Capítulo 3

Diseno y desarrollo de la aplicación

3.1. Arquitectura general

3.2. Desarrollo del cliente

3.2.1. Interfaz gráfica

3.2.2. Configuración automática mediante UPnP

3.2.3. Protocolo de intercambio de mensajes

3.3. Implementación del tracker

3.3.1. Base de datos y persistencia

3.3.2. API de comunicación

3.4. Pruebas y evaluación

3.4.1. Casos de prueba realizados

3.4.2. Análisis de resultados

Capítulo 4

Conclusiones y Trabajo Futuro

Conclusiones del trabajo y líneas de trabajo futuro.

Antes de la entrega de actas de cada convocatoria, en el plazo que se indica en el calendario de los trabajos de fin de máster, el estudiante entregará en el Campus Virtual la versión final de la memoria en PDF. En la portada de la misma deberán figurar, como se ha señalado anteriormente, la convocatoria y la calificación obtenida. Asimismo, el estudiante también entregará todo el material que tenga concedido en préstamo a lo largo del curso.

Chapter 5

Introduction

Introduction to the subject area. This chapter contains the translation of Chapter 1.

Chapter 6

Conclusions and Future Work

Conclusions and future lines of work. This chapter contains the translation of Chapter 4.

Bibliografía

Cohen, B. Incentives build robustness in bittorrent. 2003. Workshop on Economics of Peer-to-Peer Systems.

Coulouris, G., Dollimore, J. y Kindberg, T. *Distributed Systems: Concepts and Design*. Addison-Wesley, 2011.

Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. 2008.

Oram, A. *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly Media, 2001.

Risson, J. y Moors, T. Survey of research towards robust peer-to-peer networks. *Computer Networks*, 2006.

Schollmeier, R. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. En *Proceedings First International Conference on Peer-to-Peer Computing*, páginas 101–102. 2001.

Apéndice A

Título del Apéndice A

Contenido del apéndice

Apéndice B

Título del Apéndice B