Cannot Take My Allocation: Enforcing Fairness by Considering Demand and Payment in Clouds

Gyuyeong Kim and Wonjun Lee
Dept. of Computer Science and Engineering
Korea University
WCU Future Network Optimization Technology Center (FNOT)
wlee@korea.ac.kr

***Abstract*—Cloud data centers provide cost-effective and elastic resource management environments through multi-tenancy. However, we observe that existing policies cause unfair bandwidth allocation. A max-min based policy allocates lower bandwidth to a tenant whose the size of flows is mostly large, and DCTCP falls short of high utilization and minimum performance guarantee. Moreover, commodity clouds are vulnerable to denial of service (DoS) attacks, which can be attempted by a malicious tenant. The aforementioned deficiencies decrease the motivation to use clouds and revenue of cloud providers. To address the problem, we propose *Spiderweb*, a fair network sharing mechanism for cloud data centers, which allocates bandwidth to tenants in proportion to the amount of demand and payment during oversubscription. Our proposal delivers both a cloud-specialized fair network sharing environment and robustness against DoS attacks. We show that our proposal meets the desirable sharing properties including proportional fairness, performance isolation, and high utilization through analytical allocation scenarios.**

I. Introduction

Cloud data centers deliver an elastic resource utilization environment based on a pay-as-you-go policy [1]. Tenants in data centers run their services by managing virtual machines (VM) dynamically under the necessity. However, with growth of network performance sensitive services, oversubscription has been a major issue of today clouds. A measurement paper observes that 86% of the links in data centers show oversubscription lasting at least 10 seconds and even over 100 seconds [2]. To deal with resource competitions among tenants under oversubscription, an elaborate network sharing policy is imperative. Unfortunately, commodity clouds exclude network resources from the considerations due to the variability and distributed nature of network resources.

In existing regimes, both benign and malicious tenants can cause unfair bandwidth allocation. Consider that a provider applies the fair share of DCTCP [3] during oversubscription. In spite of a large payment, a tenant cannot obtain gratifying allocation. Moreover, a possibility of resource dissipation exists to a tenant, whose demand is low. Max-min fairness also cannot be a substitution, because it always assigns lower allocation to a tenant, whose dominant flows are large-sized. The other possible unfair allocation scenario is a DoS attack. A malicious tenant can occupy dominant share by sending unlimited UDP packets to arbitrary destination VMs [4], [5]. The aforementioned unfair allocation scenarios diminish the motivation of tenants to use clouds, and it leads decrease of a provider's revenue as a result.

To mitigate the problem, there have been proposals in the literature. Rate control based approaches limit transmission of sender VMs [6], [4]. Their solutions capacitate data centers avoid additional oversubscription, but do not support any fairness mechanism. On the other hand, some efforts assign entity weights to a source VM [5] or a tenant [7]. Entity weights limit the volume of data that a VM or tenant can send across time, but it causes unfair allocation to destination VMs. Moreover, a malicious tenant can increase the allocation by faking the number of destination VMs. Resource reservation model of the existing approaches [8], [9] presents efficiency of static quota allocation, but they have a vulnerability to low link utilization.

In this paper, we propose *Spiderweb*, a fair network sharing policy. Our approach trims oversubscription in proportion to the amount of demand and payment level of tenants. *Spiderweb* allocates higher bandwidth to a tenant, whose payment is larger and demand is lower. By considering a socio-economic aspect that people pay attention to cost much than profit, we derive allocation from a viewpoint of loss. Although there exist prior efforts that take payment into account, our solution reflects either resource demand of tenants as a primary factor to clarify where the responsibility of oversubscription lies. Considering payment and demand also remedy the shortcomings of clouds for DoS attacks. Because one who desires to send unlimited packets, he must spend a huge sum of money. Moreover, in spite of a large payment, a malicious tenant cannot obtain a dominant share due to the fact that larger demand is disadvantageous during oversubscription. *Spiderweb* achieves high utilization, hence a satisfied tenant donates surplusage to unhappy tenants.

The remainder of the paper is organized as follows. We inspect the solutions of precursors (§II), and show the details of unfair allocation scenarios (§III). We describe overview of *Spiderweb* (§IV) and details (§V). After that, we present how our proposal achieves fairness and DoS robustness (§VI). Lastly, we conclude our work with future directions (§VII).

II. Related Work

The rapid growth of clouds has led academia to study network performance in multi-tenant environments.

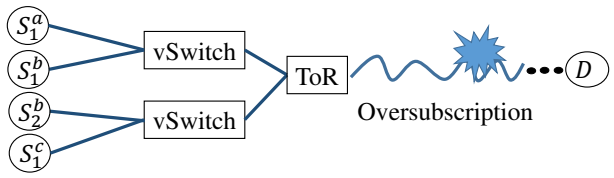


Fig. 1. A single oversubscription scenario

TABLE I
ALLOCATION OF DCTCP AND MAX-MIN FAIRNESS

Policy	**Tenant** a	**Tenant** b	**Tenant** c	**Sum**
Origianl Demand	30Mbps	110Mbps	10Mbps	150Mbps
DCTCP	25Mbps	50Mbps	25Mbps	100Mbps
Max-min Fairness	30Mbps	60Mbps	10Mbps	100Mbps

Seawall [5] paved the way on a cloud network sharing research by considering that data centers are vulnerable to a DoS attack of a malicious tenant. The effort reserves bandwidth in proportion to the number of destinations to source VMs. However, since a malicious tenant can occupy much bandwidth by increasing the number of destinations artificially, tenants are still exposed to security threats. Netshare [7] is similar to Seawall, but it assigns weights to a tenant based on payment to seek network-wide fairness. However, controlling network-wide allocation to handle a link level oversubscription has the possibility of high complexity. Moreover, increasing allocation on an unoversubscripted link is not a direct solution for an oversubscripted link. In contrast with Seawall and Netshare, we design a system attribute oversubscription to tenants directly in proportion to the amount of demand and payment level. Therefore, a malicious tenant cannot hurt the others even if he has enormous both demand and payment.

Gatekeeper [6] and EyeQ [4] decrease transmission of a sender, who causes oversubscription using UDP. Both of them are a rate-limiting system, but EyeQ has a different feature on using 10% of a link capacity as a buffer to avoid oversubscription. However, the benefit of oversubscription avoidance from abandonment of the amount of capacity needs more convintive evidences. Besides, we believe the policies of precursors are unfair, because the latest sender always become a suspect when a link has less available bandwidth. We trim a oversubscripted link by rate-limiting, but *Spiderweb* specifies how much each tenant will be charged on oversubscription.

Oktopus [9] and Secondnet [8] virtualize data centers for efficient resource management. They have a strong point to reduce network management complexity. Both solutions support static quota allocation, but a reservation method cannot use underutilized links. By considering traffic burstiness, the deficiency of a static allocation policy is substantial. *Spiderweb* is a work-conserving and hose model based solution. Therefore, a tenant can utilize an unused link capacity.

III. UNFAIR ALLOCATION SCENARIOS

In this section, we show our observations concerning unfair bandwidth allocation by presenting example scenarios. We categorize unfair allocation scenarios into two cases. One is unfair bandwidth allocation among benign tenants, and the other is unfairness by a malicious tenant. Unfairness of both scenarios decrease satisfaction of tenants and revenue of a provider as a result.

A. Allocation among benign tenants

Let us consider a single link oversubscription at any given time, where three tenants a, b, and c exist as described in Figure 1.

S_m^u, a m the source VM S of a tenant u transmits packets to an arbitrary destination VM D. An outgoing link from a Top of Rack (ToR) switch falls in oversubscription due to immoderate inputs. DCTCP and a max-min fairness based policy are available options in the literature. We plot the original demand and allocation under each regime in Table I. Note that we consider a per-tenant allocation.

DCTCP adheres to the fairness policy of TCP, which assigns identical bandwidth to all flows. For the sake of simplicity, we assume that each VM S_m^u has as many flows. We observe that equal share causes degradation of link utilization because tenant c obtains 2.5x allocation much higher than the original demand. A max-min fairness allocates bandwidth to a flow having lower demand by priority. If tenant b runs a service, whose the size of flows are mostly large, a provider allocates lower bandwidth to tenant b. In an extreme case, there exists the possibility of starvation if the other services are comprised of small-sized flows.

B. Unfairness by a malicious tenant

Sharing underlying infrastructure of clouds implies potential security problems [10], [5], [4]. A possible threat is a DoS attack, which has been a recurring problem on the Internet. Cloud data centers are much vulnerable to DoS attacks than Internet due to its higher density. In the absence of performance isolation, disturbing the others is an undemanding job in commodity clouds. A simple DoS attack abuses a connectionless protocol, UDP. Let a VM of tenant a sends unlimited UDP packets, whereas VMs of tenant b and c transmit TCP packets. During oversubscription, VMs using TCP decrease their rate by implicit feedback, but VMs of tenant b stick to his transmission rate. Consequentially, a malicious tenant a obtains dominant allocation.

IV. OVERVIEW OF THE SYSTEM

We propose *Spiderweb*, a fair cloud network sharing policy, that clarifies where the responsibility of oversubscription lies and neutralizes malicious behaviors.

Our system leverages multi-bit feedback for oversubscription signaling. Both vSwitches and switches monitor their ingress/egress traffic to sense oversubscription. A switch on duty calculates the fair allocation of involved tenants, and generates a feedback packet. A feedback packet contains not

only oversubscription notification but also the quantity of allocation. The switch transmits a packet to the vSwitch on a sender VM. Since a vSwitch is on the top of a physical machine, we exploit a vSwitch as a hypervisor. After signaling, a vSwitch decreases the transmission rate of a sender VM. A network omits feedback mechanism if the switch detecting oversubscription is a vSwitch. At the same time, a switch blocks additional flow access by admission control to avoid additional loss.

Calculating allocation of *Spiderweb* is comprised of four phases. The first phase is loss fairness, which attributes oversubscription to tenants in proportion to their demands. The second is payment proportinality that allocates bandwidth as much as a tenant paid. Owing to payment level, tenants are incentivized during oversubscription. Since fairness depends on a viewpoint of providers, we seek a balanced point of tradeoff with a coefficient α for the third step. A provider can adjust the value of α, and we inspect the impact of the coefficient in Section VI. The last step redistributes spare resources to unhappy tenants for high utilization.

In our regime, we assume a tenant specifies expected average amount of demand for job completion. We consider job completion times as a significant performance metric in clouds for the recent growth of partition/aggregation applications. A tenant is happy if a job is performed before deadline. Therefore, we chase meeting deadline even during oversubscription. The quantity of demand is derived from Equation 1.

$$RequiredDemand = \frac{TotalDemand}{JobDeadline} \tag{1}$$

Note that source VMs increase their sending rate as so as possible to reduce job completion times, hence our approach does not fix job completion times to deadline.

V. SPIDERWEB

In this section, we describe the details of the system. We first discuss the desirable properties for cloud network sharing systems, and describes allocation computation and enforcement mechanism.

A. Design Rationale

Before describing our mechanism, we narrate the desirable properties to design an algorithm.

Loss-based Fairness Our design aims to compute not the quantity of allocation during overusbscription, but the quantity of loss. We believe that the level of concern on loss of tenants is much higher by considering social-economic aspects of clouds.

Performance Isolation We believe that a case of minimum performance isolation is neutralizing unfair allocation by a malicious tenant. The motivation stems from the difficulties of perfect isolation due to capacity limitations and bursty packet drops in data centers. In this context, the simplest definition of performance isolation is *"Do not harm the others."*.

Demand Fidelity Each tenant has an obligation to decrease allocation in proportion to his demand during oversubscription. Because an allocation (e.g. identical bandwidth allocation) is unfair to a tenant, whose demand has only marginal influence to oversubscription.

Payment Fidelity Cloud providers should embrace payment level of tenants to mitigate oversubscription, because clouds obey a pay-as-you-go policy. Note that there is a trade-off between demand and payment fidelity properties, and we show how they could be balanced in Section V.

High Utilization If a tenant obtains higher allocation than his demand, a provider reallocates surplusage to unhappy tenants. The property aims to maximize tenant satisfaction so that a provider can increase his revenue.

B. Bandwidth allocation

We narrate how a switch determines the allocation step by step. The process consists of four steps, oversubscription volume calculation, payment proportionality, balancing by a coefficient α, and surplusage redistribution. The computation process is designed to meet the aforementioned desirable properties. To illustrate our proposal, we exploit a scenario in Table I where three tenants a, b, and c are involved in a oversubscripted link with demands, 30Mbps, 110Mbps, and 10Mbps, respectively.

Step 1: By the first phase, a switch calculates oversubscription volume V, which is defined in Equation 2.

$$\begin{aligned} V &= \frac{\Sigma d - C}{\Sigma d} \\ &= 1 - \frac{C}{\Sigma d} \end{aligned} \tag{2}$$

Overusbscription volume V is a portion of exceedances on a link. In the example, aggregate demand Σd is 150Mbps and we assume a link capacity C as 100Mpbs, then V is 33%, which is derived from $(150-100)/150$. The volume reflects demand of tenants during oversubscription. A network decreases sending rate of a source having the utmost demand. Therefore, a malicious tenant cannot carry out his attack successfully.

$$\begin{aligned} x_i &= d_i - d_i \cdot V \\ &= d_i(1 - V) \end{aligned} \tag{3}$$

Now we compute temporary allocations of the tenants x_i, which considers V. By Equation 3, the values of x_i are 20Mbps, 73Mbps, and 7Mbps, respectively.

Step 2: A temporary outcome of Step 1 still has a possibility of unfairness. In spite of a large payment due to requirements of services, a tenant cannot obtain sufficient allocation. To address the issue, we consider the payment level of tenants. Commodity cloud data centers do not state specified network performance guarantee by payment level. Therefore, we consider an SLA, which has the five payment levels of network performance. The payment level works as an entity weight for tenants, $p_i/\Sigma p_i$. Consider each tenant has the payment level, 5, 3, and 1. By Equation 4, the expected allocations y_i are 56Mbps, 33Mbps, and 11Mbps, respectively.

$$y_i = C \cdot \frac{p_i}{\Sigma p_i} \tag{4}$$

Algorithm 1 Surplusage redistribution algorithm

```
/* g_i is an allocation of a tenant u_i */
while there is a remain surplusage VM do
    while there is a tenant to check do
        if u_i is an unhappy tenant then
            allocate surplusage in proportion to the payment
            update the value of g_i
            if u_i is a overprovisioned tenant then
                gather surplusage from u_i
                update the value of g_i
            end if
        end if
    end while
end while
```

Note that we neglect allocations of Step 1 here, and we combinate Step 1 and 2 in next Step 3.

Step 3: The allocated bandwidth of a tenant i, z_i is an association of cost fairness (Step 1) and payment proportionality (Step 2). We leave room to a provider for choices of balance by a coefficient α as we draw in Equation 5. Because everyone's opinion to fairness is bound to be subjective. In the example, we consider a provider, who fixes the α as 0.5. Tenants in the example obtain bandwidth Z 38Mbps, 53Mbps, 9Mbps, respectively.

$$z_i = (1 - \alpha) \cdot x_i + \alpha \cdot y_i \qquad (0 \leq \alpha \leq 1) \tag{5}$$

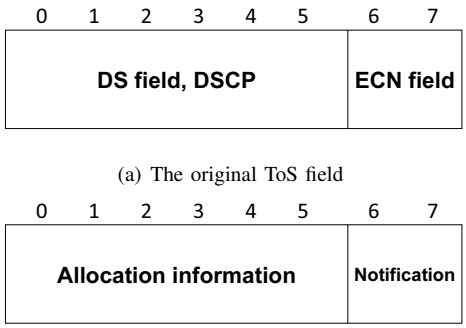
Step 4: To achieve high utilization, we design a policy to redistribute over-provided resources to unsatisfied tenants. While the demand of tenant a is 30Mbps, but a policy assigns 38Mbps. Tenant b and c still require more allocation, and our mechanism distributes the amount of spare resources, 8Mbps (38Mbps-30Mbps) in proportion to the payment level. The algorithm repeats redistribution until all spare bandwidth are consumed. We illustrate the process in Algorithm 1. The outcome of Algorithm 1, G is a final allocation. For our example, G are 30Mbps, 60Mbps, and 10Mbps, respectively.

C. Enforcement

Our enforcement performs based on a feedback mechanism. The logic is simple. When oversubscription occurs, a switch sends a feedback packet to source VMs. Then, a vSwitch limits sending rate of VMs. However, implementing efficient feedback delivery is a challenge.

To implement our system, we exploit multi-bit feedback mechanism. Implicit feedback of current data centers is vulnerable, because packet drops cause high cost to performance-sensitive applications. An available option is ECN feedback, but it only notifies whether oversubscription occurs. We wrap both oversubscription singal and allocation information in a feedback packet.

8-bit ToS field in packet header is a candidate for implementation. The DS field and ECN feedback field consist of ToS field as described in Figure 2 (a). Since the ToS field is not directly utilized over many years, we shape it as a container for our feedback mechanism, and Figure 2 (b) depicts our modification. One might point out that multi-bit feedback is not suitable because it hurts standardized form. However, the modification does not harm general network operations because the ToS field is an uninfluential component. Indeed, IETF has updated the structure of ToS field for many years. We believe tuning only ECN feedback field without modifying the DS field is available by moderate designs. Thanks to multi-bit feedback implementation using only one more bit on Internet [11], we already confirmed the possibility. However, we remain the work for cloud data centers as the future work.



(a) The original ToS field

(b) The modified ToS field for our system

Fig. 2. Packet header modification for multi-bit feedback

Admission control prevents tenants from additional damage. A switch involved in oversubscription rejects any flow access of tenants. Due to burstiness of traffic volume, we configure switches to release closure state when a link is stabilized. Note that a provider can determine the threshold, which indicates a link is unoversubscripted.

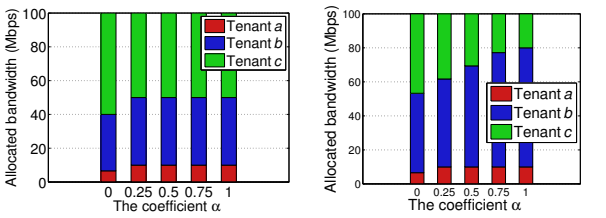
VI. Analysis

In this section, we now analyze our proposal. We consider a single oversubscripted link where three tenants exist. Since our solution is for oversubscription, we focus on a peak demand at any given time. Table II depicts demand and payment of the scenarios. We first explore the impact of the coefficient α, and compare *Spiderweb* against alternatives. Our analysis shows that *Spiderweb* achieves proportional fairness, high utilization, and neutralizes malicious behaviors of a tenant.

A. The impact of the coefficient α

For the first analysis, we inspect the influence by the coefficient α. Figure 3 presents allocation of the coefficient fluctuation. We observe that a policy allocates bandwidth in proportion to demand and payment. Thanks to surplusage redistribution, allocation increase of both tenant a and b is halted. In Figure 3 (a), a provider supports performance guarantee of tenant a and b regardless of the value of α over 0.25. Tenant c suffers from performance degradation due to small payment.

Now we set tenant b and c have identical demand for a next scenario to analyze malicious behavior neutralization. Figrue 3 (b) shows clearly the fluctuation of allocation. Consider tenant



(a) Allocation of the scenario 1 (b) Allocation of the scenario 2

Fig. 3. The impact of the coefficient α

TABLE II
SCENARIOS FOR THE ANALYSIS

Parameters	**Tenant** a	**Tenant** b	**Tenant** c
Demand 1	10Mbps	40Mbps	70Mbps
Demand 2	10Mbps	70Mbps	70Mbps
Payment	10	5	1

c is malicious, and he has indiscernible traffic. We observe that an attack of tenant c fails due to its high demand and low payment. To disturb the performance of the others, tenant c must either decrease his demand or increase payment. However, decreasing rate indicates the abandonment of an attack, and increasing payment causes immoderate expensive cost.

B. Comparison against alternatives

Comparing *Spiderweb* against available alternatives, max-min fairness and DCTCP emphasizes vulnerability of the alternatives. In Figure 4, only tenant a is happy under allocation of comparison targets. Max-min fairness provides about two-thirds of the demand for tenant b and c, and DCTCP shows the worse allocation. The allocation underutilizes over 20Mbps of a capacity for a case of tenant a. On the other hand, allocation of *Spiderweb* presents high-utilized bandwidth distribution.

Fairness is a subjective theme, hence we believe an optimal value of α does not exist. The aforementioned analysis presents that the efficiency of allocation varies by parameter fluctuation. We believe a provider can determine suboptimal value by considering the traffic characteristics of the services in his data centers.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed fair network sharing in cloud data centers. We noted the five desirable properties including loss-based fairness, performance isolation, demand fidelity, payment fidelity, and high utilization. By considering properties, we proposed a policy, which allocates bandwidth in proportion to demand and payment level, and neutralizes a DoS attack. Our analysis shows that our approach meets the aforementioned properties, and implies that a provider can determine suboptimal value of a coefficient α by considering the service characteristics in a data center.

For the future work, we will focus on extension of *Spiderweb* to cover performance issues. Especially, the reduction

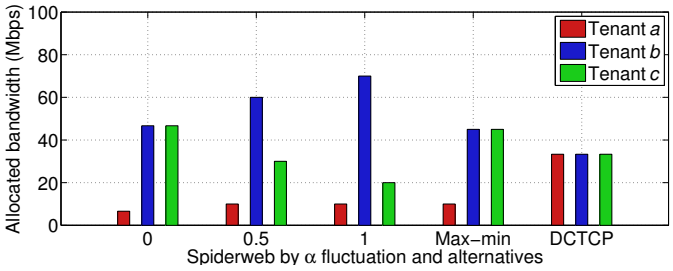


Fig. 4. Comparison against alternatives

of the average job completion times is a major problem. Moreover, since this paper does not present implementation of the proposal, we will implement the system, and analyze some policy design issues including performance guarantee granularity.

ACKNOWLEDGMENTS

This research was jointly sponsored by MEST, Korea under WCU (R33-2008-000-10044-0), MSIP, Korea under grant (No. 2013R1A2A2A01014000), MEST, Korea under Basic Science Research Program (2011-0012216), MKE/KEIT, Korea under the IT R&D program [KI001810041244, SmartTV 2.0 Software Platform], and MSIP, Korea under ITRC NIPA-2013-(H0301-13-3001), and NRF of Korea under the project of Global Ph.D. Fellowship conducted from 2012. Wonjun Lee is the corresponding author.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010.

[2] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *Proc. of ACM IMC*, 2009.

[3] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center tcp (dctcp)," in *Proc. of ACM SIGCOMM*, 2010.

[4] V. Jeyakumar, M. Alizadeh, D. Mazières, B. Prabhakar, and C. Kim, "Eyeq: practical network performance isolation for the multi-tenant cloud," in *Proc. of USENIX HotCloud*, 2012.

[5] A. Shieh, S. Kandula, A. Greenberg, C. Kim, and B. Saha, "Sharing the data center network," in *Proc. of USENIX NSDI*, 2011.

[6] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. Guedes, "Gatekeeper: supporting bandwidth guarantees for multi-tenant datacenter networks," in *Proc. of USENIX WIOV*, 2011.

[7] V. T. Lam, S. Radhakrishnan, R. Pan, A. Vahdat, and G. Varghese, "Netshare and stochastic netshare: predictable bandwidth allocation for data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 3, pp. 5–11, 2012.

[8] C. Guo, G. Lu, H. J. Wang, S. Yang, C. Kong, P. Sun, W. Wu, and Y. Zhang, "Secondnet: a data center network virtualization architecture with bandwidth guarantees," in *Proc. of ACM Co-NEXT*, 2010.

[9] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron, "Towards predictable datacenter networks," in *Proc. of ACM SIGCOMM*, 2011.

[10] Y. Mundada, A. Ramachandran, and N. Feamster, "Silverline: data and network isolation for cloud services," in *Proc. of USENIX HotCloud*, 2011.

[11] Y. Xia, L. Subramanian, I. Stoica, and S. Kalyanaraman, "One more bit is enough," in *Proc. of ACM SIGCOMM*, 2005.