POLITECNICO DI MILANO

DIPARTIMENTO DI
ELETTRONICA,
INFORMAZIONE
E BIOINGEGNERIA

Scuola di ingegneria industriale e dell'informazione

Corso di laurea in ingegneria delle telecomunicazioni

Anno Accademico 2012/2013

Abuso delle Content Centric Networks per distribuire contenuto indesiderato

Relatore

Giacomo Verticale

Autore

Riccardo Raspadori, matricola 781402

Abstract

Today's IP-based Internet architecture is plagued by several security threats.
Not only that, but in a world in which content production is exploding thanks to loss of friction in their creation and distribution, its client-server model for the retrieval of information no longer suits the needs of its users.
So, new paradigms, like Content Centric Networking, which propose to put contents, instead of places and connections, as the pivotal point of the system, have emerged in recent years.
With them, new sets of threats have been thought that specifically target the properties of the new proposals, especially the widespread use of caches.
In this work, it is proposed and deeply analyzed such an attack, and it is clearly shown, through carefully designed simulations, how meaningful advantages can be provided to its perpetrator.
In the concluding chapter, some possible countermeasures described in earlier studies and not, are presented and their adaptability to the case at hand is discussed.
Their effectiveness, though, is material for further studies.

Sommario

L'attuale architettura di Internet, basata su IP, è afflitta da molteplici problematiche di sicurezza.
Non solo questo, ma in un mondo nel quale il fenomeno della produzione di contenuti sta esplodendo, grazie all'abbassamento della difficoltà nella loro creazione e distribuzione, il suo modello client-server per il recupero dei contenuti non è più adatto a soddisfare le esigenze degli utenti.
Per questo, nuovi paradigmi, come quello del Content Centric Networking, che propongono di porre i contenuti, invece che i nodi e le connessioni, come centro attorno al quale gira il sistema, sono emersi negli ultimi anni.
Assieme ad essi, sono stati pensati nuovi insiemi di minacce che prendono di mira le specifiche proprietà delle nuove proposte, specialmente la capillare diffusione delle cache.
In questo lavoro, un attacco di questo tipo è proposto ed attentamente analizzato ed è chiaramente mostrato, attraverso simulazioni attentamente concepite, come vantaggi significativi possano essere ottenuti dal suo perpetratore.
Nel capitolo conclusivo sono presentate alcune possibili contromisure descritte e no in precedenti studi e viene discussa la loro potenziale adattabilità al caso in analisi.
La loro efficacia, però, costituisce materiale per future ricerche.

Indice dei contenuti

Indice delle figure

Indice delle tabelle

Indice dei grafici

Capitolo 1 – Introduzione

Ai tempi in cui IP venne progettato e successivamente implementato per diventare la base fondamentale dell'architettura di Internet, i suoi ideatori non avevano preventivato l'incredibile sviluppo nell'estensione e nell'utilizzo di quest'ultima a cui si è assistito nei successivi decenni.
Tale crescita ha dapprima riguardato il numero dei nodi connessi ed in seguito la varietà di applicazioni e servizi che è stato possibile creare su questa nuova piattaforma in grado di raggiungere anche gli angoli più remoti del pianeta.
Il modello client-server di internet è quindi progressivamente entrato in crisi; siccome agli albori i computer connessi erano un numero molto limitato e non c'erano segnali che potessero far presagire il cambiamento epocale costituito dalla loro capillare diffusione, lo spazio di indirizzamento fu progettato per essere molto largo secondo i canoni dell'epoca, ma non abbastanza secondo quelli attuali.
All'emersione pratica della problematica è stato fatto ricorso a misure che potessero mitigare il problema mantenendo però la compatibilità con lo standard già stabilito, inizialmente nella forma del *Network Address Translation* (NAT) e poi nella transizione alla versione 6 del protocollo, che, pur risolvendo in modo definitivo il problema della disponibilità degli indirizzi, introduce ulteriore overhead e l'esigenza di nuovi apparati che possano gestire entrambe le versioni (IPv4 e IPv6).
Questa transizione ad IPv6 è stata annunciata per anni, senza mai prendere effettivamente piede; solo ultimamente, secondo un report [21] di Akamai, il noto provider CDN, il cambiamento sta effettivamente avvenendo, con il numero di richieste IPv6 servite passato da 10 a 20 miliardi al giorno in 6 mesi.
Inoltre, come conseguenza della presenza fisica dei computer nelle sole organizzazioni governative o nelle università, la rete era sempre considerata come sicura, poiché tutte le entità che ne facevano parte potevano ragionevolmente essere considerate fidate, per cui tutti i protocolli di comunicazione in tale rete, compreso IP, vennero progettati senza considerare adeguatamente gli aspetti di sicurezza, poiché quella non era l'esigenza dell'epoca e non si pensava che lo potesse divenire in futuro.
Chiaramente, con l'aumento sia degli utenti, che degli interessi attorno alle informazioni scambiate su Internet, queste supposizioni di fiducia nell'interlocutore sono venute meno ed il protocollo IP si è mostrato in tutta la sua vulnerabilità rispetto a tali problematiche.
Una terza importante scossa alle fondamenta dell'attuale architettura di Internet è entrata in atto negli anni più recenti e si tratta dell'esplosione dei contenuti disponibili attraverso la rete.
Mano a mano che gli utilizzatori crescevano, anche il numero dei cosiddetti produttori di contenuti cresceva; testate giornalistiche hanno cominciato a pubblicare le loro notizie online e numerosi individui, spesso coloro più tecnicamente avvezzi, iniziavano a trasformarsi da consumatori in maniera esclusiva a loro stessi produttori.
Durante grossomodo i primi 15 anni di vita del *World Wide Web*, però, le barriere di ingresso per effettuare tale passaggio sono state così elevate che il numero di produttori è sempre stato molto inferiore rispetto a quello di consumatori.

Ma con l'avvento dei primi servizi di *social networking*, come YouTube, MySpace e successivamente Facebook, Twitter e via via tutti gli altri e successivamente con la diffusione degli *smartphone*, la cardinalità degli insiemi di produttori e consumatori si è sempre più velocemente avvicinata, dal momento che la barriera di ingresso è ormai così bassa che virtualmente tutti i consumatori possono essere produttori.
Se si pensa a qual è il metodo di interazione con la rete più comune per la maggioranza degli utenti, si può sostenere che esso consista nell'inserimento di una query su Google.
Il motore di ricerca poi funge da router e traduttore e produce una serie di puntatori a posizioni all'interno della rete, ovvero pagine su server web.
Ciò accade perché l'essere umano non sa a priori dove si trovi un'informazione alla quale è interessato, ma inserisce il nome di questa sul motore di ricerca, che come output restituisce il dove essa può essere rinvenuta e tale strato di traduzione è una conseguenza del design di IP, che, invece, richiede la conoscenza preventiva del dove il contenuto desiderato possa essere trovato.
Ci troviamo quindi con un problema, l'incremento massivo della quantità di contenuti ed informazioni presenti in rete e la soluzione sbagliata, un'architettura che opera non basandosi sul cosa viene richiesto, ma sul dove questo può essere trovato.
Questa è quindi la motivazione che ha spinto numerose ricerche ad essere intraprese nella direzione di paradigmi architetturali vertenti dal principio sulla sicurezza ed incentrati sui contenuti piuttosto che sulla loro posizione.
L'architettura alternativa che sta ricevendo maggiore interesse è senz'altro quella proposta da Jacobson et al. in [22] [23], che prende il nome di *Content Centric Networking* (CCN d'ora in avanti) e che verrà discussa in maggiore dettaglio nel capitolo successivo.
Qui resta da dire che pur proponendo un sistema di base molto più sicuro di quanto possibile col solo IP, le CCN hanno caratteristiche che a loro volta rendono il paradigma sensibile a determinate tipologie di attacco.
Lo scopo di questo lavoro è quello di studiare in maniera analitica ed approfondita una variante di una di queste minacce per determinarne il grado di efficacia.
In particolare, tramite l'esecuzione di numerose simulazioni, verrà provato come un potenziale produttore malevolo di contenuti può essere in grado di utilizzare le cache in possesso degli utenti di una CCN per favorire i propri contenuti, falsificandone la popolarità.
Inoltre, si quantificheranno i vantaggi che questo approccio può portare, così come gli effetti che reca agli ignari utenti i cui dispositivi vengono compromessi.
Nel capitolo conclusivo sono poi elencate alcune potenziali contromisure che meritano ulteriore approfondimento, il che è però al di fuori dell'ambito del presente testo.

Il resto del documento è così organizzato: nel capitolo 2 si offrirà una breve panoramica sul funzionamento delle CCN e sulle caratteristiche si sicurezza.
Il capitolo 3 riassumerà le tipologie di attacco rivolte nello specifico a queste reti e fornirà una panoramica sullo stato dell'arte della ricerca nell'ambito.
Il capitolo 4 presenterà la descrizione del problema e le motivazioni dell'attaccante, insieme agli scenari specifici realizzati, ai parametri ed alle metriche utilizzate nell'analisi.
Il capitolo 5 discuterà i risultati ottenuti nel corso delle simulazioni svolte.
Infine nel capitolo 6 saranno riassunte le contromisure proposte e presentate le conclusioni.

Capitolo 2 – Le Content Centric Networks e lo scenario di riferimento

La struttura di questa introduzione di alto livello seguirà una logica di illustrazione delle caratteristiche tecniche delle CCN e di come queste si possono paragonare a quelle delle reti basate su IP.
Dapprima ci si concentrerà sulla descrizione dei meccanismi di base che consentono il funzionamento dell'infrastruttura, poi si illustreranno le proprietà che fanno di questo paradigma una soluzione che da maggiori garanzie di sicurezza rispetto ad IP.
Infine verrà proposta un'illustrazione della situazione generale cui il presente lavoro fa riferimento e dalla quale poi discende la topologia specifica che è utilizzata nelle simulazioni.

2.1 - Struttura della comunicazione

La comunicazione all'interno di una CCN è completamente in mano al ricevitore dell'informazione.
Infatti, solo due tipi di pacchetto sono presenti in CCN, *interest* e *data* e nessuno di essi contiene alcuna informazione né sul mittente, né sul ricevente.
Gli interest costituiscono le richieste, mentre i dati sono le informazioni con cui la sorgente risponde.
Un contenuto può essere consegnato solo in risposta ad un precedente interest e all'invio di un singolo interest deve corrispondere solo un singolo contenuto.
Inoltre è l'host che ha inviato l'interest che deve curarsi di ritrasmetterlo se questo non viene soddisfatto.
Perciò, a differenza di IP, un host non ne può contattare un secondo in maniera diretta, può però raggiungere qualsiasi contenuto il secondo abbia generato e viceversa.
Da notare anche il fatto che molteplici interest possono essere soddisfatti da un singolo invio di un contenuto, il che fa risparmiare banda.
Un interest si dice soddisfatto quando il nome del contenuto per cui esso è indirizzato ed il nome del contenuto stesso sono uguali.
Ad esempio, un interest destinato a */nyc/moma/klimt/hope2* verrebbe soddisfatto da un ipotetico database contenente dati su New York City, mentre */nyc/moma/monet/themanneporte* non restituirebbe alcunché, dal momento che il quadro è ospitato al Metropolitan Museum of Art e non al MoMA.
CCN fa uso pervasivo del caching, il che significa che i contenuti vengono memorizzati da ciascun hop lungo il percorso dalla sorgente del dato al suo destinatario.

2.2 - Struttura dei nodi ed instradamento delle informazioni

Ogni nodo è dotato di molteplici interfacce, che vengono definite come *faces.*
Il termine face in CCN non indica per forza un'interfaccia di rete, il che significa che potrebbe anche trattarsi di un'interfaccia verso un'applicazione locale al nodo.
Ciascuno dei nodi appartenenti ad una CCN è costituito da 3 parti essenziali, elencate nel seguito:

- Content Store (o CS, indicato genericamente come cache nel seguito): non è nient'altro che la memoria dedicata alla memorizzazione dei contenuti e che, pertanto, permette al nodo di soddisfare gli interest in arrivo per i contenuti qui memorizzati.
- Pending Interest Table (PIT): si tratta di una tabella in cui il nodo inserisce il nome di ogni interest ricevuto insieme alla face da cui esso è stato ricevuto, eliminandolo quando esso consegna sul percorso inverso il contenuto.
  Contrariamente ai router IP, questo significa che i router CCN mantengono uno stato che tiene traccia di cosa viene consegnato e cosa no in risposta alle richieste.
- Forwarding Information Base (FIB): questa è l'equivalente della tabella di routing IP; per ciascun nome di cui il nodo è a conoscenza viene associato un insieme di face utilizzabili per raggiungere una potenziale sorgente.

Quando un interest giunge ad un nodo, prima di tutto viene controllata la presenza di un contenuto con tale nome nel CS; se così fosse il contenuto viene automaticamente consegnato attraverso la stessa interfaccia da cui è giunto l'interest.
Se, al contrario, il contenuto non è trovato, allora viene controllata la PIT; se lo stesso nome richiesto è già presente, la face da cui proviene l'interest viene aggiunta alla riga associata al nome e l'interest stesso è eliminato.
In questo modo si evita di instradare inutilmente una nuova richiesta, ma l'interest sarà lo stesso soddisfatto perché il contenuto verrà consegnato a tutte le interfacce presenti nella riga relativa al suo nome.
Se, invece, nella PIT non c'è traccia del nome, una nuova riga viene creata e ad essa è associata la face da cui questo interest è provenuto.
In questo caso viene consultata anche la FIB, perché l'interest va instradato sulla face indicata da tale tabella.
Perciò, in contrasto a quanto viene fatto nel routing IP, vengono instradati solo gli interest e mai i contenuti, che invece si limitano a seguire il percorso tracciato dai primi.
Inoltre, grazie al fatto che ogni nodo è dotato di una cache, c'è anche la possibilità che un contenuto sia restituito da uno dei nodi situati intermedi del percorso, liberando quindi i collegamenti a monte.

2.3 - Modello di sicurezza

Il problema di sicurezza che si pone nell'ambito della moderna infrastruttura di rete e che l'architettura CCN si propone di risolvere *by design* è quello del rinvenimento di contenuti validi ed integri, cioè che siano effettivamente corrispondenti a ciò che è stato richiesto e che non siano stati modificati irregolarmente durante il tragitto.
Allo stato corrente delle cose, nelle reti IP, vengono verificate le sorgenti tramite le firme digitali e vengono cifrati i pacchetti, il che significa che la fiducia nella consegna di informazioni valide ed integre è riposta nella sorgente e nel canale.
Di converso, in CCN, viene autenticato il legame tra nome e contenuto stesso, ovvero la firma viene calcolata tramite un hash sia del nome, che del contenuto, che di alcune informazioni aggiuntive che facilitano la fase di verifica della firma.

In questo modo si ottengono vantaggi su molteplici fronti; il nome può essere leggibile, per un essere umano o per un'applicazione, al contrario di quanto proposto in altri approcci [24], che assegnano al contenuto il risultato dell'hash del contenuto.
Soprattutto la fiducia del richiedente può essere associata al contenuto stesso, piuttosto che al modo o al luogo da cui questo è stato rinvenuto, il che significa la possibilità di poter ricevere un contenuto perfettamente genuino da qualsiasi parte della rete, non solo da sorgenti e attraverso collegamenti certificati.
Alla ricezione del contenuto richiesto, un utente ha l'obbligo di verificarne la firma per accertarne la validità ed integrità.
Chiaramente, per fare questo, deve esistere un meccanismo per la gestione della fiducia, anche se nelle CCN molti dei compiti pratici annessi a tale pratica vengono semplificati, come il rinvenimento delle chiavi, che divengono semplicemente altri contenuti, o il fatto che la pubblicazione di una chiave genera automaticamente un certificato, dato il legame crittografico imposto tra ogni oggetto ed il proprio nome.
O ancora la possibilità di creare meccanismi di gestione della fiducia ad hoc a seconda delle specifiche esigenze di un'applicazione o servizio.
Grazie all'ampio ed intelligente uso che viene fatto della crittografia, attraverso i metodi fino ad ora descritti, architetture di questo tipo sono costruite su solide basi per garantire un livello di protezione adeguato per le comunicazioni che avvengono su di esse.
Questo campo di ricerca è ancora relativamente nuovo, perciò molto lavoro rimane ancora da effettuare sugli aspetti della sicurezza e molti dettagli del modello visto sono ad oggi ancora da rifinire.
Se però, da un punto di vista della genuinità dei contenuti l'architettura si presenta come altamente sicura, rimangono aperte altre opportunità per entità malevole, soprattutto legate all'abbondanza di cache nella rete.
Vedremo i dettagli di questa discussione nel seguente capitolo.

2.4 - Definizione dello scenario generale di riferimento

Una delle numerose rappresentazioni astratte della internet è quella mostrata nella figura seguente (Fig. 2.1).

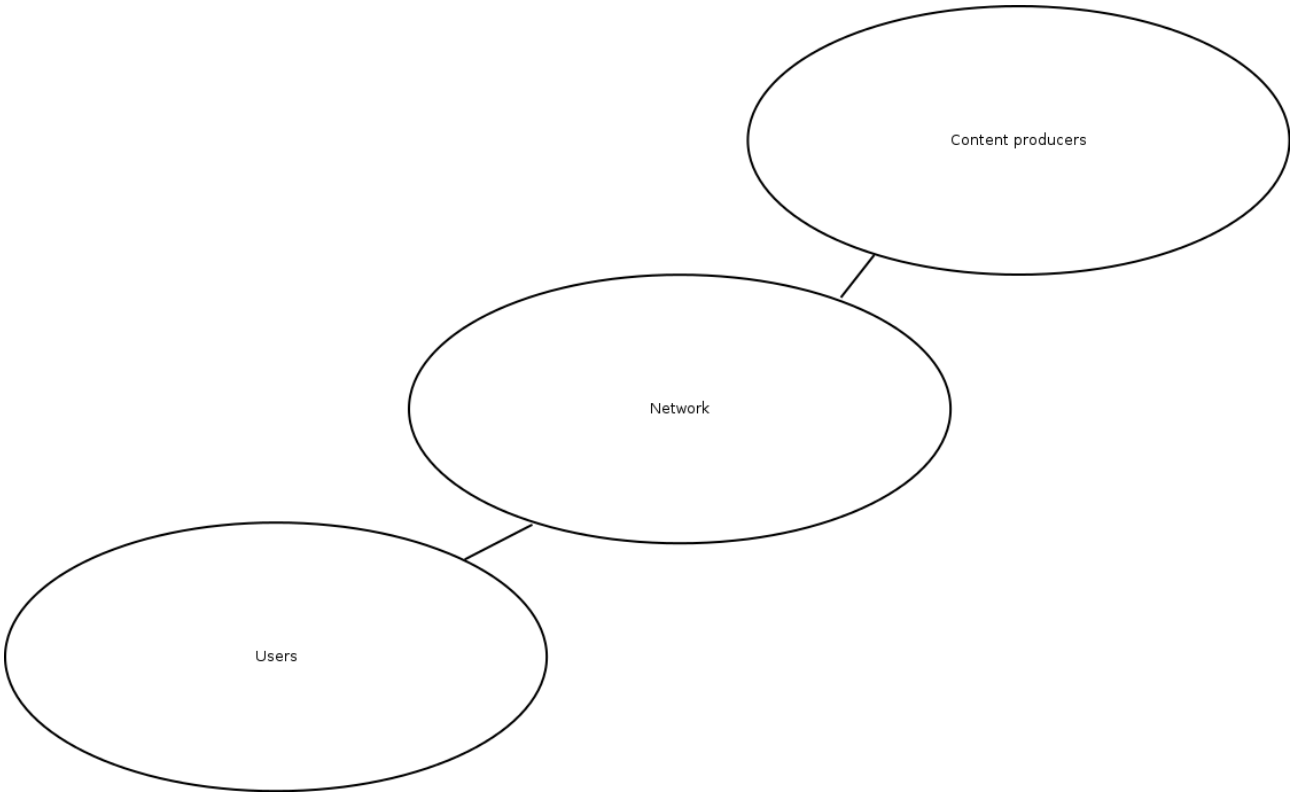


Fig.2.1 Situazione generica di riferimento

Collegati dall'agglomerato di router, switch e fibre ottiche che costituisce lo strato di rete, i mondi di utenti, o consumatori, e produttori di contenuti si scambiano informazioni e dati. La rete permette il passaggio di richieste di contenuti dagli utenti ai produttori ed i pacchetti contenenti i dati in verso opposto.
Chiaramente questa rappresentazione è semplicistica; non tutti gli utenti saranno vicini tra loro, così come tutti i produttori.
Nella realtà è del tutto possibile che alcuni utenti siano in termini di rete, cioè di numero di hop o di ritardo, più vicini ai produttori di contenuti che ad altri utenti, soprattutto vista l'esponenziale crescita dei cosiddetti User Generated Contents, che, di fatto, significa che ciascun utente è di per se un potenziale produttore; questo è valido in maniera ancora maggiore in questi anni, caratterizzati dall'esplosione della diffusione degli smartphone, dispositivi sempre più potenti e mobili per natura, che accrescono enormemente le potenzialità di creazione e condivisione di contenuti originali in ogni momento.
Detto questo, uno schema come quello di figura 1 non è affatto obsoleto, in quanto, tralasciando i casi di comunicazione punto-punto a corto raggio tra due dispositivi, tra un produttore di un contenuto e l'utente che lo richiede, sta sempre uno strato di rete avente l'incarico di consegnare la richiesta al produttore ed il contenuto all'utente.
Chiaramente l'attraversamento dell'infrastruttura di rete è un'operazione che porta all'accumulo di ritardo, spesso considerevole, sulla trasmissione dei messaggi; ciò può essere

causato da numerosi fattori come la velocità di propagazione, la serializzazione, gli overhead introdotti dai protocolli, i tempi di calcolo di router e switch e la congestione dei canali.
Per anni, almeno nell'ambito delle reti di accesso per uso residenziale, il ritardo ed i motivi che lo causavano sono stati trascurati in favore di un interesse più spinto per la velocità della linea, la quale è stata caratterizzata da un aumento costante nel corso del tempo.
Da quando si sono diffusi applicazioni e servizi sempre più orientati verso un accesso in tempo reale alla sorgente del contenuto, ecco che la discussione sul ritardo accumulato durante il trasferimento dei dati e sulle misure per contrastarlo si è animata.
Grazie al sempre più capillare uso della fibra ottica ed all'aumento della capacità di elaborazione degli apparati di rete, oltre che ad un miglioramento dei protocolli[3], grossi passi avanti in tal senso sono stati fatti, tanto da poter permettere la visione di eventi live non solo attraverso una connessione ad internet fissa, ma anche in mobilità.

Tuttavia la soluzione più efficace per combattere il ritardo di trasmissione non risiede tanto nell'aumentare la velocità con cui i pacchetti attraversano la rete, ma nel ridurre il percorso che i pacchetti devono effettuare dalla loro origine fino agli utenti finali.
Questo può essere fatto attraverso la costruzione di cache di dati nelle parti della rete idealmente più vicine agli utenti, proprio allo scopo di rendere il più sottile possibile lo strato di rete attraversato dai dati.
Il concetto di cache non è affatto nuovo o esclusivo del mondo delle reti di comunicazione, basti pensare ad esempio ai microprocessori, dotati di diversi livelli di cache, che permettono di risparmiare diversi viaggi fino alla memoria centrale, decrementando così il tempo di esecuzione delle istruzioni e migliorando di conseguenza l'esperienza d'uso e le prestazioni.
Nella moderna architettura di internet si fa estensivo uso del caching, concretizzato soprattutto nelle CDN o Content Delivery Network, composte da server che fungono da cache distribuite in punti strategici della rete.
Una delle caratteristiche di queste cache è la loro staticità; i contenuti presenti al loro interno sono lì memorizzati in seguito ad accordi commerciali tra il provider CDN e i produttori di contenuti e tale catalogo di contenuti disponibili non viene mutato dalle richieste che la cache riceve dagli utenti (se non indirettamente).
Si noti come questo contrasti con la prospettiva di utilizzo delle cache in ambito CCN; molto più piccole, in numero nettamente maggiore (in linea di principio ogni dispositivo funge da cache per tutti gli altri), distribuite tra gli utenti, ma soprattutto dinamiche e quindi influenzabili direttamente dalla domanda a cui esse vengono esposte.

Questa dinamicità rappresenta allo stesso momento una forza, ma anche una potenziale grossa vulnerabilità.
Una forza per tutti i motivi visti finora; una rete del futuro in cui la richiesta emessa da un certo utente deve attraversare un numero di hop ridotto al minimo ed in cui lo stesso vale per l'oggetto della richiesta è allettante sia dal punto di vista dell'utente, che percepisce un deciso miglioramento dell'esperienza d'uso, sia da quello dell'operatore di rete, che vede le parti metro e core della propria infrastruttura scaricarsi di molto peso.

La considerazione che, però, è sorta e che costituisce l'evento generatore del presente lavoro riguarda la possibilità che questa massiva rete di cache possa essere utilizzata per scopi impropri, non considerati fino a questo momento.
I vettori verso cui si è sviluppata l'indagine presentata in questo lavoro sono molteplici e possono essere riassunti nell'elenco che segue:

- La possibilità di un utilizzo improprio delle cache all'interno di una rete basata sull'architettura CCN e la concretezza di tale ipotesi basata sui risultati di simulazioni di scenari significativi
- Le possibili entità interessate a trarre vantaggio da una tanto capillare diffusione delle cache in rete
- I vantaggi che una di queste entità potrebbe ottenere, il che innesca la discussione su potenziali incentivi e deterrenti oltre che considerazioni sui costi
- Le strategie adottabili per ottenere la massima efficacia rispetto agli obiettivi
- L'esistenza di possibili contromisure

Tutti questi punti verranno trattati nel dettaglio nei capitoli successivi.
Si partirà, nel capitolo 4, intanto con un'analisi generale dei primi tre di questi punti, cercando così di confrontare direttamente lo scenario di riferimento appena descritto con ciò che invece succede in presenza dell'attacco.

Capitolo 3 – Stato dell'arte

3.1 - Introduzione al capitolo e tassonomia degli attacchi

Nel capitolo precedente è stata data una panoramica generale sugli elementi costituenti ed i principi di funzionamento fondanti di un'architettura CCN.
In quella discussione sono stati anche introdotti i pilastri su cui si basa la sicurezza di questo tipo di architettura, soprattutto rispetto a quella basata su IP.
Nel presente capitolo, invece, parleremo nello specifico degli attacchi la cui ideazione ed esecuzione è facilitata dalle caratteristiche di queste reti e lo faremo citando numerosi lavori di ricerca che già sono stati pubblicati, nonostante la relativa gioventù di questo settore specifico e dell'idea di Content Centric Networking in generale, a dimostrazione del notevole interesse che la tematica ha generato nella comunità scientifica.
La totalità degli studi esaminati affronta le problematiche causate da attacchi che possono essere inseriti in due macro categorie:

1. Attacchi di tipo *Denial of Service* (DoS o DDoS): come ben noto, questi attacchi sono oggigiorno estremamente diffusi e pericolosi nelle attuali reti basate su IP.
   Il loro funzionamento prevede il sovraccarico di una risorsa della rete, che può essere un collegamento o un nodo, allo scopo di impedire l'accesso a tale risorsa a chiunque altro.
   Mentre nelle reti IP tale effetto di solito viene ottenuto tramite l'invio di quantità enormi di dati (tipicamente nella forma di pacchetti TCP di tipo SYN), una tecnica definita *flooding*, in direzione della risorsa oggetto dell'attacco, nelle CCN può essere raggiunto o tramite la medesima metodologia, ma l'utilizzo dei messaggi di interest, oppure utilizzando il meccanismo della cosiddetta *cache pollution*, che verrà illustrata meglio in seguito.
2. Attacchi riguardanti la privacy degli utenti: nel capitolo precedente, quando sono state descritte le caratteristiche fondamentali delle CCN, una delle più importanti e potenzialmente di maggior impatto per le prestazioni è sicuramente la pervasività delle cache.
   Tuttavia, per come queste sono intese, si prestano all'aggressione tramite tecniche che consentono ad un potenziale attaccante di estrapolare da esse informazioni importanti e che prendono il nome generico di *cache snooping*.
   Allo stato attuale, infatti, i contenuti che vengono richiesti dagli utenti di una CCN sono immagazzinati all'interno di tutte le cache che attraversano.
   Ciò permette ad un attaccante di costruire un profilo degli utenti collegati ad una cache sotto osservazione in base ai contenuti in essa presenti.
   Risulta superfluo sottolineare l'importanza di questa problematica, soprattutto alla luce delle recenti scoperte riguardanti le capacità e le attitudini che certe agenzie governative possono mettere in gioco.

3.2 - Panoramica sugli studi esaminati

Parte dei lavori qui presentati si concentra su una delle due tipologie di attacco sopra descritte, mentre altri offrono una panoramica più completa, che comprende entrambe le categorie.
La maggior parte di essi contiene anche una parte dedicata a possibili contromisure per il tipo di attacco scelto, a volte illustrate solo a livello di principio, mentre altre anche implementate e provate su scenari simulati.
Dal momento che nell'ultimo capitolo del presente testo verranno trattate le contromisure, molti degli studi presentati in questa sezione saranno lì ripresi per commentare le tattiche proposte e per valutare se, nello specifico, esse possono essere utilizzate per contrastare l'attacco proposto e descritto nel successivo capitolo.
In [15] vengono descritti gli attacchi di *cache pollution* nelle reti IP.
Nonostante non si tratti di CCN, questo studio rimane importante perché definisce una ulteriore categorizzazione di questi tipi di attacco e propone uno schema per contrastare entrambe le sue forme.
Innanzi tutto l'idea alla base di questi attacchi è quella, per il perpetratore, di inserire nelle cache contenuti di proprio gradimento, provocando in questo modo l'eliminazione di altri contenuti legittimamente richiesti e, di conseguenza, la loro consegna potenzialmente lungo l'intero percorso da produttore ad utente.
Questo ovviamente genera un maggiore utilizzo dei collegamenti di quanto necessario, il che ha come estrema conseguenza un *denial of service* per gli altri utilizzatori della stessa tratta.
Nelle CCN il meccanismo con cui tale tecnica viene implementata è quello di consegna standard del dato, ovvero tramite la dinamica degli interest inviati dall'attaccante ad un produttore e la successiva risposta nella forma del contenuto, che viene immagazzinato in tutte le cache che sono attraversate lungo il percorso di consegna.
Si può intuire come questa problematica nelle CCN sia di particolare importanza, data l'ubiquità delle cache nell'architettura.
La tassonomia definita nello studio menzionato suddivide gli attacchi di *cache pollution* in generale in due distinte categorie:

- *Locality disruption*: sotto questa nomenclatura ricadono quegli attacchi in cui un'entità richiede contenuti costantemente nuovi ed impopolari, che quindi devono essere sempre rinvenuti presso i produttori, sono poi memorizzati nelle cache di rete alterandone quindi il contenuto, e mai più richiesti o richiesti solo dopo che sono stati epurati dalla cache e quindi devono essere di nuovo consegnati direttamente dai produttori.
  Come gli autori notano, questo comportamento non è per forza indice di attacco; infatti i *web crawlers* dei motori di ricerca agiscono secondo tale logica.
  Essi dimostrano però che un *web crawler* malevolo è in grado di compromettere le prestazioni di una cache in modo ancora più grave.
- *False locality*: questi attacchi vengono eseguiti richiedendo continuamente gli stessi contenuti, il che genera un effetto di falsa popolarità e diffusione degli stessi.
  Gli autori sostengono che questo sia l'attacco maggiormente pericoloso, in quanto, mentre l'iniziale disseminazione dei contenuti può richiedere parecchie ore, il loro

completo refresh si riesca a completare nel giro di 10 minuti, facilitando per l'attaccante il compito di mantenere la cache nello stato desiderato.

In seguito lo studio si concentra sulla valutazione di 3 meccanismi differenti di rimpiazzo dei contenuti (LRU, LFU e GDSF) e come questi si comportano quando le cache sono affette dagli attacchi sopra descritti; ne risulta che LRU e LFU sono maggiormente resistenti al *locality disruption*, mentre in generale GDSF si comporta meglio in presenza di *false locality* (LRU è più resistente all'aumentare delle richieste dell'attaccante, ma ha prestazioni peggiori di GDSF se a variare è la capacità del canale di accesso).
Infine viene proposto un meccanismo, basato su *bloom filters*, per il rilevamento di entrambi gli attacchi, sia quando sono eseguiti separatamente che allo stesso momento.
Nei test condotti viene provata l'efficacia di tale metodo, che però richiede una precisa identificazione degli utenti per ottenere misure accurate del loro comportamento ed un periodo di alcune ore per il rilevamento e la successiva reazione.

In [6] gli autori si dedicano agli attacchi di tipo *cache pollution* nelle CCN; l'obiettivo dell'articolo è poi quello di illustrare, implementare e testare un meccanismo proattivo per la loro prevenzione, che rimane tutt'oggi l'unico tentativo in tal senso a conoscenza dell'autore.
Questa tecnica, denominata *Cacheshield*, si basa su un semplice concetto; la memorizzazione dei contenuti nelle cache non deve avvenire deterministicamente, ma statisticamente, con una probabilità maggiore se il contenuto ha avuto un maggior numero di richieste.
In particolare viene impiegata una funzione di *shielding* che calcola la soglia che deve essere superata in un test probabilistico perché un contenuto venga inserito in cache e uno dei due parametri della funzione proposta è proprio il numero di richieste giunte per tale contenuto.
I risultati proposti nello studio dimostrano l'efficacia di questo metodo nei particolari casi che gli autori hanno deciso di prendere in esame.

[7] tratta il tema della *cache pollution* nelle CCN, compiendo sperimentazioni sulla loro efficacia in topologie estese e sulla reale capacità del metodo di prevenzione descritto in [6].
In merito, gli autori conducono test che prevedono una fase iniziale con richieste per contenuti emesse da utenti legittimi ed una fase successiva in cui l'attaccante comincia la sua azione seguendo il pattern del *false locality*.
Queste simulazioni vengono eseguite sia nella topologia presentata in [6], che in una più estesa e complessa; dal momento che i risultati sono simili per entrambe, Conti et al. concludono che il loro lavoro fornisce ulteriori prove della pericolosità dell'attacco in oggetto anche in topologie estese e complesse.
Inoltre mostrano la facilità con cui la tecnica proattiva proposta in [6] può essere resa completamente inefficace dall'attaccante; egli può semplicemente aumentare la frequenza della richiesta di un certo contenuto per aumentare la probabilità che questo sia memorizzato e nel fare questo non deve necessariamente aumentare la frequenza totale, ma abbassare il numero di contenuti che richiede.
Questa pratica effettivamente mette in scacco il sistema, che favorisce quei contenuti che sono più soggetto di richiesta.

Infine essi propongono un nuovo meccanismo di rilevamento di questi attacchi, che secondo le loro stime è sufficientemente leggero da poter essere implementato ed efficientemente eseguito sui router.
Il metodo si può ascrivere alla categoria dei sistemi di rilevamento di tipo *anomaly-based*; i router devono eseguire rapidamente una valutazione di certe metriche statistiche dei contenuti che vengono richiesti e successivamente consegnati attraverso di essi in condizioni normali ed inoltre calcolano le soglie di varianza che, se superate, fanno scattare l'allarme.
I test condotti provano l'efficacia della tecnica per entrambe le topologie utilizzate.

In [16] gli autori propongono un sistema di rilevamento dei contenuti caratterizzati da una firma digitale che sia, per un qualsiasi motivo, non corretta ma che possono lo stesso essere presenti in qualche cache all'interno della rete, provocandone quindi l'inquinamento.
L'assunto fondamentale da cui essi procedono è che sia infattibile verificare la firma digitale di ogni contenuto in transito per ogni router, perciò il loro sistema prevede la collaborazione degli utenti, che devono, secondo il paradigma CCN, obbligatoriamente verificare la firma di ciascun contenuto che ricevono.
L'idea presentata è perciò quella di instaurare un meccanismo con cui gli utenti possono riportare un feedback ai router su quei contenuti che non hanno superato il controllo della firma, facendo uso di nuovi interest e del loro campo *exclude*.
I router costruiscono una lista ordinata dei contenuti caratterizzata da un indice della loro bontà che viene degradato alla ricezione di questi nuovi messaggi.
Questa tecnica è poi testata in 3 differenti topologie di rete, di differente estensione e complessità ed i risultati ottenuti provano l'efficacia del metodo, che riesce a portare al 100% in breve tempo la percentuale di contenuti con firma digitale corretta consegnati agli utenti.

[17] rappresenta uno dei primi studi riguardanti il tema degli attacchi di tipo *Denial of Service* nell'ambito delle CCN ed in particolare gli autori illustrano le attuali tecniche impiegate nelle reti IP e sostengono la loro inefficacia nelle CCN.
Viene quindi descritta la distinzione fra le due tecniche che possono causare DoS nelle CCN, vale a dire *interest flooding* e *cache pollution*.
Per quanto concerne l'*interest flooding*, Gasti et al. suggeriscono l'utilizzo di statistiche sul numero degli interest non soddisfatti per identificare potenziali attacchi e su un limite imposto alla ricezione di interest da una certa interfaccia e per un certo *namespace* (ad alto livello è il corrispondente CCN del concetto di indirizzo IP) per la loro mitigazione.
Per ciò che invece riguarda *cache pollution*, come in [16] gli approcci proposti partono dalle difficoltà estreme insite nelle verifica delle firme dei contenuti ed offrono diversi spunti per ricerche più dettagliate; l'idea di base dietro ciascuno di questi è quella della suddivisione del carico della verifica fra molteplici entità e dell'inserimento nel sistema anche degli utenti, tramite un opportuno sistema di feedback.

[18] si occupa in maniera specifica degli attacchi basati su *interest flooding*, in particolare indirizzati a saturare la PIT dei router.
Gli autori propongono sia misure di prevenzione, che di diagnosi e reazione agli attacchi.

Le prime consistono in un controllo del flusso di interest verso una certa interfaccia basato sull'utilizzo della PIT o sull'utilizzo di messaggi di errore in risposta ad interest non soddisfatti che abbiano la capacità di liberare le PIT dei router che li ricevono.
Viene sottolineato però che queste tecniche proattive possono essere esse stesse fonti di vulnerabilità, perciò non vengono ulteriormente sviluppate.
Il paper invece si concentra sulla proposta di un sistema di rilevamento, chiamato *Poseidon*, e di una successiva strategia di reazione.
Tale sistema, implementato sui router, lavora misurando, per ciascuna interfaccia, gli interest soddisfatti in un dato tempo e lo spazio che essi occupano nella PIT nel medesimo intervallo.
Nel caso in cui i valori misurati siano superiori a certe soglie, viene fatta scattare la fase reattiva, la quale prevede sia una limitazione degli interest sull'interfaccia identificata come punto di ingresso degli interest malevoli, che l'invio di messaggi di allerta da parte del router che rileva l'attacco ai nodi di rete a valle.
Questi messaggi di allerta sono intesi come particolari contenuti, perciò il protocollo CCN non viene modificato.
Infine gli autori mostrano i risultati che il loro sistema riesce a raggiungere in due differenti ma estese topologie.

Un altro approccio al problema dell'*interest flooding* viene illustrato in [19].
Le tecniche descritte in [18] e [19] sono accomunate dal fatto che i router CCN mantengono uno stato delle richieste che inoltrano, nella forma della PIT, al contrario di quelli IP.
Questo permette loro di calcolare varie statistiche, tra le quali, come visto in precedenza, il numero di interest soddisfatti provenienti da una certa interfaccia.
Oltre a ciò i due studi condividono anche una modalità di reazione, vale a dire la limitazione imposta all'instradamento di futuri interest ricevuti dall'interfaccia incriminata.
Dove però [19] si differenzia è nel fatto che questa riduzione viene fatta dipendere alla metrica misurata in maniera proporzionale.
Gli autori poi conducono simulazioni volte a dimostrare l'efficacia delle soluzioni da loro proposte.

[20] tratta le problematiche riguardanti la privacy degli utenti di CCN, mostrando come un utente malevolo possa ottenere informazioni sui contenuti presenti in una cache osservando i tempi di risposta a richieste nel *namespace* desiderato.
Gli autori propongono poi una serie di misure per mitigare il problema, partendo da quelle che maggiormente si focalizzano sulla protezione della privacy, penalizzando però l'esperienza d'uso, e giungendo ad altre che offrono un migliore compromesso fra queste due necessità.
Tra questi, sono degni di nota alcuni schemi che introducono casualità nei tempi di risposta della cache, che tuttavia richiedono che i contenuti siano statisticamente indipendenti tra loro.
Sviluppano inoltre un modello formale allo scopo di stabilire quanto un dato algoritmo di caching sia orientato alla privacy o all'esperienza di utilizzo.
In [14] l'autore propone una panoramica completa sulla sicurezza delle CCN, elencando molteplici tipologie di attacco.

Tra queste vengono anche introdotte idee per nuovi possibili metodi di aggressione che vertono sulle debolezze insite nell'architettura CCN, perché direttamente collegate ad alcuni dei principi fondamentali.
Una di queste, descritta nella sezione 3.4.1 (Pagg. 17-18), risulta essere simile come concetto e scopo a quella presentata e testata nel presente lavoro.
La versione descritta in [14], classificabile come *false locality*, prevede che un attaccante richieda continuamente un certo contenuto per mantenerlo memorizzato nelle cache, anche dopo che il produttore originario ha smesso di distribuirlo.
Pur simile come idea, ciò che viene qui proposto differisce perché identifica chiaramente parti specifiche dell'architettura CCN in cui i contenuti possono essere immagazzinati con maggiore efficacia.
Inoltre, pur menzionandolo, l'autore non approfondisce l'indagine su questo concetto, si limita a proporre qualche generica contromisura e stabilisce il grado di rischio come basso, poiché l'attacco richiederebbe grande complessità, contromisure sarebbero disponibili ed un alta frequenza delle richieste sarebbe, potenzialmente, necessaria.
Le conclusioni scaturite dagli esperimenti svolti nel presente studio recano prove che supportano tesi in contrasto con quelle appena esposte, anche se rimane vero che efficaci contromisure per attacchi di *false locality* esistono.
Il focus di [14] è, come [20], la privacy degli utenti delle CCN e l'autore svolge un approfondimento su 3 particolari casistiche, ovvero l'ottenimento di una copia dei contenuti di una certa cache, l'analisi dell'accesso ad un determinato contenuto e la completa clonazione di una conversazione fra due utenti.
Oltre a questo vengono proposte diverse contromisure ed un algoritmo per il calcolo preciso del *characteristic time* di una cache, definito come il tempo trascorso fra l'immissione di un contenuto in cache e la sua epurazione.
Infine viene mostrato che, tramite la conduzione di estensive simulazioni, questo algoritmo è in grado di fornire risultati precisi con un irrisorio volume di traffico da parte dell'attaccante, anche se il tempo di esecuzione cresce al crescere del *characteristic time* stesso.

3.4 - Osservazioni finali

In questo capitolo sono stati presentati molti dei lavori più significativi nel panorama della sicurezza nell'ambito delle CCN.
Come si può intuire dal loro numero e dagli anni di pubblicazione, questo campo è in grande fermento, data la relativa giovinezza della formalizzazione del concetto di CCN.
Nonostante questo si possono già identificare dei pattern ricorrenti ed è possibile classificare ciascuna di queste pubblicazioni come relativa ad una delle categorie descritte nella parte iniziale del capitolo.
Pur potendo ascrivere anche il presente testo ad una di quelle tipologie, nello specifico degli attacchi di *false locality*, di certo l'obiettivo dell'attacco qui proposto non è il DoS, il che pone già un primo fattore di differenziazione rispetto agli studi citati; inoltre nessuno di essi tratta gli effetti di *cache pollution* quando le cache interessate sono quelle degli utenti, ma vengono sempre considerate solo quelle dei router.

Infine, pur essendo vero che l'idea di base è simile a quanto descritto in [14], rispetto a quel lavoro, l'attacco descritto viene approfondito e studiato e le conclusioni emerse risultano contrastanti con quelle espresse in [14].

Capitolo 4 – Descrizione dello scenario di attacco simulato

In questo capitolo, nella sezione 4.1, viene definito il paradigma di attacco proposto, i suoi possibili perpetratori e gli scopi che essi vogliono perseguire.
Nella sezione 4.2 vengono successivamente caratterizzati limiti dell'attacco, vincoli e compromessi imposti o scelti dall'attaccante.
Nella sezione 4.3 verrà descritta la specifica topologia utilizzata per le simulazioni, così come i ruoli e le ipotesi riguardanti ognuna delle entità che ne fanno parte.
Infine, nella sezione 4.4, saranno illustrate le metriche, le variabili ed i valori o intervalli di valori scelti per ciascuna di esse; ognuna di queste scelte sarà discussa e giustificata approfonditamente.

4.1 - Definizione dello scenario generale di attacco

In precedenza è stato scritto che la presenza di una cache all'interno di ogni dispositivo collegato ad una CCN crea un vantaggio sia per l'utente finale che per l'operatore della rete, che vede la propria rete scaricarsi e quindi decongestionarsi.
Pur rimanendo tutto ciò valido, una delle entità mostrate nello schema presentato nel capitolo 2 non è stata menzionata, pur godendo di innegabili vantaggi in questa configurazione: il produttore di contenuti.
In figura 4.1 si può visualizzare un esempio di processo di "disseminazione" dei contenuti in rete; esso prevede in prima battuta la pubblicazione dei contenuti da parte del produttore, i quali vengono poi richiesti dai consumatori.

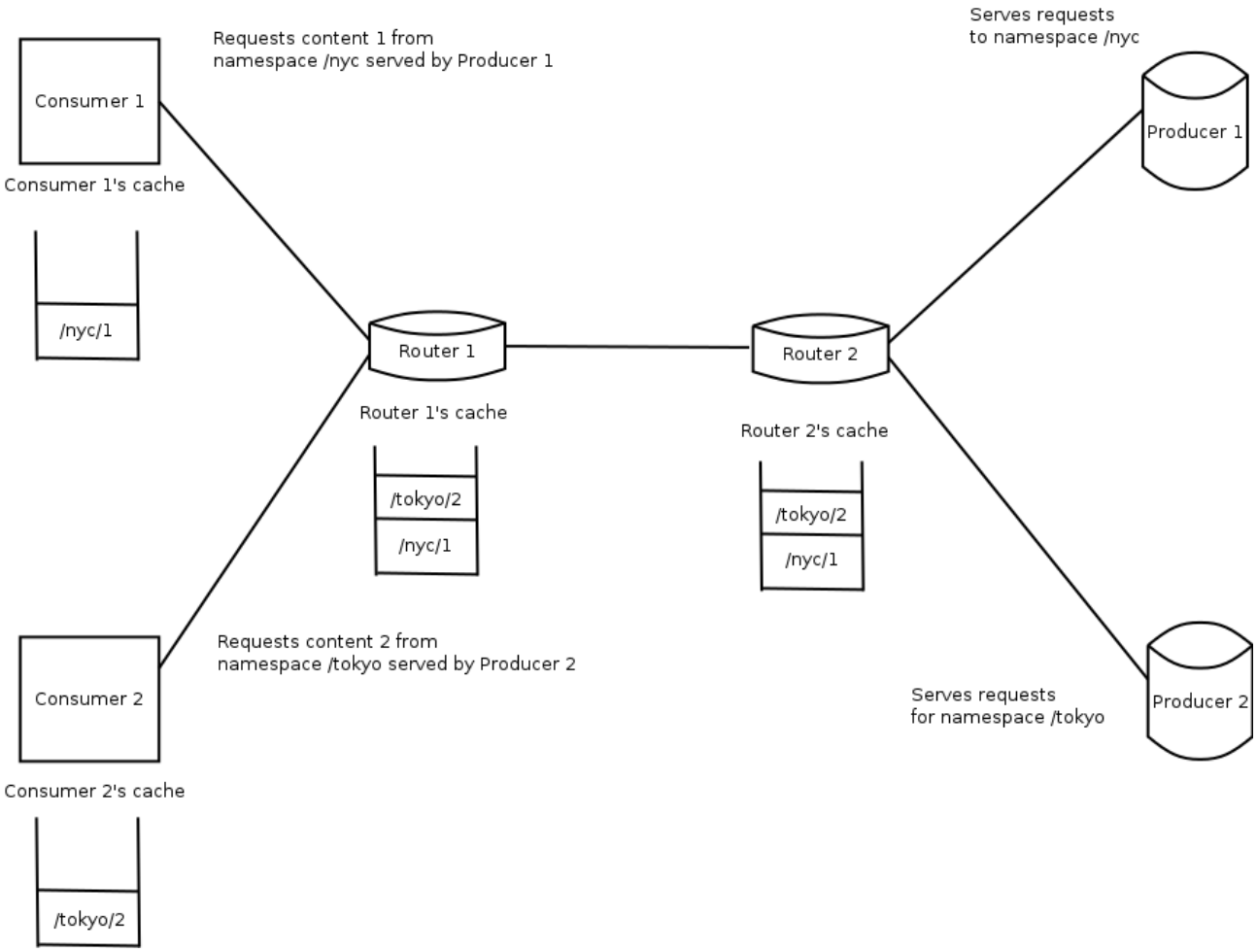


Fig.4.1 – Disseminazione di contenuti in una CCN

Man mano che questi contenuti viaggiano all'interno della rete, essi sono memorizzati in ciascuna delle cache che attraversano, idealmente presente in ogni nodo, fino a quando non raggiungono l'utente o gli utenti che avevano emesso la richiesta, completando così il processo.
In questo modo, la richiesta del prossimo utente per quello stesso contenuto non avrà necessità di pervenire al produttore, ma si fermerà al primo nodo che ha disponibile quel contenuto, garantendo al produttore alcuni importanti vantaggi:

- Il tempo di permanenza in rete del contenuto è stato appena prolungato, in quanto esso è disponibile in un nodo presso cui non lo era in precedenza ed inoltre è balzato in testa alla graduatoria nella cache che ha servito il contenuto, supponendo che questa sia organizzata secondo la policy LRU
- Non dovendo trasmettere ulteriormente il contenuto, il produttore ha un risparmio, poiché deve far trasportare meno dei propri dati al suo operatore di rete
- L'utente che sta usufruendo del servizio o del contenuto offerto dal produttore sarà più soddisfatto perché riceve il contenuto con un ritardo molto basso; ciò quindi, oltre che essere un fattore benefico per il cliente, lo è anche per il produttore

La dinamica che si innesca è quindi quella di un "avvicinamento" dei contenuti al cliente, ma non di tutti in maniera generalizzata, bensì di quelli più popolari tra i clienti stessi.
Infatti, qualunque sia la politica di rimpiazzo (replacement policy) adottata dalle varie cache della CCN, questa regola ha sempre valore; inevitabilmente ciò che è molto richiesto dai clienti sarà più vicino ad essi ed avrà tempi di permanenza in rete superiori, mentre ciò che non è popolare finirà con l'essere relegato sempre più verso l'alto della catena gerarchica, perciò lontano dai clienti, memorizzato e servito solo dai router più in prossimità del produttore, se non dal produttore stesso.
È questa la chiave del ragionamento; se solo i contenuti più popolari permettono ai rispettivi produttori di godere dei vantaggi citati in precedenza, cosa possono fare i produttori di contenuti meno popolari per godere degli stessi vantaggi?
Se tali produttori non sono intenzionati a produrre contenuto maggiormente popolare e sono tantomeno dotati di senso etico, essi hanno la possibilità di influenzare la domanda, promuovendo così i loro contenuti e, in ultima analisi, di beneficiare in parte maggiore rispetto a quanto loro spetterebbe dell'architettura di rete.

Pertanto è stata appena descritta una tipologia di entità che potrebbe essere molto interessata a sfruttare le opzioni presentate da questa molteplicità di cache presenti all'interno della rete, in particolare quelle più vicine agli utenti che tale entità vuole servire.
Questa non è l'unica però; un'altra tipologia di entità potrebbe essere interessata a disseminare un particolare tipo di contenuto all'interno del dispositivo o dei dispositivi di uno specifico utente.
Chiaramente questo è un tipo di attacco più mirato, ma non per questo meno pericoloso ed abilitato potenzialmente dal solo uso della cache da parte della vittima.
D'ora in avanti sarà preso in esame il caso di un produttore che può essere definito malevolo, poiché è l'entità che vuole avvantaggiarsi indebitamente, che fa uso di un attaccante, il cui ruolo sarà quello di pianificare una strategia di attacco per poi metterla in pratica.

Fino a questo momento si è affrontata la discussione sul chi e sul perché avrebbe interesse a sviluppare un attacco del tipo descritto nel presente documento, ma non è stato ancora menzionato il come, ovvero le tecniche pratiche con cui l'attaccante può influenzare a suo favore la domanda dei contenuti.
Per avere successo, all'attaccante sarà sufficiente compromettere una serie di nodi che emetteranno poi le richieste verso i contenuti serviti dal produttore malevolo.
L'attaccante in sostanza costruisce o acquista una botnet che generi un numero di richieste tali per cui il produttore malevolo possa godere dei tre benefici espressi in precedenza.
Perciò la premessa fondamentale perché questo tipo di attacco, che è ascrivibile alla categoria *false locality*, abbia successo è che il perpetratore sia in grado di compromettere un insieme di dispositivi appartenenti al gruppo degli utenti.
Quello che l'attaccante invece non è in grado di fare è compromettere uno qualsiasi dei nodi presenti all'interno della rete; non è possibile quindi modificare la programmazione di router e switch in alcun modo per l'attaccante.
A questo punto i nodi compromessi non devono fare altro che richiedere al produttore malevolo quei contenuti che esso è interessato a disseminare in rete.

Tali contenuti quindi verranno memorizzati all'interno delle cache di questi nodi, divenendo disponibili maggiormente in prossimità rispetto agli altri utenti della rete.
Come si può notare, l'attacco è concettualmente molto semplice, dal momento che usa gli strumenti che l'architettura CCN mette a disposizione in una maniera che può molto facilmente passare inosservata, perché il comportamento anomalo degli utenti compromessi può essere perfettamente spiegato come un cambiamento di interesse da parte di essi per i contenuti.
Ciò influisce sensibilmente anche sulla sua pericolosità, in quanto risulta più complicata la sua rilevazione, oltre che il suo disinnesco.

4.2 - I trade-offs dell'attaccante

La panoramica vista finora appare vantaggiosa per l'attaccante e complicata per l'esperto di sicurezza che deve contrastare l'attacco, o anche solo identificarlo, tuttavia ci sono degli aspetti che l'attaccante deve tenere presente e che possono rendere l'attacco poco efficace, oltre che un importante fattore di limitazione, sicuramente valido al giorno d'oggi.

4.2.1 - Capacità di canale in upload

Ciò a cui viene fatto riferimento è la capacità in uplink delle linee della rete di accesso, ancora troppo bassa per supportare una visione di rete in cui potenzialmente lo smartphone di un utente possa effettuare l'upload di un video in alta definizione.
Pur concedendo che nel momento storico in cui viene effettuata la stesura del presente documento, il limite sopra menzionato sia sicuramente da non sottovalutare, si sottolinea come le reti di nuova generazione (fibra ottica fino all'utente per la rete fissa, LTE ed LTE Advanced per quella mobile) siano focalizzate dal loro concepimento sulla capacità in upload in maniera molto maggiore rispetto a quanto fatto con le tecnologie di generazione precedente (ADSL e UMTS).
Proprio per questo trend di aumento dell'attenzione da parte dell'industria delle telecomunicazioni sull'ampliamento della banda disponibile all'utente per fungere da produttore dei contenuti, si ritiene che tale limite verrà superato col tempo e con la diffusione sempre più ampia delle tecnologie di nuova generazione.
Va inoltre sottolineato che l'ostacolo in questione non riguarda il solo attacco, ma più in generale l'architettura CCN, che per funzionare come da progetto necessita per forza di un ampio canale dagli utenti alla rete.

4.2.2 - Rilevabilità dell'attacco

Per creare un vantaggio consistente al produttore malevolo, l'attaccante ha bisogno di compromettere un insieme di nodi appartenenti agli utenti.
Dal momento che la cache a disposizione di ciascuno ha una dimensione limitata, i contenuti richiesti in maniera fraudolenta dalla vittima entreranno in competizione con quelli richiesti legittimamente da essa, escludendone pertanto almeno una parte.

Questo effetto collaterale è indesiderato sia per la vittima che per l'attaccante, che vuole evitare di degradare in maniera evidente l'esperienza di utilizzo degli utenti.
Questo studio non vuole rappresentare un'indagine quantitativa riguardante i fattori che più impattano sull'esperienza d'uso degli utenti di una CCN, ed in particolare sul suo degrado in termini di percezione in corrispondenza di un peggioramento di tali fattori; tuttavia ci sono due di questi fattori che si ipotizza possano fungere da buoni indicatori della qualità di esperienza d'uso e sui i quali si è cercato di introdurre il minimo grado di perturbazione possibile.
Questi due fattori sono l'*hit ratio* che il nodo compromesso è in grado di ottenere dalla propria cache e dalle altre presenti in rete ed il ritardo misurato dal momento in cui parte una richiesta legittima ed il momento in cui il contenuto viene ricevuto.
Non è ancora chiaro in che misura l'andamento di questi fattori sia legato all'esperienza d'uso, ma l'ipotesi che un loro significativo degrado risulti in un peggioramento di essa appare del tutto ragionevole.
Per questo motivo il bilanciamento accurato da parte dell'attaccante del vantaggio creato al produttore malevolo con il minor danno possibile causato ai nodi compromessi rappresenta l'obiettivo primario.

4.2.3 - Competitività di un nodo compromesso

Oltre ai due già illustrati, un altro fattore che può limitare l'efficacia dell'attacco è il grado di competitività e quindi di attività che la vittima produce.
Questa metrica è dipendente dalla dimensione della cache, dalla quantità di traffico legittimamente generato e dalla cardinalità dell'insieme dei contenuti illecitamente richiesti.
La dimensione della cache chiaramente influisce poiché a parità di volume di traffico legittimo e della grandezza dell'insieme dei contenuti illeciti richiesti, una cache più piccola porterà meno vantaggi per l'attaccante, perché può contenere fisicamente meno oggetti.
La quantità di traffico legittimo incide perché una maggiore frequenza delle richieste può significare che più frequentemente il tempo di permanenza in cache dei contenuti richiesti viene aumentato o che l'aggiunta di nuovi contenuti alla cache avviene più spesso, degradandone l'utilità per l'attaccante.
L'ultimo aspetto va in realtà in favore dell'attaccante ed è qualcosa che è possibile controllare, cioè quanti contenuti differenti del catalogo del produttore malevolo la vittima viene costretta a richiedere.
Considerando che l'attaccante ha la necessità di tenere un basso profilo, questa rappresenta un'arma importante poiché, fissata la frequenza delle richieste illecite, egli avrà maggiore successo a trattenere i contenuti in cache se il loro numero è basso, perché ogni contenuto riceverà un numero maggiore di richieste.
Questi fattori rappresentano alcuni dei parametri utilizzati nelle simulazioni, pertanto saranno strumento di indagini quantitative nelle sezioni successive.

4.2.4 - Numero dei nodi compromessi

Tra gli scopi del presente documento non c'è quello di definire un numero ottimo di nodi per i quali l'attacco raggiunge il massimo dell'efficacia, perciò si esprimeranno qui alcune considerazioni qualitative, a cui si aggiungeranno osservazioni basate sui dati forniti dai risultati degli esperimenti condotti.
Il numero di nodi certamente influenza l'efficacia dell'attacco ed idealmente più tale numero è alto, più sono i vantaggi per l'attaccante, perché esso ha così modo di disseminare i contenuti in maniera più ampia, potenzialmente aumentandone sia il tempo di permanenza in rete, che la disponibilità.
Allo stesso tempo non può essere evitata una considerazione dei costi che l'aumento dei nodi ha; in primo luogo l'atto stesso della compromissione ha un costo in se, monetario oppure di tempo.
In secondo luogo non va dimenticato che i nodi compromessi devono ricevere i contenuti da qualche parte e, soprattutto in certe condizioni, questa sorgente può benissimo essere il produttore malevolo, il quale, nel tentativo di scaricare sulle vittime le proprie consegne si può vedere costretto a recapitare i contenuti ad un numero maggiore di nodi.
Nel capitolo successivo, in particolare quando si valuteranno le diverse strategie a disposizione dell'attaccante per evidenziare quella di maggior successo, verranno esaminati gli effetti ed i benefici che un nodo in più in una rete di vittime può portare, ma anche gli aspetti meno positivi, come quello appena citato.

4.2.5 - Riassunto della sezione

In questa sezione si è illustrato il contesto generale di riferimento, affrontando il tema del miglioramento delle prestazioni e dell'esperienza di utilizzo per gli utenti delle CCN, soprattutto puntualizzando l'importanza che il caching riveste nel perseguimento di questi obiettivi, al di là dei progressi fatti e tutt'ora in corso sul fronte dell'abbattimento del ritardo tramite il miglioramento delle connessioni e dei protocolli.
Si è visto poi come la pervasività delle cache professata nell'ambito CCN possa rappresentare un'opportunità per entità interessate ad ottenere significativi vantaggi modificando artificialmente a loro favore la domanda per i contenuti.
Sono stati presentati due tipi di tali entità e le loro motivazioni sono state chiaramente espresse in una maniera che si ritiene possa aver fatto comprendere la loro concretezza ed appetibilità.
In seguito sono state esposte le limitazioni principali dell'attacco, oltre che i compromessi che l'attaccante vuole attuare (bilanciamento tra efficacia e degrado dell'esperienza d'uso) o che è costretto ad accettare (maggior numero di nodi compromessi equivale anche a maggiori richieste da soddisfare per il produttore malevolo).
Come parole finali a chiusura della sezione si vuole affermare nuovamente e riassumere la catena logica fondamentale che si è utilizzata: per incrementare le prestazioni degli utenti occorre diminuire il ritardo ed aumentare l'*hit ratio*, ma siccome questi effetti si realizzano solo per i contenuti più popolari, un produttore malevolo potrebbe volere influire sulla domanda per rendere popolari i propri contenuti.

Per fare ciò si avvale di un attacco che, attraverso la compromissione dei nodi sotto il controllo degli utenti, permette ai propri contenuti di rimanere per lungo tempo in rete e di godere di un'ampia diffusione.
Nella prossima sezione verrà descritto lo scenario così come è stato simulato, insieme ai parametri ed alle metriche utilizzate.

4.3 - Descrizione dello scenario simulato

In questa sezione verranno definite la topologia di rete, il funzionamento di base, le assunzioni generali, le descrizioni dei ruoli ricoperti da ciascuna delle entità, comprendenti di potenzialità e limitazioni ed infine viene data una breve panoramica sullo strumento utilizzato per le simulazioni, il software open source ndnSIM.
Si procede ora quindi col mostrare la topologia di rete, visibile nella figura sottostante (Fig. 4.1):

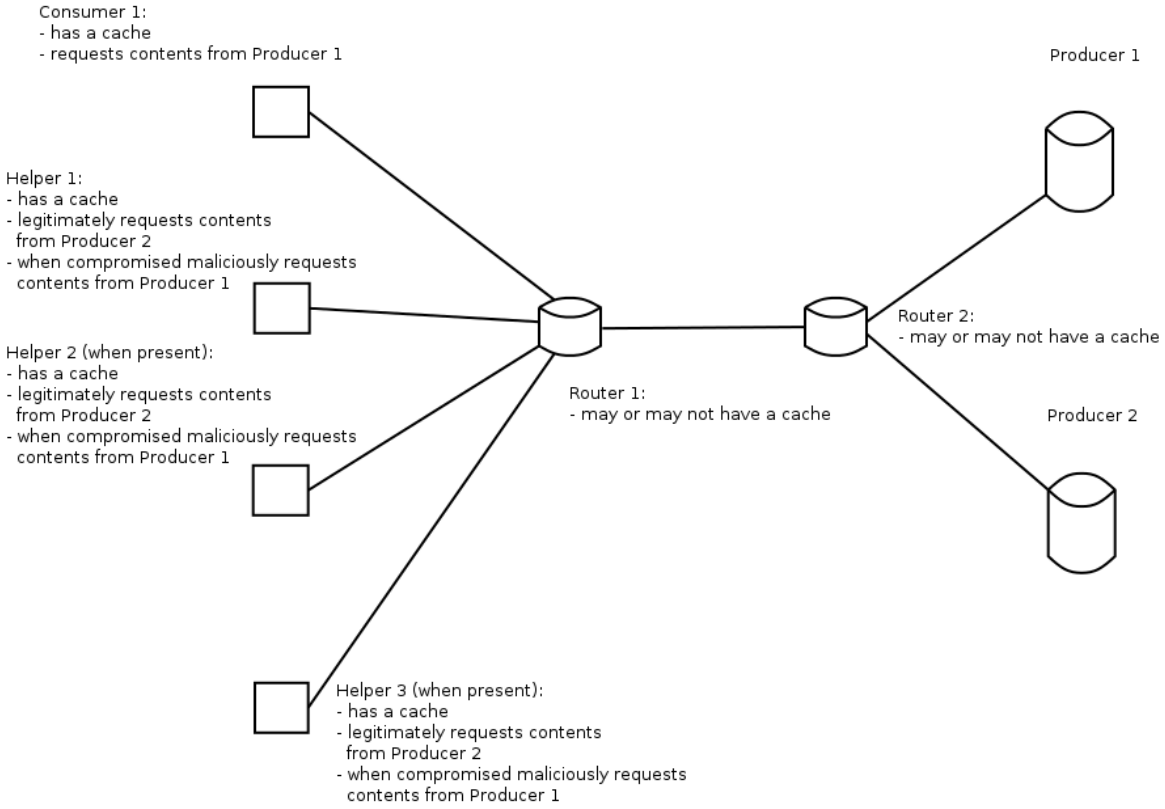


Fig.4.2 Scenario specifico di riferimento

La rete è fondamentalmente suddivisa in due domini, ciascuno facente capo ad uno dei router; la parte sulla sinistra dello schema è l'area che racchiude gli utenti, coloro che inviano messaggi di interest, mentre quella sulla destra comprende i produttori, che ricevono i messaggi di interest a cui rispondono coi contenuti.
Questa disposizione è stata scelta per evidenziare al massimo i vantaggi dell'utilizzo delle cache, oltre che per incarnare in maniera specifica l'ambito generale descritto nel precedente capitolo; dal momento che l'appartenenza alla stessa area indica un minore ritardo nel transito di pacchetti, se ci fosse stato un utente nella stessa area dei produttori, tale utente avrebbe potuto ricevere i contenuti direttamente da essi, risultando quindi di scarsa importanza per lo studio condotto.

I componenti fissi della rete sono i nodi utente Consumer 1 e Helper 1, i due router ed i due produttori; la quantità di nodi utente presenti è un parametro simulativo, pertanto sono indicati nella topologia due ulteriori nodi utente (Helper 2 e 3), ma questi sono presenti solo in alcuni dei test simulativi.
In ogni caso il loro ruolo è esattamente lo stesso di Helper 1, pertanto ne condividono descrizione, proprietà, limitazioni e potenzialità.

4.3.1 - Caratteristiche comuni dei nodi utente

I nodi utente sono stati pensati per approssimare il più fedelmente possibile generici utenti della rete internet, pertanto gli aspetti che li accomunano sono molteplici.
Innanzi tutto la curva della domanda, ovvero la funzione che determina la probabilità che un utente richieda un particolare contenuto in un insieme, ha un andamento dettato dalla legge di Zipf.
È ormai ampiamente diffuso in letteratura l'utilizzo di distribuzioni di tipo Zipf come approssimazioni di flussi di traffico rappresentanti richieste di contenuti, poiché numerosi studi ne hanno stabilito la validità[1][2].
Ciò significa che dato un insieme di contenuti, ordinato per popolarità decrescente, la probabilità che il contenuto k-mo venga scelto è dato dalla relazione $k^{-\alpha}$, dove α è generalmente compreso tra 0 e 1.
Un'altra importante assunzione che viene fatta è che, per tutta la durata della simulazione, gli utenti richiedano solo contenuti all'interno di insiemi predeterminati e che quindi la domanda non possa variare in maniera dinamica.
Si tratta sicuramente di un'ipotesi piuttosto forte e che è possibile trovare limitante per lo studio, ma quella di definire un insieme e formulare le richieste solo per tale insieme è pratica diffusa in letteratura [5][6], mentre è più raro assistere a studi che tengano in considerazione tale caratteristica [7].
Si vuole sottolineare però che la durata di ciascun test svolto per questo studio è limitata ad un intervallo temporale corto (che verrà definito successivamente quando saranno mostrati tutti i parametri simulativi), pertanto è più plausibile che la domanda rimanga stazionaria nel periodo di osservazione.
In secondo luogo tutti i nodi utente sono dotati di una cache, poiché questo costituisce uno degli assunti fondamentali del paradigma CCN.
Ciascuno dei nodi utente emette richieste, nella forma di messaggi interest, verso uno specifico produttore ad una frequenza che costituisce uno dei parametri di test, con intervalli tra le richieste distribuiti in maniera esponenziale negativa.
Supponiamo poi che nessuno di essi effettui operazioni anomale sulla rete (ad esempio un interest flooding); in altre parole ognuno di essi si limita ad utilizzare ordinariamente la rete.
Completata la definizione generica di un nodo utente, si passa ad esaminare le poche differenze tra le due tipologie.

4.3.2 - Caratteristiche differenti tra Consumer 1 e gli helper

Ciò che distingue Consumer 1 da Helper è il fatto che il primo è il beneficiario inconsapevole dell'attacco, mentre il secondo è inconsapevolmente vittima di esso.
Le altre differenze riguardano frequenza delle richieste e dimensione della cache; mentre per Helper queste grandezze sono variabili nelle simulazioni, per Consumer 1 vengono fissate in un modo e secondo motivazioni che verranno descritte in seguito, quando si parlerà in maniera specifica della scelta dei parametri.

4.3.3 - Helper

Questi nodi, che fungono da rappresentanti per l'attività di consumatori di contenuti della CCN, sono quelli che vengono compromessi quando l'attacco viene eseguito.
La compromissione ha luogo attraverso l'installazione sul nodo in questione di una applicazione controllata dall'attaccante, il quale può quindi variare alcuni parametri, come frequenza delle richieste e loro distribuzione statistica, oltre che definire quali e quanti contenuti far illecitamente richiedere ai nodi compromessi.
Questo da un buon grado di libertà all'attaccante, anche se importanti parametri rimangono fuori dalla sua portata; non è infatti in grado di agire sulla quantità di traffico legittimamente inviato dalle altre applicazioni presenti sul nodo, così come non può aumentare la cache a sua disposizione.

Inoltre, per motivi di semplicità di analisi e per rispettare il limite sulla rilevabilità dell'attacco descritto nella precedente sezione, si è scelto di fissare due dei parametri citati poco sopra, vale a dire frequenza delle richieste illecite e loro distribuzione statistica.
La frequenza è tenuta bassa par far sì che il traffico generato abbia maggiore possibilità di passare inosservato e la distribuzione statistica in realtà non c'è; infatti vengono richiesti tutti i contenuti sequenzialmente, in maniera, quindi, deterministica.
Non è stata scelta alcuna particolare distribuzione statistica perché l'intento dell'attaccante è quello di disseminare tutti i contenuti allo stesso modo.
È chiaro che per raggiungere questo obiettivo l'altro approccio sarebbe stato quello di scegliere in maniera casuale il contenuto da richiedere, adottando in tal modo una distribuzione uniforme, ma ciò è stato evitato per le sopracitate ragioni di semplicità.
Anche con la modalità prescelta si riesce ad ottenere il risultato voluto, ovvero che ciascun contenuto ai fini pratici viene richiesto lo stesso numero di volte (salvo per il troncamento della serie alla conclusione della simulazione).
Vedremo quando si parlerà di possibili contromisure per l'attacco descritto, come un rimedio ad una delle proposte consista nel richiedere i contenuti secondo una distribuzione che segue la legge di Zipf, in maniera analoga a quanto fanno le altre applicazioni, queste sotto il controllo dell'utente.
Il perché diverrà chiaro nel contesto di quella discussione, che verrà trattata nel capitolo 6.

Ciò che rimane come parametro su cui l'attaccante può agire è la quantità e l'identità stessa dei contenuti da far richiedere al nodo compromesso.

Questo fatto apre una serie di possibilità su come distribuire il carico fra molteplici nodi compromessi; un aspetto questo che meriterebbe un approfondimento maggiore di quanto è stato possibile fare in questo lavoro.
In ogni caso, nel capitolo 5, in cui verrà presentato ciò che è emerso dalle simulazioni, quella del *load balancing* è una tematica affrontata grazie al supporto fornito da alcuni risultati di indagini preliminari in questa direzione.

4.3.4 - Router

Il routing in ambito CCN rappresenta al momento ancora un territorio in cui è attiva l'esplorazione.
Pur essendoci molteplici proposte, alcune delle quali particolarmente interessanti [4], nessuna di queste si è imposta come standard.
Pertanto qui ipotizziamo che la configurazione dei router sia stata creata seguendo un algoritmo, la cui definizione non rientra nello scopo della presente tesi, che associ ad ogni tratta (rappresentata da una entry nella FIB) un valore che rappresenti il ritardo accumulato seguendo tale tratta.
Una volta ordinate le tratte per valori di ritardo crescente, queste vengono esplorate una ad una finché vengono ricevute ritrasmissioni per lo stesso contenuto.
Perciò al primo arrivo di un interest, questo verrà instradato verso la prima tratta presente nella FIB per quel contenuto, mentre a ciascuna ritrasmissione successiva corrisponde ciclicamente la tratta posizionata successivamente nella FIB, fino a quando esse non vengono esaurite.
Siccome si suppone che tutti i contenuti richiesti siano disponibili presso i produttori, in realtà il contenuto richiesto viene sempre consegnato.
Nell'eventualità di più tratte posizionate allo stesso livello, questo protocollo di routing instrada la richiesta verso tutte le tratte allo stesso momento; questo però rimane l'unico caso in cui un parallelismo nell'instradamento delle richieste viene attuato.
Il parallelismo appena citato ha una diretta conseguenza sui risultati dell'attacco, poiché contribuisce a diminuire gli interest che giungono al produttore, il che è chiaramente desiderabile; d'altro canto però il ritardo viene aumentato, in quanto ogni richiesta da parte di Consumer 1 viene prima inviata agli helper e solo allo scadere dell'intervallo di ritrasmissione una nuova richiesta viene generata da Consumer 1 e successivamente instradata verso il produttore.
È stato in questo modo esposto un altro limite dell'attacco, dovuto a scelte architetturali alle quali è difficile porre rimedio lavorando nei limiti di possibilità imposti all'attaccante.
Vedremo in seguito quale influenza quantitativa questo aspetto avrà sull'efficacia dell'attacco e quali sono i possibili rimedi attuabili dall'attaccante.
Rimane da citare un'ultima eccezione al funzionamento del protocollo appena illustrato, di facile comprensione; se viene ricevuto un interest dalla stessa tratta su cui esso andrebbe instradato, il router lo instrada invece sulla tratta successiva nella lista.
In questo modo viene evitata l'introduzione di un ulteriore ed inutile ritardo.

Per ragioni di semplicità, le tratte sono state configurate in maniera statica, ma come se fossero state generate dal protocollo sopra descritto.
Fin dall'inizio dei test i router hanno, di conseguenza, conoscenza di quali siano le destinazioni per ciascun tipo di richiesta e di quale sia il loro ordine.

Sono due gli scenari presentati in questo studio; il primo in cui l'architettura CCN è completamente implementata, quindi entrambi i router sono dotati di una cache, ed il secondo che simula un'introduzione meno ortodossa di CCN (oltre che meno costosa), in cui non sono presenti cache nei router.
È stato scelto di presentare anche questo secondo scenario per due ragioni; la prima è che nel caso di un'implementazione futura su larga scala di CCN è ragionevole che questa possa avvenire in maniera graduale e che, di conseguenza, si possano trovare parti della rete in cui essa sia solo parziale.
Si ritiene che l'aspetto che possa con maggiore probabilità essere inizialmente trascurato sia proprio quello dell'inserimento di cache nei router, per ragioni squisitamente economiche.
Il secondo motivo è strettamente legato al primo, poiché l'assenza di cache interne alla rete costituisce un maggior incentivo per il produttore malevolo ad avvantaggiarsi di quelle presenti nei nodi utente.
Pertanto simulazioni effettuate considerando questo scenario sono interessanti perché mostrano quanto significativo sia questo incentivo per l'attaccante.

4.3.5 - Produttori

Questi nodi sono programmati per rispondere con un pacchetto di dati ad ogni interest ricevuto.
Sono stati utilizzati due nodi che forniscono i contenuti, Producer 1 che riveste il ruolo di produttore malevolo e Producer 2 che invece non attua alcuna misura per rendere i propri contenuti maggiormente popolari.
Essi offrono contenuti appartenenti a due domini che non si sovrappongono, perciò i flussi di traffico che collegano utenti e produttori rimangono ben distinti, per una maggiore facilità di analisi.
I produttori, in sintesi, svolgono il ruolo di "magazzini" che non fanno altro che fornire ciò che viene loro richiesto.

4.3.6 - Funzionamento in assenza di attacco

La dinamica di funzionamento della rete può essere schematizzata dal seguente diagramma (in cui sono stati omessi Helper 2 e 3 per semplificare) (Fig. 4.3):

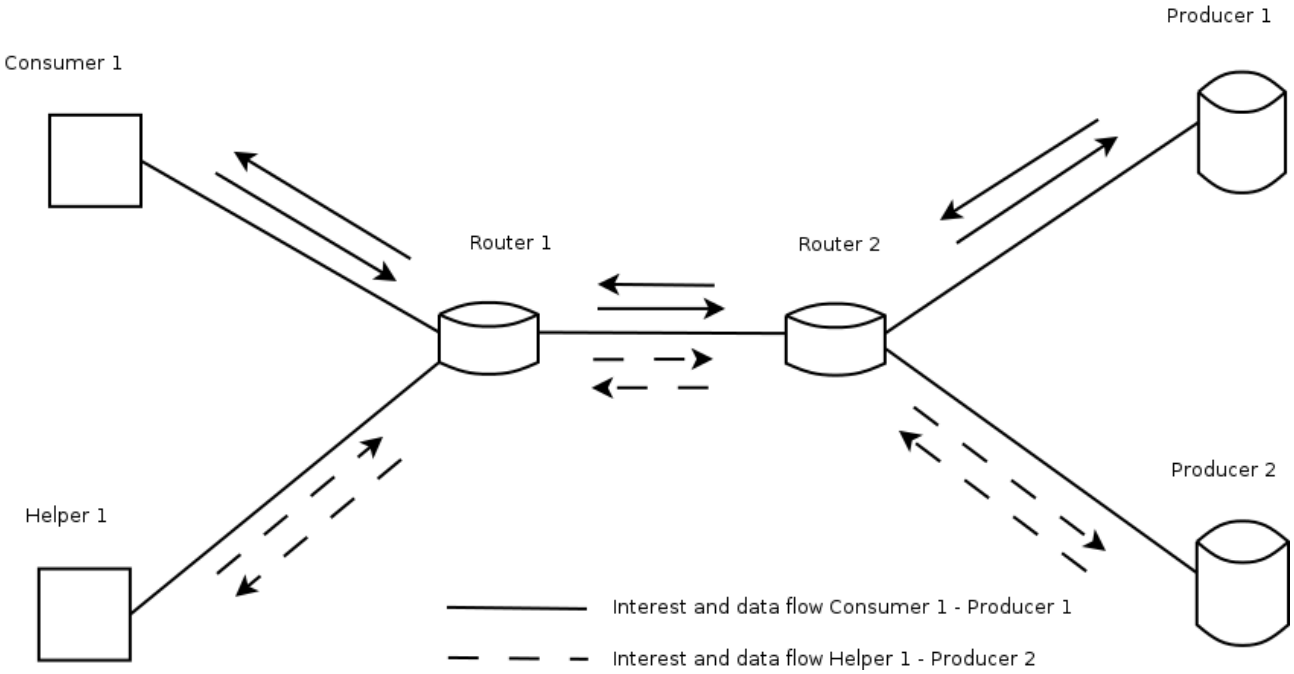


Fig. 4.3 Dinamiche di rete in assenza di attacco

Come la figura esplica efficacemente il funzionamento è piuttosto lineare; grazie alle tratte così configurate ed alla non sovrapposizione dei contenuti, i due flussi rimangono ben distinti ed indipendenti.
È chiaro che la richiesta raggiungerà il rispettivo produttore solo se durante il percorso questa non troverà il contenuto memorizzato in una delle cache.
Resta a questo punto solo da esaminare il comportamento dei nodi in presenza di attacco, il che viene fatto nel paragrafo successivo.

4.3.7 - Funzionamento in presenza di attacco

In caso di attacco il comportamento viene illustrato nella figura seguente (Fig. 4.4):

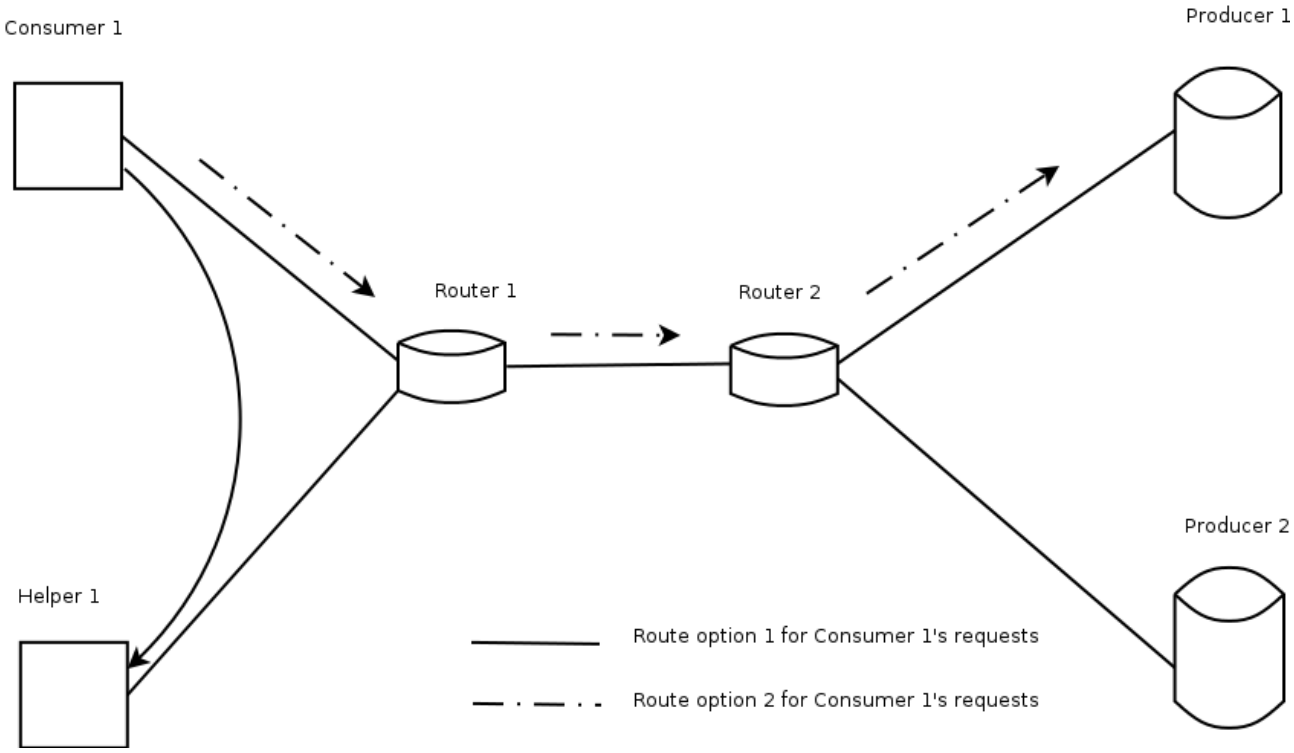


Fig. 4.4 Dinamiche di rete in presenza di attacco

Quando Consumer 1 richiede un contenuto, la sua richiesta viene dapprima dirottata verso Helper 1, che rappresenta la prima opzione in quanto più vicino, in grado quindi di offrire una velocità superiore nella consegna, se il contenuto richiesto è in cache.
Se ciò non avviene, la conseguente ritrasmissione dell'interest per lo stesso contenuto viene instradata lungo la tratta successiva, in questo caso verso Producer 1.
Siccome la cache di Helper 1 va alimentata dei contenuti richiesti da Consumer 1 per essere utile, l'attaccante procede a far richiedere tali contenuti anche ad Helper 1, ora sotto il suo controllo.
Quali e quanti di questi contenuti Helper 1 debba richiedere viene deciso dal produttore malevolo in base a cosa egli voglia disseminare in rete.
Nel caso siano presenti più vittime cosa far richiedere a chi è un parametro, come già specificato, modificabile dall'attaccante.
Sempre se sono presenti più vittime, la prima richiesta di Consumer 1 viene instradata parallelamente a ciascuna di esse.

4.3.8 - Il simulatore

Il software utilizzato per costruire lo scenario e tutti i test simulativi è ndnSIM, introdotto nel 2012 con il white paper "ndnSIM: NDN simulator for NS-3" di Alexander Afanasyev, Ilya Moiseenko e Lixia Zhang.
Si tratta di un'espansione per il noto simulatore di rete ns-3 [8] che implementa tutte le funzionalità e gli aspetti dell'architettura CCN.
Una delle sue prerogative più interessanti è la modularità, il che permette di modificare in maniera indipendente le singole parti costituenti dei nodi, per esempio variando il tipo di content store utilizzato nei singoli nodi ma tenendo allo stesso tempo costante la strategia di forwarding dei pacchetti.
Un altro utilissimo strumento è il sistema di creazione di tracce della simulazione, anch'esso interamente personalizzabile a seconda delle esigenze.
La sua struttura permette di creare agganci in qualsiasi punto del modulo che fornisce i dati per poi convogliarli al tracer, che si occupa della parte di presentazione dei dati.
Nel complesso ndnSIM è un software complesso, ma potente e flessibile; una volta fatti propri i meccanismi che stanno dietro al suo funzionamento si può avere accesso ad una piattaforma in grado di simulare estese topologie CCN in ogni loro aspetto ed interamente modificabile ed adattabile ad ogni necessità.

4.3.9 - Software di analisi

Si vuole infine menzionare il programma utilizzato per lo svolgimento delle analisi dei risultati forniti da ndnSIM, ovvero il noto pacchetto open source per l'analisi statistica, R.

4.3.10 - Riassunto della sezione

In questa sezione è stata definita la topologia di rete utilizzata durante la simulazione.
In accordo con il quadro generale presentato nella precedente sezione, essa è costituita dalle due aree popolate da utenti e produttori e collegate tra loro dallo strato di rete, nel caso specifico composto da due router.
Il comportamento di ciascuno dei nodi è stato modellato in maniera da fornire una rappresentazione più fedele possibile di quello che può essere il loro comportamento reale.
Consumer 1 è un utente che richiede contenuti a Producer 1, il produttore malevolo, per cui, anche se inconsapevolmente, risulta un beneficiario diretto dell'attacco.
Gli helper sono utenti che richiedono contenuti a Producer 2 e che vengono compromessi per prendere possesso delle loro cache tramite la richiesta di contenuti a Producer 1.
La configurazione dei router è ipotizzata essere il risultato di un algoritmo che produce una lista di tratte ordinata secondo valori di ritardo crescenti e che viene esplorata una tratta alla volta per ciascuna ritrasmissione.
Infine sono stati brevemente citate le due applicazioni principalmente utilizzate per costruire ed eseguire i test simulativi e per analizzarne i risultati prodotti, ndnSIM e R.
Nel prossimo capitolo verranno presentati i singoli scenari simulati, definiti parametri e metriche delle simulazioni, descritte le ipotesi preventive in congiunzione coi risultati numerici e discusse le osservazioni.

4.4 - Descrizione dei parametri e delle metriche di simulazione

Questa sezione ha lo scopo di introdurre al lettore le diverse configurazioni che sono state scelte per le simulazioni, in aggiunta ai parametri utilizzati ed alle motivazioni che hanno portato alla loro scelta.
Oltre a questo vengono discusse le metriche che si è deciso di utilizzare per dare una caratterizzazione sia dei vantaggi che un produttore malevolo può ottenere, sia delle penalità imposte agli utenti compromessi.

4.4.1 - Gli scenari

Due sono i vettori che si è deciso di esplorare, così come due sono i valori che queste variabili possono assumere, il che produce quattro distinti scenari, schematicamente riassunti nella tabella sotto riportata (Tab. 4.1):

	Absence of caches in the routers (*nocaches*)	Presence of caches in the routers (*caches*)
Users-Producers RTT = 30 ms (*short*)	Scenario A1: *nocaches-short*	Scenario A2: *caches-short*
Users-Producers RTT = 130 ms (*long*)	Scenario B1: *nocaches-long*	Scenario B2: *caches-long*

Tab. 4.1 Differenti scenari

4.4.1.1 - Scenario A1

Questo scenario è volto a simulare una rete di media estensione; 30 ms per l'RTT è un valore che si può osservare tra due punti situati in differenti ma vicine nazioni d'Europa (ad esempio dall'Italia alla Francia, Spagna, Germania o Olanda).
Si precisa che il valore di RTT considerato è una media di ciò che si può osservare tra due punti ad una distanza grossomodo compresa tra 600 e 1200 km; in buona sostanza è come se gli utenti si trovassero a Milano e i produttori a Berlino, Amsterdam o Parigi.
La scelta di escludere le cache dai router è motivata, come già brevemente menzionato in precedenza, dalla supposizione che un'eventuale implementazione dell'architettura CCN possa avvenire in maniera graduale ed in maniera più rapida da parte degli utenti piuttosto che dell'infrastruttura di rete.
Questo porterebbe ad avere router che sì gestiscono l'indirizzamento nominale e le altre caratteristiche di una CCN, ma non il caching, per ragioni di costo.
Pertanto in questo scenario ci troviamo di fronte ad una CCN parzialmente implementata e di estensione "europea" (per lo meno nel raggio massimo di circa 1200 km).

4.4.1.2 - Scenario A2

Questa volta ci troviamo nelle condizioni precedenti per ciò che concerne l'estensione della rete, ma con CCN completamente implementata, anche a livello di cache interne ai router lungo il tragitto.
Sono stati analizzati anche gli scenari che prevedono l'utilizzo delle cache nei router per due motivi; il primo è che in tal modo si implementa completamente l'architettura CCN, mentre il secondo risiede nella volontà di comprendere la differenza tra l'effetto dell'attivazione delle cache nei router e quello che invece ha la disponibilità di numerose cache all'interno dei nodi degli utenti per il produttore malevolo.
Della capacità di tali cache si parlerà in seguito, quando saranno affrontati e discussi tutti i parametri simulativi insieme alle motivazioni che hanno portato all'assegnamento degli specifici valori.

4.4.1.3 - Scenario B1

In questo frangente si vogliono mostrare gli effetti dell'attacco all'interno di una rete maggiormente estesa rispetto al caso precedente, come denota il RTT tra utenti e produttori di 130 ms.
Questo valore si può registrare tra due nodi posti ad una distanza di qualche migliaio di km, come quella che potrebbe separare un utente in Italia da un produttore in nord o centro America o in uno dei paesi del Medio Oriente.
Perciò in questo caso ci troviamo di fronte ad una CCN che copra un'area geografica piuttosto estesa, ma in cui il caching è solo prerogativa dei nodi utente.
La motivazione dietro questa situazione e quella simulata in B2 sta nel fatto che un produttore più lontano rispetto ai propri utenti può vedere un incentivo in più nel perpetrare questo attacco, che promette in teoria di abbassare la quantità di dati che il produttore deve immettere in rete e di migliorare le prestazioni viste dagli utenti proprio grazie all'accorciamento della distanza tra essi ed i contenuti da loro richiesti.
Al di là di questi aspetti però, questo scenario, così come B2, non dovrebbe discostarsi come dinamiche della rete da quelli discussi in precedenza.

4.4.1.4 - Scenario B2

Questo scenario segue lo stesso razionale che ha generato B1, ma con in più l'inserimento delle cache nei router, per le medesime ragioni illustrate nel paragrafo relativo ad A2.

4.4.2 - Note finali sugli scenari

La scelta delle differenti situazioni da rappresentare è frutto di un attento ragionamento che ha riguardato vari aspetti.
Uno di quelli più importanti è il tentativo di approssimare la realtà nella maniera più fedele possibile ogni volta in cui ciò rientra nei limiti di fattibilità imposti dal simulatore o dai tempi di esecuzione.

Ove questo non sia realizzabile, si è comunque cercato di avanzare ipotesi che potessero coprire casistiche concepibili nel mondo reale.
L'altra serie di considerazioni riguarda ovviamente le semplificazioni introdotte nel modello; lo scopo del presente studio è quello di presentare il problema ben definito nei capitoli precedenti cercando di analizzarne la gravità e la sua dipendenza da alcuni fattori che sono stati anticipatamente giudicati come possibili agenti significativi in questo processo e non di variare qualsiasi tipo di parametro per coprire ogni genere di situazione che si possa presentare.
Per questo motivo sono presenti numerose concessioni semplificative rispetto alla realtà, che emergeranno nel corso dell'analisi.
È da questo bilanciamento che sono nati topologia e scenari descritti; da una parte la prima a rappresentare un'evidente approssimazione, a causa del numero limitato di nodi presenti per cercare di contenere la complessità del sistema, dall'altra i secondi, progettati per dare una caratterizzazione maggiormente incline alla realtà.

4.4.3 - Parametri numerici

Le variabili simulative ed i valori o intervalli di valori associati a ciascuna di esse sono riassunti nella tabella seguente (Tab. 4.2):

Parameter name	**Value(s) (or range of values)**	
Link speed	1 Gbps (in both directions)	
Link latency	RTT = 30 ms	5 ms for all links
	RTT = 130 ms	5 ms for Con/Hel-Rtr1 links 10 ms for Rtr1-Rtr2 link 50 ms for Rtr2-Prod links
Routers' cache size	100 contents	
Consumer 1's cache size	100 contents	
Helpers' cache size	300, 600 or 1000 contents	
Cache replacement policy	LRU	
Number of contents served by Producers	10000	
Number of contents illegitimately requested by helpers	1000 (the 1000 most popular served by Producer 1)	
Content size	1000 B (excluding header)	
Consumer 1's interest rate	100 requests/s	
Helpers' legitimate interest rate (normalized by illicit interest rate)	1, 5, or 10 (in some tests from 1 to 10 with step 1)	
Zipf exponent (valid for all users)	0.9	
Number of users (compromised or not)	4	
Number of helpers (compromised users)	1, 2 or 3	
Single test duration	7200 s (2 h)	

Tab. 4.2 Parametri simulativi

Nel seguito si scende maggiormente nel dettaglio di ciascuna di queste variabili e si esaminano i motivi per la loro scelta e per quella dei valori che esse assumono nei test.

4.4.3.1 - Velocità dei link

Per evitare di introdurre colli di bottiglia che potessero influenzare le simulazioni si è deciso di utilizzare linee che dessero ampie garanzie su questo fronte; date le piccole dimensioni dei pacchetti circolanti, queste non hanno mai la possibilità di essere saturate, nonostante si possano presentare notevoli picchi di traffico, soprattutto nella tratta che congiunge i due router.
Inoltre, come semplificazione, i link sono supposti essere ideali, perciò non introducono errori durante il transito.
Essenzialmente sono state simulate delle tratte in fibra ottica in ogni parte della rete, con l'aggiunta dell'idealità che riguarda l'assenza degli errori di trasmissione.

4.4.3.2 - Ritardo di tratta

Il ritardo accumulato sulla singola tratta è all'interno di valori considerati tipici per un collegamento in fibra.
Nel caso degli scenari *long* (B1 e B2), non è detto che l'aumento del ritardo nelle tratte tra Router 1 e Router 2 e tra Router 2 ed i Producer sia dato solamente da linee più lunghe, ma anche dall'attraversamento di ulteriori hop che nella topologia di riferimento non vengono rappresentati.
Questo tuttavia è irrilevante ai fini della discussione, se non per il fatto che se ci fossero più hop da attraversare sulla linea il caching dei contenuti potrebbe avvenire anche in tali punti; questa eventualità viene esclusa per semplicità di trattazione e viene quindi supposto che semplicemente i dati transitanti su tali linee accumulino ritardi rispettivamente di 10 e 50 ms ad ogni passaggio.

4.4.3.3 - Grandezza delle cache dei router

Qui il limite è stato posto al valore relativamente basso di 100 contenuti, che rappresenta lo 0.5% del totale dei contenuti disponibili in rete.
Le ragioni che hanno portato a questa decisione sono due; da una parte questa indagine non vuole dimostrare l'efficacia delle cache in rete, perciò non costituisce scenario di interesse quello in cui le cache dei router possono assorbire in larga parte o completamente la domanda da parte degli utenti.
Dall'altra ci sono sempre le considerazioni di costo, questa volta con riferimento in maniera specifica a quelli da sostenere per il mantenimento di cache di grandi dimensioni in ciascun router della rete, che funzionano e sono soggette a pesante utilizzo ogni giorno a tutte le ore.
Per questi motivi, nonostante, come già precedentemente menzionato, si ritenga importante valutare l'eventuale impatto di cache di rete nello scenario proposto, esse verranno mantenute di dimensione fissata e limitata.

4.4.3.4 - Grandezza della cache di Consumer 1

Il valore scelto per l'estensione della cache di Consumer 1, ovvero dell'utente che inconsapevolmente beneficia dell'attacco, ha motivazioni che seguono molto da vicino quelle indicate nel paragrafo precedente.
Qui però le considerazioni di costo perdono forza; se gli altri utenti (helper) possono avere cache da 300 e fino a 1000 contenuti, perché anche Consumer 1, sempre un utente, non dovrebbe disporne?
In questo caso, in effetti, l'unica motivazione per la ristrettezza delle dimensioni è sullo scarso interesse nel caso in cui Consumer 1 sia dotato di un'ampia cache; se così fosse infatti, la maggior parte delle richieste potrebbe essere facilmente servito in maniera diretta da tale cache, con gli interest che solo raramente lascerebbero il nodo stesso.
È stato comunque deciso di dotare Consumer 1 di una cache sia per aderire al modello CCN, che per evitare un'eccessiva ed improbabile disparità tra le varie categorie di utenti.

4.4.3.5 - Grandezza della cache degli helper

Per questo parametro era stato pensato inizialmente un insieme di valori più basso e simile a quello alla fine scelto per Consumer 1, adducendo come motivazione la ragionevolezza dell'ipotesi di avere cache di dimensioni simili tra gli utenti.
Tuttavia, siccome le cache degli helper devono poter contenere sia i contenuti di legittimo interesse per l'utente, che quelli di interesse per l'attaccante, esse dovranno per forza essere di dimensione maggiormente elevata rispetto a quanto inizialmente ipotizzato.
Questo costituisce un ulteriore punto da considerare per l'attaccante, che dovrà scegliere le proprie vittime anche in base allo spazio a disposizione per i propri contenuti.
Infatti, anticipando per un momento l'analisi dei risultati, test simulativi condotti col parametro in oggetto impostato a valori di 100 e 200 contenuti hanno mostrato l'inefficacia dell'attacco, tranne in ben specifici casi.
Data la loro insignificanza, questi risultati vengono menzionati solo nel presente paragrafo, perciò non compariranno nella presentazione più avanti nel capitolo.

4.4.3.6 – Politica di rotazione della cache

Data l'ampia diffusione di questo meccanismo di gestione della rotazione dei contenuti in cache [9], soprattutto in ambito CCN [10][11][12] e la volontà di contenere la complessità del sistema, è stato deciso di utilizzare soltanto LRU per ogni cache presente.
Quando un interest arriva ad una cache che utilizza questa policy e che ha il contenuto richiesto, esso viene posto in cima alla pila, poiché è il contenuto che ha ricevuto la richiesta più recente.
Se un nuovo contenuto deve essere memorizzato, lo spazio necessario, a meno che quello libero non sia già sufficiente, viene creato rimuovendo oggetti dal fondo della pila, che corrispondono a quelli che hanno ricevuto le richieste più lontane nel tempo.
Uno sviluppo futuro potrebbe essere quello di valutare gli scenari di attacco descritti al variare del tipo di policy utilizzata.

4.4.3.7 - Numero di contenuti offerti dai producer

Per questo parametro si è voluto cercare un bilanciamento tra una libreria di contenuti sufficientemente ampia per approssimare adeguatamente quella di un realistico produttore, ma non estesa in modo esagerato per evitare di disperdere eccessivamente gli interest e vanificare in una certa misura i vantaggi delle cache.
Un buon compromesso in questo è stato ipotizzato essere il valore di 10000 contenuti.
Oltre a queste considerazioni, un altro aspetto fondamentale che ha inciso sulla scelta del valore da assegnare non solo a questo parametro, ma anche alla frequenza delle richieste ed alla durata dei test, è il funzionamento della classe di ndnSIM che simula l'utente che richiede contenuti secondo la legge di Zipf, chiamata ConsumerZipfMandelbrot.
Alla pagina che contiene la descrizione del comportamento di tale classe (http://ndnsim.net/applications.html), è possibile visualizzare vari grafici che mostrano i vari gradi di approssimazione della simulazione nei confronti della controparte teorica (mostrati per comodità nelle seguenti Figg. 4.5 e 4.6):

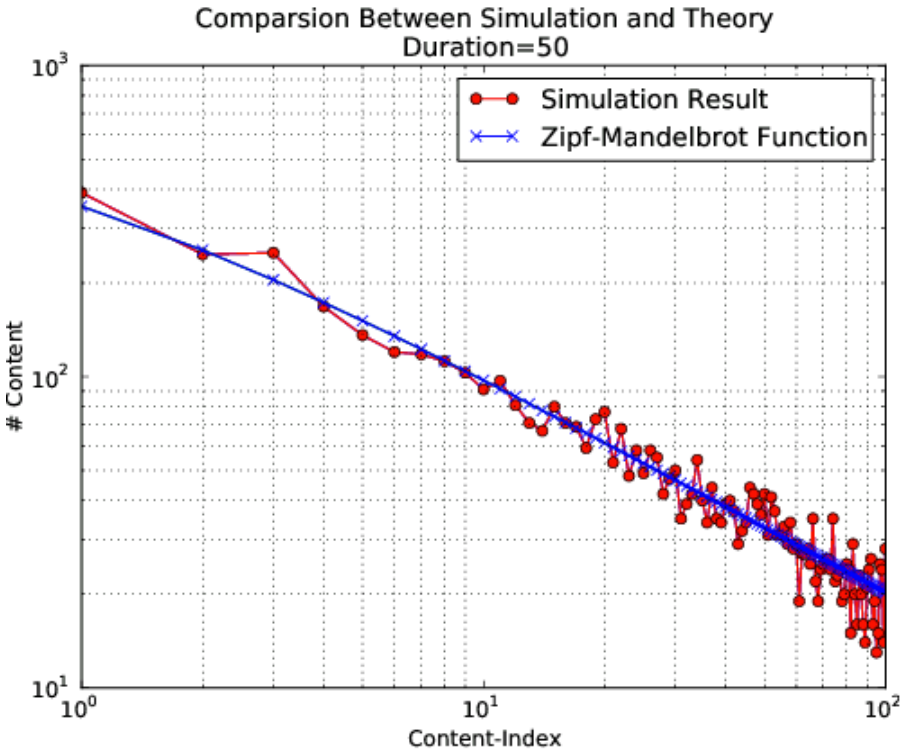


Fig. 4.5 Approssimazione della legge di Zipf in ndnSIM (1)

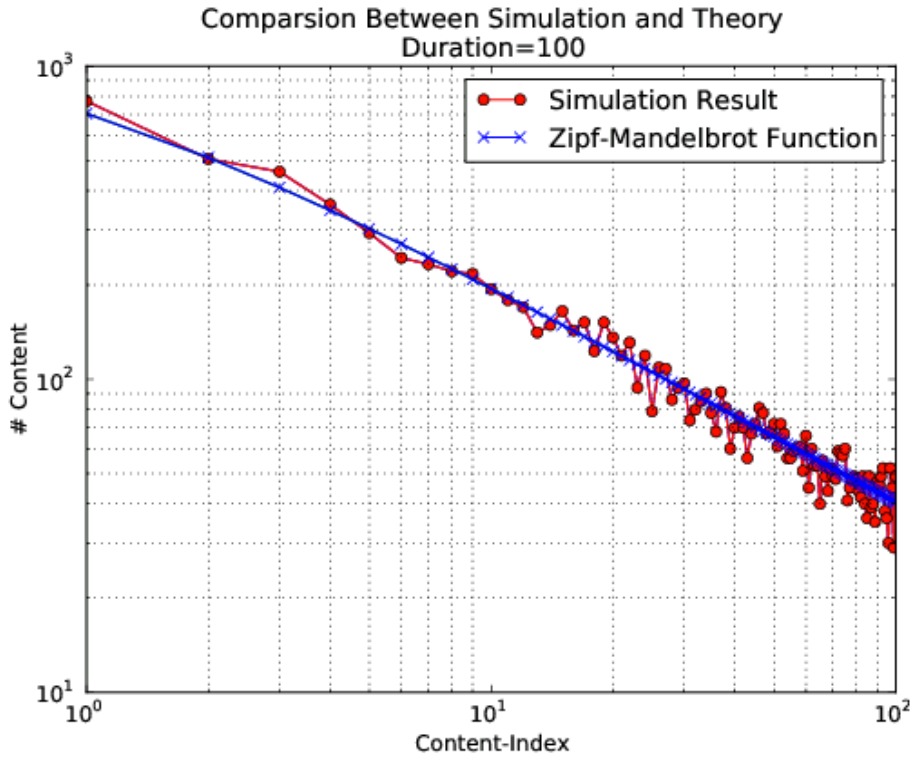


Fig. 4.6 Approssimazione della legge di ZIpf in ndnSIM (2)

In questi casi è stata usata una frequenza di emissione degli interest di 100 al secondo, che, moltiplicata per la durata degli esperimenti risulta in rispettivamente un totale di 5000 e 10000 interest emessi nel corso delle simulazioni.
Siccome in entrambi i casi il numero totale di contenuti unici richiesti è 100, si ottiene un'approssimazione per lo meno ragionevole della legge di Zipf nel primo caso (Fig. 6.2), quando il rapporto fra numero totale di interest generati su numero di contenuti unici è 50 ed una ancora migliore nel secondo (Fig. 6.3), in cui il rapporto è 100.
Con i valori utilizzati nel presente lavoro tale rapporto è di 72 (100 richieste/s*7200 s/10000 contenuti), perciò l'accuratezza, se ogni altro parametro rimane costante, si colloca circa a metà tra i due casi sopra menzionati.
Dato che al crescere di questo rapporto cresce l'accuratezza del modello simulativo, è palese che c'è stato lo sforzo di renderlo il più alto possibile, mantenendo però al contempo ragionevoli gli altri parametri; il che vuol dire un'ampiezza del catalogo non troppo bassa, una frequenza di interest non esageratamente elevata ed un tempo di esecuzione dei test abbastanza lungo perché si potesse raggiungere una condizione di stabilità del sistema, ma non troppo allo scopo di poter eseguire il maggior numero possibile di simulazioni.

4.4.3.8 - Numero di contenuti illegittimamente richiesti dagli helper

Come spiegato nei capitoli precedenti, quando un nodo utente viene compromesso, diventando quindi un "helper", l'attaccante farà in modo, tramite un'applicazione, di far richiedere a questo nodo i contenuti serviti da Producer 1.
Dato che tale fatto non è noto all'utente, queste richieste sono state denominate illegittime o illecite, nel senso che non sono state prodotte secondo la volontà dell'utente e, volendo puntualizzare, nemmeno dall'utente stesso in senso stretto.
Queste richieste hanno una caratteristica fondamentale; sono dirette verso i 1000 contenuti più popolari.
L'ultima affermazione è soggetta alle seguenti ipotesi: l'attaccante deve avere la possibilità di conoscere quali siano i contenuti maggiormente popolari, o perché analizza egli stesso il traffico in arrivo a Producer 1 e costruisce quindi il grafico della popolarità, oppure perché riceve questa informazione direttamente dal produttore.
In ogni caso l'informazione è nota prima dell'inizio dell'attacco e rimane valida per tutta la durata della simulazione; questo implica che la curva della domanda non cambi nell'arco del test, come precedentemente già puntualizzato.
Resta da motivare la scelta dello specifico numero di contenuti.
Il numero 1000 si spiega attraverso considerazioni che il produttore deve fare e che vertono su un bilanciamento tra numero di contenuti che si vuole disseminare, numero di nodi che si è disposti a compromettere e ciò che essenzialmente si può riassumere come grado di efficacia della disseminazione.
È ovvio che l'ideale obiettivo per il produttore sarebbe quello di distribuire il maggior numero di contenuti all'interno della più vasta area geografica e per il maggior tempo possibile.
Per raggiungere questo obiettivo però c'è bisogno di un grande numero di nodi, ben distribuito sul territorio, e che emette un numero di richieste sufficiente a tenere in cache i contenuti.
Perciò il produttore dovrà accettare compromessi su tutti questi fattori per ottenere risultati soddisfacenti a costi accettabili, che comunque devono restare competitivi con la situazione originaria.
Tenendo basso il numero dei nodi compromessi e la frequenza di richieste illecite che ciascuno di essi produce, l'unica strada per far rimanere elevata l'efficacia dell'attacco è evitare di distribuire queste richieste su un insieme troppo alto di contenuti, col risultato di non riuscire a tenerli in cache abbastanza a lungo, il che si traduce nella limitazione sul numero di contenuti totali da richiedere.
È ovvio che in presenza di un solo helper, su di esso ricade l'intera mole di richieste da inviare, che saranno quindi dirette verso l'intero insieme di 1000 contenuti.
Se sono presenti nodi compromessi aggiuntivi, il comportamento deciso non è quello di far richiedere a ciascuno tutti i 1000 contenuti, ma di suddividere tra di loro il carico come sotto riportato nella tabella sottostante (Tab. 4.3).

	2 helpers	3 helpers
Same behavior	• 1 helper requests contents from 1 to 250 • The other requests contents from 251 to 1000	• 1 helper requests contents from 1 to 250 • The other 2 request contents from 251 to 1000 in an alternating fashion
Different behavior	• The helper with the lowest legitimate request frequency sends interests for contents from 1 to 250 • The other requests contents from 251 to 1000	• The helper with the lowest legitimate request frequency sends interests for contents from 1 to 250 • The other 2 request contents from 251 to 1000 in an alternating fashion

Tab. 4.3 Suddivisione del carico tra le vittime

Quando tutte le vittime emettono interest legittimi alla stessa frequenza si dice che hanno lo stesso comportamento, mentre hanno diverso comportamento quando tale frequenza differisce anche solo per uno di loro rispetto agli altri.
Nel caso di comportamento differente la logica che viene applicata è quella di caricare maggiormente (con i contenuti più popolari) l'helper che offre il minimo grado di competizione per la sua cache, ovvero quello invia interest legittimi alla minima frequenza.
Il valore di 250 contenuti più popolari è stato scelto dopo l'osservazione della curva che rappresenta il numero di richieste soddisfatte da Producer 1 per ciascun contenuto in una condizione di non attacco (Fig. 4.6).
Come si vede nella figura, il 75% delle richieste di Consumer 1 soddisfatte da Producer 1 per i 1000 contenuti più popolari in realtà è concentrata nei primi 244 contenuti e, dal momento che tale numero non è elevato in termini assoluti (anche le cache di minime dimensioni possono contenere questo numero di contenuti), è stato deciso di assegnare tale carico nella sua interezza alla cache meno contestata.
Quando sono presenti 3 helper, si pone il problema di come distribuire il restante carico tra coloro che devono servire i contenuti tra 251 e 1000.
Per semplificare e cercare di trovare una distribuzione equa, è stato scelto di far richiedere ad uno tutti i contenuti con ID dispari ed all'altro tutti quelli con ID pari.
Questo tema appena introdotto della distribuzione del lavoro tra le vittime è stato solo superficialmente toccato in questo lavoro, dal momento che sono stati svolti solo test preliminari volti a farsi un'idea di massima dell'efficacia di alcuni approcci, ma, secondo l'opinione di chi scrive, costituisce un interessante problema di "load balancing" in un contesto diverso da quello comune; pertanto si può ascrivere alla categoria delle possibili aree di interesse per futuri sviluppi.

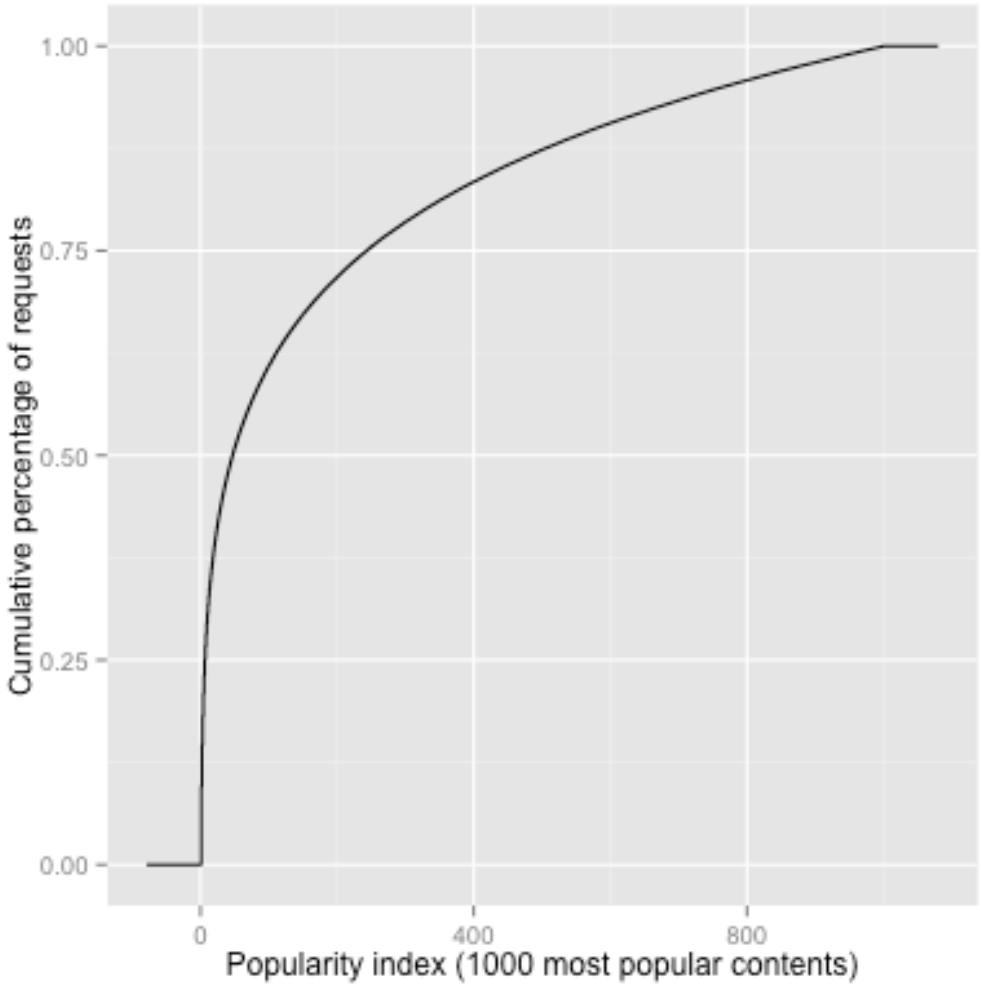


Fig.4.7 Andamento delle richieste soddisfatte da Producer 1

4.4.3.9 - Dimensione dei singoli contenuti

Le scelte possibili per questo parametro sono due; un valore costante, o in accordo ad una distribuzione geometrica avente il valore medio scelto.
Entrambe le strade sono state percorse in letteratura, la prima in [7] e [13], mentre la seconda in [5] e [11]; per questo lavoro si è scelto, per ragioni di semplicità, di seguire la prima delle due alternative.
Ciò rende anche possibile normalizzare la capienza delle cache, esprimendola quindi in termini di contenuti piuttosto che di byte.

4.4.3.10 - Frequenza degli interessi di Consumer 1

La motivazione dietro la scelta di questo parametro è stata data nel paragrafo relativo alla quantità di contenuti serviti dai produttori, poiché questi due fattori sono legati tra loro e determinanti per la bontà dell'approssimazione della legge di Zipf offerta da ndnSIM.
Inoltre, uno dei motivi utili ad incentivare un produttore a perpetrare questo tipo di attacco, è che la domanda per i suoi contenuti sia alta; da qui la decisione di tenere questo valore alto.

4.4.3.11 - Frequenza degli interessi legittimi degli helper

Questa grandezza non è espressa in interest al secondo, ma come rapporto con la frequenza delle richieste illecite emesse dagli helper, la quale rimane fissa per tutti i test e ad un valore non elevato per mantenere basso il profilo dell'attacco.
Ciò serve per dare immediatamente un'idea del grado di contesa della cache.
Per grado di contesa si intende la difficoltà che l'attaccante incontra nel tenere memorizzati nella cache in questione i contenuti di Producer 1.
Questa grandezza dipende dal rapporto fra richieste legittime/illegittime emesse dall'helper, dalla dimensione della cache e dal numero di contenuti unici illegittimamente richiesti.
L'idea iniziale era quella di far variare il rapporto fra richieste legittime/illegittime con continuità fra 1 e 10, quindi da una condizione di uguale frequenza di emissione ad una di frequenza di richieste legittime 10 volte superiore rispetto a quella delle richieste illegittime.
È stato deciso di esplorare solo i casi di attività legittima pari o superiore a quella illegittima per capire se un attaccante può ottenere benefici significativi anche da una posizione di svantaggio, mentre non ci si è interessati di indagare un comportamento aggressivo dell'attaccante; da una parte perché uno degli obiettivi è quello di rendere l'attacco poco visibile ed identificabile, dall'altra perché appare ovvio l'ottenimento di un vantaggio se l'attaccante può sovrastare l'utente legittimo in termini di traffico.
Compiuti i primi esperimenti con cache di dimensioni abbastanza ampie (600 e 1000 contenuti), ci si è resi conto che nella maggior parte dei casi, le metriche misurate (descritte in seguito) avevano andamenti piuttosto lineari, per cui è stato deciso, per evitare di accumulare risultati poco informativi, di studiare il comportamento del sistema per 3 valori specifici del parametro e non più con continuità.
Questi valori sono 1, 5 e 10 e possono essere etichettati come rispettivamente basso, medio ed alto grado di contesa.
Pertanto i risultati di alcune simulazioni saranno rappresentati secondo l'intera scala come inizialmente previsto, ma la maggioranza di essi conterrà solo i 3 valori appena citati.

4.4.3.12 - Esponente di ZIpf

Contrariamente a ciò che avviene per la modellizzazione della curva di popolarità dei contenuti richiesti, che viene comunemente approssimata da una distribuzione di tipo Zipf, non esiste una valore preciso e condiviso in letteratura per l'esponente da attribuirle [5].
Gli approcci utilizzati possono consistere nella selezione di un singolo valore in un range che può andare da 0.8 [6] fino a 2.5 [10], oppure nella variazione dell'esponente all'interno di un certo intervallo [5].
Per questo lavoro è stato scelto il primo approccio, che, pur limitato, consente di diminuire la complessità ed i tempi di svolgimento dei test.
Sarebbe sicuramente un'interessante analisi da effettuare quella di osservare le dinamiche del sistema testato al variare dell'esponente della distribuzione.
Data la grande variabilità ed incertezza in letteratura nella scelta di questo parametro ci si è affidati ai risultati emersi in [1], scegliendo quindi un valore inferiore all'unità.

4.4.3.13 - Numero di utenti ed utenti compromessi

Anche in questo caso si è presentata la scelta di focalizzare o meno l'attenzione del presente studio sul comportamento del sistema al variare dell'estensione e composizione della topologia, il che fa sempre parte della ricerca di un bilanciamento fra completezza di analisi e semplicità.
Il risultato è che altri fattori descritti precedentemente sono stati ritenuti più interessanti per questa indagine, perciò si è deciso di limitare le simulazioni ad una sola topologia, in cui si potessero avere da nessuno a tutti i nodi utente compromessi ed al servizio dell'attaccante.

4.4.3.14 - Durata delle simulazioni

La motivazione dietro alla scelta di questo parametro si lega strettamente alla discussione già affrontata nel paragrafo dedicato al numero di contenuti offerti dai produttori, pertanto non verrà qui ripetuta.
Rispetto a quanto è possibile leggere negli altri studi, può sembrare che la durata di 2 ore sia bassa, ma la natura piuttosto regolare del traffico simulato, che risulta dalle scelte effettuate per i parametri (soprattutto dal fatto che l'insieme totale dei contenuti rimane statico), fa sì che il sistema raggiunga la stabilità in breve tempo e che questa sia mantenuta nel corso del tempo, rendendo improbabili cospicue variazioni delle metriche anche prolungando il tempo di osservazione.
Oltre a quanto detto nel paragrafo precedentemente citato in merito alla durata, si vuole anche aggiungere che una durata inferiore rende l'ipotesi di staticità della domanda dei contenuti maggiormente plausibile.

4.4.3.15 - Nota finale sul funzionamento di RTO in ndnSIM

Per come viene gestito il routing (instradamento ad una tratta per volta), ritrasmissioni avvengono ogniqualvolta un contenuto non viene trovato nelle cache degli helper e quindi Consumer 1 è costretto a richiederlo e Producer 1 a servirlo.
Siccome la scelta è stata quella di non fissare un valore per il timeout di ritrasmissione ed affidarsi invece al meccanismo di controllo implementato in ndnSIM, è opportuno scendere maggiormente nel dettaglio del suo funzionamento.
RTO parte da un valore iniziale, configurabile, ma lasciato a quello di default di ndnSIM, vale a dire 1 secondo.
RTO è mantenuto a questo valore fino alla prima ricezione di un contenuto, dopo la quale cala fino al suo valore minimo, posto conservativamente ad un valore elevato (60 ms e 150 ms per i due scenari rispettivamente).
A questo punto tutte le richieste che non hanno ancora ricevuto risposta e la cui somma dell'istante della prima trasmissione con il nuovo valore di RTO ricade precedentemente rispetto all'istante corrente vengono ritrasmesse e RTO viene raddoppiato per ciascuno di questi nuovi invii.
Una volta che tutti i contenuti appartenenti a questo gruppo sono stati ricevuti, RTO ritorna al valore minimo.

4.4.4 - Metriche

Come già discusso nei capitoli precedenti l'obiettivo dello studio è quello di valutare gli effetti dell'attacco sotto 3 punti di vista: quello del produttore malevolo, quello dell'utente interessato ai contenuti generati dal produttore malevolo e quello dei generici utenti, soprattutto quando sono vittime dell'attacco.
Di seguito vengono illustrate e discusse le metriche utilizzate.

4.4.4.1 - Hit ratio degli utenti (Consumer 1 e helper)

L'hit ratio è una metrica fondamentale in ogni studio che riguardi un sistema in cui siano presenti cache.
È chiaro che uno degli obiettivi principali è far sì che questa percentuale sia la più elevata possibile per Consumer 1, mentre per ciò che concerne gli helper si vuole cercare di deprimerla il meno possibile.
Come già menzionato in precedenza è difficile stabilire quanto le differenze in questa metrica possano essere percepibili per un utente medio, tuttavia può essere ragionevole considerare un calo di circa il 5% come difficilmente percepibile e cali oltre il 10% come percepibili.
Per questo l'orientamento consisterà nel considerare come migliori quelle soluzioni che vedono un abbassamento dell'hit ratio per gli helper inferiore al 10%.
La misura di questa metrica è stata condotta considerando il numero delle richieste soddisfatte da una cache, il che è stato poi diviso per il numero delle richieste totali soddisfatte; tutto questo per ciascun utente.
Ciò è stato possibile grazie alla modifica dell'header di un pacchetto di dati in modo che questo contenesse un ulteriore bit ad indicare se il contenuto provenisse da una cache, che veniva poi letto dal ricevitore e registrato in uno degli output della simulazione.
Sostanzialmente un contenuto è stato considerato proveniente da una cache se ricevuto da qualsiasi entità che non fosse un produttore.
Dal calcolo dell'hit ratio, per ragioni di semplicità, sono stati esclusi quegli hit dovuti a ritrasmissioni, ovvero un hit di una qualsiasi cache viene considerato tale solo se è ottenuto al primo tentativo di trasmissione.
Su questa scelta si potrebbe discutere ulteriormente, ma qui ci limiteremo a dire che l'esclusione di queste casistiche dal calcolo dell'hit ratio non sembra avere avuto, anche dopo analisi maggiormente approfondite dei risultati, effetti facilmente riscontrabili.
Dove questo dovesse accadere, tuttavia, le differenze verranno evidenziate.

4.4.4.2 - Contenuti medi consegnati dal produttore (conteggiati solo per quelli in offloading)

La misurazione viene svolta al nodo produttore, presso il quale, ad ogni secondo, vengono tracciati i pacchetti che esso consegna.
Tale valore viene poi mediato sul numero di secondi di simulazione.
Questa è una metrica che, chiaramente, interessa molto al produttore malevolo ed in particolare una che esso vuole tenere bassa il più possibile, perché significa che la rete deve accettare meno del suo traffico visto che i suoi contenuti sono consegnati dagli helper.
Ciò che è monitorato è solo l'insieme dei contenuti che vengono fatti richiedere anche dagli helper, per il resto del catalogo questa metrica rimane pressoché costante in tutti i test.

4.4.4.3 - Durata media dell'intervallo tra due richieste successive giunte al produttore per un contenuto in offloading

Oltre ad abbassare le consegne medie che deve effettuare, il produttore è anche interessato a prolungare il tempo di permanenza in rete dei propri contenuti.
È possibile dare un'idea di questa importante grandezza calcolando quanto tempo mediamente intercorre tra due richieste che da Consumer 1 vengono instradate verso Producer 1.
Vengono conteggiate quelle provenienti da Consumer 1 perché il produttore malevolo è interessato ad alzare il tempo di intercorrenza tra il servizio di due richieste legittime per i suoi contenuti, visto che quelle provenienti dagli helper sono sì propedeutiche al prolungamento del tempo di permanenza in cache, ma non sono indice di un "interesse" vero e proprio nel contenuto.
Il punto migliore in cui prendere tale misura sarebbe Router 2, perché le richieste in uscita da esso sicuramente vanno ai produttori e non ad un'ulteriore cache.
Purtroppo, dato che gli interest sono anonimi in CCN (non ci sono indirizzi con cui poter tracciare la provenienza di una richiesta), un approccio di questo tipo renderebbe impossibile distinguere tra quelli inviati da Consumer 1 e quelli inviati dagli helper.
Pertanto si è scelto di misurare questo dato nel Router 1, poiché esso può distinguere le richieste somministrategli da Consumer 1 grazie alla face di arrivo, pur dovendo mettere in conto un grado di errore dovuto a quelle richieste che Router 1 instrada verso Router 2 e che vengono servite dalla cache di quest'ultimo.
Tuttavia si ritiene che l'errore commesso non sia significativo, date le dimensioni contenute delle cache dei router ed anche il fatto che la cache di Router 2 può registrare un hit solo se ben 3 livelli precedenti (quella di Consumer 1 stesso, quella di Router 1 ed infine quella di uno degli helper) di cache hanno registrato un miss.
In altre parole si può considerare questa misura come una stima leggermente al ribasso del reale valore, il che può essere considerato ragionevole.
Il produttore è ovviamente interessato ad alzare la metrica in questione con l'attacco rispetto alla condizione di non attacco.

4.4.4.4 - Ritardi nella ricezione dei contenuti (sia di Consumer 1 che degli helper)

Altra metrica molto importante è il ritardo con cui vengono ricevuti i contenuti dagli utenti, perché questa è una delle più influenti sulle prestazioni di molte applicazioni e servizi, e, di conseguenza, sull'esperienza d'uso degli utenti.
Il ritardo viene quindi misurato nei nodi degli utenti come l'intervallo di tempo trascorso dalla prima trasmissione dell'interest fino a quello di ricezione del corrispondente contenuto.
Per il Consumer 1 viene misurato il ritardo medio sui contenuti in offloading, perché per il produttore l'obiettivo è quello di abbassare il più possibile il ritardo sui contenuti più popolari.

Per gli helper viene invece misurato il ritardo medio su tutti i contenuti, perché l'intenzione in questo caso è quello di non peggiorare le generiche prestazioni degli helper, quindi di non alzare eccessivamente questo valore.
Anche qui, come nel caso dell'hit ratio, è difficile stabilire esattamente l'effetto delle variazioni sull'esperienza d'uso, ma verranno considerate come migliori quelle situazioni in cui questa metrica è meno compromessa rispetto ad altre, più vantaggiose per il produttore malevolo, ma in cui il ritardo è maggiormente affetto.

4.4.5 - Riassunto della sezione

Nella sezione 4.5 sono stati definiti gli scenari utilizzati per le simulazioni, la cui scelta è stata basata sulla ricerca di un bilanciamento tra vari fattori, tra cui la volontà di descrivere situazioni che possono avere un ragionevole riscontro nelle realtà, i costi e naturalmente il mantenimento di un grado gestibile di complessità.
I parametri simulativi sono stati scelti in base a simili considerazioni, con la preferenza per quelli che sono stati ritenuti significativi ed in grado di influenzare maggiormente le dinamiche di funzionamento.
Molte altre sarebbero potute essere le prove effettuate, le quali avrebbero senz'altro prodotto ulteriori spunti di discussione, ma la necessità di contenere la complessità ed i tempi ha causato la loro esclusione.
Le metriche definite sono state selezionate per la loro capacità di dare una descrizione completa dell'efficacia dell'attacco e dei suoi effetti sugli utenti, oltre a rappresentare scelte molto comuni e condivise in letteratura.
Si ritiene quindi con questa ultima sezione di aver dato a chi legge una panoramica completa sul problema, dalla sua descrizione di alto livello fino ai dettagli sui singoli parametri simulativi, pertanto così si chiude il capitolo.
Nel prossimo verranno mostrati i risultati ottenuti abbinati alle osservazioni che ciascuno dei casi esaminati ha generato.

Capitolo 5 – Risultati simulativi

Premessa

Il presente capitolo ha lo scopo di rispondere attraverso considerazioni basate su dati ad alcune questioni fondamentali:

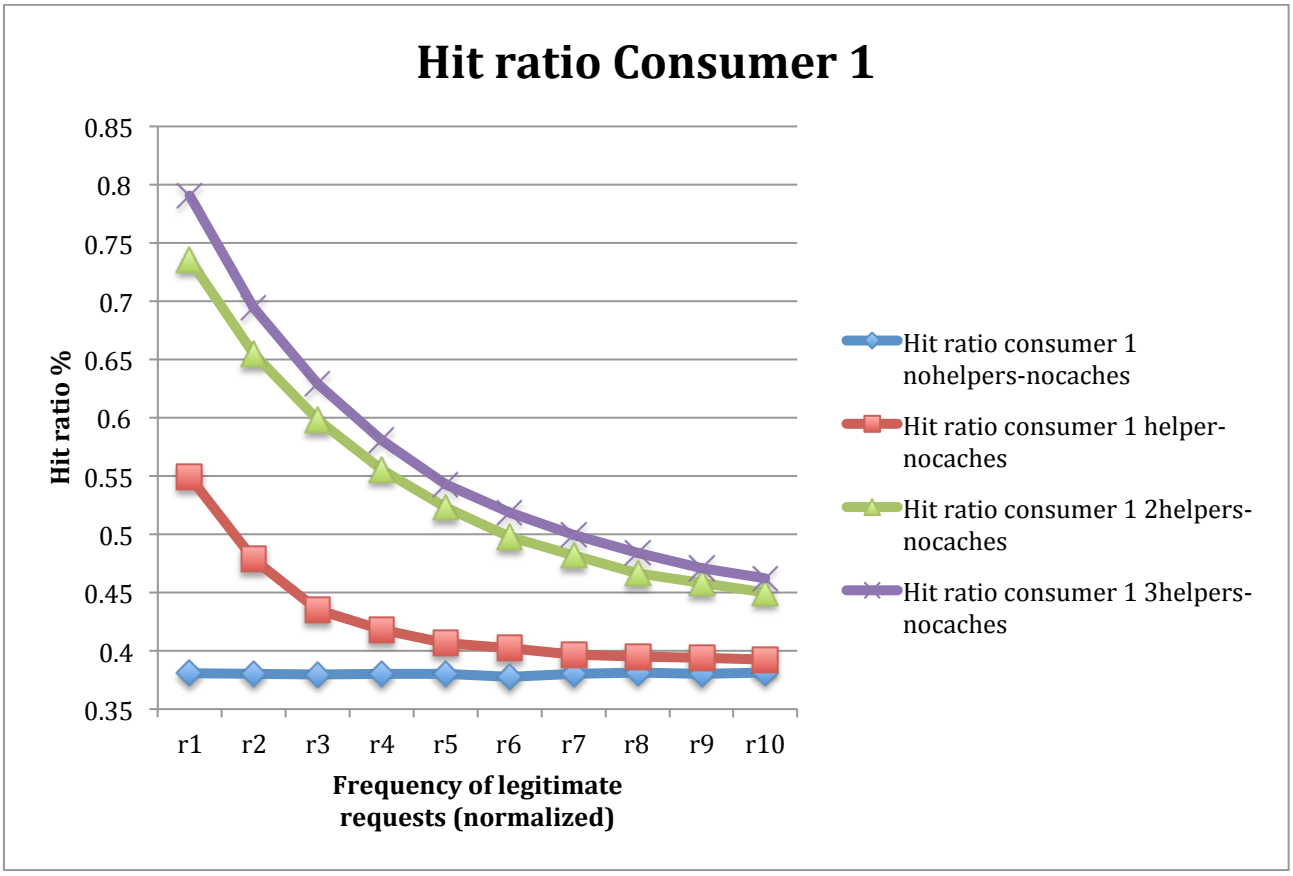
1. Date le ipotesi di partenza, può un attaccante ottenere un vantaggio dalla compromissione delle cache degli utenti?
2. Se sì, come si può quantificare questo vantaggio?
3. Quali sono le condizioni che maggiormente lo influenzano?
4. In che misura gli utenti compromessi sono disturbati dagli effetti collaterali?

Fissate topologia e metriche, sono state svolte simulazioni in ndnSIM con tutte le permutazioni possibili dei parametri definiti nel capitolo precedente.
Laddove siano state riscontrate mancanze nel simulatore, modifiche sono state apportate per farlo conformare maggiormente alle necessità del presente studio.
I dati raccolti sono stati poi ordinati, confrontati, messi in relazione ed infine interpretati.
L'esposizione in generale seguirà la seguente scaletta: verranno analizzati uno ad uno gli scenari, nell'ordine A1, A2, B1, B2.
Per ognuno di essi si procederà all'illustrazione di ogni metrica, partendo dall'hit ratio di Consumer 1 per i contenuti in offloading, consegne medie effettuate da Producer 1 per i contenuti in offloading, ritardo misurato da Consumer 1 per i contenuti in offloading, hit ratio degli helper per tutti i contenuti, ritardo degli helper per tutti i contenuti.
Per ciascuna metrica ogni grafico raccoglie i risultati delle simulazioni che in comune hanno le dimensioni delle cache dei nodi compromessi, partendo da 300, per passare a 600 e poi a 1000 contenuti.
Infine, in ogni grafico verranno presentati tutti e 4 gli andamenti risultanti dalle simulazioni svolte in configurazioni con nessuno, 1, 2, e 3 utenti compromessi ed al variare del grado di contesa da essi offerto, per ogni step da 1 a 10 quando possibile, o per i livelli 1, 5 e 10, che rappresentano rispettivamente basso, medio ed alto grado di contesa.

5.1 – Test con stesso grado di contesa fra tutti gli helper

5.1.1 - Scenario A1 (nessuna cache nei router, RTT di 30 ms)

Cominceremo il giro delle metriche con l'hit ratio di Consumer 1.

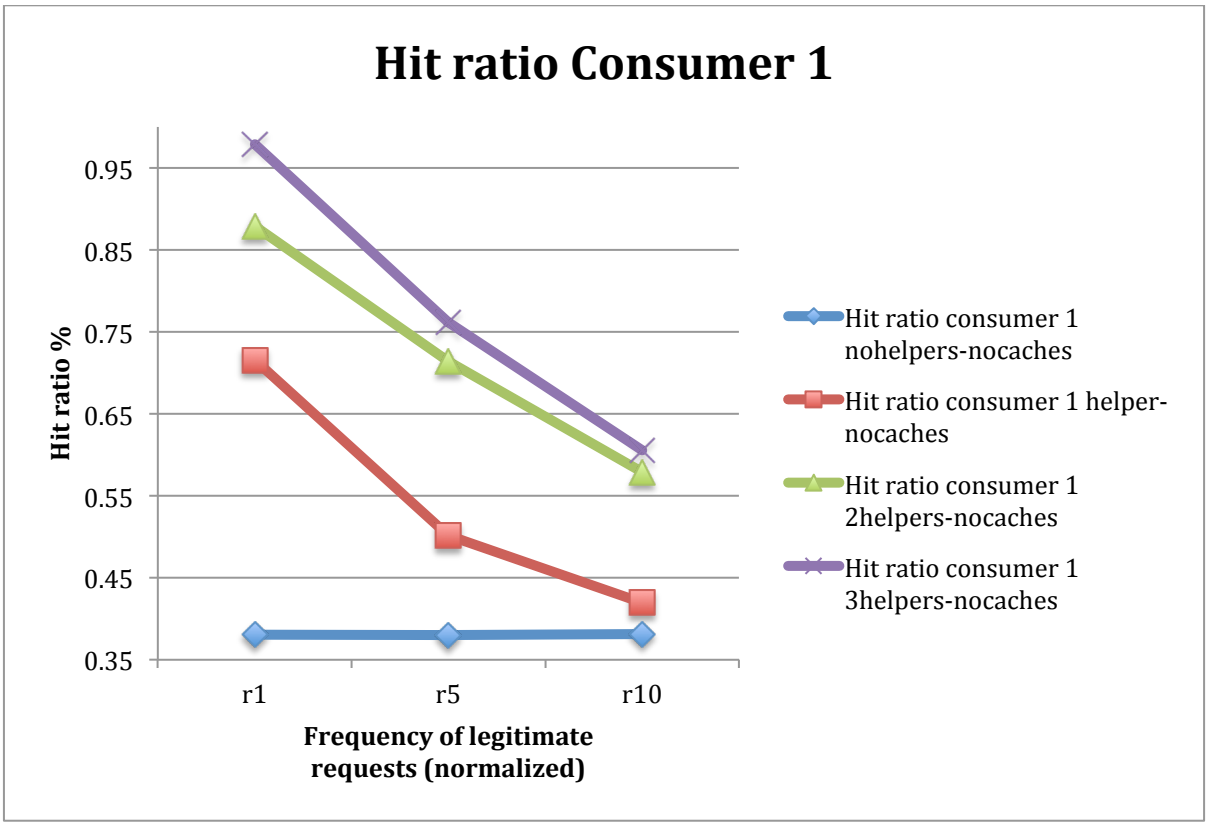


Graf. 5.1 Hit ratio di Consumer 1 – Scenario A1 – Cache da 300 contenuti - Stesso comportamento

Quelle mostrate in figura rappresentano simulazioni svolte con dimensione della cache pari a 300 contenuti per tutti gli helper.
Nei test che vengono mostrati in questa prima sezione si suppone che gli helper abbiano tra loro lo stesso comportamento in termini di rapporto tra le frequenze delle richieste, come descritto nei precedenti paragrafi.
Ogni curva rappresenta quindi l'hit ratio di Consumer 1 in situazioni di diverso numero di helper attivi.
Ci sono varie osservazioni da fare, la prima delle quali è che l'hit ratio di Consumer 1 senza attacco rimane costante sul valore di circa 38%.
Come era lecito aspettarsi infatti, questo valore non dipende dalla frequenza delle richieste degli altri utenti.
Successivamente si nota come effettivamente l'attacco produca risultati tangibili sotto questo punto di vista, portando l'hit ratio ad un valore molto elevato come il 79% nel caso migliore, ovvero con 3 helper che però offrono un basso grado di contestazione, con la richiesta di contenuti legittimi che pareggia quella di contenuti illeciti.

Altra osservazione di cui prendere nota è che tutte le curve hanno un rapido declino dopo il picco iniziale ed in generale l'effetto dell'attacco ha perso molta della sua incisività quando le cache divengono mediamente contestate, questo specialmente nel caso di singolo nodo compromesso.
Un ultimo punto di interesse riguarda gli scarsi vantaggi ottenibili, in questa configurazione, dall'aggiunta di un ulteriore helper dopo il secondo, mossa che porta solo marginali vantaggi.

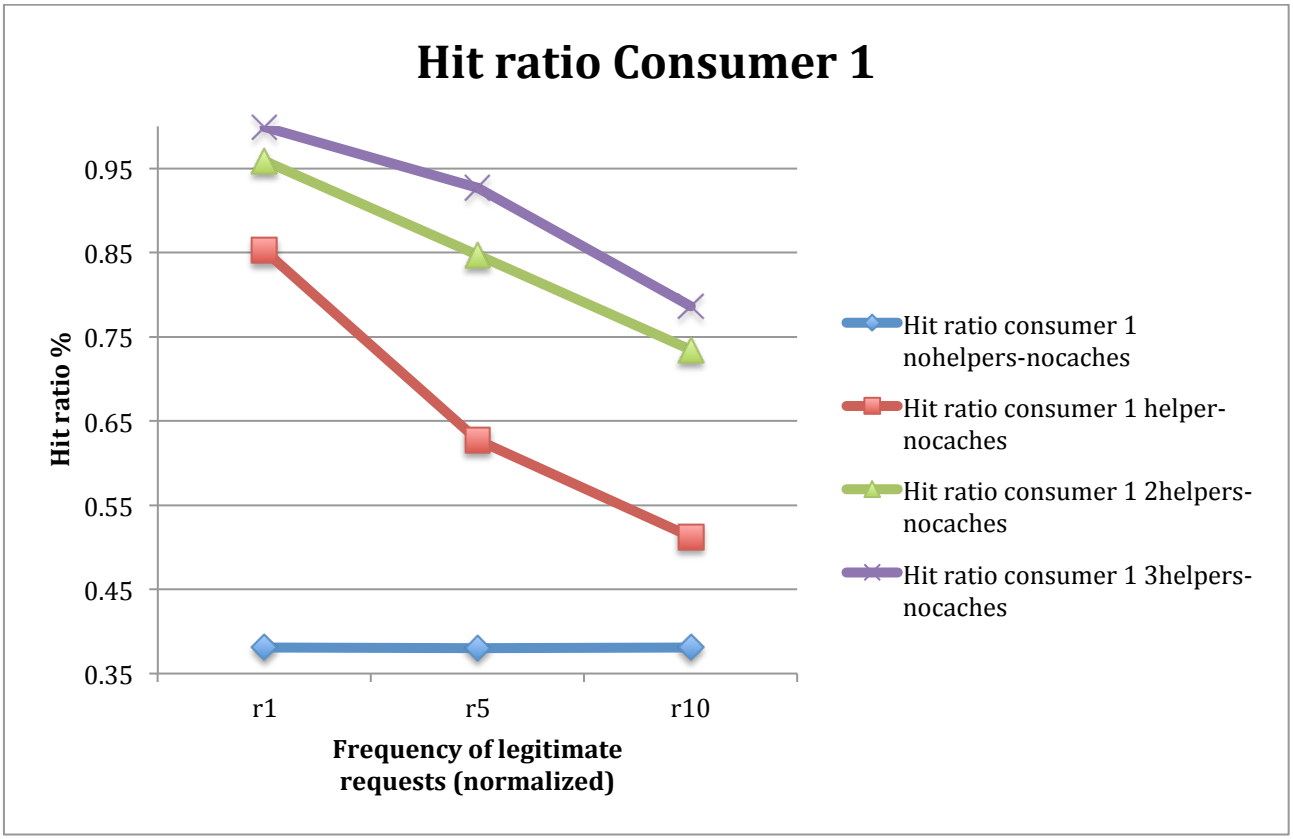
Per facilità di confronto verrà ora mostrato lo stesso grafico, ma con le cache degli helper da 600 contenuti.



Graf. 5.2 Hit ratio di Consumer 1 – Scenario A1 – Cache da 600 contenuti - Stesso comportamento

Come spiegato in precedenza, queste simulazioni sono state svolte solo per tre diversi rapporti di frequenza degli helper, tuttavia l'andamento della metrica è piuttosto chiaro anche senza mostrare tutti i punti intermedi.
In questa serie di curve ciò che si può notare è che c'è un deciso, ed atteso, aumento dell'hit ratio su tutta la linea.
Il picco massimo questa volta va addirittura a sfiorare il 100%, ma successivamente si avrà modo di vedere a quale prezzo.
In generale però non ci sono significative variazioni rispetto alle osservazioni effettuate in precedenza; per assicurare un buon hit ratio in ogni condizione di contestazione delle cache possibile, il compromesso migliore sembra consistere nella scelta di 2 helper.

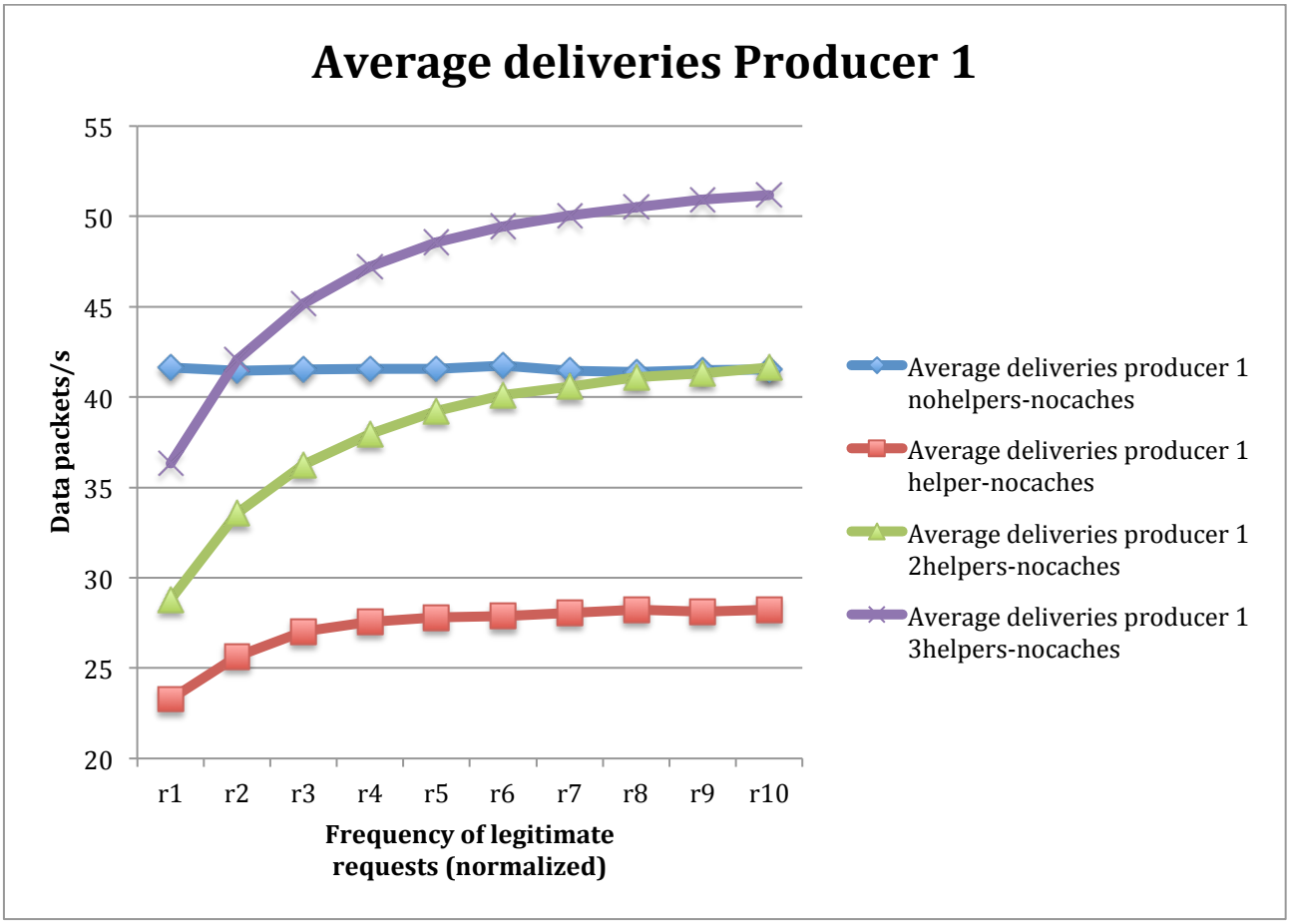
Vediamo ora molto brevemente la situazione per cache da 1000 contenuti.



Graf. 5.3 Hit ratio di Consumer 1 – Scenario A1 – Cache da 1000 contenuti - Stesso comportamento

È possibile osservare un nuovo aumento dell'hit ratio su tutta la linea, con valori che raggiungono vette molto elevate per tutti i casi e che, pur calando, rimangono consistentemente in regioni molto interessanti.
Anche nel peggiore dei casi, con un singolo helper che genera un volume di traffico legittimo 10 volte superiore a quello illegittimo, l'hit ratio che Consumer 1 riesce ad ottenere passa dal 38% al 51%, grazie all'ampia cache a disposizione di Helper 1.

Passiamo a questo punto alla valutazione della seconda metrica, la media dei contenuti consegnati da Producer 1 presi tra i 1000 più popolari, ovvero quelli in offloading, esaminando prima di tutto la configurazione con cache da 300 contenuti.

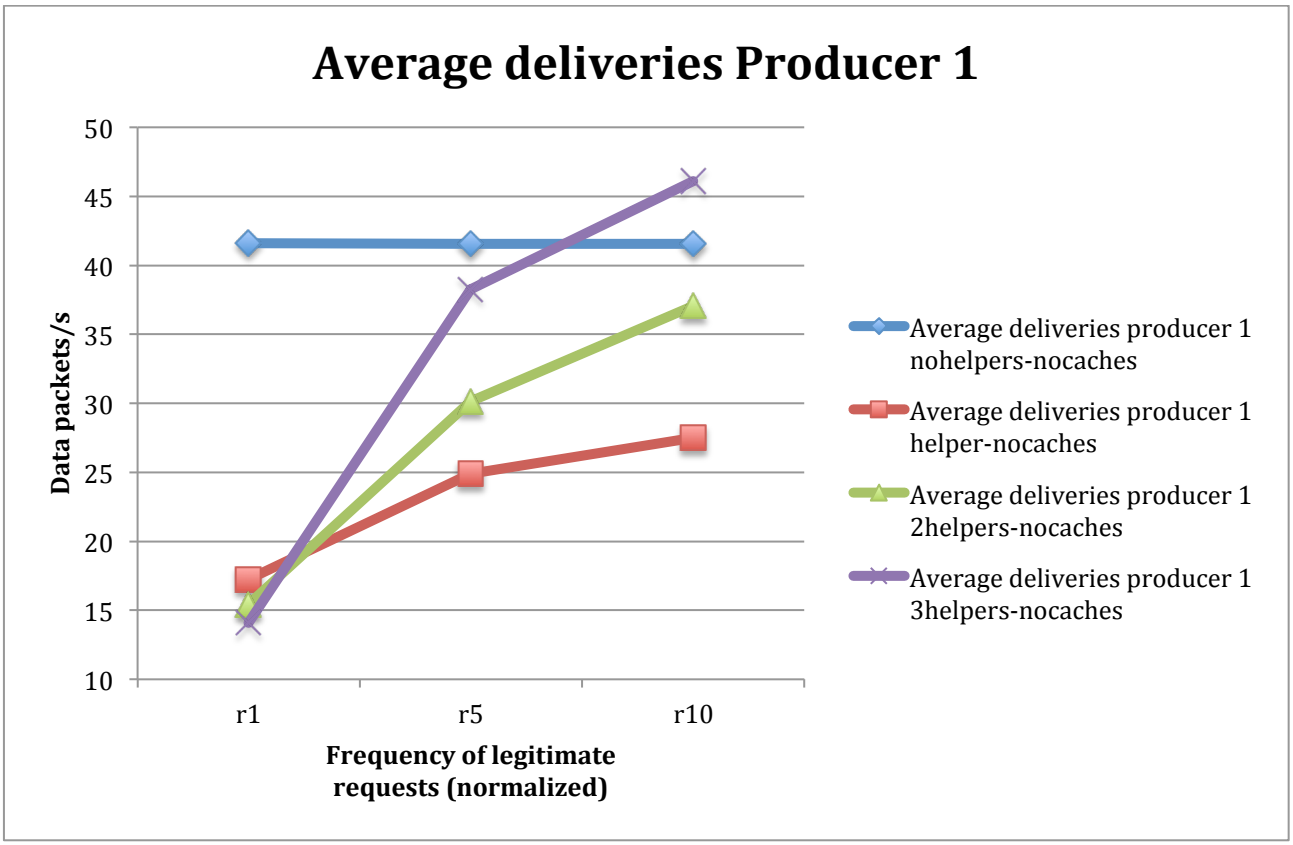


Graf. 5.4 Numero medio di consegne di Producer1 – Scenario A1 – Cache da 300 contenuti - Stesso comportamento

Nel caso di questa metrica, il comportamento desiderato è quello di distanziarsi il più possibile, verso il basso, dalla retta rappresentata in azzurro nel grafico soprastante, cioè quella rappresentante il numero di richieste soddisfatte dal produttore nel caso di base.
Ciò che è evidente è il prezzo che si deve pagare per i risultati visti precedentemente; con 2 helper la situazione può essere comunque giudicata conveniente, visto che allo stesso numero di consegne (41 al secondo), nel caso peggiore, corrisponde un hit ratio più elevato (anche se non di molto, dal 38% al 45%), mentre nei casi di bassa contesa occorre consegnare circa 29 pacchetti dati al secondo per un incremento dell'hit ratio molto significativo (dal 38% al 73%).
Passando ad esaminare il caso dei 3 helper, si vede come anche da questo punto di vista l'aggiunta di una cache causi più problemi che vantaggi, visto che il guadagno in termini di hit ratio rispetto alla configurazione con 2 helper è marginale in ogni situazione (in media è circa di 3 punti percentuali), ma le consegne da effettuare per il produttore malevolo sono decisamente superiori (in media intorno alle 9 in più al secondo).
L'aspetto invece da considerare sotto una nuova luce è la situazione creata dalla presenza di un singolo helper, che rappresenta il caso duale rispetto ai 2 helper, poiché con una sola vittima (e quindi bassi costi), il produttore può osservare una decisa diminuzione delle richieste che deve servire (circa 14 in media in meno), ma senza penalizzare, nemmeno nel caso peggiore, l'hit ratio di Consumer 1, metrica che, anzi, può migliorare se Helper 1 non offre molta competizione.

Si cominciano ad intravedere le prime considerazioni tattiche da effettuare per l'attaccante, basate sugli obiettivi stabiliti dal produttore malevolo, un argomento che verrà maggiormente sviluppato in seguito, quando verranno descritti i risultati delle simulazioni con differente comportamento da parte degli helper.

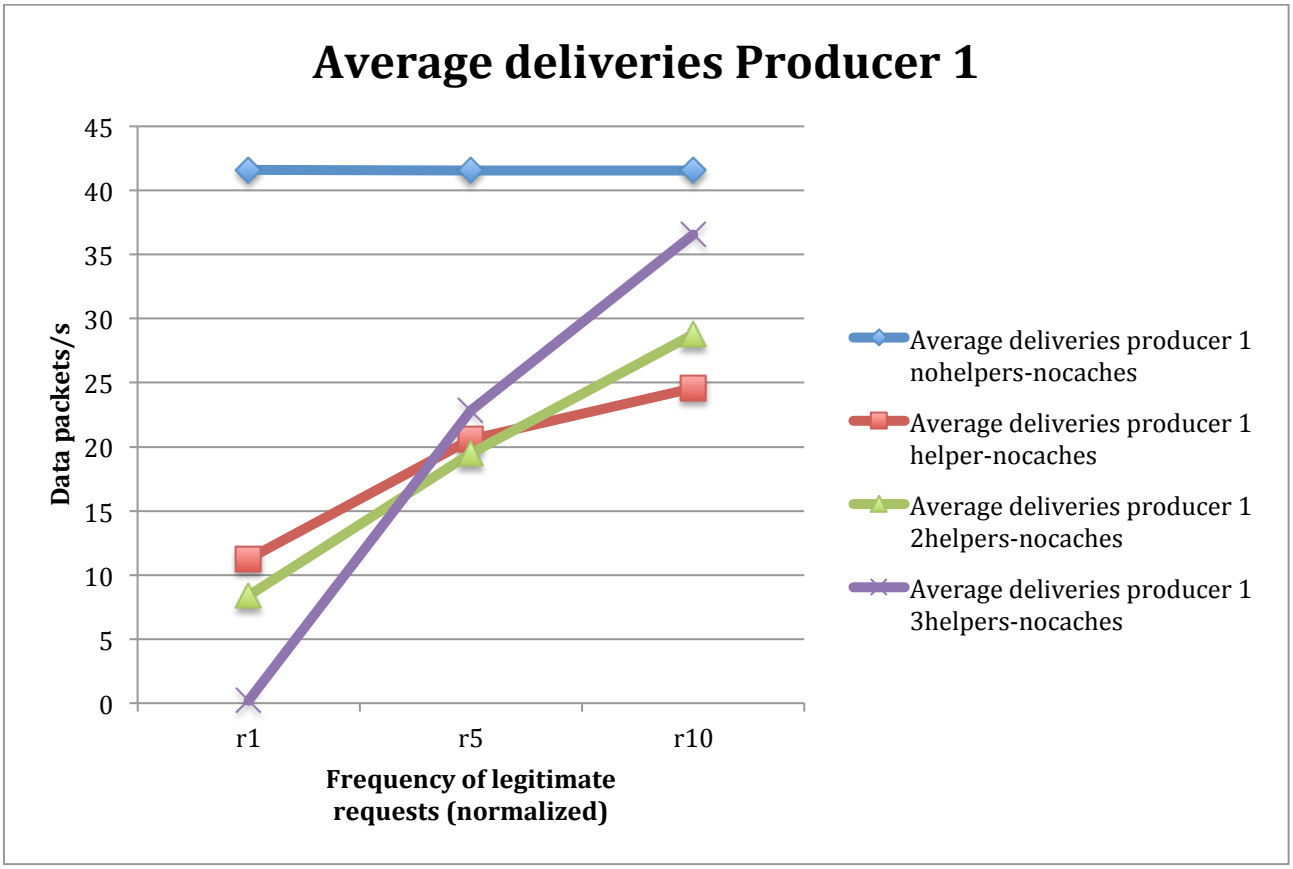
Come in precedenza, verrà ora analizzato il caso di cache da 600 contenuti.



Graf. 5.5 Numero medio di consegne di Producer1 – Scenario A1 – Cache da 600 contenuti - Stesso comportamento

Con cache di dimensioni maggiori a disposizione, la situazione assume una connotazione più rosea, grazie all'abbassamento generale della metrica, anche se le indicazioni di massima rimangono le stesse emerse dal grafico precedente.
Questa volta però si può vedere un vantaggio sulla carta più consistente nel passaggio da 2 a 3 helper nel caso di minima contesa, poiché le consegne rimangono le stesse, ma l'hit ratio cresce dall'88% al 98%.
Risulta tuttavia opinabile la reale utilità di una tale incremento se già la base di partenza è di poco inferiore al 90%, considerando poi che se il grado di contesa dovesse aumentare, la forbice tra le due configurazioni si allargherebbe molto in fretta ed in maniera consistente (per i gradi di contesa medio ed alto, la differenza si attesta a 8 e 9 consegne).
Rimane valido anche il discorso affrontato in precedenza per il singolo helper, anche se in questo caso le riduzioni osservate nelle consegne sono di intensità minore rispetto alle simulazioni con 2 e 3 helper, tranne che per il caso di minore contesa, in cui questa metrica passa da 23 a 17.

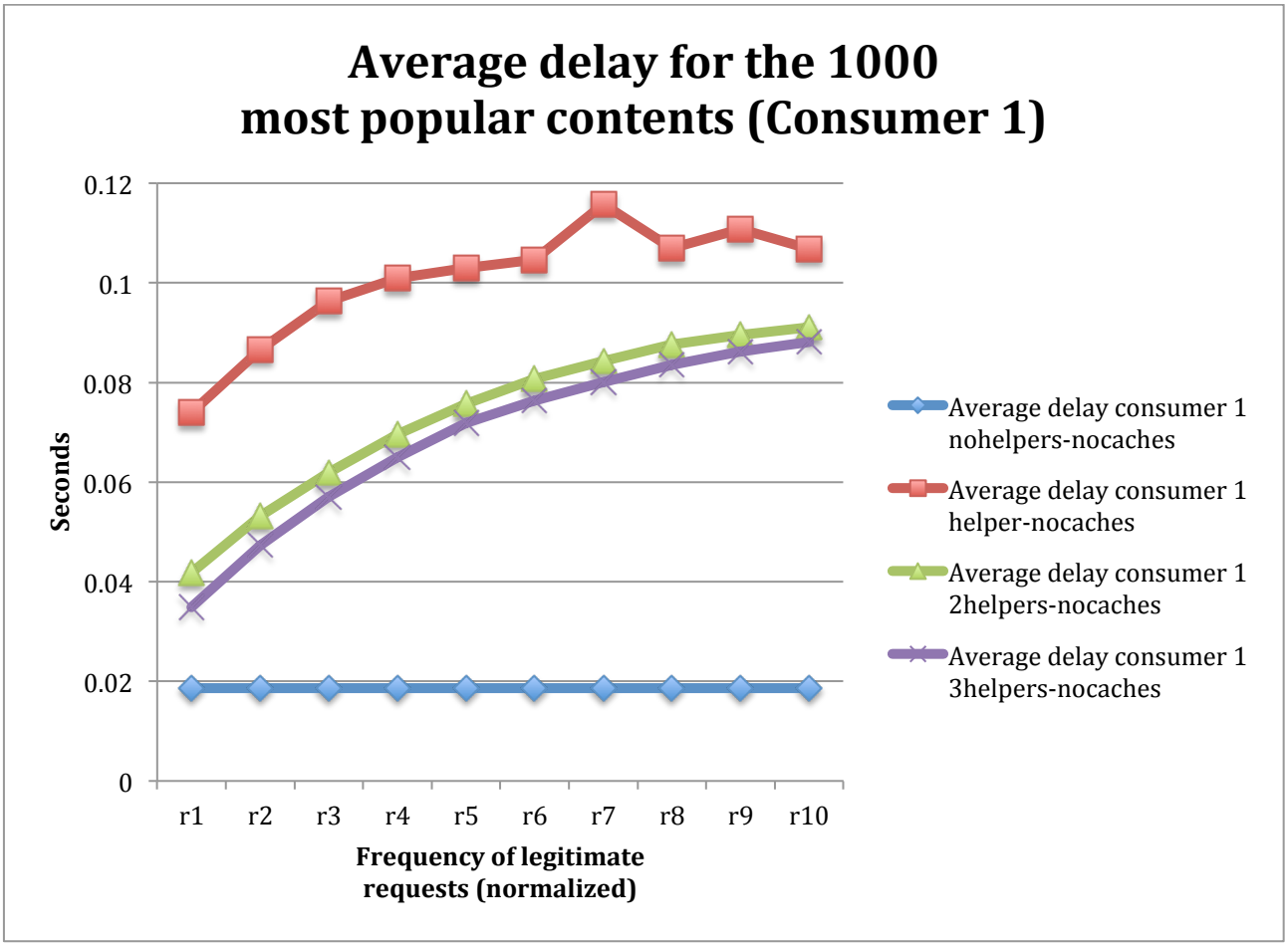
Come nel caso precedente, per ultimo vedremo quello che riguarda le cache da 1000 contenuti.



Graf. 5.6 Numero medio di consegne di Producer1 – Scenario A1 – Cache da 1000 contenuti - Stesso comportamento

Con cache da 1000 contenuti il carico sul produttore malevolo viene notevolmente sollevato, specialmente se gli utenti offrono poco traffico, rendendo, senza considerare il costo di compromissione di una cache aggiuntiva, maggiormente conveniente l'utilizzo di 2 o addirittura di 3 helper, quest'ultimo caso se si rimane nella regione di basso/medio grado di contesa.
Osservando le cifre relative al solo caso peggiore (massimo grado di contesa) si vede che con la "collaborazione" di 2 helper rispetto ad 1, Consumer 1 riesce a godere di un hit ratio del 73% contro il 51%, con un incremento delle consegne da 24 a 28, cioè del 16.7%.
Nel caso di 3 helper l'hit ratio viene sì portato dal 51% al 78%, ma le consegne vanno da 24 a 36, un aumento del 50%.
Dato di cui prendere nota è il numero di consegne di Producer 1 nel caso migliore di 3 helper che offrono poca contesa, che raggiunge praticamente lo 0, sgravandolo quasi del tutto del compito delle consegne al di là dell'iniziale disseminazione.

È ora arrivato il momento di valutare i risultati delle simulazioni per ciò che concerne il ritardo dei contenuti in offloading visto da Consumer 1.

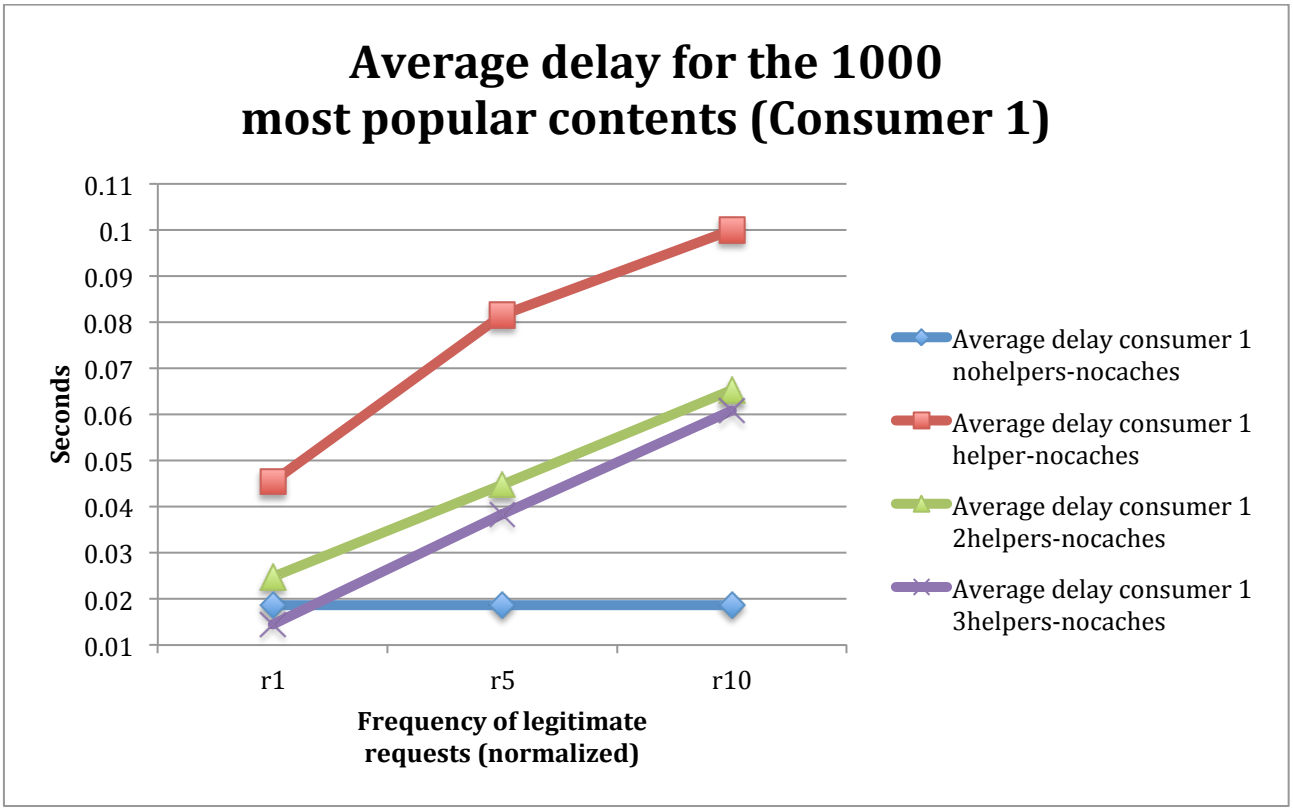


Graf. 5.7 Ritardo medio di Consumer 1 – Scenario A1 – Cache da 300 contenuti - Stesso comportamento

Non c'è modo di girare attorno al fatto che, per questa configurazione di topologia e parametri, queste figure non abbiano un bell'aspetto.
Ciò che si può vedere è che il ritardo rimane costantemente molto basso in assenza di attacco, risultando in media pari a circa 18 ms, mentre per ogni istanza in cui l'attacco è in corso si va verso un peggioramento delle prestazioni che si materializza in un raddoppio del ritardo nel caso migliore ed in suo aumento fino 6 volte il suo valore originario.
In termini di valori assoluti, quando sono compromessi 2 o 3 utenti la situazione è migliore rispetto al singolo helper, ma il loro andamento rimane comunque punitivo per il produttore malevolo, perché il passaggio da basso a medio grado di contesa produce un raddoppio rispetto al ritardo di partenza di circa 40 ms e l'aggiunta di un helper, da 2 a 3, non produce miglioramenti significativi.
Il principale fattore in questi valori così elevati è senza dubbio da ricercare nella strategia di routing prescelta; essa avvantaggia il produttore malevolo quando si tratta di richieste medie, visto che molte di queste vengono soddisfatte dagli helper e mai ricevute dal produttore, ma pone anche uno svantaggio consistente visto che le richieste che non vengono soddisfatte dalle cache della prima linea (Consumer 1, Router 1 e gli helper), devono attendere di essere ritrasmesse e soddisfatte dal produttore, il che spiega gli alti ritardi visti.
La politica di routing che porrebbe rimedio alla situazione prevede di inviare l'interesse ricevuto da una face a tutte le altre contemporaneamente.

Se così fosse però il produttore malevolo sarebbe nuovamente costretto ad inviare la risposta e quindi il contenuto, vanificando i vantaggi in termini di consegne medie apprezzati in precedenza.
Una soluzione a questo problema, qui solo teorizzata e non implementata, potrebbe essere per il produttore malevolo di rispondere agli interest solo in maniera statistica e cioè con una frequenza bassa e determinata dal calcolo della probabilità che l'interest provenga da uno degli helper; calcolo facilitato se gli helper emettono richieste di contenuti in maniera deterministica e decisa dal produttore.
Tutto ciò vale dal momento che egli non può distinguere gli interest degli helper da quelli degli utenti normale, a meno che non riesca a marcare i primi in qualche modo.
Qualche miglioramento potrebbe essere anche ottenuto riducendo la quantità di contenuti in offloading, visto che in tal modo le richieste che ciascun helper emette coprirebbero un insieme inferiore di contenuti, con maggiori possibilità di tenerli in cache.
Vedremo inoltre, osservando i risultati delle simulazioni con dimensioni delle cache più elevate, che ci sono alcuni casi in cui questa metrica viene in realtà resa pari, se non inferiore, rispetto al caso di base.

A tal proposito vengono ora mostrate le curve ottenute con le cache da 600 contenuti.

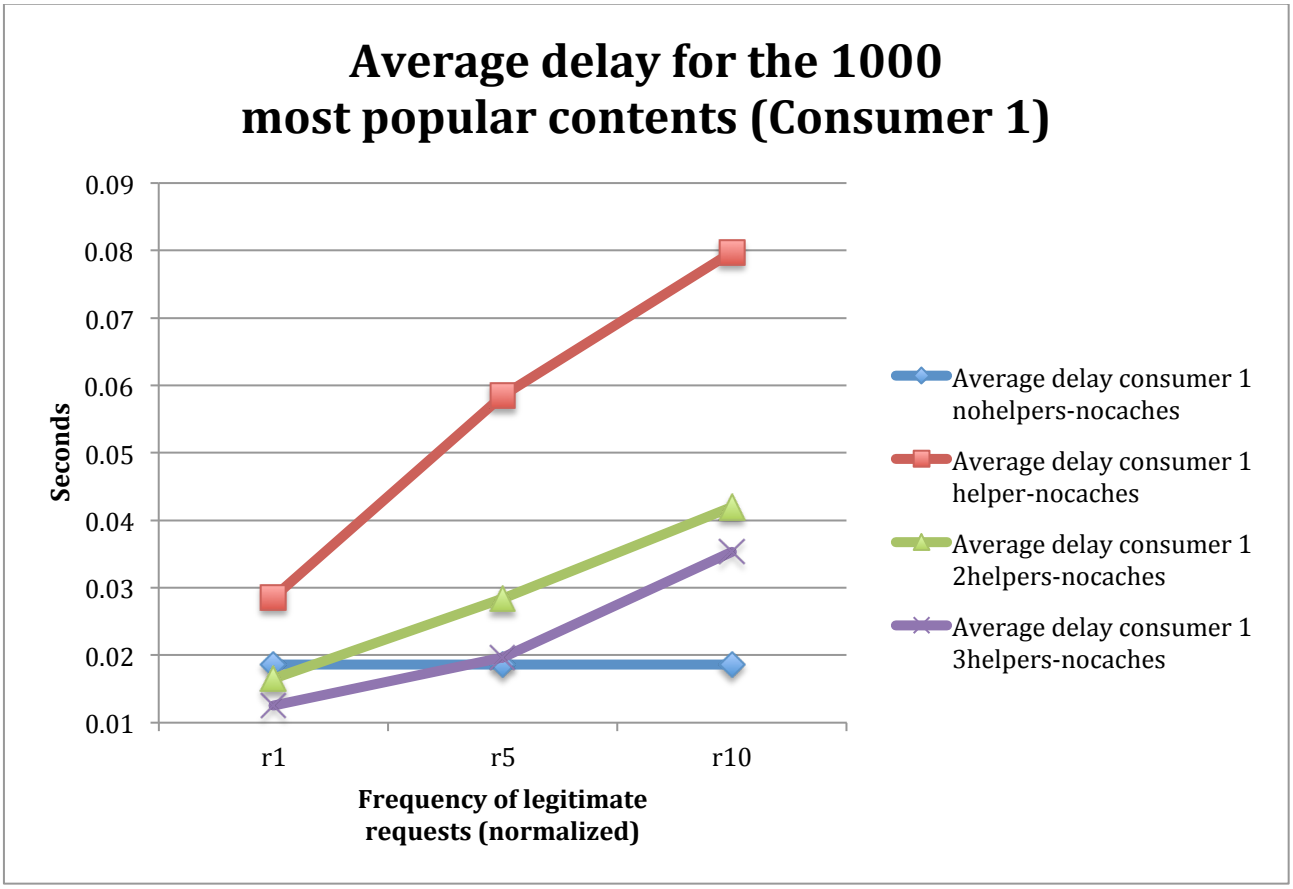


Graf. 5.8 Ritardo medio di Consumer 1 – Scenario A1 – Cache da 600 contenuti - Stesso comportamento

Sfruttando la possibilità di memorizzare più contenuti nelle cache degli utenti compromessi, l'attaccante riesce a ridurre notevolmente il ritardo visto da Consumer 1 rispetto a quanto visto in precedenza.
In particolare in presenza di cache poco contestate, la metrica scende di 29 ms, 17 ms e 21 ms per attacchi con singolo helper, 2 e 3 helper rispettivamente riferendosi al caso precedente.

Nel caso poi di 3 helper, il ritardo nella condizione appena citata registra 14 ms di ritardo, risultando quindi inferiore al caso senza attacco, che si colloca a 18 ms, mentre con 2 helper il valore ottenuto è di 24 ms.
Le prestazioni poi peggiorano rapidamente una volta che l'intensità della domanda per contenuti legittimi e quindi il grado di contesa della cache cresce, velocizzando l'epurazione dei contenuti illeciti.
Le considerazioni fatte per la situazione precedente rimangono ovviamente valide anche per questa; si è osservato però che l'aumento di dimensione delle cache, il che nella pratica significa trovare utenti da compromettere con maggiore spazio a disposizione, è una strategia efficace per combattere il problema del ritardo elevato.

Si osserva infine il comportamento del sistema quando le cache possono memorizzare fino a 1000 contenuti.

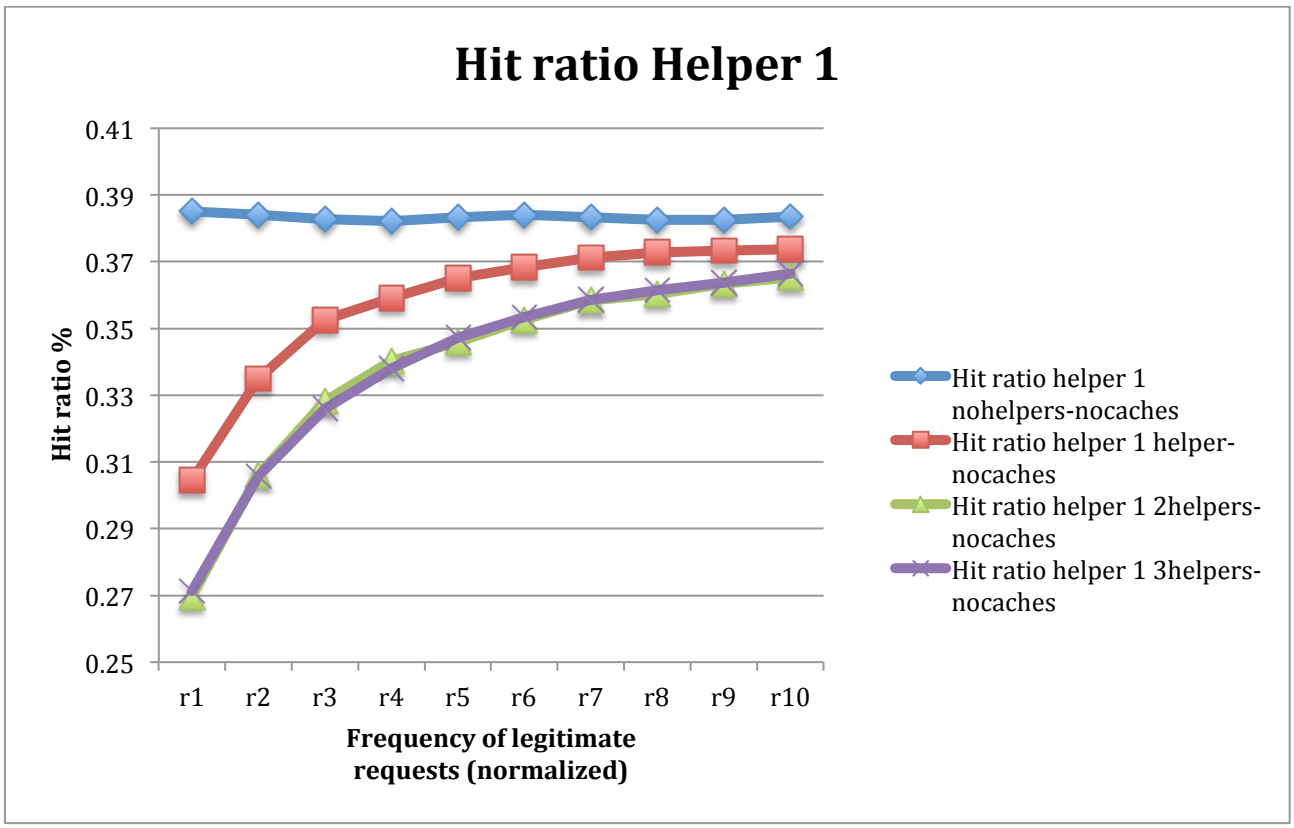


Graf. 5.9 Ritardo medio di Consumer 1 – Scenario A1 – Cache da 1000 contenuti - Stesso comportamento

Possiamo osservare che il ritardo continua ad abbassarsi in maniera simile a quanto fatto nel passaggio tra le simulazioni con cache da 300 e da 600 contenuti, mantenendo una certa linearità nella sua riduzione all'aumentare della dimensione delle cache, almeno considerando un medio/alto grado di contesa.
Con cache di estensione generosa come questa diventa appetibile anche un attacco con un singolo helper che generi poco traffico legittimo; il ritardo in questa condizione è pari a 28 ms, appena 10 in più rispetto alla situazione di base.

Dopo avere esaminato e discusso approfonditamente le metriche che vanno a diretto vantaggio del produttore malevolo e dei suoi utenti, è necessario affrontare il discorso del danno procurato a chi è vittima suo malgrado dell'attacco.
Come già menzionato in precedenza, se da una parte l'attaccante vuole avvantaggiarsi delle cache degli utenti, dall'altra ha anche la necessità di rendere la probabilità che questa mossa venga scoperta minore possibile.
Procedendo come per le altre metriche vediamo quanto l'hit ratio ed il ritardo dell'utente medio vengono danneggiati.
Si prendono in considerazione le metriche solo riferite ad Helper 1, poiché a meno di lievi differenze, il loro andamento è lo stesso di quelle proprie degli altri helper.

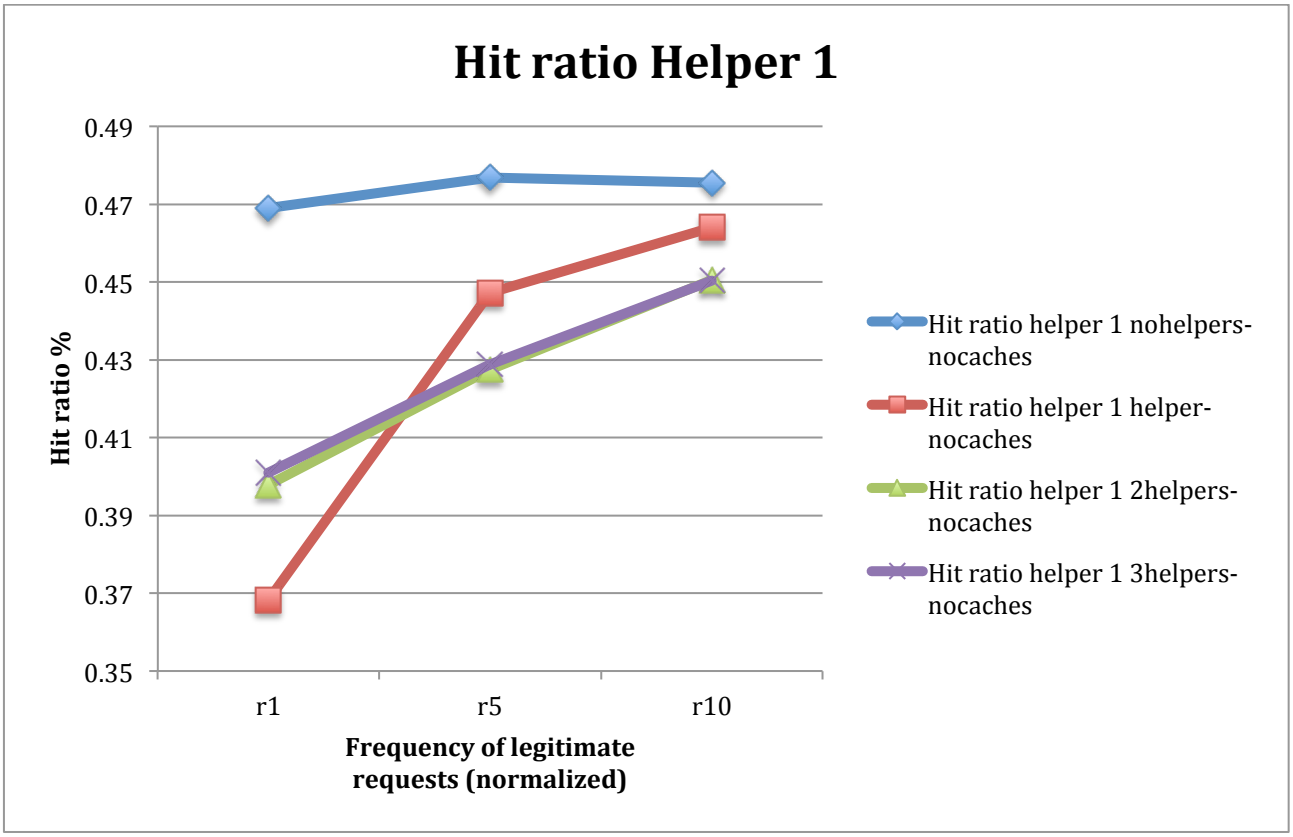
Si inizia dalle cache da 300 contenuti.



Graf. 5.10 Hit ratio di Helper 1 – Scenario A1 – Cache da 300 contenuti - Stesso comportamento

La prima cosa da notare è il valore che assume questa metrica in condizioni normali, pari al 38%.
Se il grado di contesa è basso questa percentuale scende sensibilmente, fino al 31% con un singolo helper e 27% con 2 e 3 helper, però risale velocemente al crescere della frequenza di richieste legittime, per attestarsi a livelli intorno al 37% e 35% già per un grado medio di contesa; date le differenze davvero minime col caso di base, risulta difficile pensare che questo decremento possa causare problemi prestazionali agli utenti.

Esamineremo ora il sistema con cache da 600 contenuti.



Graf. 5.11 Hit ratio di Helper 1 – Scenario A1 – Cache da 600 contenuti - Stesso comportamento

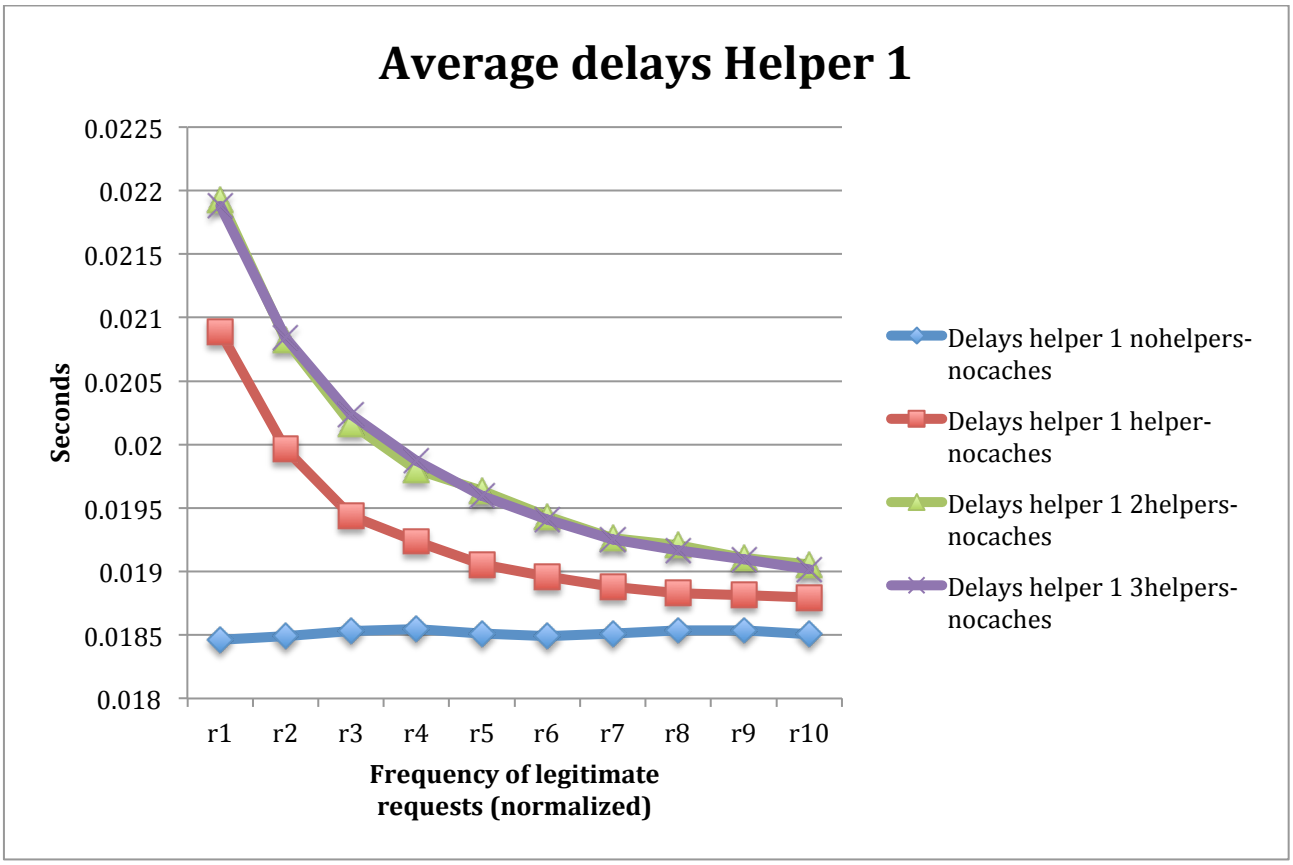
Il pattern si ripete anche con questa configurazione delle cache; si registra un brusco calo rispetto all'assenza di attacco, che raggiunge i 10 punti percentuali quando un solo nodo utente è compromesso, ma all'aumentare del grado di contesa lo strappo viene ricucito, per raggiungere sostanzialmente la parità al massimo valore assegnato alla frequenza delle richieste legittime.
Il punto di differenza in questo caso risiede nel fatto che questa volta il calo più elevato viene osservato con un solo helper attivo, mentre prima questo avveniva con 2 o 3 helper.
La spiegazione risiede nel fatto che un solo helper deve richiedere più contenuti rispetto a quando può dividere il carico con un altro helper; dato il basso grado di contesa offerto dalla vittima, la cache ha una proporzione maggiore di contenuti illegittimi quando c'è un solo helper piuttosto che quando il carico viene suddiviso tra più di essi.
Data l'estrema vicinanza dell'hit ratio tra le casistiche di attacco e quella di non attacco quando il grado di contesa è medio/alto, le conclusioni restano quelle tratte nel caso precedente, vale a dire che l'attacco, sotto le condizioni citate, riesca nell'obiettivo di non peggiorare significativamente le prestazioni degli utenti.

Infine non viene mostrato il grafico per la configurazione dello scenario che prevede cache da 1000 contenuti, dal momento che sarebbe solo una ripetizione dei precedenti due, con la differenza dei valori più elevati in generale data la maggiore disponibilità di spazio nelle cache.
Si segnala solo che nel caso peggiore, cioè basso grado di contesa, la decrescita dell'hit ratio che si verifica risulta maggiore di quello osservato nei due casi precedenti; qui infatti l'hit

ratio nel caso di base si attesta al 54%, ma quando un helper è compromesso esso cala fino al 40%.

La metrica che è rimasta da esaminare per quanto riguarda gli helper è il ritardo nella ricezione dei contenuti da essi legittimamente richiesti.
Come per l'hit ratio, l'attaccante desidera che il ritardo sia affetto il meno possibile dall'attacco e quindi che permanga ai livelli misurati in assenza di attacco.
Sempre come per le precedenti osservazioni, qui si mostrerà il ritardo per Helper 1, viste le minime differenze riscontrate nei comportamenti tra i vari helper.

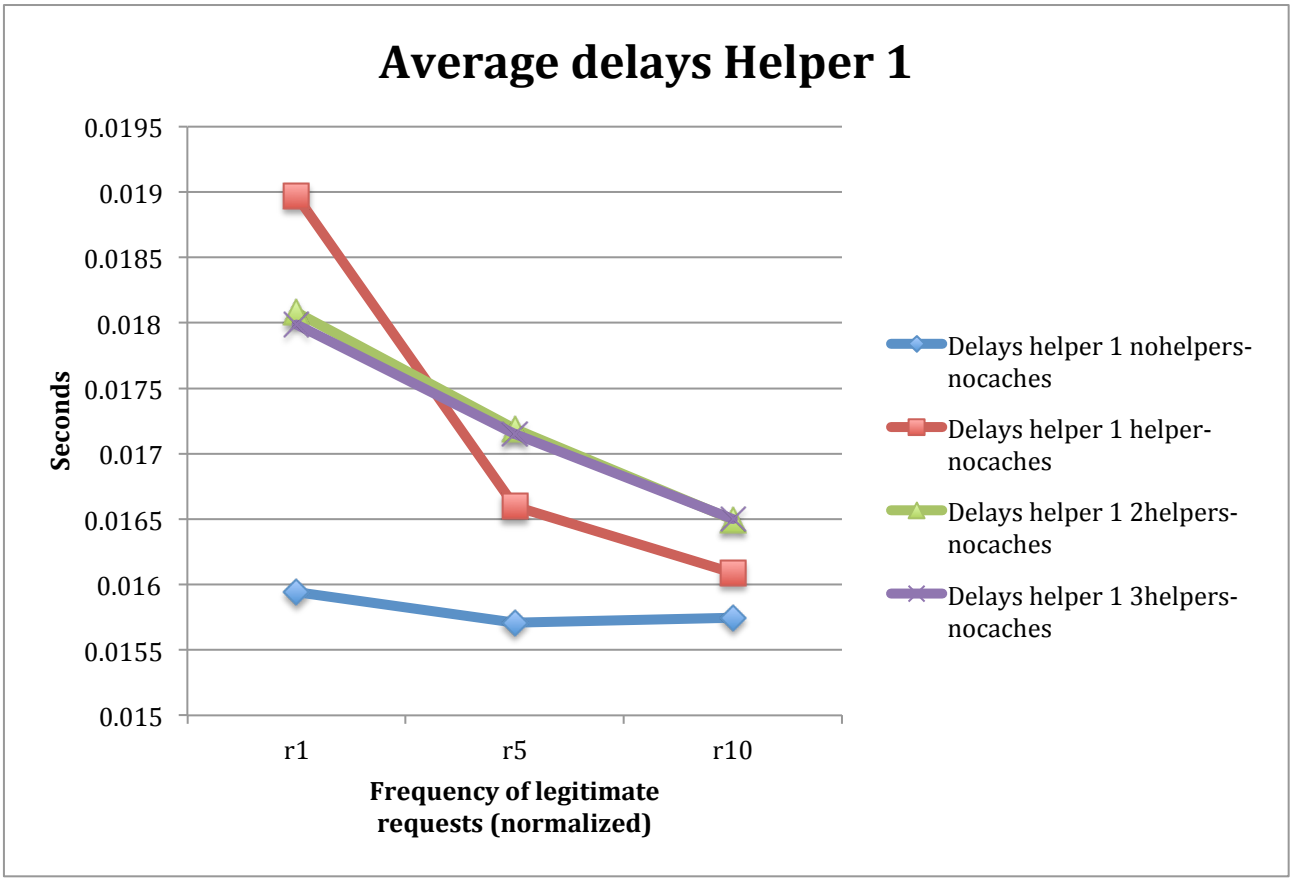
Vediamo i risultati partendo come al solito dalle cache da 300 contenuti.



Graf. 5.12 Ritardo medio di Helper 1 – Scenario A1 – Cache da 300 contenuti - Stesso comportamento

Osservando i valori assoluti, il loro incremento non risulta significativo, anche se sicuramente da tenere in considerazione.
Questo incremento del ritardo è, com'era lecito attendersi, inversamente proporzionale rispetto al grado di contesa della cache.
Ciò di cui prendere nota, tuttavia, è l'incremento percentuale, in modo da effettuare un confronto con la situazione negli scenari B.
Prendendo in considerazione le percentuali, si ha un aumento dal valore di base del 13% con un singolo helper e del 19% in presenza di 2 o più helper, nei casi peggiori, ma con una media degli aumenti del 4 e 7% rispettivamente.
Visti in percentuale i picchi di tali incrementi non sono da sottovalutare nei casi di ritardi di base più elevati.

Vedremo nel seguito se tali valori vengono confermati dalle simulazioni svolte negli scenari B.
Vediamo ora la situazione del ritardo con cache da 600 contenuti.



Graf. 5.13 Ritardo medio di Helper 1 – Scenario A1 – Cache da 600 contenuti - Stesso comportamento

Il ritardo sperimentato senza attacco è chiaramente più contenuto rispetto al caso precedente, data la più elevata dimensione delle cache.
Notiamo rispetto a prima, e come per l'hit ratio, che se la cache è poco contesa, un helper è sfavorito se da solo, al contrario di quando può dividere il carico con un secondo helper e la spiegazione data nel paragrafo dell'hit ratio rimane valida anche per questa metrica.
Come valori in percentuale si osservano gli stessi aumenti di picco del caso precedente, anche se invertiti, come detto poco sopra, mentre i valori medi sono più elevati e pari al 9% in entrambi i casi.

Per le cache da 1000 contenuti non viene riportato il grafico, ma solo i valori percentuali per il successivo confronto, dal momento che le curve sono pressoché identiche ed i valori si abbassano leggermente (il ritardo base passa da circa 16 ms di prima a circa 13.5 ms).
L'aumento massimo registrato in termini percentuali è pari al 28% tra il ritardo di base e quello con un singolo helper in condizioni di bassa contesa e pari al 10% tra quello di base ed i due helper.
I valori medi di aumento sono rispettivamente del 13% e del 9%.

L'ultima metrica da valutare per lo scenario A1 è il tempo medio di intercorrenza fra 2 interest successivi per lo stesso contenuto inviati da Consumer 1 ed instradati verso il produttore malevolo.
Questi risultati sono riassunti nella tabella sottostante (Tab. 4.4):

	Cache size = 300 contents			Cache size = 600 contents			Cache size = 1000 contents		
	R1	R5	R10	R1	R5	R10	R1	R5	R10
No helpers	43 s	43 s	43 s	43 s	43 s	43 s	43 s	43 s	43 s
1 helper	82 s	77 s	77s	230 s	78 s	78 s	466 s	143 s	78 s
2 helpers	100 s	80 s	78 s	999 s	93 s	81 s	1195 s	606 s	100 s
3 helpers	131 s	86 s	81 s	1687 s	114 s	90 s	6704 s	692 s	123 s

Tab. 5.1 Tempo medio di permanenza in rete dei contenuti – Scenario A1

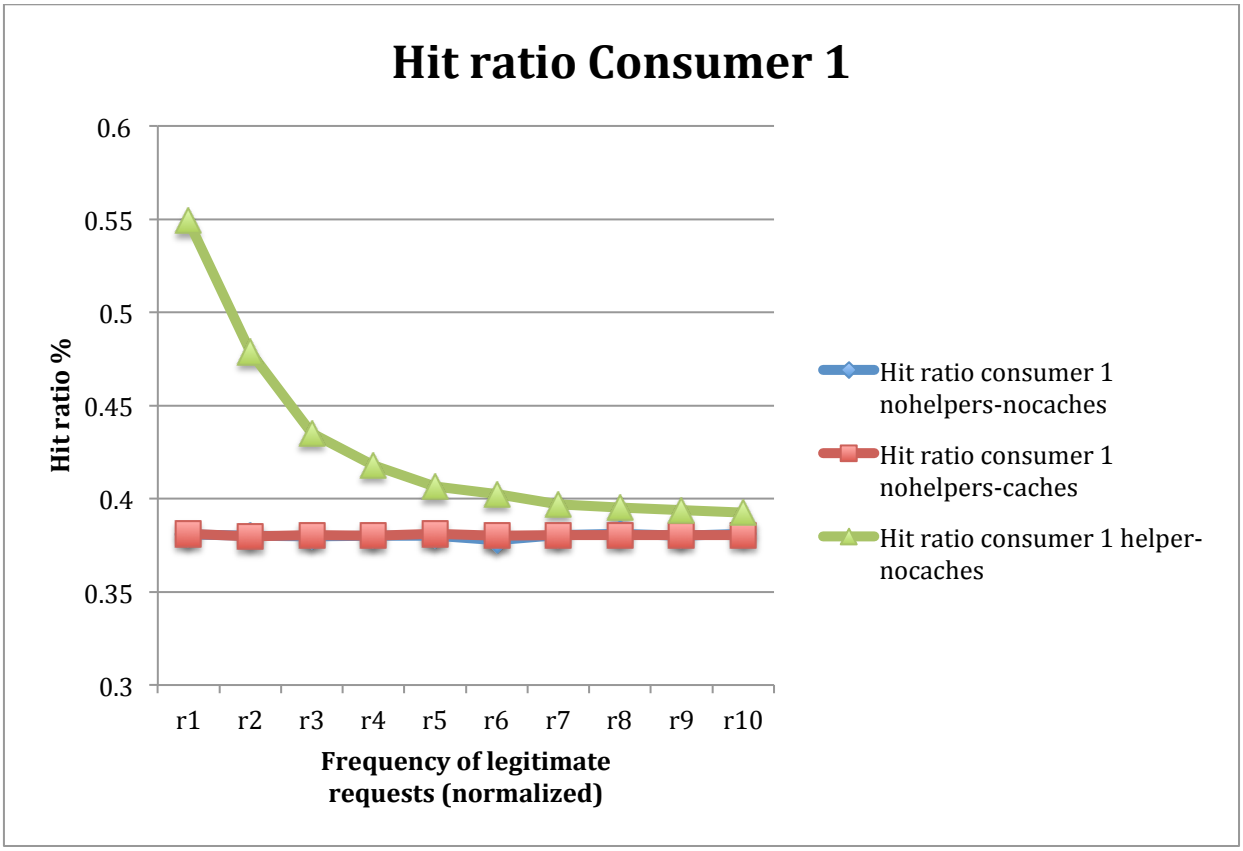
Nel caso di base la statistica non è influenzata né dalla frequenza di richiesta, né dalle dimensioni delle cache degli helper (non compromessi) e si attesta al valore medio fra tutti i contenuti di 43 s.
Nel peggiore dei casi (singolo helper, alto grado di contesa) il produttore malevolo può ottenere un vantaggio in termini di prolungamento di questo intervallo pari al 76.7% (da 43 a 77 secondi), ovvero piuttosto consistente.
In termini assoluti i migliori benefici si possono trarre con le cache di maggiori dimensioni, meno contese e quando il carico viene condiviso da più helper.
L'aumento di numero di nodi compromessi ha senza dubbio un effetto già notevole, visto che il risultato ottenuto senza helper raddoppia o va molto vicino a raddoppiare in quasi tutti i casi, ma è la combinazione di questo fattore con dimensioni delle cache più generose che produce i risultati più elevati, con durate medie di permanenza in rete dell'ordine anche di decine di minuti.
Al contempo è anche molto rilevante l'effetto che il passaggio da basso a medio grado di contesa opera sui risultati; basta una frequenza di emissione di richieste legittime 5 volte più elevata del valore di base per ridurre di un fattore 10 il valore medio rilevato nei casi di 2 helper con cache da 300 contenuti e di 2 e 3 helper con cache da 1000 contenuti
Si può quindi trarre la conclusione che la tecnica di attacco illustrata nel presente lavoro ha sicuramente grande efficacia nel garantire un miglioramento del tempo di permanenza in rete dei contenuti, in alcuni casi con dei risultati molto significativi.

5.1.2 - Conclusioni sullo scenario A1

I dati risultanti dalle simulazioni svolte riportati in questa sezione provano le ipotesi di partenza, ovvero la possibilità pratica per un produttore malevolo di condurre un attacco al fine di migliorare l'esperienza d'uso dei propri utenti o di risparmiare sul numero di consegne effettuate.
La chiave del discorso sta in quella lettera "o"; non è possibile, almeno nelle configurazioni viste finora, ottenere benefici significativi in tutti gli aspetti allo stesso momento.
Per questo il produttore deve prima di tutto scegliere il suo interesse maggiore, che da un lato consiste nel miglioramento dell'esperienza d'uso dei propri utenti, dall'altro nel risparmio in termini di consegne.
Un produttore che sia interessato alla soluzione di minimo costo in grado di diminuire il numero di consegne da effettuare si può accontentare di un singolo helper anche con una disponibilità ridotta di spazio nella cache; questa configurazione garantisce un calo delle consegne che può andare dal 31 al 43%, senza danneggiare troppo severamente l'hit ratio ed il ritardo medio di Helper 1 e lasciando intatto se non migliorato l'hit ratio di Consumer 1.
Dall'altra parte, chi volesse migliorare le prestazioni viste da Consumer 1 dovrebbe continuare a contribuire con un certo volume di consegne (anche aumentarlo rispetto al normale) ed in questo caso ci sono numerose soluzioni che possono essere adottate, anche se quella che pare offrire il migliore bilanciamento prevede 2 helper, con cache di medie dimensioni ed un basso o medio grado di contesa.
Con questa configurazione, l'hit ratio di Consumer 1 verrebbe notevolmente aumentato e anche le consegne subirebbero un effetto positivo, con una diminuzione del 26 e 63% nei casi di media e bassa contesa rispettivamente; a questo si aggiunge un raddoppio ed una triplicazione del tempo di permanenza in rete dei contenuti sempre per media e bassa contesa.
Inoltre gli effetti collaterali causati alle prestazioni delle vittime sono sì presenti, ma quanto essi siano riscontrabili nella pratica rimane dubbio.
Il vero problema emerso in questa sezione riguarda il ritardo sperimentato da Consumer 1 durante l'attacco; non solo quest'ultimo è in grado di migliorare la metrica in oggetto solo in pochi casi, ma spesso essa è piuttosto danneggiata.
Le uniche configurazioni che riescono a contenere questo effetto nefasto consistono in 2 o 3 helper, grandi cache e basso o medio grado di contesa.
Come già ampiamente discusso nel paragrafo dedicato al ritardo, la scelta del meccanismo di routing penalizza severamente questa metrica.
L'alternativa proposta, ovvero l'inoltro contemporaneo a tutte le face degli interest ricevuti, è perfettamente ragionevole ed ha, agli occhi di chi scrive, la possibilità di essere implementata.
Tuttavia questi rimangono fattori sui quali l'attaccante non è in grado di influire, poiché è stato stabilito nelle ipotesi sulle possibilità che esso ha, che la modifica del comportamento dei router non è fra queste, pertanto le uniche valide considerazioni su come combattere la problematica del ritardo riguardano solamente gli aspetti su cui l'attaccante ha la possibilità di agire.
Compiute queste ultime osservazioni, passiamo ora a valutare l'influenza (o mancanza della stessa) che le cache collocate nei router possono avere sullo scenario descritto.

5.1.3 - Scenario A2 (cache nei router e RTT da 30 ms)

L'aspetto che si vuole esaminare per primo riguarda la differenza che un produttore malevolo può apprezzare quando sono presenti le cache in rete rispetto a quando non lo sono in condizioni normali, perciò non in presenza dell'attacco.
Questo è interessante perché possiamo vedere la situazione di cache di rete e quella di singolo helper come in concorrenza fra di loro; in fondo entrambe le soluzioni hanno la potenzialità di portare simili vantaggi sia per il produttore che per gli utenti che esso serve.
Vediamo allora queste 3 soluzioni come differiscono fra loro nelle metriche osservate, cominciando dall'hit ratio di Consumer 1 (vedremo per brevità i soli casi con cache da 300 contenuti per l'helper).



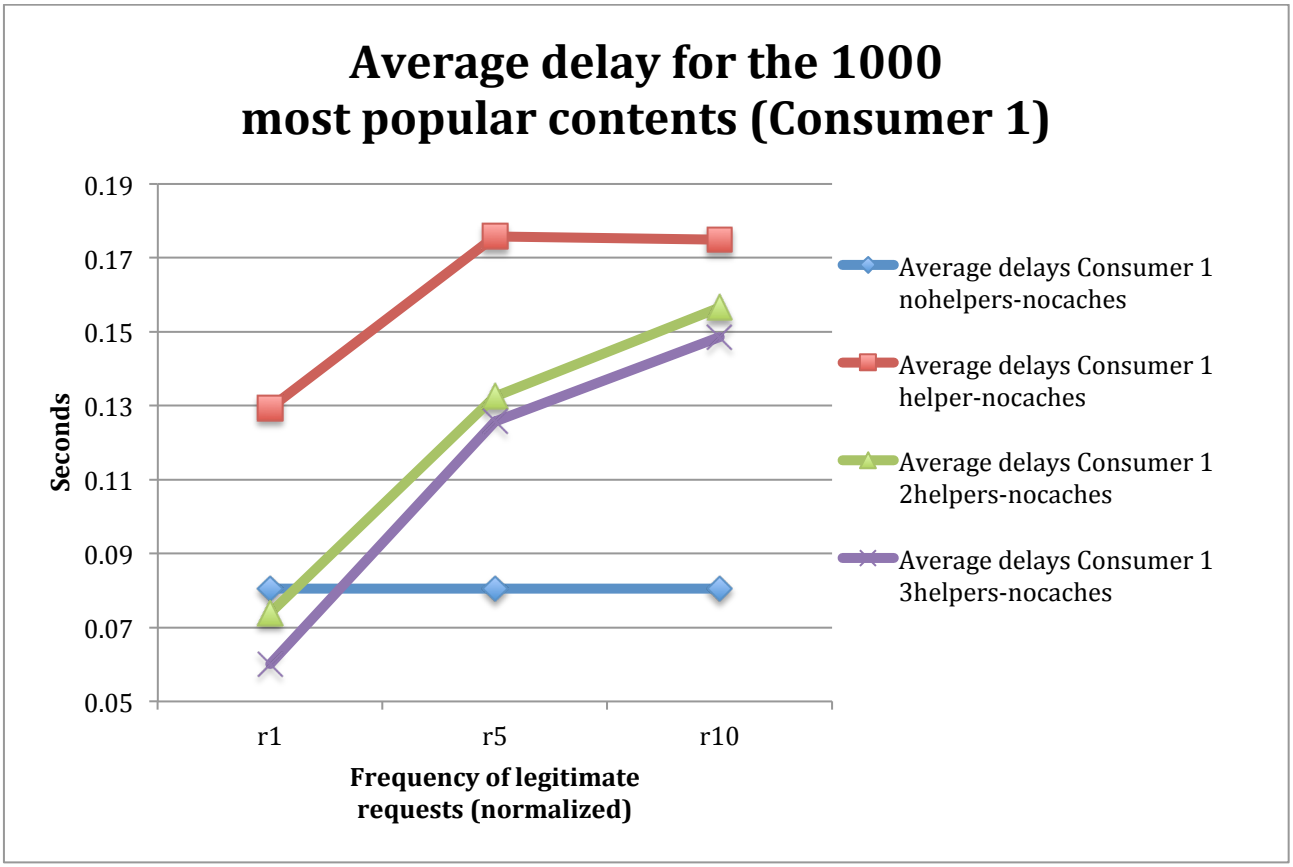
Graf. 5.14 Differenze fra presenza ed assenza di cache di rete

Nessuna variazione viene riscontrata tra le situazioni di base con cache e senza cache, la retta rimane costante esattamente sullo stesso valore del 38% indipendentemente, come giusto che sia, dalla frequenza delle richieste degli helper.
Questo risultato da una prima indicazione sulla bassa o nulla incisività che la presenza di cache nei router ha nello scenario esaminato, a causa di molteplici fattori: tra questi uno consiste nella loro dimensione relativamente bassa ed un altro nella loro condivisione tra 4 utenti, soprattutto per quello che riguarda la cache di Router 1.
Ma la causa preponderante del fenomeno è da attribuire alla presenza di cache locali nei nodi, che svolgono un'importante funzione di assorbimento di primo livello.

È importante sottolineare che non si vuole qui mettere in discussione l'utilità delle cache, ma solo che in questa particolare istanza e con questi parametri di configurazione esse non danno un valore aggiunto.
D'altra parte una dimostrazione di tale fatto è data dallo scenario con helper, che rappresenta di fatto una cache condivisa tra Consumer 1 e lo stesso Helper 1.
Dal momento che essa è di dimensioni superiori e condivisa solo tra questi due utenti rispetto a quella in Router 1, i vantaggi che Consumer 1 riesce a trarre sono tangibili.
I grafici che rappresentano gli andamenti delle altre metriche mostrano lo stesso comportamento e conducono alle stesse conclusioni, pertanto non verranno mostrati.
Quello che resta da dire è che una condizione come quella appena vista, cioè di una rete che in termini di risorse utili al caching non riesce a produrre vantaggi né per il produttore né per gli utenti, può rappresentare anch'essa un incentivo a perpetrare l'attacco.

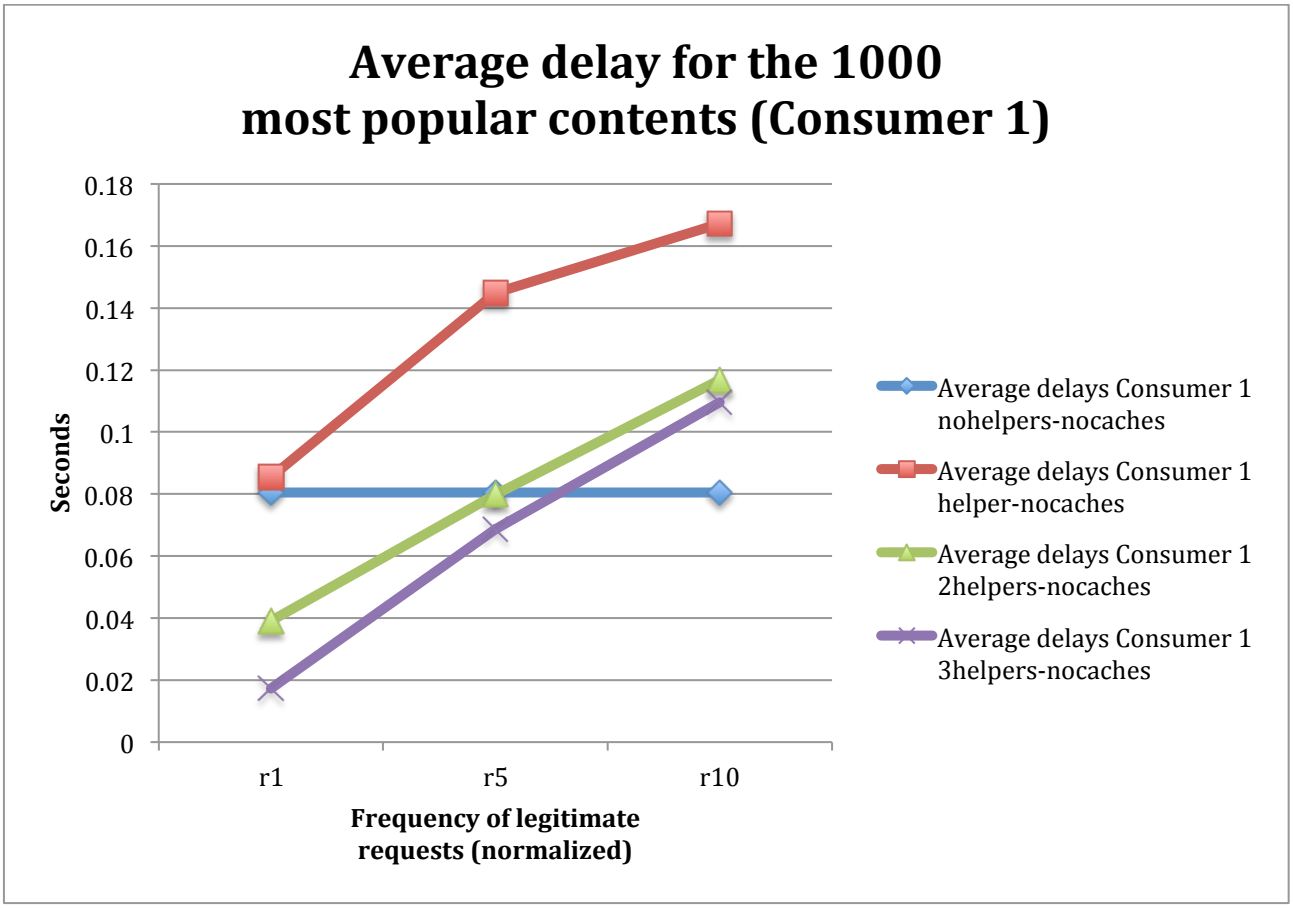
5.1.4 - Scenario B1/B2 (assenza/presenza di cache nei router, RTT da 130 ms)

L'analisi di questi due scenari procederà in parallelo, per rendere l'esposizione più concisa, ma si farà principalmente riferimento alle simulazioni svolte in assenza di cache in rete; verranno solo notificate eventuali differenze, nei contesti in cui esse dovessero emergere.
Questa situazione dovrebbe risultare quella maggiormente appetibile per un produttore malevolo, poiché si aggiunge, rispetto ai casi precedenti, l'elemento della distanza tra utenti e produttore, che significa maggiore ritardo per ricevere i contenuti.
Vedremo soprattutto se la discussione proprio riguardante il ritardo fatta in precedenza sia ancora valida, oppure se, nonostante le penalità già discusse, questa volta si assista ad un miglioramento più significativo anche di questa metrica.
I risultati ottenuti dalle simulazioni svolte mostrano che, 4 delle 6 metriche osservate, non ci sono differenze sostanziali rispetto ai precedenti scenari e la stessa affermazione è valida anche per il confronto svolto fra B1 e B2.
Tra queste le variazioni più consistenti, anche se sempre poco rilevanti, riguardano le consegne medie da parte di Producer 1, che sono lievemente ridotte a causa dell'aumento di RTT.
Una delle eccezioni alla disamina appena svolta riguarda il ritardo calcolato dal punto di vista di Consumer 1, che merita un approfondimento specifico.
Per esaminare i dati si seguirà il solito ordine, già tracciato nelle discussioni precedenti, perciò per primi vedremo i dati che riguardano le simulazioni svolte con dimensioni delle cache degli helper pari a 300 contenuti.



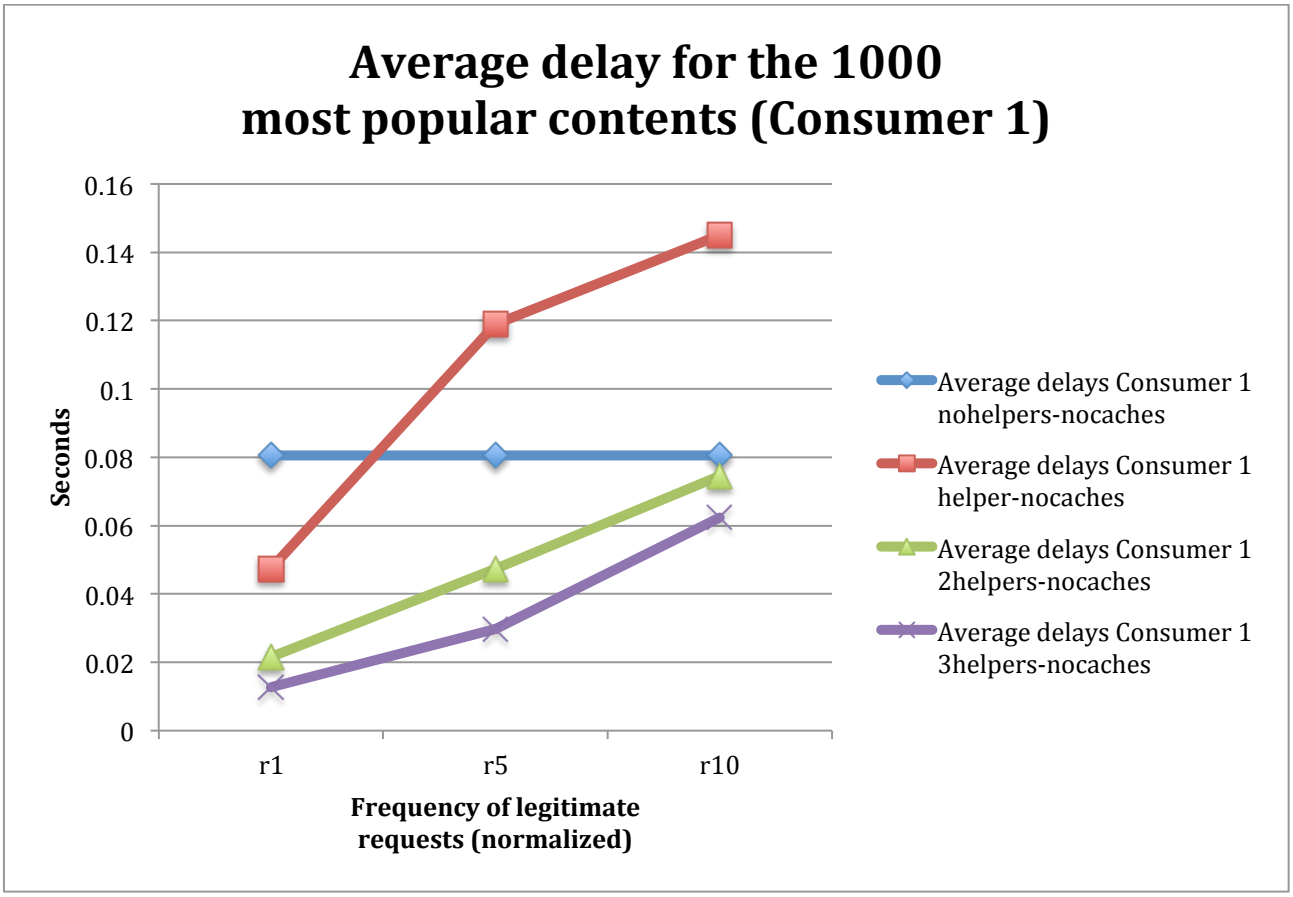
Graf. 5.15 Ritardo medio di Consumer 1 – Scenario B1 – Cache da 300 contenuti - Stesso comportamento

Rispetto alla situazione riscontrata negli scenari A1 e A2, il ritardo nel caso di base è ovviamente molto più elevato e si attesta sugli 80 ms.
Ciò che cambia è che in questo caso bastano già 2 cache piccole e poco contestate per migliorare, anche se di poco (73 ms di ritardo medio), questo risultato.
La situazione migliora ulteriormente con l'aggiunta di una cache, che porta la metrica a toccare i 60 ms.
Inoltre, mentre con RTT da 30 ms i ritardi nelle casistiche con gli helper sono 3 o più volte elevati di quelli del caso di base, nello scenario B1 queste differenze sono maggiormente contenute, con il caso peggiore rappresentato da un raddoppio quando è presente una singola cache compromessa altamente contestata.
Tutto questo significa che ovviamente le problematiche di cui si è parlato in relazione al ritardo non vengono a meno, ma sono mitigate dal fatto che la rete è più estesa.
Per questo motivo l'offloading di contenuti da parte del produttore malevolo sugli utenti compromessi risulta maggiormente efficace anche per ridurre i ritardi sperimentati dagli utilizzatori dei contenuti da egli prodotti, il che rende più appetibile l'attacco per tale produttore.
Si passa ora alla valutazione dei casi con cache da 600 contenuti.



Graf. 5.16 Ritardo medio di Consumer 1 – Scenario B1 – Cache da 600 contenuti - Stesso comportamento

Con una disponibilità di spazio maggiore nelle cache il ritardo cala ulteriormente su tutta la linea, rimanendo al di sotto della soglia tracciata dal caso di base sia per un grado di contesa basso che medio se 2 o 3 helper sono coinvolti nell'attacco.
In particolare nel caso migliore di bassa contesa questa metrica si dimezza e riduce ad un quarto nei due casi sopracitati rispettivamente.
Se invece si considera il caso peggiore si rileva un ritardo superiore del 46 e 36% a quanto misurato nel caso di base per 2 helper e 3 helper, mentre nello scenario A1/A2 questo valore triplicava.
Infine si vuole far notare che con un singolo helper scarsamente contestato, i due valori sono vicinissimi (85 ms per l'helper e 80 ms nel caso base), il che rende ancora più appetibile questo attacco per un produttore malevolo il cui obiettivo primario sia quello di risparmiare sulle consegne effettuate piuttosto che l'aumento di qualità del servizio per i propri utenti.
Con l'aumento del grado di contesa il singolo helper non è in grado di tenere basso il ritardo, ma anche nel suo caso l'aumento del ritardo è meno grave di quanto osservato negli scenari A1/A2; rimane comunque considerevole, fino al punto di essere doppio quando la cache è contestata al massimo grado.
Per concludere, anche qui non è visibile una significativa differenza prestazionale tra 2 e 3 helper, se si eccettua il caso a minimo grado di contesa, in cui il ritardo viene dimezzato quando l'attaccante è in grado di compromettere un utente in più.



Graf. 5.17 Ritardo medio di Consumer 1 – Scenario B1 – Cache da 1000 contenuti - Stesso comportamento

Con cache da 1000 contenuti è possibile apprezzare che si raggiunge la situazione in cui il ritardo con 2 e 3 helper è inferiore a quello ottenibile senza helper per ogni grado di contesa. Di particolare interesse sono le prestazioni ottenute da 2 helper scarsamente contesi .
In questo caso, infatti, si ottiene un risultato abbastanza vicino a quello raggiungibile con 3 helper (21 ms contro 12 ms) e un dimezzamento rispetto a quanto visto con le cache da 600 contenuti.
Infine si nota che per la prima volta anche con un singolo helper, per un basso grado di contesa, si riesce ad abbassare il ritardo, quasi dimezzandolo, sotto al livello base, il che è importante da notare per i motivi cui si accennava al termine del precedente paragrafo.

Si può quindi concludere che i guadagni ottenibili conducendo l'attacco descritto da un produttore malevolo che si trovi distante dagli utenti che vuole servire sono tangibili anche sotto il profilo del ritardo, oltre che delle altre metriche analizzate, come già dimostrato nel corso della discussione riguardante gli scenari A1/A2.
Tutto questo ferme restando le penalità imposte agli utenti che finiscono vittime degli attacchi, soprattutto in termini di ritardo; infatti, in questi due scenari le percentuali di aumento del ritardo sperimentato dagli helper rispetto alla baseline sono consistenti con quanto già visto in precedenza (12, 18 e 18% per helper, 2 helper e 3 helper con cache da 300 contenuti e grado di contesa basso), il che significa in termini assoluti un aumento di 10 e 14 ms, che sono, tuttavia, abbastanza contenuti tranne che per le applicazioni più sensibili a questa metrica.

L'altra metrica che maggiormente è influenzata dal più alto valore di RTT riguarda sempre il tempo, nello specifico quello medio di permanenza in rete dei contenuti.
Come nel caso precedente, essi sono riportati nella tabella sotto riportata (Tab. 5.2):

	Cache size = 300 contents			Cache size = 600 contents			Cache size = 1000 contents		
	R1	R5	R10	R1	R5	R10	R1	R5	R10
No helpers	42 s	42 s	42 s	42 s	42 s	42 s	42 s	42 s	42 s
1 helper	97 s	92 s	90 s	226 s	96 s	92 s	435 s	141 s	96 s
2 helpers	101 s	98 s	94 s	983 s	114 s	100 s	1191 s	380 s	118 s
3 helpers	133 s	107 s	98 s	1679 s	140 s	111 s	6793 s	481 s	149 s

Tab. 5.2 Tempo medio di permanenza in rete dei contenuti – Scenario B1 – Stesso comportamento

Con riferimento al caso precedente, mentre in assenza di helper non c'è sostanziale differenza, l'aumento di questa metrica è pari in media al 9% per un singolo helper, al 3% per 2 helper ed al 5% per 3 helper.
Due fattori che contribuiscono pesantemente a queste medie, portandole verso il basso sono i tempi ottenuti in presenza di 2 e 3 helper con cache da 1000 contenuti e medio grado di contesa, situazioni per le quali si registrano cali di circa il 40% passando dagli scenari A1/A2 a quelli oggetto della discussione.

Come è apparso dalle discussioni riguardanti le singole metriche, l'attacco condotto nelle condizioni descritte dagli scenari B1/B2 risulta maggiormente efficace ed appetibile.
Non solo, infatti, i guadagni registrati in precedenza rimangono intatti, ma si riscontrano importanti vantaggi anche sul fronte del ritardo medio con cui Consumer 1 riesce ad accedere ai contenuti in offloading, tallone d'Achille della situazione presente negli scenari A1/A2.
Come nota conclusiva si riesaminano qui le scelte strategiche, come è stato per la sezione precedente.
Ancora una volta si esamineranno le differenti esigenze di due diversi produttori; uno di questi è semplicemente interessato ad ottenere una diminuzione delle proprie consegne, interessandosi soltanto a non intaccare in maniera troppo debilitante le prestazioni degli helper e di Consumer 1.
L'altro invece punta a dare una migliore qualità del servizio agli utenti, cercando di avere un impatto minimo su quella degli helper ed è disposto a mantenere lo stesso numero di consegne, o anche ad aumentarlo in alcuni casi, o a spendere ed esporsi in maniera superiore per la compromissione di migliori vittime.
Per il produttore malevolo del primo tipo un approccio che coinvolga un solo helper sembra la migliore soluzione anche in questo scenario, grazie alla decisa riduzione delle consegne che esso deve effettuare (che da 41 mediamente si dimezzano), che si uniscono ad un buon miglioramento dell'hit ratio (dal 38 fino anche al 84% nel caso migliore) e ad un ritardo che

con cache da 1000 contenuti scende per l'unica volta sotto a quanto ottenibile in assenza di helper.
Inoltre questa soluzione è anche quella che produce il più alto miglioramento del tempo di permanenza in rete dei contenuti, che si alza del 20% rispetto a prima.
Di converso però, situazioni in cui queste metriche sono avvantaggiate corrispondono ad un più sensibile peggioramento delle performance lato helper, perciò il produttore deve prestare almeno un po' di attenzione a tali aspetti.
L'altro punto di vista da considerare è che nonostante il miglioramento dei risultati in termini di ritardo visto da Consumer 1, esso tende a crescere in maniera molto forte anche solo passando da un grado di contesa basso ad uno medio, il che chiaramente rappresenta un problema soprattutto se la frequenza di interest legittimi emessi dall'helper varia con un alto ritmo.
Di partenza però l'attaccante dovrebbe ricercare un utente da compromettere che generi poco traffico ma che abbia una cache di medie o grandi dimensioni a disposizione.
Il peggioramento prestazionale che questo utente (considerando basso grado di contesa e cache da 600 contenuti) vedrebbe consisterebbe in un calo di 10 punti percentuali dell'hit ratio e di un aumento medio di 13 ms del proprio ritardo.
D'altra parte il produttore godrebbe di una drastica riduzione delle consegne, da 41 a 15, e ed i suoi utenti un aumento dell'hit ratio dal 38 al 70% ed un aumento del ritardo di soli 5 ms.
Passando invece ad esaminare il caso del produttore appartenente alla seconda tipologia, si può dire che esso abbia la possibilità di spaziare fra molteplici scelte.
A causa del visibile peggioramento che il ritardo sperimentato da Consumer 1 ha passando da un basso ad un medio grado di contesa (da 73 a 130 ms con 2 helper e da 60 a 125 ms con 3 helper), si ritiene che le cache da 300 contenuti non siano abbastanza contenitive per gli scopi stabiliti in precedenza, perciò queste dovrebbero essere usate solo in accoppiata a nodi poco attivi per la maggior parte del tempo.
Restringendo quindi la scelta delle dimensioni delle cache a 600 e 1000 contenuti, si ritiene che la soluzione più equilibrata sia anche questa volta quella che comprende 2 helper e che non ci sia un incentivo sufficiente, in generale, per compromettere un utente in più.
Questo può essere tuttavia trovato per alcuni casi di nicchia; infatti con 3 helper un produttore malevolo riesce ad offrire ai propri utenti un ritardo molto basso in termini assoluti (12 ms) e, soprattutto, nel migliore caso in assoluto a ridurre praticamente a 0 il numero di consegne che deve effettuare, di fatto scaricando completamente questo onere sugli helper con solo un minimo di disseminazione iniziale.
Detto questo, avendo la disponibilità di 2 cache da 600 contenuti è possibile per Consumer 1 arrivare ad un hit ratio significativamente superiore al caso di base in ogni condizione di contesa e ad un ritardo che può essere dimezzato se le cache sono poco contese, pari se lo sono mediamente e superiore di 30 ms se il grado di contesa è elevato.
Allo stesso tempo Producer 1 vede le proprie consegne abbassarsi in ogni condizione e gli helper hanno solo lievi svantaggi (un calo fino al 7% dell'hit ratio ed un aumento al più di 10 ms sul ritardo).
Questa soluzione permette oltretutto di migliorare il tempo di ritenzione dei contenuti in rete in media del 23%.

5.2 – Test con differenti gradi di contesa degli helper

Fino a questo momento tutte le osservazioni fatte hanno avuto come fattor comune, tra gli altri, quello di presumere che tutti gli helper effettuassero lo stesso volume di traffico, ovvero che richiedessero contenuti tramite messaggi di interest alla stessa frequenza.
Ciò può essere indice, ad esempio, di locazione geografica degli utenti; poiché tipicamente l'utilizzo che un utente medio fa dei propri dispositivi connessi è molto superiore durante le ore di luce che quelle di buio, un simile grado di attività fra alcuni di essi si può osservare quando questi si trovano in zone abbastanza vicine da essere all'incirca nella stessa fase della giornata.
Allo stesso modo una differenza anche ampia di attività è facilmente osservabile tra diversi utenti sparsi per il globo.
Chiaramente non è la sola locazione geografica che può fungere da fattore determinante per le differenze di volume di attività generata, ma solo uno dei tanti, il che significa che però che tale situazione è abbastanza comune da meritare approfondimento.
Ovviamente l'analisi verterà sui soli scenari con 2 e 3 helper, gli unici che possono presentare differenze nel comportamento degli helper, i cui risultati verranno messi a confronto con quanto già visto nella sezione precedente, in modo da permettere la creazione di una visione prospettica sulla discussione.
Perciò nella presente sezione ci si concentrerà su questo tipo di tematica, procedendo nello stesso ordine descritto nella sezione precedente, con una piccola variazione, che consiste nel raggruppamento dei dati riguardanti la capienza delle cache degli utenti compromessi.
Ogni grafico contiene quindi una curva per ciascuna combinazione tra i due vettori di capienza della cache e stesso o differente comportamento degli helper.
Un'altra lieve divergenza con il metodo di presentazione dei dati rispetto alla precedente sezione riguarda il parametro inserito nell'asse delle ascisse: mentre prima era il valore numerico della frequenza delle richieste legittime inviate dall'helper su quelle illecite, ora, essendo questi diversi tra gli helper, si è deciso di inserire il grado di contesa al posto della frequenza, con le possibilità sempre costituite da basso, medio ed alto.
La corrispondenza fra i possibili gradi e le frequenze numeriche degli helper (sempre normalizzate) è riassunto nella tabella sottostante:

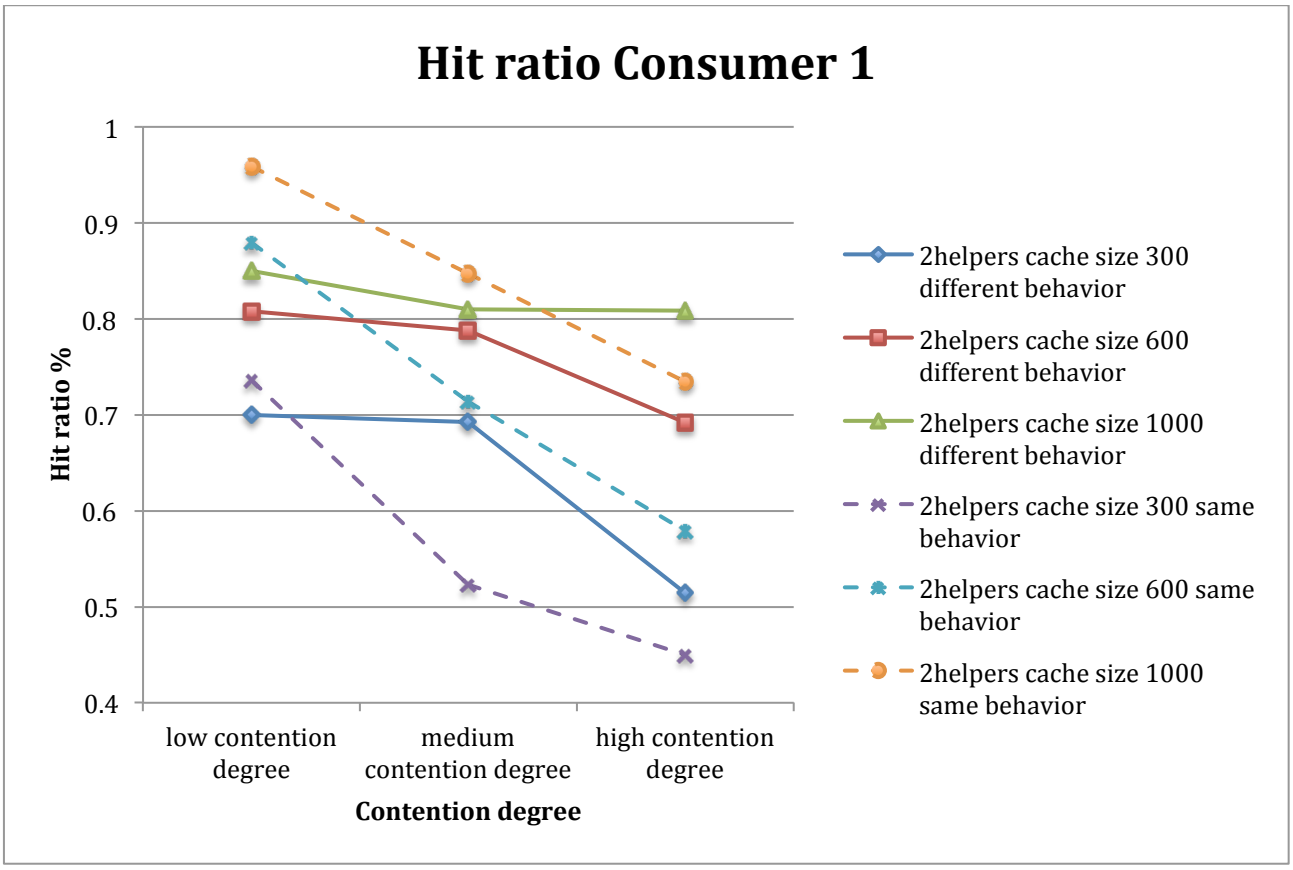
	Helper 1	Helper 2	Helper 3 (if present)
Low contention degree	1	5	10
Medium contention degree	1	10	10
High contention degree	5	10	10

Tab. 5.3 Corrispondenza fra grado di contesa e frequenza di richiesta

La discussione sulla suddivisione del carico fra i 2 o 3 utenti è stata già affrontata nel paragrafo titolato "Numero di contenuti illegittimamente richiesti dagli helper", alle pagine 23 e 24 del capitolo 4 e per le simulazioni i cui risultati sono riportati nella presente sezione sono state impiegate le considerazioni in esso presentate.
Ora che tutte le puntualizzazioni riguardanti la sezione sono state fatte, i prossimi paragrafi conterranno i grafici ed i loro commenti per ciascuno scenario.

5.2.1 - Scenari A1/A2 (RTT di 30 ms)

Viene mostrato in seguito il grafico dell'hit ratio di Consumer 1.

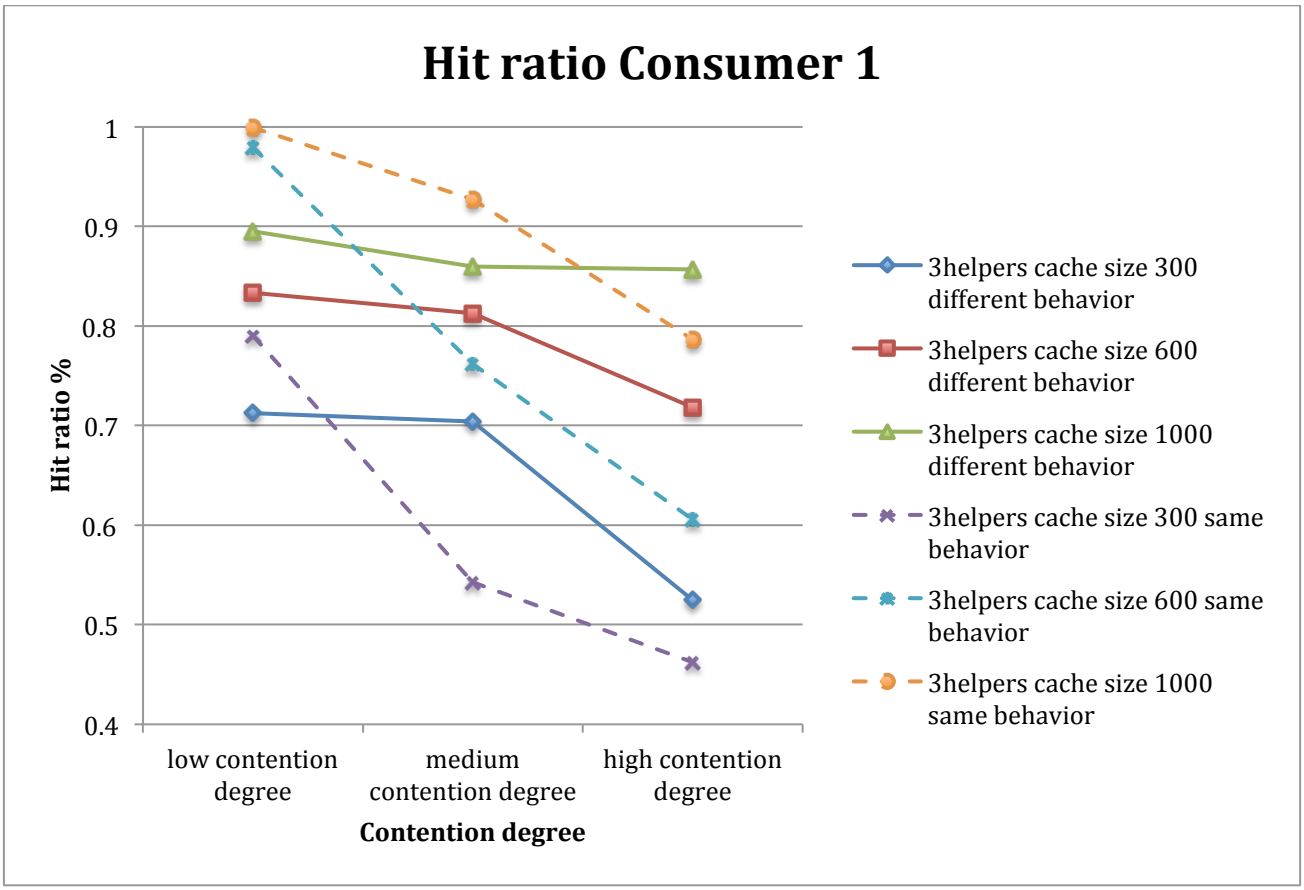


Graf. 5.18 Hit ratio di Consumer 1 – Scenario A1 – 2 helper – Confronto fra stesso e diverso comportamento

Le linee tratteggiate costituiscono l'andamento della metrica con stesso comportamento da parte degli utenti.
Il primo elemento che immediatamente balza agli occhi è che i risultati assoluti più elevati rimangono quelli ottenibili quando gli utenti hanno lo stesso comportamento ed effettuano poche richieste, anche se non di molto, in quanto, confrontando l'uno con l'altro gli scenari in cui gli helper hanno a disposizioni cache della medesima grandezza, si nota un hit ratio superiore del 4%, 7% e 10% per cache da 300, 600 e 1000 contenuti rispettivamente.
L'altra osservazione importante è che nel passaggio tra basso e medio grado di contesa gli scenari in cui il comportamento degli helper è differente, il calo di questa metrica è praticamente insignificante, trattandosi dello 0%, 2% e 4% per le 3 possibili dimensioni delle cache, ben diversamente da ciò che avviene se il comportamento è lo stesso, situazione nel quale lo stesso decadimento è molto più drastico, essendo del 21%, 17% e 11% rispettivamente.
Particolarmente evidente è la differenza quando le cache possono memorizzare 300 contenuti, con la curva che rappresenta differente comportamento che rimane pressoché costante sul 70% e quella relativa allo stesso comportamento che decade al 52%.
Chiaramente il fattore che fa la differenza in questi casi è la disponibilità di un helper molto poco conteso.

Analizzando la parte più a destra del grafico, ovvero quella di transizione tra medio ed alto grado di contesa, ciò che emerge è che la limitazione nella dimensione delle cache comincia a diventare un problema rilevante anche in caso di diverso comportamento, anche se le prestazioni restano comunque superiori a quello di stesso comportamento.
In particolare con cache da 600 e 1000 contenuti l'attaccante è in grado di mantenere un hit ratio piuttosto elevato in ogni condizione, col caso peggiore che si posiziona al 69% ed 80% rispettivamente.

Si passa ora a valutare sempre l'hit ratio di Consumer 1, ma questa volta quando 3 helper sono coinvolti nell'attacco.

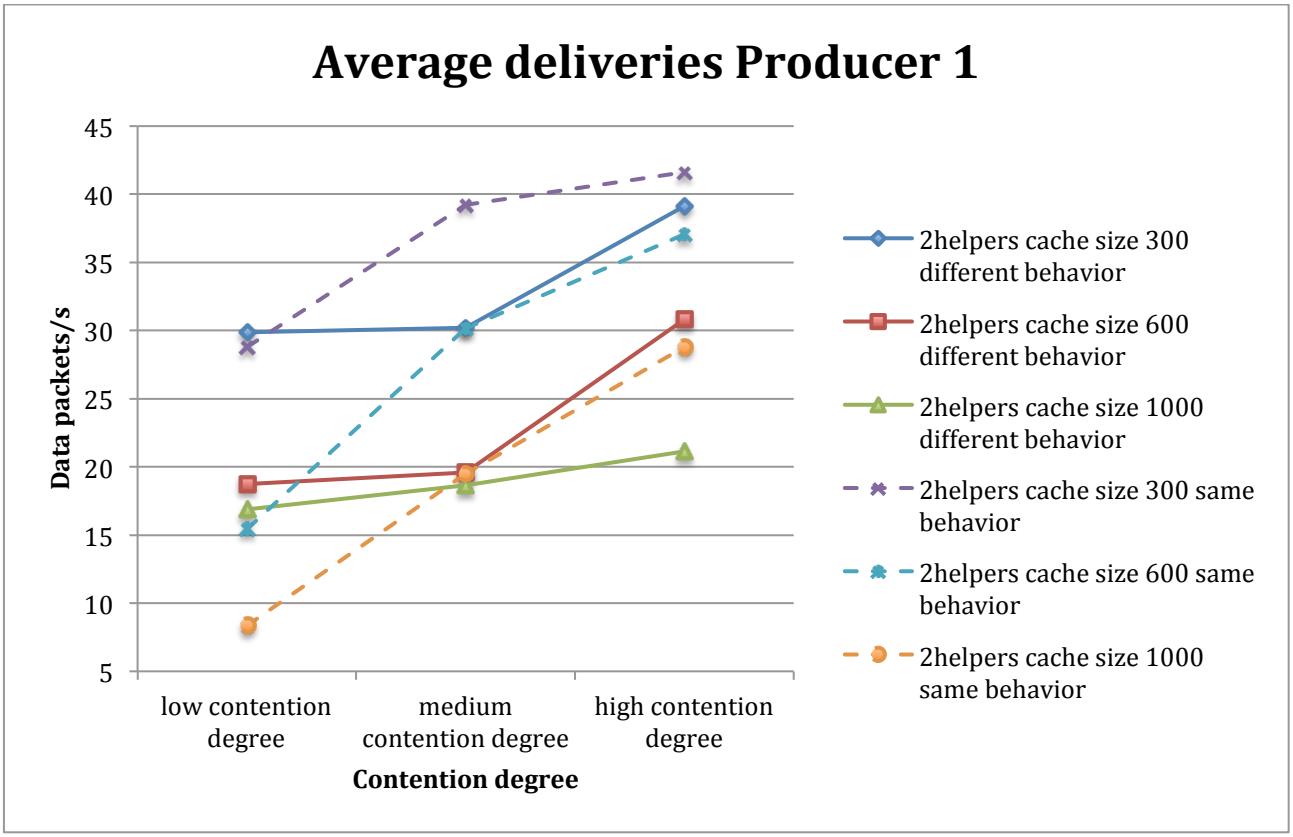


Graf. 5.19 Hit ratio di Consumer 1 – Scenario A1 – 3 helper - Confronto fra stesso e diverso comportamento

I risultati suggeriscono conclusioni simili a quanto è stato illustrato nel precedente paragrafo, anche se qualche nota riguardante i numeri specifici del caso in esame è doveroso farla.
Partendo come prima dalla situazione migliore, questa volta il gap presente tra helper aventi lo stesso comportamento e helper aventi comportamento diverso è più elevato e ciò è dovuto al fatto che anche nella migliore delle ipotesi uno di essi genera sempre un elevato volume di traffico.
In percentuale il divario prestazionale è dell'8%, 15% e 10% per cache da 300, 600 e 1000 contenuti rispettivamente.
Tuttavia è interessante osservare che, anche nel caso di lavoro suddiviso tra 3 utenti, garantisce risultati migliori il fatto che una tra le tre cache sia minimamente contesa piuttosto che la disponibilità di 3 mediamente contese.

Nelle parti centrale e destra del grafico, la differenza fra le due casistiche si riduce rispetto a quanto visto nel paragrafo precedente per dimensioni delle cache da 300 e 600 contenuti (qui il margine di vantaggio è del 16% e 5%), mentre essa aumenta per cache da 1000 contenuti (6% in più quando gli utenti hanno lo stesso comportamento).
Come ultima nota anche dal confronto di queste situazioni emerge che in generale i guadagni prestazionali non siano sufficientemente elevati da giustificare l'utilizzo di un terzo helper quando 2 sono già operativi, a meno che l'attaccante non sia interessato ad arrivare al massimo valore di hit ratio possibile ed al suo mantenimento ad un valore così elevato.

La prossima metrica da discutere consiste nel numero medio di consegne di contenuti in offloading effettuate da Producer1.

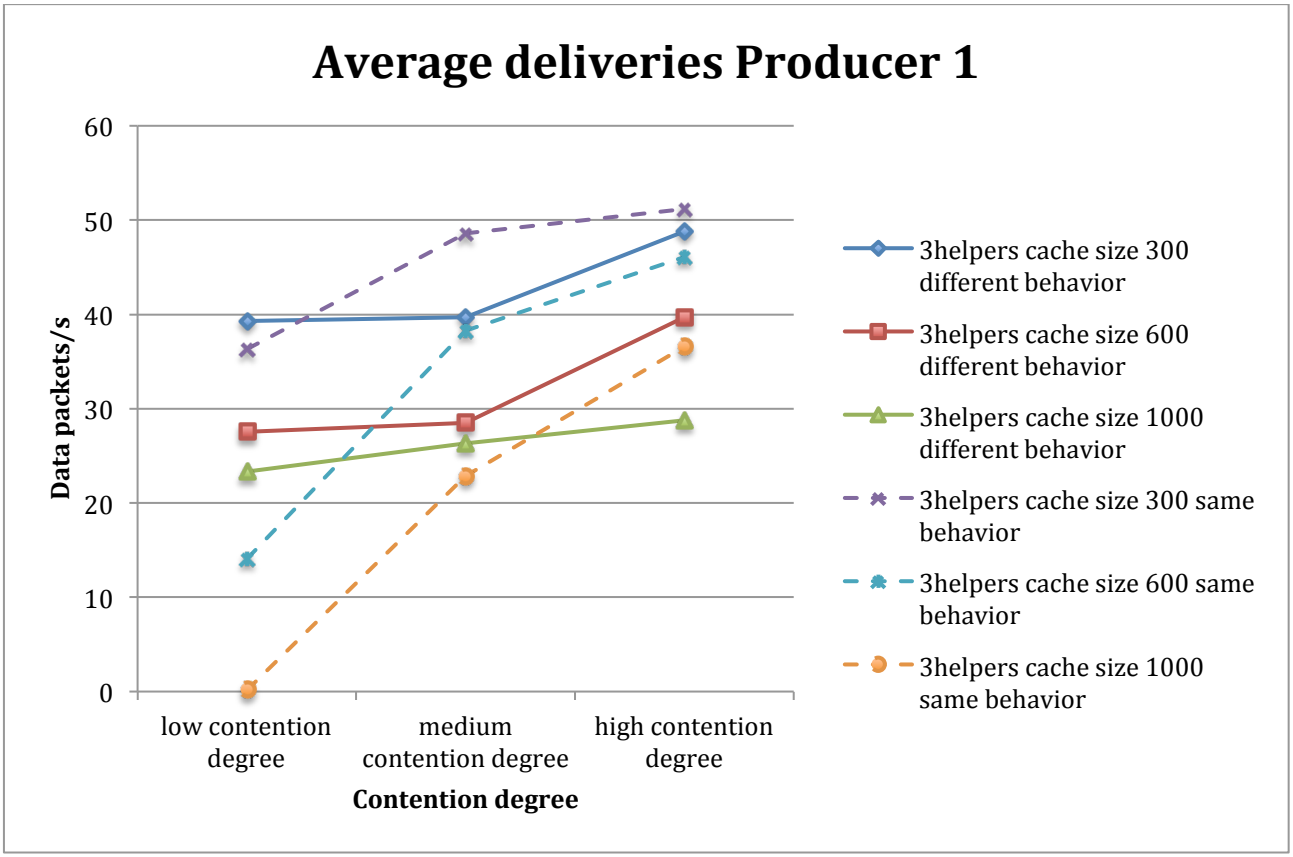


Graf. 5.20 Numero medio di consegne Producer1 – Scenario A1 – 2 helper – Confronto fra stesso e diverso comportamento

Anche dall'osservazione di questo grafico emerge una rappresentazione simile a quella discussa nei precedenti paragrafi.
La disponibilità di un utente che fa poco traffico, insieme al fatto che è tale utente quello che porta la maggior parte del carico, gioca nuovamente un ruolo fondamentale nel mantenimento di un alto livello prestazionale, che in questo caso si traduce in un basso numero medio di consegne effettuate dal produttore malevolo.
Qui, come in precedenza, il livello minimo si può raggiungere solo avendo a disposizione 2 cache molto grandi e scarsamente contese, ma non è possibile mantenere queste prestazioni man mano che il grado di contesa aumenta, cosa che invece si può fare grazie alla disponibilità di utenti dal differente comportamento.
L'aspetto forse più interessante che riguarda questa metrica in modo specifico è la differenza risibile osservabile, per medio ed alto grado di contesa, tra utenti dallo stesso comportamento

con cache da 600 e 1000 contenuti e tra quelli con differente comportamento e cache da 300 e 600 contenuti rispettivamente.
Il fatto che la differenza sia così bassa significa che dal punto di vista prestazionale, per questa particolare metrica, la scelta di compromettere utenti con disponibilità di cache di dimensioni maggiori o che offrano diverso grado di contesa non è rilevante, il che permette all'attaccante di effettuarla valutando esclusivamente altri fattori, come il costo della compromissione dell'uno o dell'altro tipo.

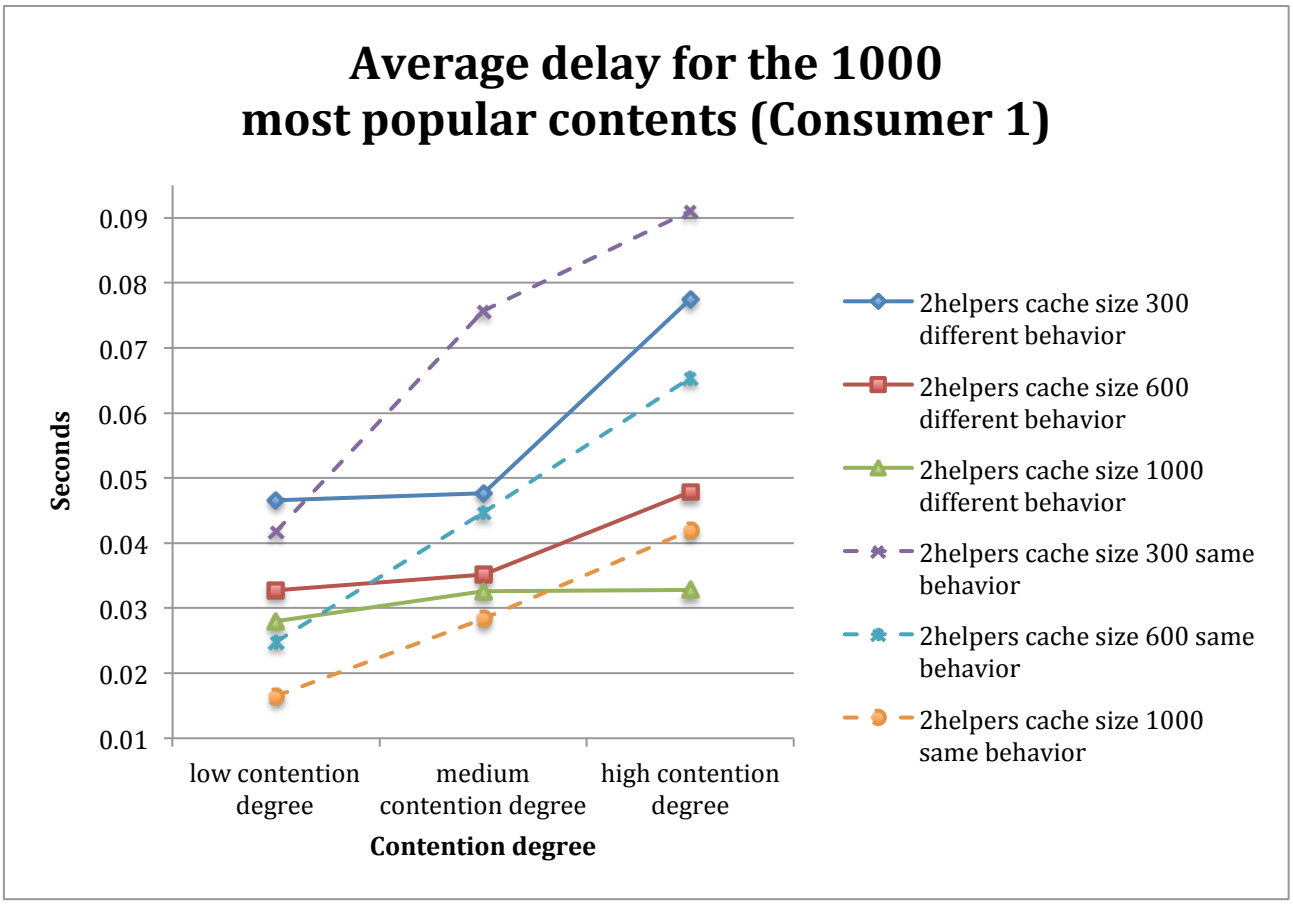
Vediamo ora la situazione relativa ai 3 helper.



Graf. 5.21 Numero medio di consegne Producer1 – Scenario A1 – 3 helper – Confronto fra stesso e diverso comportamento

I discorsi fatti nelle conclusioni della precedente sezione e nella discussione sull'hit ratio di Consumer 1 e le differenze fra 2 e 3 helper mantengono in questo frangente la loro validità.
Abbiamo quindi davanti agli occhi un generale aumento della metrica in oggetto, che significa un peggioramento delle prestazioni per Producer1.
I miglioramenti percepiti sono riferiti in maniera specifica agli scenari con cache dalle dimensioni più elevate (600 o 1000 contenuti), in presenza di utenti generanti sia lo stesso che differente volume di traffico, che garantiscono nel primo caso le migliori prestazioni possibili (14 e 0.1 pacchetti al secondo consegnati), nel secondo il mantenimento di un buon livello delle stesse.
Anche qui si riscontra la sostanziale parità tra le situazioni evidenziate nel paragrafo precedente, anche se in questo caso la differenza è più marcata, soprattutto se si confrontano utenti con diverso comportamento e cache da 600 contenuti ed utenti con stesso comportamento ma cache da 1000 contenuti.

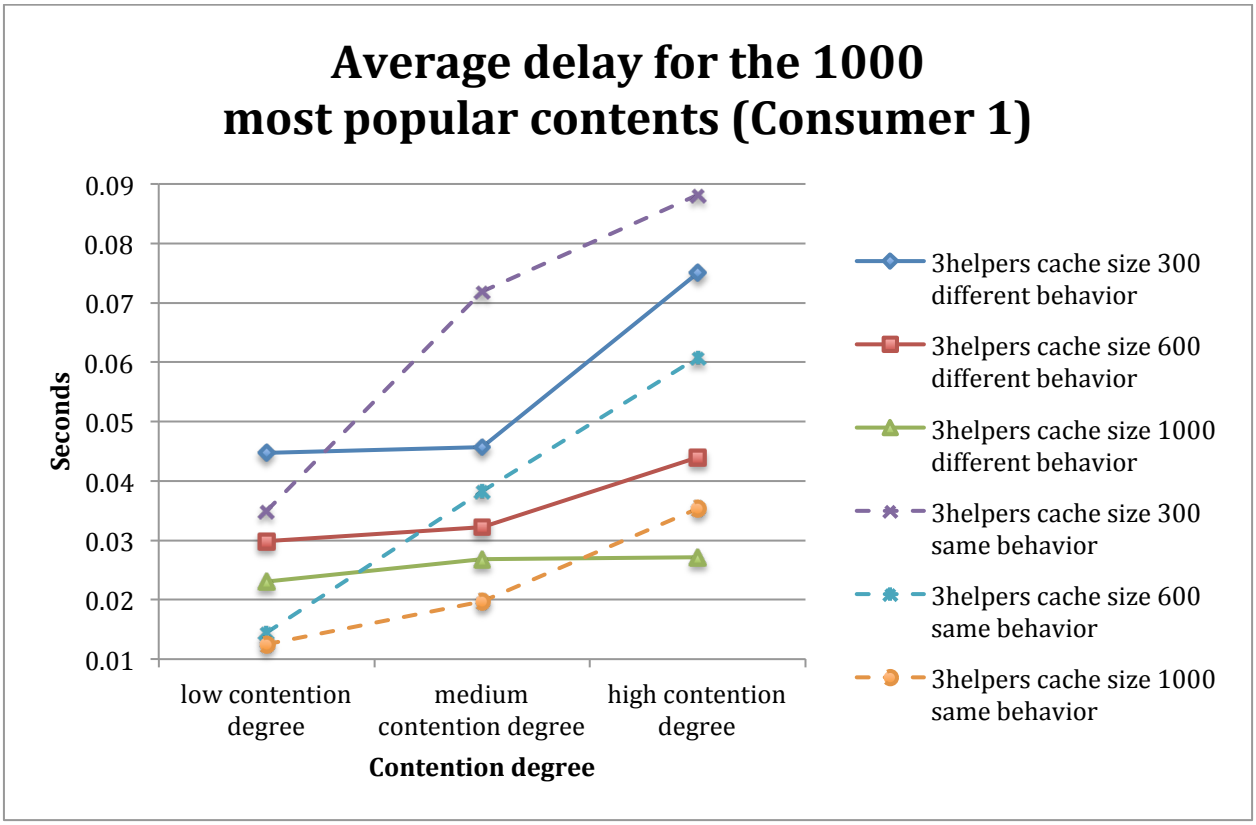
Vediamo ora il ritardo registrato da Consumer 1 sui 1000 contenuti più popolari offerti da Producer1, quelli in offloading.



Graf. 5.22 Ritardo medio di Consumer 1 – Scenario A1 – 2 helper – Confronto fra stesso e diverso comportamento

Per mettere questi risultati in prospettiva, si ricorda che il valore assunto da questa metrica nel caso di base, sia con cache nei router che senza è di 18 ms.
Vediamo chiaramente che nemmeno l'utilizzo di helper con gradi di contesa diversi è abbastanza per tenere consistentemente il ritardo su un valore così basso, ma, visto quanto detto finora c'era da attenderselo.
Quello che è interessante notare è la significativa differenza che si può verificare in situazioni di media ed alta contesa; ad esempio per quanto riguarda cache da 300 contenuti, se queste sono caratterizzate dallo stesso grado di contesa il ritardo si attesta a 75 ms, mentre per gradi di contesa differenti il valore è di 44 ms, un buon 41% in meno.

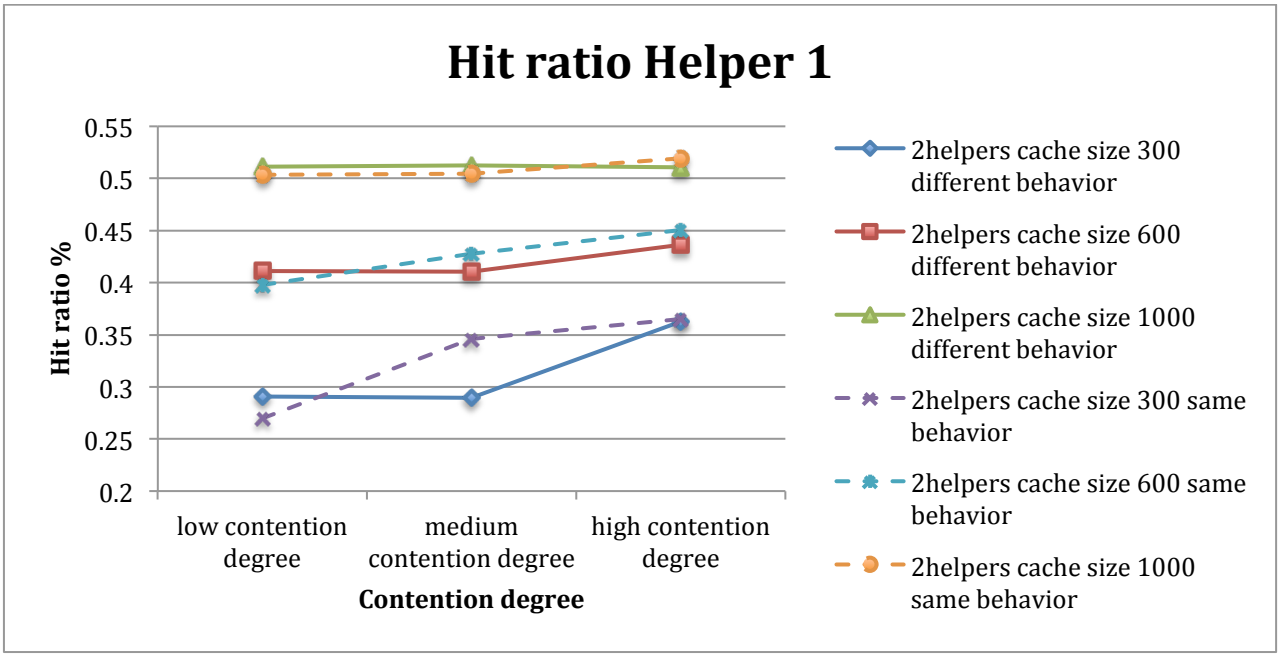
Segue ora il grafico relativo ai 3 helper.



Graf. 5.23 Ritardo medio di Consumer 1 – Scenario A1 – 3 helper – Confronto fra stesso e diverso comportamento

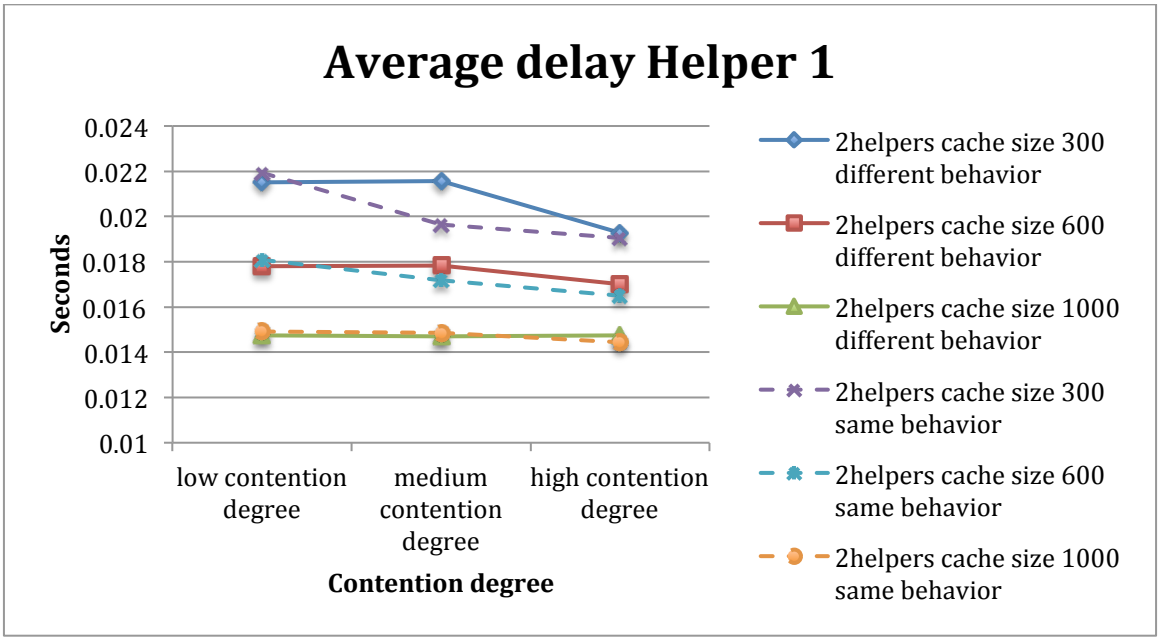
3 helper a differente grado di utilizzo diventano sicuramente un'opzione interessante quando si vuole raggiungere un ritardo molto basso e mantenerlo in ogni condizione, ma solo per cache capienti.
Infatti, con 3 helper aventi cache da 1000 contenuti, questa metrica può essere portata ad un livello molto vicino a quanto Consumer 1 è in grado di ottenere senza attacco (25 ms contro 18 ms).

Nella discussione che segue si andranno ad illustrare le metriche che riguardano gli helper compromessi, perciò i loro hit ratio e ritardo.
In particolare sono riportati i valori ottenuti dall'helper nelle condizioni peggiori, vale a dire quello più sfruttato e che offre la minima contesa fra tutti, che è Helper 1 in ogni caso.
Inoltre, date le irrilevanti differenze sussistenti fra 2 e 3 helper, solo i dati riguardanti la prima delle due casistiche vengono qui rappresentati e commentati, partendo con l'hit ratio, mostrato nel prossimo grafico.



Graf. 5.24 Hit ratio di Helper 1– Scenario A1 – 2 helper – Confronto fra stesso e diverso comportamento

Ancora una volta si ricorda che il valore di base per questo parametro è del 38% per cache da 300 contenuti, 47% per cache da 600 contenuti e 55% per quelle da 1000 contenuti.
Le differenze sono praticamente nulle in quasi tutte le situazioni, se si eccettuano i 6 punti percentuali che dividono helper con cache da 300 contenuti aventi stesso comportamento e quelli aventi diverso comportamento e che sono dovuti al basso grado di contesa offerto da Helper 1.
Questa divergenza si annulla però sia per grado di contesa basso che alto.
Come evidenziato anche nella discussione sul numero medio di consegne, il fatto che non ci sia differenza significa che l'attaccante non ha di fatto alcuna penalità nella scelta tra i due tipi di vittime da questo punto di vista; altri sono quindi i fattori che egli può tenere di più in considerazione per operare tale decisione.
Per quello che concerne il ritardo, la situazione è presentata nella figura sottostante:



Graf. 5.25 Ritardo medio Helper 1– Scenario A1 – 2 helper – Confronto fra stesso e diverso comportamento

Come riferimento, in questo caso i valori di base per ciascuna differente dimensione della cache sono 18 ms per 300 contenuti, 15 per 600 contenuti e 13 per 1000 contenuti.
La figura è pressoché speculare a quella precedente, perciò la discussione rimane la stessa.

A seguire viene illustrata la situazione dei tempi di permanenza in rete (Tab. 5.4), per la quale si offre un confronto diretto con ciò che è stato rilevato e mostrato nella precedente sezione.

	Cache size = 300 contents			Cache size = 600 contents			Cache size = 1000 contents		
	lcd	mcd	hcd	lcd	mcd	hcd	lcd	mcd	hcd
2 helpers same behavior	100 s	80 s	78 s	999 s	93 s	81 s	1195 s	606 s	100 s
2 helpers different behavior	89 s	87 s	78 s	959 s	832 s	86 s	902 s	968 s	631 s
3 helpers same behavior	131 s	86 s	81 s	1687 s	114 s	90 s	6704 s	692 s	123 s
3 helpers different behavior	93 s	89 s	80 s	1023 s	986 s	95 s	961 s	972 s	575 s

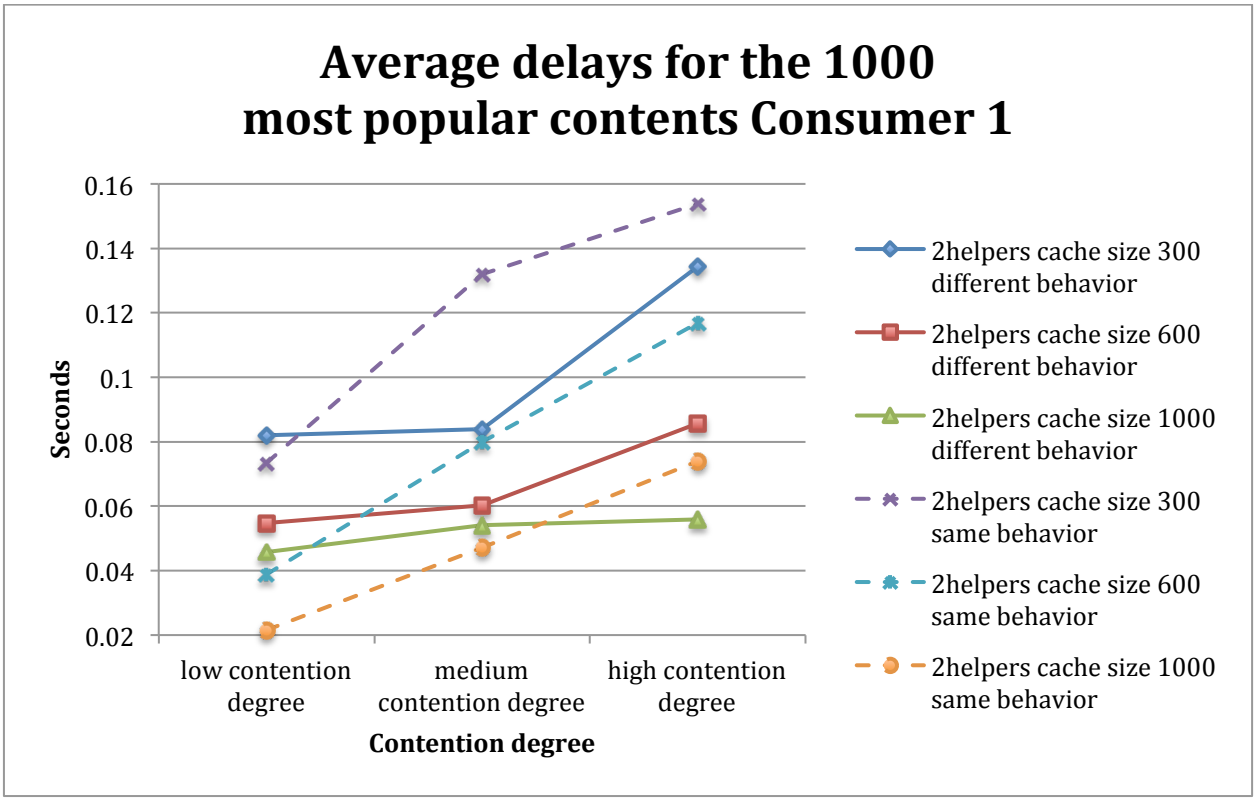
Tab. 5.4 Confronto fra stesso e diverso comportamento per i tempi medi di permanenza in rete – Scenario A1

Le considerazioni che possono essere fatte per questa metrica può essere simile a quelle espresse per le metriche precedenti, in quanto si verifica ancora che quando 2 o 3 helper hanno comportamento omogeneo fra loro si raggiungono i risultati massimi per cache poco contese e grandi, ma questo viene seguito da un peggioramento deciso appena il grado di contesa si eleva.
Al contrario, quando gli helper hanno tra loro comportamento differente, questo calo di fatto viene spostato al passaggio da medio ad alto livello di contesa.
Come ultimo spunto viene fatto notare che per alto grado di contesa le prestazioni con cache da 300 e 600 contenuti vengono tutte più o meno livellate tra gli 80 ed i 95 secondi, mostrando chiaramente un effetto a collo di bottiglia dovuto a dimensioni delle cache insufficienti.
Quando tale collo di bottiglia è rimosso portando le cache ad accogliere 1000 contenuti, un altro ne emerge, ossia quello dovuto a tutti gli helper che causano un alto grado di contesa, con la propria frequenza di richieste legittime al valore massimo.

5.2.2 - Scenari B1/B2 (RTT di 130 ms)

Una volta conclusa la discussione riguardante tutte le metriche per gli scenari A1/A2 è ora il momento di valutare quelli ottenuti per i rimanenti due scenari.
Tuttavia, come è emerso durante l'esposizione della precedente sezione, molte delle metriche non presentano grandi differenze tra i 4 scenari, soprattutto quelle che riguardano gli hit ratio e le consegne, ma anche il ritardo dal punto di vista degli helper non cambia sostanzialmente se visto in termini percentuali.

Verranno di conseguenza illustrate solo le due metriche che mostrano cambiamenti di sorta, ovvero il ritardo di Consumer 1 ed i tempi medi di permanenza in rete dei contenuti.
Inoltre, anche in questo caso, i risultati sono in grado di rappresentare sia la situazione con 2 helper che quella con 3, con la differenza che i valori per il secondo caso sono inferiori, ma essi scalano con una costante di proporzionalità molto simile a quella che si può apprezzare nei dati relativi ai ritardi misurati con stesso comportamento degli utenti compromessi, visibili nella sezione precedente.
Per questo motivo un grafico specifico che rappresenti i risultati ottenuti dalle simulazioni svolte con 3 helper a differente grado di contesa sarà omesso.
Si procede quindi con la discussione del ritardo che Consumer 1 osserva nella ricezione dei 1000 contenuti più popolari.



Graf. 5.26 Ritardo medio Consumer 1 – Scenario B1 – 2 helper – Confronto fra stesso e diverso comportamento

Come per le precedenti disamine per prima cosa si ripresenta il dato del caso di base, quindi in assenza di helper, che è di 80 ms.
Gli andamenti visualizzati in figura sono del tutto simili a quelli già visti nei precedenti grafici; questo tuttavia mette in evidenza un paio di aspetti interessanti.
Il primo riguarda il fatto che per quasi ogni grado di contesa con cache da 600 contenuti è possibile rimanere sotto alla soglia degli 80 ms ed in caso di massimo grado si supera questo limite di appena 5 ms.
L'altra considerazione importante riguarda la bassa e media contesa con cache a 300 contenuti; in queste situazioni il ritardo è appena superiore agli 80 ms, il che significa che l'attaccante può permettersi di utilizzare 2 helper con cache piccole, purché uno di essi offra

basso livello di contesa invece che 2 helper con cache doppiamente più grandi, che potrebbero essere più difficili da trovare.

Infine vediamo la tabella riassuntiva del tempo medio di permanenza in rete dei contenuti.

	Cache size = 300 contents			Cache size = 600 contents			Cache size = 1000 contents		
	lcd	mcd	hcd	lcd	mcd	hcd	lcd	mcd	hcd
2 helpers same behavior	101 s	98 s	94 s	983 s	114 s	100 s	1191s	380 s	118 s
2 helpers different behavior	108 s	105 s	94 s	807 s	898 s	105 s	865 s	990 s	351 s
3 helpers same behavior	133 s	107 s	98 s	1679 s	140 s	111 s	6793 s	481 s	149 s
3 helpers different behavior	113 s	109 s	99 s	971 s	900 s	117 s	950 s	936 s	366 s

Tab. 5.5 Confronto fra stesso e diverso comportamento per i tempi medi di permanenza in rete – Scenario B1

Il pattern anche in questo caso si ripete nel confronto fra comportamento omogeneo ed eterogeneo degli helper.
I valori di picco ottenuti con la prima variante sono però questa volta sensibilmente superiori, soprattutto nel caso dei 3 helper.
Confrontando infine questi valori con quelli ottenuti per gli scenari A1/A2 nella sezione corrente in media la variazione è praticamente nulla, ma anche in questo frangente ci sono due casi che presentano valori altamente diversi, nello specifico il risultato visibile in tabella per 2 helper con cache da 1000 contenuti ed alto grado di contesa che cala del 44% rispetto al corrispondente valore dello scenario A1 e quello riguardante i 3 helper con cache da 1000 contenuti ed alto grado di contesa in discesa del 36%.

5.2.3 - Conclusioni della sezione

Come considerazioni finali di questa sezione si vuole esplicitare il concetto chiave emerso dall'osservazione attenta dei risultati emersi nel corso delle simulazioni.
Questo concetto è la dualità degli approcci presentati; da un lato abbiamo la compromissione di diversi utenti omogenei nel loro comportamento, che porta alle prestazioni di picco più elevate in tutte le metriche quando le condizioni sono le migliori possibili per l'attaccante, ovvero che prevedano vittime con ampie cache a disposizione e che mantengano un basso grado di contesa.
Queste alte prestazioni decadono generalmente in maniera lineare all'aumentare del livello di contesa ed in modo più grave se le cache compromesse sono piccole.
Dall'altro lato abbiamo invece la possibilità per l'attaccante di compromettere utenti disomogenei per il volume di traffico che generano.
Il punto di forza di questo approccio non risiede nelle prestazioni massime raggiungibili, che sono sempre inferiori di quelle ottenibili rispetto all'altro caso, ma nella stabilità e consistenza

delle stesse lungo il range dei gradi di contesa, eccezion fatta per cache di dimensioni contenute ed alto grado di contesa.
Esistono anche delle situazioni particolari in cui i risultati tendono a sovrapporsi; in generale si tratta di quei casi dove il trade-off per l'attaccante è tra cache più o meno contenitive e comportamento degli utenti più o meno omogeneo, perciò egli può basare le sue azioni su considerazioni riguardanti ad esempio i costi.
In generale un buon compromesso per l'attaccante è rappresentato da 2 helper con cache da 600 contenuti; questo tipo di soluzione garantisce buon vantaggi per Consumer 1 e Producer1, senza peggiorare sensibilmente le metriche importanti per Helper 1.

5.3 - Riassunto del capitolo e considerazioni finali

In questo capitolo sono stati presentati i dati di tutte le simulazioni svolte, con spiegazioni, analisi e commenti approfonditi a corredo.
Tutto ciò è stato svolto per ciascuna configurazione di test che si è ritenuto avere l'importanza di comparire nel presente documento; dove invece questo non sia stato determinato come necessario sono state asserite le ragioni per l'omissione e differenze minori sono state comunque segnalate ove presenti.
Il capitolo è stato aperto da 4 domande aperte, le cui risposte sono contenute nei dati oggetto della presentazione.
Tuttavia, è parso calzante riproporre qui tali domande, ognuna corredata della rispettiva risposta esplicita.
Tali domande erano:

- D1: Date le ipotesi di partenza, può un attaccante ottenere un vantaggio dalla compromissione delle cache degli utenti?
- R1: Sì. Una volta che l'attaccante ha effettuato le proprie scelte sul numero e tipo di utenti da compromettere, commisurandolo al quantitativo di contenuti da distribuire e successivamente preso possesso di questi nodi, egli può ottenere vantaggi in termini sia di hit ratio, che di consegne medie da parte del produttore, che di tempi medi di permanenza in rete dei contenuti, ma in determinate situazioni anche del ritardo sperimentato dagli utenti serviti dal produttore malevolo.
- D2: Se sì, come si può quantificare questo vantaggio?
- R2: Il vantaggio può passare da inesistente ad estremamente significativo, ma in generale si attesta su livelli piuttosto elevati, soprattutto per quanto riguarda l'hit ratio ed il tempo di permanenza in rete dei contenuti.
  Sotto la maggior parte delle condizioni poi anche il numero medio di consegne che il produttore deve effettuare subisce riduzioni drastiche, fino all'azzeramento nel migliore dei casi, salvo una iniziale fase di disseminazione.
  Infine, negli scenari in cui il produttore si trova più lontano dagli utenti, anche il ritardo può essere efficacemente ridotto fino ad un quarto del suo valore iniziale.
- D3: Quali sono le condizioni che maggiormente lo influenzano?
- R3: I maggiori fattori che contribuiscono all'efficacia dell'attacco comprendono soprattutto grado di contesa offerto dagli utenti compromessi, capienza delle loro cache e diversità od omogeneità del loro comportamento.

In seconda battuta viene invece il numero dei nodi compromessi, con l'efficacia che aumenta notevolmente nel passaggio da nessuno ad un helper e da uno a 2 helper, ma con effetti minori da 2 a 3 helper, configurazione che porta vantaggi soprattutto in certi casi maggiormente di nicchia.
Infine non è stato rilevato alcun effetto prodotto dalla presenza o meno di cache nei router di rete, per le configurazioni testate nel presente studio.

- D4: In che misura gli utenti compromessi sono disturbati dagli effetti collaterali?
- R4: Le metriche scelte per rappresentare le performance degli helper non hanno mostrato grossi scostamenti dall'assenza di attacco alla loro compromissione, soprattutto per quanto riguarda il ritardo.
  L'hit ratio è stato valutato più severamente compromesso solo in casi particolarmente sfavorevoli per la vittima.

Capitolo 6 – Contromisure

Nel precedente capitolo è emersa l'efficacia dell'attacco dal punto di vista del produttore malevolo.
Grazie allo sfruttamento di risorse messe a disposizione di ogni utente di una CCN, come le cache ad ogni livello della gerarchia ed un meccanismo di movimento dei dati in rete interamente gestito e controllato da ogni utente, l'attaccante, una volta assunto il controllo di un numero sufficiente di utenti compromessi è in grado di migliorare la qualità del servizio offerto dal produttore malevolo, senza al contempo deprimere oltremodo quella delle proprie vittime.
Proprio il fatto che il perpetratore sia in grado di mettere in piedi un attacco senza generare un alto volume di traffico e senza sfruttare vere e proprie falle dell'architettura, ma solo le normali modalità con cui essa serve le richieste e memorizza i contenuti rende difficile il progetto di contromisure efficaci.
Il comportamento di un nodo sotto l'effetto di tale attacco potrebbe essere indistinguibile da uno legittimo.
Supponendo ad esempio che i contenuti richiesti costituissero i brani di una playlist, essi lo sarebbero probabilmente con alta frequenza.
È doveroso però sottolineare che contromisure specifiche ed efficaci contro gli attacchi di tipo *false locality* sono state presentate [15], anche se non tengono conto di alcune peculiarità dello specifico attacco presentato nel capitolo 4.
La discussione riguardante questo tipo di contromisura verrà svolta, come per le altre, più approfonditamente nel paragrafo a parte che le verrà dedicato nel prosieguo del capitolo.
Andiamo quindi ad illustrare una per una tutte le contromisure che sono state considerate potenzialmente efficaci per contrastare questo attacco.
Come vedremo per alcuni di questi casi si tratta di meccanismi che possono consentire il rilevamento dell'attacco, altre costituiscono pratiche che, se adottate possono renderlo completamente inefficace.

6.1 - Honeypots

Questo metodo permette di arrivare al rilevamento dell'attacco tramite il camuffamento di una stazione di monitoraggio per utente comune.
Quando questo utente fittizio viene compromesso, il suo traffico può essere monitorato e quello malevolo immediatamente emergerà e potrà essere perciò contrassegnato come tale e studiato, allo scopo di risalire poi al produttore malevolo, ai suoi contenuti ed infine, seguendone le tracce, agli altri utenti compromessi o direttamente al centro di controllo dell'attaccante, nel caso l'applicazione malevola lasciasse trasparire qualche traccia.
La honeypot dovrebbe essere chiaramente mascherata in modo appetibile per l'attaccante (produrre poco traffico ma in maniera credibile e avere un'ampia cache) e posizionata in maniera visibile, oppure si dovrebbe essere in grado di forzarne la compromissione.
Supponiamo infatti che sia noto che l'attaccante utilizzi le pratiche del social engineering per la compromissione; la honeypot dovrebbe essere nelle condizioni di poter essere adescata in

questa maniera, il che presuppone la disponibilità di qualche informazione sull'attacco in corso e sull'attaccante.

6.2 - Osservazione dei pattern di traffico

Dal momento che la curva della domanda dei contenuti richiesti dagli utenti legittimi segue la legge di Zipf, mentre il pattern delle richieste legittime è regolare e deterministico, un router di bordo che fosse in grado di costruire questo tipo di statistica per ogni namespace ed ogni interfaccia potrebbe identificare con successo un pattern non corrispondente a quello regolare.
Tuttavia un attaccante potrebbe facilmente inficiare tale meccanismo cambiando il pattern delle richieste da regolare a tipo ZIpf.
Non è chiaro però che tipo di effetto questo potrebbe avere sull'efficacia dell'attacco, perché se fosse utilizzato un pattern che si conformasse alla distribuzione di tipo Zipf, allora i contenuti maggiormente richiesti e quindi con maggiori chance di permanenza in memoria sarebbero quelli alla cima della distribuzione.
Anche l'eventuale trade-off che si presenterebbe all'attaccante tra efficacia e rilevabilità sarebbe tutto da definire.

6.3 - Contromisure per l'attacco di *false locality* presentate in letteratura

In [15] vengono proposte e testate due misure atte a contrastare gli attacchi di *false locality*, rimanendo però nell'ambito delle reti IP.
Da una parte viene presentato un approccio in cui il tentativo è quello di identificare gli attaccanti, mentre dall'altra i contenuti che sono oggetto delle richieste.
Secondo il primo vengono misurate, per ciascun contenuto ed in un periodo sufficientemente lungo, numero di richieste ripetute e la frazione che queste richieste costituiscono rispetto al totale che ha ricevuto un hit della cache e quegli utenti per cui queste due metriche eccedono valori di soglia vengono marcati come attaccanti.
Nel secondo la metrica misurata per ciascun contenuto e all'interno di un certo intervallo di tempo è la percentuale di quel tempo che il contenuto passa nella cache; ciò serve a misurare il grado di permanenza del contenuto nella cache.
Dopo aver identificato i file con un alto tempo di permanenza nella cache, per ciascuno di essi viene misurato il rapporto fra numero di richieste ripetute e numero di IP unici che hanno effettuato queste richieste.
Dato che, secondo gli autori, solitamente gli utenti tendono a richiedere un file solo un numero limitato di volte, se la metrica appena descritta risulta superiore ad una certa soglia il contenuto viene marcato come avente popolarità alterata.
Le prove condotte poi dimostrano in generale l'efficacia di questi meccanismi.
Le questioni che rimangono aperte riguardano l'adattabilità di queste soluzioni al caso presentato nel presente lavoro.
Senz'altro non si possono utilizzare metriche che tengano conto degli indirizzi IP, visto che tale concetto non esiste in CCN, e l'applicazione di tali metodi per essere efficace dovrebbe avvenire per ogni utente.
Perciò, a livello concettuale, si potrebbe immaginare un sistema di protezione locale a ciascun utente di una CCN, con possibilità di monitoraggio della cache per ricavarne metriche che

possano identificare l'attacco; queste metriche possono essere quelle presentate in [15] e sopracitate, dove al posto degli indirizzi IP potrebbe essere usato un identificativo per le varie applicazioni che sono in esecuzione sul dispositivo dell'utente.
In questa maniera, se si utilizzasse il secondo dei metodi presentati, un'applicazione dal comportamento simile a quello descritto nel capitolo 4 potrebbe essere smascherata per avere una frequenza di richieste ripetute troppo elevata per una singola applicazione.
L'implementazione e prova dell'efficacia di questo tipo di contromisura nello specifico caso qui presentato è al di fuori degli scopi del presente testo, ma può costituire un interessante sviluppo per un futuro lavoro.
In [7] viene sviluppato un meccanismo di identificazione degli attacchi di *cache pollution* che si basa sulla misura delle statistiche caratteristiche di un tipo di traffico normale e il rilevamento di anomalie che violino in maniera troppo evidente queste metriche.
Gli autori ne provano l'efficacia attraverso una serie di sperimentazioni, anche se qualche dubbio rimane su come questo sistema si comporti se gli utenti dovessero cambiare repentinamente l'insieme dei contenuti che richiedono, situazione che potrebbe portare a segnalazioni di falsi positivi.

6.4 - Periodico svuotamento delle cache

Come affermato in [15], per un attaccante che operi un tentativo di *cache pollution* è più complicato, lungo e dispendioso il processo di riempimento iniziale delle cache piuttosto che il successivo mantenimento dello stato creato.
Dato questo fatto, un'operazione che potrebbe fornire una forma di disincentivazione al perpetratore dell'attacco potrebbe consistere nella forzata ripetizione del processo di riempimento della cache, tramite il suo periodico svuotamento, poiché aumenterebbe il numero medio di consegne effettuate dal produttore e diminuirebbe i tempi medi di permanenza in rete dei contenuti.
Ovviamente tale misura negherebbe alcuni dei benefici del caching e potrebbe portare ad un degrado delle prestazioni per gli utilizzatori legittimi, perciò occorrerebbe trovare una corretta misura che offra un equilibrio fra sicurezza ed esperienza d'uso degli utenti.
Inoltre, non esistendo ancora implementazioni pratiche di una CCN, non risulta chiaro che cosa costituisca una cache, se l'intera memoria di un dispositivo oppure una sua parte dedicata.
Se fosse la prima delle ipotesi ad essere implementata, è chiaro che questa contromisura non potrebbe svuotare l'intera memoria del dispositivo e sarebbe pertanto inattuabile.

6.5 - Invio di interest digitalmente firmati

Supponiamo che esista, all'interno dei dispositivi degli utenti, un'area definita "sicura", in cui risiedono tutte le applicazioni dell'utente e che qualsiasi messaggio proveniente da quell'area sia cifrato ed autenticato tramite firma digitale.
Se esistesse, in ogni dispositivo, un sistema di protezione che semplicemente impedisse l'invio di qualsiasi messaggio non proveniente da tale area, quindi non autenticato e se fosse impraticabile per l'applicazione malevola progettata dall'attaccante accedere a quest'area, agli interest da questa emessi sarebbe impedito l'invio.

Questo genere di contromisura renderebbe più complicato lo sfruttamento del nodo in caso di compromissione tramite una vulnerabilità del sistema e la conseguente installazione ed esecuzione del codice malevolo mantenendo l'inconsapevolezza dell'utente.
Infatti, anche se l'attaccante fosse in grado di installare l'applicazione senza l'intervento dell'utente, questa non risiederebbe all'interno dell'area sicura, pertanto sarebbe del tutto inutile.
Per contro, se invece tale compromissione avvenisse attraverso pratiche di *social engineering*, l'attaccante potrebbe essere in grado di ottenere il permesso da parte dell'utente di installare l'applicazione nell'area sicura, legittimando quindi il traffico da essa generato.
Seppur possibile come concetto, questa soluzione presenterebbe molte difficoltà pratiche, soprattutto riguardanti le modalità della sua implementazione in sistemi reali, dal momento che una delle maggiori difficoltà nella realizzazione di sistemi di sicurezza è come evitare che la loro stessa implementazione possa contenere falle da sfruttare.
Oltre alle difficoltà pratiche, un altro limite di tale contromisura risiede nell'aspetto etico; autenticare il traffico di un dispositivo significherebbe renderlo effettivamente tracciabile in maniera inequivocabile nel suo percorso lungo la rete, con tutte le implicazioni ben note che ciò può avere.
Poco importa che il traffico venga autenticato per dispositivo e non per utente, dal momento che i dispositivi stessi stanno diventando sempre più legati all'individuo che li utilizza.
Per concludere, questa contromisura si può ritenere come valida ma molto più orientata ad un modello di rete compartimentalizzata e sicura, piuttosto che libera e semplice per l'utente.

Capitolo 7 – Conclusioni e sviluppi futuri

Questo lavoro ha presentato prima di tutto una panoramica sulle caratteristiche ed il funzionamento delle architetture CCN e soprattutto dei suoi aspetti di sicurezza ed a quanto questi rendano le CCN maggiormente sicure contro le minacce e gli attacchi che stanno affliggendo le reti basate su IP.
Tale affermazione non significa che non sia possibile mettere in campo nuove forme e tipologie di attacco che prendano di mira gli aspetti specifici delle CCN.
Dopo una carrellata delle differenti tecniche di aggressione verso una CCN descritte nella letteratura sull'argomento, è stato illustrato un approccio nuovo al tema delle *cache pollution*.
I contributi innovativi del presente lavoro pertanto si possono riassumere nei seguenti punti:

- Lo spostamento dell'obiettivo dell'attacco dalle cache dei router della rete a quelle locali degli utenti.
- La conduzione di un'indagine articolata ed approfondita volta alla quantificazione del problema presentato in termini di vantaggi ottenibili dal produttore e dai suoi consumatori e di impatto sulle performance degli utenti compromessi.

L'indagine di cui sopra ha portato ai seguenti risultati:

- Vantaggi molto sensibili possono essere ottenuti dal produttore in termini di hit ratio degli utenti da esso serviti, di numero medio di contenuti da esso consegnati e di tempo medio di permanenza in rete di tali contenuti.
- Vantaggi sensibili, se la distanza tra produttore e i propri utenti è elevata, possono essere ottenuti in termini di ritardo medio visto da tali utenti.
- Sono emersi interessanti trade-off che l'attaccante deve considerare nel momento della scelta delle vittime da compromettere.
- I danni causati alle prestazioni degli utenti possono essere considerati generalmente bassi, a parte qualche caso maggiormente punitivo, specialmente quando è una singola cache ad essere compromessa.

Infine sono state discusse alcune possibili contromisure che potrebbero essere impiegate per il rilevamento e la neutralizzazione dell'attacco descritto.
Futuri interessanti studi potrebbero andare sia nella direzione di una maggiore comprensione del problema, in particolare della sua analisi per un più ampio spettro di casi, sia verso l'implementazione delle misure preventive qui discusse o lo sviluppo di nuove.

Bibliografia

[1] L. Breslau, P. Cue, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and zipf-like distributions: Evidence and implications,” in INFOCOM, 1999, pp. 126–134

[2] A. Mahanti, C. Williamson, and D. Eager, “Traffic analysis of a web proxy caching hierarchy,” IEEE Comm. Surveys and Tutorials, vol. 14, no. 3, pp. 416–23, 2000

[3] http://www.chromium.org/spdy/spdy-whitepaper

[4] Mahmudul Hoque, Syed Obaid Amin, Adam Alyyan, Beichuan Zhang, Lixia Zhang, Lan Wang, “NLSR: Named-data Link State Routing Protocol”, The 3rd ACM SIGCOMM Workshop on Information-Centric Networking, August 2013

[5] Giuseppe Rossini, Dario Rossi, “A dive into the caching performance of Content Centric Networking”, Technical report, Telecom ParisTech, 2011

[6] Mengjun Xie, Indra Widjaja, Haining Wang, “Enhancing Cache Robustness for Content-Centric Networking”, In Proceedings of the IEEE Infocom, 2012

[7] Mauro Conti, Paolo Gasti, Marco Teoli, “A lightweight mechanism for detection of cache pollution attacks in named data networking”, Computer Networks, 2013

[8] http://www.nsnam.org

[9] Predrag Jelenkovic´, “Least Recently Used Caching with Zipf’s Law Requests”, In The Sixth INFORMS Telecommunications Conference, Boca Raton, Florida, 2002

[10] G. Carofiglio, M. Gallo, and L. Muscariello “Bandwidth and Storage Sharing Performance in Information Centric Networking” , in ACM SIGCOMM, ICN Worskhop, 2011, pp. 1–6.

[11] Giovanna Carofiglio, Massimo Gallo, Luca Muscariello, and Diego Perino, “Modeling Data Transfer in Content-Centric Networking”, in ITC, 2011

[12] J. Choi, J. Han, E. Cho, T. T. Kwon, and Y. Choi, “A Survey on Content-Oriented Networking for Efficient Content Delivery”, IEEE Communications Magazine, pp. 121–127, 2011

[13] I. Psaras, R. G. Clegg, R. Landa, W. K. Chai, and G. Pavlou, “Modelling and Evaluation of CCN-caching Trees”, FIP Networking, 2011

[14] Tobias Lauinger: “Security & Scalability of Content-Centric Networking,” Master’s Thesis, TU Darm- stadt, Darmstadt, Germany and Eurécom, Sophia-Antipolis, France, September 2010.

[15] L. Deng, Y. Gao, Y. Chen, A. Kuzmanovic, Pollution attacks and defenses for internet caching systems, Comput. Netw. 52 (5) (2008) 871 935–956.

[16] C. Ghali, G. Tsudik, E. Uzun, Needle in a Haystack: Mitigating Content Poisoning in Named-Data Networking, in SENT 2014, NDSS Workshop on Security of Emerging Networking Technologies, February 23, 2014, San Diego, USA

[17] P. Gasti, G. Tsudik, E. Uzun, and L. Zhang, “DoS and DDoS in Named-Data Networking,” ArXiv e-prints, Tech. Rep. 1208.0952, August 2012

[18] A. Compagno, M. Conti, P. Gasti, G. Tsudik, Poseidon: Mitigating Interest Flooding DDoS Attacks in Named Data Networking, Tech. Rep. 1303.4823, ArXiv e-prints (March 2013)

[19] A. Afanasyev, P. Mahadevan, I. Moiseenko, E. Uzun, and L. Zhang. Interest flooding attack and countermeasures in Named Data Networking. In *IFIP Networking*, 2013

[20] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, Cache privacy in named-data networking, in: ICDCS 2013, 2013

[21] http://www.networkcomputing.com/ipv6/ipv6-adoption-here-at-last/240166360

[22] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in ACM CoNEXT, 2009, pp. 1–12

[23] D. Smetters, V. Jacobson, “Securing Network Content”, in Technical report, PARC, October 2009

[24] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. Lecture Notes in Computer Science, 2009:46, 200