REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ DES MATHEMATIQUES ET D'INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité : Réseaux et Télécommunication

Par :

BOULENOUAR MOHAMED ALI et BOUZIANE RANIA LILIA

Sur le thème

LES MECANISMES DE CACHE DANS LES RESEAUX NDN : ETUDE ET SIMULATION

Soutenu publiquement le 12 / 06 / 2024 à Tiaret devant le jury composé de :

Mr BENGHANI ABDELMALEK	MCA	Université Tiaret	Président
Mr NASSANE SAMIR	MAA	Université Tiaret	Encadrant
Mr MOSTEFAOUI Sid Ahmed Mokhtar	MCA	Université Tiaret	Co- Encadrant
Mme ZENATI ANISSA	MAA	Université Tiaret	Examinatrice

2023-2024

قال تعالى (فَتَعَالَى اللَّهُ الْمَلِكُ الْحَقُّ وَلَا تَعْجَلْ بِالْقُرْآنِ مِنْ قَبْلِ أَنْ يُقْضَى إِلَيْكَ وَحْيُهُ وَقُلْ رَبِّ زِدْنِي عِلْمًا) (طه: 114)، وقال تعالى (يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ)(المجادلة: 11)

(اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ*خَلَقَ الْإِنْسَانَ مِنْ عَلَقٍ*اقْرَأْ وَرَبُّكَ الْأَكْرَمُ*الَّذِي عَلَّمَ بِالْقَلَمِ*عَلَّمَ الْإِنْسَانَ مَا لَمْ يَعْلَمْ). سورة العلق، آية: 5-1

Dédicace

À mes chers parents, Djamila et Noureddine, vos encouragements et votre soutien indéfectible ont été le phare guidant mes pas tout au long de ce parcours. À ma seconde mère Fatiha, ta présence et tes conseils ont été une source de réconfort inestimable. À mon frère Riadh, ma sœur Zoulikha et ma sœur Ndajat, votre amour fraternel a été un baume apaisant dans les moments difficiles. À mon ami Taibi Med, ton aide morale a été un véritable moteur pour aller de l'avant. À mon ami d'enfance Garbi Mohamed, ton soutien sans faille m'a donné la force de poursuivre mes objectifs. Et à tous mes amis qui m'ont souhaité bonne chance, votre confiance en moi a été un puissant catalyseur de réussite. Votre présence dans ma vie est une bénédiction, et chacun de vous occupe une place particulière dans mon cœur.

Mohamed ali

Dédicace

Avec l'aide du tout puissant, nous avons pu finir ce présent mémoire, lequel je dédie à ma chère maman Amel et mon cher papa Mohammed à qui je dois mon éducation, ma conduite et mon bonheur, sans oublier ma grand-mère paternelle "Ma Aicha" et mon grand-père maternelle "boyelhadj" et ma grand-mère maternelle " Mima" ainsi qu'à mes tantes.

Je dédie aussi à :

-Mes frères : "Ayoub et Haroun "

- Ma sœur "Douaa "

-Mes cousines

Et à toute ma famille et mes chères amies.

Ranouche

Remerciments

Au terme de ce modeste travail, nous tenons à exprimer notre profonde gratitude. En premier lieu, nous remercions Allah qui nous a donné la force d'achever cette réalisation. Nous souhaitons également remercier toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce mémoire.

Nous adressons nos sincères expressions de sympathie et de gratitude à notre encadreur, Monsieur Nassan Samir, pour sa disponibilité et son aide précieuse. Il nous a accordé sa confiance en nous confiant ce sujet de mémoire. Nous n'oublions pas Monsieur Mestefaoui Sid Ahmed Mokhtar, notre co-encadreur, pour ses conseils motivants et son soutien constant.

Nous tenons également à remercier les membres du jury, Monsieur Benghani Abdelmalek et Madame Zenati Anissa, pour leur temps et leur expertise.

Table Des Matières

Liste des Tablau

Liste des Figures

Liste des Abréviations

NDN : Named Data Networking / Réseaux Nommes de Données

IP : Internet Protocol / Protocole Internet

QoS : Quality of Service / Qualité de Service

ICN : Information-Centric Networking / Réseaux Centrés sur l'Information

CCN : Content-Centric Networking / Réseaux Centrés sur le Contenu

PIT : Pending Interest Table / Table des Intérêts en Attente

FIB : Forwarding Information Base / Base d'Informations de Routage

CS : Content Store / Magasin de Contenu

LRU : Least Recently Used / Moins Récemment Utilisé

LFU : Least Frequently Used / Moins Fréquemment Utilisé

LRFU : Least Recently/Frequently Used / Moins Récemment/Fréquemment Utilisé

LFU-Aging : Least Frequently Used - Aging / Moins Fréquemment Utilisé - Vieillissement

LCE : Leave Copy Everywhere / Laisser une Copie Partout

LCD : Leave Copy Down / Laisser une Copie en Bas

MCD : Move Copy Down / Déplacer la Copie en Bas

ARC : Adaptive Replacement Cache / Cache de Remplacement Adaptatif

SLRU : Segmented Least Recently Used / Segmenté Moins Récemment Utilisé

WLRFU : Window Least Recently/Frequently Used / Fenêtre Moins Récemment/Fréquemment Utilisé

TCP : Transmission Control Protocol / Protocole de Contrôle de Transmission

BGP : Border Gateway Protocol / Protocole de Passerelle de Bordure

IPv4 : Internet Protocol version 4 / Protocole Internet version 4

IPv6 : Internet Protocol version 6 / Protocole Internet version 6

GDSF : Greedy Dual-Size Frequency / Fréquence de Taille Double Gourmande

GD-Size : Greedy Dual-Size / Taille Double Gourmande

WGDSF : Weighted Greedy Dual-Size Frequency / Fréquence de Taille Double Gourmande Pondérée

RESUME

Le Named Data Networking (NDN) est une architecture de communication réseau qui cherche à transformer la manière dont les informations sont échangées sur Internet. Contrairement à l'architecture actuelle fondée sur le protocole IP (Internet Protocol), qui se concentre sur le routage des paquets de données en fonction des adresses IP, NDN se focalise sur le routage de l'information en se basant sur son nom.

Ce mémoire, requis pour l'obtention du diplôme de master en informatique, représente une analyse approfondie des aspects fondamentaux d'un réseau NDN (Named Data Networking). Il explore en détail des concepts tels que l'architecture du réseau, le routage, la sécurité, et la mise en cache de contenu. L'objectif principal est d'évaluer et de mettre en évidence les avantages potentiels de cette architecture par rapport au modèle IP traditionnel, lequel présente des limitations. Des simulations sont également conduites pour évaluer les performances, en mettant l'accent sur des aspects tels que le délai de transmission et la réussite du cache. En se concentrant sur le thème "Les mécanismes de cache dans les réseaux NDN", cette étude apporte une contribution significative à la compréhension du sujet tout en ouvrant de nouvelles perspectives pour l'Internet du futur.

Mots-clés :

Named Data Networking (NDN), Mise en cache de contenu, réussite du cache.

SUMMARY

Named Data Networking (NDN) is a network communications architecture that seeks to transform the way information is exchanged over the Internet. Unlike the current Internet Protocol (IP)-based architecture, which focuses on routing data packets based on IP addresses, NDN focuses on routing information based on its name.

This dissertation, required for the Master's degree in Computer Science, represents an in-depth analysis of the fundamental aspects of an NDN (Named Data Networking) network. It explores in detail concepts such as network architecture, routing, security and content caching. The main objective is to evaluate and highlight the potential advantages of this architecture over the traditional IP model, which has its limitations. Simulations are also conducted to evaluate performance, focusing on aspects such as transmission delay and cache hit. By focusing on the theme "Caching mechanisms in NDN networks", this study makes a significant contribution to the understanding of the subject while opening up new perspectives for the Internet of the future.

Keywords:

Named Data Networking (NDN), Content caching, cache success.

ملخص

هي بنية اتصالات شبكية تسعى إلى تغيير طريقة تبادل المعلومات عبر الإنترنت. على عكس (NDN) شبكة البيانات المسماة ، والتي تركز على توجيه حزم البيانات بناءً على عناوين بروتوكول (IP) البنية الحالية القائمة على بروتوكول الإنترنت الإنترنت، تركز شبكة البيانات المسماة على توجيه المعلومات بناءً على اسمها

NDN تمثل هذه الأطروحة المطلوبة لنيل درجة الماجستير في علوم الحاسب الآلي تحليلاً متعمقًا للجوانب الأساسية لشبكة (شبكات البيانات المسماة). وهي تستكشف بالتفصيل مفاهيم مثل بنية الشبكة والتوجيه والأمان والتخزين المؤقت للمحتوى. والهدف الرئيسي هو تقييم وتسليط الضوء على المزايا المحتملة لهذه البنية وإبرازها مقارنةً بنموذج بروتوكول الإنترنت التقليدي الذي له حدوده. كما يتم إجراء عمليات محاكاة لتقييم الأداء، مع التركيز على جوانب مثل تأخير الإرسال وضرب ، تقدم هذه الدراسة "NDN ذاكرة التخزين المؤقت. من خلال التركيز على موضوع "آليات التخزين المؤقت في شبكات مساهمة كبيرة في فهم الموضوع مع فتح آفاق جديدة لإنترنت المستقبل

الكلمات المفتاحية

، التخزين المؤقت للمحتوى، نجاح التخزين المؤقت(NDN) شبكات البيانات المسماة

Introduction Générale

L'Internet actuel est principalement basé sur le protocole IP qui a été conçu dans les années 1970 par un groupe de chercheurs universitaires pour interconnecter leurs réseaux locaux, principalement pour des applications telles que le transfert de fichiers et le courrier électronique [1]. Ce réseau, initialement conçu comme un simple moyen de transmission d'informations, s'est métamorphosé en une plateforme essentielle à la vie moderne, qui a radicalement transformé notre façon de vivre et de communiquer. Des services en ligne tels que les réseaux sociaux, la diffusion vidéo en continu, la banque en ligne et le commerce électronique ont émergé, créant ainsi de nouveaux impératifs pour les utilisateurs. Ces impératifs incluent des exigences de bande passante élevée, une latence minimale, une sécurité renforcée, la protection de la vie privée, la mobilité, et l'accessibilité, pour n'en nommer que quelques-uns. Cependant, l'architecture IP actuelle, qui repose sur l'identification des hôtes par des adresses IP, présente des limites qui la rendent inadaptée aux besoins de demain, telles que l'inefficacité pour la diffusion de contenu, la difficulté de répondre à la croissance du trafic Internet et la vulnérabilité aux pannes [1]. Pour y répondre, une nouvelle approche a été proposée : l'ICN, ou Réseau Centré sur l'Information.

L'ICN [2] est une nouvelle architecture réseau qui vise à répondre à ces limites. L'idée de l'ICN est de placer l'accent sur le contenu, plutôt que sur les hôtes. Dans l'ICN, le contenu est identifié par des noms, qui sont uniques et indépendants de son emplacement ou de son serveur d'hébergement. Lorsque vous souhaitez accéder à un contenu sur un réseau ICN, votre ordinateur envoie une requête à un service de noms ICN, qui renvoie le contenu à votre ordinateur.

L'ICN permet de transmettre le contenu une seule fois à tous les hôtes qui en font la demande. Par exemple, si vous diffusez une vidéo en streaming, l'ICN peut transmettre la vidéo une seule fois à tous les hôtes qui la regardent, ce qui permet de réduire considérablement le trafic réseau et optimiser davantage la bande passante utilisée. De plus, elle est plus résiliente aux pannes, car elle n'est pas centralisée. Si un routeur tombe en panne, l'ICN peut continuer à fonctionner en utilisant d'autres routeurs [3].

Il existe plusieurs architectures ICN différentes qui ont été proposées. Ces architectures diffèrent en termes de mécanismes de routage, de mécanismes de cache et de mécanismes de sécurité. Parmi les architectures ICN les plus connues, on peut citer: Content-Centric Networking (CCN) [4], Named Data Networking (NDN) [5], Data Oriented Network Architecture (DONA) [6] et Network of Information (NetInf) [7].

Dans ce travail nous nous intéressons au réseau NDN, en mettant particulièrement l'accent sur ses mécanismes de mise en cache. Nous allons faire une étude et une simulation de ces mécanismes pour mieux comprendre les avantages offerts par le réseau NDN. Nous débutons par une introduction générale aux réseaux NDN, décrivant leurs caractéristiques, leurs avantages ainsi que l'architecture IP et ses limitations. Ensuite, nous passons à l'examen des différents mécanismes de cache dans les réseaux NDN, en détaillant les avantages et les inconvénients de chacun. Par la suite, nous abordons l'implémentation et la simulation de ces mécanismes, suivies

d'une analyse et d'une évaluation des résultats obtenus. Enfin, nous concluons notre travail par une synthèse générale.

Chapitre 1
Architecture NDN

Chapitre 1 : Architecture NDN

Introduction

L'Internet repose sur le protocole IP (Internet Protocol) qui permet à divers ordinateurs et appareils de se connecter entre eux grâce à des adresses IP distinctes. Les données sont envoyées sous forme de paquets qui sont acheminés par des routeurs intermédiaires jusqu'à leur destinataire.

Toutefois, face à l'augmentation exponentielle des données et aux changements des besoins des utilisateurs, plusieurs problèmes se posent, notamment la sécurité et la mobilité, etc. Ce qui rend le protocole IP actuel inadapté pour satisfaire les demandes actuelles et futures de l'Internet. C'est pourquoi de nouvelles technologies comme ICN dont NDN fait partie sont élaborées.

Named Data Networking (NDN) est un nouveau modèle de communication réseau qui utilise un nom de contenu pour identifier et acheminer les données au lieu de l'adresse IP. L'idée est de substituer la communication basée sur l'adresse IP par une communication basée sur le contenu, où chaque paquet de données est un objet doté d'un nom unique [3].

Dans ce chapitre, nous allons présenter les limites de l'architecture de l'internet actuel et ses défis. Nous fournirons une brève introduction à l'architecture Content Centric Networking (CCN), avant de nous concentrer sur l'architecture NDN.

1. Architecture IP

Internet [8] est un réseau mondial de réseaux qui permet aux ordinateurs de communiquer entre eux. Il est basé sur un ensemble de protocoles, dont le protocole Internet (IP), qui est responsable de l'adressage et du routage des données. Chaque ordinateur connecté à Internet dispose d'une adresse IP, qui est un numéro unique de 32 bits (IPv4) ou 128 bits (IPv6). L'adresse IP permet à un ordinateur de savoir comment envoyer des données à un autre ordinateur.

Les adresses IP sont divisées en deux parties :

- La partie réseau, qui identifie le réseau auquel l'ordinateur appartient.
- La partie hôte, qui identifie l'ordinateur sur le réseau.

Par exemple, l'adresse IP 192.168.1.100 a la partie réseau 192.168.1 et la partie hôte 100.

Lorsque des données sont envoyées d'un ordinateur à un autre, elles sont divisées en petits paquets de données. Chaque paquet IP contient l'adresse IP de l'ordinateur expéditeur et l'adresse IP de l'ordinateur destinataire. Les paquets IP sont ensuite acheminés d'un ordinateur à l'autre via un réseau de routeurs. Les routeurs utilisent les adresses IP pour déterminer l'itinéraire le plus court pour acheminer les paquets vers leur destination.

2. Les limites de l'architecture actuelle d'internet :

L'architecture actuelle d'Internet présente plusieurs limitations, et de nombreux chercheurs et experts ont discuté de ces défis dans des articles scientifiques. Voici quelques-unes des limites couramment évoquées :

2.1 Sécurité et Confidentialité : [9]

- L'architecture actuelle d'Internet est vulnérable aux attaques et aux intrusions, ce qui met en danger la sécurité des données et la confidentialité des utilisateurs.
- Le manque de mécanismes de sécurité intégrés au niveau du routage et de la transmission des données rend difficile la protection contre les cyberattaques et les cybercrimes.
- La centralisation de l'information et des services sur de grandes plateformes pose des risques de surveillance et de contrôle excessif.

2.2 Evolutivité du Routage :

Le système de routage actuel, basé sur le protocole BGP, n'est pas conçu pour gérer l'explosion du nombre d'appareils et de connexions qui se connectent à Internet. La croissance exponentielle du trafic et des données met à rude épreuve l'infrastructure de routage, ce qui peut entraîner des congestions et des pannes. Le manque de flexibilité et d'adaptation du routage rend difficile l'optimisation des performances et la gestion des flux de données.[10].

2.3 Qualité de Service (QoS) :

L'architecture actuelle d'internet ne garantit pas la qualité de service pour les applications en temps réel, comme la voix sur IP (VoIP) ou la vidéoconférence. Le manque de priorisation du trafic peut entraîner des congestions et des délais importants[11].

2.4 Besoins de Bande Passante croissants :

La demande en bande passante augmente de manière exponentielle avec la croissance du nombre d'utilisateurs et d'applications, L'infrastructure actuelle ne peut pas suivre le rythme de cette croissance, ce qui peut entraîner des congestions et des ralentissements de transmission.

2.5 Interopérabilité et Fragmentation :

L'absence d'une architecture internet standard a divisé le réseau en différents groupes isolés, causant des difficultés d'interopérabilité entre divers services et applications. Cette division complique les échanges de données entre systèmes, freinant l'innovation et le progrès des nouveaux services. Ainsi, favoriser un réseau plus ouvert et interopérable est crucial pour l'avenir de l'internet [12].

2.6 Épuisement des Adresses IP (IPv4) :

Le nombre d'adresses IP disponibles sous le protocole IPv4 est limité et risque de s'épuiser à court terme. Le passage au protocole IPv6, qui offre un espace d'adressage beaucoup plus large, est un processus complexe et long qui n'est pas encore achevé. L'épuisement des adresses IPv4 peut freiner la croissance d'Internet et limiter l'accès au réseau pour de nouveaux utilisateurs.

2.7 Centralisation des données :

La centralisation des données implique la concentration des informations dans des serveurs spécifiques ou des centres de données. Le modèle client-serveur est fondamental, où les clients demandent des services ou des données aux serveurs centralisés. Cependant, cette centralisation engendre des problèmes de disponibilité de données, et de scalabilité.

3. L'approche des réseaux orientés contenus : Une réponse aux problèmes de l'Internet actuel :

La transformation significative de l'utilisation d'Internet et les défis préalablement abordés ont conduit au développement de l'approche des réseaux centrés sur l'information (ICN)[2]. Cette approche se focalise sur la distribution des contenus indépendamment de leurs hôtes d'origine. Elle considère le contenu nommé comme l'élément central du réseau, intégrant de manière native la mise en cache. Ainsi, une copie du contenu demandé peut être récupérée à partir du nœud le plus approprié du réseau, répondant ainsi aux exigences identifiées pour une distribution de contenus plus efficace que celle de l'Internet actuel. De surcroît, grâce aux noms de contenus agissant en tant qu'identifiants indépendants des localisateurs, la mobilité n'est plus un problème. Pour répondre aux impératifs de sécurité, l'ICN remplace le modèle traditionnel de sécurisation des connexions par un modèle axé sur les contenus, en incorporant des mécanismes cryptographiques dans le contenu lui-même et en utilisant un système de nommage approprié.

3.1. ICN et CCN : Les origines de NDN :

Le réseau centré sur l'Information (ICN) [13]et le Réseau Centré sur le Contenu (CCN) sont des paradigmes de réseaux qui mettent l'accent sur la gestion et l'accès aux contenus plutôt que sur la communication avec des hôtes spécifiques. Named Data Networking (NDN) a émergé en tant qu'implémentation spécifique de ces concepts. Voici une exploration des origines de NDN dans le contexte de l'ICN et du CCN :

3.2. Réseau centré sur l'Information (ICN) :

L'ICN est un modèle de réseau qui cherche à transformer la manière dont les données sont gérées et échangées sur Internet en se focalisant sur les contenus plutôt que sur les adresses des hôtes. Cette approche a émergé en réponse aux limites de l'architecture actuelle d'Internet. Elle propose de nommer les contenus de manière unique et de les rendre directement accessibles, facilitant ainsi la distribution efficace des données[3].

3.3. Réseau Centré sur le Contenu (CCN) :

Le CCN, développé à partir de 2006 au centre de recherche PARC par Van Jacobson, constitue une implémentation concrète de l'ICN. Il vise à résoudre des limitations clés de l'architecture Internet actuelle, en mettant l'accent sur la gestion des données, la sécurité et la mobilité des dispositifs. Le CCN attribue des noms uniques aux contenus, transformant ainsi la manière dont les données sont échangées [14]

3.4. Named Data Networking (NDN):

Named Data Networking (NDN) est directement influencé par le CCN. Sous la direction de Lixia Zhang à l'Université de Californie à Los Angeles (UCLA), NDN a évolué en tant qu'architecture spécifique, mettant en œuvre les principes fondamentaux de l'ICN et du CCN[15].

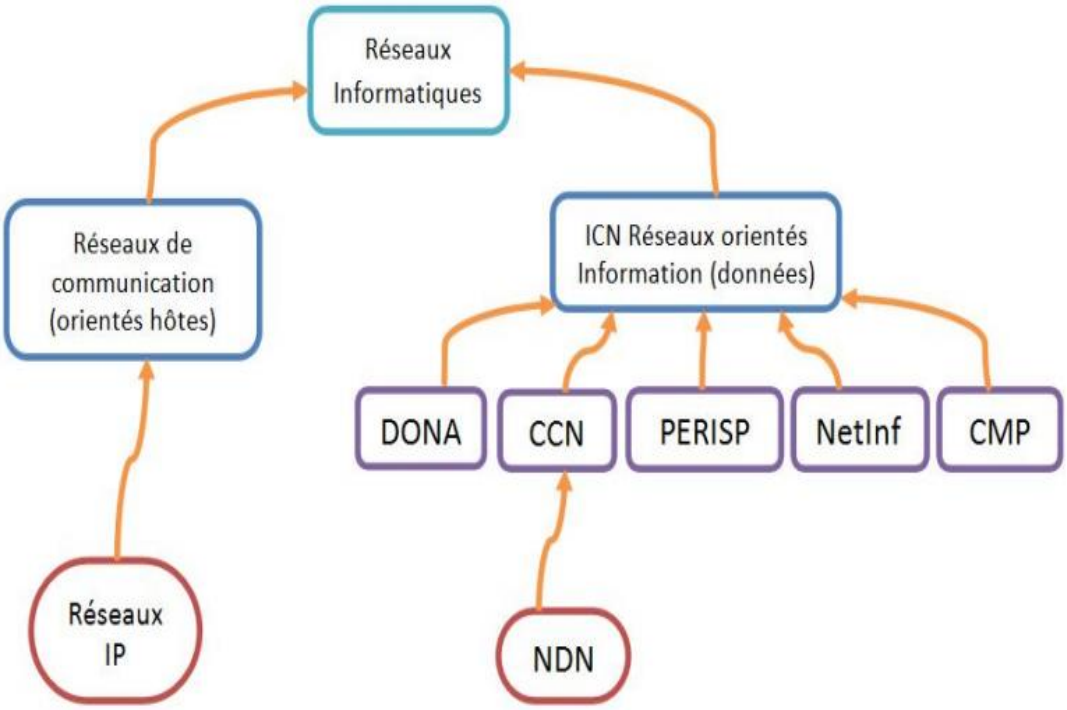


Figure 1 NDN et les principales architectures réseau [16]

4. Comment NDN relève-t-il les défis de l'IP et répond-il aux problèmes de rareté et de sécurité dans la communication de données ?

NDN[17] relève les défis de l'IP en proposant une approche centrée sur le contenu plutôt que sur les adresses IP. Contrairement à l'IP, qui utilise des adresses IP pour identifier les hôtes de destination, NDN utilise des noms de contenu pour identifier les données. Cela permet à NDN de résoudre les problèmes de rareté d'adresses IP et de faciliter la mobilité des utilisateurs, car les données peuvent être accessibles par leur nom de contenu plutôt que par une adresse IP spécifique. En ce qui concerne la sécurité, NDN utilise une approche de sécurité basée sur la signature numérique pour garantir l'intégrité et l'authenticité des données. Toutes les données sont signées par leur producteur et vérifiées par le consommateur, ce qui permet de garantir que les données n'ont pas été altérées ou falsifiées en transit. De plus, NDN utilise le chiffrement pour protéger les données en transit et les noms de contenu sont également chiffrés pour protéger la vie privée des utilisateurs. Enfin, NDN utilise également des mécanismes de mise en cache en réseau pour améliorer l'efficacité de la communication de données. Les données sont stockées

en cache sur les nœuds du réseau, ce qui permet de réduire la latence et d'améliorer les performances globales du réseau.

5. Les trois types d'entités principales dans l'Architecture NDN :

5.1 Producteur (Producer):

- Le producteur représente l'entité qui crée ou génère les données. Il publie ces données sur le réseau NDN, les associant à un nom unique. Lorsqu'un consommateur émet une requête pour un ensemble de données spécifique, le producteur peut répondre en envoyant les données associées à ce nom.

5.2 Consommateur (Consumer):

- Le consommateur est l'entité qui initie une demande de données dans le réseau NDN. Au lieu de faire référence à des adresses IP et de spécifier la source des données, le consommateur formule une requête **"Interest"** en utilisant le nom des données qu'il recherche. Lorsque la requête est diffusée dans le réseau, les routeurs NDN utilisent ce nom pour localiser et acheminer les données appropriées vers le consommateur.

5.3 Routeur (Router):

- Les routeurs NDN utilisent l'information de nom pour prendre des décisions de routage. Lorsqu'un routeur reçoit une requête de consommateur, il utilise le nom de la demande pour déterminer où trouver les données. De même, lorsqu'un routeur reçoit des données d'un producteur, il utilise le nom associé pour décider comment acheminer ces données vers les consommateurs qui ont émis des requêtes correspondantes. Les routeurs sont essentiels pour gérer la mise en cache des données, et faciliter la recherche et le routage dans le réseau NDN.

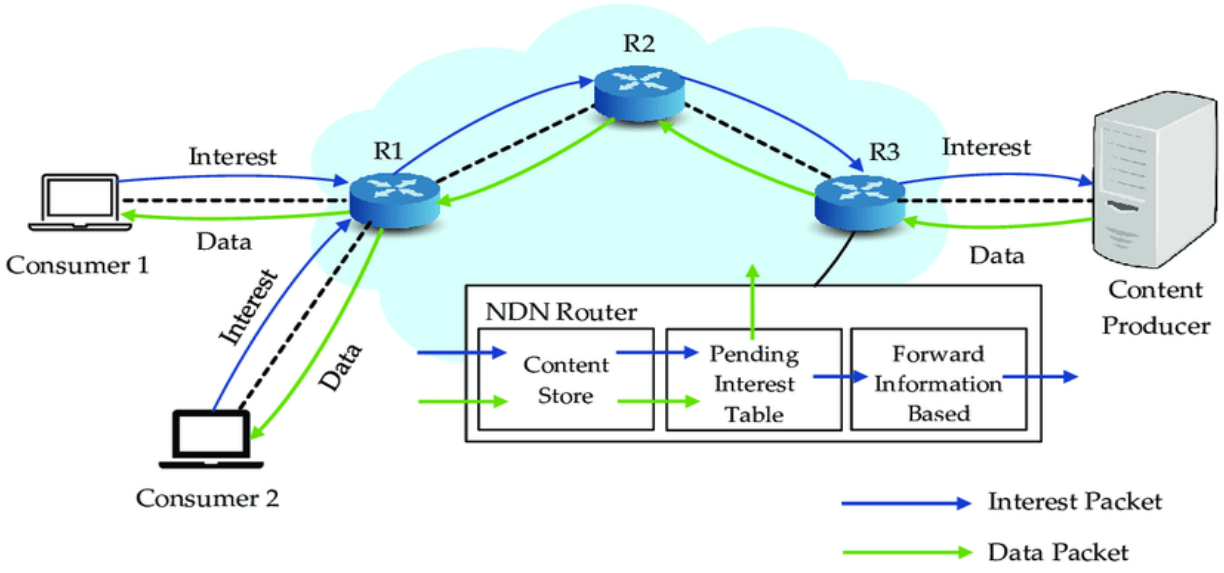


Figure 2 Les trois types d'entités principales dans l'Architecture NDN [18]

6. Les types de paquets dans NDN :

6.1. Paquets d'intérêt (Interest Packets) : Ces paquets sont envoyés par les nœuds demandeurs pour solliciter des données spécifiques. Lorsqu'un nœud souhaite obtenir un certain contenu, il envoie un paquet d'intérêt contenant le nom du contenu recherché. Ce paquet est propagé à travers le réseau à la recherche des données correspondantes.

6.2. Paquets de données (Data Packets) : Lorsqu'un nœud détenteur des données reçoit un paquet d'intérêt correspondant à un contenu qu'il possède, il répond en envoyant un paquet de données contenant le contenu demandé. Ces paquets de données sont signés par le producteur de contenu pour garantir leur authenticité et peuvent être mis en cache dans le réseau pour une utilisation ultérieure.

Ces deux types de paquets constituent le mécanisme fondamental de communication dans NDN, où les nœuds demandeurs sollicitent des données spécifiques en envoyant des paquets d'intérêt, et les nœuds détenteurs des données répondent en envoyant des paquets de données correspondants.

Figure 3 Les types de paquets dans NDN [19]

7. Les structures de données clés dans NDN :

Les structures de données adoptées pour le fonctionnement et la gestion des communications dans le cadre de NDN sont [20]:

7.1. Table des intérêts en attente (PIT : Pending Interest Table) : Cette table conserve et suit tous les "Intérêts" qui n'ont pas encore été satisfaits, afin que les contenus

retournés puissent être envoyés aux demandeurs. Chaque entrée dans la table PIT contient un Intérêt associé à une ou plusieurs interfaces physiques entrantes et sortantes.

7.2. Base d'Informations d'acheminement (FIB : Forwarding Information Base) : Cette structure de données cartographie les préfixes de noms vers une ou plusieurs interfaces réseau physiques, spécifiant les directions où les Intérêts peuvent être acheminés.

7.3. Magasin de contenu (CS : Content Store) : Cette structure met temporairement en mémoire tampon les contenus qui traversent le routeur, permettant une récupération rapide des données par différents consommateurs.

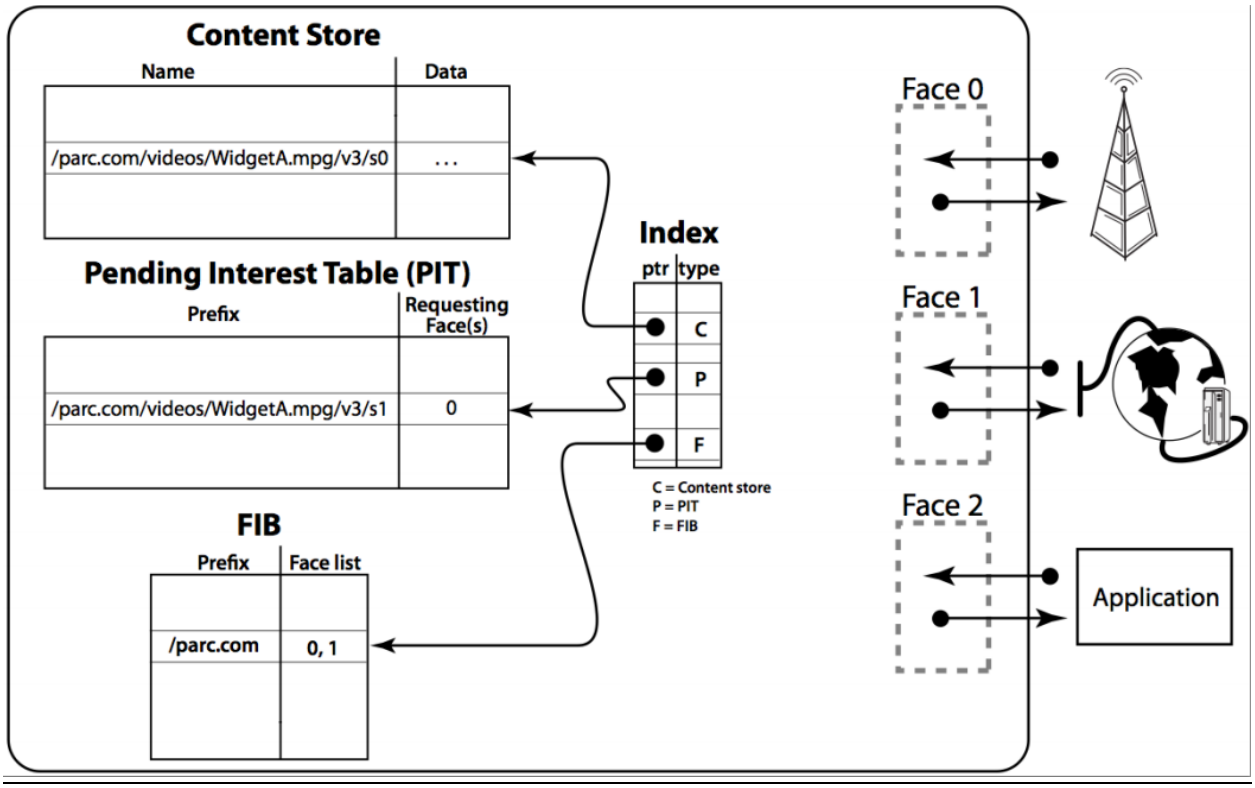


Figure 4 Les structures de données clés dans Named Data Networking (NDN) [21]

8. Comment NDN gère-t-il les paquets d'intérêt et de données :

Dans NDN, la gestion des paquets d'intérêt (Interest) et de données (Data) repose sur les structures de données mentionnées précédemment. Lorsqu'un consommateur émet un paquet d'intérêt pour demander un certain contenu, le routeur le plus proche consulte d'abord son CS local pour voir s'il dispose déjà de ce contenu en cache. Si le contenu est trouvé dans le CS, il est immédiatement retourné au consommateur. Dans le cas où le contenu n'est pas présent dans le CS, le routeur crée une entrée correspondante dans sa PIT pour mettre cet intérêt en attente. Ensuite, le paquet d'intérêt est envoyé vers les prochain routeur en utilisant la FIB, qui est une table de routage basée sur les préfixes de noms. Chaque routeur intermédiaire examine le paquet d'intérêt et utilise sa propre FIB pour déterminer la prochaine étape du routage. Si le routeur

dispose du contenu demandé dans son CS, il le retourne immédiatement. Sinon, il met à jour sa PIT avec l'intérêt entrant et le transmet plus loin dans le réseau. Lorsqu'un paquet de données correspondant à un intérêt est finalement trouvé, il est transmis de manière inverse vers le routeur ayant émis cet intérêt en utilisant les informations stockées dans la PIT. Ce processus se poursuit jusqu'à ce que le paquet de données atteigne le consommateur initial, qui peut alors utiliser les données reçues. En intégrant ces structures de données, les réseaux NDN parviennent à acheminer efficacement les paquets d'intérêt et de données, en minimisant ainsi le trafic réseau inutile [22].

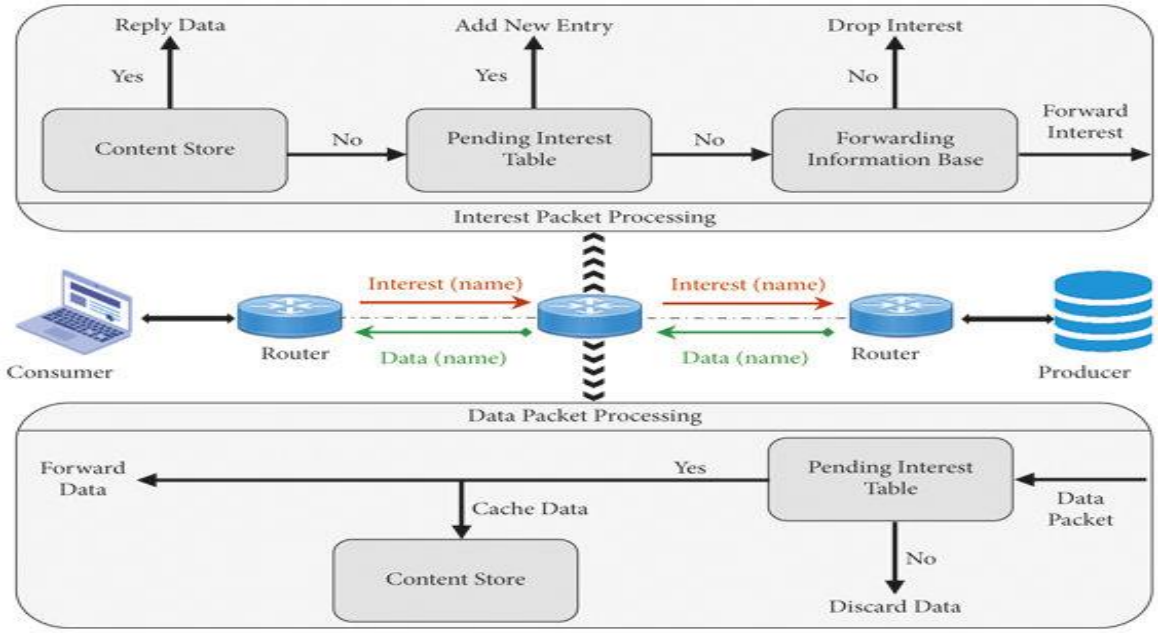


Figure 5 Gestion des paquets d'intérêt et de données dans NDN [22]

9. Analyse comparative entre l'IP et l'NDN :

9.1 Comparaison entre NDN et les réseaux basés sur IP :

Ce tableau présente une analyse comparative entre NDN et les réseaux fondés sur le protocole IP [23] :

Tableau 1 Comparaison entre NDN et les réseaux basés sur IP

Fonctionnalité	NDN	IP
Stockage de contenu	Le CS permet de stocker des copies de données aux différents nœuds du réseau, rapprochant ainsi le contenu des consommateurs et réduisant le besoin de le transférer sur de longues distances. Cela diminue le trafic global de réseau et améliore le débit perçu.	Ne dispose pas d'un mécanisme équivalent. Les données sont stockées sur des serveurs dédiés. Chaque accès nécessite un échange de paquets entre le client et le serveur, augmentant le trafic réseau et les latences.
Débit	Peut avoir un débit plus élevé que les réseaux IP Le CS peut réduire le trafic et améliorer le débit	Le débit dépend de la bande passante et des protocoles de routage. La congestion réduit le débit, surtout sur de longues distances.
Latence	Inférieur à celle du réseau IP. NDN permet une meilleure adaptation à la congestion du réseau.	Liée à la distance entre les nœuds. La congestion peut augmenter les délais, impactant ainsi la réactivité des applications.
Perte de paquets	Les deux architectures sont sensibles à la bande passante et au trafic. La congestion peut entraîner des pertes de paquets, mais le CS distribué de NDN peut aider à les atténuer.	Les deux architectures sont sensibles au trafic et à la congestion. La congestion peut engendrer des pertes de paquets.

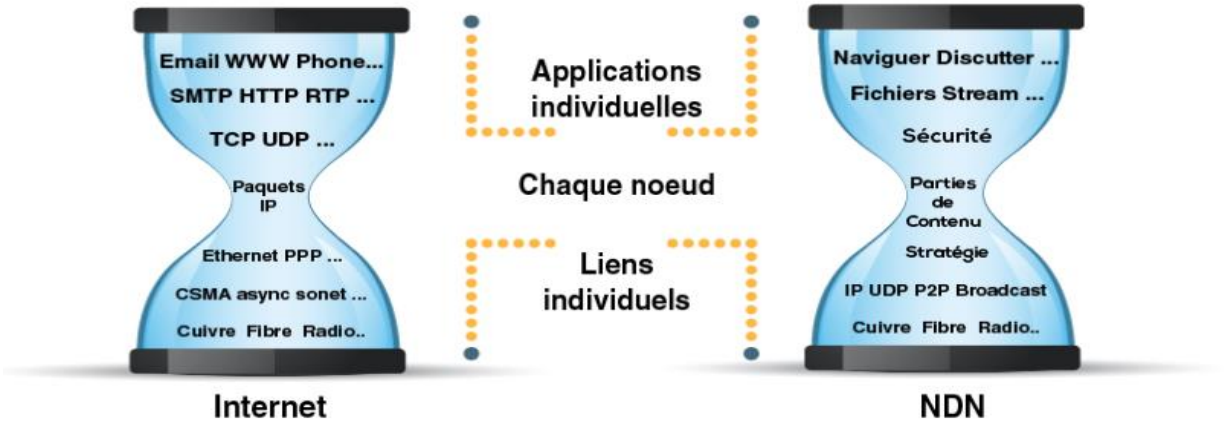


Figure 6 L'architecture sablier d'internet : TCP/IP vs NDN [19]

9.2. Les avantages de NDN par rapport aux réseaux IP :

- Un stockage de contenus efficace.
- Un débit potentiellement plus élevé.
- Des délais d'acheminement réduits.

NDN peut offrir certains avantages par rapport aux réseaux IP dans certaines situations. Cependant, il est important de noter que ces résultats sont basés sur une simulation spécifique et qu'ils peuvent varier en fonction des conditions réelles. De plus, NDN est une technologie relativement nouvelle et elle n'est pas encore largement déployée. Les réseaux IP sont la norme actuelle et ils offrent un large éventail de fonctionnalités et de services. Il est donc probable que les réseaux IP continueront à être utilisés pour la plupart des applications pendant encore de nombreuses années.

En résumé, les différences de performance entre NDN et les réseaux IP résultent de leurs philosophies et mécanismes sous-jacents. NDN vise à optimiser l'accès aux données en les rapprochant des consommateurs, tandis que les réseaux IP reposent sur une communication client-serveur plus traditionnelle. Chacune des architectures présente des avantages et des inconvénients, et leur performance optimale dépend de facteurs spécifiques comme la topologie du réseau, le type de trafic et les applications utilisées.

10. Les principales caractéristiques de NDN :

Les principales caractéristiques de paradigme NDN (Named Data Networking) pour la future architecture Internet sont les suivantes :

10.1. Nommage :

Les réseaux NDN utilisent des objets de données nommés (NDO) pour représenter divers contenus, tels que des pages Web, des vidéos, des images, etc. Le nommage est souvent hiérarchique et permet une identification unique des contenus.

10.2. Routage basé sur le nom :

Les réseaux NDN intègrent des mécanismes de routage basés sur le nom, où les demandes de contenu sont acheminées en fonction des noms de contenu plutôt que des adresses d'hôtes. Cela permet une réactivité élevée en cas de changements soudains dans le réseau.

10.3. Mise en cache dans le réseau :

La mise en cache des données au sein des nœuds du réseau est une caractéristique clé de l'approche NDN. Cela permet d'améliorer les performances de distribution de contenu en réduisant les temps de réponse et en allégeant les charges des serveurs.

10.4. Sécurité et authenticité du contenu :

Dans les réseaux NDN, la priorité est donnée à la garantie de l'authenticité et de l'intégrité du contenu. Cela se fait notamment en signant numériquement les correspondances entre les noms et le contenu pour assurer la sécurité des données.

10.5. Décentralisation de l'information :

Les NDN se concentrent sur le contenu lui-même plutôt que sur sa localisation sur Internet, ce qui permet une distribution plus efficace des contenus et une meilleure utilisation des ressources réseau.

10.6. Amélioration de l'efficacité du réseau :

L'architecture NDN vise à améliorer l'efficacité du réseau, sa scalabilité et la distribution du contenu, en réponse aux demandes croissantes de bande passante des utilisateurs. Ces caractéristiques définissent les fondements de l'architecture NDN et illustrent comment ce paradigme cherche à surmonter les limitations du modèle actuel de l'Internet. Il met l'accent sur le contenu, la sécurité et la mise en cache pour atteindre cet objectif.

11. Les principaux avantages de NDN :

La transition d'un modèle de communication hôte-à-hôte à un modèle de communication centré sur le contenu dans le contexte de Named Data Networking (NDN) offre plusieurs avantages potentiels, notamment[24] :

11.1. Scalabilité améliorée :

En dissociant le contenu des hôtes spécifiques, l'NDN peut potentiellement mieux s'adapter à la croissance du volume de contenu et d'utilisateurs sur Internet.

11.2. Distribution efficace de contenu :

Les modèles centrés sur le contenu peuvent conduire à une distribution plus rentable du contenu en tirant parti de la mise en cache dans le réseau et en optimisant l'allocation de bande passante en fonction de la demande de contenu.

11.3. Nommage persistant et unique du contenu :

L'approche Named Data Networking NDN permet un nommage persistant et unique du contenu, indépendamment de son emplacement, de son stockage ou de son moyen de transport, ce qui peut simplifier la récupération et la gestion du contenu.

11.4. Amélioration de la sécurité et de l'authentification :

La focalisation de l'NDN sur l'authentification des objets d'information (IO) plutôt que sur les points finaux peut renforcer la sécurité et protéger contre l'accès non autorisé ou la manipulation du contenu.

11.5. Prise en charge de la mobilité et du multirattachement :

Les modèles centrés sur le contenu peuvent offrir un meilleur support pour les appareils mobiles et le multirattachement, permettant une connectivité transparente et répondant aux défis de la mobilité des hôtes dans l'architecture Internet actuelle.

11.6. Robustesse et tolérance aux perturbations :

La conception de l'NDN peut potentiellement améliorer la robustesse et la résilience des scénarios de communication, la rendant plus adaptable aux conditions réseau difficiles.

Ces avantages potentiels mettent en lumière les avantages de la transition vers un modèle de communication centré sur le contenu au sein de l'NDN, en abordant les principaux défis et limitations de l'architecture Internet actuelle tout en offrant des opportunités d'amélioration de l'efficacité, de la scalabilité et de la sécurité.

12. Le nommage dans NDN :

Dans NDN (Named Data Networking)[24], le nommage se compose de plusieurs composants de longueur variable et organisés de manière hiérarchique. Les noms NDN sont similaires aux URL traditionnelles, mais ils ne sont pas nécessairement lisibles par l'homme. Les noms peuvent contenir des informations telles que des segments et des numéros de version, et chaque nom dans NDN contient un digest SHA256 qui aide à résoudre les ambiguïtés de contenu. Pour assurer l'intégrité et l'authenticité du contenu, les noms et le contenu sont sécurisées par une signature numérique et sont livrées avec le contenu lui-même. Les noms NDN peuvent être utilisés pour demander des objets de données, qui peuvent être stockés dans le Content Store (CS) d'un routeur NDN. Les noms sont également utilisés pour acheminer les paquets de données et les paquets d'intérêt dans le réseau NDN.

Figure 7 La structure des noms dans NDN [25]

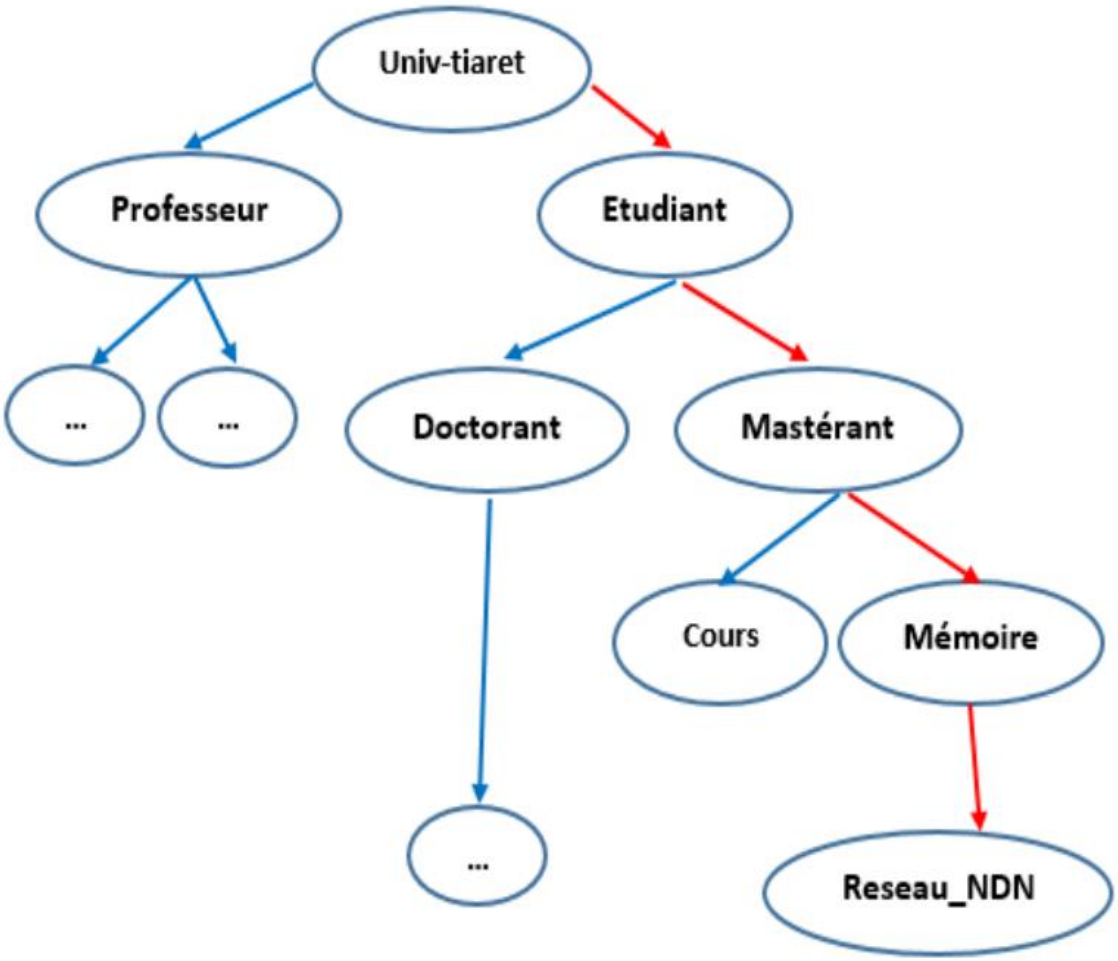


Figure 8 Le schéma de nommage dans l'architecture NDN [25]

13. La sécurité dans NDN :

La sécurité est une caractéristique fondamentale de Named Data Networking (NDN)[17]. NDN intègre plusieurs mécanismes de sécurité pour garantir l'intégrité, l'authenticité et la confidentialité des données échangées. Voici quelques aspects clés de la sécurité dans NDN :

13.1. Signature numérique :

Chaque paquet de données (D_pkt) dans NDN est cryptographiquement signé par le fournisseur de contenu, ce qui permet à chaque consommateur de vérifier l'authenticité des données qu'il reçoit.

13.2. Chiffrement :

NDN utilise le chiffrement pour protéger les données en transit. Cela garantit que seuls les destinataires autorisés peuvent accéder aux données et que la confidentialité des informations est préservée.

13.3. Contrôle d'accès : NDN prend en charge le contrôle d'accès basé sur les noms de contenu, ce qui permet de définir des politiques de sécurité pour restreindre l'accès aux données en fonction de leur nom.

13.4. Mécanismes de confiance : NDN intègre des mécanismes de gestion de la confiance pour garantir que seules les sources de données fiables sont autorisées à publier des données dans le réseau.

En résumé, NDN offre une architecture de sécurité robuste qui vise à protéger les données tout au long de leur cycle de vie, de la publication à la consommation, en garantissant l'authenticité, l'intégrité et la confidentialité des données échangées.

14. Les avantages de la mise en cache et du routage dans NDN :

La mise en cache et le routage des données jouent un rôle essentiel dans l'optimisation de l'efficacité du réseau ainsi que dans l'amélioration des performances de distribution de contenu au sein de l'NDN, apportant ainsi des bénéfices multiples [24] :

14.1. Amélioration de l'utilisation des ressources réseau :

La mise en cache permet de stocker temporairement des contenus populaires à proximité des utilisateurs, réduisant ainsi la latence et l'utilisation de la bande passante en évitant des requêtes répétées vers des sources distantes.

14.2. Réduction de la charge sur les serveurs d'origine :

En stockant les contenus de manière distribuée dans le réseau, la mise en cache réduit la charge sur les serveurs d'origine, améliorant ainsi leur disponibilité et réduisant les goulots d'étranglement potentiels.

14.3. Optimisation du routage des requêtes :

Le routage des données basé sur les noms dans l'NDN permet une distribution efficace des requêtes en fonction des noms de contenu, favorisant ainsi une utilisation optimale des ressources réseau.

14.4. Réduction de la congestion réseau :

En acheminant les requêtes vers des caches locaux plutôt que vers des serveurs distants, le routage des données contribue à réduire la congestion du réseau et à améliorer la réactivité globale du système.

14.5. Meilleure gestion de la demande de contenu :

La mise en cache et le routage des données permettent de mieux gérer la demande de contenu en fournissant des mécanismes pour localiser et livrer efficacement les contenus demandés.

En combinant la mise en cache efficace avec des stratégies de routage basées sur les noms de contenu, l'NDN vise à optimiser l'utilisation des ressources réseau, à réduire la charge sur les

serveurs d'origine et à améliorer la distribution de contenu, contribuant ainsi à une meilleure efficacité globale du réseau.

Conclusion :

L'évolution rapide des besoins de communication et la croissance exponentielle du trafic Internet ont mis en lumière les limitations de l'architecture IP actuelle. Face à ces défis, l'approche des réseaux orientés contenus, notamment incarnée par le CCN et plus spécifiquement par le réseau NDN**,** est envisagée.

La solution NDN propose de passer d'une architecture centrée sur les hôtes à une architecture centrée sur le contenu, où l'identification se fait par des noms de contenu plutôt que des adresses IP. Cela permet une distribution efficace du contenu, une meilleure évolutivité pour répondre à la croissance du trafic, une résilience accrue aux pannes, et une amélioration de la sécurité et de la confidentialité.

Le réseau NDN, en particulier, met en œuvre ces principes en utilisant des mécanismes de cache pour stocker temporairement les données à différents nœuds du réseau, réduisant ainsi la charge sur les serveurs d'origine et améliorant l'efficacité du réseau. La gestion des paquets d'intérêt et de données, ainsi que la sécurité intégrée grâce à la signature numérique et au chiffrement, renforcent la fiabilité et l'authenticité des échanges dans le réseau NDN.

La comparaison entre NDN et les réseaux IP met en évidence les avantages potentiels de NDN en termes de stockage de contenu, de débit potentiellement plus élevé, et de latences réduites. Cependant, il est souligné que ces résultats sont basés sur des simulations spécifiques [23] et que les réseaux IP restent la norme actuelle avec une large gamme de fonctionnalités.

Pour conclure, l’approche NDN offre une perspective prometteuse pour l'avenir de l'Internet en abordant les limitations de l'architecture IP actuelle. Cependant, la transition vers cette nouvelle architecture nécessitera une adoption progressive et des efforts concertés pour assurer une interopérabilité et une évolutivité optimales.

Chapitre 2

La Mise en Cache Dans NDN

Chapitre 2 La Mise en Cache Dans NDN

INTRODUCTION

La mise en cache représente une avancée majeure dans l'architecture réseau, permettant de réduire le trafic en stockant des copies de contenu dans un emplacement temporaire unique pour un accès rapide ultérieur. Dans l'architecture NDN, cette pratique offre des avantages comparatifs par rapport à l'approche traditionnelle TCP/IP. Ce qui distingue les routeurs NDN, c'est leur capacité à réutiliser les données grâce à leur identification par des noms, contrairement à simplement mettre en mémoire tampon le contenu. Ce chapitre explore le concept de la mise en cache dans les réseaux NDN, en examinant les différentes stratégies associées à cette pratique révolutionnaire.

1. L'importance du cache dans les réseaux NDN

La mise en cache joue un rôle critique dans les réseaux NDN [26], offrant de nombreux avantages qui contribuent à une livraison de contenu efficace et à des performances réseau optimales. L'architecture distinctive de NDN, centrée autour des données nommées, permet la mise en œuvre de stratégies de mise en cache qui ne sont pas viables dans les réseaux TCP/IP conventionnels. Contrairement à la mise en cache traditionnelle dans les réseaux TCP/IP, l'approche de NDN est axée sur le consommateur et se concentre sur les noms de contenu plutôt que sur les identifiants de réseau d'hôtes. Cela permet à NDN de mettre naturellement en cache les contenus populaires près des utilisateurs, entraînant une réduction de la latence des demandes et une amélioration globale de l'expérience utilisateur.

L'un des principaux avantages de la mise en cache dans les réseaux NDN est la réduction de la congestion et l'accélération de la livraison de contenu grâce au stockage et à la récupération de contenus populaires. En mettant en cache les données plus près du bord du réseau, NDN permet aux consommateurs d'accéder au contenu avec une latence plus faible et réduit le trafic dans le cœur du réseau. Cela a des implications pour la réduction des coûts d'infrastructure globaux et des coûts d'interconnexion pour les opérateurs réseau [20].

En plus de réduire la congestion, la mise en cache dans les réseaux NDN simplifie la configuration du réseau et intègre la sécurité au niveau des données. Chaque paquet de données est cryptographiquement signé par son producteur, garantissant ainsi la vérifiabilité de son origine. De plus, l'architecture NDN permet la mise en œuvre de stratégies d'acheminement intelligentes et élimine les boucles, contribuant ainsi à un contrôle de flux efficace.

Tandis que les routeurs IP traditionnels mettent en cache les données uniquement à des fins de mise en file d'attente et de planification, les routeurs NDN utilisent les données mises en cache pour répondre aux demandes, ce qui permet une utilisation plus efficace des ressources de stockage. Le processus de prise de décision pour mettre en cache des données dans les nœuds NDN implique des stratégies de décision, en fonction de facteurs tels que la disponibilité du cache, le trafic réseau ou la popularité du contenu [17][23].

Dans l'ensemble, l'importance de la mise en cache dans les réseaux NDN ne peut être surestimée. Elle facilite la distribution efficace de contenu sans dépendre d'infrastructures spécialisées telles que les CDN (Content Delivery Network) ou des contrats de service coûteux, tout en permettant une distribution de contenu à l'échelle mondiale de manière efficace et démocratisée.

2. Processus de mise en cache dans NDN

La mise en cache dans les réseaux NDN consiste à stocker localement les données nommées les plus demandées afin d'accélérer leur accès ultérieur. Lorsqu'un routeur du réseau reçoit une requête pour des données nommées, il vérifie d'abord s'il dispose de ces données dans son cache local. Si c'est le cas, le routeur peut répondre à la requête directement à partir de son cache, ce qui réduit la latence et la charge sur le réseau. Dans les réseaux NDN, la mise en cache[27] est étroitement liée aux tables PIT (Pending Interest Table) et FIB (Forwarding Information Base), qui sont des composants essentiels du routage et du traitement des requêtes. La décision de mettre en cache une donnée nommée est généralement basée sur des critères tels que la fréquence des demandes, la taille des données et les politiques de gestion de cache.

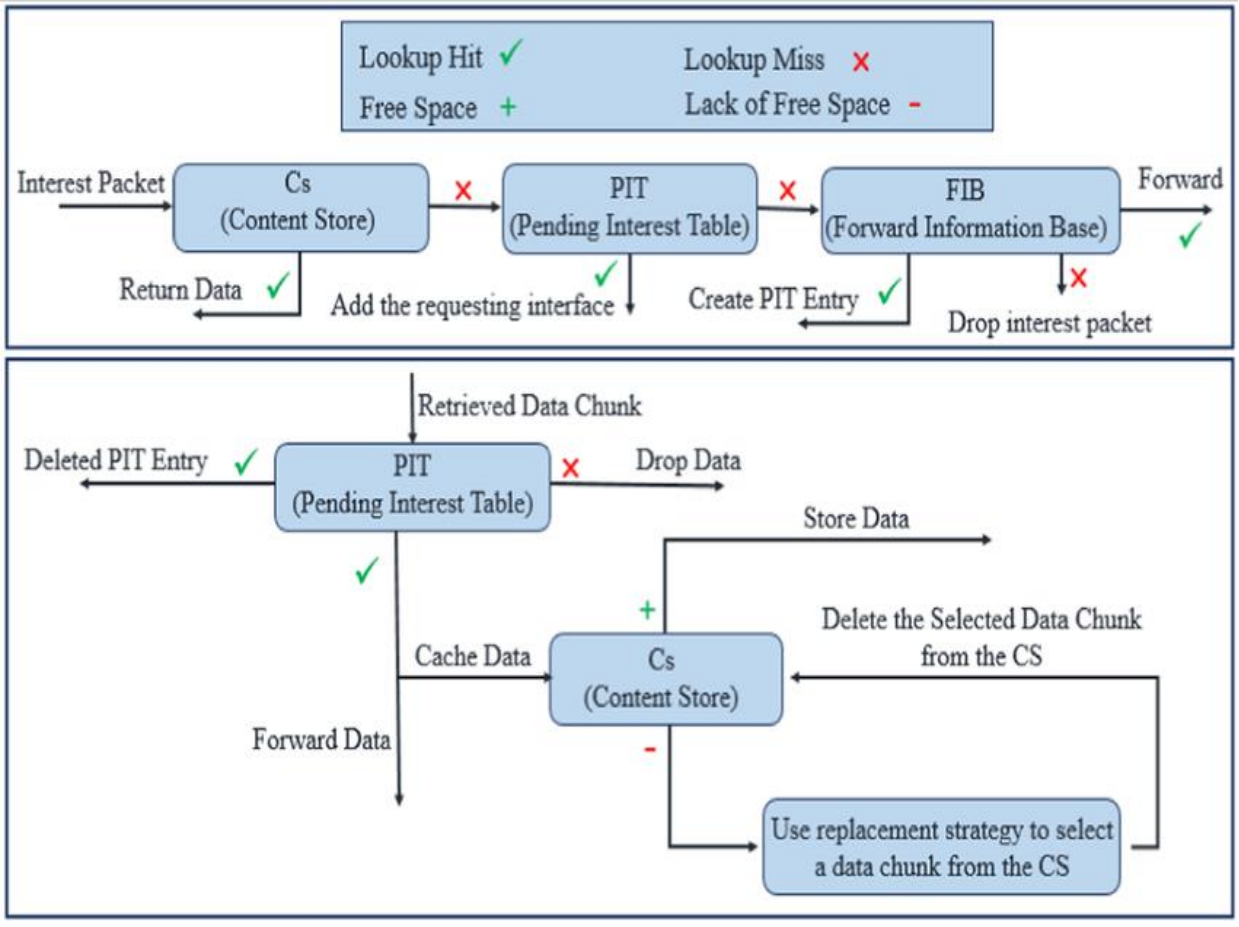


Figure 9 La mise en cache dans NDN [28]

Les données mises en cache sont souvent associées à des entrées dans la table PIT, qui enregistre les requêtes d'intérêt en attente pour ces données. Lorsqu'une donnée nommée est mise en cache, le nœud peut également mettre à jour la table PIT en conséquence, en satisfaisant les requêtes d'intérêt en attente. Parallèlement, la mise en cache est également influencée par la table FIB, qui stocke des informations de routage permettant de déterminer la manière dont les paquets de données doivent être transmis vers leur destination. Les entrées de la table FIB indiquent les prochains sauts pour atteindre les nœuds capables de fournir les données demandées. Ainsi, lorsqu'une donnée nommée est mise en cache, le nœud peut mettre à jour la table FIB pour diriger les futures requêtes directement vers son propre cache ou vers d'autres nœuds du réseau détenant les données demandées[27].

De plus, la mise en cache contribue à améliorer la résilience du réseau en réduisant les risques de perte de données dues à des pannes ou à des problèmes de connectivité. Ainsi, la mise en cache dans les réseaux NDN joue un rôle crucial dans l'amélioration des performances, de l'efficacité et de la fiabilité du réseau [27].

3. La classification des mécanismes de mise en cache dans NDN

Dans les réseaux NDN, deux stratégies de mise en cache sont simultanément employées : celle du placement et celle du remplacement du cache.

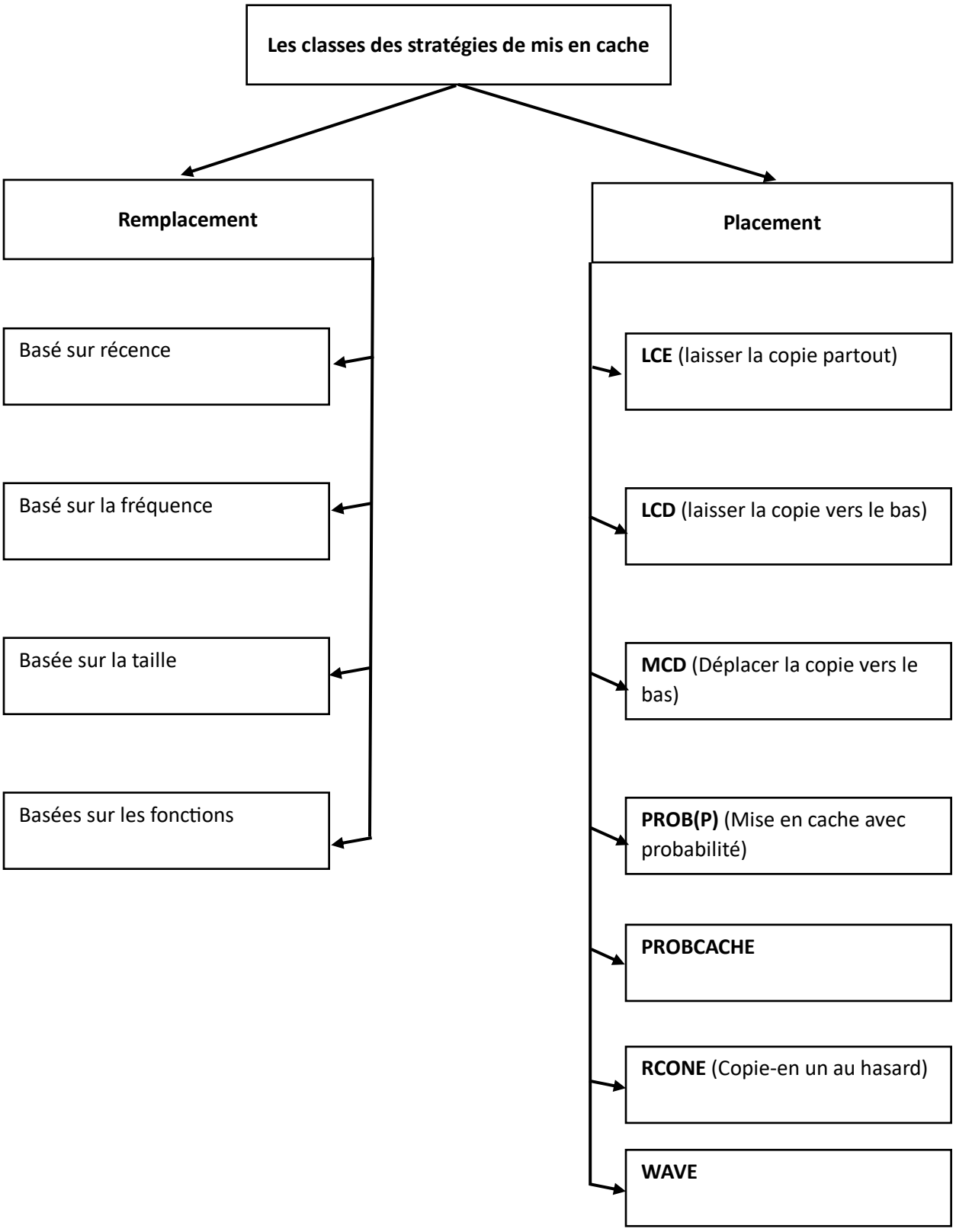


Figure 10 Classification des stratégies de mise en cache[29][30]

3.1 Stratégies de placement de cache

Ces stratégies visent à déterminer où placer en cache les données. Voici quelques exemples de stratégies de placement :

3.1.1. La stratégie LCE

L'algorithme LCE (Leave Copy Everywhere : Laissez une copie partout) [30] représente la stratégie de mise en cache par défaut des réseaux NDN. Son objectif principal est de distribuer efficacement le contenu en laissant une copie à chaque routeur traversé par la demande de ce contenu, réduisant ainsi les délais d'accès pour les utilisateurs ultérieurs.

Voici comment fonctionne l'algorithme LCE dans NDN :

1. Lorsqu'une demande de contenu est émise par un consommateur, elle est transmise à travers le réseau vers le routeur le plus proche.
2. Le routeur examine la demande et vérifie s'il a déjà une copie du contenu demandé dans sa mémoire cache CS.
3. Si le contenu est présent dans le CS du routeur, il est immédiatement renvoyé au consommateur qui a émis la demande.
4. Si le contenu n'est pas présent dans le CS du routeur, celui-ci récupère le contenu depuis la source d'origine (producteur) et le transmet à l'utilisateur.
5. Avant de transmettre le contenu à l'utilisateur, les routeurs traversés enregistrent une copie du contenu dans leurs mémoires cache CS.
6. Ensuite, chaque routeur traversé transmet le contenu au routeur suivant le long du chemin vers l'utilisateur.
7. Ce processus se répète à chaque demande de contenu, ce qui permet de distribuer les copies du contenu à travers le réseau et d'améliorer la disponibilité et la rapidité d'accès au contenu pour les utilisateurs ultérieurs.

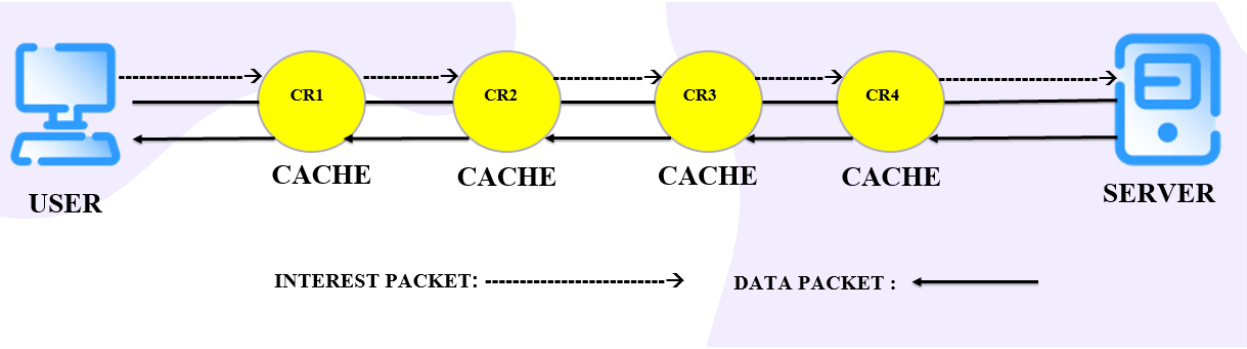


Figure 11 Stratégies de placement de cache LCE

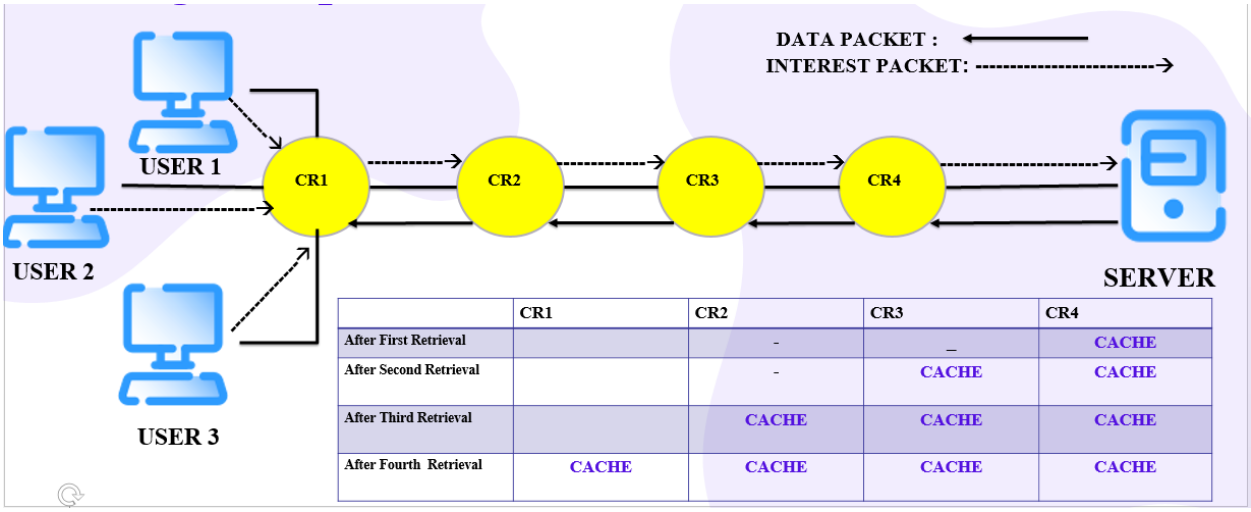
3.1.2. La stratégie LCD

Contrairement à la stratégie LCE, où les données sont copiées et maintenues à plusieurs endroits du réseau, la stratégie LCD (Leave Copy Down : Laisser la copie vers le bas) choisit de conserver

une copie de la donnée uniquement au niveau des nœuds descendants dans l'arborescence du réseau [30].

Avec la stratégie LCD, la copie des données est propagée vers le bas dans l'arborescence du réseau, ce qui signifie que les données sont stockées uniquement dans les nœuds descendants d'un nœud particulier, plutôt que d'être répliquées sur tous les nœuds comme dans le cas de LCE [30].

En ne maintenant les copies que dans les nœuds descendants, la stratégie LCD peut être plus efficace en termes d'utilisation de l'espace de stockage, car elle ne crée pas de copies redondantes dans tout le réseau. De plus, en évitant la duplication excessive des données, LCD peut contribuer à réduire la surcharge du réseau et à améliorer l'efficacité de la gestion des données, surtout dans les réseaux distribués de grande taille. Cependant, la stratégie LCD peut rendre la gestion de la cohérence des données plus complexe. Puisque les données sont répliquées de manière sélective, il peut être nécessaire de mettre en œuvre des mécanismes de synchronisation pour assurer la cohérence des données entre les nœuds du réseau.



	CR1	CR2	CR3	CR4
After First Retrieval		-	_	CACHE
After Second Retrieval		-	CACHE	CACHE
After Third Retrieval		CACHE	CACHE	CACHE
After Fourth Retrieval	CACHE	CACHE	CACHE	CACHE

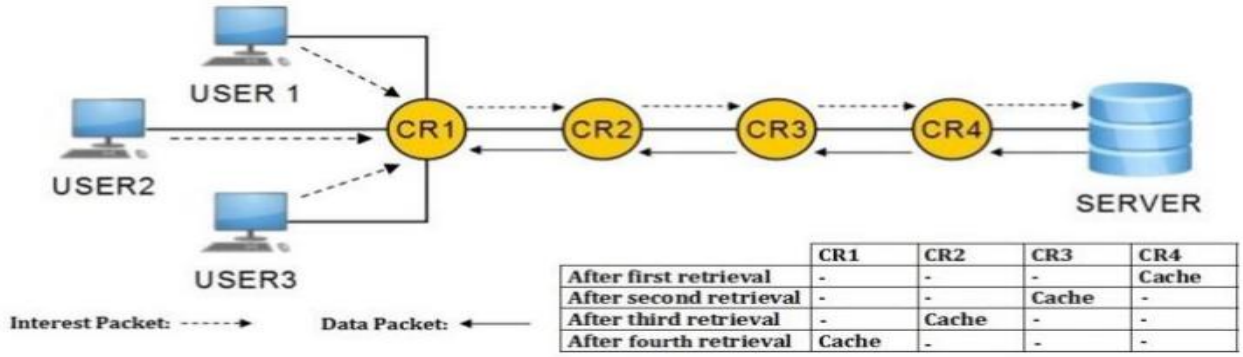
Figure 12 Stratégies de placement de cache LCD

3.1.3. La stratégie MCD

Contrairement à d'autres stratégies telles que LCE ou LCD, où les copies de données sont placées statiquement ou répliquées à divers endroits du réseau, la stratégie MCD (Move Copy Down : Déplacer la copie vers le bas) implique un déplacement dynamique des copies de données vers le bas de l'arborescence du réseau à mesure que les requêtes sont satisfaites, en les plaçant près des nœuds demandeurs [30].

Plutôt que de maintenir des copies de données statiques à divers endroits du réseau, MCD ajuste constamment l'emplacement des copies en fonction de la demande des utilisateurs, ce qui signifie que les copies de données sont déplacées progressivement vers les nœuds qui les demandent le plus fréquemment [30].

Une des motivations derrière MCD est de placer les copies de données aussi près que possible des nœuds qui les demandent, ce qui permet de réduire les temps de latence et à améliorer les performances du système en répondant plus rapidement aux requêtes. De plus, MCD contribue à optimiser l'utilisation des ressources du réseau. Plutôt que de maintenir des copies de données inutiles à des endroits peu sollicités, MCD les déplace là où elles sont le plus nécessaires. Elle peut également aider à équilibrer la charge du réseau en déplaçant les copies de données vers les nœuds qui subissent le plus de demandes, ce qui peut aider à éviter la surcharge de certains nœuds tout en répartissant la charge de manière plus uniforme sur l'ensemble du réseau. Cependant, sa mise en œuvre peut être plus complexe que d'autres stratégies de gestion de cache. Il peut nécessiter des mécanismes sophistiqués de suivi de la demande des utilisateurs et de déplacement des copies de données de manière efficace et sécurisée à travers le réseau.



	CR1	CR2	CR3	CR4
After first retrieval	-	-	-	Cache
After second retrieval	-	-	Cache	-
After third retrieval	-	Cache	-	-
After fourth retrieval	Cache	-	-	-

Figure 13 Stratégies de placement de cache MCD[29]

3.1.4. La stratégie PROB(P)

Contrairement aux méthodes de mise en cache traditionnelles qui se basent sur des politiques déterministes pour décider quelles données mettre en cache, PROB(P) utilise des algorithmes probabilistes pour décider quelles données conserver dans le cache et quelles données supprimer en fonction de leur probabilité d'être demandés à nouveau, ce qui signifie que les décisions de mise en cache sont adaptatives et peuvent évoluer au fil du temps en réponse aux changements dans les schémas de demande de données [31]. La probabilité qu'un contenu c soit demandé à nouveau par le nœud i est définie comme suit :

$$\boldsymbol{P(c,i) = P_{c,i} \times \frac{b_{c,i,r}}{d_{r,s}}}$$

Où :

- $p_{c,i}$: Popularité locale du contenu c au nœud n_i .
- $b_{c,i,r}$: Nombre de demandes pour le contenu c au nœud n_i .
- $d_{r,s}$: Nombre de demandes pour le contenu le plus demandé au nœud n_i .
- Le rapport $b_{c,i,r} / d_{r,s}$ sert à normaliser $p_{c,i}$ sur une échelle de 0 à 1.

L'idée fondamentale derrière PROB est d'attribuer à chaque donnée une probabilité d'être demandé à nouveau dans un futur proche, qui peut être calculée par exemple en se basant sur des statistiques historiques de demandes. PROB(P) peut être particulièrement bénéfique dans les environnements où les schémas de demande de données sont variables ou difficiles à prédire à l'avance. Elle vise à optimiser l'utilisation des ressources en conservant les données qui ont une forte probabilité d'être demandés à nouveau et en supprimant ceux avec une faible probabilité pour libérer de l'espace pour des données plus pertinentes[32].

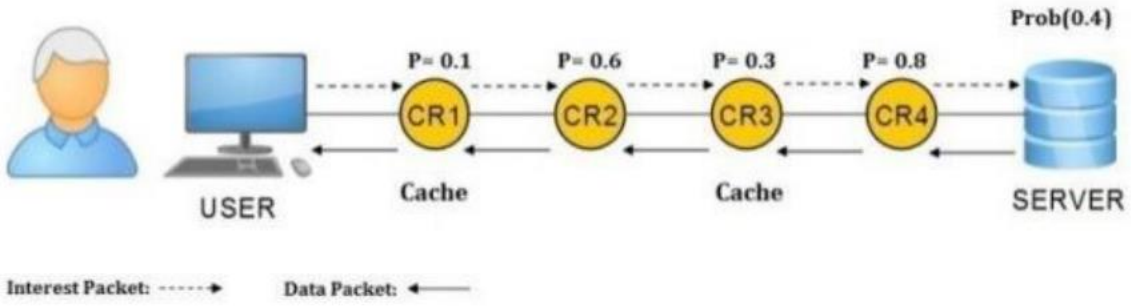


Figure 14 Stratégies de placement de cache PROB(P) [29]

3.1.5. La stratégie Wave

La stratégie "Wave" est une approche de gestion de données distribuées qui est inspirée de la propagation d'une vague. Dans ce contexte, une copie de données est propagée dans le réseau de manière itérative, chaque nœud qui reçoit une copie la transmet à ses nœuds voisins, créant ainsi une " vague " de copies de données se propageant à travers le réseau. Initialement, une copie de données est placée dans un nœud source ou un ensemble de nœuds sources. Ensuite, à chaque étape de propagation, les nœuds qui ont déjà reçu une copie de données la transmettent à leurs nœuds voisins qui ne l'ont pas encore reçue [29]. Voici une explication détaillée de son fonctionnement (voir la Figure ci-dessous) :

1. Propagation itérative des données : Dans la stratégie "Wave", une copie de données est propagée dans le réseau de manière itérative. Initialement, cette copie est placée dans un nœud source ou un ensemble de nœuds sources.

2. Transfert aux nœuds voisins : À chaque étape de propagation, les nœuds qui ont déjà reçu une copie de données la transmettent à leurs nœuds voisins qui ne l'ont pas encore reçue. Cela crée une propagation en cascade des données à travers le réseau, similaire à la manière dont une vague se propage dans un milieu liquide.

3. Formation d'une "vague" de copies de données : En transmettant les copies de données de proche en proche, une "vague" de données se forme, se propageant à travers le réseau. Chaque nœud qui reçoit une copie de données devient à son tour un émetteur, contribuant ainsi à la propagation continue de la vague.

4. Réplication et redondance : Cette approche permet une réplication distribuée des données dans le réseau, ce qui peut améliorer la disponibilité et la redondance des données. En cas de défaillance d'un nœud ou d'une connexion, les données peuvent toujours être récupérées à partir d'autres nœuds qui ont reçu des copies de la vague de données.

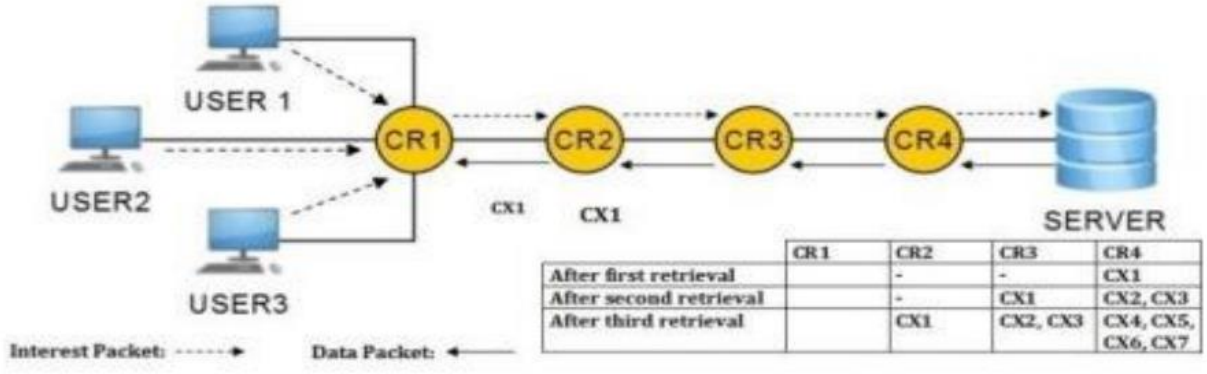


Figure 15 Stratégie de placement de cache WAVE [29]

3.1.6. La stratégie RCOne

Cette stratégie consiste à choisir aléatoirement un seul nœud sur le chemin pour conserver une copie de la donnée demandée [27]. RCOne (Randomly Copy One : Copiez-en un au hasard) est souvent choisie pour sa simplicité et pour éviter la surcharge des nœuds et répartir équitablement la charge de stockage dans le réseau. Comparée à d'autres stratégies qui impliquent des calculs complexes ou des algorithmes sophistiqués, RCOne est relativement facile à mettre en œuvre et à gérer. Cependant, elle peut entraîner des performances variables en fonction de la chance de choisir un nœud proche du demandeur.

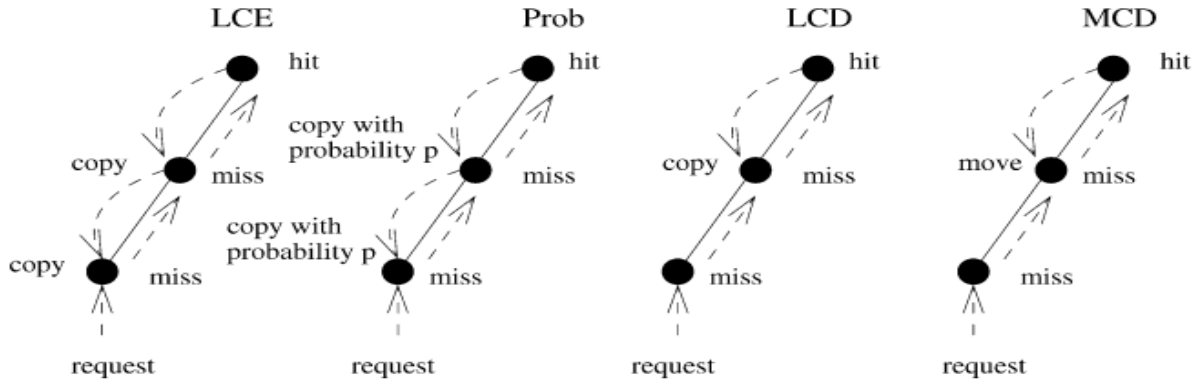


Figure 16 Fonctionement de LCE, Prob, LCD et MCD[30]

3.2. Politiques de remplacement du cache

La stratégie de remplacement vise à sélectionner les contenus à retirer de la mémoire cache (CS) afin de libérer de l'espace pour de nouvelles entrées, en raison de sa taille limitée. En d'autres termes, lorsque le cache est plein et qu'un nouveau contenu doit être ajouté, un contenu existant est supprimé pour faire de la place. Dans la suite, nous étudierons différentes stratégies de

remplacement pour gagner une compréhension approfondie de leurs mécanismes, de leurs avantages et de leurs limites. Voici quelques-unes des politiques de remplacement :

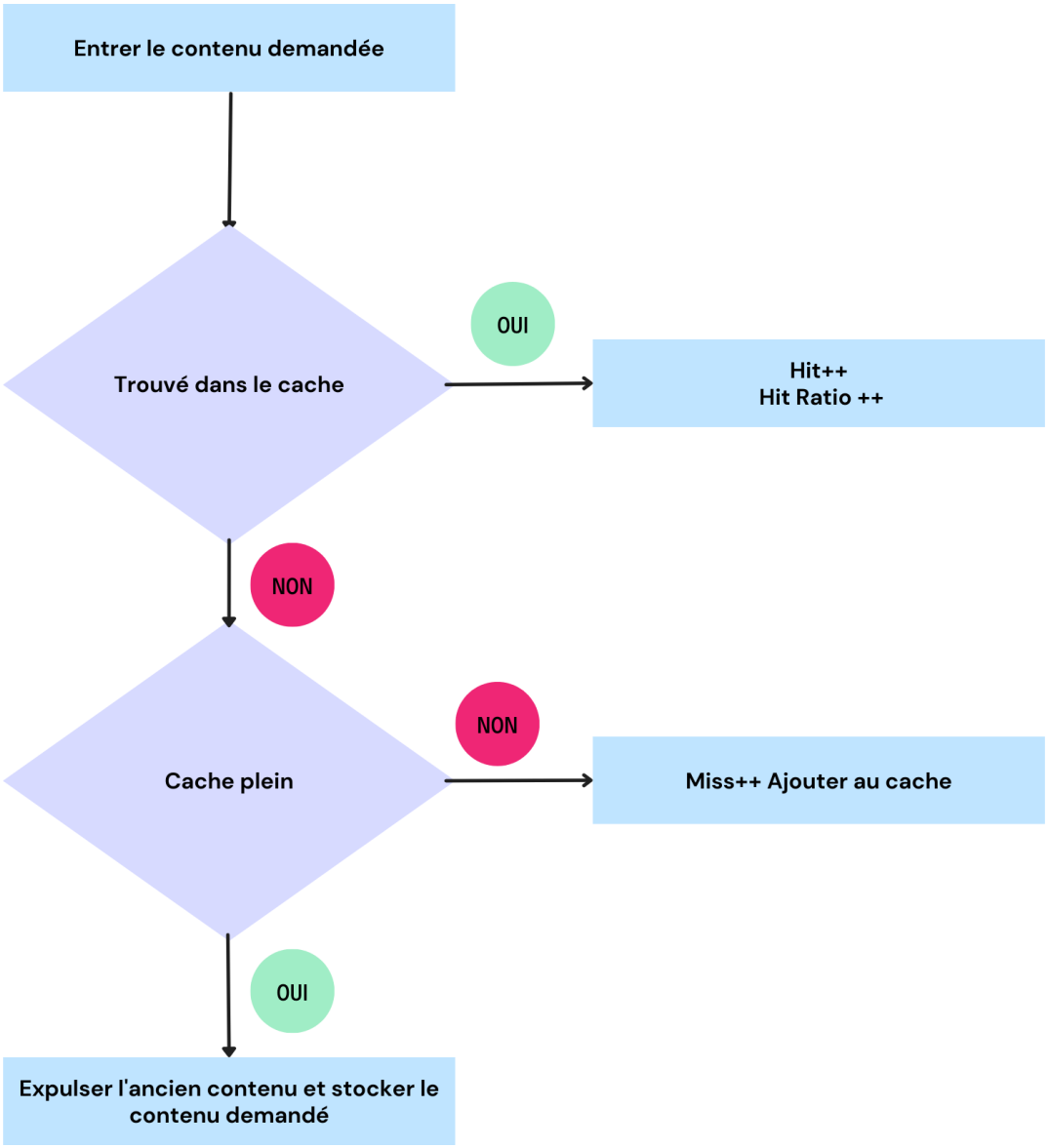


Figure 17 Processus de Remplacement de Cache

3.2.1. Politiques basées sur la récence des données

Ces politiques éliminent les données qui n'ont pas été accédées récemment, supposant que les données récemment accédées ont plus de chances d'être demandées à nouveau. Autrement dit, ces stratégies exploitent le principe de localité temporelle, qui stipule que les éléments qui ont été utilisés récemment sont plus susceptibles d'être utilisés à nouveau dans un futur proche. Elles offrent les avantages suivants :

- Simplicité : Leur mise en œuvre et leur compréhension sont aisées.
- Efficacité : Elles peuvent améliorer le taux de réussite du cache et réduire le temps de réponse.
- Flexibilité : Elles peuvent s'adapter à divers types de contenus et de trafic.

3.2.1.1. La politique LRU

Le LRU (Least Recently Used : Le moins récemment utilisé) est un algorithme de remplacement de cache qui fonctionne en remplaçant les données qui n'ont pas été accédées depuis longtemps. L'idée principale derrière LRU est de supposer que les données qui ont été accédées récemment ont plus de chances d'être accédées à nouveau dans un proche avenir. L'algorithme LRU maintient une liste ordonnée des données en fonction de leur dernier accès, de sorte que les données les moins récemment utilisées sont remplacées en premier lorsqu'une nouvelle donnée doit être mise en cache. LRU est largement utilisé en raison de sa simplicité et de sa capacité à exploiter la localité temporelle des accès aux données[30]. Voici l'algorithme de base pour la stratégie de remplacement LRU :

1. Initialiser une structure de données afin de suivre les contenus présents dans le cache ainsi que leurs temps d'accès. Cela pourrait être mis en œuvre en utilisant une combinaison d'une liste doublement chaînée et d'une table de hachage.
2. Chaque fois qu'un paquet de données est accédé :
    a. Si le paquet de données est déjà dans le cache, mettre à jour son temps d'accès avec le temps actuel.
    b. Si le paquet de données n'est pas dans le cache :
        i. Vérifier si le cache est plein.
        ii. Si le cache n'est pas plein, ajouter le paquet de données au cache avec son temps d'accès défini sur le temps actuel.
        iii. Si le cache est plein, évacuer le paquet de données le moins récemment utilisé (c'est-à-dire celui avec le plus ancien temps d'accès) du cache.
        iv. Ajouter le nouveau paquet de données au cache avec son temps d'accès défini sur le temps actuel.
3. Pour évacuer le paquet de données le moins récemment utilisé du cache :
    a. Parcourir la structure de données pour trouver le paquet de données avec le temps d'accès le plus ancien.
    b. Supprimer ce paquet de données du cache.

Voici l'algorithme LRU en pseudocode :

```
Initialiser le cache comme une structure de données appropriée (par exemple, une liste
doublement chaînée et une table de hachage).

Fonction AccéderAuPaquetDeDonnées(nomPaquet) :
    Si nomPaquet est dans le cache :
        Mettre à jour le temps d'accès de nomPaquet à l'heure actuelle.
    Sinon :
        Si le cache est plein :
            Évacuer le paquet de données le moins récemment utilisé du cache.
        Ajouter nomPaquet au cache avec son temps d'accès défini sur l'heure actuelle.
```

```
Fonction ÉvacuerPaquetDeDonnéesLeMoinsRécemmentUtilisé():
    Parcourir le cache pour trouver le paquet de données avec le temps d'accès le plus ancien.
    Supprimer ce paquet de données du cache.
```

Table 1 Algorithme de remplacement LRU[33]

Tableau 2 Exemple de l'algorithme LRU

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3
		1	1	1	3	3	3	2	2	2	2	2	2	2
Miss	Miss	Miss	Miss	Hit	Miss	Hit	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Hit

3.2.2. Politiques basées sur la fréquence d'accès aux données

Ces politiques éliminent les données moins fréquemment accédées, favorisant ainsi celles qui sont demandées plus fréquemment. Par conséquent, en privilégiant les éléments les plus fréquemment utilisés, ces stratégies augmentent la probabilité de trouver les contenus populaires dans le cache, réduisant ainsi le temps de latence.

3.2.2.1. La politique LFU

Dans cette stratégie, l'élément le moins fréquemment utilisé (Least Frequently Used) est remplacé en cas de conflit de cache. LFU se concentre sur la fréquence d'accès à un élément pour décider de son remplacement [27].

Principe de fonctionnement **:**

LFU maintient un compteur associé à chaque élément en cache pour suivre le nombre de fois qu'il a été accédé. Lorsqu'un nouvel élément doit être mis en cache et que l'espace est insuffisant, LFU sélectionne l'élément avec le compteur le plus bas, ce qui signifie qu'il a été le moins fréquemment utilisé. LFU se base sur le principe que les éléments qui ont été peu fréquemment accédés dans le passé sont moins susceptibles d'être utilisés à nouveau à l'avenir. Par conséquent, en supprimant les éléments les moins fréquemment utilisés, LFU vise à maximiser l'utilisation du cache en favorisant les éléments qui sont les plus demandés. LFU peut aider à réduire la redondance des données en favorisant les éléments qui sont fréquemment utilisés et en éliminant ceux qui ne le sont pas. Cela peut contribuer à améliorer l'efficacité de la mise en cache en favorisant les données les plus pertinentes et en évitant de surcharger le cache avec des éléments inutilisés. La mise en œuvre de LFU peut être relativement simple par rapport à d'autres stratégies plus complexes, car elle ne nécessite que le suivi du nombre de fois qu'un élément est accédé. Cependant, la gestion des compteurs et la décision de remplacement peuvent devenir plus complexes à mesure que la taille du cache augmente. LFU peut être adaptée à différents scénarios et types de données. Elle peut être particulièrement efficace dans les environnements où les modèles d'accès aux données ne changent pas fréquemment et où certains éléments sont

régulièrement plus utilisés que d'autres. Voici l'algorithme de base pour la stratégie de remplacement LFU :

1. Initialiser un cache avec une capacité maximale.
2. Maintenir une structure de données pour stocker les paquets ainsi que leurs fréquences d'accès.
3. Lorsqu'un nouveau paquet est accédé :
    a. Si le cache est plein, écarter le paquet ayant la fréquence d'accès la plus faible.
    b. Insérer le nouveau paquet avec fréquence d'accès initialisé à 1.
4. Lorsqu'un paquet existant est réaccordé :
    a. Incrémenter sa fréquence d'accès.
    b. Mettre à jour l'entrée la moins fréquemment utilisée si nécessaire.

Tableau 3 Exemple de l'algorithme LFU

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2
7	7	7	2	2	2	2	4	4	3	3	3	3	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	1	2
Miss	Miss	Miss	Miss	Hit	Miss	Hit	Miss	Miss	Miss	Hit	Hit	Hit	Miss	Miss

3.2.2.2. La politique GDSF

La politique GDSF (Greedy Dual Size Frequency) [34][35] vise à améliorer les performances du cache en considérant à la fois la taille et la fréquence des accès aux objets lors de la prise de décisions d'éviction.

Dans la politique GDSF, la clé de priorité Ki pour un objet i est calculée comme suit :

$$Ki = Fi * \frac{Ci}{Si} + L$$

Où :

- F_i : représente le comptage de fréquence de l'objet i, indiquant à quelle fréquence l'objet a été accédé.
- C_i : désigne le coût associé à l'ajout de l'objet i dans le cache.
- S_i : correspond à la taille de l'objet.
- L : est un facteur d'âge en cours qui est mis à jour chaque fois qu'un objet est remplacé, reflétant sa priorité dans la file de priorité.

En intégrant le comptage de fréquence Fi dans le calcul de la clé de priorité, la politique GDSF accorde une priorité plus élevée aux objets qui sont accédés plus fréquemment. Cela contribue à maintenir les objets populaires dans le cache pendant des périodes prolongées, ce qui peut améliorer les taux de réussite et réduire la pollution du cache. La politique GDSF est une politique de remplacement basée sur la fréquence qui vise à trouver un équilibre entre la taille de l'objet,

la fréquence d'accès et le coût pour optimiser les performances du cache. Elle permet une gestion plus efficace du cache et une amélioration globale des performances.

3.2.2.3. La politique LFU-Aging

LFU-Aging est une extension de LFU visant à prévenir la pollution du cache, qui survient lorsque le cache conserve des contenus populaires qui ne sont plus sollicités. Cette politique introduit l'utilisation d'un seuil : si la valeur de tous les compteurs dépasse ce seuil, ils sont tous divisés par deux [36].

3.2.2.4. La politique Window-LFU (WLFU)

La politique de remplacement de cache WLFU [37] est une variation de l'algorithme LFU qui prend en compte les caractéristiques de temporalité et de popularité des objets dans un cache. Voici quelques points clés concernant la politique de remplacement WLFU :

WLFU maintient une fenêtre coulissante contenant les requêtes les plus récentes et utilise l'algorithme LFU pour remplacer les objets dans le cache, en se basant sur leurs popularités dans cette fenêtre. En cas d'égalité de popularité entre plusieurs objets, l'algorithme LRU est utilisé pour identifier l'objet à remplacer. En ajustant la taille de la fenêtre de manière appropriée, cette politique de remplacement peut équilibrer efficacement les avantages de la temporalité et de la popularité pour améliorer le taux de réussite du cache. Ainsi, WLFU combine à la fois la temporalité et la popularité des accès aux objets, ce qui peut conduire à des performances supérieures par rapport à d'autres algorithmes de remplacement de cache.

Voici l'algorithme WLFU en pseudocode :

```
Variables :
  Cache : Structure de données pour stocker les éléments du cache
  WindowSize : Taille de la fenêtre temporelle en unités de temps (par exemple, heures)
  ElementFrequency : Dictionnaire pour stocker les fréquences d'accès des éléments dans la
fenêtre temporelle

Fonction Window_LFU_Insert(element)
  Si Cache est plein:
    Élément_à_évincer = Trouver_élément_à_évincer()
    Supprimer Élément_à_évincer de Cache

  Ajouter element à Cache
  Mettre à jour la fréquence d'accès de element dans ElementFrequency

Fin Fonction

Fonction Trouver_élément_à_évincer()
  MinimumFrequency = MAX_INT
  Élément_à_évincer = NULL

  Pour chaque élément dans Cache
    Si ElementFrequency[element] < MinimumFrequency
      MinimumFrequency = ElementFrequency[element]
      Élément_à_évincer = element

  Retourner Élément_à_évincer

Fin Fonction

Fonction Mettre_à_jour_fenêtre_temporelle()
  Supprimer les accès obsolètes de ElementFrequency basés sur WindowSize
```

```
  Mettre à jour la fenêtre temporelle en fonction de l'heure actuelle

Fin Fonction
```

Figure 18 : Algorithme de remplacement Window LFU [37]

3.2.3. Politiques basées sur la taille des données

Les stratégies qui se basent sur la taille prennent en considération la taille des données stockées et peuvent éliminer celles-ci en fonction de l'espace qu'elles occupent dans le cache.

3.2.3.1. La politique LRU-Min

LRU-MIN commence par vérifier s'il existe des données de taille égale ou supérieure à celle de la donnée entrante. Si tel est le cas, il en choisit une selon l'algorithme LRU. Sinon, il examine toutes les données dont la taille est supérieure à la moitié de celle de la donnée entrante. S'il en trouve, il en choisit une selon l'algorithme LRU. Si ce n'est pas le cas, il réitère le processus en utilisant un quart de la taille de la donnée, et ainsi de suite [38].

3.2.3.2. La politique SIZE

Cette approche substitue la donnée ayant la plus grande taille. En cas d'égalité de taille entre les contenus, la méthode LRU est employée [39].

3.2.3.3. La politique Greedy Dual Size (GD-Size)

La politique GD-Size [36] est une stratégie utilisée pour déterminer quels objets doivent être expulsés lorsqu'il y a une contrainte d'espace et de nouveaux objets doivent être stockés.

Dans la politique GD-Size, la clé de priorité Ki pour un objet i est calculée comme suit :

$$Ki = \frac{Ci}{Si} + L$$

Où :

- Ci : Représente le coût nécessaire pour récupérer l'objet depuis sa source d'origine. Un coût plus élevé signifie qu'il est moins souhaitable de rechercher l'objet à nouveau.
- Si : Indique simplement la quantité d'espace de stockage que l'objet occupe dans le cache. Les objets plus volumineux prennent plus de place.
- L : Introduit un facteur d'ancienneté. Il commence à 0 et est mis à jour avec la clé de priorité de l'objet expulsé lors du remplacement du cache. Cela garantit que les objets récemment expulsés ont une pénalité d'ancienneté plus élevée, ce qui les rend moins susceptibles d'être expulsés à nouveau immédiatement.

Alors, lorsque l'espace du cache devient restreint, la politique GD-Size identifie l'objet ayant la clé de priorité (Ki) la plus basse. En expulsant l'objet avec le Ki le plus bas, GD-Size vise à :

1. Minimiser le coût de récupération des objets.
2. Maximiser la quantité de données stockées dans le cache.
3. Éviter l'expulsion et la récupération fréquentes du même objet.

3.2.4. Politiques basées sur les fonctions

Ces stratégies utilisent des fonctions spécifiques pour évaluer les données présentes dans le cache et prendre des décisions d'éviction en fonction de cette évaluation.

3.2.4.1. La politique LRFU

La politique de remplacement LRFU[40] est une stratégie qui combine les techniques LRU et LFU. Dans LRFU, la décision d'éviction est basée sur une combinaison de la récence de données et de la fréquence d'accès à ces données. Chaque élément de contenu dans le cache se voit attribuer un score CRF en fonction de sa récence et de sa fréquence d'accès. Bien que la formule exacte de calcul de ce score puisse varier, elle prend généralement en compte des facteurs tels que le temps écoulé depuis le dernier accès et le nombre total d'accès. Les éléments de contenu avec les scores les plus bas sont plus susceptibles d'être rejetés du cache.

Le score CRF est calculé comme suit [41] :

$$\mathbf{CRF_{Current}\ (data) = F(0) + F(TimeCurrent - TimeLast) * CRF_{Last}\ (data)}$$

Où :

TimeCurrent : le temps d'accès actuel, **TimeLast:** le temps du dernier accès. La distance entre **TimeCurrent** et **TimeLast** représente l'aspect récence et la récursivité avec l'addition représente l'aspect fréquence.

$$F(X) = (\frac{1}{2})^{\lambda x}$$

Si λ=0, alors F(x)=1 et LRFU devient LFU.

Si λ=1, alors F(x)=$(\frac{1}{2})^{x}$, et LRFU devient LRU.

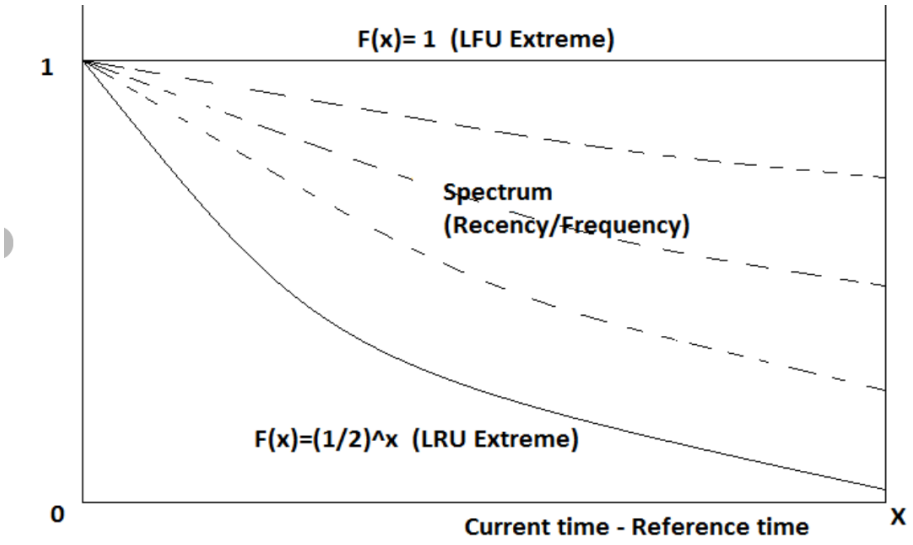


Figure 19 : Les spectres de LRFU selon la fonction F (X) [40]

3.2.4.2. La politique Weighted Greedy-Dual-Size-Frequency (WGDSF)

L'algorithme de remplacement WGDSF est une méthode de gestion de cache qui prend en compte à la fois la taille des données stockées dans le cache, leur fréquence d'accès et leur importance relative pour décider quelles données doivent être remplacées lorsqu'une nouvelle donnée doit être placée dans un cache plein[34].

Voici comment fonctionne l'algorithme WGDSF :

a. On assigne un poids à chaque donnée en mémoire en fonction de sa taille et de sa fréquence d'accès. Les données de taille plus petite et moins fréquemment utilisées se voient attribuer un poids plus faible, tandis que les données de taille plus grande et plus fréquemment utilisées ont un poids plus élevé.
b. Lorsqu'il est nécessaire de remplacer une donnée en mémoire, l'algorithme WGDSF sélectionne la donnée à remplacer en fonction de son poids. Plus le poids d'une donnée est faible, plus elle est susceptible d'être remplacée.
c. Si plusieurs données ont le même poids, l'algorithme utilise la politique LRU pour choisir quelle donnée doit être remplacée. Cela signifie qu'il examine les dates d'accès des données concernées et remplace la donnée qui n'a pas été accédée depuis la plus longue période de temps.
d. Lorsqu'une donnée est placée dans le cache, son poids est réinitialisé à zéro, ce qui signifie qu'elle ne sera pas considérée comme candidate à la substitution lors de la prochaine itération de l'algorithme.
e. L'algorithme WGDSF répète ce processus jusqu'à ce qu'il n'y ait plus de données à remplacer.

Le fonctionnement de l'algorithme WGDSF est détaillé comme suit :

1. L'algorithme reçoit une demande pour la donnée j.

2. Si la donnée j est déjà présente dans le cache :

   a. Incrémente le compteur de fréquence d'accès (fr(j)) de la donnée j.
   b. Calcule le score de remplacement H(j) en fonction de la taille SC(j), du poids de l'importance WDT(j), et de la fréquence WTF(j) de la donnée j.
   c. Renvoie la donnée j.

3. Sinon :

   d. Récupère la donnée j à partir de la source de données.
   e. Initialise le compteur de fréquence d'accès fr(j) à 1.
   f. Calcule le score de remplacement H(j) en fonction de la taille SC(j), du poids de l'importance WDT(j), et de la fréquence WTF(j) de la donnée j.
   g. Ajoute la taille de la donnée j à la variable Used, qui maintient la taille totale des données dans le cache.

4. Si la taille totale des données dans le cache (Used) est inférieure à la taille totale du cache (Total), alors ajoute simplement la donnée j au cache.

5. Sinon :

   h. Recalcule les valeurs de H pour toutes les données présentes dans le cache.
   i. Sélectionne les données {m1, m2, ..., mk} ayant les valeurs de H les plus basses et qui satisfont les conditions suivantes :
      i. La somme des tailles de ces données est inférieure ou égale à la taille totale du cache (Total).
      ii. Les valeurs de H sont dans un ordre croissant.
   j. Si la donnée j est déjà présente dans le cache, alors la supprimer.
   k. Sinon, évacue la donnée ayant le score le plus bas parmi {m1, m2, ..., mk}, libère de l'espace en mettant à jour la variable Used et ajoute la donnée j au cache.

6. Met à jour la variable L avec la valeur minimale des H des données restantes dans le cache après le remplacement.

Cet algorithme prend en compte à la fois la taille, la fréquence d'accès et l'importance des données pour décider quelles données doivent être stockées dans le cache et lesquelles doivent être évacuées lorsque le cache est plein.

Voici le pseudo-algorithme de WGDSF :

```
Fonction WGDSF (document j)
    Si j est dans le cache alors
        fr(j) = fr(j) + 1  // Mettre à jour la fréquence d'accès
        H(j) = L + SC(j) * WDT(j) * WTF(j)  // Calcul du score de remplacement
        Renvoyer j
    Sinon
        Obtenir j à partir de la source de données
        fr(j) = 1  // Initialiser la fréquence d'accès
        H(j) = L + SC(j) * WDT(j) * WTF(j)  // Calcul du score de remplacement
```

```
        Used = Used + Taille(j)  // Mettre à jour l'espace utilisé dans le cache
        Si Used < Total alors
            Charger j dans le cache
        Sinon
            Recalculer les valeurs H de tous les documents en cache
            Choisir les documents {m1, m2, ..., mk} avec les valeurs H les plus basses, qui
satisfont les conditions suivantes :
                Used - Somme_taille(m1, m2, ..., mk) ≤ Total
                H(m1) ≤ H(m2) ≤ ... ≤ H(mk)
            Si j est dans {m1, m2, ..., mk} alors
                Supprimer j
            Sinon
                L = minH = H(mk)
                Used = Used - Somme_taille(m1, m2, ..., mk)
                Éviction de {m1, m2, ..., mk} avec la valeur H la plus basse, et charger j dans
le cache
            Fin Si
        Fin Si
    Fin Si
Fin Fonction
```

Figure 20 Algorithme de remplacement WGDSF [34]

3.2.4.3. La politique Window-LRFU (WLRFU)

La politique Window-LRFU[41] [42]englobe les politiques LRU et Window-LFU par un paramètre réglable λ ∈ [0, 1] faisant référence à l'idée de la politique LRFU. Comme Window-LFU, la stratégie Window-LRFU utilise une fenêtre glissante de taille fixe pour enregistrer l'historique d'accès récent aux données. Lorsqu'une donnée est référencée, son identifiant est ajouté au début de la fenêtre, et le dernier élément de la fenêtre est expulsé.

En se basant sur cette fenêtre glissante, la stratégie Window-LRFU calcule, pour chaque donnée mise en cache x, le même score que celui adoptée par la politique LRFU. Ici, Le score Window-CRF (CRFW) est calculée comme suit :

$$CRFW(x) = \begin{cases} \sum_{i}^{k} F(S_i),\ si\ x\ est\ dans\ la\ fenetre. \\ 0, si\ x\ n'est\ pas\ dans\ la\ fenetre. \end{cases}$$

Où :

$$F(X) = (\frac{1}{2})^{\lambda x}$$

k est le nombre de fois que la donnée x apparaît dans la fenêtre et S_i ($1 \le i \le k$) désigne la distance entre le début de la fenêtre et la k-ème position référencée de la donnée x dans la fenêtre.

Par exemple, supposons qu'une donnée x apparaisse quatre fois dans une fenêtre de 12 données aux positions 2, 5, 7 et 10 (Figure 21). Donc, la valeur CRFW de la donnée x peut être calculée comme suit : CRFW(x) = F(2) + F(5) + F(7) + F(10). Lorsque le cache devient plein, les données ayant le plus faible score sont rejetées pour faire de la place aux nouvelles entrées.

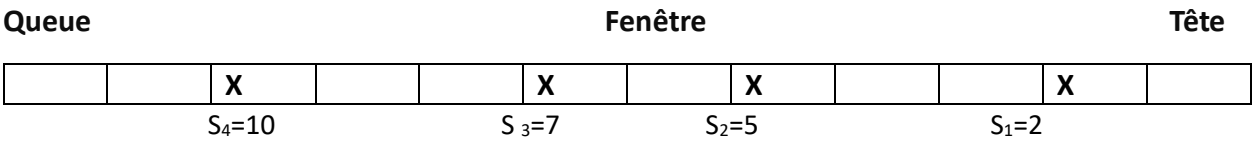


Figure 21 Une fenêtre de 12 donnée. La donnée x se trouve aux positions 2, 5, 7 et 10 dans la fenêtre à l'heure actuelle du système.[41]

3.2.5. Politiques basées sur le partitionnement

3.2.5.1. La politique Adaptive Replacement Cache (ARC)

L'algorithme ARC [43] est une technique utilisée pour améliorer les performances de la mise en cache. Il vise à optimiser l'utilisation du cache en adaptant dynamiquement la taille des listes de cache en fonction du comportement d'accès aux données. L'ARC maintient deux listes, et ajuste leurs tailles en fonction des accès récents aux données. Cela permet à l'algorithme ARC de s'adapter aux changements dans les modèles d'accès aux données et de fournir des performances de cache optimales dans divers scénarios d'utilisation.

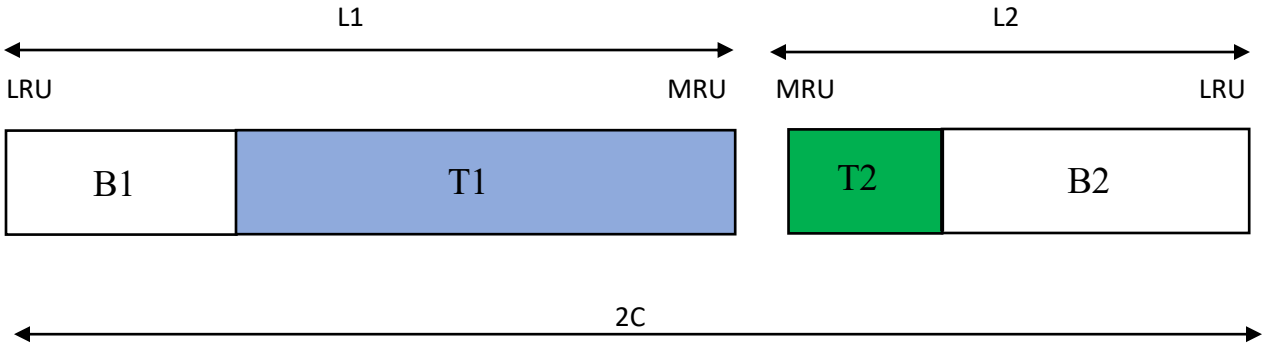


Figure 22 Algorithme ARC[44]

Pour améliorer les performances, ARC utilise deux listes de cache (T1 et T2) avec deux tampons de prédiction (B1 et B2) : T1 pour les données récupérées une fois, T2 pour les données récupérées au moins deux fois. B1 est assigné pour gérer le contenu récemment évincé du cache T1, tandis que B2 sert à gérer le contenu récemment effacé du cache T2. Ainsi, les tampons B1 et B2 contiennent les données qui ont été récemment rejetées et qui sont prévues comme étant les plus susceptibles d'être accédées dans un avenir proche. Basé sur leurs taux de succès, ARC ajuste dynamiquement ses listes de cache (T1 et T2) en réponse aux changements dans la distribution des données. En d'autres termes, si des données précédemment extraites de la liste T1 sont revisitées, ARC étend la portée de la liste, ressemblant au comportement de LRU. De plus, lorsque des données récemment supprimées de la liste T2 sont de nouveau sollicitées, ARC agrandit cette liste, montrant un comportement plus proche de celui de LFU. De cette manière, ARC combine de manière ajustable la récence et la fréquence en prenant en compte l'historique récent des expulsions.

3.2.5.2. La politique Segmented-LRU (SLRU)

L'algorithme SLRU [45] , une variante de l'algorithme LRU, est utilisé pour déterminer quelles données garder en mémoire cache et lesquelles retirer lorsque la capacité de la mémoire cache est atteinte. Son concept principal consiste à diviser la mémoire cache en deux segments distincts :

1. **Le segment protégé :** C'est là où les données fréquemment utilisées sont stockées pour les protéger contre l'éviction du cache.
2. **Le segment probatoire :** C'est l'endroit où les nouvelles données, qui n'ont jamais été accédées, sont initialement stockées.

Lorsqu'une donnée est accédée, l'algorithme SLRU commence par vérifier si elle se trouve dans le segment protégé. Si tel est le cas, la donnée est déplacée vers le début MRU (l'élément le plus récemment utilisé) du segment protégé, car elle est considérée comme importante et susceptible d'être réutilisée. Si la donnée n'est pas dans le segment protégé, cela signifie qu'elle est probablement dans le segment probatoire. Si elle se trouve dans ce dernier, elle est alors déplacée vers la fin LRU (l'élément le moins récemment utilisé) du segment protégé.
Lorsque de nouvelles données doivent être insérées dans le segment protégé mais que celui-ci est plein, l'algorithme retire la donnée la moins récemment utilisée (LRU) du segment protégé et la déplace vers le début du segment probatoire, libérant ainsi de l'espace pour la nouvelle donnée. Cette approche permet d'éviter que le cache soit saturé avec des données peu utiles, en maintenant les données fréquemment utilisées dans le segment protégé où elles sont moins susceptibles d'être évacuées.
L'algorithme SLRU améliore les performances de la gestion de cache en séparant les données fréquemment utilisées des données moins utilisées, ce qui permet de mieux utiliser la capacité limitée de la mémoire cache.

Conclusion

Dans ce chapitre, nous avons exploré le concept de la mise en cache dans les réseaux NDN, mettant en lumière son rôle central dans l'optimisation des performances, de l'efficacité et de la fiabilité du réseau. À travers une analyse approfondie des différentes stratégies de mise en cache associées à NDN, nous avons mis en évidence les avantages comparatifs de cette architecture par rapport aux réseaux traditionnels basés sur TCP/IP.

Nous avons souligné l'importance de la mise en cache dans les réseaux NDN, en insistant sur son rôle dans la réduction de la congestion, l'amélioration de la distribution de contenu, et la simplification de la configuration du réseau. De plus, nous avons examiné en détail le processus de mise en cache dans NDN, notamment les mécanismes associés à la gestion efficace des tables PIT et FIB.

Enfin, nous avons étudié en profondeur les stratégies de placement et de remplacement du cache, en analysant leurs mécanismes, avantages et limites. Des stratégies basées sur la récence et la fréquence aux celles basées sur la taille et les fonctions, nous avons exploré un large éventail de méthodes utilisées pour optimiser les ressources de cache dans les réseaux NDN.

Chapitre 3
Simulation et interprétation

Chapitre 3 Simulation et Interprétation

INTRODUCTION

Dans l'ère du numérique où les données occupent une place prépondérante, l'optimisation des systèmes de mise en cache revêt une importance cruciale. Les réseaux NDN ont émergé comme une solution prometteuse pour relever les défis liés à la distribution efficace de données à grande échelle. Au cœur de cette architecture réside le mécanisme de mise en cache, qui joue un rôle déterminant dans la réduction de la latence, l'amélioration du débit et la diminution de la charge sur les serveurs de contenu.

Dans ce chapitre, nous analysons les performances de diverses stratégies de remplacement de cache dans le cadre des réseaux NDN, en utilisant l'outil de simulation ccnSim. Nous nous concentrons notamment sur les algorithmes LRU, LFU, LRFU et RANDOM. En utilisant des métriques de performance telles que le taux de réussite du cache et la latence, notre objectif est d'identifier les stratégies les plus performantes, de comprendre les caractéristiques inhérentes à chaque approche et ainsi, de guider leur choix dans des déploiements réels de réseaux NDN.

1. Outils de simulation

Pour la réalisation de notre simulation, nous avons utilisé le simulateur ccnSim qui est un simulateur de CCN écrit en C++ et utilisé sous le framework Omnet++. Ces applications ont été installées sur une station de travail HP EliteBook 830 G5 avec un processeur Intel(R) Core(TM) i5-8350U et 16 Go de RAM, fonctionnant sur la plate-forme Windows 11 Pro.

Figure 23 Logo du omnet++

1.1. ccnSim

ccnSim [46] est une extension d'OMNeT++[47] spécialement conçue pour modéliser, simuler et analyser les performances des réseaux CCN et NDN. ccnSim comprend des modèles prédéfinis pour divers composants de réseau tels que les routeurs, les producteurs de contenu et les consommateurs. Il propose également des stratégies de mise en cache et de routage

préconfigurées. De plus, il permet d'exécuter des simulations de réseaux à grande échelle tout en utilisant de manière modérée les ressources CPU et mémoire.

Les commandes suivantes servent à installer ccnSim sur omnet++ :

cd $CCNSIM_DIR

cp ./patch/omnet-5x/ctopology.h $OMNET_DIR/include/omnetpp

cp ./patch/omnet-5x/ctopology.cc $OMNET_DIR/src/sim

cd $OMNET_DIR && make && cd $CCNSIM_DIR

./scripts/makemake.sh

make

1.1.1. La configuration et lancement de simulation

Les étapes suivantes décrivent comment lancer notre simulateur :

1. Ouvrez un terminal et tapez ''omnetpp'' pour lancer OMNeT++.
2. Accédez au dossier ccnSim dans l'explorateur de projet.
3. Allez dans le dossier network pour choisir et modifier une topologie.
4. Ouvrez le fichier de configuration ''ED_TTL-omnetpp.ini'', configurez les paramètres de simulation nécessaires, puis lancez la simulation.

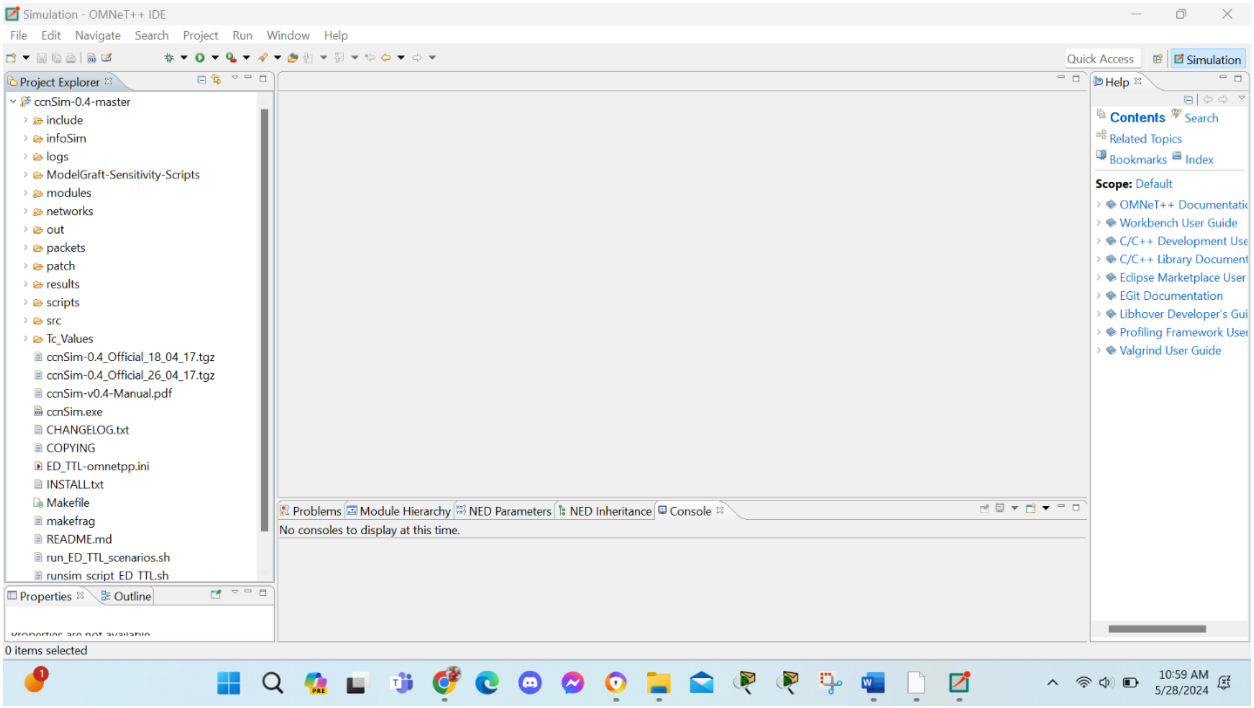


Figure 24 Interface OMNet++

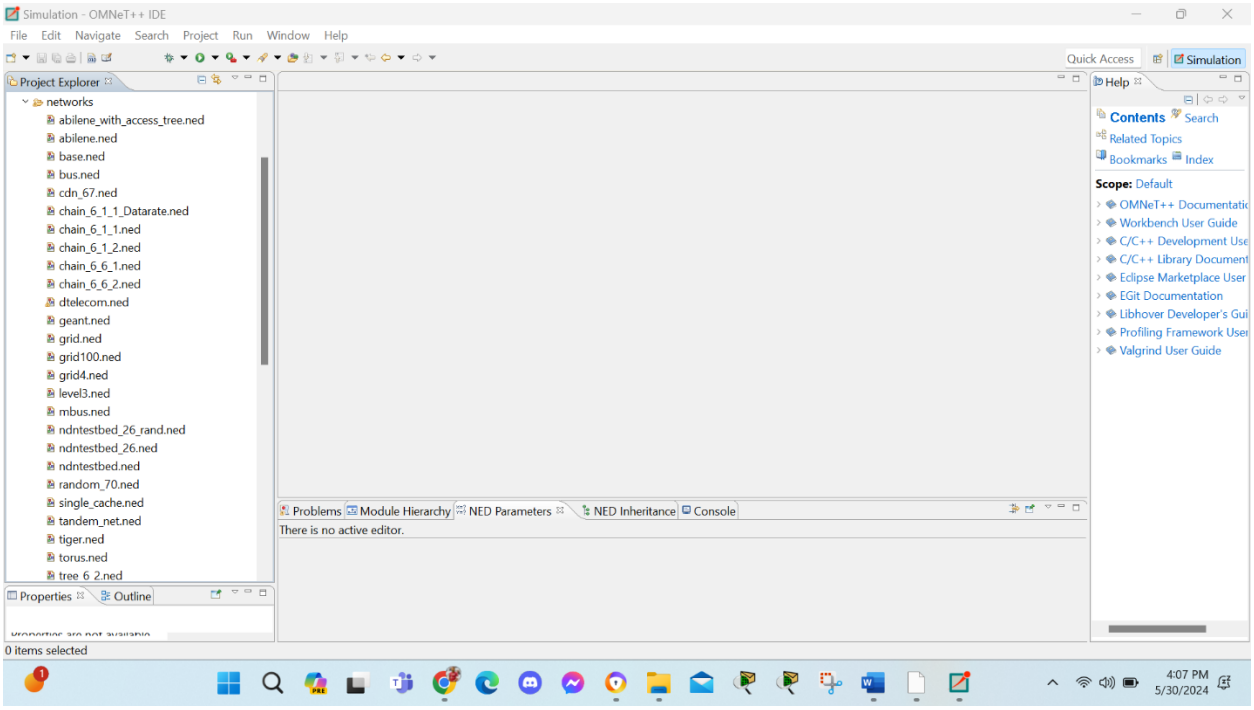


Figure 25 Les differentes topologies dans le dossier "networks"

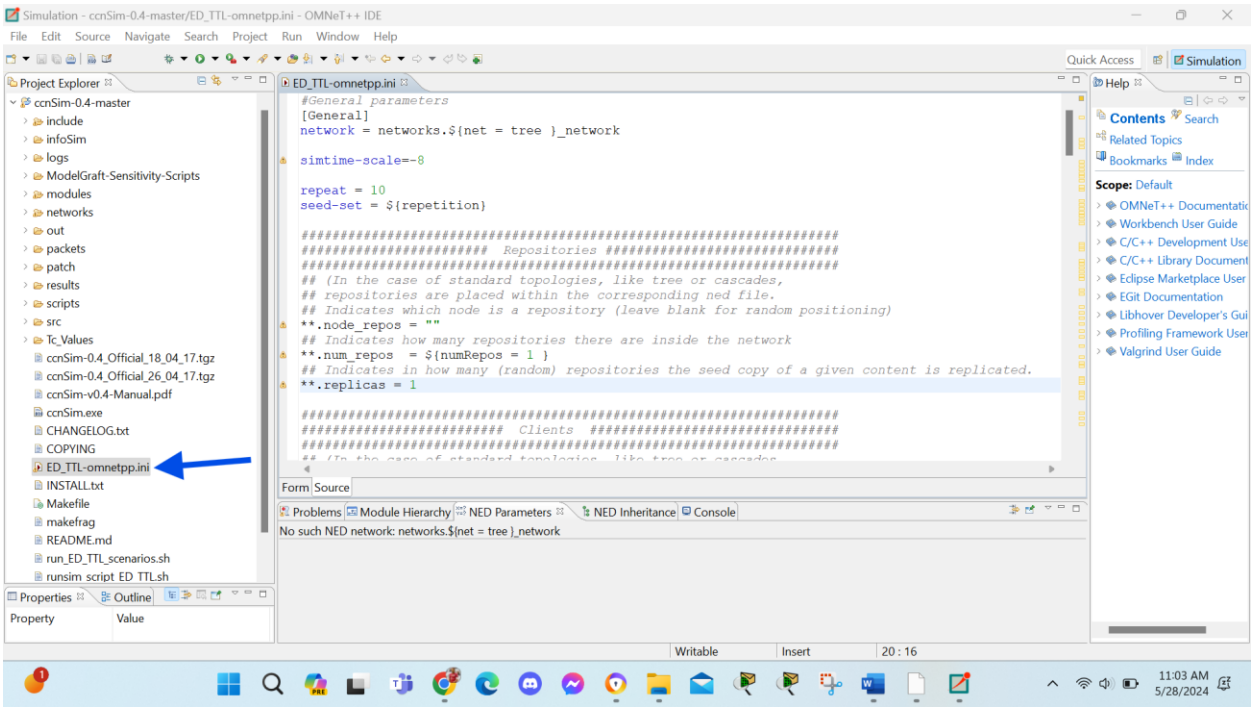


Figure 26 Fichier ED_TTL-omnetpp.ini pour ajouter les paramètres et lancer la configuration

1.1.2. Ajout des stratégies de remplacements

Le simulateur ccnSim offre la possibilité d'ajouter de nouveaux algorithmes de mise en cache à son environnement. Ainsi, nous avons développé et inclus deux nouvelles stratégies de

remplacement de cache, LFU et LRFU, en complément des stratégies LRU et RANDOM prédéfinies. Voici la démarche à suivre pour effectuer cet ajout :

1. Tout d'abord, créez le fichier, par exemple lfu.cc dans le dossier src, et lfu.h dans le dossier include.
2. Ensuite, ajoutez le nom de la stratégie dans le fichier cache.ned situé dans le module cache.
3. Enfin, ajoutez-le dans le makefile de ccnsim.

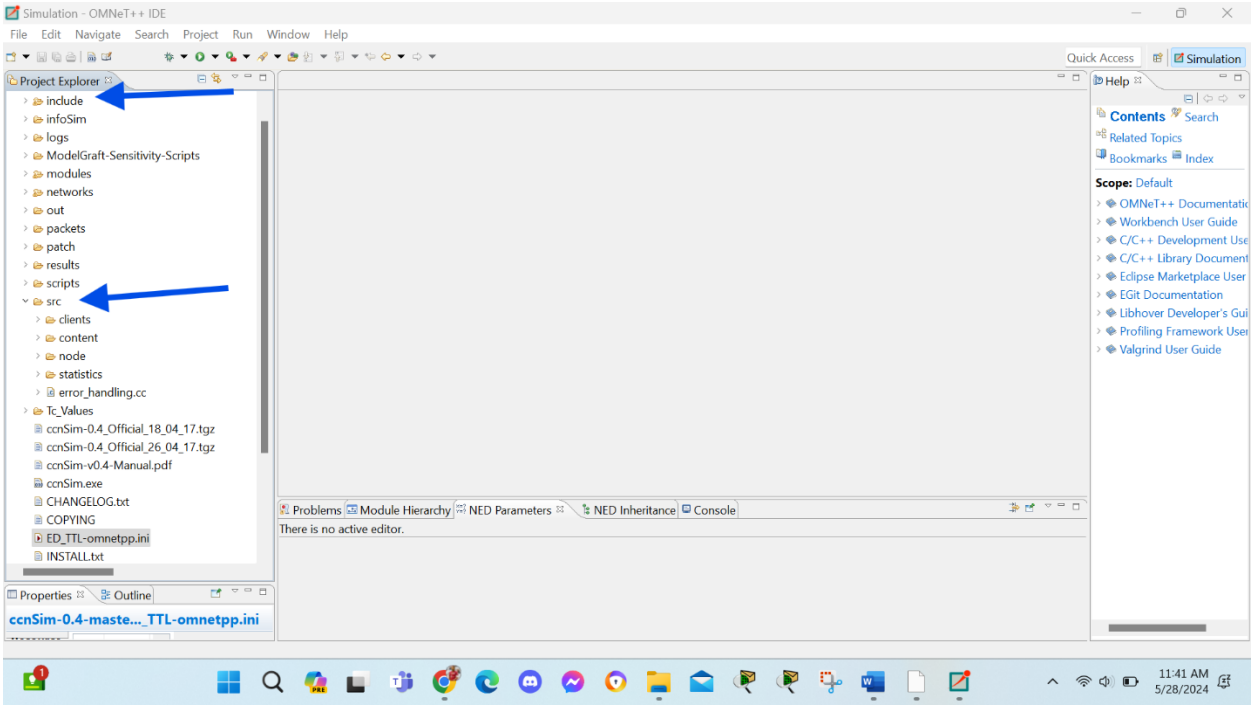


Figure 27 Les dossiers dans lesquels nous ajoutons les nouvelles stratégies

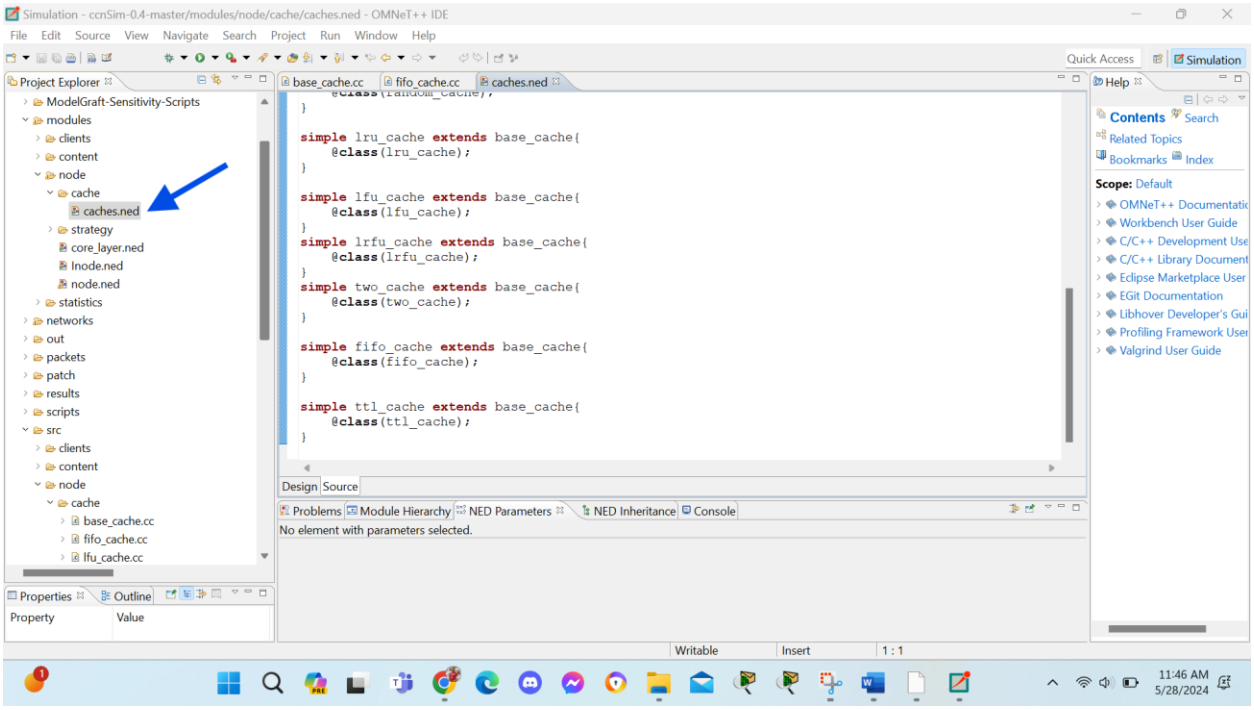


Figure 28 L'ajout de la stratégie dans le fichier cache.ned

Les stratégies de remplacement de cache que nous avons utilisées dans la simulation sont :

- LRU (Le moins récemment utilisé) basée sur le temps d'accès. Cette politique remplace le contenu le plus ancien.
- LFU (Le moins fréquemment utilisé) basé sur la fréquence d'accès. Elle remplace le contenu le moins souvent accédé.
- LRFU combine les algorithmes LRU et LFU. Elle remplace le contenu ayant la plus faible valeur de CRF. Ce dernier utilise à la fois le facteur de récence de LRU et la caractéristique de fréquence de LFU.
- RANDOM sélectionne de manière aléatoire une donnée à remplacer.

En simulant ces quatre stratégies de remplacement dans différents scénarios, nous pourrons évaluer leurs performances en termes d'efficacité du cache et comprendre par conséquent leur comportement.

Conclusion Générale

L'évolution rapide des besoins de communication et la croissance exponentielle du trafic Internet mettent en évidence les limites de l'architecture IP actuelle. Pour y remédier, NDN propose une architecture centrée sur le contenu, offrant une distribution plus efficace, une meilleure évolutivité, une résilience accrue aux pannes, ainsi qu'une sécurité et une confidentialité améliorées.

Les mécanismes de mise en cache jouent un rôle essentiel dans l'optimisation des performances du réseau NDN. Les études et simulations dans ce contexte montrent qu'un choix judicieux de stratégie de remplacement est crucial pour améliorer les taux de réussite du cache, réduire les délais de récupération des données et, par conséquent, maximiser l'efficacité globale du réseau.

Ainsi, les résultats de notre étude contribueront à guider les futures décisions de conception et de déploiement des réseaux NDN, aboutissant à une expérience utilisateur améliorée et à une utilisation plus efficace des ressources réseau.

En conclusion, l'approche NDN offre une perspective prometteuse pour l'avenir de l'Internet, mais la transition vers ce nouveau modèle nécessitera une adoption progressive et des efforts concertés pour assurer une interopérabilité et une évolutivité optimales.

Bibliographies

[1] J. Roberts, “The clean-slate approach to future Internet design: A survey of research initiatives,” *Ann Telecommun*, vol. 64, pp. 271–276, Jan. 2009, doi: 10.1007/s12243-009-0109-y.

[2] A. V Vasilakos, Z. Li, G. Simon, and W. You, “Information centric network: Research challenges and opportunities,” *Journal of Network and Computer Applications*, vol. 52, pp. 1–10, 2015, doi: https://doi.org/10.1016/j.jnca.2015.02.001.

[3] G. Xylomenos *et al.*, “A Survey of Information-Centric Networking Research,” *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014, doi: 10.1109/SURV.2013.070813.00063.

[4] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, “Networking Named Content,” *Commun. ACM*, vol. 55, pp. 117–124, Jan. 2012, doi: 10.1145/1658939.1658941.

[5] V. Jacobson *et al.*, “Named Data Networking Next Phase (NDN-NP) Project May 2014 - April 2015 Annual Report,” 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:34153098

[6] T. Koponen *et al.*, “A data-oriented (and beyond) network architecture,” *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 181–192, Aug. 2007, doi: 10.1145/1282427.1282402.

[7] C. Dannewitz, D. Kutscher, B. Ohlman, S. Farrell, B. Ahlgren, and H. Karl, “Network of Information (NetInf) – An information-centric networking architecture,” *Comput Commun*, vol. 36, no. 7, pp. 721–735, 2013, doi: https://doi.org/10.1016/j.comcom.2013.01.009.

[8] B. Leiner *et al.*, “A Brief History of the Internet,” *Computer Communication Review*, vol. 39, pp. 22–31, Jan. 2009, doi: 10.1145/1629607.1629613.

[9] T. Borgohain, U. Kumar, and S. Sanyal, “Survey of Security and Privacy Issues of Internet of Things,” *International Journal of Advanced Networking and Applications*, vol. 6, pp. 2372–2378, Feb. 2015.

[10] R. Smith, A. Eberlein, and F. Halsall, “An Overview Of Future Routing Technologies,” Feb. 1999.

[11] G. Huston, “Quality of Service on the Internet: Fact, Fiction, or Compromise?,” 2003. [Online]. Available: https://api.semanticscholar.org/CorpusID:2945323

[12] C. Guimarães, J. Quevedo, R. Ferreira, D. Corujo, and R. L. Aguiar, “Exploring interoperability assessment for Future Internet Architectures roll out,” *Journal of Network and Computer Applications*, vol. 136, pp. 38–56, 2019, doi: https://doi.org/10.1016/j.jnca.2019.04.008.

[13] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A survey of information-centric networking,” *IEEE Communications Magazine*, vol. 50, no. 7, pp. 26–36, 2012, doi: 10.1109/MCOM.2012.6231276.

[14] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. Braynard, “Networking named content,” *Commun ACM*, vol. 55, pp. 117–124, 2012, [Online]. Available: https://api.semanticscholar.org/CorpusID:528`95555

[15] L. Zhang *et al.*, “Named data networking,” *Comput. Commun. Rev.*, vol. 44, pp. 66–73, 2014, [Online]. Available: https://api.semanticscholar.org/CorpusID:8317810

[16] Amar ABANE, “Mise en œuvre des concepts NDN et IoT dans le domaine des Smart Cities : Exemple du parking intelligent,” Mémoire de Fin d’Études de MASTER ACADÉMIQUE, UNIVERSITÉ MOULOUD MAMMERI DE TIZI-OUZOU, TIZI-OUZOU, 2016.

[17] D. Saxena, V. Raychoudhury, N. Suri, C. Becker, and J. Cao, “Named Data Networking: A survey,” *Comput Sci Rev*, vol. 19, pp. 15–55, 2016, doi: https://doi.org/10.1016/j.cosrev.2016.01.001.

[18] H. Htet Hlaing, M. Mambo, and Y. Funamoto, “Secure Content Distribution with Access Control Enforcement in Named Data Networking,” *Sensors*, vol. 21, Feb. 2021, doi: 10.3390/s21134477.

[19] B. A. Amira and B. Djamila, “La sécurité des communications dans les NDN (Named Data Networks),” Mémoire de Master, Université Amar Telidji Laghouat, Laghouat, 2016.

[20] J. Ran, N. Lv, D. Zhang, Y. yuan Ma, and Z. Xie, “On Performance of Cache Policies in Named Data Networking,” 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:58762371

[21] “Named Data Networking.” Accessed: Feb. 11, 2024. [Online]. Available: https://named-data.net/project/archoverview/

[22] C. Pu, “Pro NDN : MCDM-Based Interest Forwarding and Cooperative Data Caching for Named Data Networking,” *Journal of Computer Networks and Communications*, vol. 2021, pp. 1–16, Mar. 2021, doi: 10.1155/2021/6640511.

[23] M. N. D. Satria, F. H. Ilma, and N. R. Syambas, “Performance comparison of named data networking and IP-based networking in palapa ring network,” in *2017 3rd International Conference on Wireless and Telematics (ICWT)*, 2017, pp. 43–48. doi: 10.1109/ICWT.2017.8284136.

[24] A. Barakabitze and T. Xiaoheng, “Caching and data routing in Information Centric Networking (ICN): the Future Internet Perspective,” : *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. Vol.4, pp. 8–20, Feb. 2014.

[25] S. BELGOUMENE and F. SACKO, “Etude et simulation d’un réseau NDN,” MEMOIRE DE MASTER, UNIVERSITE IBN KHALDOUN, Tiaret, 2023.

[26] S. Al-Ahmadi, “A New Efficient Cache Replacement Strategy for Named Data Networking,” *International journal of Computer Networks & Communications*, vol. 13, pp. 19–35, Mar. 2021, doi: 10.5121/ijcnc.2021.13502.

[27] E. Silva, J. Macedo, and A. Costa, “NDN Content Store and Caching Policies: Performance Evaluation,” *Computers*, vol. 11, p. 37, Feb. 2022, doi: 10.3390/computers11030037.

[28] M. Hosseinzadeh, N. Moghim, S. Taheri, and N. Gholami, “A new cache replacement policy in named data network based on FIB table information,” *Telecommun Syst*, pp. 1–12, May 2024, doi: 10.1007/s11235-024-01140-7.

[29] M. Alkhazaleh, S. Aljunid, and N. Sabri, “A COMPREHENSIVE SURVEY OF INFORMATION- CENTRIC NETWORK: CONTENT CACHING STRATEGIES PERSPECTIVE,” *Journal of Critical Reviews*, vol. 7, pp. 2272–2287, Feb. 2020, doi: 10.31838/jcr.07.04.359.

[30] N. Laoutaris, H. Che, and I. Stavrakakis, “The LCD interconnection of LRU caches and its analysis,” *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, 2006, doi: https://doi.org/10.1016/j.peva.2005.05.003.

[31] H. Wu, J. Li, J. Zhi, Y. Ren, and L. Li, “Design and Evaluation of Probabilistic Caching in Information-Centric Networking,” *IEEE Access*, vol. 6, pp. 32754–32768, 2018, doi: 10.1109/ACCESS.2018.2841417.

[32] Y. Qin, W. Yang, and W. Liu, “A Probability-based Caching Strategy with Consistent Hash in Named Data Networking,” in *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, 2018, pp. 67–72. doi: 10.1109/HOTICN.2018.8606014.

[33] T. Tanwir, G. Hendrantoro, and A. Affandi, “Combination of fifo-lru cache replacement algorithms on proxy server to improve speed of response to object requests from clients,” vol. 12, pp. 710–715, May 2017.

[34] T. Ma, J. Qu, W. Shen, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, “Weighted Greedy Dual Size Frequency Based Caching Replacement Algorithm,” *IEEE Access*, vol. 6, pp. 7214–7223, 2018, doi: 10.1109/ACCESS.2018.2790381.

[35] W. Ali, “Performance Improvement of Web Proxy Cache Replacement using Intelligent Greedy-Dual Approaches,” *International Journal of Advanced Computer Science and Applications*, vol. 9, Mar. 2018, doi: 10.14569/IJACSA.2018.090810.

[36] M. Arlitt, L. Cherkasova, J. Dilley, R. Friedrich, and T. Jin, “Evaluating content management techniques for Web proxy caches,” *SIGMETRICS Perform. Eval. Rev.*, vol. 27, no. 4, pp. 3–11, Mar. 2000, doi: 10.1145/346000.346003.

[37] G. Karakostas and D. Serpanos, “Exploitation of different types of locality for Web caches,” in *Proceedings - IEEE Symposium on Computers and Communications*, Mar. 2002, pp. 207–212. doi: 10.1109/ISCC.2002.1021680.

[38] M. Abrams, C. R. Standridge, G. Abdulla, S. M. Williams, and E. A. Fox, “Caching Proxies: Limitations and Potentials,” *Proceedings of the Fourth International Conference on World Wide Web*, 1995, [Online]. Available: https://api.semanticscholar.org/CorpusID:2638802

[39] M. Abrams, C. Standridge, G. Abdulla, E. Fox, and S. Williams, “Removal Policies in Network Caches for World-Wide Web Documents.,” in *ACM SIGCOMM Computer Communication Review*, Mar. 1996, pp. 293–305. doi: 10.1145/248156.248182.

[40] A. Yasin and A. Abusamra, “Least Recently Plus Five Least Frequently Replacement Policy (LR+5LF),” *International Arab Journal of Information Technology*, vol. 9, Mar. 2012.

[41] S. Bai, X. Bai, and X. Che, “Window-LRFU: A cache replacement policy subsumes the LRU and window-LFU policies,” *Concurr Comput*, vol. 28, p. n/a-n/a, Mar. 2015, doi: 10.1002/cpe.3730.

[42] S. Nassane, S. M Mostefaoui, A. Alem, and B. Mebarek, “Window-LRFU Scheme in Named Data Networking,” in *the First National Conference on Computational Systems and Applications (NCCSA’2024)*, Khemis Melliana, Algeria: 14-15, May 2024. [Online]. Available: http://nccsa2024.univ-dbkm.dz/index.html.

[43] P. Singh, R. Kumar, S. Kannaujia, and N. Sarma, “Adaptive Replacement Cache Policy in Named Data Networking,” in *2021 International Conference on Intelligent Technologies (CONIT)*, 2021, pp. 1–5. doi: 10.1109/CONIT51480.2021.9498489.

[44] Y.-J. Kim and J. Kim, “ARC-H: Adaptive replacement cache management for heterogeneous storage devices,” *Journal of Systems Architecture*, vol. 58, no. 2, pp. 86–97, 2012, doi: https://doi.org/10.1016/j.sysarc.2011.12.002.

[45] Y. A.-B. Ibrahim, A. M. Mohamed, Sh. S. Ahmed, and A.-T. Ahmed, “Cache Policies for Smartphone in Flexible Learning: A Comparative Study,” *MANSOURA ENGINEERING JOURNAL*, vol. 41, no. 3, Sep. 2016.

[46] R. Chiocchetti, D. Rossi, and G. Rossini, “ccnSim: An highly scalable CCN simulator,” *2013 IEEE International Conference on Communications (ICC)*, pp. 2309–2314, 2013, [Online]. Available: https://api.semanticscholar.org/CorpusID:17852713

[47] A. Varga and R. Hornig, “An overview of the OMNeT++ simulation environment,” May 2008, p. 60. doi: 10.1145/1416222.1416290.

[48] N. Syambas, H. Situmorang, and M. A. P. Putra, “Least Recently Frequently Used Replacement Policy in Named Data Network,” May 2019, pp. 1–4. doi: 10.1109/ICWT47785.2019.8978218.

[49] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: evidence and implications,” in *IEEE INFOCOM ’99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, 1999, pp. 126–134 vol.1. doi: 10.1109/INFCOM.1999.749260.