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 :

DAHAM Mohamed Abd Errahmane

Sur le thème

Les politiques d'admission dans le cache des réseaux NDN

Soutenu publiquement le 11 / 06 / 2024 à Tiaret devant le jury composé de :

Mr BEKKAR Mohamed	MAA	Université Tiaret	Président
Mr MOSTEFAOUI Sid Ahmed Mokhtar	MCA	Université Tiaret	Co- Encadrant
Mr NASSANE Samir	MAA	Université Tiaret	Encadrant
Mr DAOUD Mohamed Amine	MCB	Université Tiaret	Examinateur

2023-2024

Table des matières

Table des figures

Liste des tableaux

REMERCIEMENT

Louange à Allah, le Tout-Puissant et Miséricordieux, qui m'a accordé la sagesse et la santé nécessaires pour mener à bien ce travail malgré les défis rencontrés.

Je suis profondément reconnaissant envers le Professeur **MOSTEFAOUI Sid Ahmed Mokhtar** pour son soutien indéfectible, ses précieux conseils et sa confiance en mes capacités. Sa disponibilité et ses encouragements constants ont été une source d'inspiration et m'ont permis de progresser avec confiance.

Je suis particulièrement reconnaissant envers **NASSANE Samir**, qui a patiemment relu et corrigé ce mémoire à plusieurs reprises, contribuant ainsi grandement à sa qualité et à sa clarté. Pour son aide précieuse dans la mise en place de la simulation. Sa disponibilité, ses conseils avisés et son soutien constant ont été une source de motivation essentielle.

Mes sincères remerciements vont également à M. **DAOUD Mohamed Amine** et M. **BEKKAR Mohamed** pour avoir accepté de faire partie du jury de ce mémoire. Votre expertise et vos commentaires constructifs sont grandement appréciés.

Un grand merci aux professeurs et à l'administration de la faculté d'informatique et de mathématiques pour leur encadrement et leur soutien tout au long de mes études.

Enfin, je remercie tous ceux qui, de près ou de loin, ont contribué à la réalisation de ce travail par leurs conseils, leurs encouragements et leurs commentaires constructifs.

DAHAM Mohamed Abderrahmene.

DEDICACE

Tout d'abord, je rends grâce à Allah, le Tout-Puissant, qui m'a accordé la force et la persévérance nécessaires pour mener à bien cette thèse, une expérience enrichissante et gratifiante.

Mes plus sincères remerciements vont à mes parents "Aziz , Cherifa", mon grand-père "ABED Mohammed", mon grand-mère " Mima DALILA" mes sœurs "Aya, Nadjlaa, Zineb" et à toute ma famille, piliers de soutien et d'encouragement tout au long de ce parcours, tant moralement que matériellement. Une pensée particulière pour ma grand-mère "BOSSIRI Meriem, KEBOUCHE Zineb, DAHAM Ahmed, KEBICHE Mohamed", et pour tous les défunts, qu'ils soient de ma famille ou de mes connaissances, que Dieu les accueille en Sa Miséricorde et leur accorde le Paradis.

Comme le dit si bien le proverbe anglais : « **Many hands make light work** », je souhaite exprimer ma gratitude à tous ceux qui, de près ou de loin, ont contribué à ma compréhension et à mon apprentissage dans les domaines de l'informatique, des réseaux, et de l'art de vivre. À chaque personne, jeune ou âgée, homme ou femme, qui a influencé positivement ma vie et ma vision du monde, je vous remercie du fond du cœur. Que ce travail puisse éclairer les générations futures et tous ceux qui le liront.

"It does not matter how slowly you go as long as you do not stop." Confucius.

Résumé

Le réseau NDN (Named Data Networking) représente une révolution dans la gestion des données en ligne, se distinguant des systèmes traditionnels par son accent sur l'identification et l'accès aux données via des noms plutôt que des adresses IP. Fondé sur des principes tels que le nommage des données, la séparation des intérêts et des données, ainsi que la sécurité intégrée, le NDN offre une approche novatrice pour répondre aux besoins de l'Internet moderne. Au cœur de cette architecture, la mise en cache joue un rôle central. En effet, elle permet de réduire la latence en stockant temporairement les données transférées, d'optimiser l'utilisation de la bande passante en évitant les transferts redondants, et de renforcer la résilience du réseau en garantissant la disponibilité des données même en cas de défaillance de certains nœuds. En outre, la mise en cache contribue à améliorer la scalabilité en gérant efficacement la croissance de la charge sur le réseau, tout en renforçant la sécurité en réduisant les vulnérabilités potentielles. Ainsi, la mise en cache s'avère être un élément essentiel du NDN, contribuant à sa performance, son efficacité et sa sécurité globales, et démontrant son potentiel à façonner l'avenir de l'Internet.

Mots clés : Réseau NDN, mise en cache, nommage, performance.

Abstract

The NDN (Named Data Networking) network marks a paradigm shift in managing online data, departing from conventional systems by prioritizing data identification and access through names instead of IP addresses. Founded on principles like data naming, the separation of interests and data, and integrated security, NDN presents an innovative solution to address the requirements of today's internet landscape. At the core of this architecture lies caching, which assumes a pivotal role. It diminishes latency by temporarily storing transmitted data, optimizes bandwidth utilization by circumventing redundant transfers, and fortifies network resilience by ensuring data availability, even amidst node failures. Furthermore, caching aids in bolstering scalability by adeptly handling network load escalations, all while augmenting security by mitigating potential vulnerabilities. Thus, caching emerges as a fundamental element of NDN, bolstering its overall

efficacy, efficiency, and security, thereby underscoring its potential to redefine the future of the internet.

Keywords : NDN Network, caching, naming, performance.

Introduction générale

"**NDN is not just about changing the plumbing of the Internet ; it's about changing the way we think about the Internet**"

Van Jacobson, PARC Research Fellow and NDN pioneer

Le monde numérique tel que nous le connaissons a été profondément influencé par l'architecture Internet actuelle, qui repose sur l'infrastructure IP. Cependant, cette architecture est confrontée à des difficultés croissantes en matière de sécurité, de gestion des adresses et d'efficacité dans la distribution de contenu. Afin de surmonter ces défis, l'architecture de réseau de données nommées (NDN) a été développée, mettant l'accent sur les données elles-mêmes plutôt que sur leur emplacement.

Contrairement à l'IP, qui se base sur des adresses de destination, le NDN utilise des noms de contenu. Parmi les avantages clés de l'architecture NDN, on trouve :

- Sécurité intégrée : Chaque paquet de données est signé, garantissant ainsi l'authenticité et l'intégrité des données.
- Efficacité de la diffusion : Le NDN est conçu pour optimiser la diffusion de contenu, ce qui est particulièrement avantageux pour les applications de streaming et de partage de fichiers.
- Mobilité et résilience : Le NDN gère la mobilité des utilisateurs et des appareils de manière plus efficace grâce à son modèle de routage basé sur les noms.
- Mise en cache : Le NDN tire parti du caching pour améliorer l'efficacité du réseau.

La mise en cache est l'un des aspects les plus innovants et efficaces de l'architecture NDN. Elle permet de stocker temporairement les données à différents points du réseau, réduisant ainsi la latence et la charge des serveurs d'origine. Contrairement à l'architecture IP traditionnelle où le cache est principalement géré par des serveurs dédiés, le NDN autorise tout nœud du réseau à stocker des copies des données.

Pour optimiser la mise en cache dans l'architecture NDN, différents algorithmes d'admission au cache sont utilisés. Ces algorithmes déterminent quelles données

doivent être mises en cache et où, garantissant ainsi une mise en cache intelligente et stratégique des données.

Ce mémoire se structure en trois chapitres comme suit :

Chapitre 1 : Introduction à l'architecture IP, exposant ses limites et l'évolution vers la nouvelle architecture réseau NDN et ses caractéristiques.

Chapitre 2 : Présentation des différents algorithmes d'admission au cache dans NDN, incluant LCE, LCD, Probcache, Betweenness et TinyLFY, avec une focalisation sur le dernier algorithme.

Chapitre 3 : Description du travail réalisé dans notre projet à travers l'utilisation de simulations, suivie d'une discussion et d'une comparaison des résultats des tests effectués.

Chapitre 1

Architecture NDN

1.1 Introduction

L'Internet est basé sur le protocole de communication IP, également connu sous le nom de protocole Internet, qui a été créé pour permettre à divers ordinateurs et périphériques d'utiliser des adresses IP distinctes pour communiquer entre eux. Les données sont transmises sous forme de paquets de données qui sont routés jusqu'à leur destination via des serveurs intermédiaires. Cependant, avec la croissance exponentielle des données et l'évolution des exigences de l'utilisateur, plusieurs problèmes se posent, en particulier la sécurité et la mobilité, etc. Cela signifie que le protocole IP actuel n'est pas suffisant pour répondre aux besoins d'Internet actuels et futurs. Pour cette raison, de nouvelles technologies telles que ICN, dont NDN fait partie, sont développées. Un nouveau modèle de communication de réseau appelé **Named Data Networking (NDN)** utilise un nom de contenu plutôt que l'adresse IP pour identifier et router les données. Le concept consiste à remplacer la communication centrée sur le contenu par une communication centrée sur le contenu, où chaque paquet de données est considéré comme un objet identifié par un nom unique.

Dans ce chapitre, nous présentons les limites et les défis de l'architecture de l'internet actuel. Nous décrivons les architectures Information-Centric Networking (ICN), Content Centric Networking (CCN) et nous nous concentrerons sur l'architecture NDN et sa caractéristique.

1.2 Architecture IP

Les ordinateurs peuvent communiquer entre eux via un réseau mondial appelé Internet [9]. Celui-ci repose sur un certain nombre de protocoles, dont le protocole Internet (IP), qui gère l'adressage et le routage des données[32].

Le protocole IP fonctionne en attribuant une adresse IP unique à chaque appareil connecté à un réseau. Lorsqu'un appareil veut envoyer des données à un autre, il encapsule les données en paquets IP et inclut l'adresse IP de destination. Les paquets sont ensuite routés à travers le réseau en fonction des adresses IP jusqu'à ce qu'ils atteignent le périphérique de destination. Ce processus est connu sous le nom de routage IP [22].

1.3 Les limites de l'architecture actuelle d'internet

L'architecture actuelle d'internet été conçue pour faciliter la communication entre les différents réseaux. Cependant, avec la croissance exponentielle de l'utilisation d'internet pour la distribution de contenus tels que le commerce électronique, les médias numériques et les applications mobiles, il est devenu évident que cette architecture présente des limites dans ce domaine. Le système de distribution diffère du système de communication en ce sens qu'il se concentre sur la disponibilité des informations plutôt que sur leur acheminement. Cela rend complexe et sujet aux erreurs l'utilisation d'un protocole de communication point-à-point pour résoudre les problèmes liés à la distribution. Il ya des autres problèmes qui n'est pas résoudre, nous citons su dessus :

1.3.1 La sécurité et la confidentialité

1. Il n'y a pas de chiffrement par défaut dans l'architecture actuelle, ce qui rend impossible le chiffrement par défaut des données qui circulent sur le réseau. Cela signifie que des tiers non autorisés peuvent intercepter et lire des informations sensibles.
2. Vulnérabilités des protocoles : Le protocole HTTP et d'autres protocoles utilisés sur Internet peuvent être vulnérables aux attaques telles que l'interception de données, les attaques de type "man-in-the-middle" et les attaques par injection.
3. Manque de contrôle d'accès : l'architecture actuelle ne fournit pas de mécanismes robustes de contrôle d'accès pour protéger les ressources réseau. Cela peut provoquer des problèmes de sécurité tels que l'accès non autorisé aux serveurs et aux données sensibles.
4. Faible résistance aux attaques DDoS : Les attaques par déni de service distribué (DDoS) peuvent facilement submerger les serveurs et les réseaux, rendant les services inaccessibles aux utilisateurs légitimes.

1.3.2 La distribution des contenus

1. Latence et congestion : lors de la distribution de contenus, l'architecture actuelle peut entraîner des problèmes de latence et de congestion. Les données peuvent mettre du temps à atteindre leur destination, surtout si elles doivent passer par plusieurs routeurs intermédiaires.

2. Inégalité de bande passante : l'architecture actuelle ne garantit pas une répartition équitable de la bande passante entre les utilisateurs et les fournisseurs de contenu. Cela peut causer des problèmes de performance lors de la diffusion en ligne, surtout pendant les périodes de forte demande.

3. Les fournisseurs de services Internet (FSI) ont une grande influence sur la distribution du contenu. Ils peuvent mettre en place des politiques de gestion du trafic qui favorisent certains contenus par rapport à d'autres, ce qui peut entraîner une inégalité d'accès aux contenus..

4. Défis pour les nouveaux acteurs de la distribution de contenu : l'architecture actuelle de l'Internet peut rendre difficile pour les nouveaux acteurs de la distribution de contenu de rivaliser avec les grands fournisseurs. Cela peut limiter la distribution de contenu en ligne.

1.3.3 La Mobilité

1. Handover inefficace : pendant le processus de transfert d'un appareil mobile d'un point d'accès à un autre, il peut y avoir des retards et des interruptions de service. Cela peut causer des problèmes de connexion et des interruptions dans les applications en cours d'utilisation.

2. Le protocole IPv4 est actuellement utilisé dans l'architecture de l'Internet, ce qui limite le nombre d'adresses IP disponibles. Cela peut causer des problèmes lorsque de nombreux appareils mobiles sont connectés à l'Internet, en particulier avec la croissance exponentielle des objets connectés.

3. Sécurité des réseaux sans fil : Les réseaux sans fil utilisés pour connecter les appareils mobiles, tels que les réseaux Wi-Fi et cellulaires, peuvent être vulnérables aux attaques de sécurité.

Ainsi, l'Internet actuel requiert une évolution vers des réseaux plus efficaces, plus adaptables et plus durables. Cette transition vers une nouvelle architecture réseau pourrait répondre aux besoins croissants en termes de connectivité, de disponibilité et de sécurité des données

1.4 Le réseau orienté contenu : une alternative à l'approche IP

Le réseau orienté contenu (ICN) est une architecture de réseautage qui se concentre sur le contenu plutôt que sur les hôtes. Il implique l'utilisation de noms de contenu, un routage basé sur les contenus, ainsi que la livraison et le stockage des contenus dans le réseau. Les architectures orientées contenu incluent DONA (Data-Oriented Network Architecture), NetInf (Network of Information), CCN (Content-Centric Networking) et NDN (Named Data Networking). L'ICN implique que les utilisateurs finaux expriment leurs intérêts pour des contenus, tandis que les routeurs sont chargés de cacher, récupérer et distribuer ces contenus demandés [19].

1.5 ICN et CCN : Les origines de NDN

1.5.1 L'architecture de réseau ICN (Information-Centric Networking)

Le réseau centré sur l'information (ICN) est un concept de réseautage prometteur qui pourrait fournir des solutions améliorées pour les applications de communication. L'approche ICN a été créée en réponse à la transformation significative de l'utilisation d'Internet et aux défis cités précédemment [17]. La distribution des contenus indépendamment de leurs hôtes d'origine est le principal objectif de cette méthode. L'ICN se concentre sur la gestion et l'accès à l'information plutôt que sur l'identification et la communication entre des hôtes spécifiques. Contrairement aux réseaux conventionnels basés sur IP, où les communications sont basées sur les adresses IP des hôtes, l'ICN se concentre sur l'acheminement et la récupération du contenu en utilisant des noms uniques.

Les noms, plutôt que les adresses IP, sont utilisés pour identifier les données dans un réseau ICN. Peu importe où il se trouve physiquement, les utilisateurs peuvent demander du contenu en utilisant son nom, et le réseau se charge de trouver et de livrer le contenu demandé. Cela permet une meilleure utilisation des ressources du réseau et une récupération de contenu plus efficace. L'approche des réseaux centrés sur l'information vise à fournir une infrastructure réseau plus résiliente et adaptable pour la distribution de contenu et la mobilité, en utilisant la mise en cache, la communication multipartite et les modèles d'interaction déconnectés [4].

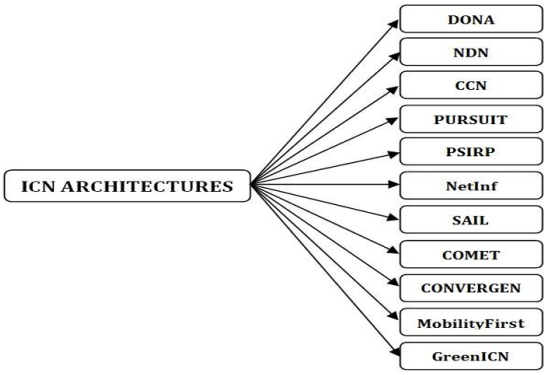


FIGURE 1.1: Architectures ICN [6]

L'ICN présente de nombreux avantages par rapport aux réseaux IP conventionnels. En premier lieu, il permet une distribution de contenu plus efficace en utilisant la mise en cache de contenu à différents niveaux du réseau. Cela réduit la congestion du réseau et améliore les performances globales. De plus, l'ICN offre une meilleure sécurité grâce à l'utilisation de mécanismes de signature numérique pour vérifier l'authenticité et l'intégrité du contenu. Cependant, l'ICN rencontre également des problèmes. Les politiques de contrôle d'accès et la gestion des noms de contenu sont des aspects complexes à prendre en compte. De plus, la transition d'une architecture ICN à une infrastructure IP traditionnelle nécessite des efforts de déploiement et de compatibilité. La mise en cache dans les réseaux centrés sur l'information améliore la vitesse de distribution des données, mais des défis subsistent dans le déploiement et la gestion du cache, nécessitant des recherches supplémentaires [16].

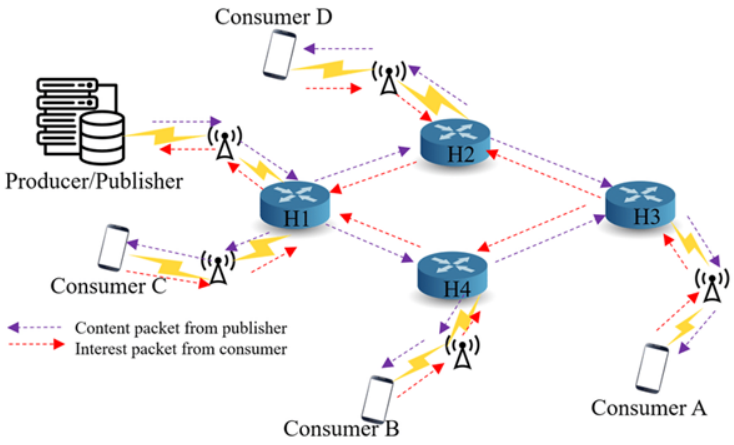


FIGURE 1.2: L'architecture de réseau ICN [5]

1.5.2 Content Centric Networking (CCN)

Content-Centric Networking (CCN) est une architecture d'Internet alternative qui se concentre sur le contenu et les services plutôt que sur la communication entre hôtes [30]. Elle vise à libérer le contenu et les services des adresses du réseau, ce qui entraîne des conséquences systémiques souhaitables [27].

L'origine de CCN remonte à la proposition initiale de Van Jacobson en 2006 dans son article "Networking Named Content" [2]. Depuis lors, CCN a évolué et a donné naissance à Named Data Networking (NDN), une implémentation spécifique de CCN développée par le projet de recherche NDN.

1.5.2.1 Évolution de CCN

Le CCN a évolué en réponse aux limitations de l'architecture client-serveur traditionnelle d'Internet. Pour comprendre cette évolution, Van Jacobson a proposé CCN dans sa version originale en 2006 dans son article "Networking Named Content". En introduisant des concepts clés tels que le routage basé sur le contenu et la mise en cache au niveau du réseau, cette proposition a posé les bases de l'architecture CCN. Au fil du temps, CCN a connu des changements importants, en particulier avec l'apparition des réseaux de données nommés (NDN). En fait, NDN est une implémentation spécifique de CCN développée par le projet de recherche NDN. Cette implémentation a apporté des améliorations et des contributions importantes à l'architecture CCN.

1.5.2.2 Les aspects clés que CCN concentrée

1. Modèle de distribution de contenu : CCN a introduit le concept de contenu nommé, où le contenu est identifié par un nom plutôt que par une adresse IP. En permettant aux utilisateurs de demander spécifiquement le contenu souhaité, quelle que soit sa localisation, cela permet une récupération du contenu plus efficace et flexible.
2. Routage basé sur le contenu : CCN a introduit un routage basé sur le contenu, où les routeurs CCN utilisent des tables de routage basées sur le contenu pour déterminer le chemin d'acheminement des paquets. Ces tables de routage contiennent des entrées qui associent les noms de contenu aux interfaces de sortie correspondantes.
3. Mise en cache au niveau du réseau : CCN encourage la mise en cache du contenu au niveau des routeurs, ce qui permet de stocker temporairement le contenu demandé et de le servir localement aux utilisateurs ultérieurs. Cela améliore les performances globales du réseau en réduisant la latence et la charge sur les serveurs d'origine.

4. Sécurité et confidentialité : Avec l'évolution de CCN, la sécurité et la confidentialité des données sont devenues des priorités. Pour garantir l'authenticité des données et protéger le contenu contre les attaques, des mécanismes de chiffrement et de signature numérique ont été intégrés.

5. Multicast natif : CCN prend en charge le multicast natif, permettant la diffusion simultanée d'un seul paquet de contenu à plusieurs destinataires. Cela réduit la charge sur le réseau et évite la duplication du trafic, favorisant ainsi une distribution plus efficace du contenu.

L'architecture CCNx est la version actuelle de l'architecture CCN. Cisco a développé CCNx pour étendre le modèle de CCN en y ajoutant des fonctionnalités avancées telles que la prise en charge de la mobilité et la qualité de service (QoS). L'architecture CCN, ainsi que son implémentation NDN, continuent d'évoluer et de susciter un intérêt croissant dans la communauté de la recherche et de l'industrie, représentant une approche novatrice pour la distribution de contenu sur Internet.

1.6 Named Data Networking (NDN)

1.6.1 Definition

1.6.1.1 Definition 1

Named Data Networking est un concept de réseautage qui vise à transmettre des données sur un lien une seule fois, même si les demandes sont faites à des moments différents. Il met également l'accent sur la vérification de l'origine et de l'authenticité des données pour le destinataire, tout en préservant l'anonymat du destinataire vis-à-vis de l'expéditeur. De plus, Named Data Networking (NDN) utilise des noms au lieu d'adresses pour identifier les données [29].

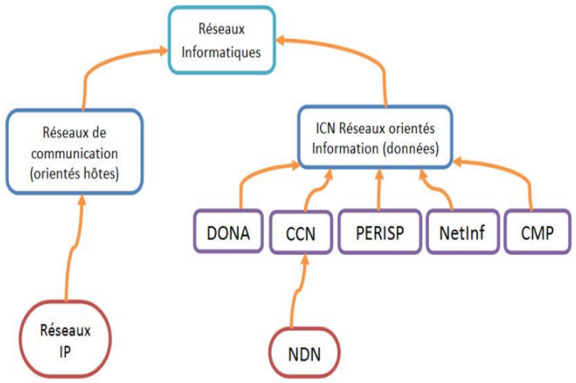


FIGURE 1.3: NDN et les principales architectures réseau [7]

1.6.1.2 Definition 2

NDN représente un paradigme de réseautage innovant conçu pour améliorer le débit et l'efficacité de la transmission de données par rapport aux réseaux IP conventionnels. Ceci est réalisé en employant un magasin de contenu (Content Store) pour stocker les données fréquemment demandées, réduisant ainsi la nécessité pour les données de parcourir de longues distances. NDN est compatible avec l'infrastructure IP existante et peut tirer parti des principaux services IP tels que le DNS et les protocoles de routage avec des ajustements minimaux.

1.6.2 Les entités principales composent l'architecture NDN

Pour permettre la distribution efficace et sécurisée du contenu dans l'architecture NDN, il existe trois types d'entités principales : les producteurs, les consommateurs et les routeurs, qui travaillent ensemble. Les producteurs créent du contenu, les consommateurs le demandent, et les routeurs traitent les demandes et les réponses entre les producteurs et les consommateurs. Le bon fonctionnement du réseau NDN dépend de la collaboration entre ces entités.

1. **Producteurs** : Les entités qui produisent et publient du contenu dans le réseau NDN sont appelées producteurs. Ils sont chargés de créer et de diffuser des ensembles de contenu identifiés par des noms distincts. Les producteurs peuvent être des serveurs, des capteurs, des appareils IoT ou tout autre dispositif capable de générer du contenu.
2. **Les consommateurs** : Les entités qui demandent et récupèrent du contenu spécifique dans le réseau NDN sont appelées consommateurs. Ils envoient des requêtes "**Interest**". pour obtenir des parties de contenu en utilisant les noms associés à ces parties de contenu. Les consommateurs peuvent être

des applications, des utilisateurs finaux ou d'autres entités ayant besoin d'accéder à des données particulières.

3. **Routeurs** : Les routeurs sont les nœuds intermédiaires du réseau NDN qui transmettent les requêtes des consommateurs aux producteurs appropriés et renvoient les réponses des producteurs aux consommateurs. Ils sont chargés de router et de transférer des paquets de données dans le réseau NDN, prenant des décisions de routage et maintenant les tables de routage en utilisant les informations de nom et de contenu.

1.6.3 Les types de paquets dans NDN

Dans NDN, les requêtes et les réponses sont traitées au niveau d'un paquet réseau. Chaque requête, qui inclut le nom des données désirées, est encapsulée dans un paquet NDN Interest. En retour, une seule réponse est fournie sous la forme d'un paquet de données NDN. Ces paquets de données sont immuables et sont munis d'une signature générée à partir de la clé cryptographique du producteur, établissant un lien entre son nom et son contenu [42] [41] [13].En NDN, on distingue principalement deux types de paquets :

1.6.3.1 Paquets d'intérêt

Dans le réseau NDN, un nœud envoie un paquet d'intérêt pour demander une donnée spécifique, identifiée par son nom. Ce paquet contient le nom de la donnée recherchée ainsi que des informations telles que le délai d'attente maximal pour la réponse. Lorsqu'un nœud reçoit ce paquet, il vérifie d'abord si la donnée demandée est déjà présente dans son cache local ; sinon, il entreprend de la rechercher ailleurs. Généralement, ce sont les consommateurs de données qui émettent ces paquets d'intérêt en direction des producteurs de données.

1.6.3.2 Paquets de données

Ces paquets contiennent les données effectives demandées par les paquets d'intérêt. Un nœud répond à un message Interest en envoyant un paquet de données lorsqu'il possède la donnée demandée dans son cache local ou lorsqu'il l'a récupérée d'une source externe. Ce paquet inclut la donnée elle-même, ainsi que son nom et d'autres métadonnées associées. Les nœuds du réseau ont la possibilité de mettre en cache ces paquets de données pour une utilisation ultérieure. Les paquets de données sont émis par les producteurs de données en direction des consommateurs de données.

Un paquet de données contient également une variété de champs, tels qu'un nom, une signature et parfois des métadonnées sur les données. Il est important

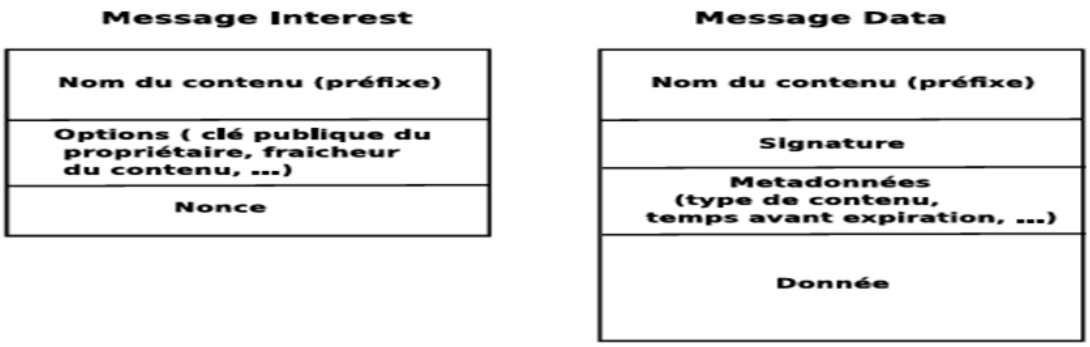


FIGURE 1.4: les paquets Interest et Data utilisés dans NDN [8]

de noter que le format de ce type de paquet est évolutif, ce qui signifie que de nouveaux types de paquets et de champs peuvent être ajoutés pour prendre en charge de nouvelles fonctionnalités.

1.6.4 Composants clés de l'architecture NDN

Les principaux composants du réseau NDN sont le magasin de contenu (Content Store), le tableau des intérêts en attente (Pending Interest Table - PIT) et la Base d'Informations d'Acheminement (Forwarding Information Base - FIB) [39]. Ces trois éléments sont essentiels pour le routage et la gestion des requêtes et des réponses dans l'architecture NDN :

1.6.4.1 le magasin de contenu (CS : Content Store)

Chaque routeur NDN contient une mémoire cache locale appelée Content Store. Cette dernière stocke temporairement les contenus les plus demandés. Lorsqu'un routeur reçoit une requête pour un contenu, il vérifie d'abord s'il a déjà ce contenu dans son Content Store. Si c'est le cas, il peut le servir directement, réduisant ainsi la latence et la charge du réseau. Le Content Store est régi par des politiques de remplacement.

1.6.4.2 La table d'intérêt en attente (PIT : Pending Interest Table)

Chaque routeur NDN possède une table d'intérêt en attente (PIT). Lorsqu'un routeur reçoit une demande de contenu, il vérifie d'abord s'il la possède déjà dans sa PIT. Si c'est le cas, cela signifie qu'il a déjà envoyé une requête pour ce contenu et attend toujours la réponse. L'interface d'entrée de la requête est alors ajoutée

à la liste des interfaces sortantes dans la PIT par le routeur. Le routeur utilise la PIT pour déterminer les interfaces de sortie appropriées lorsque la réponse est reçue, afin de transmettre la réponse aux clients qui ont initialement demandé le contenu.

1.6.4.3 Base d'Informations d'acheminement (FIB : Forwarding Information Base)

La table de routage (FIB) est un ensemble de données qui conserve des renseignements sur les meilleures routes pour atteindre des préfixes NDN précis. Les routeurs NDN l'emploient afin de transmettre de manière efficace les paquets d'intérêt et les paquets de données. Dans la FIB du routeur NDN, une entrée contient les champs suivants : <préfixe de nom, durée inactive, identification de l'interface, préférence de routage, RTT (Round-Trip Time), statut, limite de taux>.

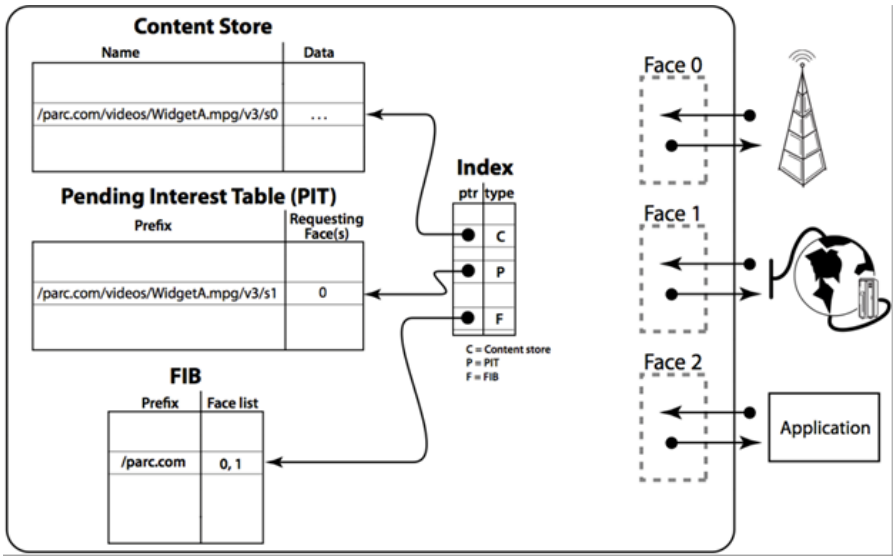


FIGURE 1.5: Composants d'un routeur NDN [26]

Les grands principes du fonctionnement de l'architecture NDN sont représentés sur les figures 1.6 et 1.7. Lorsqu'un utilisateur veut accéder à un contenu, il envoie un message Interest (message 1). Quand le Noeud1 reçoit cet Interest, plusieurs cas de figure sont possibles : (1) soit il connaît le prochain nœud pour transmettre le message, car une entrée est déjà dans sa FIB, auquel cas il transmet la requête (message 2) ; (2) soit il ne possède pas la règle pour atteindre ce contenu et va transmettre cette requête sur toutes ses autres interfaces (flooding). Dans notre exemple, on part du principe que les règles sont déjà dans les tables de transmission (FIB) des nœuds. Le Noeud2 l'envoie à son tour au Noeud3 (message 3) qui

la transfère au serveur (message 4). Ce dernier répond grâce à un message Data transportant la donnée. Ce message Data va utiliser le chemin inverse de l'Interest pour arriver à l'utilisateur (messages 5 à 8), car chaque nœud a conservé cette entrée dans sa PIT. Tous les nœuds ayant retransmis le message Data vont pouvoir conserver une copie locale dans leurs mémoires cache (CS), en fonction des mécanismes de cache instanciés sur chacun des nœuds. Ainsi, lorsqu'un utilisateur émet à nouveau une requête pour ce même contenu (message 9), le Noeud1 va pouvoir utiliser sa copie locale pour répondre (message 10). Ce principe permet de rapprocher le contenu des utilisateurs, de réduire le délai d'accès au contenu et de limiter la portée des requêtes, évitant ainsi la surcharge du réseau [8].

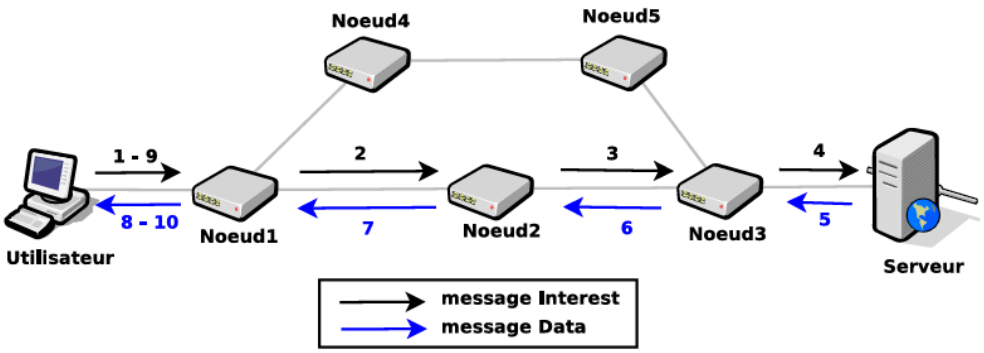


FIGURE 1.6: Processus de communication dans NDN [8]

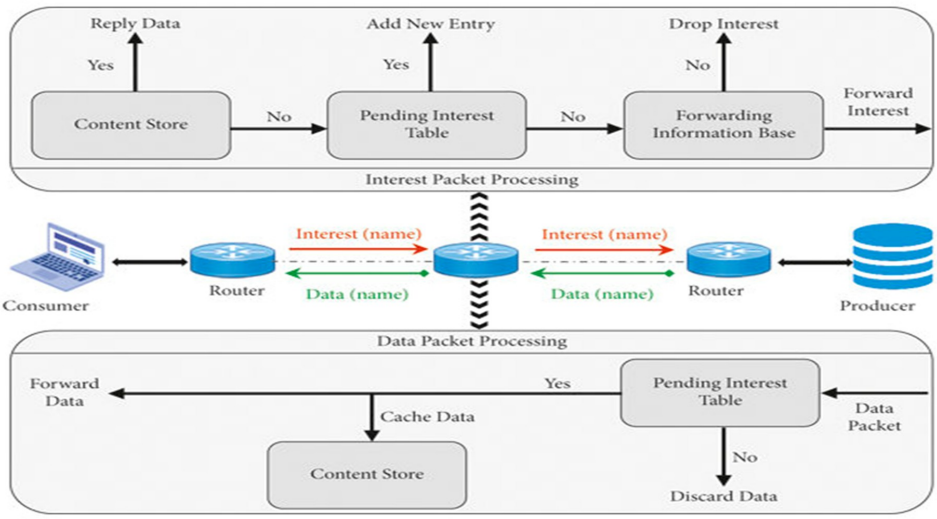


FIGURE 1.7: Traitement des paquets d'intérêts et de données dans NDN [14]

1.6.5 Analyse comparative entre les réseaux IP et NDN

Ce tableau présente une analyse comparative entre les réseaux basés sur l'IP et ceux basés sur l'NDN [35] :

Fonctionnalité	**Architecture IP**	**Architecture NDN**
Communication	Basée sur les adresses des hôtes (où envoyer)	Basée sur les noms des contenus (quoi envoyer)
Routage	Par adresse de destination	Par nom de contenu
Sécurité	Par canal sécurisé (TLS, IPSec, etc.)	Par contenu signé (signature numérique, etc.)
Caching	Par proxy ou CDN	Par routeur
Mobilité	Par redirection d'adresse (Mobile IP, etc.)	Par réexpression d'intérêt
Multicast	Par protocole spécifique (IGMP, PIM, etc.)	Par diffusion sélective
Débit	Dépend de la bande passante et du contrôle de congestion	Dépend de la disponibilité des données et du contrôle de congestion
Latence	Dépend de la distance et du nombre de sauts	Dépend de la popularité et de la localisation des données
Perte de paquets	Peut être causée par des erreurs de transmission ou des congestions	Peut être causée par des attaques ou des intérêts non satisfaits

TABLE 1.1: Comparaison entre l'architecture IP et l'architecture NDN

1.6.6 Systèmes de nommage dans NDN

Dans l'Internet IP actuel, bien que les sites web soient localisés par des adresses IP, il est plus courant d'utiliser des noms pour les désigner. En effet, "www.google.com" est plus facile à mémoriser que l'adresse IP "172.217.168.196" associée à ce site web. Lorsqu'un utilisateur effectue une requête vers un site web en utilisant son nom, ce nom est converti en une adresse IP via le système DNS (Domain Name System) [31].

Le nommage est aujourd'hui l'aspect le plus important de NDN, car toutes les autres fonctionnalités de NDN (architecture, routage, découverte des noms, etc.) dépendent étroitement de la façon dont les données et les domaines sont nommés [7].

Voici un exemple de nommage dans NDN avec le site Google : Dans NDN, le nom de la racine est généralement représenté par un seul segment, typiquement 'ndn'. Par conséquent, le nom de la racine serait '/ndn'. Le nom du site Google serait représenté par un segment, par exemple 'google'. Donc, le nom du site serait '/ndn/google'.

Les sous-répertoires dans NDN peuvent être utilisés pour organiser les données de manière logique. Dans le cas de Google, nous pouvons utiliser des sous-répertoires pour représenter les différents services ou sections du site. Par exemple, si nous voulons accéder à la section 'Actualités' de Google, nous pouvons ajouter un segment 'actualites' à notre nom. Le nom complet serait donc '/ndn/google/actualites'.

Si le site Google comporte des niveaux supplémentaires dans sa structure, nous pouvons les ajouter sous forme de segments supplémentaires dans le nom. Par exemple, si la section 'Actualités' a des sous-sections telles que 'Sports' et 'Politique', nous pouvons ajouter les segments correspondants à notre nom. Le nom complet pour accéder aux actualités sportives de Google serait donc '/ndn/google/actualites/sports'.

1.6.7 Avantages de NDN

Un réseau NDN offre de nombreux avantages potentiels aux utilisateurs et aux communautés, parmi lesquels [18] :

1. Simplification de la configuration des réseaux : NDN envoie des paquets d'intérêt et de données à l'aide des noms de la couche d'application, ce qui évite la nécessité de configurer les réseaux avec des adresses IP. En cas de nombreux appareils connectés, cette simplification est particulièrement bénéfique.
2. Sécurité des données centrées : Le mécanisme de sécurité des données de NDN garantit la sécurité de chaque donnée produite tout au long de sa durée de vie, éliminant ainsi le besoin d'isolement physique et logiciel ainsi que de sécurisation des canaux de communication.
3. Le NDN peut améliorer l'efficacité de la distribution de contenu en offrant un streaming plus rapide, des temps de buffering réduits et une expérience utilisateur améliorée.
4. 4. Les avantages potentiels de l'utilisation de NDN dans les milieux de santé incluent la gestion sécurisée d'un grand nombre de données patients, la garantie de la confidentialité et l'amélioration de la rapidité de récupération des données par les professionnels de la santé.
5. Surmonter la congestion du réseau avec NDN en se concentrant sur le contenu plutôt que sur l'emplacement peut aider à réduire cette congestion

en distribuant plus efficacement le contenu populaire sur plusieurs nœuds, ce qui diminue la pression exercée sur des routes réseau spécifiques

6. La mise en cache joue un rôle essentiel dans l'amélioration de l'expérience des utilisateurs avec NDN, en permettant un accès plus rapide au contenu fréquemment demandé, une réduction de la latence et une amélioration générale des performances du réseau.

1.7 Conclusion

La croissance exponentielle du trafic Internet et les besoins de communication en constante évolution ont révélé les limites de l'architecture IP actuelle. Pour répondre à ces défis, une approche émergente qui se présente comme solution potentielle est celle des réseaux orientés contenus, notamment le CCN et plus précisément le réseau NDN.

La proposition de la solution NDN implique une transition d'une architecture centrée sur les hôtes vers une architecture centrée sur le contenu. Dans cette approche, l'identification se fait à l'aide de noms de contenu plutôt que d'adresses IP. Cette transition vise à favoriser une diffusion optimale du contenu, une plus grande flexibilité pour gérer la croissance du trafic, une meilleure résilience aux pannes, ainsi qu'une amélioration de la sécurité et de la confidentialité des informations.

Le réseau NDN met en œuvre ces principes en utilisant des mécanismes de cache pour temporairement stocker les données à différents points du réseau, réduisant ainsi la charge sur les serveurs d'origine et améliorant l'efficacité globale du réseau. De plus, la sécurité est renforcée grâce à l'utilisation de signatures numériques et de chiffrement, garantissant la fiabilité et l'authenticité des échanges au sein du réseau NDN.

En comparant NDN aux réseaux IP, on peut constater les avantages potentiels de NDN en ce qui concerne la disponibilité de contenu, un débit potentiellement plus élevé et des latences réduites. En résumé, l'approche NDN présente un avenir prometteur pour l'Internet, en tenant compte des limitations de l'architecture IP actuelle. Toutefois, la transition vers cette nouvelle architecture nécessitera une adoption progressive et des efforts coordonnés pour assurer une interopérabilité et une évolutivité optimales.

Chapitre 2

Les mécanismes d'admission au cache dans NDN

2.1 Introduction

Les mécanismes d'admission au cache jouent un rôle important dans les réseaux NDN. Ils sont chargés de garantir que les demandes de contenu sont satisfaites de manière efficace et équitable.

Ce chapitre explore les différents types de mécanismes d'admission dans NDN, en particulier l'algorithme TinyLFU, et examine en détail ses principes, son fonctionnement et ses applications. L'objectif principal est de fournir une compréhension approfondie de TinyLFU et de son rôle dans l'amélioration des performances des systèmes de mise en cache.

2.2 La Mise en cache

Un cache est un mécanisme de stockage à grande vitesse conçu pour accélérer l'accès aux données.

Dans NDN, la mise en cache implique le stockage des données dans les routeurs par lesquels elles sont transmises, afin d'obtenir une réponse rapide à chaque besoin de ces données. Les étapes de mise en cache dans NDN sont les suivantes :

1. **Envoi d'un paquet d'intérêt :** lorsqu'un utilisateur demande des informations, il envoie un paquet d'intérêt. Chaque routeur vérifie alors son magasin de contenu pour confirmer la présence des données demandées.
2. **Ajouter la demande à la table PIT et la transmettre :** En l'absence des données demandées dans le magasin de contenu du routeur, celui-ci enregistre la demande dans sa table PIT (table des intérêts en attente). Ensuite, la demande est envoyée au producteur des données pour traitement.

3. **L'enregistrement des données :**Une fois la demande satisfaite, des copies des données sont stockées sur les routeurs le long du chemin vers le consommateur, selon la stratégie de placement adoptée.

On note que la performance du cache dans NDN est influencée par deux facteurs principaux :

1. La stratégie de placement du cache.
2. La politique de remplacement (éviction) du cache [34] [40].

Ainsi, La décision de mettre en cache les données dans les routeurs intermédiaires repose sur deux questions principales : 1) Où placer le contenu ?, 2) Quel contenu remplacer (éliminer) en premier ?.

En général, on évalue la performance du cache en se basant sur les quatre indicateurs suivants :

1. le taux de réussite du cache (hit ratio) : indique la proportion de demandes de données qui sont satisfaites à partir du cache plutôt que d'être récupérées à partir du producteur original. Un taux de réussite élevé du cache indique une utilisation efficace du cache et une réduction du trafic vers la source d'origine, ce qui peut améliorer les performances globales du système.
2. La latence : représente le temps total écoulé entre la génération d'une demande de contenu et sa réception par le consommateur.
3. Le nombre moyen de sauts parcourus : mesure la distance moyenne entre le consommateur et le plus proche routeur ayant satisfait ses demandes. Un nombre moyen de sauts plus faible dans NDN indique une meilleure efficacité du cache et une réduction du trafic sur le réseau, ce qui peut contribuer à des performances optimales et une utilisation plus efficace des ressources [36].

2.3 Les Mécanisme d'admission au cache dans NDN

Dans NDN, la stratégie de placement du cache est souvent considérée comme une méthode d'admission au cache car elle aide à décider quels contenus doivent être conservés dans les routeurs intermédiaires pour répondre aux demandes ultérieures des consommateurs. Dans certains cas, il est possible qu'aucune politique de contrôle d'admission ne soit en place. Ainsi, toutes les données entrantes sont acceptées, ce qui peut entraîner une pollution du cache et avoir un impact négatif sur les performances globales du réseau. Dans NDN, où des caches résident sur chaque routeur, les algorithmes d'admission au cache deviennent un facteur de plus en plus important à prendre en compte.

FIGURE 2.1: Stratégie d'admission au cache

En fait, les politiques d'admission se réfèrent aux règles ou aux algorithmes utilisés pour décider quels éléments sont autorisés à être stockés dans un cache et lesquels doivent être supprimés pour faire place à de nouvelles données. Il est important de mettre en place ces politiques afin d'améliorer les performances du cache en garantissant la conservation des données fréquemment consultées ou importantes dans le cache, ce qui améliore l'efficacité globale du système. Ainsi, un nouvel élément n'est inséré dans le cache que si une politique d'admission détermine que le taux de réussite du cache est susceptible de bénéficier de son stockage à la place de la victime du cache.

Une politique d'admission de cache efficace dans les réseaux NDN doit :

1. **Identifier les données populaires** : Déterminer quelles données sont susceptibles d'être redemandées par les utilisateurs.
2. **Évaluer la pertinence des données** : Prendre en compte la fraîcheur et l'actualité des informations pour décider de leur conservation.
3. **Prendre des décisions rapides** : Réagir rapidement aux changements dans les modèles de demande de données.

2.4 Choix et types des algorithmes

Au début, les algorithmes d'admission au cache sont conçus en tirant parti de la coopération des caches des routeurs dans un réseau afin que le même paquet de données ne soit pas inséré dans différents caches[37].

Les algorithmes d'admission sont divisés en deux grands types :

1. Ceux qui exploitent la coopération des caches des routeurs dans le réseau, tels que ProbCache, LCE (Leave Copy Everywhere) et Betweenness.
2. Ceux qui opèrent de manière autonome, sans cette coopération entre les caches des routeurs, tels que LCD (Leave Copy Down) et TinyLFU.

2.5 Politiques d'admission dans NDN

Dans les architectures NDN, chaque routeur le long du chemin de livraison est équipé d'un cache de contenu qui peut temporairement contenir les paquets de données qui passent. Les décisions de mise en cache de chaque routeur sur le chemin de livraison sont prises par des algorithmes d'admission au cache, qui déterminent si un paquet de données particulier doit être inséré dans le cache du routeur. L'objectif de ces algorithmes est de distribuer efficacement le contenu dans les caches réseau tout en minimisant la redondance et en améliorant les performances globales du réseau. Nous présentons ci-dessous les politiques d'admission au cache les plus connues.

2.5.1 ProbCache

C'est un algorithme d'admission probabiliste permettant de sélectionner, le long d'un chemin, les routeurs qui doivent stocker les contenus. Il vise à laisser de l'espace de cache pour d'autres contenus et à distribuer équitablement les données sur le réseau. Son objectif est de répondre à la question suivante : pendant combien de temps pouvons-nous stocker le contenu dans un routeur donné pour minimiser le trafic redondant et maximiser les bénéfices ? [33].

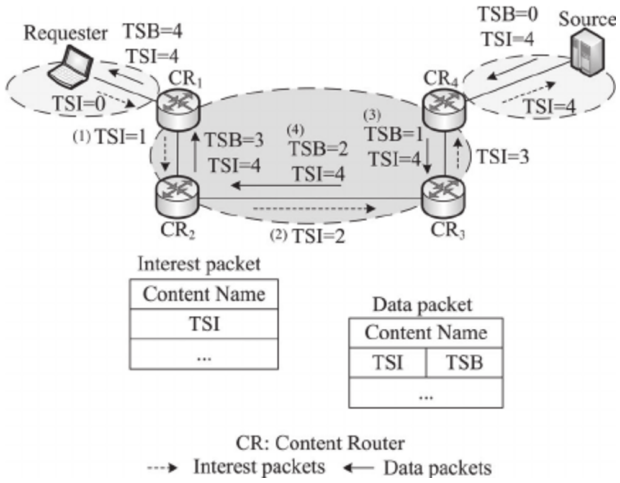


FIGURE 2.2: Illustration des opérations de Probcache [43]

En termes plus simples, le système ajoute un champ "Time Since Inception" (TSI) à l'en-tête du paquet Interest, et deux champs (TSI et "Time Since Birth" (TSB)) à l'en-tête du paquet Data. Lorsqu'un utilisateur envoie un paquet Interest, le TSI est réglé à zéro. Chaque fois qu'un routeur de contenu reçoit ce paquet, il

augmente le TSI de un, comme le montrent (1) et (2) dans la figure 2.2. De même, lorsqu'un paquet Data est envoyé par une source de contenu, le TSB est réglé à zéro et le TSI est défini comme celui du paquet Interest correspondant. Cependant, chaque fois qu'un routeur de contenu reçoit un paquet Data, il augmente le TSB de un, mais laisse le TSI inchangé. comme le montrent (3) et (4) dans la figure 2.2. probcache calcule les deux valeurs suivantes :

1. **Estimation de la capacité de mise en cache des chemins (TimesIn)** Le nombre de fois que le chemin peut cacher ce paquet est représenté par le facteur TimesIn.

$$TimesIn(x) = \frac{\sum_{i=1}^{TSI-(TSB-1)} N_i}{T_w N_c} \tag{2.1}$$

Où :

N_i : Nombre de contenu d'un routeur cache.
T_w : Temps de conservation du contenu sur le chemin de livraison. Elle est estimée à 10s.
N_c : Taille moyenne que peut avoir une cache le long du chemin
$\sum_{i=1}^{TSI-(TSB-1)} N_i$: C'est la somme de la soustraction de TSI moins TSB pour chaque paquet de données, on obtient le nombre total de caches restants sur le chemin, au lieu du nombre total de caches du fournisseur de contenu au demandeur de contenu.

2. **Cache basé sur le poids (Cache Weight)** Lorsque le paquet de données atteint le routeur de contenu, la valeur TSI et la valeur TSB sont utilisées pour calculer le Cache Weight.

$$CacheWeight(x) = \frac{TSI(x)}{TSB(x)} \tag{2.2}$$

Ensuite, le routeur prend la décision d'admission au cache d'un contenu x en se basant sur la valeur calculée comme suit de ProbCache(x) :

$$ProbCache(x) = TimesIn(x) * CacheWeight(x) \tag{2.3}$$

ProbCache vise également à réduire la redondance de mise en cache et a tendance à stocker les contenus populaires. Avec ProbCache, les routeurs prennent des décisions de mise en cache de manière collaborative, car ils échangent les valeurs TSI et TSB tout au long du chemin. Ainsi, ProbCache est un système d'admission collaboratif qui utilise une collaboration implicite entre les routeurs[43].

2.5.2 Betweenness

La stratégie d'admission Betweenness est une approche de gestion du cache utilisée dans les réseaux de données nommées NDN. Elle repose sur le calcul de la centralité de l'intermédiarité (betweenness centrality) pour chaque nœud du réseau. La centralité de l'intermédiarité mesure le nombre de fois qu'un nœud est situé sur le chemin le plus court entre deux autres nœuds dans un graphe.

Dans le contexte de NDN, la stratégie Betweenness utilise cette mesure pour identifier les nœuds du réseau qui se trouvent sur de nombreux chemins de livraison de données. Ces nœuds, ayant une forte centralité de l'intermédiarité, sont considérés comme des points stratégiques pour placer des caches de contenu. En plaçant des caches sur ces nœuds, la stratégie de placement Betweenness vise à améliorer l'efficacité de la distribution de contenu en optimisant la localisation des caches pour les données les plus demandées.

Ainsi, la stratégie de placement Betweenness de NDN cherche à placer des caches de contenu sur les nœuds du réseau qui se trouvent sur de nombreux chemins de données, ce qui permet de mieux servir les requêtes de contenu et d'améliorer les performances globales du réseau [44].

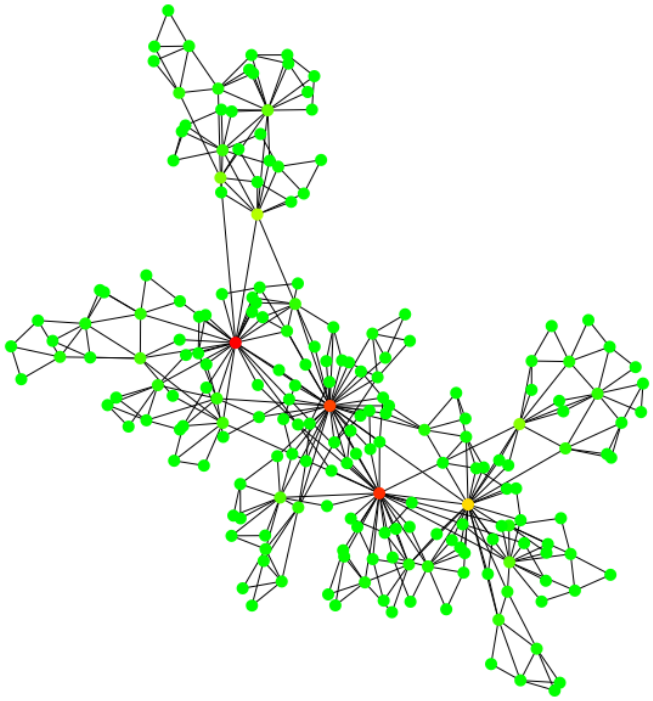


FIGURE 2.3: La stratégie de mise en cache Betweenness[23]

2.5.3 Leave Copy Everywhere (LCE)

La stratégie LCE (Leave Copy Everywhere) est une méthode simple et répandue pour la mise en cache des contenus dans les réseaux de données nommées NDN. Voici comment elle fonctionne [16] :

1. **Caching universel** : Lorsqu'un paquet de données (content object) est transmis à travers le réseau, chaque routeur sur le chemin de livraison stocke une copie de ce paquet dans son cache local.
2. **Pas de Sélection** : Contrairement à d'autres stratégies de mise en cache qui choisissent de manière sélective les contenus à cacher, LCE ne fait aucune distinction. Chaque routeur qui voit passer un contenu le met en cache.
3. **Distribution large** : En laissant une copie de chaque contenu à chaque nœud traversé, LCE maximise la disponibilité des contenus dans le réseau.

2.5.3.1 Avantages

— **Simplicité de mise en œuvre** : LCE est facile à implémenter car elle ne nécessite pas de calculs complexes ou d'algorithmes sophistiqués pour décider quels contenus mettre en cache.
— **Disponibilité accrue** : Avec des copies de contenu disséminées à travers le réseau, les chances de trouver une copie locale de la donnée demandée augmentent, réduisant la latence de récupération.

2.5.3.2 Inconvénients

— **Efficacité du cache** : Cette stratégie peut entraîner une utilisation inefficace de l'espace de cache disponible, car les contenus peuvent être dupliqués de manière excessive à travers le réseau.
— **Redondance élevée** : La duplication systématique de chaque contenu à chaque nœud peut engendrer une redondance significative, utilisant inutilement de l'espace de cache précieux et potentiellement évinçant des contenus qui auraient pu être plus utiles.

La stratégie d'admission LCE est une approche de mise en cache simple et universelle dans les réseaux NDN. Bien qu'elle améliore la disponibilité des contenus, elle peut entraîner une inefficacité de l'utilisation du cache en raison de la redondance élevée. Elle est souvent comparée à d'autres stratégies plus sophistiquées qui cherchent à optimiser l'utilisation de l'espace de cache tout en maintenant une bonne disponibilité des contenus.

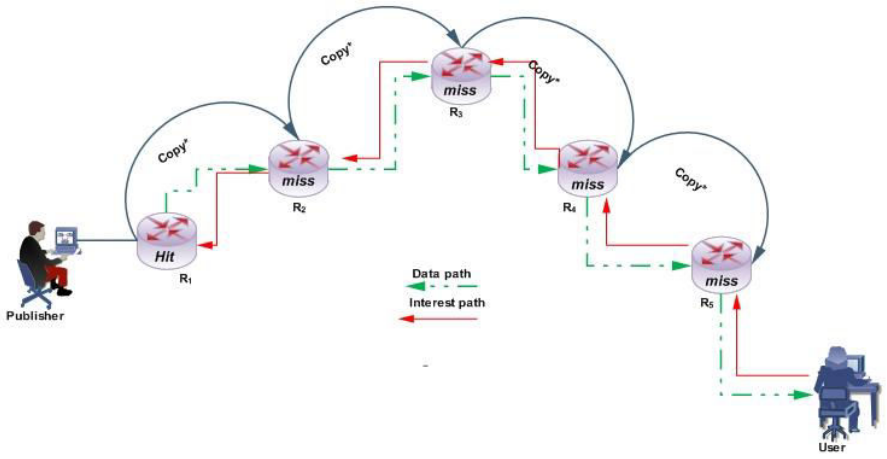


FIGURE 2.4: La stratégie de mise en cache LCE[3]

2.5.4 Leave Copy Down (LCD)

La stratégie LCD (Leave Copy Down) de NDN est une méthode optimisée de mise en cache qui réduit la redondance en plaçant des copies de contenu de manière plus sélective. En laissant une copie uniquement dans le nœud immédiatement en aval du nœud qui a servi la demande, LCD vise à équilibrer l'utilisation efficace de l'espace de cache et la disponibilité des contenus. Cette approche peut améliorer l'efficacité globale du réseau en évitant une utilisation excessive et redondante des caches tout en maintenant une bonne performance en termes de latence pour les demandes de contenu. Voici une définition et une explication de son fonctionnement :

1. **Mise en cache différée** : Contrairement à LCE, où chaque nœud sur le chemin de livraison stocke une copie du contenu, LCD ne place une copie que dans le nœud juste en aval du nœud qui a servi la demande.

2. **Chemin de livraison** : Lorsqu'un paquet de données est demandé par un utilisateur, il traverse plusieurs nœuds. Le nœud qui possède déjà le contenu le sert et transmet la donnée à l'utilisateur final.

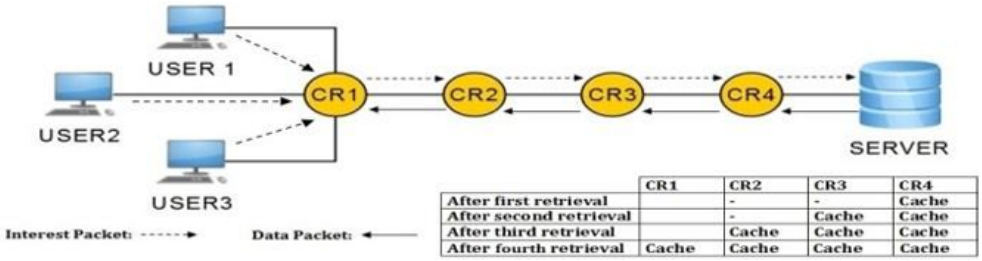
3. **Placement du cache** : Au lieu de copier le contenu à chaque nœud traversé, LCD laisse une copie du contenu uniquement dans le nœud immédiatement en aval du nœud qui a fourni le contenu. Cela signifie que le contenu est copié à un seul saut en direction de l'utilisateur final, à partir du nœud qui avait déjà la donnée en cache.

2.5.4.1 Avantages

— **Utilisation efficace du cache** : En limitant le placement des copies, LCD utilise l'espace de cache de manière plus judicieuse, réduisant la redondance excessive et laissant de la place pour d'autres contenus potentiellement plus utiles.
— **Réduction de la redondance** : Évite de surcharger les caches des nœuds avec des copies multiples du même contenu, optimisant ainsi l'utilisation des ressources du réseau.

2.5.4.2 Inconvénients

— **Complexité accrue** : La mise en œuvre de LCD peut être plus complexe que des stratégies plus simples comme LCE, car elle nécessite de suivre le chemin de retour de la demande et de comprendre la structure du réseau.
— **Disponibilité potentiellement réduite** : En plaçant des copies de contenu moins fréquemment, il peut y avoir des cas où des demandes subséquentes doivent parcourir une plus grande distance pour trouver le contenu, augmentant ainsi la latence.



	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 2.5: La stratégie de mise en cache LCD[6]

2.6 TinyLFU

TinyLFU, proposé par Einziger et Roy Friedman en 2014 [21], eest un algorithme d'admission efficace qui vise à décider intelligemment si un élément doit être placé dans le cache ou non. Contrairement aux algorithmes traditionnels qui se concentrent principalement sur l'éviction des éléments du cache, TinyLFU se distingue par sa capacité à estimer la fréquence d'accès future d'un élément avant de décider de l'admettre dans le cache [21]. Cette approche proactive permet à TinyLFU de prendre des décisions d'admission éclairées, conduisant ainsi à une utilisation plus efficace de l'espace de cache et à une amélioration des performances globales du réseau [28].

TinyLFU repose sur le principe de prédire la probabilité qu'une ressource soit requise à nouveau dans un avenir proche. Contrairement aux politiques traditionnelles qui se basent uniquement sur l'ordre chronologique des accès, TinyLFU estime la probabilité d'accès future à une ressource en fonction de son historique d'accès et de sa fréquence d'accès. Cette approche probabiliste lui permet de mieux anticiper les besoins futurs en données et de maintenir en cache les éléments les plus pertinents.

2.7 Architecture de TinyLFU

Dans la figure 2.6, la politique de remplacement du cache identifie une victime, tandis que TinyLFU détermine si remplacer cette victime par le nouvel élément serait bénéfique pour améliorer le taux de réussite. À cet effet, TinyLFU recueille des statistiques sur la fréquence des éléments dans un historique récent significatif et utilise des techniques de comptage approximatif pour estimer efficacement leur fréquence d'accès future. Si la fréquence estimée du nouvel élément est plus élevée que celle de la victime, alors TinyLFU intègre ce nouvel élément dans le cache. Sinon, la victime demeure conservée dans le cache.

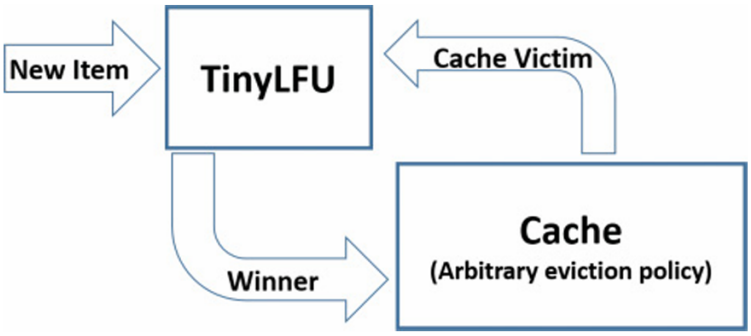


FIGURE 2.6: Normal cache et TinyLFU

2.7.1 Comptage approximatif

Le comptage approximatif repose sur l'utilisation de structures de données compactes et efficaces pour estimer la fréquence d'accès des éléments. Plutôt que de stocker une valeur de comptage précise pour chaque élément, le comptage approximatif utilise des techniques telles que les filtres de Bloom ou les compteurs de fréquence probabilistes pour fournir une estimation suffisamment précise de la fréquence d'accès. Cette approche présente les avantages suivants :

1. **Réduction de l'empreinte mémoire** : En utilisant des structures de données compactes, le comptage approximatif permet de réduire la quantité de mémoire nécessaire pour stocker les statistiques de fréquence d'accès.
2. **Simplicité de calcul** : Les opérations sur les structures de données utilisées pour le comptage approximatif sont généralement plus simples et plus rapides que celles requises pour les méthodes de comptage précises, ce qui entraîne des gains de performance.
3. **Performances maintenues** : Malgré l'approximation, les techniques de comptage approximatif utilisées par TinyLFU offrent généralement des performances de gestion de cache satisfaisantes, ce qui en fait une option attrayante pour les environnements où la mémoire et les ressources de calcul sont limitées.

2.7.1.1 Le filtre de Bloom

Le filtre de Bloom [10], est une structure de données probabiliste qui permet de tester efficacement l'appartenance d'un élément à un ensemble.

Il sacrifie une légère précision au profit d'une meilleure efficacité en temps et en espace, ce qui en fait une conception très intelligente.

Le filtre de Bloom utilise un tableau de bits initialisé à 0 pour représenter un ensemble vide. Lors de l'insertion d'un élément x, k fonctions de hachage indépendantes sont utilisées pour obtenir k positions dans le tableau, qui sont ensuite mises à 1 [12].

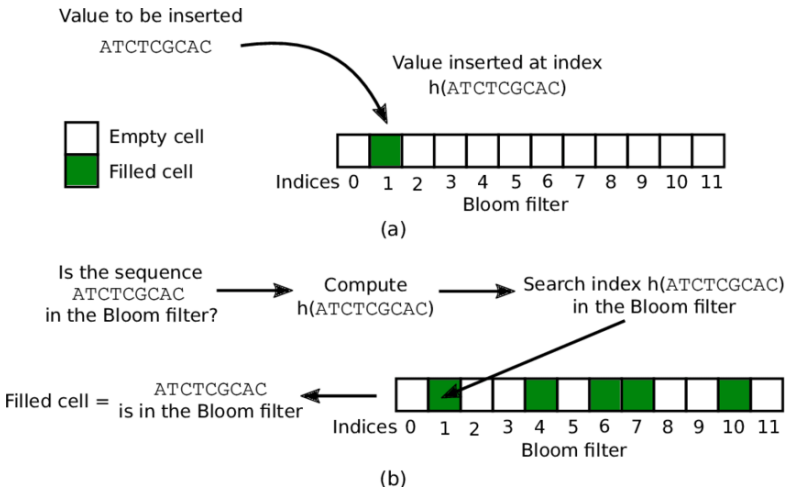


FIGURE 2.7: Déroulement de BLOOM FILTER

Le processus de recherche est similaire à l'insertion : les mêmes k fonctions de hachage sont appliquées à l'élément recherché pour obtenir k positions. Si tous les

bits à ces positions sont à 1, l'élément est probablement présent dans l'ensemble. Sinon, si au moins un bit est à 0, l'élément est certainement absent [38].

Cependant, des faux positifs peuvent se produire lorsque les positions correspondant à un élément y sont toutes mises à 1 par d'autres éléments insérés précédemment [15]. L'utilisation de k fonctions de hachage indépendantes permet de réduire ce risque.

Le filtre de Bloom offre un avantage considérable en termes d'espace et de temps par rapport à d'autres structures de données d'ensemble telles que les arbres binaires de recherche équilibrés, les tries ou les tables de hachage [25].Il ne stocke pas les éléments eux-mêmes mais seulement quelques bits obtenus par hachage, dans un tableau compact. La taille du filtre est indépendante de la taille des éléments stockés. Avec un taux d'erreur de 1% et un nombre optimal de fonctions de hachage, un filtre de Bloom ne nécessite en moyenne que 9,6 bits par élément [11].

Dans le contexte de la gestion de cache, les filtres de Bloom peuvent être utilisés pour estimer si une donnée est probablement présente dans le cache. Plutôt que de stocker l'ensemble complet des éléments présents dans le cache, un filtre de Bloom peut être utilisé pour vérifier rapidement la présence potentielle d'une donnée, ce qui permet d'économiser de la mémoire. Cependant, en raison de la possibilité de faux positifs, il est utilisé comme un filtre préliminaire avant de rechercher la donnée dans le cache réel.

2.7.1.2 Filtre de Bloom à Comptage (CBF)

Le Filtre de Bloom à Comptage (CBF) est une extension du filtre de Bloom classique [1]. Tandis que le filtre de Bloom original est conçu pour tester l'appartenance d'un élément à un ensemble de manière probabiliste, le CBF ajoute la capacité de suivre la fréquence d'occurrence des éléments. Cette caractéristique permet non seulement de vérifier l'appartenance, mais aussi de gérer l'insertion et la suppression des éléments [20].

2.7.1.2.1 Structure et Fonctionnement Les CBF offrent divers bénéfices par rapport aux filtres de Bloom traditionnels lorsqu'il s'agit d'évaluer les fréquences d'accès :

1. **Structure** :
   — **Tableau de Compteurs** : Contrairement au filtre de Bloom classique, qui utilise un tableau de bits, le CBF utilise un tableau de compteurs. Chaque compteur peut représenter une valeur entière, typiquement une valeur de 4 bits ou plus, selon la précision souhaitée.

— **Fonctions de Hachage** : Comme le filtre de Bloom, le CBF utilise plusieurs fonctions de hachage (k fonctions de hachage) pour déterminer les positions dans le tableau de compteurs.

2. **Insertion d'un Élément** : Lorsqu'un élément est inséré dans le CBF, il est haché par les k fonctions de hachage. Les k positions correspondantes dans le tableau de compteurs sont incrémentées de 1. Cela permet de suivre combien de fois un élément a été inséré.

3. **Recherche d'un Élément** : Pour vérifier si un élément est probablement présent, il est haché par les mêmes k fonctions de hachage. Si tous les compteurs aux k positions sont supérieurs à zéro, l'élément est probablement présent. Sinon, si au moins un compteur est à zéro, l'élément est certainement absent.

4. **Suppression d'un Élément** : Pour supprimer un élément, celui-ci est haché par les k fonctions de hachage. Les compteurs aux k positions correspondantes sont décrémentés de 1. Si un compteur atteint zéro, cela indique que l'élément n'est plus présent dans le filtre.

2.7.1.2.2 Avantage de CBF Les CBF offrent divers bénéfices par rapport aux filtres de Bloom traditionnels lorsqu'il s'agit d'évaluer les fréquences d'accès :

— Les CBF permettent de stocker une approximation de la fréquence d'occurrence de chaque élément, tandis que les filtres de Bloom classiques ne fournissent qu'une information binaire sur l'appartenance d'un élément.
— Les CBF supportent les opérations d'incrémentation et de décrémentation des compteurs, ce qui les rend adaptés au suivi dynamique des fréquences d'accès dans les systèmes de cache.
— Les CBF peuvent être mis à jour de manière efficace, ce qui est essentiel pour suivre les changements de popularité des éléments au fil du temps.

2.7.1.2.3 Limitations de CBF

— **Faux Positifs** : Comme le filtre de Bloom classique, le CBF peut produire des faux positifs, où il indique qu'un élément est présent alors qu'il ne l'est pas. Toutefois, les faux négatifs (indiquer qu'un élément est absent alors qu'il est présent) sont évités.
— **Augmentation de la Mémoire** : L'utilisation de compteurs au lieu de bits simples augmente l'empreinte mémoire, bien que cela soit nécessaire pour permettre la suppression et le suivi des fréquences.

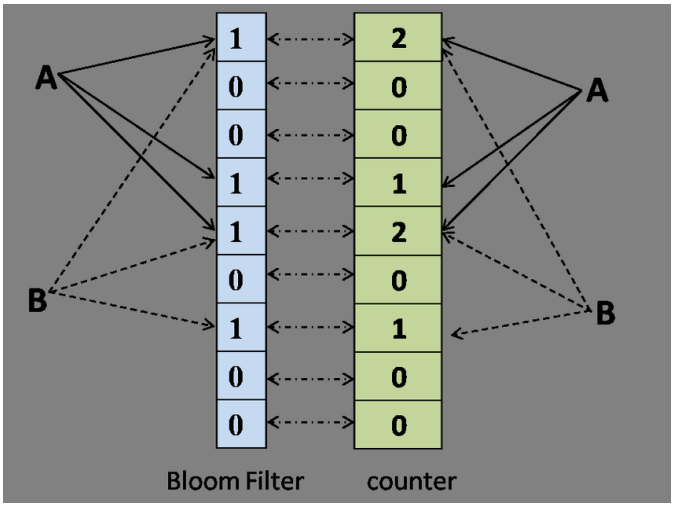


FIGURE 2.8: Description de Counting Bloom Filter [24]

2.7.1.3 Count-Min Sketch (CM-SKETCH)

CM-Sketch est une structure de données probabiliste utilisée dans TinyLFU pour stocker et estimer les fréquences d'accès des éléments de manière efficace en termes de mémoire et de calcul [21].

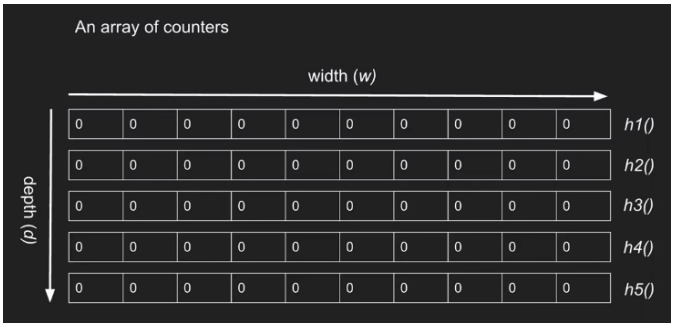


FIGURE 2.9: Tableau de compteurs

2.7.1.3.1 Principe de Fonctionnement Le Count-Min Sketch repose sur un tableau bidimensionnel de compteurs et plusieurs fonctions de hachage. Voici comment il fonctionne :

1. structure :
   — **Tableau de Compteurs** : Le Count-Min Sketch utilise un tableau de compteurs de dimensions d*w où d est le nombre de fonctions de hachage et w est la largeur de chaque tableau de hachage.

— **Fonctions de Hachage** : d fonctions de hachage indépendantes sont utilisées pour mapper chaque élément à une position dans chaque rangée du tableau de compteurs.

2. **Insertion d'un Élément** :

   Lorsqu'un élément x est inséré, il est haché par les d fonctions de hachage, chacune produisant une position différente dans les d rangées du tableau de compteurs. Les compteurs aux positions obtenues sont incrémentés de 1.

3. **Estimation de la Fréquence d'un Élément** : Pour estimer la fréquence d'un élément x, celui-ci est haché par les mêmes d fonctions de hachage pour obtenir d positions. La fréquence estimée est le minimum des valeurs des compteurs aux positions obtenues. Cela permet d'atténuer l'impact des collisions, où plusieurs éléments peuvent être mappés aux mêmes positions. CM-SKETCH permet une estimation rapide et peu coûteuse en mémoire, mais peut surestimer les fréquences en raison de collisions dans les fonctions de hachage [38].

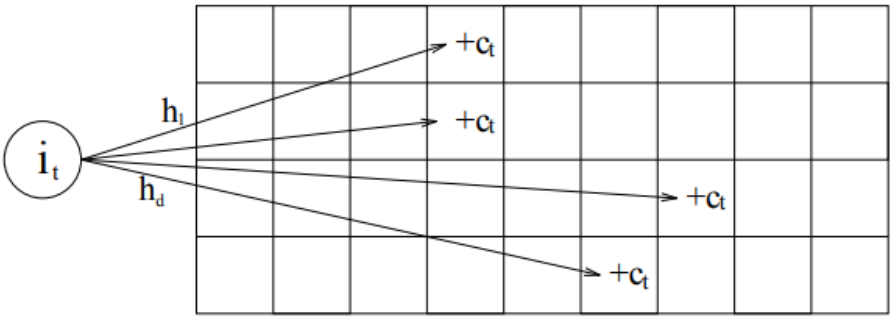


FIGURE 2.10: Vision global de CM-SKETCH

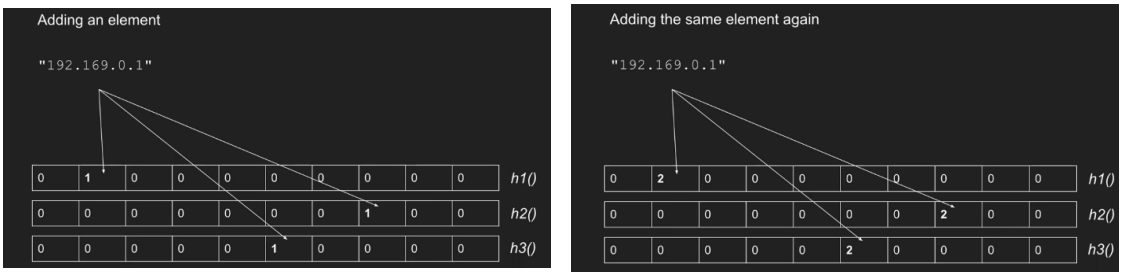


FIGURE 2.11: Ajouté le même élement deux fois dans CM-SKETCH

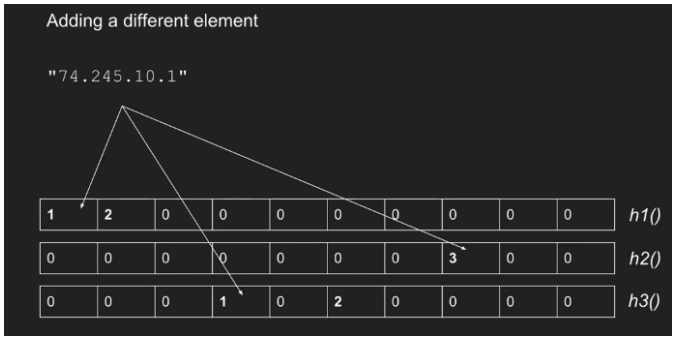


FIGURE 2.12: Ajouté un diffirent élement de qui est à figure 2.7.1.3.1 dans CM-SKETCH

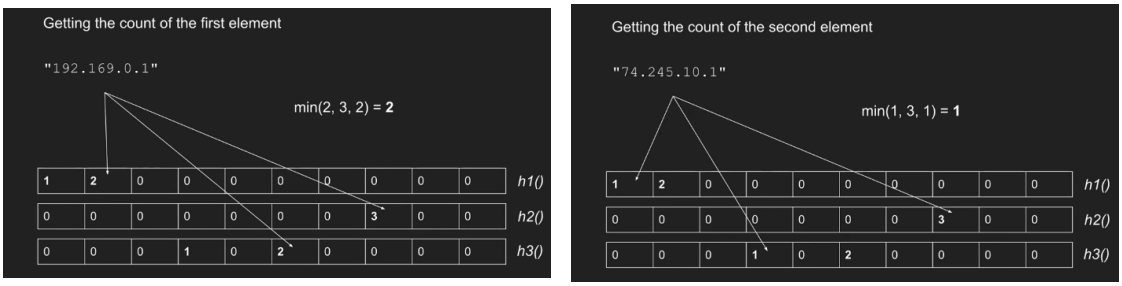


FIGURE 2.13: Obtenir le décompte du premier élément

2.7.1.3.2 Avantages du Count-Min Sketch

- **Efficacité en Mémoire** : Le Count-Min Sketch est très compact et nécessite moins de mémoire que des structures de données exactes comme les tables de hachage pour des estimations similaires. La mémoire utilisée est proportionnelle à dw où d et w peuvent être ajustés pour obtenir un compromis entre précision et utilisation de mémoire.
- **Vitesse** :Les opérations d'insertion et d'estimation sont très rapides, car elles impliquent seulement quelques calculs de hachage et des opérations sur les compteurs.
- **Simplicité de Mise en Œuvre** : Le Count-Min Sketch est simple à mettre en œuvre et ne nécessite pas de structures de données complexes.

2.7.1.3.3 Limitations du Count-Min Sketch

- **Faux Positifs** : Le Count-Min Sketch peut produire des surestimations de la fréquence en raison des collisions de hachage. Les collisions augmentent la valeur des compteurs de manière incorrecte, surtout dans des environnements à haute cardinalité.

— **Pas de Suppression** : Contrairement au Counting Bloom Filter, le Count-Min Sketch ne permet pas de décrémenter les compteurs de manière fiable, ce qui limite son utilisation dans des applications où des éléments doivent être supprimés.

2.8 Déroulement de Algorithme TinyLFU

TinyLFU combine le CBF et le CM Sketch pour obtenir un compromis entre précision et empreinte mémoire [21] :

— Le CBF est utilisé comme première étape pour tester rapidement la présence d'un élément dans les statistiques des fréquences.
— Si un élément est trouvé dans le CBF, le CM Sketch est consulté pour obtenir une estimation plus précise de sa fréquence.
— Les fréquences estimées sont utilisées par la politique TinyLFU pour prendre des décisions d'admission au cache.

Ainsi, TinyLFU optimise les performances du cache en conservant en mémoire les éléments les plus pertinents, c'est-à-dire ceux qui sont à la fois fréquemment accédés et récemment utilisés. Son processus de fonctionnement sophistiqué garantit une utilisation efficace des ressources du cache tout en maximisant les taux de réussite des accès.

2.9 Conclusion

Ce chapitre a offert une analyse détaillée des mécanismes d'admission au cache dans les réseaux Named Data Networking (NDN), en mettant en lumière les stratégies les plus connues telles que LCE, ProbCache, LCD, Betweenness et TinyLFU.

Nous avons porté une attention particulière à l'algorithme TinyLFU, remarquable par sa capacité à estimer la fréquence d'accès d'un élément avant de l'admettre dans le cache [21]. Cette approche permet à TinyLFU de prendre de bonnes décisions d'admission, ce qui conduit à une amélioration significative des performances globales du réseau [28].

Le prochain chapitre portera sur l'évaluation des performances de TinyLFU dans NDN en comparaison avec d'autres politiques d'admission. Cette évaluation ouvrira la voie à de nouvelles améliorations et adaptations de cet algorithme, contribuant ainsi à l'optimisation des performances de mise en cache dans les réseaux NDN.

Chapitre 3

SIMULATION ET INTERPRETATION

3.1 Introduction

À l'ère de l'Internet des objets (IoT) et de la croissance exponentielle des données, les réseaux NDN émergent comme une solution prometteuse pour relever les défis de la distribution de contenu à grande échelle grâce à leur système de mise en cache distribué. Dans ce contexte, les algorithmes d'admission au cache jouent un rôle essentiel dans l'optimisation des performances des réseaux NDN en déterminant intelligemment quelles données doivent être stockées dans les caches. C'est dans cette optique que nous avons mené une simulation dans un environnement NDN visant à évaluer et comparer les performances de différents algorithmes d'admission. L'objectif principal de cette simulation est d'analyser en détail le comportement et l'efficacité de quatre algorithmes d'admission au cache populaires dans le contexte des réseaux NDN : LCE, LCD, ProbCache et TinyLFU. Pour atteindre cet objectif, nous avons utilisé le simulateur CCNSIM, spécialement conçu pour les réseaux NDN. Cette simulation nous permet d'évaluer les performances des algorithmes d'admission au cache dans des conditions proches de la réalité, tout en offrant la flexibilité nécessaire pour explorer différentes configurations de paramètres.

3.2 Conclusion

Dans ce chapitre, nous avons analysé les performances de divers algorithmes d'admission au cache dans le cadre de la topologie NDNTestBed. Les algorithmes étudiés incluent TinyLFU, LCD, ProbCache et LCE. Nous avons évalué leurs performances en fonction de deux critères : le taux de réussite (C_hit) et la latence. Pour chaque algorithme, nous avons fait varier trois paramètres, à savoir le taux d'intérêt des consommateurs (λ), le facteur de popularité (α) et la taille du cache (CS, Cache Size), afin d'examiner leur impact sur les performances. Les résultats obtenus soulignent l'importance de sélectionner un algorithme d'admission de cache approprié pour optimiser les performances du réseau NDN. En comparant les performances des algorithmes, nous avons constaté que l'algorithme TinyLFU offre les meilleurs résultats en termes de taux de réussite (C_hit) et de latence.

Conclusion Génerale

Le modèle NDN marque une avancée révolutionnaire dans l'architecture des réseaux, en redéfinissant la manière dont les données sont transportées et accessibles. Contrairement aux réseaux classiques basés sur les adresses IP, le NDN met l'accent sur le contenu lui-même, permettant une distribution plus efficace et sécurisée des informations.

La mise en cache joue un rôle essentiel dans les routeurs NDN, visant à stocker temporairement des copies de données pour accélérer les temps d'accès et réduire la charge sur les serveurs d'origine. Cette pratique améliore la performance réseau, diminue la latence et économise la bande passante, en plaçant les données fréquemment demandées plus près des utilisateurs.

Dans le contexte de NDN, l'admission au cache revêt une importance encore plus grande. En fait, les politiques d'admission au cache représentent un pivot fondamental dans l'optimisation de la performance des réseaux NDN. Notre étude approfondie met en lumière plusieurs algorithmes d'admission au cache, tels que LCE, LCD, Probcache, Betweenness et TinyLFU, chacun offrant des perspectives uniques et des avantages distincts en matière de gestion de cache.

Nous avons constaté qu'un choix judicieux des politiques d'admission au cache peut peut avoir un impact significatif sur les performances de réseau. En intégrant ces stratégies, nous avons observé des améliorations substantielles en termes de latence, de bande passante et d'efficacité de la distribution des données.

Il est impératif que la communauté scientifique et les acteurs industriels collaborent étroitement pour exploiter pleinement ces politiques. L'intégration harmonieuse de ces politiques avancées promet non seulement une meilleure utilisation des ressources, mais aussi une expérience utilisateur enrichie et une robustesse accrue face aux défis émergents tels que la sécurité et la scalabilité.

En somme, le potentiel des politiques d'admission au cache des réseaux NDN est immense et inspirant, ouvrant la voie à une nouvelle ère de réseaux intelligents et adaptatifs, transformant notre interaction avec le monde numérique de manière extraordinaire et durable.

Bibliographie

[1] *A. broder and m. mitzenmacher, network applications of bloom filters : A survey, internet mathematics, vol. 1, no. 4, pp. 485-509, 2004, doi :10.1080/15427951.2004.10129096.*

[2] *A. m. i. z. l. . z. b. afanasyev, «ccnx 1.0 implementation : high-performance named-object networking, » ieee communications magazine, vol. 52, n° 11, pp. 118-124, 2014.*

[3] *Abdullahi, i. arif, a., 2016. cache-skip approach for information-centric network. arpn journal of engineering and applied sciences, 11, pp.3413-3418.*

[4] *Ahlgren, b., dannewitz, c., imbrenda, c., kutscher, d., ohlman, b., 2012. a survey of information-centric networking. ieee communications magazine, 50.*

[5] *Ahmad, h., zubair islam, m., haider, a., ali, r. and kim, h.s., 2021. intelligent stretch reduction in information-centric networking towards 5g-tactile internet realization, doi : 10.48550/arxiv.2103.08856.*

[6] *Alkhazaleh, n.s.s.a.a., 2020. 'a comprehensive survey of information-centric network : content caching strategies perspective', vol. 7, no. 1, issn-2394-5125.*

[7] *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, universite mouloud mammeri de tizi-ouzou, tizi-ouzou, 2016.*

[8] *Aubry, e. (2017). protocole de routage pour l'architecture ndn. thèse de doctorat. université de lorraine, École doctorale iaem lorraine - informatique, automatique, Électronique - Électrotechnique, mathématiques de lorraine. disponible à : https ://www.theses.fr/2017lorr0267.*

[9] *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.*

[10] *Bloom, b.h., 1970. space/time trade-offs in hash coding with allowable errors. communications of the acm, 13(7), pp.422-426, doi : 10.1145/362686.362692.*

[11] *Bose, p., guo, h., kranakis, e., maheshwari, a., morin, p., morrison, j., smid, m. and tang, y., 2008. on the false-positive rate of bloom filters. information processing letters, 108(4), pp.210-213, doi : 10.1016/j.ipl.2008.05.018.*

[12] *Broder, a. and mitzenmacher, m., 2004. network applications of bloom filters : A survey. internet mathematics, 1(4), pp.485-509, doi : 10.1080/15427951.2004.10129096.*

[13] *Burke, j., refaei, t., wang, l., zhang, b., zhang, l., 2018. a brief introduction to named data networking, doi : 10.1109/milcom.2018.8599682.*

[14] *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.*

[15] *Christensen, k., roginsky, a. and jimeno, m., 2010. a new analysis of the false positive rate of a bloom filter. information processing letters, 110(21), pp.944-949, doi : 10.1016/j.ipl.2010.07.024.*

[16] *Din, i., hassan, s., khan, m., guizani, m., ghazali, o., habbal, a., 2018. caching in information-centric networking : Strategies, challenges, and future research directions. ieee communications surveys tutorials, 20, pp. 1443-1474, doi : https ://doi.org/10.1109/comst.2017.2787609.*

[17] *Dirk, k., börje, o., stephen, f., bengt, a., holger, karl. "network of information (netinf)-an information-centric networking architecture, 2013. computer communications 36 no.*

[18] *Elídio, t. da, joaquim, m. h. de m., antónio, l. d. costa. (2022). ndn content store and caching policies : A performance evaluation. computers, 11(11), 171, doi :10.3390/computers11110171.*

[19] *Emiliano, d.c., mohamed, a.k., ersin, uzun. "privacy in content-oriented networking : T. and countermeasures, 2013. acm sigcomm computer communication review 43 no.*

[20] *G. einziger and r. friedman, counting with tinylfu : A simple and efficient cache admission policy, acm trans. storage, vol. 13, no. 4, pp. 1-17, 2017, doi : 10.1145/3149371.*

[21] *G. einziger, r. friedman, and b. manes, tinylfu : A highly efficient cache admission policy, in 2014 22nd euromicro international conference on parallel, distributed, and network-based processing, 2014, pp. 146-153, doi : 10.1109/pdp.2014.34.*

[22] *Goralski, w., 2017. the illustrated network how tcpip works in a modern network. morgan kaufmann.*

[23] *Graphstream project. *betweenness centrality*. available at : https ://graphstream-project.org/doc/algorithms/betweenness-centrality/.*

[24] *Gsvsse : Enabling efficient, secure, verifiable searchable symmetric encryption, 2020 *ieee transactions on knowledge and data engineering*, pp(99) :1-1, doi : 10.1109/tkde.2020.3025348.*

[25] *Guo, d., wu, j., chen, h., yuan, y. and luo, x., 2010. the dynamic bloom filters. ieee transactions on knowledge and data engineering, 22(1), pp.120-133, doi : 10.1109/tkde.2009.57.*

[26] *Jacobson, v., smetters, d.k., thornton, j.d., plass, m.f., briggs, n.h., braynard, r.l., 2009. networking named content. acm conext '09, doi : 10.1145/1658939.1658941.*

[27] *Marc, m., d., s., jose, g.-l.-a., 2007. contentcentric networking.*

[28] *N. megiddo and d. s. modha, arc : A self-tuning, low overhead replacement cache, in 2nd usenix conference on file and storage technologies (fast '03), 2003, pp. 115-130.*

[29] *Networking, n.d., 2012. named data networking.*

[30] *Oehlmann, f., 2013. contentcentric networking, doi : 10.2312/net-2013-02-*$1_0$6.

[31] *Paul v. mockapetris and kevin j. dunlap. development of the domain name system. in in proc. acm sigcomm, pages 123–133, 1998.*

[32] *Postel, j., 1981. internet protocol. rfc, 791, pp. 1-51, doi : https ://-doi.org/10.17487/rfc0791.*

[33] *Psaras, i., chai, w.k. and pavlou, g., 2012. 'probabilistic in-network caching for information-centric networks'. in : Acm sigcomm workshop on information-centric networking, pp.55-60.*

[34] *Rossini, g. ; rossi, d. coupling caching and forwarding : Benefits, analysis, and implementation. in proceedings of the 1st acm conference on information-centric networking, acm-icn '14, paris france, 24–26 september 2014 ; pp. 127–136.*

[35] *Sabir, z. et amine, a., 2020. ndn vs tcp/ip : Which one is the best suitable for connected vehicles ? in : Dos santos, s., maslouhi, m. et okoudjou, k. (eds) recent advances in mathematics and technology. cham : Birkhäuser, pp. 131-1421.*

[36] *Saxena, d., raychaudhury, v., suri, n., becker, c., cao, j. (2023). named data networking : A survey. computer networks, 215, 109332.*

[37] *Takemasa, j., koizumi, y., hasegawa, t., 2019. lightweight cache admission algorithm for fast ndn software routers. journal of information processing, 27, pp.125-134, doi : doi :10.2197/ipsjjip.27.125.*

[38] *Tarkoma, s., rothenberg, c.e. and lagerspetz, e., 2012. theory and practice of bloom filters for distributed systems. ieee communications surveys tutorials, 14(1), pp.131-155, doi : 10.1109/surv.2011.031611.00024.*

[39] *Tian, s., patrick, c., 2012. scalable ndn forwarding concepts issues and principles.*

[40] *Yamamoto, m. a survey of caching networks in content oriented networks. ieice trans. commun. 2016, 99, 961–973.*

[41] *Yingdi, y., haitao, z., eric, n., spyridon, m., yanbiao, l., alexander, a., lixia, zhang., 2018. an overview of security support in named data networking. ieee communications magazine 56, doi : 10.1109/mcom.2018.1700438.*

[42] *Zhang, l., afanasyev, a., burke, j., jacobson, v., claffy, k., crowley, p., papadopoulos, c., wang, l., zhang, b., 2014. named data networking. comput. commun. rev., 44, pp. 66-73, doi : https ://doi.org/10.1145/2656877.2656887.*

[43] *Zhang, m., luo, h. zhang, h., 2015. 'a survey of caching mechanisms in information-centric networking'. ieee communications surveys tutorials, pp.1-1, doi : 10.1109/comst.2015.2420097.*

[44] *Zheng, q., kan, y., chen, j. and wang, s., 2019. 'a cache replication strategy based on betweenness and edge popularity in named data networking'. in : **2019 ieee international conference on communications (icc)**, shanghai, china, 20-24 may 2019, pp.1-7, doi : 10.1109/icc.2019.8761900.*