Reducing the Flow Completion Time Tail in Datacenter Networks

Étudiant: PHU Ba Duong, *Promotion 19 l'IFI* Encadrant: Dr.NGUYEN Hong Quang, *l'IFI*

***Abstract*—Ce document fourni une vue assez complète du problème de la longe traîne du temps de l'achèvement du flux qui a été présenté dans l'article *DeTail : Reducing the Flow Completion Time Tail in Datacenter Networks*. En effet, cet article a présenté le problème de la longe traîne existant dans les Datacenters et nous a proposé une solution appelé DeTail pour le résoudre. Dans ce rapport, nous allons voire plusieurs approches différentes qui ont été concentrés dans les recherches à traiter le problème de la longe traîne dans Datacenter.**

***Index Terms*—flow completion time, long tailed, tail latency, low latency.**

I. Introduction

L'Article *DeTail : Reducing the Flow Completion Time Tail in Datacenter Networks* nous a proposé une solution pour un problème inévitable appelé la longe traîne de l'achèvement du flux dans Datacenter autrement dit, le temps nécessaire à finir une tâche dépasse un délai prédéterminé. Le problème venant des applications Web, moteurs de recherche qui se développent de façons sophistiquées. Cela rend qu'une page typiquement peut-être exigent plusieurs flux de intra-datacenter. Pour répondre au développement, les deux conceptions les workflows de Partition-Aggregator et les workdlows séquentielles qui sont utilisées très largement dans Datacenter. Et, l'utilisation de ces conceptions a produit le problème de longe traîne. L'apparition de longe traîne provoquait des conséquences très graves. Les auteurs de cet article en fait, ont donné trois causes essentiels entraînant l'augmentation de temps d'attente. Et puis, ils ont proposé une solution appelé DeTail. DeTail, en fait, est une nouvelle conception de cross-layer visant à résoudre le problème de longe traîne. DeTail est une approche efficace car, il peut attaquer à trois causes de longe traîne, qui sont cités en haut. Toutefois, DeTail exige trops changement d'infrastructure de Datacenter. Cela devinera plus difficile lorsqu'on veut déloyer DeTail dans un grand Datacenter. Dans ce rapport, nous allons voire en détaillé les causes de longe traîne, et puis les impacts des long traîne. Ensuite, nous allons voire les approches différentes qu'il faut concentrer sur la recherche à résoudre le problème de longe traîne de l'achèvement du flux. Et les propositions sont proposées auprès de DeTail. En fin, nous allons faire une comparaison entre eux pour trouver une bonne proposition qui peuvent implémenter dans les Datacenters.

II. Longe Traîne et les impacts de la longe traîne sur workflows

La longue traîne est un problème de la distribution. En utilisant la CDF (Cumulative Distribution Function) F(t) = P(x < t) (P: probabilité) si la valeur de t n'augmente pas trop quand $0 < F(t) < 1$. Lorsque F(t) atteint presque 1 la valeur de t augmente rapidement. Cela appelé la longue traîne. La longue traîne dans cette recherche sont les contraintes de temps d'attente pour que les worknodes finissent leurs tâches qui jouent le rôle de t dans la formule au-dessus
. Autrement dit, nous sommes en train d'observer le problème quand le temps nécessaire à finir une tâches qui dépasse un délai prédéterminé. Avant de faire voire les impacts de longe traîne, nous revoyons les deux conceptions qui sont utilisé dans le Datacenter sont *workflows de Paritition-Aggregator* et *workflows séquentielles*.

A. Workflows de Partition-Aggregator

Partition-Aggregator workflows se compose trois parties Top-level aggregator(TLA), mid-level aggregator(MLA) et worker node. TLA reçois les requêtes. Il divise les calculs exigés aux MLAs. MLA divise les calculs aux worknodes. Worknode calcule et retourne les résultats à MLA. MLA combinne les résultats et les retourne à TLA. Le longe traîne se produit lorsqu'il y a des worknodes qui ne peuvent pas finir leurs tâches à temps et envoient leurs résultats à MLA. Dans ce cas, MLA traite ce problème, soit combiner ses résultats et l'envoyer à TLA pour répondre le délai mais il y aura une mauvais présentation de la contenu du résultat, soit attendre les travails de worknodes restants pour avoir une bonne présentation de la contenu mais dépasser le temps permis.

B. Workflows séquentielles

Dans workflow séquentielles, Un seul front-end serveur télécharge le données de back-end serveurs pour chaque création de page. La requête prochaine dépende de la requête précédant. Chaque donnée récupérée utilise un flux. Dans ce cas, le nombre d'extraction de données transmis par une page sont diminués lors que la longe traîne de temps d'achèvement de flux se produit.

L'apparition de longe traîne amène des conséquences très sérieux. La performance du Datacenter sont diminuée. Les résultats obtenus de Datacenter ont mauvais présentation (dans le problème du site internet), et diminution de la satisfaction d'utilisateur.

III. Les causes de la longue traîne

Dans cette section, nous allons voire les trois causes principales de longe traîne qui sont mentionnés dans l'article *DeTail : Reducing the Flow Completion Time Tail in Datacenter Networks*. Ce sont la perte et retransmission, l'absence de la priorité du trafic et l'inégalité de l'équilibrage de charge (uneven load balancing).

A. Pertes et retransmissions

Cette sous-partie, nous nous sommes intéressons à deux valeurs RTT(Round-Trip time) et RTO (Round transmission timeout). Lorsque une pertes se produit, les émetteurs doit attendre une durée de RTO pour détecter la pertes. Probablement, RTO est toujours supérieur RTT. La distance de temps d'attente entre RTO et RTT est la racine de longe traîne.

B. Absence de priorité

En effet, les flux échangés dans les Datacenters sont divisés en deux types. Ce sont les flux courts et les flux longes. Les flux courts sont les flux avec les taille de 100Kb à 1MB. Les flux longes sont les flux avec la taille de 1MB à des GBs. Normalement, les flux courts sont les flux de latence-sensible (c'est à dire : ils doivent arriver à la destination à temps) et les flux longes sont les flux de latence insensible. Lorsque, la congestion se produit, les flux courts sont mis dernière les flux longe dans la fil d'attente. Par conséquente, les flux courts sont ratés leurs délais. Cela entraînera le longe traîne.

C. L'inégalité de l'équilibrage de charge (load balancing)

Actuellement, les Datacenters sont construits en utilisant le topologies Clos tel que fat-tree[3]. Par conséquent, il y a plusieurs chemin qui transmettent les paquets d'une source à une destination avec le même coût. Cependant, une fois que la congestion se produit, les livraison des paquets ne sont pas transmis en utilisant les autres chemins libres. Les chemins très chargés va donc continuer à recevoir les paquets à transmettre tandis que les chemins libre n'ont pas été utilisé pour transmettre les paquets. Ainsi, les paquets ne peuvent pas arrivé à la destination. La longe traîne est donc inévitable.

IV. Solution

Nous avons besoin des solutions pour diminuer ou bien supprimer la longe traîne de l'achèvement du flux dans les Datacenters. Une solution efficace attaquera un ou plusieurs causes des trois causes en haut. En fait, il y avait plusieurs propositions qui permet de réduire le longue traîne tel que DCTCP, D3, Hedera, MPTCP, DIBS, Lastnane, CP, D3, D2TCP. Pourtant, en résumé, toutes les propositions sont basé sur les quatre approches différentes suivantes :

1. ***Réduction de longueur de fil d'attente***
2. ***Accélération de retransmission***
3. ***Donner la priorité à les flux courts***
4. ***Utilisation de multi-chemins***

Nous allons voire en détaillé pas à pas les quatre approches et les propositions utilise ces approches

A. Réduction de longueur de fil d'attente

Cette approche vise à limiter le nombre des paquets sont mis dans la fil d'attente en définissant un seuil. Lorsque le longueur de fil d'attente de switch ou end-host occupé dépasse une seuil, ils envoient une signal au switch précédent pour réduire la vitesse d'envoi les paquets. Les deux propositions appliquent cette approches sont DCTCP(Data center TCP), HULL(High-bandwidth Ultra Low Latency). Ils sont basé sur Explicit Congestion Notification(ECN). ECN est une extension de Protocole d'Internet et de Protocole de Contrôle de Transmission.ECN permet de notifier les congestion sans avoir besoin de refuser les paquets.

B. Accélération de retransmission

Cette approche vise à la perte et retransmission. Une fois que la perte se produit nous avons besoin d'intervenir le processus de retransmission. Par exemple, au lieu d'attendre une durée de RTO pour détecter la perte et retransmettre le paquet. Nous avons retransmette les paquet avant que le délai de RTO arrive. Il y a DIBS(Detour induced buffer sharing), Lastnane et CP qui apliquent cette approche. DIBS est déployé dans les switches. Il change le port à transmettre les paquets quand ces paquets sont retransmis. Lastnane génère un message au switch et l'envoie à émetteur quand un paquet est refusé. CP utilise l'entête de paquet refusé et récepteur reconnaît l'absence des données. Il va envoyer SACK pour la demande de la retransmission.

C. Donner la priorité à les flux courts

Cette approches vise à la cause de l'absence de la priorité entre les flux longes et les flux courts. Au lieu de mettre les paquets de flux courts au derrière de flux, les flux courts sont ajouté les priorité plus haut que les flux longes. Les flux courts seront donc traités avant le traitement des flux longes. Suivre cette approches, nous avons des propositions remarquable et divisées en deux sous-types.

1.Délai comme priorité

Nous avons D3(Deadline driven delivery), PDQ(Preemptive Distributed Quick) et D2TCP(Deadline-aware Data center TCP)

2.Priorité assigné par les applications

Nous avons DeTail et pFabric. Ces deux propositions auront besoin de changer les éléments de réseau.

D. Utilisation de multi-chemins

La dernière approches, les multi-chemins sont utilisés efficacement. Les chemins libres seront utilisé une fois que les congestions se produisent dans autres chemins. Cette approche est appliqué dans les propositions DeTail, et RepFlow. DeTail utilise une fonction de hachage pour

trouve le port de sortie. RepFlow est une technique fonctionnant en dupliquant les paquets et les transmettre dans autre sens. Par conséquent, la probabilité de congestions causé par les flux longes sera diminué exponentiellement. Par exemple, entre les hôte A et hôte B, si nous avons trois chemins avec le même coût et la probabilité de la congestion sur un seul chemin est $\frac{1}{3}$, alors la probabilité de la congestion sur deux chemins à même temps sera $\frac{1}{9}$.

La proposition la plus remarquable est DeTail. Cette proposition qui s'est présenté dans deux approches. Il est la solution proposé dans l'article. Nous allons donc le voire en détaillé DeTail est une proposition visant à trois cause de la longe traîne. Fondamentalement, DeTail est une nouvelle conception de stack de cross-layer. La figure suivante illustre la coneeption de DeTail[1]

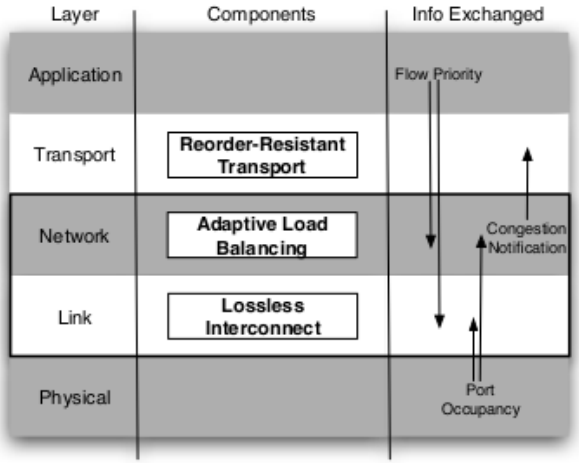

Figure 1. Nouvelle conception de cross-layer

Observons cette figure, nous voyons les couches sont ajoutés les composants appropriés. La couche de liaison a le composant *Lossless Interconnect* La couche de réseau a le composant *Adaptive Load Balancing*, la couche de transport a le composant *Reoder-Resistant Transport*.
Lossless Interconnect : Assurer qu'il n'y a pas de perte
Adaptive Load Balancing : Assurer l'égalité du trafic
Reoder-Resistant Transport : Réorganiser les paquets dérangés
Pour implémenter cette nouvelle conception, DeTail a besoin d'une conception de switch CIOQ[1]

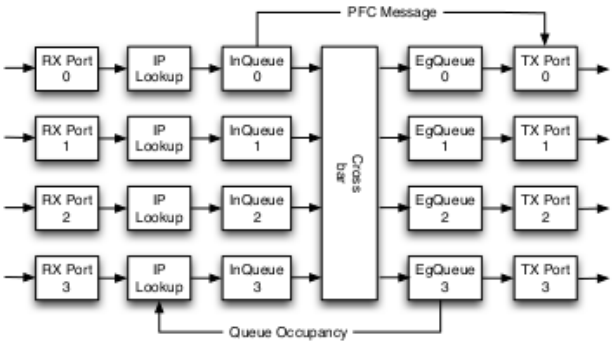

Figure 2. Nouveau switch utilisé par DeTail

Ce switch est déployé les trois composants comme décrit au-dessus . En effet, DeTail et aussi les autres propositions toujours exigeront des changements des éléments de Datacenter actuel pour fonctionner et réduire le longe traîne. Dépendre de l'approches utilisé et la façon d'utiliser l'approche. Il y aura des changement adaptative. La table suivante est une comparaison les objectives et les changement nécessaires une fois qu'on applique les propositions [2]

Approches	Proposition	Objectif		Modification à		
		FCT	Délai	TCP	Switches	Applications
1. Réduction du longueur de fil d'attente	DCTCP	mean		x		
	HULL	mean		x	x	
2. Accélération de retransmission	D3	non	x	x	x	
	PDQ	mean	x	x	x	
	D2TCP	tail	x	x		
	MCP	tail	x	x		
	pFabric	mean&tail	x	x	x	x
	DeTail	tail	x	x	x	x
3. Donner la priorité pour flux courts	Repflow	Mean& tail				x
4. Utilisation de multi-chemins	DIBS	tail		x	x	
	FastLane	tail		x	x	
	CP	tail		x	x	

Table 1 : La comparaison en terme d'objectif et les modifications nécessaires des propositions[2]

L'apparition de DeTail a l'air une bonne solution pour le problème de longue traîne. Alors que, DCTCP, D3 ne visent pas à l'inégalité de l'équilibrage de charge (load balancing). Hedera n'a pas améliorer la performance pour flux courtes dans la création de page. MPTCP ne suffit pas pour protéger les paquets perdus et des retransmission, ne effectue que efficacement pour la taille de paquet de 70kB.[1] DeTail vise à traiter les trois causes de longe traîne. Cependant, en voyant cette table, DeTail exigera les trois modifications TCP, Switches, et Application. Ces modifications seront très difficile une fois qu'on veut déployer DeTail dans un grande Datacenter. Cela va changer l'infrastructure de Datacenter. Une proposition, qui peux diminuer efficacement la longe traîne de Datacenter, et facile à déployer est vraiment nécessaire actuellement. Pour surmonter la limité de DeTail, les deux propositions nous voulons présenter dans ce rapport qui sont récemment publiées appelées RepNet et FA-DCTCP.

E. RepNet

RepNet est développé de RepFlow et surmonter les désavantage de RepFlow. Plus exact, elle est une combinaison de RepFlow et RepSYN. Essentiellement, RepNet fonctionne comme RepFlow, mais, RepSYN est utilisés pour éviter le case quand RepFlow qui duplique les paquets transmis, provoque la saturation de réseau.

F. FA-DCTCP

FA-DCTCP est basé sur DCTCP et vise à distinguer les traitement entre les flux courts et les flux longes. FA-DCTCP fonctionne dans deux étapes[4].

1. Identifier les flux courts et les flux longes
2. Concevoir un algorithme de contrôle de congestion pour les flux courts et les flux longes

1) Identifier les flux courts et les flux longes[4]: Pour identifier les types des flux, on propose deux approches

- *Basé sur le port :* Les flux utilise les ports différentes auront les types de flux correspondante.
- *Utiliser un seuil dans buffer de TCP socket :* Définir un seuil de longueur de flux. Si le flux échangé est supérieur un seuil, nous considérons qu'il est de type des flux longes.

2) Algorithme de contrôle de congestion pour les flux courts et les flux longes: [4] - Après avoir identifié les types de flux. FA-DCTCP utilise un algorithme de contrôle de congestion s'approprie pour chaque type de flux

Algorithm 1 FA-DCTCP congestion window calculation [5]

CurrentCwnd : Valeur actuelle de la fenêtre de congestion
Alpha : Fraction des paquets qui sont marqué par ECN
NewCwnd : La fenetre de congestion calculé par l'algorithm
Entrée : < CurrentCwnd, Alpha >
Sortie : < NewCwnd >
// si flux est de type de longe
if FLOW-TYPE == ELEPHANT **then**
//calculer CWND comme DCTCP, en utilisant la loi de
//diminution de DCTCP
CwndNew = CurrentCwnd x (1 - (Alpha/2))
else //si flux est de type court, calculer
//CWND en utilisant diminution de FA-DCTCP
if (Alpha > .6) **then**
//durant le haut niveau de congestion, réduire la fenêtre
//de congestion à taux plus haute
CwndNew = CurrentCwnd x (1 - ((Alpha) 2 /2))
else
//durant le haut niveau de congestion, réduire la fenêtre
//de congestion à taux plus basse
CwndNew = CurrentCwnd x (1 - ((Alpha) 3 /2))
end if
end if

Selon cet algorithme, la valeur de CwndNew est calculé dépendre de le niveau de congestion. Pour voire les advantages de RepNet et FA-DCTCP par rapport au DeTail, nous faisons une comparaison entre trois propositions DeTail, RepNet et FA-DCTCP

Proposition	Objectif	Modification à		
		TCP	Switches	Application
DeTail	Longe traîne	x	x	x
RepNet	Longe traîne	x		x
FA-DCTCP	Longe traîne	x		

Table 2 : La comparaison entre DeTail, RepNet et FA-DCTCP

En regardant la table de comparaion, nous voyons que l'implémentation de RepNet et FA-DCTCP dans les Datacenter peuvent devenir plus facile que DeTail. Car, ils n'exigent pas trop de changement de l'infrastructure de Datacenter comme DeTail. De plus, avec FA-DCTCP, on a besoin seulement d'ajouter environ 50 lignes de code dans TCP[4]. Ainsi, l'implémentation de FA-DCTCP deviendra beaucoup plus pratique dans les grands Datacenters.

V. Conclusion

Nous avons découvert un des problèmes très essentiels dans les Datacenters intitulé la longe traîne du temps de l'achèvement du flux. Nous avons vu les cause de ce problèmes et les quatre approches différentes pour donner les propositions différentes visant à diminuer la longe traîne. Nous avons observé DeTail en détaillé, une des propositions efficace résolvant le problème de longe traîne et fait la comparaison entre les propositions. Et aussi, nous avons trouvé que DeTail est une solution complète in terme de traitement le longe traîne de l'achèvement des flux. Cependant, en terme de pratique, DeTail n'est pas une proposition efficace dans un grand Datacenter. Puisque, DeTail exige le changement des éléments du réseau. Nous avons aussi trouvé deux autre solutions RepNet et FA-DCTCP qui peuvent surmonter les inconveniences de DeTail et peuvent s'adapter aux Datacenters actuels.

References

[1] David Zats,Tathagata Das,Prashanth Mohan,Dhruba Borthakur, Randy H. Katz, *"DeTail: Reducing the Flow Completion Time Tail in Datacenter Networks"*, 3rd ed, University of California at Berkeley March 15, 2012.
[2] Shuhao Liu, Hong Xu, Zhiping Cai, *"Low Latency Datacenter Networking: A Short Survey"*, University of Hong Kong 31 july 2014.
[3] Shuhao Liu, Wei Bai, Hong Xu , Kai Chen , Zhiping Cai, *"RepNet: Cutting Tail Latency in Data Center Networks with Flow Replication"*, University of Hong Kong 26 jan 2015 pages 1-5.
[4] Sijo Joy, *"Improving Flow Completion Time and Throughput in Data Center Networks"*, Thesis, University of Ottawa, Canada Feb 2015, pages 1-34
[5] Sijo Joy, Amiya Nayak, *"Improving Flow Completion Time for Short Flows in Datacenter Networks"*, School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada.