International Journal of Combined Research & Development (IJCRD)
eISSN:2321-225X;pISSN:2321-2241 Volume: 2; Issue: 4; April-2014

A Superior Prediction-Based System for Cloud Bandwidth and Cost Reduction

Ganapathi Avabrath
Student, RITM, Bangalore-64
gansudabi@yahoo.com

Prof. Vaishali Kulkarni
Associate Professor, RITM, Bangalore-64
Vrkulkarni.cs@gmail.com

Abstract:

Cloud computing is a fast growing field which is arguably a new computing paradigm. In cloud computing, computing resources are provided as services over the internet and users can access resources by paying. When we are trying to minimize the cloud cost, transmission cost plays a major role. In this paper, we present PACK (Predictive ACKs) mechanism, a novel end-to-end traffic redundancy elimination (TRE) system, designed for cloud computing customers. Unlike previous solutions, PACK does not require the server to continuously maintain clients' status. This makes PACK very suitable for pervasive computation environments that combine client mobility and server migration to maintain cloud elasticity. PACK is based on a novel TRE technique, which allows the client to use newly received chunks to identify previously received chunk chains, which in turn can be used as reliable predictors to future transmitted chunks.

Categories and Subject Descriptors

C.2.m [Computer-Communication Networks]: Miscellaneous

General Terms

Algorithms, Design, Measurement

Keywords: ***Caching, Cloud computing, Network optimization, Traffic redundancy elimination***

1. INTRODUCTION

Cloud computing is emerging style of delivery in which applications, data and resources are rapidly provisioned as standardized offerings to users with a flexible price. The cloud computing paradigm has achieved widespread adoption in recent years. Its success is due largely to customers' ability to use services on demand with a pay-as-you go [6] pricing model, which has proved convenient in many respects. Low costs and high flexibility make migrating to the cloud compelling. Cloud computing is the long dreamed vision of computing as a utility, where users can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources. By data outsourcing, users can be relieved from the burden of local data storage and maintenance. Traffic redundancy and elimination approach is used for minimizing the cost.

Traffic redundancy stems from common end-users' activities, such as repeatedly accessing, downloading, distributing and modifying the same or similar information items (documents, data, web and video). TRE is used to eliminate the transmission of redundant content and, therefore, to significantly reduce the network cost. In most common TRE solutions, both the sender and the receiver ex- amine and compare signatures of data chunks, parsed according to the data content prior to their transmission. When redundant chunks are detected, the sender replaces the transmission of each redundant chunk with its strong signature [16, 23, 20]. Commercial TRE solutions are popular at enterprise networks, and involve the deployment of two or more proprietary protocol, state synchronized middle-boxes at both the intranet entry points of data centres and branch offices, eliminating repetitive traffic between them (e.g., Cisco [15], Riverbed [18], Quantum [24], Juniper [14], Blue- coat [7], Expand Networks [9] and F5 [10]).While proprietary middle-boxes are popular point solutions within enterprises, they are not as attractive in a cloud environment. First, cloud providers cannot benefit from a technology whose goal is to reduce customer bandwidth bills, and thus are not likely to invest in one. Moreover, a fixed client-side and server-side middle-box pair solution is inefficient for a combination of a mobile environment, which detaches the client from a fixed location, and cloud-side elasticity which motivates work distribution and migration among data centres. Therefore, it is commonly agreed that a universal, software-based, end-to-end TRE is crucial in today's pervasive environment [4, 1]. This enables the use of a standard protocol stack and makes a TRE within end-to-end secured traffic (e.g., SSL) possible.In this paper, we show that cloud elasticity calls for a new TRE solution that does not require the server to continuously maintain clients' status. First, cloud load balancing and power optimizations may lead to a server-side process and data migration environment, in which TRE solutions that require full synchronization between the server and the client are hard to accomplish or may lose efficiency due to lost synchronization. Moreover, the popularity of rich media that consume high bandwidth motivates CDN solutions, in which the service point for fixed and mobile users may change dynamically according to the relative service point locations and loads.

[1]We refer as *cloud customers* to organizations that export services to the cloud, and as *users* to the end-users and devices that consume the service

Finally, if an end-to-end solution is employed, its additional computational and storage costs at the cloud-side should be weighed against its bandwidth saving gains. Clearly, a TRE solution that puts most of its computational effort on the cloudside2 may turn to be less cost-effective than the one that leverages the combined client-side capabilities. Given an end-to-end solution, we have found through our experiments that sender-based end-to-end TRE solutions [23, 1] add a considerable load to the servers, which may eradicate the cloud cost saving addressed by the TRE in the first place. Moreover, our experiments further show that current end-to-end solutions also suffer from the requirement to maintain end-to-end synchronization that may result in degraded TRE efficiency.In this paper, we present a novel receiver-based end-to-end TRE solution that relies on the power of predictions to eliminate redundant traffic between the cloud and its end-users. In this solution, each receiver observes the incoming stream and tries to match its chunks with a previously received chunk chain or a chunk chain of a local file. Using the long-term chunks meta-data information kept locally, the receiver sends to the server predictions that in- cloud chunks' signatures and easy to verify hints of the sender's future data. Upon a hint match the sender triggers the TRE operation, saving the cloud's TRE computational effort in the absence of traffic redundancy.

Offloading the computational effort from the cloud to a large group of clients forms a load distribution action, as each client processes only its TRE part. The receiver-based TRE solution addresses mobility problems common to quasi-mobile desktop/laptops computational environments. One of them is cloud elasticity due to which the servers are dynamically relocated around the federated cloud, thus causing clients to interact with multiple changing servers. Another property is IP dynamics, which compel roaming users to frequently change IP addresses. In addition to the receiver- based operation, we also suggest a hybrid approach, which allows a battery powered mobile device to shift the TRE computation over- head back to the cloud by triggering a sender-based end-to-end TRE similar to [1].To validate the receiver-based TRE concept, we implemented, tested and performed realistic experiments with PACK within a cloud environment. Our experiments demonstrate a cloud cost reduction achieved at a reasonable client effort while gaining additional bandwidth savings at the client side. The implementation code, over 25,000 lines of C and Java, can be obtained from [22]. Our implementation utilizes the TCP Options field, supporting all TCP-based applications such as web, video streaming, P2P, email, etc.We propose a new computationally light-weight chunking (fingerprinting) scheme termed PACK chunking. PACK chunking is a new alternative for Rabin fingerprinting traditionally used by RE applications. Experiments show that our approach can reach data processing speeds over 3 Gbps, at least 20% faster than Rabin fingerprinting.

In addition, we evaluate our solution and compare it to previous end-to-end solutions using terabytes of real video traffic consumed by 40,000 distinct clients, captured within an ISP, and traffic obtained in a social network service for over a month. We demonstrate that our solution achieves 30% redundancy elimination without significantly affecting the computational effort of the sender, resulting in a 20% reduction of the overall cost to the cloud customer.

The paper is organized as follows: Section 2 reviews existing TRE solutions. In Section 3 we present our receiver-based TRE solution and explain the prediction process and the prediction-based TRE mechanism. In Section 4 we present optimizations to the receiver-side algorithms. Section 5 evaluates data redundancy in a cloud and compares PACK to sender-based TRE. Section 6 details our implementation and discusses our experiments and results.

2. RELATED WORK

Several TRE techniques have been explored in recent years. A protocol-independent TRE was proposed in [23]. The paper describes a packet-level TRE, utilizing the algorithms presented in [16]. Several commercial TRE solutions described in [15] and [18], have combined the sender-based TRE ideas of [23] with the algorithmic and implementation approach of [20] along with protocol specific optimizations for middle-boxes solutions. In particular, [15] describes how to get away with three-way handshake between the sender and the receiver if a full state synchronization is maintained. [3] and [5] present redundancy-aware routing algorithm. These papers assume that the routers are equipped with data caches, and that they search those routes that make a better use of the cached data. A large-scale study of real-life traffic redundancy is presented in [11], [25] and [4]. Our paper builds on the latter's finding that "an end to end redundancy elimination solution, could obtain most of the middle-box's bandwidth savings", motivating the benefit of low cost software end-to-end solutions.

Wanax [12] is a TRE system for the developing world where storage and WAN bandwidth are scarce. It is a software-based middle-box replacement for the expensive commercial hardware. In this scheme, the sender middle-box holds back the TCP stream and sends data signatures to the receiver middle-box. The receiver checks whether the data is found in its local cache. Data chunks that are not found in the cache are fetched from the sender middle-box or a nearby receiver middle-box. Naturally, such a scheme incurs a three-way-handshake latency for non-cached data.

EndRE [1] is a sender-based end-to-end TRE for enterprise networks. It uses a new chunking scheme that is faster than the commonly- used Rabin fingerprint, but is restricted to chunks as small as 32-64 bytes. Unlike PACK, EndRE requires the server to maintain a fully and reliably synchronized cache for each client. To adhere with the server's memory requirements these caches are kept small (around 10 MB per client), making the system inadequate for medium-to-large content or long-term redundancy. EndRE is server specific, hence not suitable for a CDN or cloud environment.

To the best of our knowledge none of the previous works have addressed the requirements for a cloud computing friendly, end-to- end TRE which forms PACK's focus.

[2] We assume throughout the paper that the cloud-side, following the current Web service model, is dominated by a sender operation.

3. THE PACK ALGORITHM

For the sake of clarity, we first describe the basic receiver-driven operation of the PACK protocol. Several enhancements and optimizations are introduced in Section 4.

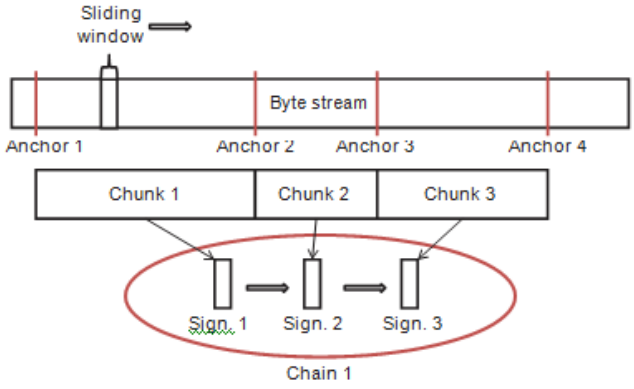
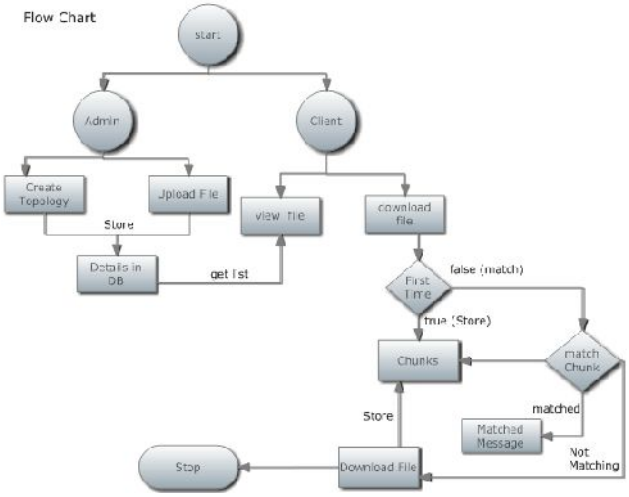


Figure 1: From stream to chain

The stream of data received at the PACK receiver is parsed to a sequence of variable size, content-based signed chunks similar to [16][20]. The chunks are then compared to the receiver local storage, termed chunk store. If a matching chunk is found in the local chunk store, the receiver retrieves the sequence of subsequent chunks, referred to as a chain, by traversing the sequence of LRU chunk pointers that are included in the chunks' metadata. Using the constructed chain, the receiver sends a prediction to the sender for the subsequent data. Part of each chunk's prediction, termed a hint, is an easy to compute function with a small enough false-positive value, such as the value of the last byte in the predicted data or a byte-wide XOR checksum of all or selected bytes. The prediction sent to the receiver includes the range of the predicted data, the hint and the signature of the chunk. The sender identifies the predicted range in its buffered data, and verifies the hint for that range. If the result matches the received hint, it continues to perform the more computationally intensive SHA-1 signature operation. Upon a signature match, the sender sends a confirmation message to the receiver, enabling it to copy the matched data from its local storage.



A) Receiver Side Chunk Storage

Predictive ACK uses the new continuous chains scheme that described in Fig. 1, in that every chunk are related to all other chunks by their recent received way of order. The Predictive ACK receivers have to keep a chunk storage, which it's a very large size cache of chunks and their metadata. Chunk's metadata includes the data chunk's signature and a single pointer to the successive chunk in the recent received stream that contain this chunk. Cache and index technique are employed to efficiently maintain and retrieve the every stored chunks and its signature and the chains created by traverse the chunk pointers.

B) Receiver Side Algorithm

The arrival of a new information, the receiver side system that respective signature for every chunk and see the match in its local (temporary) chunk storage. If the chunk's match is founded, the receiver side determines whether it's a part of a formerly received chunk chain, using the chunks' metadata (data about data) otherwise If affirmative, the receiver send a prediction to the sender side for the several new expected chain chunks. It carries a beginning point in the byte stream that is offset and the identity of several subsequent chunks.

Proc. 1 Receiver Segment Processing

```
1. if segment carries payload data then
2.     calculate chunk
3.     if reached chunk boundary then
4.         activate predAttempt()
5.     end if
6. else if PRED-ACK segment then
7.     processPredAck()
8.     activate predAttempt()
9. end if
```

Proc. 2 predAttempt()

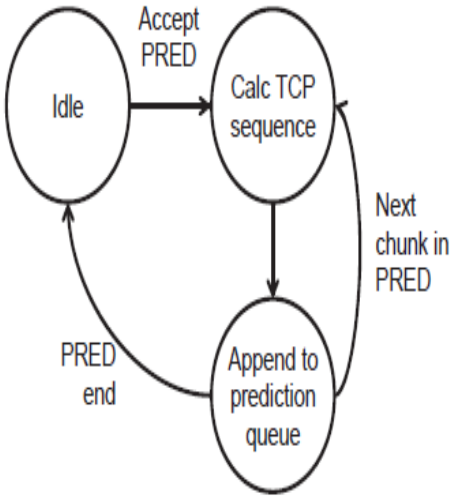
```
1. if received chunk matches one in chunk store then
2.    if foundChain(chunk) then
3.       prepare PREDs
4.       send single TCP ACK with PREDs according to Options
         free space
5.       exit
6.    end if
7. else
8.    store chunk
9.    link chunk to current chain
10. end if
11. send TCP ACK only
```

Proc. 3 processPredAck()

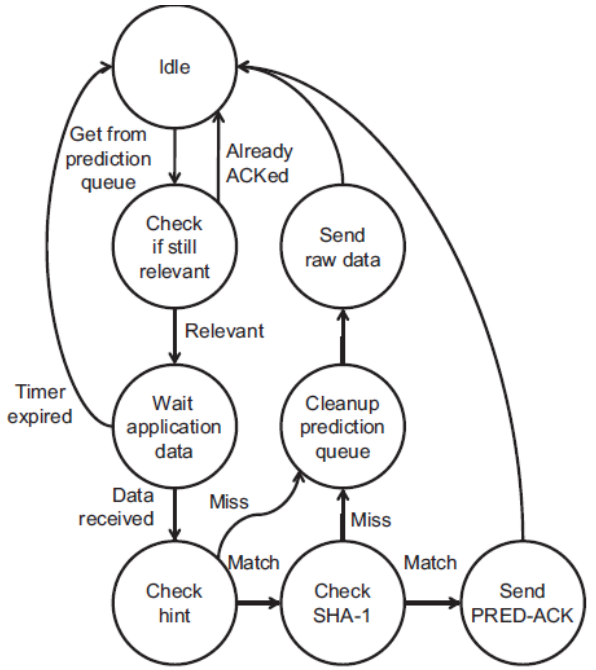
```
1. for all offset ∈ PRED-ACK do
2.    read data from chunk store
3.    put data in TCP input buffer
4. end for
```

C) Sender Side Algorithm

Sender receives a Predictive message from the receiver side and it tries to compare the received predictions to its buffered yet to be sent Information. For every prediction, the sender has to determine that corresponding TCP range of sequence and verifies it. If that hint match, the sender measures the more computationally intensive Secure Hash Algorithm- 1 signature for the predicted information range and match the result to the signature received in the Predictive message of data. In this case if the hint does not same, a computationally expansive operation is saved. If the two Secure Hash Algorith-1 signatures compare, the sender can safely assume that the receiver's prediction method is absolutely correct. and, it replace the entire outgoing buffered data with a Predictive ACK message.



(a) Filling the prediction queue



(b) Processing the prediction queue and sending PRED-ACK

4. OPTIMIZATIONS

A) Adaptive Receiver Virtual Window

Predictive ACK enable the receiver side to locally capture the sender data when a local or temporary copy is available, thus eliminating the requirement to send this information through the network. In this term the receiver's fetching of that recent local data as the reception of visual data.

B) Cloud Server Acting as a Receiver

In a developing trend, cloud computing storage is getting a dominant player from backup of store and sharing of data services to the American National Library and e -mail services. In this most of these Services, the cloud is used often the receiver of the data.

C) Hierarchal Approach

Predictive ACK's receiver side based mode is less amount of efficient if changes in the information are scattered. In this scenario, the prediction continuation are frequently interrupted, In this turn, forces the sender to retransmit to the raw data transmission until a new comparison is found at the receiver side and It reported back to the sender Side. To that end, we have to present the Predictive ACK hierarchal mode of operation.

5. CONCLUSION

Cloud computing is expected to trigger high demand for TRE solutions as the amount of data exchanged between the cloud and its users is expected to dramatically increase. The cloud environment redefines the TRE system requirements, making proprietary middle-box solutions inadequate. Consequently,

there is a rising need for a TRE solution that reduces the cloud's operational cost while accounting for application latencies, user mobility, and cloud elasticity.

In this paper, we have presented PACK, a receiver-based, cloud-friendly, end-to-end TRE that is based on novel speculative principles that reduce latency and cloud operational cost. PACK does not require the server to continuously maintain clients' status, thus enabling cloud elasticity and user mobility while preserving long-term redundancy. Moreover, PACK is capable of eliminating redundancy based on content arriving to the client from multiple servers without applying a three-way handshake.

Our evaluation using a wide collection of content types shows that PACK meets the expected design goals and has clear advantages over sender-based TRE, especially when the cloud computation cost and buffering requirements are important. Moreover,
PACK imposes additional effort on the sender only when redundancy is exploited, thus reducing the cloud overall cost. Two interesting future extensions can provide additional benefits to the PACK concept. First, our implementation maintains
chains by keeping for any chunk only the last observed subsequent chunk in an LRU fashion. An interesting extension to this work is the statistical study of chains of chunks that would enable multiple possibilities in both the chunk order and the corresponding predictions. The system may also allow making more than one prediction at a time, and it is enough that one of them will be correct for successful traffic elimination. A second promising direction is the mode of operation optimization of the hybrid sender–receiver approach based on shared decisions derived from receiver's power or server's cost changes.

REFERENCES

[1] E. Zohar, I. Cidon, and O. Mokryn, "The power of prediction: Cloud bandwidth and cost reduction," in Proc. SIGCOMM, 2011, pp. 86–97.
[2] 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, 2010.
[3] U. Manber, "Finding similar files in a large file system," in Proc. USENIX Winter Tech. Conf., 1994, pp. 1–10.
[4] N. T. Spring and D. Wetherall, "A protocol-independent technique for eliminating redundant network traffic," in Proc. SIGCOMM, 2000, vol.30, pp. 87–95.
[5] A. Muthitacharoen, B. Chen, and D. Mazières, "A low-bandwidth network file system," in Proc. SOSP, 2001, pp. 174-187.
[6] E. Lev-Ran, I. Cidon, and I. Z. Ben-Shaul, "Method and apparatus for reducing network traffic over low bandwidth links," US Patent 7636767, Nov. 2009.
[7] S.Mccanne andM. Demmer, "Content-based segmentation scheme for data compression in storage and transmission including hierarchical segment representation," US Patent 6828925, Dec. 2004.
[8] R. Williams, "Method for partitioning a block of data into subblocks and for storing and communicating such subblocks," US Patent 5990810, Nov. 1999.
[9] Juniper Networks, Sunnyvale, CA, USA, "Application acceleration," 1996 [Online]. Available: http://www.juniper.net/us/en/products-services/application-acceleration/
[10] Blue Coat Systems, Sunnyvale, CA, USA, "MACH5," 1996[Online].Available:
http://www.bluecoat.com/products/mach5
[11] Expand Networks, Riverbed Technology, San Francisco, CA, USA, "Application acceleration and WAN optimization," 1998 [Online].
Available:http://www.expand.com/technology/application-acceleration.aspx
[12] F5, Seattle,WA, USA, "WAN optimization," 1996 [Online]. Available:http://www.f5.com/solutions/acceleration/wan-optimization/
[13] A. Flint, "The next workplace revolution," Nov. 2012 [Online]. Available: http://m.theatlanticcities.com/jobs-and-economy/2012/11/nextworkplace-revolution/3904/
[14] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee, "Redundancy in network traffic: Findings and implications," in Proc. SIGMETRICS, 2009, pp. 37–48.
[15] B. Aggarwal, A. Akella, A. Anand, A. Balachandran, P. Chitnis, C. Muthukrishnan, R. Ramjee, and G. Varghese, "EndRE: An end-system redundancy elimination service for enterprises," in Proc. NSDI, 2010, pp. 28–28.
[16] "PACK source code," 2011 [Online]. Available: http://www.eyalzo.com/projects/pack
[17] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker, "Packet caches on routers: The implications of universal redundant traffic elimination," in Proc. SIGCOMM, 2008, pp. 219–230.
[18] A. Anand, V. Sekar, and A. Akella, "SmartRE: An architecture for coordinated network-wide redundancy elimination," in Proc. SIGCOMM, 2009, vol. 39, pp. 87–98.
[19] A. Gupta, A. Akella, S. Seshan, S. Shenker, and J. Wang, "Understanding and exploiting network traffic redundancy," UW-Madison, Madison, WI, USA, Tech. Rep. 1592, Apr. 2007.
[20] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Watch global, cache local: YouTube network traffic at a campus network Measurements and implications," in Proc. MMCN, 2008, pp. 1–13.
[21] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: Local algorithms for document fingerprinting," in Proc. SIGMOD, 2003, pp. 76–85.
[22] S. Ihm, K. Park, and V. Pai, "Wide-area network acceleration for the developing world," in Proc. USENIX ATC, 2010, pp. 18–23

[23] H. Stevens and C. Pettey, “Gartner says cloud computing will be as influential as e-business,” Gartner Newsroom, Jun. 26, 2008.
[24] H. Abu-Libdeh, L. Princehouse, and H.Weatherspoon, “RACS: A case for cloud storage diversity,” in Proc. SOCC, 2010, pp. 229–240.
[25] “Dropbox,” 2007 [Online]. Available: http://www.dropbox.com/ [26] E. Allen and C. M. Morris, “Library of Congress and Duracloud launch pilot program using cloud technologies to test perpetual access to digital content,” News Release, Jul. 2009.
[27] D. Hansen, “GMail filesystem over FUSE,” [Online]. Available: http://sr71.net/projects/gmailfs/
[28] J.Srinivasan,W.Wei, X.Ma, andT. Yu, “EMFS:Email-based personal cloud storage,” in Proc. NAS, 2011, pp. 248–257.
[29] “Amazon Elastic Compute Cloud (EC2),” [Online]. Available: http:// aws.amazon.com/ec2/
[30] “netfilter/iptables project: Libnetfilter_queue,” Oct. 2005 [Online]. Available: http://www.netfilter.org/projects/libnetfilter_queue
[31] A. Z. Broder, “Some applications of Rabin’s fingerprinting method,” in Sequences II: Methods in Communications, Security, and Computer Science. New York, NY, USA: Springer-Verlag, 1993, pp. 143–152.
[32] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, “TCP selective acknowledgment options,” RFC 2018, 1996.
[33] A. Medina, M. Allman, and S. Floyd, “Measuring the evolution of transport protocols in the internet,” Comput. Commun. Rev., vol. 35, no. 2, pp. 37–52, 2005.
[34] V. Jacobson, R. Braden, and D. Borman, “TCP extensions for high performance,” RFC 1323, 1992.