REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE SAAD DAHLAB DE BLIDA 1
FACULTE DES SCIENCES

PROJET DE FIN D'ETUDE POUR L'OBTENTION DU DIPLOME MASTER

SPECIALITE :

SECURITE DES SYSTEMES D'INFORMATION

Mémoire réalisé par :

HACHAD Amine

KHANISSI Balaoua

THEME :

Etude et Implémentation des Mécanismes de Sécurité pour le Routage Centré Contenu

Organisme d'accueil : USDB

Promotrice : Mme Sana AROUSSI

Présidente du jury : Mme Boustia

Jury : Mme Boutoumi

Année Universitaire : 2016-2017

Remerciement

Je remercie d'abord ALLAH le tout puissant qui m'a guidé et qui m'a donné la force et la volonté de réaliser ce travail.

Mes pensées vont vers mes parents, mes sœurs, mes frères, mes amis.

C'est grâce à leur soutien que j'ai pu réaliser ce travail. Ils savent déjà combien je leur dois.

Comme on remercie notre promotrice Mme Sana AROUSSI de nous avoir pris en charge et pour son aide.

Nos remerciements les plus sincères à toutes les personnes qui auront contribué de près ou de loin à l'élaboration de ce mémoire ainsi qu'à la réussite de cette formidable année universitaire.

Enfin, nous tenons aussi à remercier les jurés pour avoir accepté d'examiner et de juger notre travail.

Merci à tous et à toutes.

Résumé

Résumé

Les Réseaux Centrés Information (ICNs) ont introduit des nouveaux concepts et idées dans le domaine de recherche des protocoles de routage de prochaine génération, proposant une approche alternative à la suite de protocoles TCP/IP bien connue et consolidée. Un ICN envisage un réseau de périphériques de mise en cache intelligents qui transmettent non seulement des bits d'un endroit à l'autre, mais aussi un support du réseau pour fournir aux utilisateurs finaux ce qu'ils sont vraiment intéressés : les données nommées. Cependant, bien qu'une grande partie de la littérature existante souligne les avantages de ce nouveau paradigme de réseau, nous nous concentrons sur certains problèmes de sécurité liés au routage centré contenu comme l'opportunité de créer des dénis de service distribuées (DDoS), communément appelées les attaques d'inondations d'intérêt (Interest Flooding Attack, IFA). Dans notre travail, nous avons étudié ce type d'attaque d'IFA ainsi que les solutions proposées pour lutter contre cette attaque, comme Traceback. Pour valider notre étude, nous avons simulé plusieurs scénarios d'attaque en utilisant le package CCN-lite sous le simulateur OMNeT++.

Mots clés : réseaux centrés sur l'information (ICN), routage centré contenu, sécurité, DDoS, spoofing, Traceback, CCN-lite.

Abstract

Information Centric Network (ICN) has introduced new concepts and ideas in the next generation routing protocols research area, proposing an alternative approach to the well-known and consolidated TCP/IP protocol suite. ICN envisions a network of smart caching devices that not only transport bits from one place to another but also support the network to provide end users with what they are really interested in: named data. However, while a large portion of the existing literature highlights the benefits of this new network paradigm, we focus on some specific security issues related to the opportunity of mounting distributed denial of service attacks (DDoS), commonly known as Interest Flooding Attack (IFA). In our work, we studied this type of attack of IFA as well as the solutions proposed to fight against this attack, like Traceback. To validate our study, we simulated several attack scenarios using the CCN-lite package under the OMNeT ++ simulator.

Keywords: Information Centric Network, Content Centric Network, Security, DDoS, Spoofing, Traceback, CCN-lite.

Sommaire

Liste des figures

Liste des tableaux

INTRODUCTION GENERALE

Introduction

Le succès d'Internet est largement dû à la quantité de contenus mis à disposition de tous ; Et l'accès à ces contenus est aujourd'hui le service dominant des usages de l'Internet. Cependant, l'architecture traditionnelle d'Internet, basée sur le modèle de communication hôte à hôte, ne permet pas la livraison de contenu à grande échelle. Pour remédier à ce problème, les chercheurs proposent un nouveau paradigme de communication centré sur le contenu où, plutôt que de connecter des hôtes distants comme le fait actuellement IP, le réseau gère directement les éléments d'information (contenus) que les utilisateurs veulent publier, récupérer ou échanger. Ce nouveau type de réseau est connu sous différentes appellations : réseau centré contenu (Content Centric Network, CCN) ou encore plus généralement : réseau centré information (Information Centric Networking, ICN).

Problématique

De plus de la livraison des contenus, l'ICN permet aussi de traiter d'autres limitations dans l'architecture actuelle d'Internet notamment la gestion de la sécurité. En effet, ils basculent vers la sécurisation de contenus au lieu de la sécurisation du chemin (l'infrastructure et des nœuds) des contenus. En conséquence, de nouvelles attaques sont apparues avec ce nouveau modèle de sécurité, en plus des attaques héritées qui peuvent avoir un impact sur l'ICN. Par exemple, nous trouvons plusieurs attaques liées au routage centré contenu telles que les attaques d'infrastructure, de source, de Blocage mobile, de Flooding, de timing, de Jamming, de Hijacking et d'Interception.

Objectifs :

Ce sujet consiste à étudier les problèmes de sécurité des ICNs, notamment les attaques potentielles liées au routage centré contenu, comme le déni de service distribué (DDoS) et l'inondation (Flooding), et ce dans le but de proposer des solutions.

Organisation :

Notre mémoire sera organisé comme suite :

- **Chapitre I :** Nous étudierons dans ce chapitre la limitation d'Internet et la notion de réseaux centrés sur l'information ICN et leur fonctionnement de base. Nous présenterons quelques projets ICN ainsi leurs principales fonctionnalités avec une concentration sur le routage centré contenu du projet CCN (Content Centric Network).
- **Chapitre II :** Nous parlerons dans ce chapitre de la sécurité dans les CCNs avec présentation d'une classification des attaques dans les ICNs.
- **Chapitre III :** Ce chapitre décrit en détail l'attaque d'Interest Flooding et les solutions proposées pour réduire les effets de cette attaque.
- **Chapitre IV :** La première partie de ce chapitre contient une présentation de l'environnement OMNET++ avec le package CCN-lite et la deuxième partie contient les simulations d'attaque au réseaux CCN avec essaie d'implémenter l'un des solutions proposées pour traiter ce problème.
- **Conclusion et Perspectives :** Nous terminerons avec une conclusion générale et perspectives
- **Annexe :** sur la sécurité des réseaux Informatiques et quelques notions de base importante.

Chapitre I :

Réseaux Centrés sur l'Information (Information Centric Networks, ICN)

I.1. Introduction

Le concept de réseau centré sur l'information (ICN) est une approche commune importante de plusieurs activités futures de recherche sur Internet. L'approche s'appuie sur la mise en cache dans le réseau, la communication multipartite par réplication et les modèles d'interaction qui découplent les expéditeurs et les récepteurs. L'objectif est de fournir un service d'infrastructure réseau mieux adapté à l'utilisation actuelle (en particulier, la distribution de contenu et la mobilité) et plus résistant aux perturbations et aux pannes.

Au cours de ce chapitre, nous allons aborder les réseaux centrés sur l'information. Ensuite, nous allons décrire et comparer les principaux projets ICN comme : DONA, NetInf, PSIRP et CCN. Nous terminons avec une description détaillée sur le routage dans les CCNs.

I.2. Limitation d'Internet

Internet a été créé dans les années 70s pour échanger des informations ou communiquer. La généralisation de l'utilisation d'Internet à l'échelle mondiale s'est opérée plus vite que quiconque aurait pu l'imaginer. L'évolution rapide de ce réseau mondial induit un bouleversement des interactions sociales, commerciales, politiques et même personnelles **[CIS 01]**.

Les utilisations d'Internet étaient relativement simples, comme rechercher des informations sur des sites web, parler avec des amis grâce à des logiciels de messagerie instantanée, envoyer/recevoir des mails, télécharger des fichiers depuis des serveurs FTP, etc. Pour ces usages, les communications dans les réseaux sont réalisées selon un modèle de bout-en-bout, en établissant des tunnels de communications, depuis un point de terminal (un utilisateur par exemple) vers un autre point (voir *Fig. I.1*).

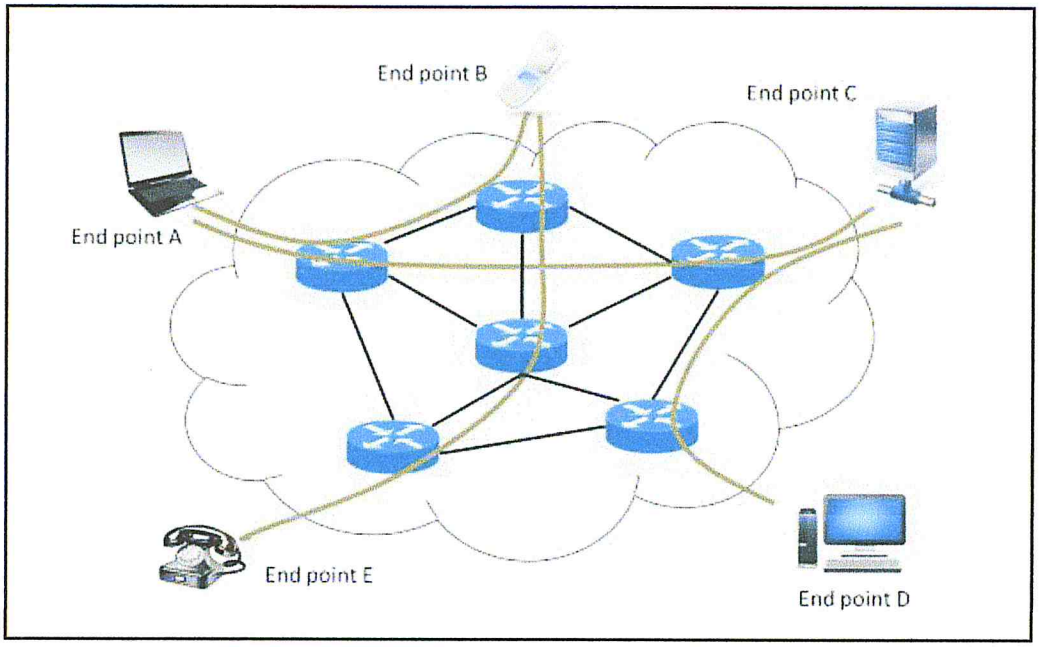


Fig. I.1. *L'architecture de réseaux IP* [WEI 14]

Si on analyse bien, on peut trouver que ce qui a changé est le concept fondamental des usages Internet. Ce qui intéresse les clients n'est plus l'endroit où il peut trouver (localisations) les informations, mais plus précisément les informations (contenus) elles-mêmes. Néanmoins, le modèle original d'Internet de type bout-en-bout n'est pas efficace pour de tels services de distributions de contenus. C'est pour cette raison que des applications P2P (Peer to Peer) et CDN (Content Delivery Networks) sont appliqués dans les réseaux. La communauté scientifique a formulé des propositions connues sous le nom de Réseau Centré sur l'Information (Information Centric Networks, ICN) pour changer la pile de protocoles réseau afin de transformer Internet en une infrastructure de distribution de contenu [SAN 15].

I.3. Fonctions de base des ICNs

Les ICNs proposent de changer l'Internet, qui est actuellement basé sur les localisations des serveurs avec des adresses bien définies, vers une architecture basée sur le nom des contenus (voir *Fig. I.2*), avec des fonctionnalités nativement intégrées comme le nommage indépendant de la localisation, le routage basé sur les noms de contenu, la faculté de cacher des contenus dans les réseaux, le multicast, la sécurisation des contenus,

la mobilité etc. Grâce à ces fonctionnalités, les ICNs sont plus efficaces pour délivrer des contenus aux utilisateurs avec une meilleure qualité et permettent aussi d'améliorer la gestion des capacités réseaux des fournisseurs des réseaux **[WEI 14]**.

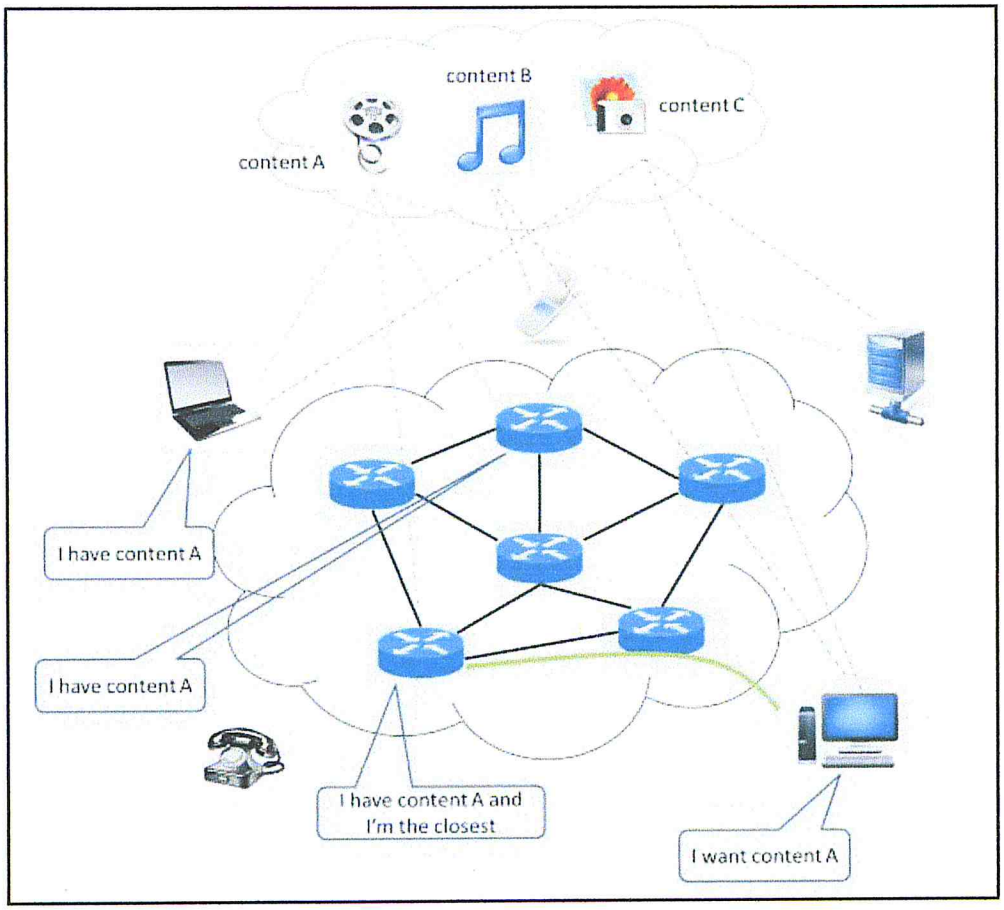


Fig. I.2. *L'architecture des réseaux ICN* [WEI 14]

Les principales fonctionnalités intégrées dans un ICN sont:

a. Nommage des contenus

Dans un ICN, l'unité de réseau de base est l'objet de contenu sur lequel toutes les activités de réseautage sont basées. , Un objet de contenu est identifié par un nom globalement unique qui est composé d'un nombre variable de composants, organisés dans une structure plate ou hiérarchique. [WEI 14].

b. Routage basé sur le nom de contenu

Le réseau ICN est un modèle basé sur le récepteur (receiver-driven en anglais). C'est-à-dire qu'un utilisateur exprime seulement son intérêt sur des contenus au réseau. Ensuite, c'est le réseau qui a pour charge de trouver les bons contenus et les meilleures sources pour ces contenus, en se basant sur leurs noms. Quand l'intérêt du client arrive finalement à une source de contenu, le contenu est délivré en suivant le chemin inverse du message d'intérêt jusqu'au client. Au final, le client est satisfait car il reçoit le contenu désiré, même s'il n'a pas connaissance de l'entité qui lui a fourni ce contenu. **[WEI 14]**

Dans notre travail, nous intéressons à cette fonction de l'ICN et en particulier à la sécurité de routage.

c. Cache des contenus

Une caractéristique importante des ICNs est la capacité de mise en cache de contenus directement dans les nœuds du réseau. Chaque morceau d'un contenu peut être mis en cache dans les nœuds de réseau se trouvant sur le chemin de la livraison du contenu, de sorte que les demandes ultérieures pourront être satisfaites plus rapidement, directement par les caches des nœuds ICN. Les paquets perdus pourront aussi être aussi récupérés plus rapidement par des retransmissions directes depuis les caches les plus proches.

d. Sécurité du contenu

ICN fournit une garantie de sécurité auto-protégée via des contenus cryptés et auto certifiés, et non pas via des connexions de communication sécurisées comme IP. Seuls les utilisateurs autorisés peuvent déchiffrer les contenus.

e. Mobilité du contenu

Étant donné qu'ICN fonctionne sur un modèle non-connecté (il n'y a pas de connexion établie dans un réseau ICN), la mobilité des utilisateurs ne modifie pas le comportement des réseaux ICN. Leurs demandes, issues de différents endroits à différents moments, sont traitées indépendamment par les réseaux ICN, chacune comme une requête unique.

I.4. Principaux projets ICN

Dans les dernières années, plusieurs projets ou solutions ont été mises en œuvre par des équipes de recherche différentes, avec l'objectif de réaliser et déployer un ICN ; Les plus connus sont DONA **[KOP 07]**, NetInf **[SAI 13]**, PSIRP **[LAG 12]** et CCN **[JAC 09]**. Dans ce qui suit, nous allons décrire les principales fonctionnalités (nommage et routage) de chacun de ces projets. Notre description s'achèvera avec une comparaison et une discussion pour justifier notre choix d'un projet dans notre travail.

I.4.1. NetInf

Network of Information (NetInf) est une architecture de réseau proposée par le projet européen FP7 appelé 4WARD et son projet de suivi appelé SAIL **[SAI 13].**

La structure de réseau NetInf est basée sur un système de résolution de dénomination (NRS - *Naming Resolution System*) et un mécanisme de routage de table de hachage multiple (MDHT - *Multiple Distributed Hash Table*). Dans NetInf, les concepts de représentation de contenu et l'objet de données d'un contenu sont clairement distincts.

Un objet d'information (*IO - Information Object*) sert à référencer directement un objet de contenu et un objet de niveau de bit (*BO - Bit-level Object*) est la donnée de contenu elle-même. Un *IO* contient trois informations : un identifiant globalement unique du contenu, un ensemble de métadonnées et un objet de données (*DO - Data Object*) qui peut être considéré comme une référence de la charge utile du contenu réel - le BO. L'identifiant de contenu contient le type de contenu (comme le document texte, le fichier audio, l'image, la page Web ou les flux en direct) et le hash de la clé publique du propriétaire (ou éditeur). Le champ de métadonnées contient une méta-liste qui fournit des informations sémantiques sur le contenu qui peut être utile pour le service de recherche, comme le débit et le codec d'un enregistrement audio ou l'auteur et l'abstrait d'un document.

Les rôles des auteurs de contenu et des éditeurs dans NetInf sont différents. Les auteurs créent, signent et modifient les IO. Un même IO peut avoir différentes versions et différentes versions peuvent avoir différents auteurs et différentes signatures. Les éditeurs ne sont pas autorisés à modifier ou à signer les IO, mais ils peuvent redistribuer le contenu.

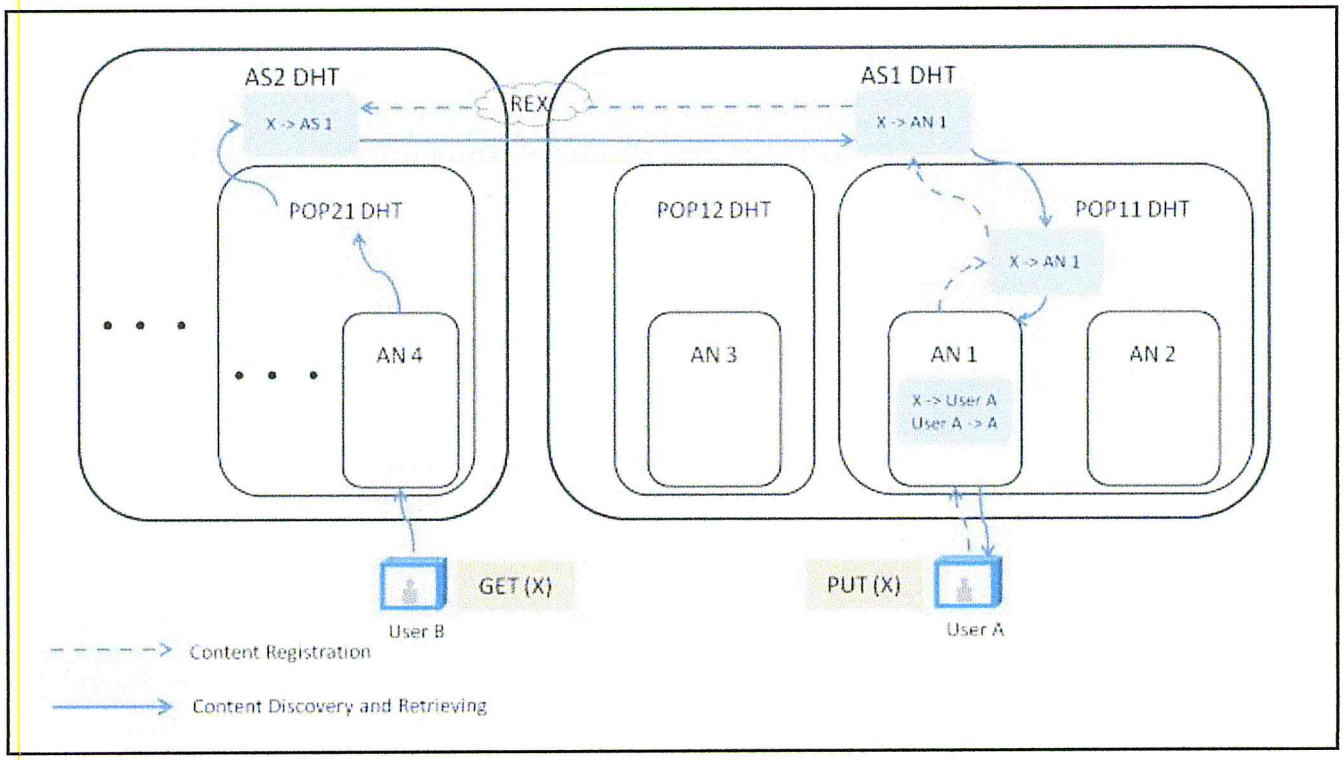


Fig. I.3. *Schéma de routage de NetInf* **[ATH 15]**

Le système de résolution de noms NetInf (NRS) prend un identifiant de contenu ou un ensemble d'attributs qui décrivent l'objet recherché comme entrée et renvoie un ensemble d'enregistrements de liaison pour les IO qui correspond à l'entrée. Les IOs incluent une référence (DOs) qui, directement ou indirectement, peut être utilisée pour récupérer le BO. Cela signifie que dans le système NetInf, une résolution en deux étapes est possible. L'application ou l'utilisateur peut choisir dans la liste des OI renvoyés celui à sélectionner pour demander les BO correspondantes, en fonction de différents critères (coût, vitesse de téléchargement, définition, qualité, etc.).

NetInf définit différentes zones de résolution de noms de niveau qui sont réalisées par un (MDHT). Chaque zone est responsable de stocker de manière persistante une BO avec l'IO d'identification correspondante. Lorsqu'un client demande un objet, une première recherche DHT est effectuée au premier niveau (par exemple, sa zone de réseaux d'accès). Si elle n'est pas trouvée, une autre recherche DHT est émise à un niveau supérieur (par

exemple, zone POP). Si elle n'est toujours pas trouvée, une autre recherche DHT est effectuée à un niveau supérieur (par exemple, niveau de domaine), etc. Lorsque la recherche DHT est réussie à un niveau donné, le résultat est retourné au client. Il est à noter que, malgré le routage par hop-by-hop et la résolution locale, le niveau DHT supérieur doit contenir des liaisons pour toutes les données enregistrées dans le domaine, avec une évolutivité possible et éventuellement des problèmes de performance. La Figure I.3 illustre le schéma de routage de l'architecture NetInf.

I.4.2. DONA

Data Oriented Network Architecture (DONA) une architecture orientée donnée qui a été proposée en 2007 par Koponen et al **[KOP 07]**. Le projet DONA se base sur l'utilisation de noms auto certifiés et incorporer des fonctionnalités de cache. Le noyau du routage dans DONA est un système hiérarchique de résolution de dénomination avec un nom de contenu plat.

Le problème de nomination dans DONA est similaire à celui de NetInf. DONA utilise une structure de nommage P : L. La partie P est le hash de la clé publique du propriétaire du contenu (le concept de principe dans DONA). Et le L est l'étiquette de contenu attribuée par le propriétaire. Les propriétaires de contenu ont la responsabilité d'assurer l'intégralité du nom P : L globalement unique.

Cependant, le routage dans DONA est différent par rapport au routage de NetInf. DONA applique un système hiérarchique de résolution de nom de contenu (*Fig. I.4*), les Poignées de Résolution (RHs - Resolution Handles). Chaque nœud RH a une base d'informations contenant trois tuples, le nom du contenu P : L, le prochain saut et la distance. Le prochain saut est d'où le nœud reçoit la publicité sur le nom du contenu.

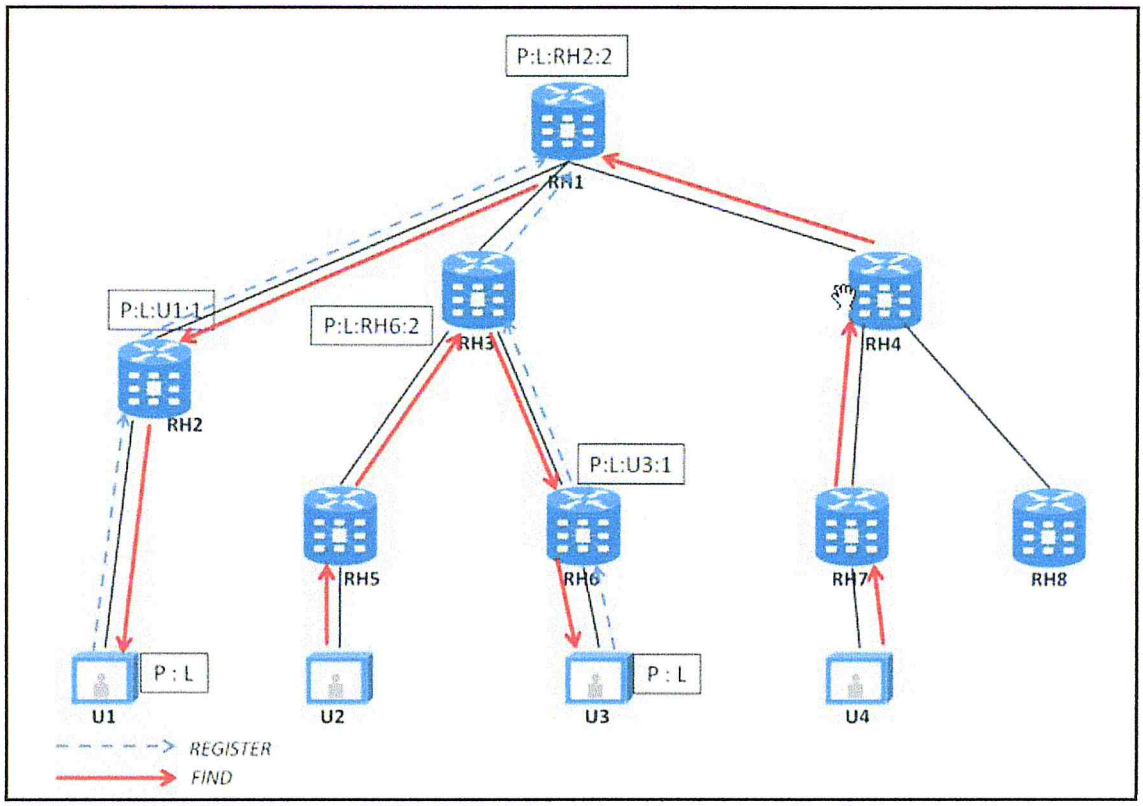


Fig. I.4. *Schéma de routage de DONA* **[ATH 15]**

Le routage DONA contient deux messages : FIND et REGISTER. Tous les deux messages sont directement basés sur le nom de contenu plat. Lorsqu'un RH reçoit un message REGISTER, il ajoutera <P : L, prochain saut, distance> dans son tableau de registre pour un nouveau message d'arrivée, ou Mettre à jour la prochaine et la distance pour une entrée existante si le nouveau message arrivé à une distance plus courte. Ensuite, la RH transmet ce message à son parent (s). Enfin, l'inscription au contenu se terminera à la plus haute (s) RH (s). Lorsqu'une HR reçoit le message FIND, il examinera le nom du contenu dans sa table de registre locale. S'il trouve une entrée correspondante, le message FIND sera transmis à travers le prochain saut de l'entrée correspondante. Sinon, la RH transfère le message FIND à son (ses) RH (s) parent(s). Comme le chemin du message FIND, chaque HR qui est sur le chemin ajoute le FIND localement, donc une fois que le FIND arrive au conteneur de contenu le plus proche, l'objet de contenu sera renvoyé par le chemin inverse de chemin FIND.

I.4.3. PSIRP

Publish Subscribe for Internet Routing Paradigm (PSIRP) **[LAG 12]** est un projet européen qui a débuté en 2008 et s'est terminé en 2010. PSIRP a proposé une ardoise propre d'architecture ICN basée sur une solution d'abonnement aux éditeurs.

PSIRP a appliqué la même structure de dénomination P : L que le DONA et NetInf. Le nom du contenu est appelé identificateurs de ressource (RIds). Le réseau PSIRP comprend un concept de base qui s'appelle Scopes, qui est identifié avec les identificateurs de scope (SIds). Les Scopes contrôler les caractéristiques d'un contenu, tel que l'accès droit, autorisations, disponibilité, accessibilité, réplication, persistance et les ressources en amont. La publication du contenu (publish) et la demande de contenu (subscribe) d'un contenu sont basé sur une paire de composition de <SId, RId>.

Le processus de routage PSIRP comprend quatre unités importantes :

Nœuds de Rendez-vous (RN), Nœuds de topologie (TN), Nœuds de Branchement (BN) et Nœuds de transfère (Forwarding Node, FN). L'ensemble des réseaux PSIRP est divisé en Domaines, qui est similaire au Système Autonome de l'Internet actuel (***Fig. I.5***). Chaque domaine contient un RN, un TN, un BN et plusieurs FN. Le RN de chaque domaine est en charge de la correspondance entre les éditeurs de contenu et les abonnés, la localisation des publications de contenu et des domaines. Chaque RN peut avoir son propre système de résolution de noms. Toutes les RNs de chaque domaine sont interconnectés avec DHT dans une Interconnexion RendezVous (RI) globale qui fait les étendues de chaque domaine sont globalement accessibles. Le TN est chargé de la gestion de la topologie de réseau intra-domaine et de l'équilibrage de charge. Il échange également l'information vectorielle du chemin d'accès avec les autres TNs inter-domaines. Le BN construit une carte de routage pour acheminer les intérêts des abonnés vers les conteneurs de contenu inter-domaine ou intra-domaine en utilisant la topologie maintenue par le TN. Enfin, le rôle des FN est d'utiliser une implémentation de transmission basée sur le filtre Bloom pour réaliser le renvoi de contenu à partir du conteneur de contenu aux abonnés. Le filtre Bloom qui s'appelle identifiant de Forwarding (FId) est accumulé pendant la livraison de l'abonnement.

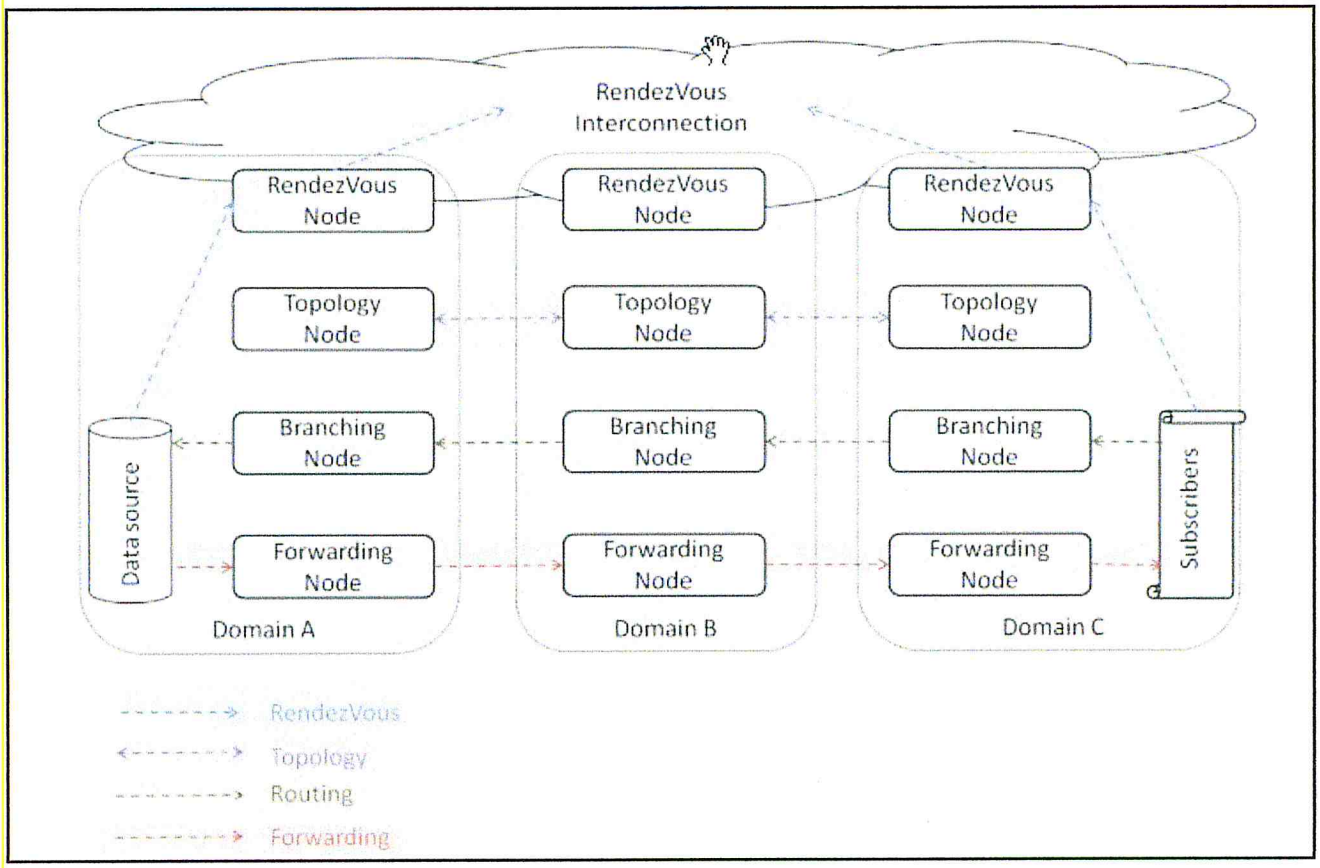


Fig. I.5. *Schéma de routage de PSIRP* **[ATH 15]**

Pour résumer, dans PSIRP, un abonné exprime son abonnement d'un contenu à la RN locale de son domaine pour obtenir le contenu emplacement du conteneur. Le BN utilise la topologie de routage qui est obtenu par le TN pour transmettre l'abonnement au contenu récipient. Le paquet d'abonnement cumule le chemin de retour dans le filtre Bloom et construit la FId. Enfin, les FN utilisent les FId pour retourner le contenu requis à l'abonné.

I.4.4. CCN

Van Jacobson propose la nouvelle architecture d'Internet orientée contenu nommée ***Content Centric Networking (CCN)*** **[JAC 09]**. Au lieu d'identifier les machines comme dans l'Internet d'aujourd'hui, l'architecture CCN (autant de propositions ICN), identifie les objets de données avec des noms. Les objets de données sont ensuite divisés en plusieurs paquets, chacun identifiant en utilisant un nom globalement unique. Les noms de CCN sont composés d'un nombre variable de composants, organisés dans une structure hiérarchique. Les noms structurés hiérarchisés, en ce qui concerne les noms à plat, ont l'avantage qu'ils peuvent être agrégés dans des préfixes, ce qui réduit considérablement la taille de la table de transfert sur le routeur d'un CCN, afin d'accélérer le processus d'acheminement (détaillé dans la section suivante). Comme présenté dans **[JAC 09]**, la figure I.6 montre un exemple du schéma de dénomination dans lequel un fichier vidéo est identifié par le nom ***/parc.com/ video/ WidgetA.mpg/ _v<timestamp>/*** et ses morceaux avec des noms du type **/parc.com/ video/ WidgetA.mpg/ _v<timestamp>/ _s_id** où **_s_id** représente l'identifiant du segment (paquet de données)

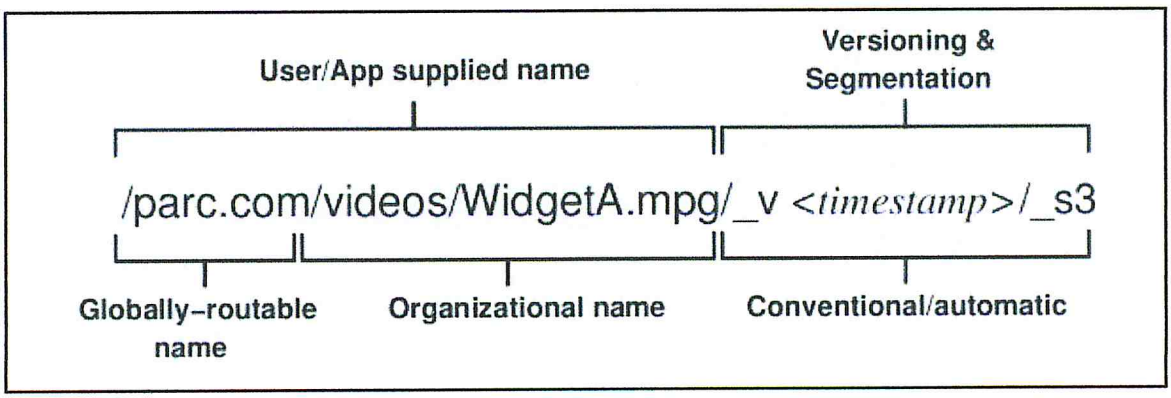


Fig. I.6. *Exemple de nom hiérarchique de CCN.* [JAC 09]

De plus, l'architecture du CCN permettrait une recherche simple et directe d'un contenu sans avoir à identifier son détenteur, comme dans le cas des réseaux IP. Un utilisateur cherche une donnée à travers son nom. Dès lors que la requête est lancée, une demande sous forme d'un paquet dit "Intérêt" est envoyée à son routeur d'accès. Si la donnée n'est pas présente dans le Content Store (CS) de ce routeur, la requête se propage au fur et à mesure dans le réseau. Une fois la donnée trouvée, elle suit le chemin inverse de la requête

de recherche jusqu'à l'utilisateur final et sera stockée dans un CS dans les routeurs CCN intermédiaires.

Cette architecture offre plusieurs possibilités de disponibilité indépendamment de l'adresse d'une machine. La sécurité est associée directement aux données et pas aux "conteneurs" (liens, routeurs, serveurs, ...) ce qui permet d'ajuster de manière très flexible le niveau de sécurité à la nature du contenu en question. Plus intéressant encore, les contenus ne sont plus associés à des conteneurs précis mais peuvent être dupliqués à volonté et stockés notamment dans des mémoires caches au sein du réseau. On discutera en détail le fonctionnement et le routage de CCN.

I.4.5. Comparaison

Dans le tableau suivant, nous comparons les 4 projets selon les fonctions de base ci-dessus.

	DONA	**NetInf**	**PSIRP**	**CCN**
Début/fin de projet	- 2007	- 2010 / 2013	- 2008 / 2010	- 2009
Nommage	- Plat	- Plat	- Plat	- Hiérarchique.
Routage	- Basé sur le nom.	- Basé sur le nom. - Résolution des noms	- Basé sur le nom. - Utilise modèle de résolution appelé « point de rendez-vous ».	- Basé sur le nom.
Cache	- Les RHs peuvent être activés avec un mécanisme de mise en cache universel.	- On-path caching - Off-path caching. - Peer caching.	- Les caches multiples d'un objet peuvent être maintenues en fonction de la portée du point de rendez-vous pour l'identifiant associé à l'objet.	- La mise en cache du contenu dans le CS d'un nœud. - Le stockage des paquets est possible à chaque nœud NDN.

Mobilité	- L'architecture DONA s'appuie sur les protocoles de transport existants, c'est-à-dire TCP, pour fournir les mécanismes d'acheminement et d'autres fonctionnalités de transport	- Différents mécanismes de transmission sont utilisés pour récupérer un objet de données, un localisateur ou des conseils de redirection en utilisant des messages de demande / réponse.	- Le processus de transfert de base repose sur les filtres bloom. - Chaque objet a un nom dérivé algorithmiquement unique à partir du nom d'origine, ce qui permet de gérer le contrôle de flux	- L'architecture CCN ne fournit aucune fonctionnalité de couche de transport. - Est fournie par l'application ou certaines bibliothèques de support.
Sécurité	- L'espace de noms auto-certifié utilisé dans les attributs DONA à l'intégrité des noms-données	- L'espace de noms auto-certifié. - la sécurité de l'objet est fournie par la cryptographie de clé publique	- L'espace de noms auto-certifié. - cryptographie à courbe elliptique (ECC) pour vérifier la signature et l'authentification	- L'éditeur signe cryptographique-ment chaque paquet de données
Code source de simulation		- openNetInf - NetInf (nilib) - GIN	- Blackadder	- CCN-lite. - SCoNet. - ccnSim . CCNPL-Sim - Mini-CCNx

Tab. I.1. *Comparaison entre des principaux projets ICN.*

Bien que ces architectures ICN (NetInf, DONA, PSIRP et CCN) diffèrent dans leurs détails, ils partagent plusieurs propriétés fondamentales : nom unique pour le contenu, routage basé sur le nom, mise en cache et assurance de l'intégrité du contenu.

En plus des autres initiatives prises par la communauté de recherche pour les futures architectures Internet, le CCN a récemment fait l'objet d'une attention particulière. Nous

pouvons constater cela du nombre des codes sources disponibles qui permettent de simuler et d'implémenter les différents protocoles. Pour cette raison, nous nous intéressons à ce projet et à ce type de réseau CCN pour étudier les problèmes/solutions de sécurité du routage centré contenu. Par conséquent, nous détaillons le routage dans le CCN dans la section suivante et la sécurité dans le CCN dans le chapitre suivant.

I.5. Routage dans CCN

L'idée générale de CCN est d'adapter les messages envoyés sur Internet à ce qu'ils sont vraiment : le contenu. Au lieu de la restriction aux communications de bout en bout entre les paires d'utilisateurs, CCN permet un échange de messages beaucoup plus flexible et efficace.

Par rapport à l'Internet actuel, les routeurs sont très différents (voir *Fig I.7*). Ils contiennent essentiellement trois tables : Le stockage de contenu (Content Store), la table (PIT) et la table (FIB) **[WEI 14]**.

- ***Content Store (CS)*** **:** le CS est un cache (ou une mémoire tampon) installé dans les nœuds CCN. Lorsqu'un nœud reçoit des données (message Data), selon les stratégies de caches définies, le nœud peut sauvegarder une copie de ces données dans son CS pour répondre aux demandes ultérieures.
- ***Pending Interest Table (PIT)*** **:** la table PIT a deux fonctionnalités principales. La première est qu'elle mémorise temporairement des messages d'intérêt « Interest » que le nœud reçoit avant de les transmettre ensuite au nœud suivant. Grâce à cette table, en retour des données, le paquet Data peut suivre les chemins inverses et finalement arriver jusqu'aux clients demandeurs. Le deuxième rôle de PIT est d'éviter de multiple envoi des mêmes messages Interest. Lorsque plusieurs messages Interest qui demandent un même contenu arrivent sur un nœud, seul le premier est renvoyé pour chercher le contenu ; les autres restent dans ce nœud et attentent la réception du contenu. Une fois reçues, les données seront retournées sur chacune des faces présentes dans la PIT.
- ***Forwarding Information Base (FIB)*** **:** la table FIB de CCN est similaire à celle d'IP. Elle est utilisée pour gérer les informations de transfert des paquets Interest vers

des sources qui ont les contenus demandés. La table FIB est remplie par des publications de contenus qui sont publiés par des fournisseurs de contenu.

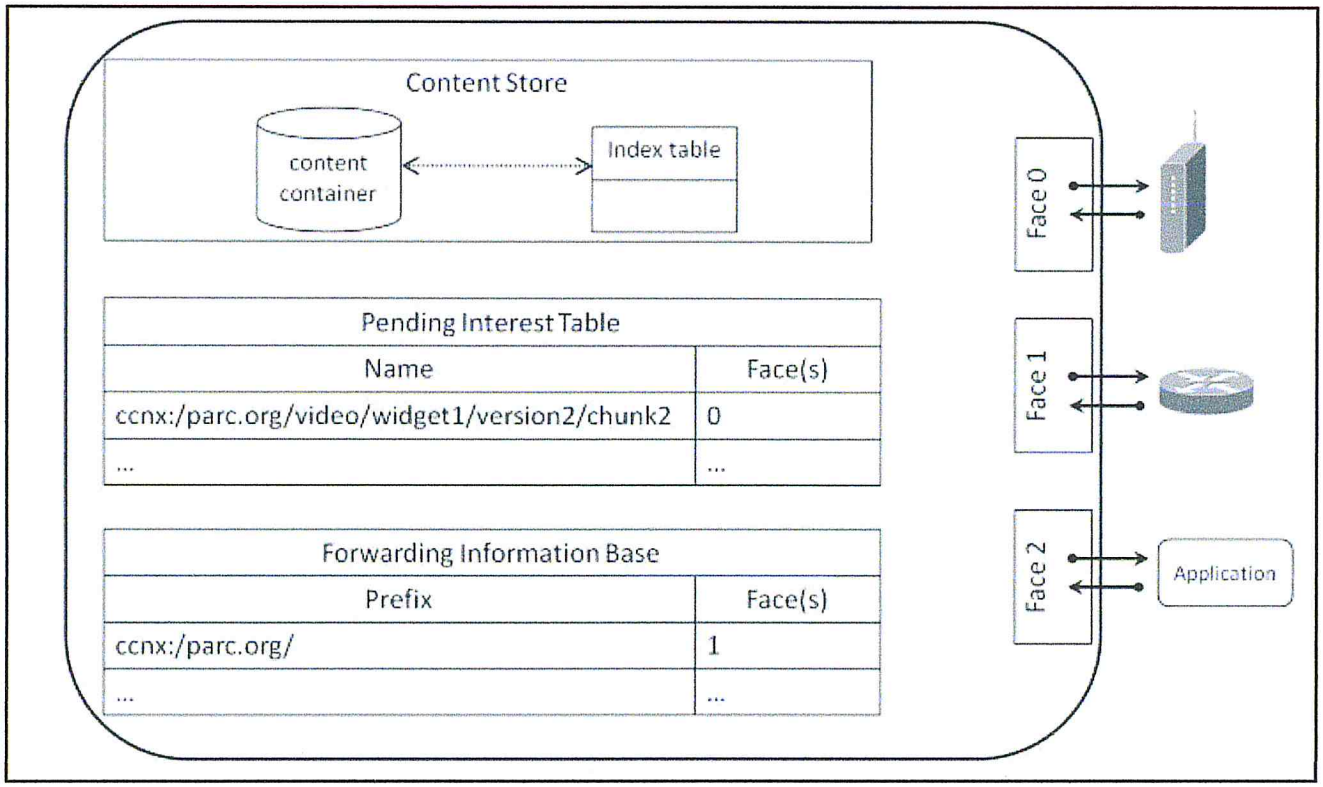


Fig. I.7. *La structure d'un nœud CCN.* **[WEI 14]**

Dans un CCN, il n'y a nul besoin d'acheminer les adresses source et destination à travers le réseau pour récupérer la donnée. Le format des paquets Intérêts et Data est explicité à la *Fig. I.8.*

A la réception d'un Intérêt par un nœud, ce dernier vérifie si le chunk demandé existe dans son Content Store. Si c'est le cas, le paquet Data sera envoyé à l'interface demandeuse. Sinon le chunk demandé sera recherché dans le PIT. S'il est trouvé, l'interface demandeuse sera rajoutée au PIT. Si les deux bases de données ne fournissent aucune information, on cherchera dans le FIB si une entrée matche avec le chunk recherché. Alors le paquet Interest sera acheminé vers les interfaces conduisant à la donnée. La table PIT sera mise à jour avec une nouvelle entrée pour le chunk en question.

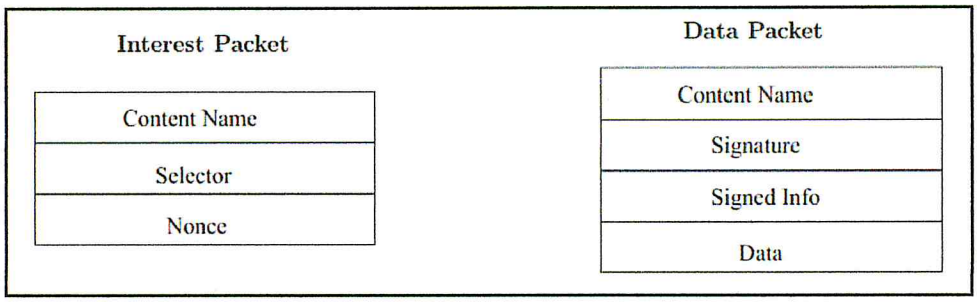


Fig. I.8. *Format des paquets CCN*

Les contenus sont divisés en morceaux (chunks), chaque morceau a typiquement la taille d'un paquet IP. CCN respecte le déroulement logique d'une requête : un utilisateur demande une donnée en émettant des paquets de type "Interest" et reçoit en retour des paquets de données de type "Data". A chaque paquet Interest correspond un seul paquet Data et chaque paquet Data correspond à un Chunk.

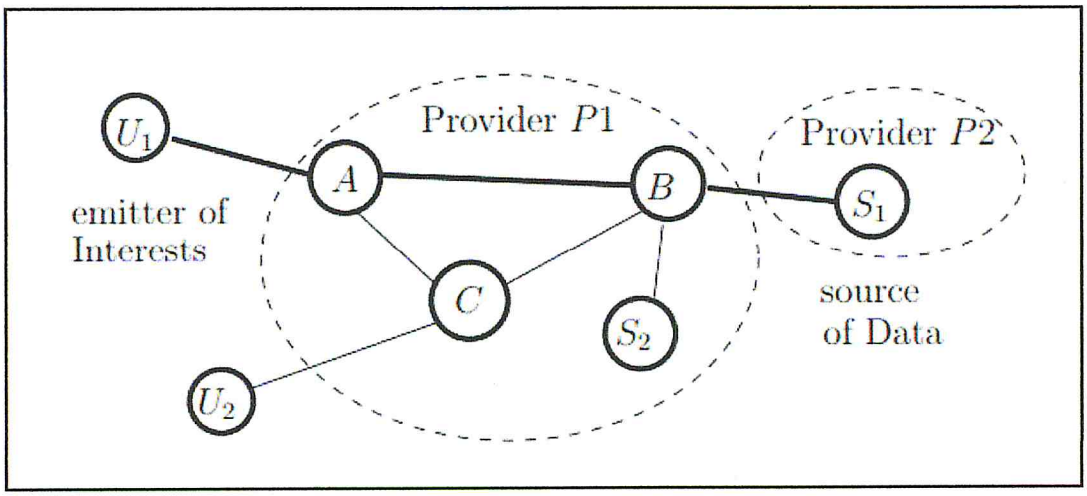


Fig. I.9. *Un segment du réseau CCN reliant un utilisateur U1 à une source S* **[NAD 14]**

La *Fig. I.9* représente un segment d'un réseau CCN. Pour récupérer des données du fournisseur **P2**, l'usager **U1** envoie des paquets "Intérêt" pour le contenu demandé au travers des routeurs **A** et **B**. Supposant que les Content Stores de **A** et **B** ne contiennent pas le document demandé, les paquets Data suivent le chemin inverse de **S1** vers **U1** en passant par **B** et **A**.

A la réception d'un paquet Data par un nœud, une recherche est effectuée dans le Content Store. Si une entrée matche, alors le paquet reçu est supprimé, car ceci implique que le chunk est déjà livré à toutes les interfaces demandeuses. Sinon la donnée sera

recherchée dans le PIT. Si une entrée matche avec la donnée reçue, elle sera acheminée vers les interfaces demandeuses. Le chunk sera typiquement stocké en même temps dans le Content Store.

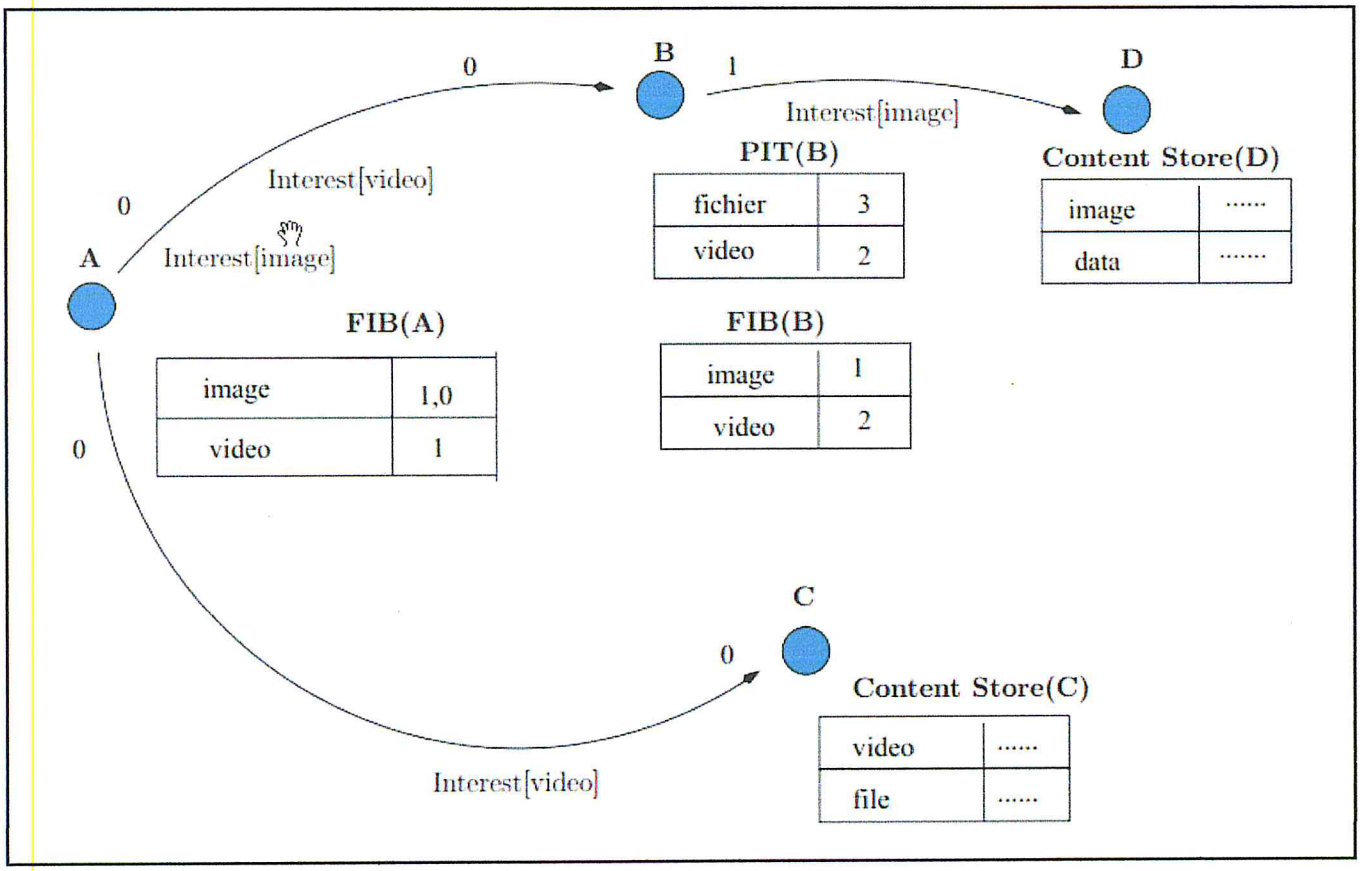


Fig. I.10. *La recherche des données dans CCN* **[NAD 14]**

Dans l'exemple de la *Fig. I.10*, le nœud **A** cherche les chunks "vidéo" et "image". Le FIB du nœud **A** indique que les paquets Intérêts doivent être acheminés vers l'interface **0** et **1** pour l'image, et vers l'interface **1** pour la vidéo. A la réception de l'Intérêt demandant la vidéo par le nœud **B**, ce dernier ignore l'intérêt reçu car le PIT contient déjà une entrée.

Cette entrée est mise à jour. Cependant, quand le nœud **B** reçoit l'intérêt demandant l'image, il l'envoie à l'interface **1** indiquée par le FIB. Le nœud **D**, par la suite, achemine la donnée vers le nœud **A**. Cette donnée sera stockée dans tous les Content Stores des nœuds l'ayant reçu ou transité.

I.6. Conclusion

Dans ce chapitre, nous avons discuté la limitation du réseau actuel d'Internet et les motivations de passer à la nouvelle architecture ICN. Nous avons présenté aussi sur les principaux projets d'ICNs, ses différentes fonctions de base de chaque projet et une petite comparaison entre ces projets. Enfin, nous avons détaillé le routage dans le CCN qui nous intéresse dans notre travail pour étudier ses problèmes de sécurité. La sécurité dans les CCNs fait l'objet du chapitre suivant.

Chapitre II :

Sécurité dans les CCNs

II.1. Introduction

ICN améliore plusieurs faces de l'expérience d'utilisateur ainsi que la sécurité, la confidentialité et les contrôles d'accès. Cependant, il pose de nouveaux défis en matière de sécurité. Dans ce chapitre, nous allons présenter les problèmes de sécurité de cette nouvelle architecture. Ensuite, nous présentons les solutions de sécurités offertes par l'architecture CCN.

II.2. Attaques dans les ICNs

II.2.1. Classification

Dans la littérature, plusieurs travaux (**[ESL 15] ,[REZ 16]**) ont essayé de recenser les différentes attaques ICN et de proposer une classification. Parmi ces travaux, nous nous sommes intéressés à l'article **[ESL 15]** qui classe les attaques ICN (nouvelles et anciennes) en quatre catégories, comme le montre dans la *Fig. II.1* : nommage, routage, mise en cache et autres attaques diverses. Cette classification dépend de la cible principale de l'attaquant. Bien que chaque attaque soit incluse dans une seule catégorie, elle peut également influer sur d'autres catégories. Par exemple, flooding et les attaques unpopular requests (requêtes impopulaires) affectent le routage et la mise en cache ICN. Dans une attaque flooding, L'objectif principal de l'attaquant est de surcharger et d'échapper les ressources de routage et par conséquent, elle affecte le système de mise en cache. Dans unpopular requests, la cible principale de l'attaquant est de violer la pertinence du cache et par conséquent, elle affecte le système de routage.

Les catégories proposées sont brièvement présentées dans les quatre paragraphes suivants :

- ***Les attaques liées au nommage*** : Les architectures ICN sont plus menacées par rapport à la vie privée vu que les demandes de contenu sont visibles pour le réseau. De nombreux attaquants tentent de surveiller l'utilisation d'Internet. Une architecture ICN offre plus d'accès aux demandes des utilisateurs ce qui augmenterait le contrôle des attaquants sur le flux d'informations et rendrait les informations de blocage beaucoup plus faciles. Dans les attaques liées à la nomination dans ICN, un attaquant tente d'empêcher la

distribution d'un contenu spécifique en bloquant la livraison de ce contenu et / ou en détectant qui demande ce contenu.

- ***Les attaques liées au routage :*** La distribution de contenu ICN dépend d'une publication et d'un abonnement asynchrone, ce qui ajoute des efforts supplémentaires pour assurer la cohérence entre les états de données distribuées. Certaines attaques comme jamming et la synchronisation visent à échouer cette cohérence de l'état, ce qui peut entraîner des flux de trafic indésirables et / ou un déni de service. D'autres attaques, comme l'infrastructure et les attaques par inondation (flooding), essayent d'épuiser les ressources comme la mémoire et le pouvoir de traitement qui sont utilisés pour soutenir, maintenir et échanger des états de contenu. En outre, l'infrastructure de l'ICN repose sur l'intégrité et l'exactitude du routage du contenu, et est donc menacée par injections toxiques de chemins et de noms.
- ***Les attaques liées à la mise en cache :*** Le cache est l'un des composants importants dans ICN car la performance de son infrastructure est basée sur la mise en cache du récepteur qui vise à fournir la copie la plus proche disponible à un utilisateur. Par conséquent, ICN est vulnérable à toutes les opérations qui polluent ou corrompent le système de mise en cache.
- ***Des Attaques diverses*** : Les menaces dans cette catégorie visent à dégrader certains services ICN et à permettre à un attaquant de faire un accès non autorisé. Ces attaques entraînent une distribution de données insuffisante ou erronée.

Il y'a plusieurs attaques dans cette catégorie comme : Maltraitance des paquets (Packet Mistreatment), violation de clé de signature (Breaching Signer's Key) et L'accès non autorisé.

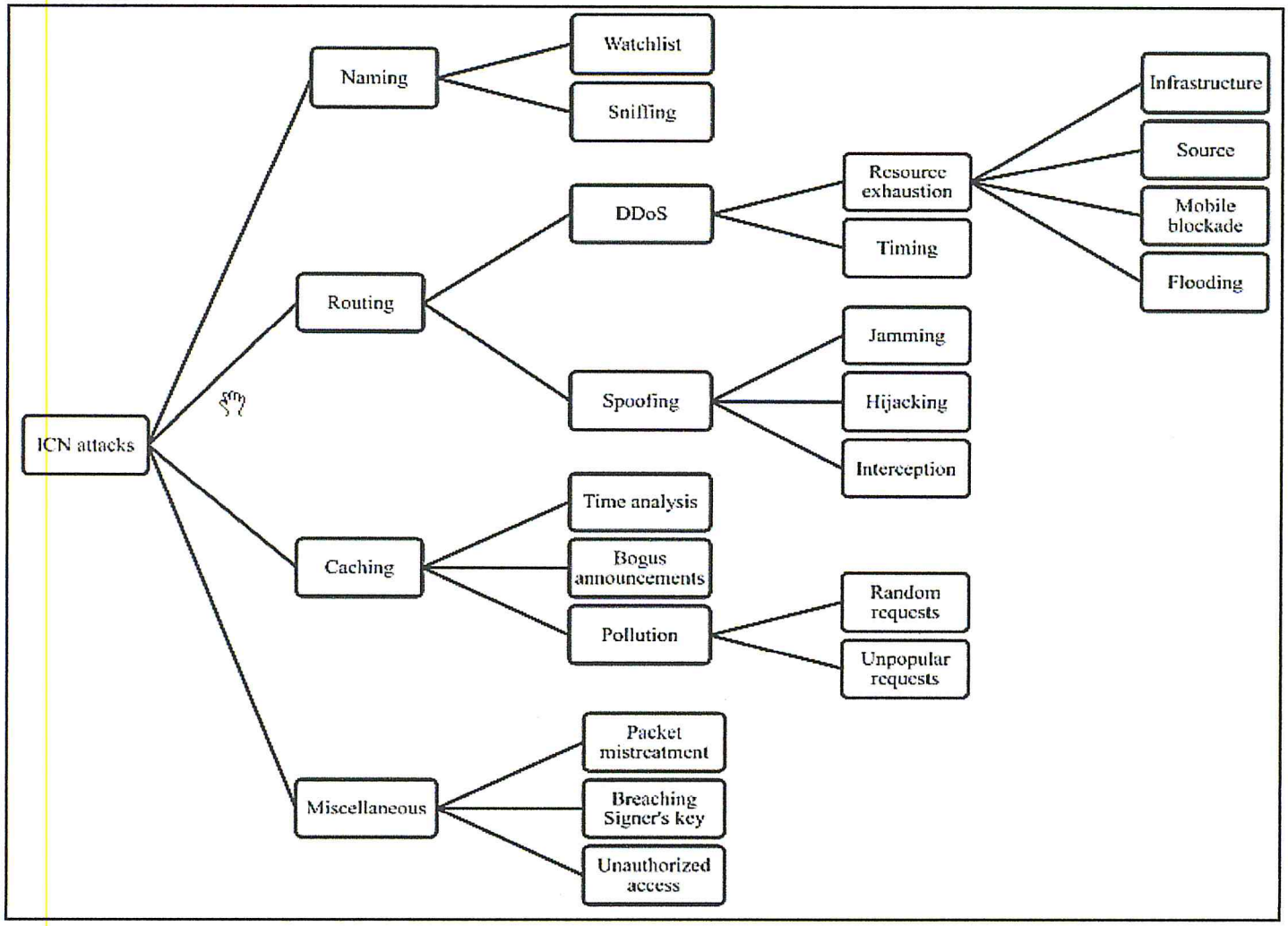


Fig. II.1. *Classification des attaques dans le réseau ICN.* [ESL 15]

II.2.2. Attaques liées au routage dans le réseau ICN

Les attaques liées au routage ont un impact sur les éléments suivants :

• ***Déni de service*** **:** Les DoS peuvent survenir en raison de nombreuses attaques dans cette catégorie, telles que l'envoi de nombreuses demandes de contenu indisponible ou d'une seule source, de blocage mobile, d'inondation (flooding), de Hijacking et de Timing. En conséquence, les temporisations intermédiaires suppriment les requêtes avec les délais expirés, ce qui peut entraîner des DoS ou au moins de longs délais.

• ***L'exténuation des ressources*** **:** Il existe de nombreuses sources d'exténuation des ressources dans l'infrastructure ICN qui proviennent d'un mauvais usage ou d'un trafic incontrôlé, comme l'envoi d'un grand nombre de requêtes et d'inondations.

• ***L'infiltration du chemin*** **:** Dans ICN, des copies de contenu sont généralement distribuées à de nombreux sites non approuvés, et il est donc difficile d'authentifier des origines valides pour le contenu. Le Hijacking et l'interception sont les principales sources d'infiltration de chemin dans le ICN, car les attaquants peuvent annoncer des routes non valides et les réclamer comme faisant confiance.

• ***Intimité*** **:** La violation de la vie privée (privacy) dans l'attaque d'interception donne à l'attaquant un accès non autorisé aux demandes de l'utilisateur, surtout lorsque l'attaquant est topologiquement proche ou sur la route vers l'utilisateur.

De ce fait, les attaques dans cette catégorie peuvent être classées dans les attaques de déni de service distribué (DDoS) et les attaques de Spoofing (attaques de falsification). Les attaques DDoS peuvent être classées dans l'épuisement des ressources et les attaques temporelles. L'épuisement des ressources peut être classé en blocs d'infrastructure, de source, de blocage mobile et les attaques de flooding. Les attaques par flooding peuvent être divisées en attaques de Jamming, de Hijacking et d'interception.

a- Infrastructure

Un attaquant envoie un grand nombre de demandes de contenu disponible/ indisponible. Lorsque les architectures ICN tentent de trouver la copie la plus proche du meilleur emplacement disponible, ces requêtes empruntent différentes routes vers la source provoquant des conditions de surcharge. Si le nombre de ces demandes est significativement élevé, cela entraîne un déni de service. Cette attaque peut être amplifiée, car les utilisateurs réguliers envoient des demandes de retransmission après un temps spécifié. Cette menace peut être atténuée car les mécanismes de routage dans ICN tentent de se diriger vers plusieurs emplacements. Comme l'illustre la *Fig. II.2*, l'attaquant, qui contrôle de nombreux systèmes finaux, envoie un grand nombre de requêtes à un ou plusieurs routeurs ICN pour remplir la table de routage et les ressources de traitement et de mémoire d'échappement. En conséquence, les routeurs attaqués transmettent ces requêtes

aux nœuds voisins, qui, à leur tour, les transmettent aux prochains nœuds voisins et ainsi de suite. Si le nombre de demandes invalides est élevé, toute demande légitime prend un temps de réponse plus long. Par conséquent, si le temps de réponse dépasse la période d'expiration de la demande, la demande peut ne pas être répondu. Ce scénario peut entraîner un déni de service ou au moins de longs délais.

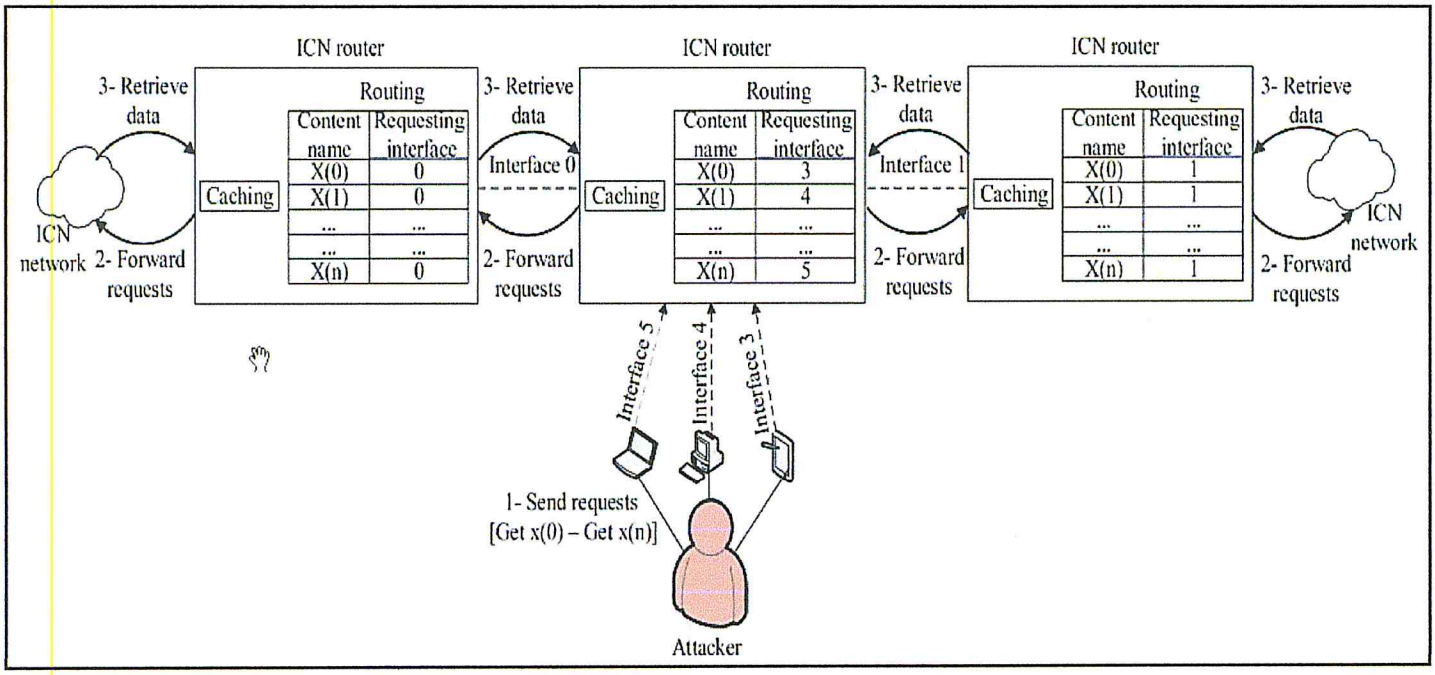


Fig. II.2. *Attaque d'infrastructure.* **[ESL 15]**

Scénario de l'attaque :

1- Un attaquant, qui contrôle de nombreux systèmes finaux, envoie un grand nombre de requêtes aux routeurs ICN.
2- Les routeurs attaqués transmettent ces demandes aux routeurs voisins et, à leur tour, les envoient à leurs routeurs voisins, etc.
3- ICN commence à récupérer ces grandes quantités de données de différents chemins et l'envoie aux emplacements demandés.

b- Source

Dans ICN, l'attaque d'une seule source peut également entraîner des conditions de surcharge pour l'infrastructure de routage. Un attaquant envoie un grand nombre de requêtes à une source de contenu spécifique pour dégrader ses performances. En conséquence, cette attaque augmente le temps de réponse de la livraison de contenu pour cette source de contenu ou son routeur d'accès. En plus de cet effet, l'attaque peut réduire le taux de retour des données et affecter les demandes de tous les nœuds dans les chemins vers les récepteurs. Le scénario d'attaque est similaire au scénario d'attaque d'infrastructure. Cette attaque affecte non seulement la source attaquée, mais affecte également le réseau global.

c- Blocage mobile

Un attaquant mobile peut surcharger une région en traversant des réseaux voisins sur des chemins circulaires tout en envoyant un nombre important de demandes de contenu. L'attaquant vise à surcharger les routeurs d'accès mobiles pour qu'il dépasse le délai d'exécution de l'état qui entraîne un blocage des réseaux régionaux. La retransmission des demandes fait partie de l'aspect de la mobilité dans un environnement ICN qui ajoute de la difficulté à détecter cette attaque.

Le scénario d'attaque présenté à la *Fig. II.3* est similaire au scénario d'attaque d'infrastructure. La différence est que l'attaquant mobile envoie un nombre élevé de requêtes aux réseaux voisins, alors que l'attaquant se déplace entre les réseaux de façon circulaire et continue.

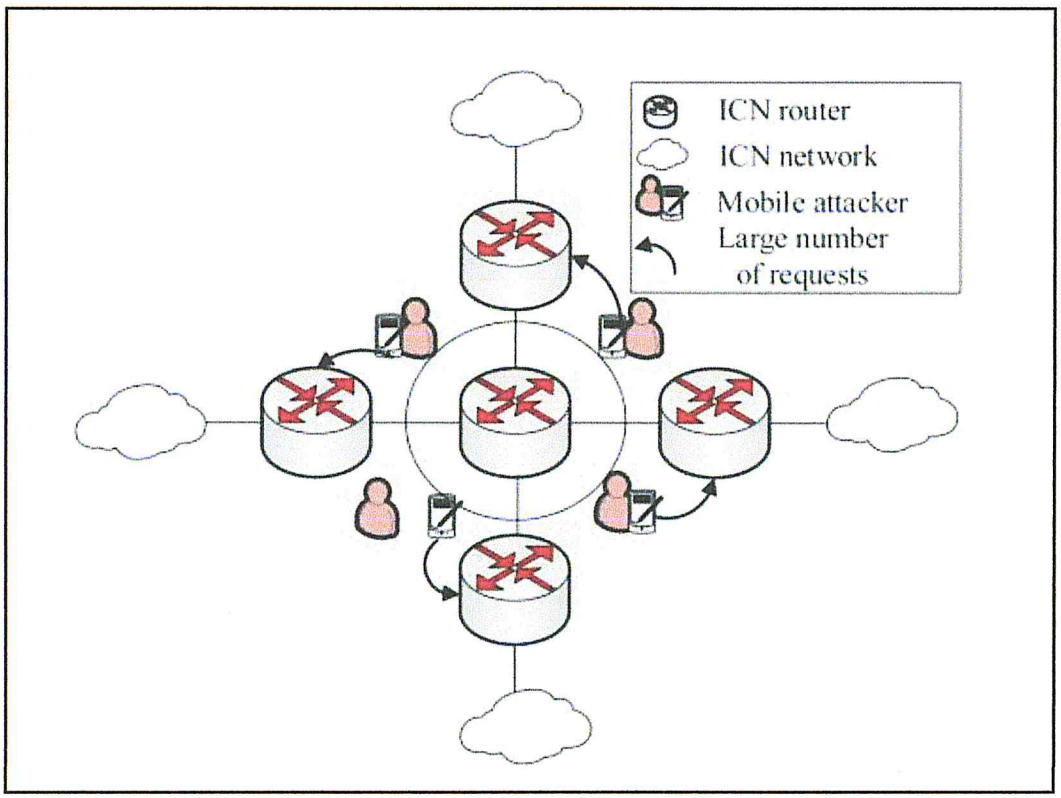


Fig. II.3. *Attaque de blocage mobile : un attaquant mobile envoie un grand nombre de requêtes, alors qu'il parcourt les réseaux voisins ICN.* **[ESL 15]**

d- Flooding

Les solutions existantes pour les attaques d'inondation dans ICN sont conçues pour limiter le nombre de requêtes, ce qui n'est pas approprié pour ICN. Un attaquant peut envoyer un grand nombre de demandes qui dépassent cette limite. Le nœud attaqué accepte un certain nombre de demandes et ignore les demandes restantes. En conséquence, l'attaquant réussit à surcharger l'infrastructure globale et nuit à tous les utilisateurs à proximité. En outre, comme ICN est une architecture centrée sur le contenu, il est difficile d'appliquer des limites pour le taux de demande par utilisateur final car il n'y a pas d'identifiant hôte. Le scénario d'attaque est également similaire au scénario d'attaque d'infrastructure. La différence est que l'attaquant envoie un certain nombre de demandes qui dépassent les limites des nœuds ICN et, par conséquent, ICN néglige les demandes légitimes dirigées vers les nœuds attaqués.

e- Timing

Il s'agit d'augmenter le délai d'attente de la demande pour certains nœuds ICN pour violer la cohérence entre la publication asynchrone ICN et le processus d'abonnement. Un attaquant envoie un grand nombre de demandes pour dégrader la performance de certains routeurs, de sorte que le routage des demandes et le transfert de données présentent des retards plus longs. Le scénario d'attaque est également similaire au scénario d'attaque d'infrastructure. La différence est que l'attaquant envoie un grand nombre de requêtes à travers une ou plusieurs routes pour augmenter le délai d'attente de la demande pour les demandes des utilisateurs légitimes.

f- Jamming

Un nœud sur un lien partagé envoie un grand nombre de demandes de contenu inutiles malveillantes. L'attaquant qui se masque en tant qu'abonné approuvé envoie les demandes malveillantes pour perturber le flux d'informations dans le système. Le réseau ICN répond et le contenu est envoyé à la destination sans récepteur. Ce scénario d'attaque est similaire au scénario d'attaque d'infrastructure. La différence, telle que présentée à la *Fig. II.4*, c'est que l'attaquant envoie des requêtes à un nœud partagé, qui l'envoie aux nœuds voisins.

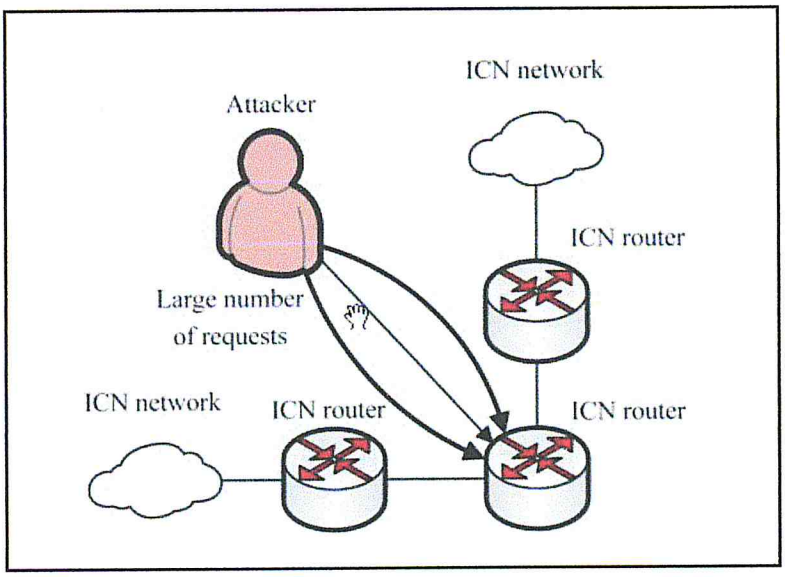


Fig. II.4. *Attaque de Jamming* **[ESL 15]**

g- Hijacking :

Contrairement aux architectures centrées sur l'hôte, tout nœud dans ICN peut mettre en cache et publier/souscrire des contenus. Un attaquant qui se masque en tant qu'éditeur de confiance peut annoncer des routes non valides pour tout contenu. Les demandes de contenu d'utilisateurs à proximité de l'attaquant sont orientées vers ces routes non valides. Par conséquent, ces demandes seront sans réponse, ce qui entraîne une DoS. L'effet de cette attaque peut être exacerbé, si l'attaquant a la possibilité de détourner des routes invalides à grande échelle. L'effet de cette attaque diminue parce que les mécanismes de routage dans ICN tentent de se diriger vers plusieurs emplacements.

Comme le montre la *Fig. II.5*, l'attaquant annonce des itinéraires non valides pour certains contenus pour attirer les demandes des utilisateurs. Lorsque les utilisateurs légitimes envoient des demandes pour l'une de ces routes malveillantes, les nœuds ICN transmettent ces requêtes aux nœuds malveillants. En conséquence, l'utilisateur légitime ne reçoit pas de réponse.

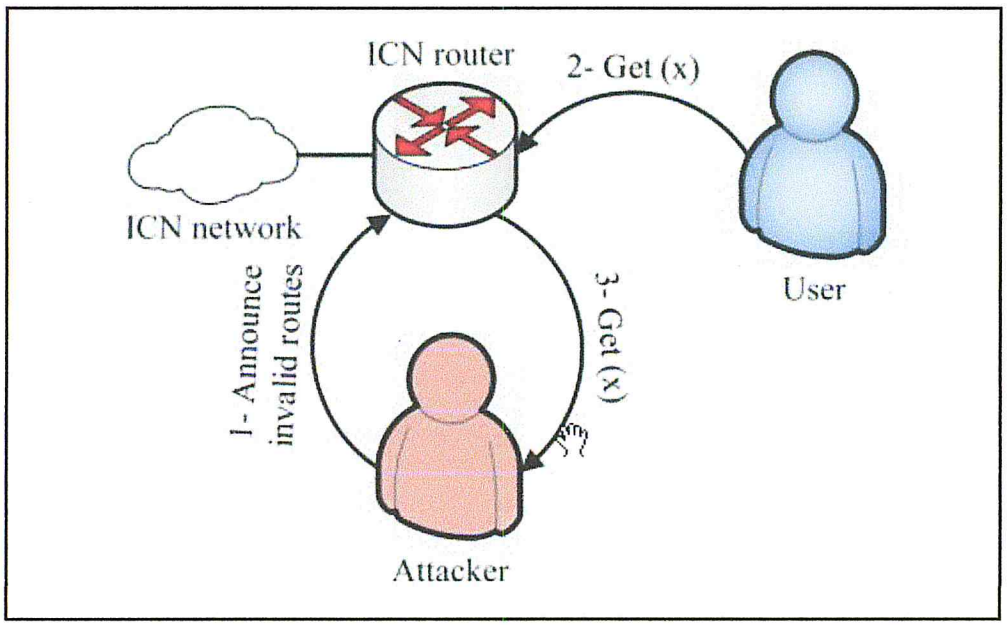


Fig. II.5. *Attaque de Hijacking* **[ESL 15]**

Scénario de l'attaque :

1- Un attaquant annonce des routes non valides pour certains contenus, y compris (x).
2- Une demande d'utilisateur pour le contenu ICN nommé (x).
3- Le routeur ICN redirige les requêtes de l'utilisateur vers les routes malveillantes de l'attaquant et, à la suite, l'utilisateur ne reçoit aucune réponse.

h- Interception :

Cette attaque est semblable à l'attaque habituelle "l'homme du milieu ". Contrairement à une attaque de détournement (Hijacking), un attaquant qui se masque en tant qu'éditeur de confiance annonce des routes invalides, tout en conservant un registre des routes valides du contenu. Les demandes de contenu peuvent ensuite être capturées et envoyées à l'emplacement approprié. Bien que le récepteur reçoive le contenu normalement, l'attaquant obtient une connaissance du contenu demandé.

Comme le montre la *Fig. II.6*, l'attaquant annonce des itinéraires non valides pour certains contenus pour attirer les demandes de l'utilisateur. Lorsque les utilisateurs légitimes envoient des demandes pour l'une des routes malveillantes, les nœuds ICN transmettent ces requêtes au nœud malveillant de l'attaquant. L'attaquant enregistre qui a demandé ce contenu puis l'envoie pour obtenir les données réelles. Lorsque les données réelles arrivent au nœud de l'attaquant, l'attaquant l'envoie de nouveau au nœud ICN demandé, qui l'envoie à son tour à l'utilisateur légitime. Pour l'utilisateur, le scénario semble être normal, mais en fait, l'attaquant viole intimité et la vie privée de l'utilisateur.

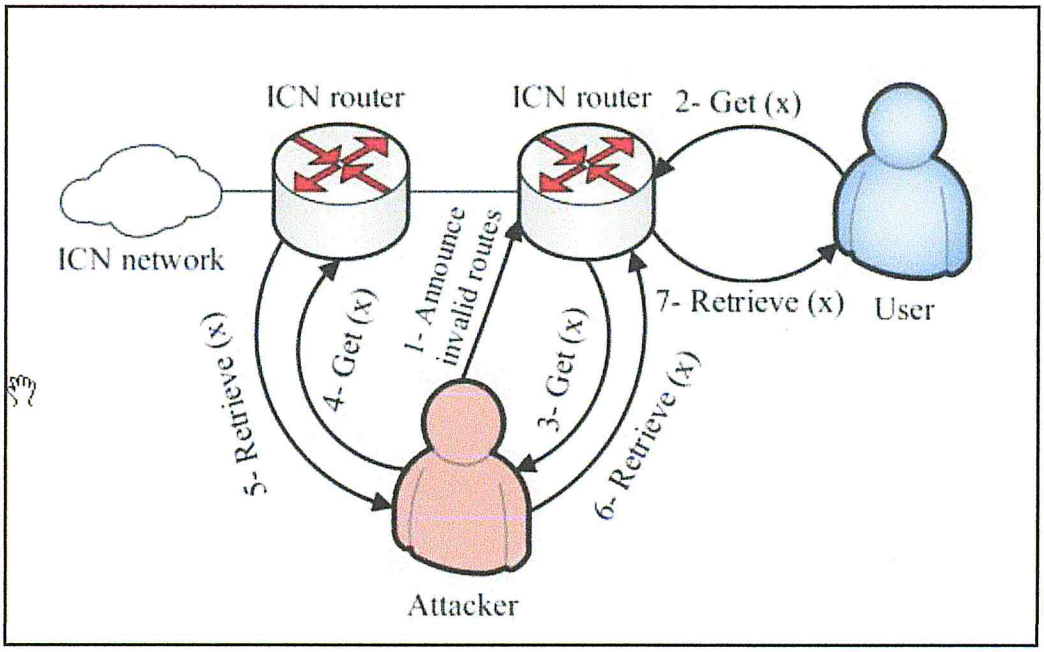


Fig. II.6. *Attaque de Interception* **[ESL 15]**

Scénario de l'attaque :

1- Un attaquant annonce des routes non valides pour certains contenus, y compris (x).
2- Une demande d'utilisateur pour le contenu ICN nommé (x).
3- Le routeur ICN redirige la demande de l'utilisateur vers les routes malveillantes de l'attaquant.
4- L'attaquant transmet la demande pour obtenir le contenu réel.
5- L'attaquant récupère le contenu (x).
6- L'attaquant transmet le contenu à l'utilisateur demandé.
7- L'utilisateur récupère le contenu (x)...

II.3. Solutions de Sécurité offertes par le CCN

Dans ce qui suit, nous décrivons comment la sécurité et la vie privée des utilisateurs bénéficient de l'architecture de l'CCN. Cela comprend l'intégrité vérifiable avec le contenu signé, l'absence d'adresses dans les intérêts, l'abandon de la résolution des noms en faveur de noms hiérarchiques et un certain degré de protection contre les attaques de déni de service (DoS).

II.3.1. L'intégrité vérifiable

Dans les CCNs, il est obligatoire pour les fournisseurs de contenu de signer le contenu et le nom avec leur clé privée. Par conséquent, le destinataire peut vérifier l'intégrité des données et de la source. Cela empêche la spoofing comme c'est possible dans l'Internet actuel. Un effet secondaire positif est la sécurité de routage. Comme tous les autres contenus, la communication entre les routeurs doit être signée. Cela empêche les adversaires d'influencer les tables de routage, à savoir la FIB dans CCN.

II.3.2. L'absence de l'adresse

CCN abandonne totalement les adresses. Ni les intérêts ni les messages de livraison de contenu ne contiennent des adresses. Les intérêts dans CCNs contiennent uniquement le nom du contenu demandé, mais pas celui qui l'a demandé. Seul le premier routeur d'acheminement connaît l'interface à partir de laquelle le contenu a été demandé. Tous les autres routeurs connaissent uniquement le routeur précédent sur le chemin de transfert. Lorsque le fournisseur de contenu renvoie le contenu, son message comprend également le nom du contenu, sa signature, l'ID de l'éditeur et des informations sur l'endroit où récupérer la clé publique de l'éditeur. Par conséquent, il n'est pas nécessaire d'adresser des adresses, bien que l'éditeur puisse être déduit de l'ID et de la clé. Du point de vue de la vie privée, ce design améliore l'anonymat car la source d'un intérêt est inconnue ou au moins difficile à découvrir.

Mais ce différent paradigme de communication a encore plus d'avantages. Dans l'Internet actuel, de nombreuses attaques exigent que l'attaquant envoie directement des

messages à la victime. Dans CCN, la victime n'a pas d'adresse, donc cette menace est complètement atténuée car aucun contenu ne peut arriver à un hôte sans que l'hôte ne demande le contenu à l'avance. Il est légèrement différent pour les éditeurs de contenu.

II.3.3. La Protection contre le déni de service

Outre la protection contre les attaques directes sur les hôtes, CCN offre même une protection contre les attaques de déni de service (DoS) sur les fournisseurs de contenu. Supposons qu'un attaquant lance une attaque DoS sur un éditeur en envoyant beaucoup d'intérêts. Ceux-ci sont simplement collectés au premier routeur et un seul intérêt est transmis. Même si l'attaquant contrôle un grand botnet qui est distribué sur de nombreux endroits qui ne partagent pas les routeurs sur le chemin de routage vers l'éditeur, l'agrégation des intérêts atténue la plupart des attaques. Ce qui s'est réellement passé est un changement dans le point d'attaque du fournisseur de contenu aux routeurs. Au lieu d'inonder l'éditeur, l'attaque remplit les tables d'intérêt en attente des routeurs. Certains chercheurs ont suggéré d'ajouter un drapeau à des intérêts qui indiquent que l'intérêt devrait être acheminé vers le fournisseur de contenu sans renvoyer de contenu mis en cache. Un tel drapeau permet aux attaquants d'inonder directement les éditeurs à nouveau.

II.3.4. La résolution du nom

CCNs est basé sur des noms hiérarchiques et ne nécessite donc pas de définition de nom explicite. Dans l'Internet actuel, la résolution des noms est fournie par DNS et offre une multitude de possibilités pour les attaquants à exploiter, par ex. Le détournement de DNS. Si les noms forfaitaires sont utilisés à la place, l'avantage est perdu.

II.4. Sécurité CCN versus Sécurité d'Internet

Même si l'architecture est différente par rapport l'Internet actuel, certains problèmes n'ont pas été résolus. Plusieurs d'entre eux ont à voir avec la surveillance. Le contournement de la surveillance - ou plus généralement la confidentialité - n'est pas définie comme principe de conception primaire de CCNs, de sorte que des techniques supplémentaires devront être construites sur l'architecture pour la soutenir.

Premièrement, le fournisseur de contenu est encore facilement identifiable. Bien qu'il soit possible de le faire via son adresse IP dans l'Internet actuel, CCN l'oblige à signer tout élément de contenu avec cette signature. En conséquence, les censeurs peuvent vérifier si le contenu arrivant provient d'une source indésirable et peut-être le filtrer. Deuxièmement, les censeurs sont actuellement en mesure de bloquer les relais connus de Tor[1] et de tester les hôtes de manière proactive afin de savoir s'ils sont des ponts Tor. S'ils suspectent un hôte, ils peuvent simplement se comporter comme s'ils étaient un utilisateur bénin et essayer d'accéder à Tor par le pont. Si cela fonctionne, ils ont identifié un pont Tor. Dans CCNs, les réseaux d'anonymat sont également envisageables. Troisièmement, les censeurs peuvent filtrer les intérêts pour les mots-clés ou les préfixes interdits dans leurs noms. D'autres techniques comme Deep Packet Inspection (DPI) sont également applicables. Enfin, des tiers de confiance comme les Autorités de certification (CA) dans l'Internet actuel seront toujours nécessaires. Même si chaque élément doit être signé par l'éditeur, le destinataire a besoin d'un tiers de confiance fournissant la clé publique afin de pouvoir vérifier l'intégrité du contenu **[ROM 16]**.

II.5. Conclusion

Dans ce chapitre, nous avons discuté la sécurité dans les ICNs en général et en particulier la sécurité native offerte par l'architecture de CCN. Bien que l'architecture CCN offre plusieurs solutions pour la protection de la sécurité et de la vie privée que l'Internet actuel, notre travail aborde spécifiquement les aspects de sécurité liés à une attaque de déni de service qui pourrait être implémentée sur CCN. Nous avons choisi le Flooding comme un type d'attaque pour l'étudier. Cette attaque est également connue dans la littérature comme Interest Flooding Attack, ce qui signifie qu'un ou plusieurs utilisateurs finissent par inonder le réseau avec des intérêts ciblés sur des ressources non existantes, qui ne gaspillent que la mémoire du routeur et les ressources de la CPU, et cette attaque peuvent également être lancée dans CCN.

[1] **TOR :** The Onion Router est un réseau informatique superposé mondial et décentralisé. Ce réseau permet d'anonymiser l'origine de connexions TCP.

Chapitre III :

L'Inondation (Flooding) dans le Routage Centré Contenu

III.1. Introduction

Bien qu'il existe des aspects de sécurité et de confidentialité qui s'améliorent grâce à CCN, cela pose également de nouveaux défis. CCN améliore plusieurs faces de l'expérience d'utilisateur ainsi que la sécurité, la confidentialité et les contrôles d'accès. Cependant, il pose de nouveaux défis en matière de sécurité. Dans ce chapitre, nous allons étudier les attaques d'Interest Flooding et présenter les principales solutions possibles.

III.2. L'attaque d'Interest Flooding

Dans les réseaux IP traditionnels, les attaques DDoS pèsent habituellement sur les terminaux, car ces éléments contiennent l'état de l'information de connexion. D'autre part, CCN est à peine basé sur le fait que les routeurs intermédiaires se maintiennent par l'état de paquet. Cette fonctionnalité permet au protocole d'éviter les boucles de routage puisque chaque intérêt est enregistré dans la table PIT et aussi pour implémenter un support de multidiffusion natif car chaque nœud "se souvient" à quel demandé. Cependant, cette fonctionnalité attaque les utilisateurs parce qu'il existe la possibilité de générer artificiellement des paquets forgés dans le seul but de gaspiller la mémoire du routeur. En particulier, nous considérons le scénario représenté dans la figure III.1.

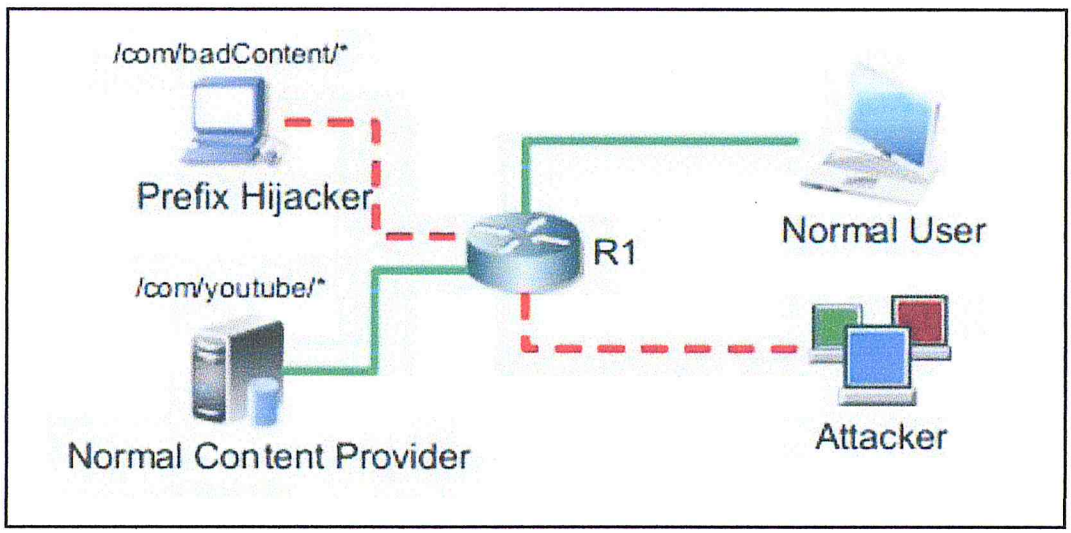


Fig. III.1. *Exemple d'attaque d'attaques d'Interest Flooding.* **[MAT 15]**

L'attaquant doit annoncer un préfixe valide pour envoyer des faux intérêts ; Dans notre exemple, "/com/badContent /*" sert à cet effet et appelons ce nœud "prefix Hijacker". En outre, l'attaquant a besoin d'un ou plusieurs clients zombie (ou même d'un grand botnet) pour commencer à envoyer des Intérêts ciblant le préfixe existant, mais avec des noms de ressources non existants (et éventuellement longs), par exemple "/com/badContent/abcdefg ...z ". De tels paquets atteindront correctement le préfixe Hijacker, c'est-à-dire la machine configurée pour recevoir ces datagrammes sans générer de réponse, mais aucune donnée utile ne sera renvoyée. Avec cette procédure simple, tous les Intérêts vus par R1 (et pour lesquels R1 crée une entrée dans le PIT) resteront dans la mémoire de l'appareil jusqu'à ce que le délai d'attente[2] expire.

III.1.1. Description du problème

Le PIT est la structure fondamentale utilisée pour maintenir l'état de chaque flux actif. Il se développe avec les utilisateurs qui envoient leurs intérêts et se rétrécit lorsque les paquets de données arrivent au routeur. Compte tenu de la vitesse d'accès requise pour une telle structure et les possibilités offertes dans ce sens par les technologies de mémoire actuelles, la taille PIT pourrait représenter comme un goulot pour l'ensemble de l'infrastructure CCN.

Le problème pourrait être exacerbé par une utilisation massive de long "Interest LifeTimes", ce qui augmenterait encore le nombre d'entrées simultanées dans le PIT. Malgré la nature de l'attrait de CCN, cette possibilité ne peut pas être exclue car il serait nécessaire de soutenir les services de publish/subscribe où les utilisateurs s'abonnent à un contenu donné qui sera produit de manière asynchrone dans le futur. Beaucoup de ces services sont mis en œuvre, par exemple, au moyen de stratégies de sondage long HTTP, qui utilisent largement de longues demandes (potentiellement sans réponse).

En général, les trente secondes (30s) sont considérés comme une valeur sûre, car les temporisateurs plus longs peuvent être indésirables pour les proxies intermédiaires placées entre le serveur et le client. Cependant, les solutions avec des valeurs plus longues

[2] Appelé LifeTime : est définie par le client dans le paquet Interest, de façon que le routeur n'a aucun contrôle sur celui-ci.

sont largement adoptées dans les applications Web communes (par exemple, Facebook et certaines applications de messagerie Web, ce qui utilise souvent des temporisations de plus d'une minute). En plus de ça, nous devons considérer qu'un ou plusieurs utilisateurs malveillants peuvent exploiter des demandes artificielles dans le but de remplir la mémoire PIT disponible sur les routeurs, impliquant ainsi une attaque de déni de service distribué (DDoS).

Avec le débordement de PIT, les utilisateurs verraient leurs intérêts rejetés par les routeurs et, par conséquent, ils subiront un taux croissant de retransmissions jusqu'à ce que le réseau s'effondre complètement **[MAT 13]**.

III.1.2. Scénario d'attaque

La figure III.2 montre un réseau très simple composé de trois routeurs, un attaquant (ou, de façon équivalente, un botnet), un client normal et une mauvaise destination, ad hoc placé par l'attaquant. Le rôle du botnet est celui de générer des intérêts vers la fausse destination. Les routeurs le long du chemin pointillé (R1 et R2) sont inondés d'intérêts sans réponse, ce qui augmente leur consommation de mémoire.

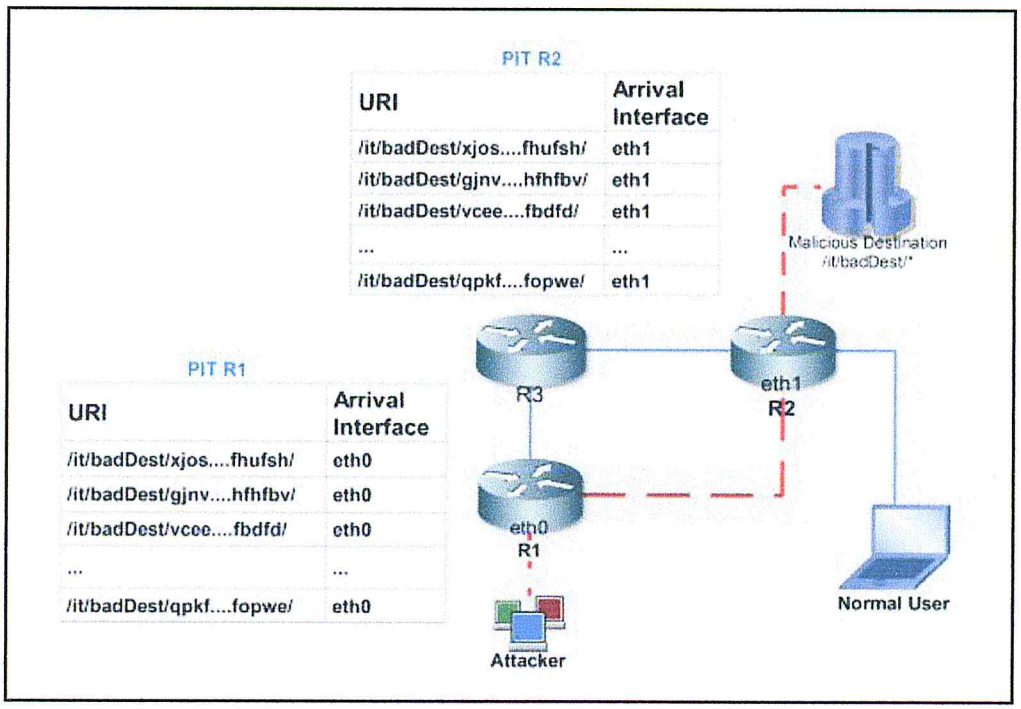


Fig. III.2. *Scénario d'attaques d'Interest Flooding.* **[MAT 13]**

III.3. Les solutions possibles

L'IFA est analysé dans plusieurs documents, qui traitent également de certaines solutions possibles et qui contiennent une évaluation préliminaire des menaces de sécurité possibles applicables à l'architecture CCN.

Compte tenu de ce problème, certaines architectures PIT possibles ont été proposées dans le but de réduire la taille de la table requise. Une première solution simple **[JAC 09]** consiste à réaliser le PIT au moyen d'une table de hachage, où chaque URI (Uniform Resource Identifier) est codé en utilisant un nombre fixe de bits. D'autres solutions possibles déploient des architectures plus complexes.

Par exemple, **[DAI 12]** propose une structure en forme d'arborescence pour organiser les entrées PIT afin d'assurer également une recherche élevée, une insertion et une vitesse de suppression, tandis que **[YOU 12]** présente une architecture PIT très efficace basée sur Bloom Filtres.

Bien que ces solutions puissent être efficaces pendant le fonctionnement normal, la dégradation des performances pourrait être connue lorsque le système se trouve sous les attaques DDoS susmentionnés. Ce type d'attaque peut être implémenté en générant de manière distributive des paquets d'intérêt qui contiennent un préfixe de destination valide, mais des noms de ressources non existants, de sorte que les routeurs renvoient correctement les Intérêts et stockent de nouvelles entrées dans leurs PIT, mais les réponses ne reviennent jamais. La destination peut éventuellement être collusion avec l'attaquant et simplement supprimer les paquets entrants. Afin de maximiser l'impact de l'attaque, un utilisateur malveillant pourrait demander toujours des ressources différentes, évitant ainsi la fusion de l'intérêt. En outre, l'attaquant peut également sélectionner de grandes valeurs pour le champ LifeTime **[MAT 13]**.

D'après *Virgilio M et al* **[MAT 15]**, trois propositions principales visent le problème spécifique d'IFA :

- Le travail de *Afanasyev et al* qui présente un cadre nommé ***Satisfaction Based Pushback*** pour limiter le nombre d'intérêts transférés pour un préfixe donné selon les statistiques sur le ratio de satisfaction des intérêts.

- Une autre contre-mesure, appelée ***Poseidon***, cette approche est similaire à la précédente en ce qu'elle rassemble des statistiques sur le trafic vu sur chaque routeur mais avec une procédure d'activation différente.

- La dernière solution appelée ***Traceback***. L'idée est d'activer une contre-mesure après que l'utilisation de la mémoire a atteint un seuil prédéfini. L'algorithme consiste à générer des paquets de données falsifiés pour les entrées qui causent le problème de débordement de mémoire.

Dans ce qui suit, nous décrivons brièvement ces solutions.

III.3.1. Satisfaction Based Pushback

L'algorithme de Pushback basé sur la satisfaction est décrit dans **[AFA 13]** et il fonctionne comme suit : chaque routeur calcule la métrique de satisfaction d'intérêt comme le rapport entre le nombre d'intérêts satisfaits sur le nombre d'intérêts transférés. Ce calcul est effectué sur une base par interface et il est une indication de la probabilité qu'un intérêt provenant d'une certaine interface soit satisfait. Une telle métrique peut être utilisée directement pour calculer les limites des Intérêts que le routeur est disposé à transférer de chaque interface.

Après le calcul, le routeur annonce ses limites aux voisins en aval afin de limiter le trafic entrant, en particulier pour les interfaces qui augmentent le fardeau de l'occupation de la mémoire PIT. Après un certain temps, le routeur calcule de nouvelles statistiques et efface son histoire avec une décroissance exponentielle, afin de restaurer les limites d'origine et donner l'opportunité à chaque interface d'avoir des intérêts plus vertueux remis

à nouveau. Notez que des mesures concernant le trafic peuvent être collectées à différentes granularités : préfixes, entrées FIB et ainsi de suite.

III.3.2. Poseidon

Poseidon **[COM 13]** est un cadre pour atténuer l'effet de l'IFA sur les réseaux CCN / NDN. Il partage certaines similitudes avec l'algorithme précédent puisqu'il collecte également des statistiques en observant le trafic transféré. La différence principale est dans la phase de détection : Poseidon est déclenché lorsque deux mesures dépassent leurs seuils correspondants. Les deux paramètres utilisés par Poseidon sont définis comme suit :

1) $\omega(r_i^j, t_k) = \frac{\#\ Interes\ ts\ from\ r_i^j\ at\ time\ t_k}{\#\ Data\ from\ r_i^j\ at\ time\ t_k}$

2) $\rho(r_i^j, t_k) = \#\ of\ PIT\ bytes\ used\ by\ r_i^j\ at\ time\ t_k$

Pour que Poseidon soit activé, ces deux mesures doivent dépasser la valeur autorisée. L'algorithme utilise les deux pour limiter le nombre de faux positifs, à savoir le nombre de fois qu'il détecte de manière erronée une attaque en cours.

Poseidon réagit à une détection d'attaque en imposant des limites au nombre d'Intérêts acceptés de l'interface qui dépassait les deux seuils et les réduisait pour cette interface. Après un certain temps, si le trafic devient normal à nouveau, Poséidon restaurera tous les seuils à leurs valeurs d'origine et les limites imposées sont désactivées.

III.3.3. Interest Traceback

L'algorithme Traceback est décrit dans **[DAI 13]** et il est conçu pour libérer les entrées PIT indésirables lorsque la quantité d'espace mémoire disponible se situe sous un seuil prédéfini. La phase de détection est plutôt simple et nécessite seulement de surveiller l'utilisation de la mémoire PIT au fil du temps. Après avoir détecté une occupation anormale de la mémoire, le processus Traceback est déclenché et un ensemble de paquets de données falsifiés est généré pour les entrées qui restent insatisfaites pendant une longue période. Les paquets de données falsifiés portent le nom nécessaire pour satisfaire les

intérêts offensants et sont transmis en aval pour libérer des ressources tout au long du chemin.

Afin de tirer parti, autant que possible, de l'espace mémoire disponible pour le PIT le seuil après lequel le Traceback est activé, est défini en tant que 90% de la mémoire occupée. Une telle limite agressive évite l'hyper-réaction de l'algorithme et permet au réseau de supporter le pic de trafic temporaire sans déclencher aucun mécanisme de blocage des intérêts. Dans **[MAT 15],** les auteurs ont conçu un code pour répondre aussi étroitement que possible à la description de la contre-mesure. En particulier, ils simulent pour chaque routeur un processus de surveillance programmé chaque seconde pour vérifier si l'occupation de la mémoire dépasse sa valeur alarmante. Si (et seulement si) dépasse le seuil (plus de 90% d'occupation de la mémoire), la fonction FindAndSend() est invoquée pour générer des paquets de données falsifiés et les faire voyager vers l'initiateur de l'attaque.

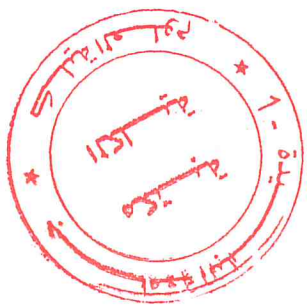
Une vision simplifiée de haut niveau d'implémentation d'*Virgilio M et al* est illustré dans la figure suivante :

```
void Traceback::FindAndSend()
{
    FOR EACH Entry in Pit
        IF IsOld(Entry)
            FOR EACH Face in Entry.FacesList
                IF Face.IsConnectedToEndUser()
                    BLOCK Face
                ELSE
                    GENERATE SpoofedData
                    SEND SpoofedData through Face
                END IF
            END LOOP
            RELEASE memory
        END IF
    END LOOP
}
```

Fig. III.3. *Algorithme de Traceback sending spoofed Data.* **[MAT 15]**

Chapitre IV :

Simulations



IV.1. Introduction

La simulation des réseaux en générale consiste principalement en la reproduction du comportement et du fonctionnement des nœuds dans un environnement informatique pour des raisons tel que : la répétition d'expérience, l'adressage des systèmes complexes, le gain de temps et la variation des paramètres de simulation. Alors que la simulation réelle s'avère coûteuse, voire impossible dans quelque cas comme la simulation réelle des CCNs. Dans ce qui suit, nous présentons le simulateur "OMNET++" que nous avons choisi dans notre travail, ainsi que le package CCN-lite qui nous aide à faire des simulations légères des CCNs. En fin, nous faisons des scénarios des attaques d'Intersest Flooding avec des solutions possibles pour cette attaque.

IV.2. Simulateur OMNET++

OMNET++ (Objective Modular Network Testbed in C++) est un IDE (Integrated Development Environment) basé sur la plateforme Eclipse. C'est un environnement open source qui fournit des outils pour la création et la configuration des modèles de réseaux (fichiers NED et INI) et des outils pour l'exécution d'un lot de programmes ainsi que pour l'analyse des résultats de simulation **[MSK 05]**.

OMNET++ est totalement programmable, paramétrable et modulaire ainsi grâce à son architecture flexible et générique. Il a été utilisé avec succès dans divers domaines, notamment **[OMN 04]** :

- La modélisation des protocoles de communications,
- La modélisation des réseaux filaires et sans fils,
- La modélisation des systèmes répartis,
- Les architectures Hardware,
- En général, il peut être utilisé pour n'importe quel système à évènements discrets pouvant être modélisé selon des entités communiquant par envoient de messages.

OMNET++ semble être le meilleur parmi les solutions open source et freeware. Il semble séduire de plus en plus la communauté scientifique et un nombre croissant de modèles sont disponibles. Ainsi, Il sera notre environnement de simulation, grâce à son architecture modulaire.

IV.3. CCN-lite

Le simulateur OMNeT++ n'est pas spécialisé pour les réseaux centrés Contenu (CCN). Pour cela, il existe plusieurs extensions, plateformes et simulateurs basés sur OMNET++ qui essayent d'introduire la notion du CCN dans OMNeT++. La plus utilisée (bien documentée) et la plus facile à manipuler est "CCN-Lite".

IV.3.1. Présentation de CCN-lite

CCN-lite est une implémentation réduite et légère - mais fonctionnellement interopérable - des protocoles CCNx et NDN **[LIT 15]**. Cela couvre :

- Le protocole de réseau centré sur le contenu de PARC
- Le projet Named-Data Networking (NDN),
- Le projet Naming-Function Networking,
- Un encodage expérimental et compact pour les environnements IoT.

La motivation initiale pour créer CCN-lite en 2011 était que le logiciel de routage CCNx de PARC (Palo Alto Research Center) est devenu énorme. CCN-lite fournit une alternative maigre à des fins éducatives pour ceux qui veulent un logiciel simple pour leurs propres expérimentations ou développements et qui n'ont pas besoin de toutes les fonctionnalités du CCN. Il nous offre :

- Un noyau CCNx écrit en langage C (1000-2000 lignes)
- Un Support de multiple plate-forme (Espace utilisateur Linux et OS X, Noyau Linux, Android, Arduino et RFduino et OMNET++)
- Une mise en œuvre partielle du protocole de gestion interopérable
- Un serveur HTTP simple pour afficher la configuration interne du relais
- Quelques extensions intéressantes.

IV.3.2. Description conceptuelle de l'intégration CCN-lite/OMNeT++ :

OMNeT ++ représente une approche Framework. Au lieu de fournir directement des composants de simulation pour les réseaux informatiques ou d'autres domaines, il fournit les machines de base et les outils pour écrire de telles simulations. Les domaines d'application spécifiques sont pris en charge par différents modèles de simulation et frameworks tels que Mobility Framework ou INET Framework. Ces modèles sont développés indépendamment de OMNeT ++.

INET Framework est une bibliothèque des modèles open source pour l'environnement de simulation OMNeT ++. Il fournit des protocoles, des agents et d'autres modèles pour les chercheurs et les étudiants qui travaillent avec les réseaux de communication. INET est particulièrement utile lors de la conception et la validation de nouveaux protocoles, ou en explorant des nouveaux ou exotiques scénarios **[INE 17]**.

La figure IV.1 représente l'intégration de CCN-lite avec OMNeT++ et INET Framework qui est nécessaire pour faire la simulation. Parmi les dépendances évidentes : OMNeT++ (> v4.2.2) simulateur et INET Framework (> v1.99.4.).

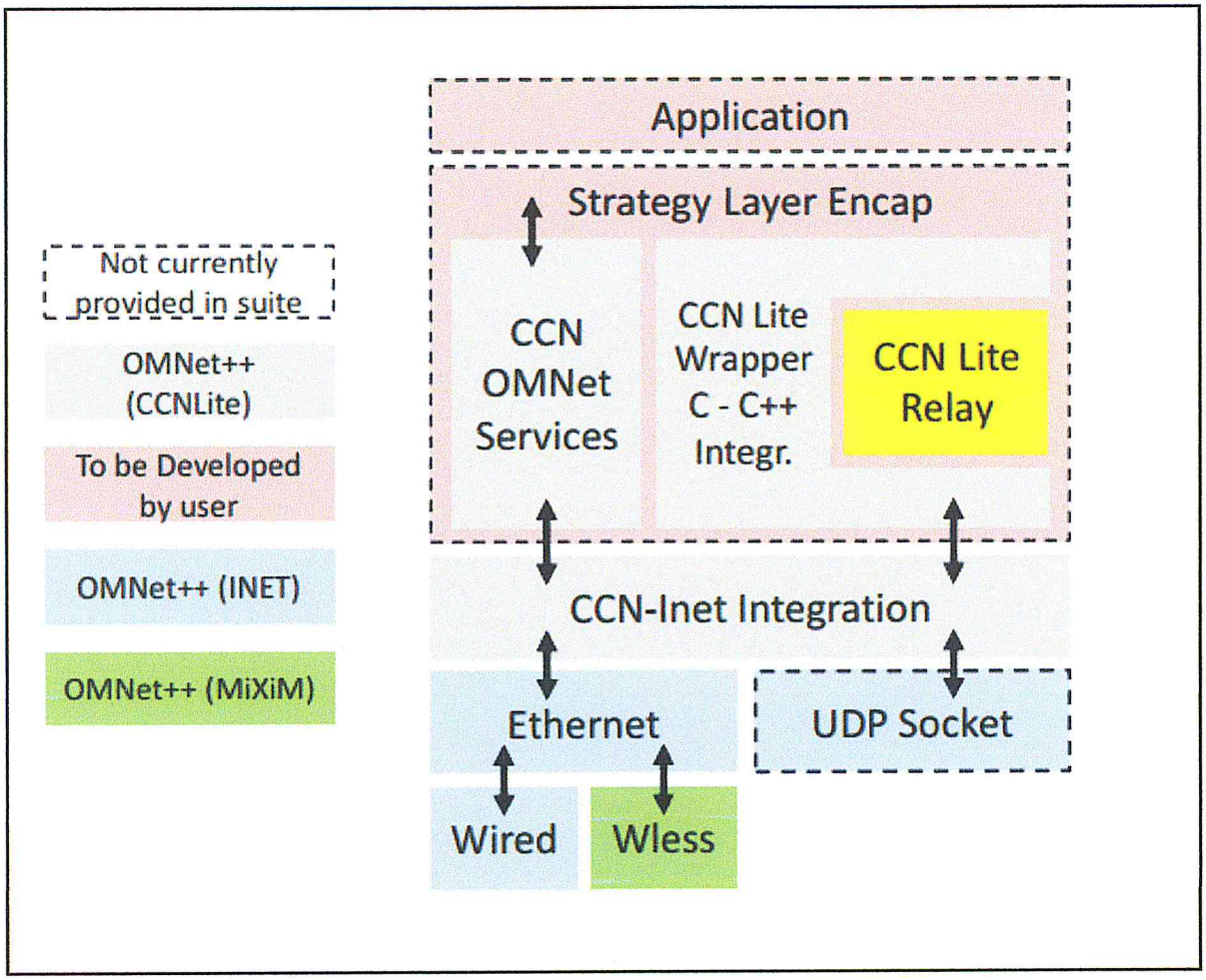


Fig. IV.1. *Schéma d'intégration CCN-lite avec OMNeT++.* **[GIT 15]**

IV.3.3. Diagramme de classe des composants CCN-lite/OMNeT++ :

Les classes les plus importantes dans le package CCN-lite sont celles du répertoire ccnlite/src/ (*voir la Fig. IV.2 et Fig. IV.3*) :

- **ccn-lite /** : Dossier contient l'implémentation réelle de CCN-lite relais en C.
- **CcnCore. {Cc, h}** : Intégration CCN-lite (C-C ++)
- **Ccn. {Cc, h, Ned} :** Services OMNeT++
- **CcnInet. {Cc, h, Ned} :** Intégration OMNeT++ / INET Framework
- **CcnAdmin. {Cc, h, Ned} :** Administrateur de Scénario

• **Parser. {Cc, h} :** Analyseur de scénario

• **CcnPacket_m. {Cc, h} :** Conteneur extensible pour les paquets CCN échangés via des nœuds CCN dans OMNeT++

• **CcnAppMessage_m. {Cc / h / msg} :** C'est un message échangé entre la couche CCN et la couche ci-dessus (stratégie ou application).

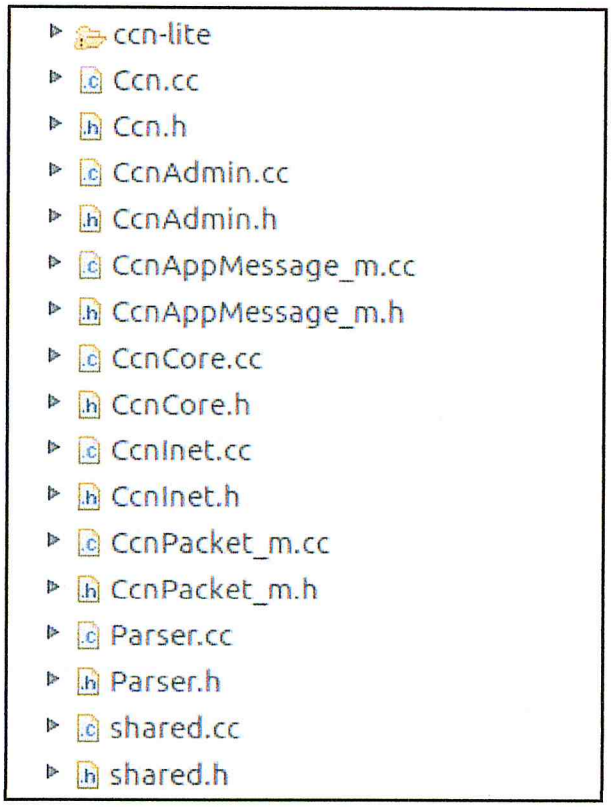


Fig. IV.2. *Les fichiers de CCN-lite.*

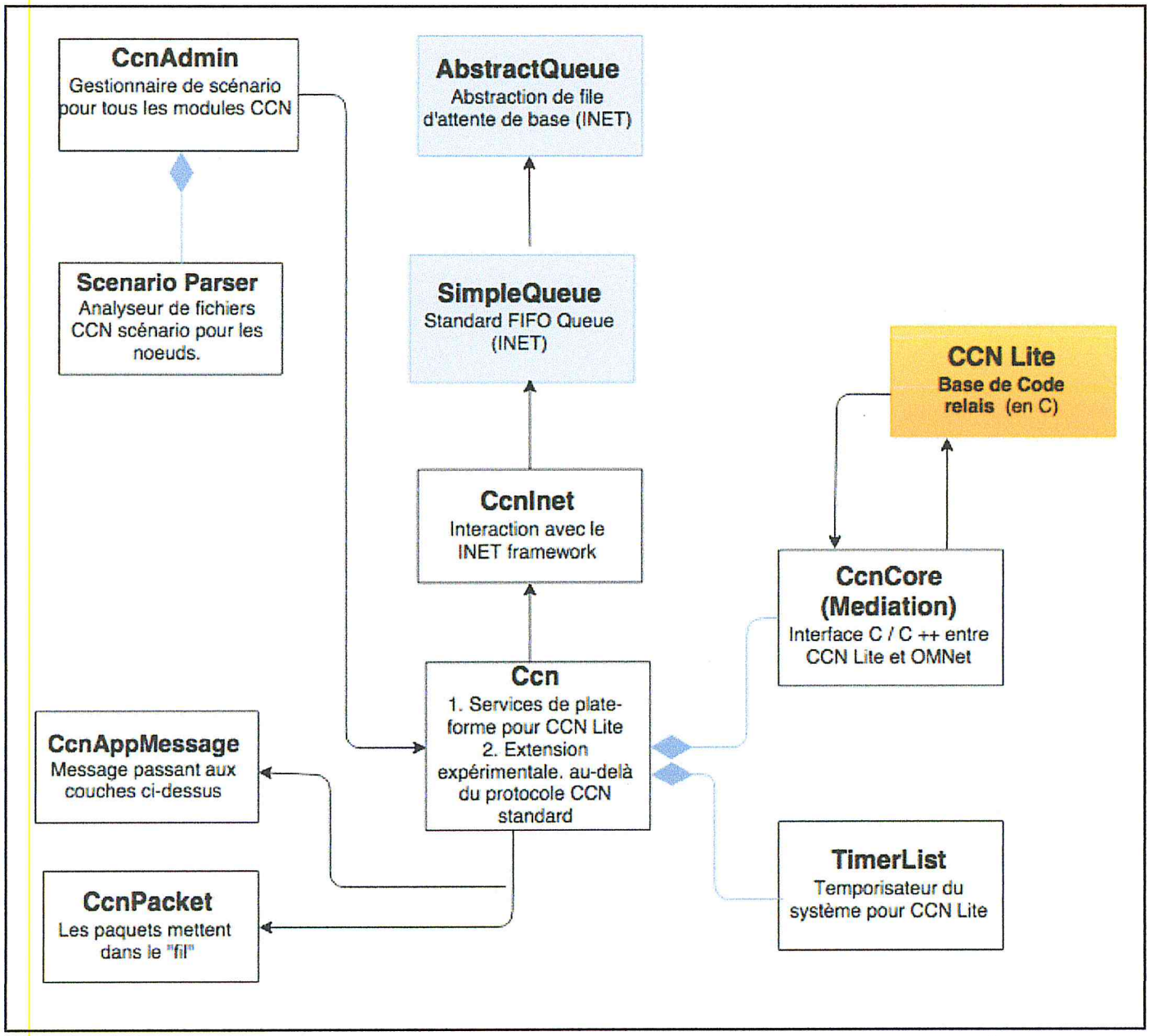


Fig. IV.3. *Diagramme de classe UML des Composants CCN-lite/OMNeT++.* **[GIT 15]**

IV.4. Environnement de travail

Nous allons détailler les environnements utilisés dans la réalisation de notre simulation.

IV.4.1. Environnement matériel

La simulation a été réalisée sur un ordinateur Lenovo dont la configuration est :

Processeur	Intel Core i5 CPU @ 2.5 GHz
Mémoire	4GB DDR4
Disque dur	1 TO

IV.4.2. Environnement logiciel

Après plusieurs essaies dans différents systèmes d'exploitation et plusieurs version d'OMNeT++ et INET, notre simulation a été réalisée dans l'environnement logiciel suivant:

- Système d'exploitation : Linux Distribution Ubuntu 16.04.
- Le simulateur OMNeT++ 4.5 ;
- La plateforme INET v 2.4.0.

IV.5. Implémentation de l'algorithme Traceback

L'algorithme de Tarceback est lancé dans un routeur après l'occupation de 90% de mémoire PIT, il permet de (dans l'ordre) :

1- Bloquer les faces qui sont directement connectées à un hacker ;
2- Générer des paquets de données falsifiées ;
3- Envoyer les paquets falsifiés afin de satisfaire les paquets d'intérêt d'attaquant ;
4- Libérer la mémoire PIT.

Cet algorithme est implémenté dans les classes suivantes :

- **traceback.h** qui contient la déclaration des structures de données et les fonctions nécessaires (Figure IV.4)

traceback.cc | traceback.h | CCNoEther_Tau_2 | omnetpp.ini | Ccn.h

```
#ifndef TRACEBACK_H_INCLUDED
#define TRACEBACK_H_INCLUDED
typedef struct SpoofedData
 {
     char* data;
 } SpoofedData;
typedef struct Face
 {
     int port;
     int connected;
     int blocked ;
 } Face;
typedef struct Entry
 {
     int date;
     char *prefix;
     Face face;
     int segment;
 } Entry;
     /* isOld() est une fonction qui vérifier que l'intérêt est encien
      * ou non dans le PIT*/
int isOld (Entry E){ };
     /* IsConnectedToEndUser() est fonction qui vérifier est que l'interface
      * est directement avec un host terminal */
int IsConnectedToEndUser (Face f){ };
     /* IBLOCK() est fonction qui Bloque une interface donnée */
void  BLOCK(Face f){};
     /* GENERATE() est fonction qui généré les données falsifiées */
SpoofedData GENERATE(){ };
     /* SEND() est fonction qui envoie les données falsifiées */
void SEND (SpoofedData d, Face f){ };
     /* RELEASE() est fonction qui libère la mémoire */
void RELEASE (Entry Pit [], int * n){};
/* la fonction principale de L'algorithme Traceback */
void FindAndSend(Entry Pit [], Face FaceList[], int n, int m);

#endif // TRACEBACK_H_INCLUDED
```

Fig. IV.4. *Le prototype des fonctions définies dans Traceback.*

- **traceback.c** qui contient le code source des fonctions traceback.h. Le fonctionnement de l'algorithme se trouve dans la fonction **FindAndSend()** (Figure IV.5)

traceback.h | *CcnAdmin.h | *CcnAdmin.cc | CcnAdmin.cc | traceback.cc

```
#include"traceback.h"
#include <stdlib.h>
#include <stdio.h>

void FindAndSend(Entry Pit [], Face  FaceList[], int n, int m)
{
    SpoofedData d;
    Face f;
    int i, j;
    for (i = 0; i<n ; i++)
    {
        if (isOld(Pit[i]))
        {
            for (j = 0; j<m; j++)
            {
                f = FaceList[j];
                if (IsConnectedToEndUser(f))
                    BLOCK(f);
                else
                    d = GENERATE();
                   SEND(d, f);
        }
            RELEASE(Pit, &n);
    }
}
}
```

Fig. IV.5. *La fonction FindAndSend()de Traceback.*

L'algorithme **Traceback** est intégré à CCN-lite comme suit :

1. Mettre les classes **traceback.h** et **traceback.c** dans le répertoire **ccn-lite\src\omnet\src.**
2. Modifier la fonction **sendBatchIntersests()** de la classe **Ccn.cc** (Figure IV.6) de telle sorte qu'elle mette à jour la structure PIT et appelle la fonction **FindAndSend()**.
3. Recompiler le code CCN-lite

traceback.h | *CcnAdmin.h | CcnAdmin.cc | CcnAdmin.cc | traceback.cc | Ccn.cc

```
 * Send interests in batch, for a range of chunks of a named content. This is
 * only a convenience method that is currently used by the CcnAdmin module
 */
bool
Ccn::sendBatchInterests(const char *contentName, int startChunk, int numChunks)
{
    /* the CcnAdmin module accesses this method atomically
     */
    Enter_Method("Ccn::sendBatchInterests()");

    /*
     * IMPORTANT NOTE: This method as is does not handle prefix requests. E.g. it
     * always assumes an exact match for a content exists. To handle prefix requests
     * (a prefix is specified and the longest prefix match object is returned) we'd
     * need to have a convention for interpreting startChunk and numChunks when out
     * of bounds. Eg. if numChunks is -1 could imply that the contentName is a prefix
     * and not an object name to be exactly matched, and in this case the startChunk
     * may be considered undefined. Then depending on the case the CcnCore::requestContent()
     * would be called with according values.
     */

    for (int i=0 ; i< numChunks ; i++) {
        DBGPRT(EVAUX, Info, this->getFullPath())
                << "Sending Interest for "
                << contentName << " - c:" << startChunk + i
                << std::endl;

        if (! ccnCore->requestContent (contentName, startChunk + i) )
            DBGPRT(EVAUX, Warn, this->getFullPath())
                    << "Interest request for "
                    << contentName << " - c: " << startChunk + i
                    << ", failed!"
                    << std::endl;
    }

    return true;
};
```

Fig. IV.6. *La classe Ccn.cc.*

IV.6. Simulations

Avant de présenter nos deux scénarios d'attaque, nous commençons par expliquer comment créer un CCN en utilisant CCN-lite et OMNet++.

IV.6.1. Création d'un CCN

Un CCN est caractérisé par le déploiement de plusieurs nœuds. Pour notre simulation, il fallait commencer par créer un CCN comme suit :

- Etape1 : Créer une nouvelle topologie de type **'Network' :**

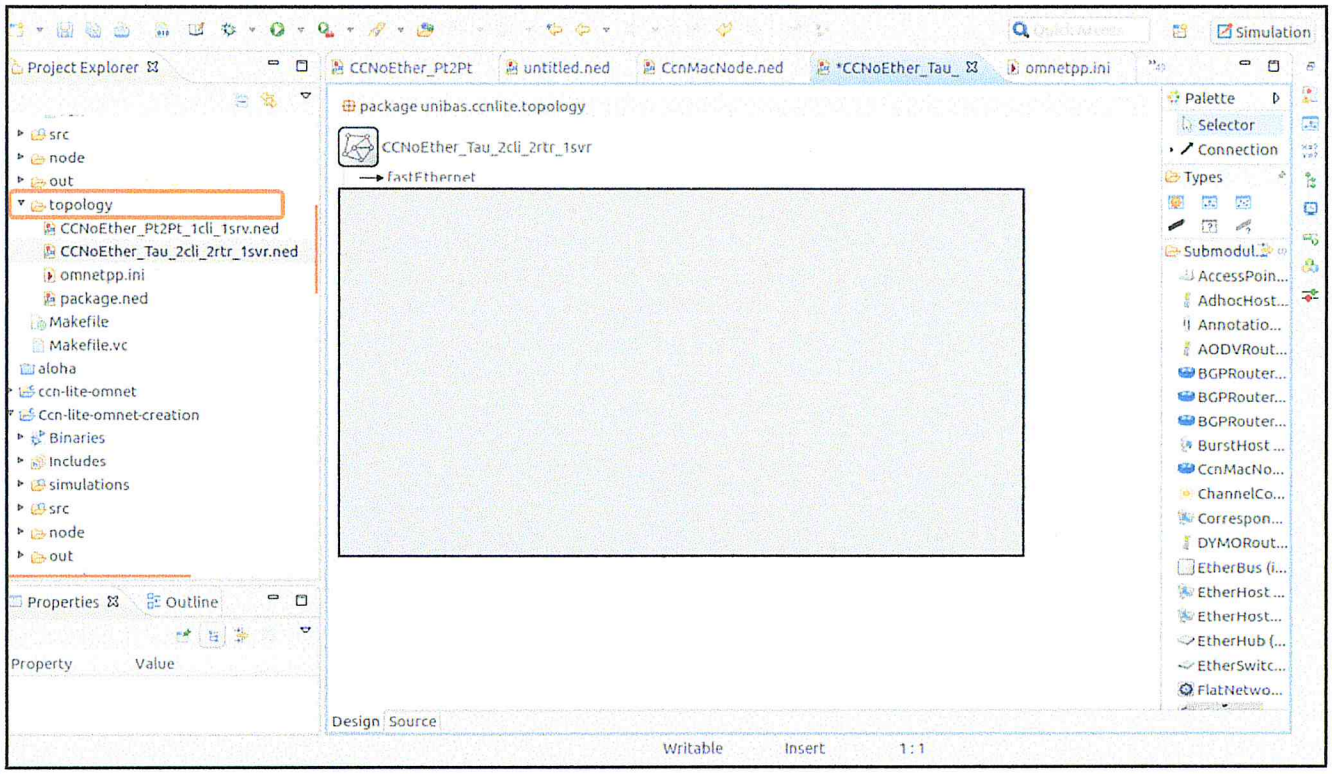


Fig. IV.7. *Création de Topologie de réseaux CCN.*

- Etape 2 : Créer les nœuds nécessaires pour notre topologie CCN : **server1**, **server2**, **client1**, **client2** et **attacker** (figure IV.8). Tous les nœuds ont la même structure « **CcnMacNode** » (figure IV.10). Le fichier graphique ".ned" (figure IV.11) illustre les différents modules utilisés par un nœud CCN

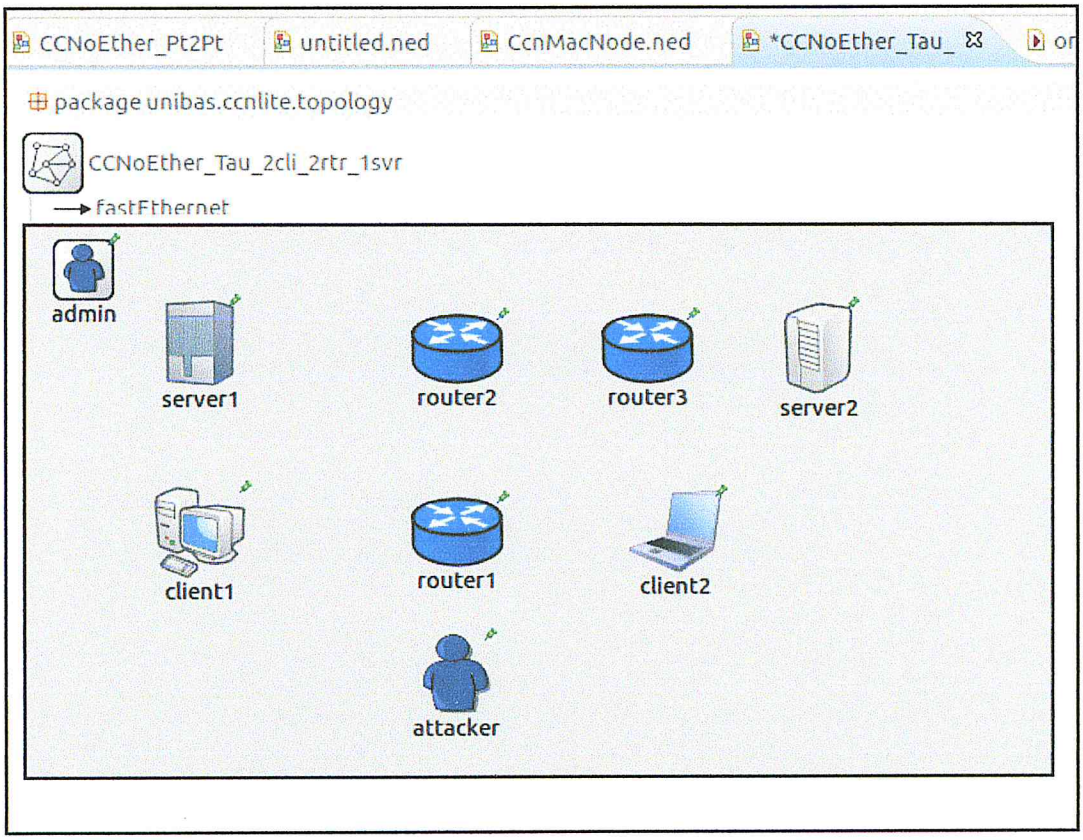


Fig. IV.8. *Création des nœuds de réseau CCN.*

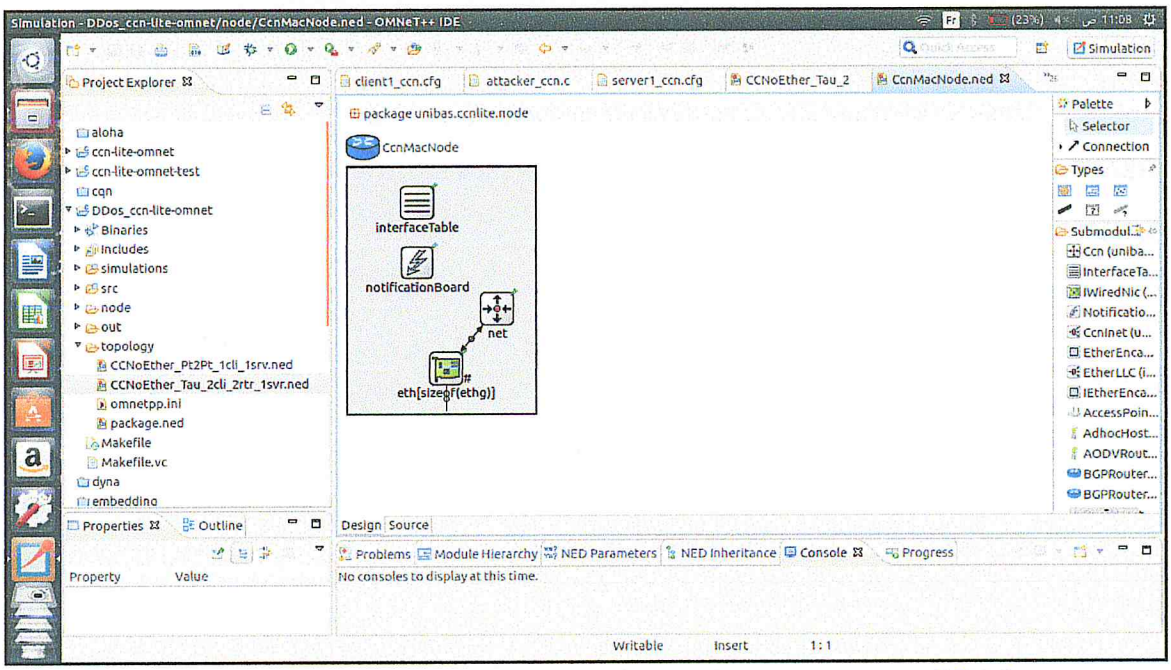


Fig. IV.9. *Interface graphique de simulateur OMNeT++ avec CCN-lite.*

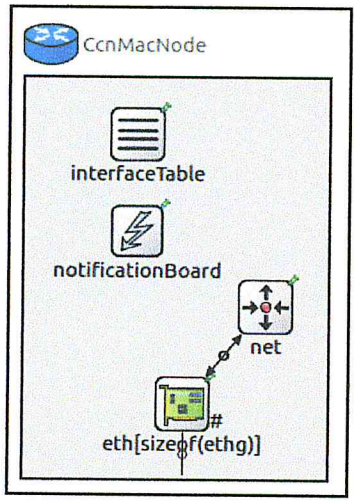


Fig. IV.10. *Fichier graphique ".ned" du nœud CCN.*

*CcnMacNode.ned | InterfaceTable. | NotificationBoa | Ccn.ned | IWir

```
import inet.networklayer.common.InterfaceTable;
import inet.linklayer.IWiredNic;
//import inet.mobility.IMobility;
import inet.base.NotificationBoard;
//import inet.mobility.models.StationaryMobility;
module CcnMacNode
{
    parameters:
        @display("i=abstract/router");
        @node;
        @labels(node,ethernet-node);
    gates:
        inout ethg[] @labels(EtherFrame-conn);
    submodules:
        // events pub-sub in a cross-layer fashion
        notificationBoard: NotificationBoard {
            parameters:
                @display("p=82,112");
        }
        // --- Network Layer
        net: Ccn {
            @display("p=176,167");
        }
        interfaceTable: InterfaceTable {
            parameters:
                @display("p=82,41");
        }
        // --- Link layer Wired Ethernet NICs
        eth[sizeof(ethg)]: <default("EthernetInterface")> like IWiredNic {
            parameters:
                @display("p=116,238,row,90;q=txQueue");
        }
    connections allowunconnected:
        // --- connect NICs to outside world and the network layer
        for i=0..sizeof(ethg)-1 {
            ethg[i] <--> eth[i].phys;
            eth[i].upperLayerOut --> net.ifIn++;    // NOTE: in older versions
            eth[i].upperLayerIn <-- net.ifOut++;
        }
}
```

Design | Source

Fig. IV.11. *Fichier de code source ".ned" d'un nœud CCN.*

- Etape 3 : Gérer les connexions entre les nœuds au niveau de la structure dans **"Connections"** du fichier de Topologie **".ned"**. Un réseau CCN est composé du module nœud décrit précédemment ainsi d'un canal pour la communication (**fastEthernet**) entre nœud.

```
connections:

    client1.ethg[0] <--> fastEthernet <--> router1.ethg[0];
    client2.ethg[0] <--> fastEthernet <--> router1.ethg[1];
    attacker.ethg[0] <--> fastEthernet <--> router1.ethg[3];
    router1.ethg[2] <--> fastEthernet <--> router2.ethg[0];
    router2.ethg[1] <--> fastEthernet <--> server1.ethg[0];
    router2.ethg[2] <--> fastEthernet <--> router3.ethg[1];
    server2.ethg[0] <--> fastEthernet <--> router3.ethg[0];
```

Fig. IV.12. *Gestion des Connexion pour tous les nœuds CCN.*

- Etape4 : Créer les fichiers de rôle pour chaque nœud dans le scénario qui sont fournis dans le fichier **".cfg "** inclus. Ces fichier contient :
  - **[eInterestMode]** : La table PIT qui contient **ContentName** (le nom de contenu), **StartChunk** (debut de segement), **ChunksCount** (nombre des segment) et **RequestTime** (temp de réponse).
  - **[ePreCacheMode] :** Le CS (le cache) qui contient **ContentName** (le nom de contenu), **StartChunk** (debut de segement), **ChunksCount** (nombre des segment) et **UpdateTime** (temp de réponse).
  - **[eFwdRulesMode] :** La table FIB qui contient **ContentPrefix** (prefix du nom de contenu), **NextHop** (interface de nœud suivant), **AccessFrom** (interface de sortie) et **UpdateTime** (temps de mise a jour).

CCNoEther_Tau_2 | omnetpp.ini | router1_ccn.cfg | client1_ccn.cfg | server1_ccn.cfg

```
[eInterestMode]
ContentName = ccnb:/b3c/wowmom/movie1   , StartChunk = 0 , ChunksCount = 10 , RequestTime = 0.1/*s*/
ContentName = ndntlv:/b3c/wowmom/movie2  , StartChunk = 0 , ChunksCount = 10, RequestTime = 0.1/*s*/

[ePreCacheMode]


[eFwdRulesMode]
ContentPrefix = ccnb:/b3c/wowmom , NextHop = router1.eth[0] , AccessFrom = client1.eth[0] , UpdateTime = 0/*s*/
ContentPrefix = ndntlv:/b3c/wowmom , NextHop = router1.eth[0] , AccessFrom = client1.eth[0] , UpdateTime = 0/*s*/

[eCommentsMode]
--------------------------------------------------------------------------------
comments go here
```

Fig. IV.13. *Fichier scénario de client1.*

- Etape5 : Créer le fichier **".ini"** (qui est lié au fichier NED) qui permet à l'utilisateur d'initialiser les paramètres des différents modules ainsi la topologie du réseau.

CCNoEther_Tau_2 | omnetpp.ini | router1_ccn.cfg | client1_ccn.cfg

```
##############################################################################
##  Experiment: CCN over Ethernet
##############################################################################

[Config CCNoEther_2c_3r_2s]

network = unibas.ccnlite.topology.CCNoEther_Tau_2cli_2rtr_1svr
description = "Example CCN over Ethernet Tau topology with 2 clients, 2 router

## topology/scenario settings
*.defaultDebugLevel = 4      ## for all simulation: 0=none, 1=Error, 2=Warning,
*.auxDebug = true            ## enable console debugging output

## per node settings
**.debugLevel = 4            ## per host: 0=none, 1=Error, 2=Warning, 3=Info, 4
**.minTxPace = 10ms
**.maxCacheSlots = 400
**.maxCacheBytes = 524288000Bytes
**.ccnCoreVersion = "CCN Lite v0.3.0"
*.client1.net.ccnScenarioFile = "client1_ccn.cfg"
*.client2.net.ccnScenarioFile = "client2_ccn.cfg"
*.attacker.net.ccnScenarioFile = " attacker_ccn.cfg"
*.router1.net.ccnScenarioFile = "router1_ccn.cfg"
*.router2.net.ccnScenarioFile = "router2_ccn.cfg"
*.server1.net.ccnScenarioFile = "server1_ccn.cfg"
*.router3.net.ccnScenarioFile = "router3_ccn.cfg"
*.server2.net.ccnScenarioFile = "server2_ccn.cfg"
```

Form | Source

Fig. IV.14. *Fichier omnetpp.ini.*

IV.6.2. Premier scénario d'attaque

a- Description du scénario

Un scénario montre un réseau CCN très simple composé de trois routeurs, un attaquant, deux clients normaux et deux serveurs. La création s'est faite comme décrit dans la section précédente.

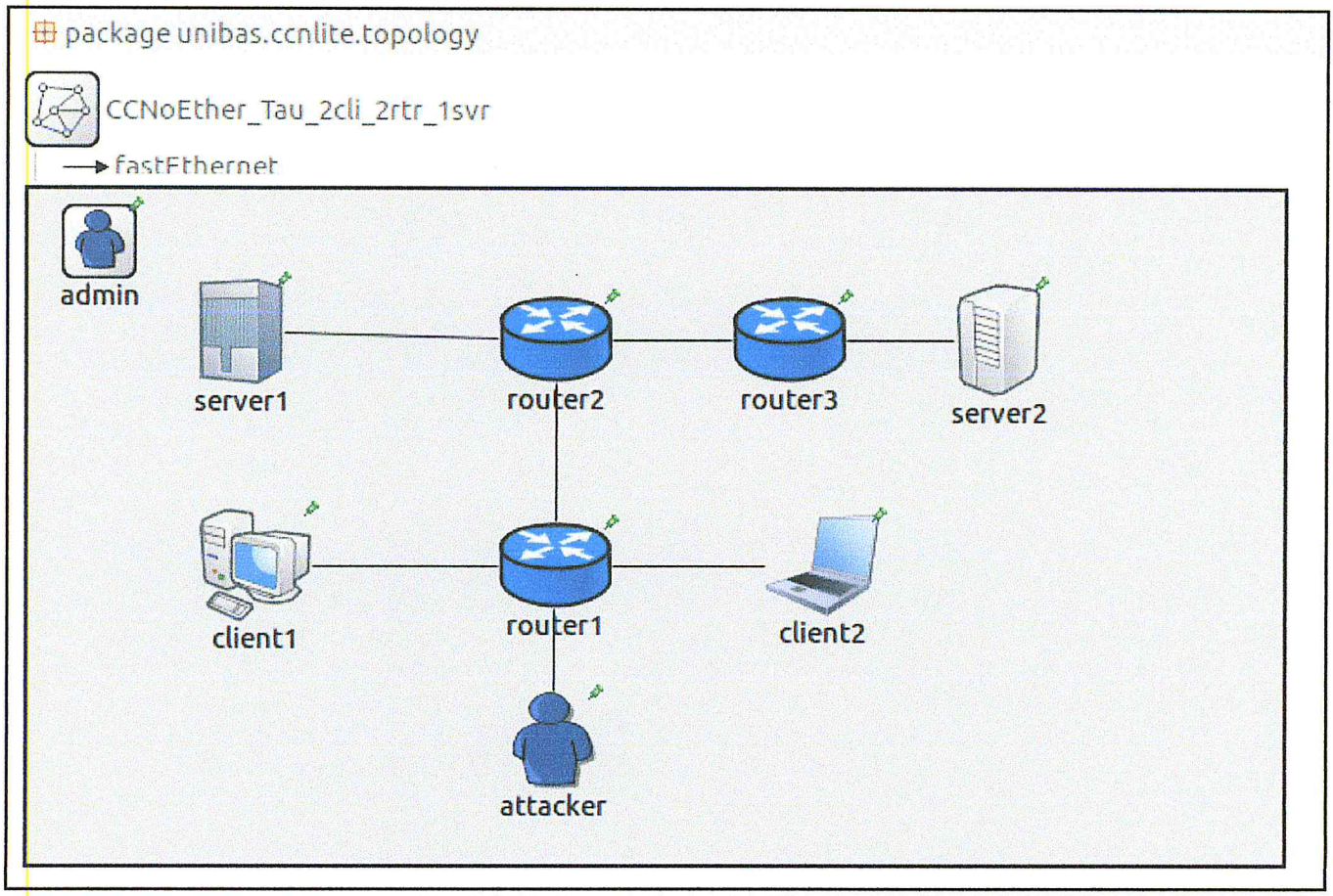


Fig. IV.15. *La topologie de 1er scénario.*

L'attaquant annonce un préfixe valide pour envoyer des faux intérêts vers le **server2**; Dans notre simulation, "**b3d/wowmom/badcnt/***" est un préfixe valide mais avec des noms de ressources non existants (**badcnt1**, **badcnt7**, …), voir la figure IV.16.

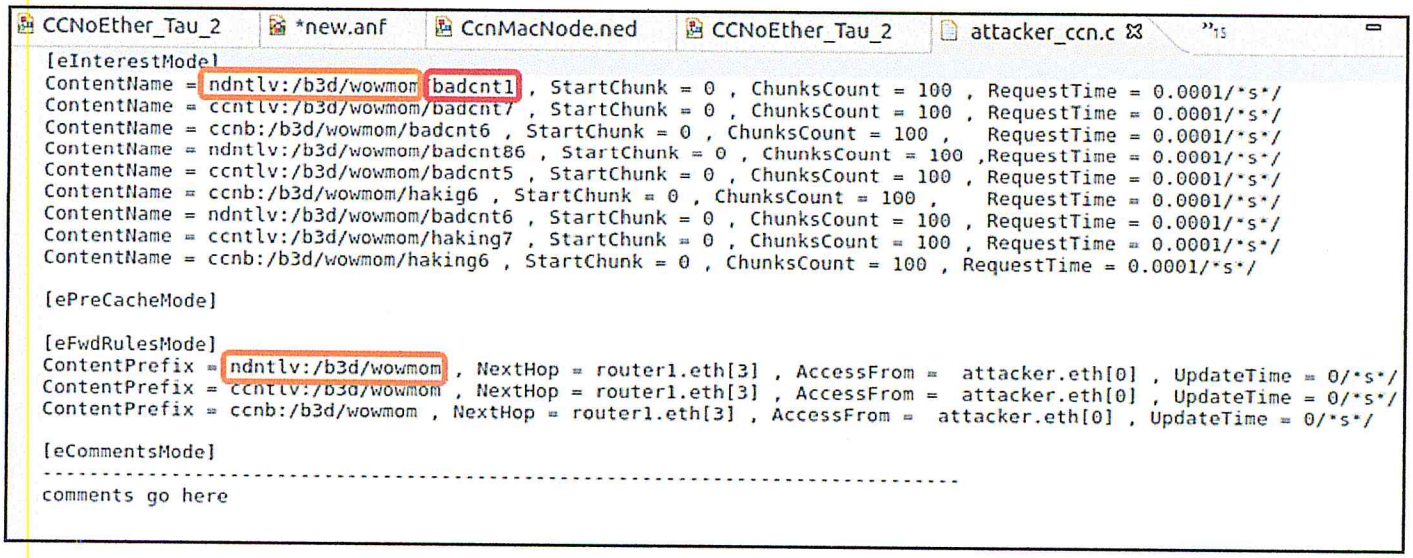


Fig. IV.16. *Les faux intérêts demandés par l'attaquant.*

Le rôle de l'attaquant est celui de générer des intérêts vers la fausse destination. Les routeurs du chemin entre (**router1** et **router 2**) sont inondés d'intérêts sans réponse, ce qui augmente leur consommation de mémoire donc le client sera bloqué et ne peut pas envoyer les intérêts valides à **Server1**.voir la figure IV.17.

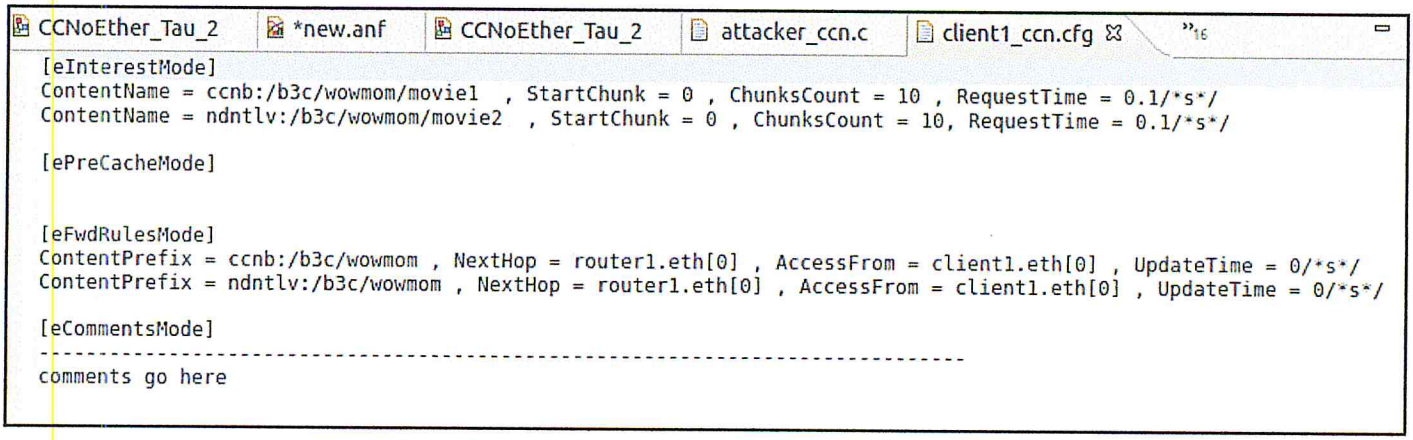


Fig. IV.17. *Les intérêts demandés par client1.*

Le **server2** est configurée pour recevoir ces datagrammes sans générer de réponse, mais aucune donnée utile ne sera renvoyée. Avec cette procédure simple, tous les Intérêts vus par **router1** (et pour lesquels **router1** crée une entrée dans le PIT) resteront dans la mémoire de l'appareil jusqu'à ce que le délai d'attente expire.

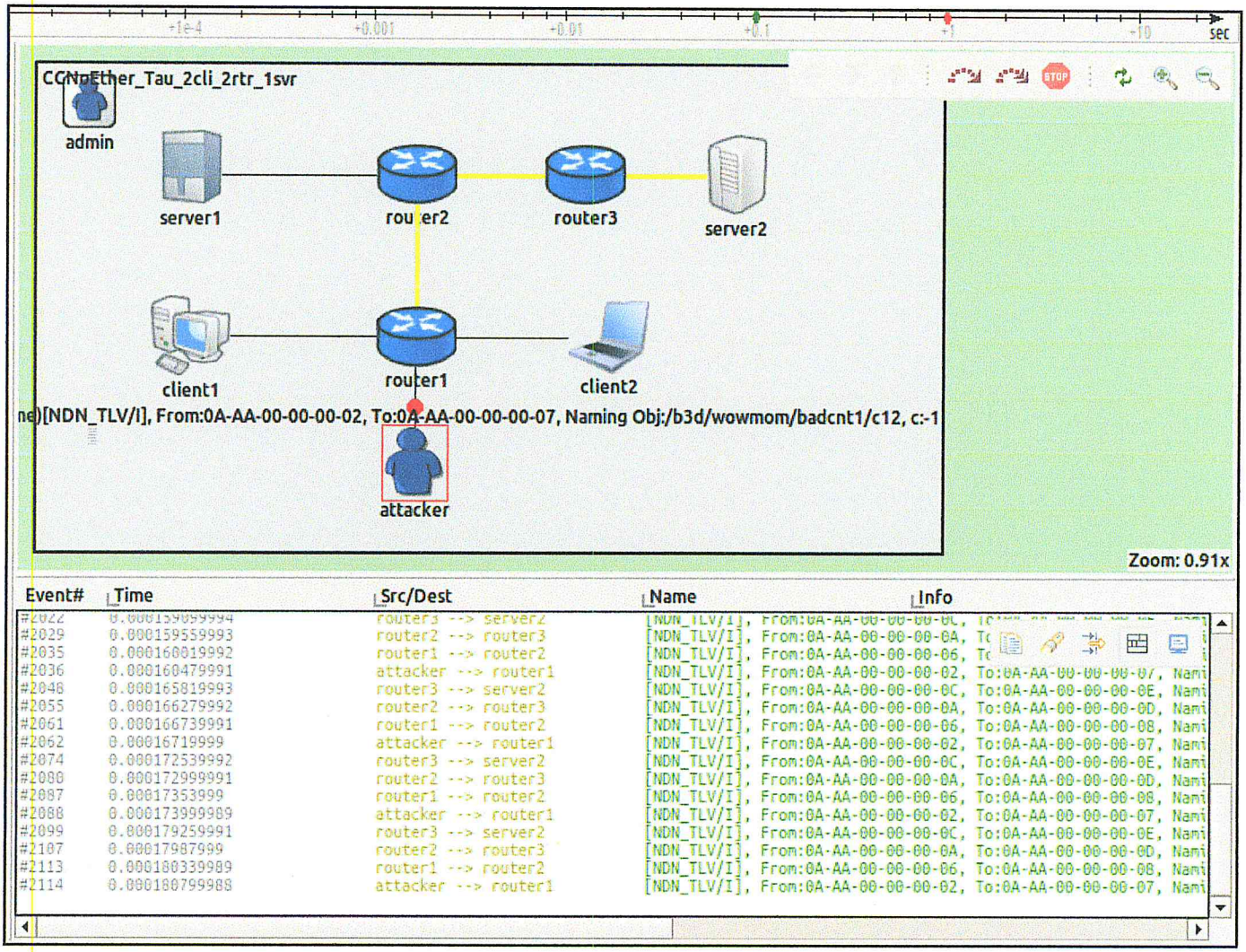


Fig. IV.18. *Surcharge de la table PIT de router1.*

b- Déroulement de Traceback

Au moment de la rédaction de ce mémoire, nous rencontrons toujours des problèmes liés à l'implémentation de l'algorithme Traceback dans CCN-lite (code source qui ne l'appartiennent pas). Donc, nous n'avons pas pu exécuter l'algorithme pour voir le résultat donné dans ce scénario. Cependant, nous allons donner dans cette section une vue sur son fonctionnement prévu pour cette attaque.

Comme dans la figure IV.19 le routeur 1 lance Traceback qui bloque la face qui est connectée avec l'attaqueur et libère la mémoire de la table PIT.

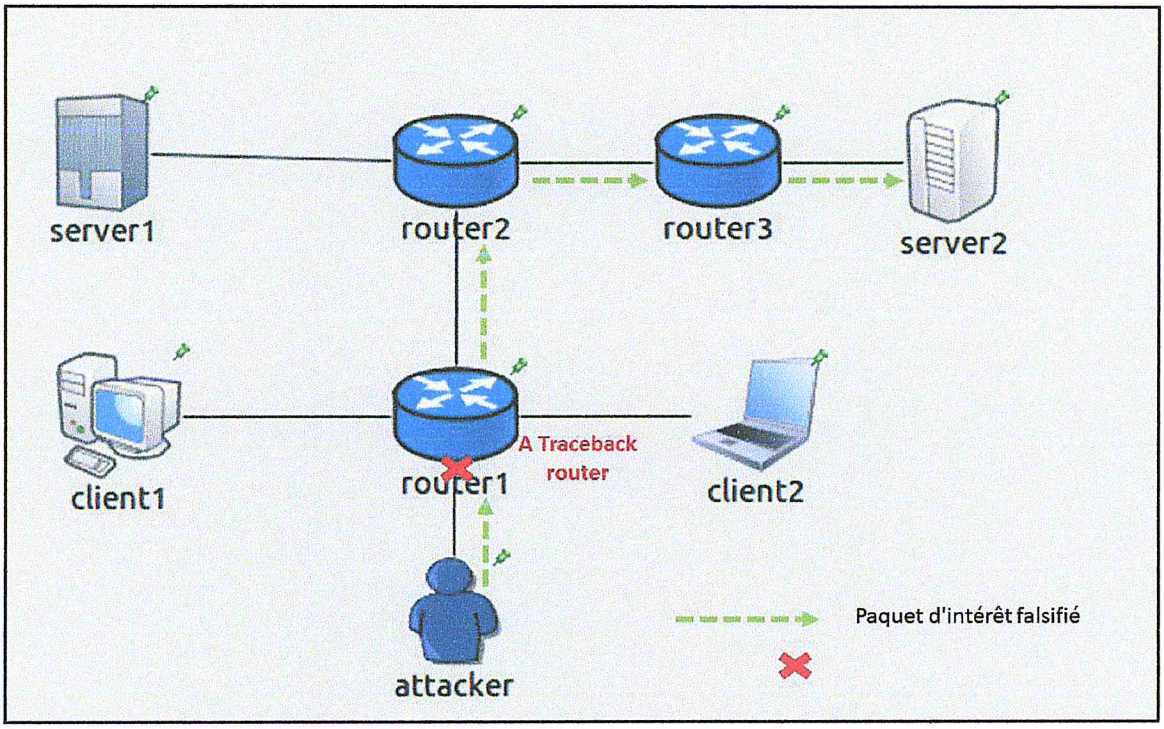


Fig. IV.19. *Traceback sur le 1er scénario.*

IV.6.3. Deuxième scénario d'attaque

a- Description du scénario

Ce scénario montre un réseau un peu compliqué composé de six routeurs, plusieurs attaquants, deux clients normaux et un serveur (**server2**).

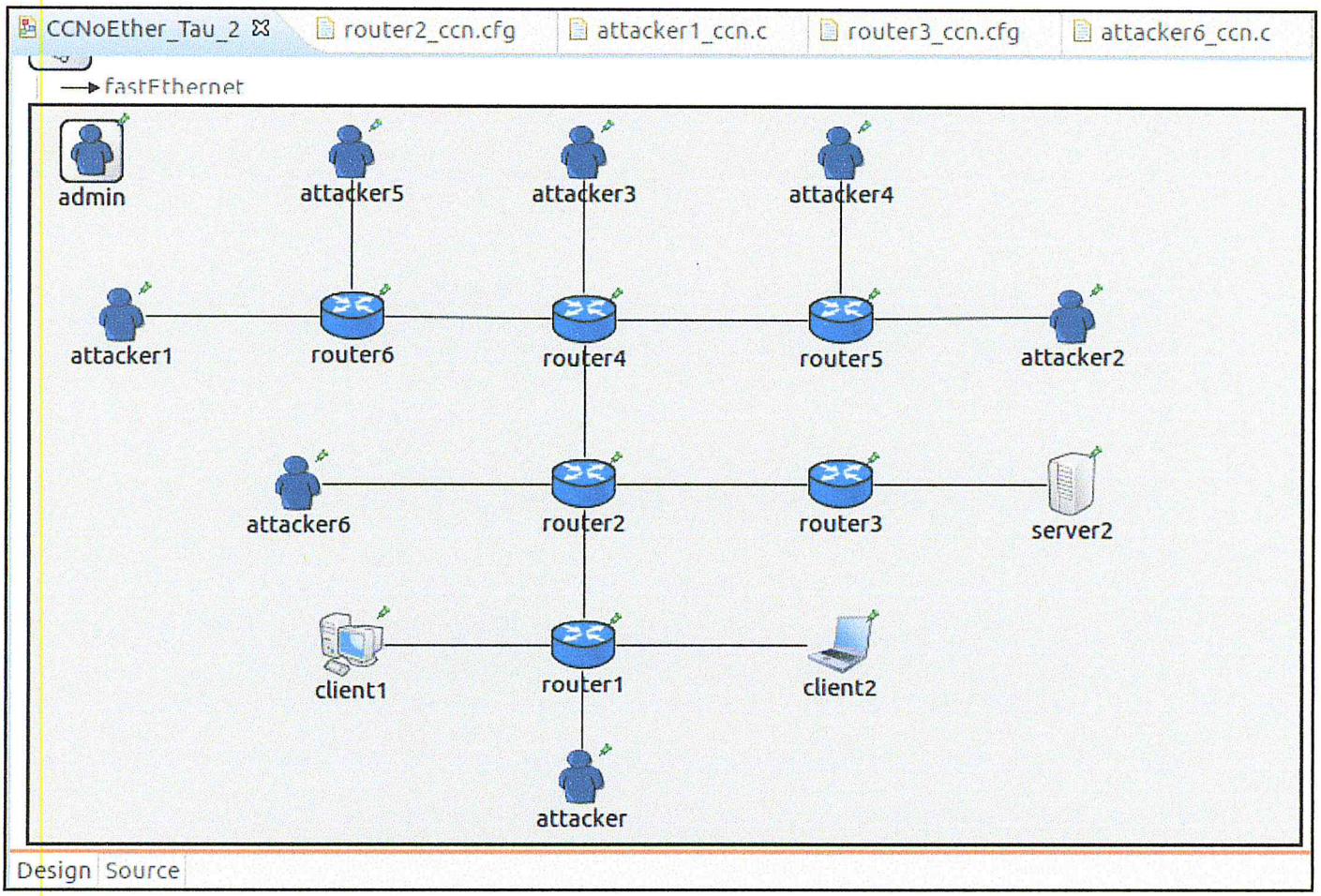


Fig. IV.20. *La topologie de 2ème scénario.*

Le rôle des attaquants est celui de générer des faux intérêts vers le serveur.

CcnAdmin.h | attacker_ccn.c | CcnAppMessage_m | CcnAppMessage_m | Ccn.cc

```
[eInterestMode]
ContentName = ndntlv:/b3d/wowmom/badcnt1h , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badcnt7h , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badhcking2h , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badhckingch , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badhckingh , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/bahckingdh , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badcnt1h , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badchckingnh , StartChunk = 0 , ChunksCount = 5, RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badcnt6h , StartChunk = 0 , ChunksCount = 3,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/bhckingh , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/bhckingadh7 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badcnthckinh , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badcnth , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badchckih , StartChunk = 0 , ChunksCount = 5 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badhckinhzck , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badcnthakin1 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badcnthakin7 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badhcking2haking , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/hckbadhckingcnt1 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badhckingcnt7 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/bahckinghackdcnt6 , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badcnt1hackin , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badchckinhac , StartChunk = 0 , ChunksCount = 5, RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badchakingnt6 , StartChunk = 0 , ChunksCount = 3,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/bhckingadchaknt1 , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/bhckihacking , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badhackinging6 , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
ContentName = ndntlv:/b3d/wowmom/badcnhaking , StartChunk = 0 , ChunksCount = 2 , RequestTime = 0.0001/*s*/
ContentName = ccntlv:/b3d/wowmom/badhackiningnt7 , StartChunk = 0 , ChunksCount = 5 , RequestTime = 0.0001/*s*/
ContentName = ccnb:/b3d/wowmom/badhchackingnt6 , StartChunk = 0 , ChunksCount = 2,   RequestTime = 0.0001/*s*/
```

Fig. IV.21. *Les faux intérêts générés par les attaquants.*

Les routeurs le long du chemin (router1, router2, router3) et autre chemin aussi sont inondés d'intérêts sans réponse, ce qui augmente leur consommation de mémoire donc les clients 1 et 2 serons bloqués et ne peuvent pas envoyer leurs intérêts valides au Serveur. Voir la figure IV.22.

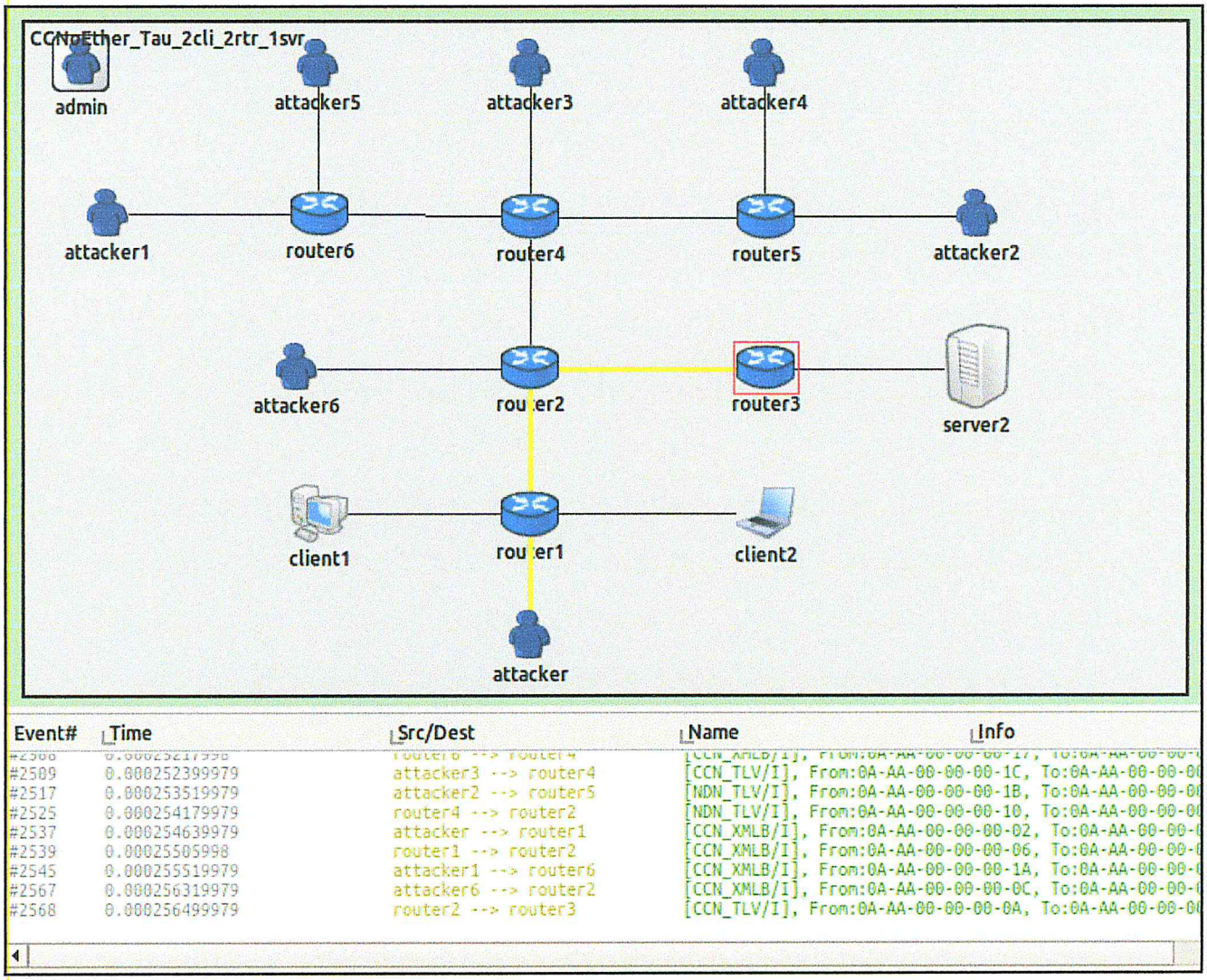


Fig. IV.22. *Le serveur 2 sous DDoS.*

La figure suivant illustre l'échange des paquets entres les attaqaunts et les routeurs une absence complète pour les demandes valides par le client 1 et 2 .

Event#	Time	Src/Dest
#2263	0.000228339983	router6 --> router4
#2271	0.000230479982	attacker2 --> router5
#2278	0.000231199982	attacker3 --> router4
#2299	0.000232179982	router4 --> router2
#2309	0.000232179983	router1 --> router2
#2311	0.000232639982	attacker --> router1
#2320	0.000233599982	attacker1 --> router6
#2328	0.000234079982	attacker6 --> router2
#2339	0.000234659982	router5 --> router4
#2344	0.000234839983	router3 --> server2
#2345	0.000235299982	router2 --> router3
#2352	0.000236339982	router6 --> router4
#2360	0.000237999981	attacker3 --> router4
#2367	0.000238479981	attacker2 --> router5
#2373	0.000239139981	router4 --> router2
#2386	0.000240099982	router1 --> router2
#2387	0.000240559981	attacker --> router1
#2403	0.000241599981	attacker1 --> router6
#2409	0.000241779981	router5 --> router4
#2418	0.000242079981	attacker6 --> router2
#2419	0.000242339981	router2 --> router3
#2420	0.000242359982	router3 --> server2
#2428	0.000244179981	router6 --> router4
#2434	0.00024479998	attacker3 --> router4
#2442	0.00024599998	attacker2 --> router5
#2449	0.00024665998	router4 --> router2
#2461	0.00024759998	attacker --> router1
#2463	0.000248019981	router1 --> router2
#2472	0.00024855998	attacker1 --> router6
#2492	0.00024919998	attacker6 --> router2
#2493	0.000249399981	router3 --> server2
#2494	0.00024945998	router2 --> router3
#2495	0.00024961998	router5 --> router4
#2508	0.00025217998	router6 --> router4
#2509	0.000252399979	attacker3 --> router4
#2517	0.000253519979	attacker2 --> router5
#2525	0.000254179979	router4 --> router2
#2537	0.000254639979	attacker --> router1
#2539	0.00025585998	router1 --> router2
#2545	0.000255519979	attacker1 --> router6
#2567	0.000256319979	attacker6 --> router2
#2568	0.000256499979	router2 --> router3

Fig. IV.23. *Surcharge de PIT par les attaquants.*

b- Déroulement de Traceback

Au moment de la rédaction de ce mémoire, nous rencontrons toujours des problèmes liés à l'implémentation de l'algorithme Traceback dans CCN-lite (code source qui ne l'appartiennent pas). Donc, nous n'avons pas pu exécuter l'algorithme pour voir le résultat donné dans ce scénario. Cependant, nous allons donner dans cette section une vue sur son fonctionnement prévu pour cette attaque.

Le routeur 2 envoie les paquets de données falsifiées afin de satisfaire les paquets d'intérêt des attaquants, et c'est transmis au routeur 1 et 4. Le routeur 4 et 1 recherche le nom du contenu du paquet de données falsifié dans son PIT, et transmet le paquet de données falsifié aux attaquants (voir figure IV.24).

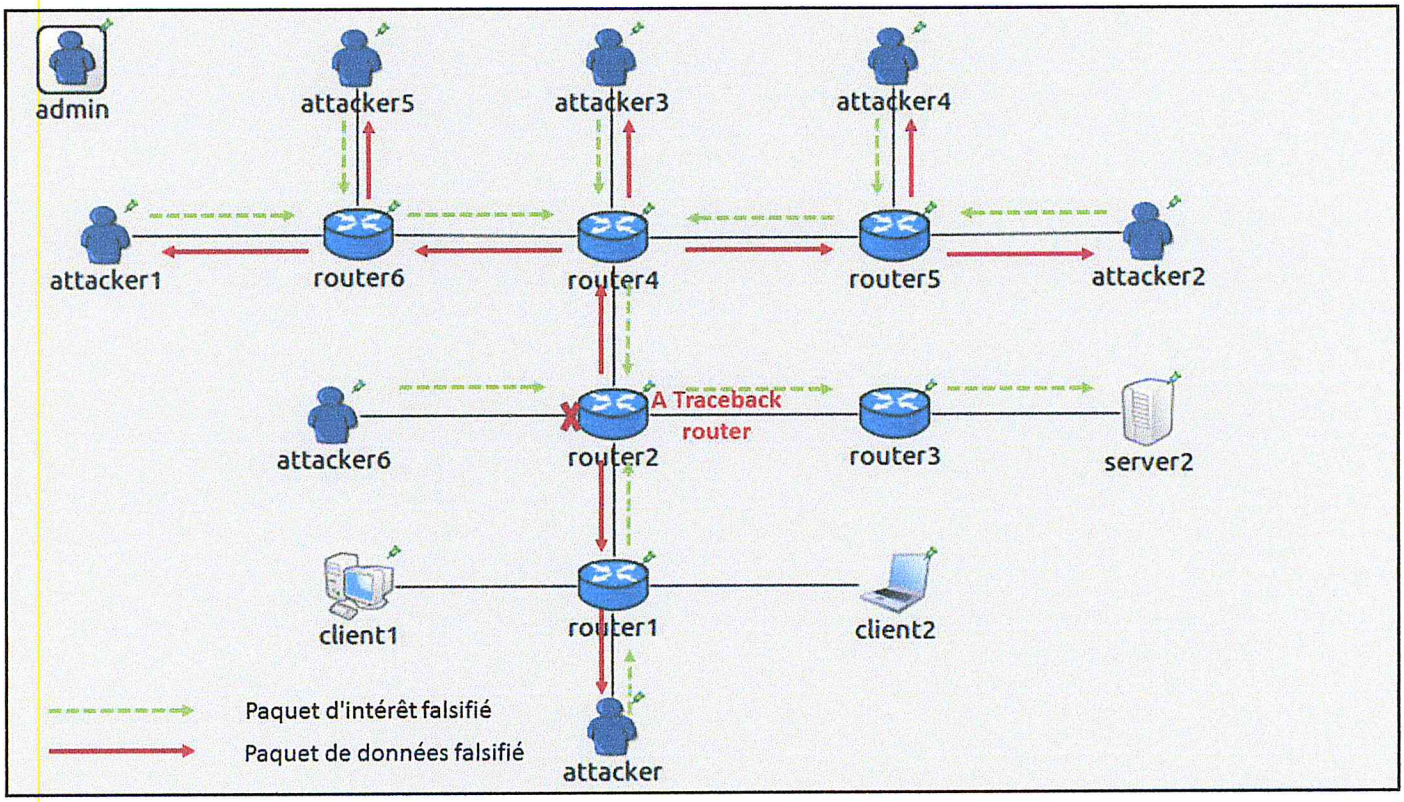


Fig. IV.24. *Traceback sur le 2ème scénario.*

IV.6. Conclusion

Dans ce chapitre, nous avons présenté l'environnement OMNET++ et faire une description pour le package CCN-lite. Ce package nous a permis de faire des simulations des attaques d'Interest Flooding dans les CCNs et d'implémenter l'algorithme Traceback pour faire face à ce type d'attaque. Deux scénarios d'attaques ont été simulés et testés.

CONCLUSION GENERALE ET PERSPECTIVES

Les architecture d'Internet du futur offre des exigences élevées en matière de diffusion de l'information, ce qui motive le milieu de la recherche à trouver d'autres solutions. Etant donné que l'une de ces solutions, ICN se concentre sur le contenu afin de fournir une distribution de contenu évolutive et efficace. Il qui le rendent unique à partir d'architectures centrées sur l'information. Il dépend principalement de la dénomination indépendante de localisation, de la mise en cache dans le réseau et du routage basé sur les noms.

Dans la littérature, il existe de nombreuses propositions pour les architectures ICN comme DONA, NetInf, NDN et PURSUIT. Après une comparaison de ces architectures, nous nous sommes orientés vers l'architecture du CCN (Content Centric Network) où nous avons étudiés son système de routage et de sécurité. Bien qu'il existe des aspects de sécurité et de confidentialité qui s'améliorent grâce à CCN, cela pose également de nouveaux défis liés au routage centré contenu telles que les attaque d'infrastructure, de source, de Blocage mobile, de Flooding, de timing, de Jamming, de Hijacking et d'Interception. Parmi ces attaques, nous nous sommes intéressés par l'attaque dénis de service distribuées (DDoS), communément appelées les attaques d'inondations d'intérêt (Interest Flooding Attack, IFA). Dans un scénario spécifique et concret d'attaques DDoS dans CCN, nous avons démontré la possibilité d'attaques DDoS où les attaquants font usage de la règle d'acheminement des paquets d'intérêt de CCN pour envoyer des paquets d'intérêt avec des noms falsifiés comme des paquets d'attaque. Par la suite, les routeurs peuvent identifier que les attaquants ciblent les tableaux d'intérêt en attente (PIT) qui est la plus grande victime.

Pour lutter contre les attaques DDoS, nous avons fait une étude détaillée des solutions existantes qui peuvent être mis en place dans les CCNs. D'après notre étude, nous avons retenu l'algorithme Traceback qui permet, entre autres, de bloquer les faces qui sont directement connectées aux hackers et de libérer la mémoire PIT. Afin de valider notre approche, nous avons intégré CCN-lite avec OMNeT++ (> v4.2.2) et INET Framework (> v1.99.4.). CCN-lite est une implémentation réduite et légère des protocoles CCN notamment le routage centré contenu. Elle nous a permis de créer un CCN, de simuler deux scénarios d'attaques DDoS et de tester l'algorithme Traceback.

Au moment de la rédaction de ce mémoire, nous rencontrons toujours des problèmes liés à l'implémentation de l'algorithme Traceback dans CCN-lite. Donc, nous

n'avons pas pu exécuter l'algorithme mais nous avons donné une vue sur son fonctionnement prévu dans nos deux scénarios d'attaque (Section VI.6.2 et Section VI.6.3). Nous pensons après plusieurs essaies qu'il est trop difficile d'implémenter l'algorithme dans CCN-lite (code source qui ne l'appartiennent pas), mais ce n'est pas impossible dans le futur proche. Une fois implémenté, plusieurs tests devront être faits sur l'algorithme pour ajuster ses paramètres et calculer ses performances. Ensuite, nous suggérons, comme perspectives, de :

- Intégrer les autres solutions de lutte contre DDoS et de comparer avec l'algorithme Traceback.
- Simuler d'autres types d'attaques liés au routage Hijacking et l'attaque d'Interception

Annexe :

Sécurité des Réseaux Informatique

I- Définition de la sécurité informatique :

La sécurité informatique est un terme général couvrant un vaste domaine du monde de l'informatique et du traitement d'informations. Les industries qui dépendent de systèmes informatiques et de réseaux pour effectuer des transactions commerciales quotidiennes et accéder à des informations d'une importance capitale considèrent leurs données comme une partie essentielle de leur actif. La sécurité informatique est un domaine vaste qui peut appréhender dans plusieurs domaines

- Les systèmes d'informations
- Les réseaux informatiques
- Les accès physiques à des salles machines

Nous nous concentrerons sur les aspects relatifs aux réseaux informatiques.

II- Objectifs de la sécurité informatique

Les solutions de sécurité doivent contribuer à satisfaire au moins les critères suivants : la confidentialité, l'intégrité et la disponibilité des ressources et des informations des réseaux et des systèmes :

- **La confidentialité**, vise à assurer que seuls les sujets (les personnes, les machines ou les logiciels) autorisés aient accès aux ressources et aux informations auxquelles ils ont droit. La confidentialité a pour objectif d'empêcher que des informations secrètes soient divulguées à des sujets non autorisés. L'objectif des attaques sur la confidentialité est d'extorquer des informations **[MIC 06]** ;
- **L'intégrité** vise à assurer que les ressources et les informations ne soient pas corrompues, altérées ou détruites par des sujets non autorisés. L'objectif des attaques sur l'intégrité est de changer, d'ajouter ou de supprimer des informations ou des ressources **[MIC 06]** ;
- **La disponibilité** vise à assurer que le système soit bien prêt à l'emploi, que les ressources et les informations soient en quelque sorte consommables, que les ressources ne soient pas saturées, que les informations, les services soient accessibles et que l'accès au système par des sujets non autorisés soit prohibé.

L'objectif des attaques sur la disponibilité est de rendre le système inexploitable ou inutilisable **[MIC 06]**.

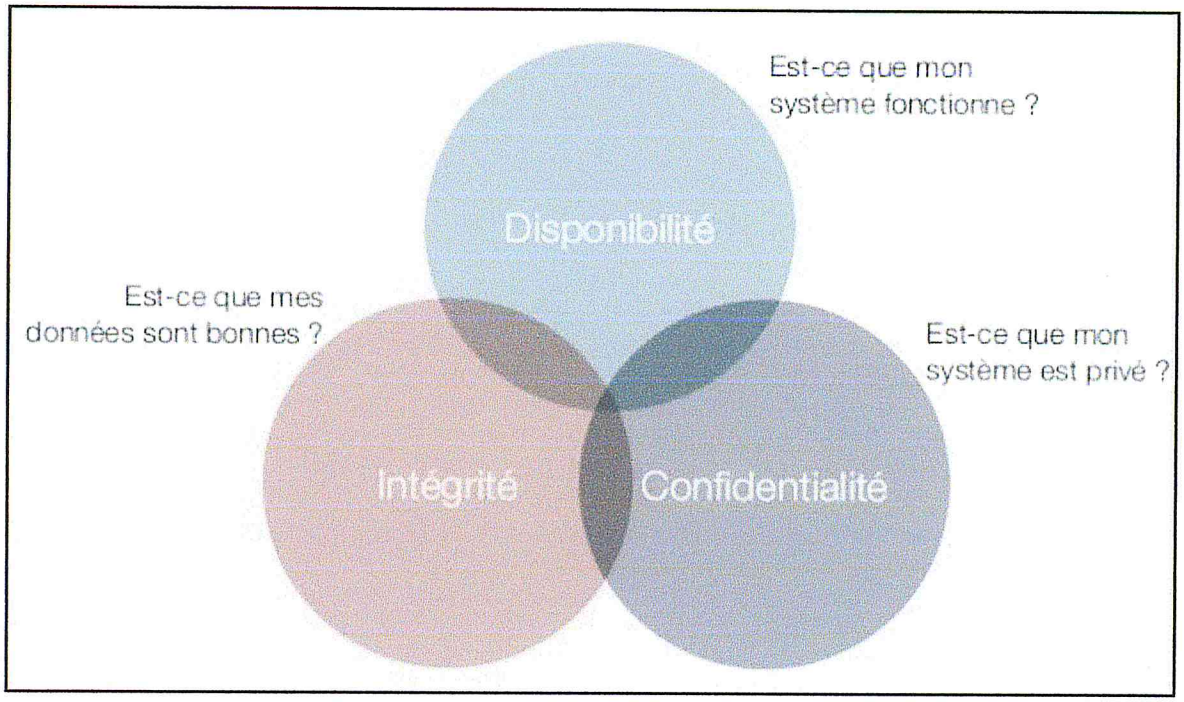


Fig. A.1. *Les objectifs de la sécurité informatique.*

La cryptologie permet de remplir largement les deux premiers objectifs en confidentialité et en intégrité. Malheureusement, il n'existe pas de modèles pour parvenir entièrement à la finalité de disponibilité.

III- Menaces de sécurité courante

Lorsqu'il est question de sécurité des réseaux, les trois notions qui interviennent habituellement dans la discussion sont la vulnérabilité, les menaces et les attaques.

III.1. Vulnérabilités :

La vulnérabilité est le degré de faiblesse inhérent à tout réseau ou périphérique. Cela concerne les routeurs, les commutateurs, les ordinateurs de bureau, les serveurs et même les périphériques de sécurité.

Les menaces viennent d'individus compétents intéressés par l'exploitation des faiblesses de sécurité. Il est prévisible que de tels individus continueront à rechercher de nouvelles faiblesses et de nouveaux exploits. Ces menaces sont mises en œuvre à l'aide d'une variété d'outils, de scripts et de programmes permettant de lancer des attaques contre des réseaux et leurs périphériques. En général, les périphériques réseaux attaqués sont des points d'extrémité comme les serveurs et les ordinateurs de bureau.

Les vulnérabilités ou faiblesses principales sont au nombre de trois :

1) **Faiblesses technologiques :** Les technologies informatiques et de réseau ont des faiblesses de sécurité intrinsèques. Celles-ci comprennent les faiblesses du protocole TCP/IP, du système d'exploitation et de l'équipement réseau.

2) **Faiblesses de configuration :** Les administrateurs réseau et les ingénieurs système doivent apprendre ce que sont les faiblesses de configuration et les compenser en configurant convenablement leurs équipements informatiques et réseau.

3) **Faiblesses dans la stratégie de sécurité :** Il existe des risques de sécurité pour le réseau si les utilisateurs ne respectent pas la stratégie de sécurité. La figure présente certaines faiblesses de la stratégie de sécurité et la manière dont elles sont exploitées.

III.2. Menaces pour l'infrastructure physique :

Lorsqu'il est question de la sécurité du réseau, on pense généralement aux pirates exploitant des vulnérabilités logicielles. Une menace moins connue est celle qui plane sur la sécurité physique des périphériques. Un assaillant peut empêcher l'utilisation des ressources du réseau si celles-ci sont physiquement compromises.

Les quatre catégories de menaces physiques sont les suivantes **[CIS 04]** :

1) **Menaces matérielles** : entraînant des dommages physiques aux serveurs, routeurs, commutateurs, installations de câblage et stations de travail.

2) **Menaces environnementales :** dues aux variations extrêmes de la température ou du taux d'humidité.

3) Menaces électriques : provenant de pointes de tension, d'une tension d'alimentation trop basse (baisse de tension), d'une alimentation non filtrée (bruit) et de la perte totale d'alimentation.

4) Menaces de maintenance : dues à un traitement inadéquat de composants électriques essentiels (décharge électrostatique), au manque de pièces de rechange critiques, à un mauvais câblage et à un mauvais étiquetage.

III.3. Menaces envers les réseaux :

Les menaces envers les réseaux peuvent être regroupés dans les quatre catégories principales suivantes (voir *Fig. I.1*) **[CIS 04]** :

1) ***Menaces non organisées*** **:** Ces menaces proviennent pour la plupart d'individus inexpérimentés qui utilisent des outils de piratage facilement accessibles comme des scripts d'interpréteur de commande et des casseurs de mot de passe. Même si elles ne sont utilisées que pour tester l'habilité de l'assaillant, ces menaces peuvent causer de sérieux dommage aux réseaux. Par exemple, le piratage du site Web d'une entreprise peut nuire à la réputation de cette entreprise. Même si le site Web est séparé des informations privées protégées derrière un pare-feu, le public n'en est pas informé. Vu de l'extérieur, le site est donc considéré comme un environnement peu sûr pour des activités commerciales.

2) ***Menaces organisées*** **:** Les menaces organisées viennent d'individus isolés ou en groupes, qui sont beaucoup plus motivés et techniquement compétents. Ces personnes connaissent les vulnérabilités des systèmes et utilisent des techniques de piratage évoluées pour pénétrer dans des entreprises qui ne se doutent de rien. Ils accèdent par intrusion à des ordinateurs d'entreprises ou d'institutions gouvernementales pour y commettre des actes de fraude, pour détruire ou altérer des données, ou simplement pour semer le chaos. Ces groupes sont souvent impliqués dans des cas de fraude ou de vol signalés aux organismes chargés de l'application de la loi. Leur technique de piratage est si complexe que seuls des enquêteurs très spécialisés arrivent à comprendre ce qui se passe.

3) ***Menaces externes*** **:** Les menaces externes peuvent provenir d'individus ou d'organisations agissant depuis l'extérieur d'une entreprise et qui ne sont pas autorisés à accéder au réseau et aux ordinateurs de cette dernière. Ils arrivent à pénétrer dans un réseau à partir d'Internet ou de serveurs d'accès commuté. Les menaces externes peuvent présenter des degrés de gravité variables selon que l'assaillant est un amateur (non organisé) ou un expert (organisé).

4) ***Menaces internes*** **:** Des menaces internes existent lorsque quelqu'un dispose d'un compte ou d'un moyen d'accès physique autorisé au réseau. Tout comme pour les menaces externes, la gravité d'une menace interne dépend de l'expertise de l'assaillant.

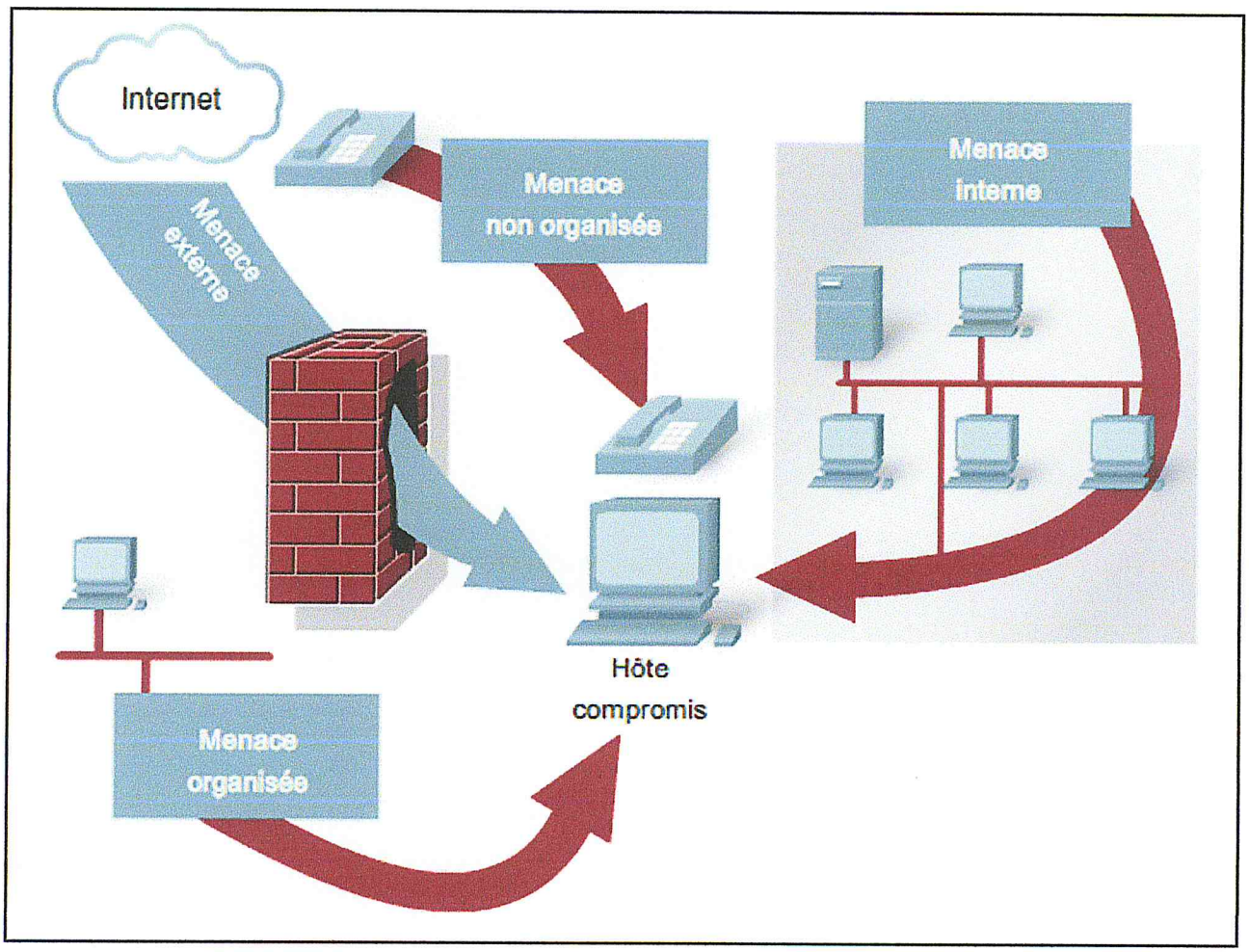


Fig. A.2. *Menaces envers les réseaux* **[CIS 04]**

IV- Menaces de sécurité courante

Il existe quatre catégories principales d'attaques **[CIS 04]** :

IV.1. Reconnaissance :

La reconnaissance est la découverte non autorisée des systèmes, de leurs adresses et de leurs services, ou encore la découverte de leurs vulnérabilités. Il s'agit d'une collecte d'informations qui, dans la plupart des cas, précède un autre type d'attaque. La reconnaissance est similaire au repérage effectué par un cambrioleur à la recherche d'habitations vulnérables, comme des maisons inoccupées, des portes faciles à ouvrir ou des fenêtres ouvertes.

Les attaques de reconnaissance peuvent avoir les formes suivantes :

- Demandes d'informations Internet
- Balayages ping
- Balayages de ports
- Analyseurs de paquets

IV.2. Accès :

L'accès au système est la possibilité pour un intrus d'accéder à un périphérique pour lequel il ne dispose pas d'un compte ou d'un mot de passe. La pénétration dans un système implique généralement l'utilisation d'un moyen de piratage, d'un script ou d'un outil exploitant une vulnérabilité connue de ce système ou de l'application attaquée.

Les attaques d'accès exploitent des vulnérabilités connues dans les services d'authentification, les services FTP et les services Web pour accéder à des comptes Web, à des bases de données confidentielles ou à toute autre information sensible. Exemple d'attaques d'accès :

- Attaques de mot de passe.
- Exploitation de la confiance
- Redirection de port
- Attaque de l'homme du milieu

IV.3. Déni de service :

Le déni de service (DoS, en anglais) apparaît lorsqu'un pirate désactive ou altère un réseau, des systèmes ou des services dans le but de refuser le service prévu aux utilisateurs normaux. Les attaques par déni de service mettent le système en panne ou le ralentissent au point de le rendre inutilisable. Le déni de service peut consister simplement à supprimer ou altérer des informations. Dans la plupart des cas, l'attaque se résume à exécuter un programme pirate ou un script. C'est pour cette raison que les attaques par déni de service sont les plus redoutées.

Les attaques DoS peuvent prendre de nombreuses formes. Elles empêchent l'utilisation d'un service par les personnes autorisées en épuisant les ressources du système. Voici quelques exemples des menaces DoS les plus courantes :

- Ping fatal
- Inondation SYN
- Attaques DDos
- Attaque Smurf

IV.4. Logiciels malveillants :

Des logiciels malveillants peuvent être installés sur un ordinateur hôte dans le but d'endommager ou d'altérer un système, de se reproduire ou d'empêcher l'accès à des réseaux, systèmes ou services. Ces programmes sont généralement appelés vers, virus et chevaux de Troie.

V- Lutte contre les attaques d'un réseau

V.1. Contre les attaques de reconnaissance :

Le chiffrement assure la protection des données susceptibles d'attaques, que ce soit sous forme d'écoute électronique, de craquage de mots de passe ou de manipulation. Pratiquement toutes les entreprises transmettent des transactions dont l'écoute électronique par un pirate peut avoir de lourdes conséquences. Le chiffrement des données sensibles garantit qu'une transmission susceptible d'être écoutée ne peut être ni observée ni altérée.

Une opération de déchiffrement est nécessaire lorsque les données chiffrées arrivent sur l'hôte de destination.

Une méthode de chiffrement appelée « *payload-only* » en anglais consiste à ne chiffrer que la partie des données utiles qui suit un en-tête UDP (User Datagram Protocol) ou TCP. De cette manière, les routeurs et les commutateurs Cisco peuvent lire les informations de couche réseau et acheminer le trafic comme tout autre paquet IP. Le chiffrement des données utiles seules permet de traiter la commutation du flux et les fonctions de listes d'accès du trafic chiffré comme s'il s'agissait d'un trafic de texte en clair tout en préservant la qualité de service (QoS) pour toutes les données.

V.2. Contre les attaques d'accès :

Il est possible de limiter les **attaques de mot de passe** en apprenant aux utilisateurs à définir des mots de passe complexes et en spécifiant une longueur de mot de passe minimale. La parade contre les attaques en force consiste à limiter les échecs de connexion. Une attaque en force peut cependant aussi s'effectuer hors ligne. Par exemple, si un pirate intercepte un mot de passe crypté lors d'une écoute électronique ou en accédant à un fichier de configuration, il pourrait tenter de casser ce mot de passe sans se connecter à l'hôte.

Les **attaques par exploitation de la confiance** peuvent être limitées par la sévérité des niveaux de confiance au sein du réseau. Par exemple, Il est possible de déployer des réseaux locaux virtuels privés dans des segments de services publics comportant plusieurs serveurs publics. Les systèmes protégés par un pare-feu ne devraient jamais se fier totalement aux systèmes situés à l'extérieur de ce pare-feu. Une telle confiance doit être limitée à des protocoles spécifiques et l'authentification doit se baser, dans la mesure du possible, sur des éléments autres que l'adresse IP.

La **redirection de port** peut être limitée principalement par des modèles de confiance appropriés, qui sont spécifiques au réseau (comme mentionné plus haut). En cas d'attaque, un système de détection des intrusions (IDS) installé sur l'hôte permet de détecter le pirate et de l'empêcher d'installer de tels utilitaires de redirection sur un hôte.

Sur un réseau étendu, la parade aux **attaques de l'homme du milieu** consiste à utiliser des tunnels (réseaux virtuels privés). Le pirate ne voit alors que du texte chiffré, impossible à déchiffrer. Les attaques de l'homme du milieu sur des réseaux locaux se font

avec des outils d'empoisonnement ARP ou d'utilitaires comme Ettercap. Sur les réseaux locaux, les attaques de l'homme du milieu peuvent être limitées en configurant la sécurité des ports sur les commutateurs.

V.3. Contre les attaques de déni de service :

Les attaques DoS et DDoS peuvent être limitées en mettant en place des listes de contrôle d'accès spéciales contre l'usurpation et le déni de service. Les FAI peuvent également mettre en œuvre une limitation du trafic non essentiel dans les segments de réseau. Un exemple typique consiste à limiter le trafic ICMP autorisé dans un réseau, étant donné que ce trafic n'est utilisé qu'à des fins de diagnostic.

V.4. Contre les attaques des logiciels malveillants :

Les applications de ce type peuvent être circonscrites par un logiciel antivirus efficace au niveau de l'utilisateur et au niveau du réseau. Les logiciels antivirus peuvent détecter la plupart des virus et des chevaux de Troie et les empêcher de se propager dans le réseau. La mise à jour de ces logiciels en fonction des nouveaux développements de virus et chevaux de Troie permet également de se prémunir plus efficacement contre les attaques. À mesure que de nouveaux virus ou chevaux de Troie apparaissent, les entreprises doivent acquérir la version la plus récente de leur logiciel antivirus.

BIBLIOGRAPHIE

[AFA 13] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and Lixia Zhang. Interest flooding attack and countermeasures in named data networking. In IFIP Networking Conference, 2013.

[ATH 15] Athanasios V. Vasilakos, Zhe Li, Gwendal Simon, Wei You. Information Centric Network: Research challenges and opportunities. Athanasios, 2013.

[CIS 01] Programme installable CISCO networking academy : CCNA Exploration 4.0: Notion de base sur les réseaux.

[CIS 04] Programme installable CISCO networking academy : CCNA Exploration 4.0: Accès au réseau étendu.

[COM 13] A. Compagno, M. Conti, P. Gasti and G. Tsudik. Poseidon: Mitigating Interest Flooding DDoS Attacks in Named Data Networking. In 38th Annual IEEE Conference on Local Computer Networks (LCN 2013), Sydney, Australia

[DAI 12] H. Dai, B. Liu, Y. Chen, and Y. Wang. On pending interest table in named data networking. In ANCS '12, Austin, TX, USA, Oct. 2012.

[DAI 13] H. Dai, Y. Wang, J. Fan, and B. Liu. Mitigate ddos attacks in ndn by interest traceback. In NOMEN'13, Turin, Italy, Apr. 2013

[ESL 15] Eslam G. AbdAllah, Hossam S. Hassanein, and Mohammad Zulkernine.: A Survey of Security Attacks in Information-Centric Networking, IEEE COMMUNICATION SURVEYS & TUTORIALS, VOL. 17, NO. 3, THIRD QUARTER 2015.

[GIT 15] OMNeT++ getting started. Consulté le : 06/03/2017, sur : https://github.com/cn-uofbasel/ccn-lite/blob/master/doc/internal/omnetpp-getting-started.pdf.

[INE 17] Introduction INET Framework. Consulté le : 25/08/2017, sur : https://inet.omnetpp.org/Introduction.html

[JAC 09] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," in CoNext 2009, 2009.

[LAG 12] Lagutin D, Visala K, Tarkoma S. Publish/subscribe for internet: PSIRP perspective. IOS Press; 2012.

[LIT 15] CCN-lite Project and Community, Consulté le : 23/03/2017, sur : http://ccn-lite.net/

[MAT 13] Matteo Virgilio, Guido Marchetto and Riccardo Sisto, PIT Overload Analysis in Content Centric Networks, Department of Control and Computer Engineering, Politecnico di Torino, Torino, Italy, 2013.

[MAT 15] Matteo Virgilio, Guido Marchetto and Riccardo Sisto, Interest Flooding Attack Countermeasures Assessment on Content Centric Networking, Department of Control and Computer Engineering, Politecnico di Torino, Torino, Italy, 2015.

[MIC 06] Michel Riguidel ; la sécurité des réseaux et des systèmes ; ENST Paris ;2006.

[MSK 05] C. Mallanda, A. Suri, V. Kunchakarra, S.S. Iyengar, R. Kannan and A. Durresi "Simulating Wireless Sensor Networks with OMNeT++", S. Sastry The University of Akron, Akron, Ohio.

[NAD 14] Mme. Nada SBIHI. "Gestion du trafic dans les réseaux orientés contenus". Thèse de Doctorat, Université Pierre et Marie Curie - Paris 6, 2014.

[OMN 04] "OMNeT++, Discrete Event Simulation System Version 4.0", User Manual.

[REZ 16] Reza Tourani, Travis Mick, Satyajayant Misra, and Gaurav Panwar. Security, Privacy, and Access Control in Information-Centric Networking : A Survey, 2016.

[ROM 16] Roman Lutz, "Security and Privacy in Future Internet Architectures", College of Information and Computer Sciences, University of Massachusetts Amherst, USA, 2016.

[SAI 13] SAIL. FP7-ICT-2009-5-257448/D-2.4. Technical report, Sail, Febreury 2013.

[SAN 15] Sandvine. "Global Internet Phenomena Report - 2H 2014. Technical report, Sandvine, 2015.

[WEI 14] Wei You. A Content-Centric Networking Node for a Realistic Efficient Implementation and Deployment. Networking and Internet Architecture. Télécom Bretagne, Université de Rennes 1, 2014.

[YOU 12] W. You, B. Mathieu, P. Truong, J. Peltier, and G. Simon. Dipit: a distributed bloom-filter based pit table for ccn nodes. In ICCCN '12, Munich, Germany, July 2012.