IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

A scalable and consistent event matching service for content-based publish subscribe systems in cloud environment

Dr.V.Goutham[1], B.Triveni [2], B.Navya[3]

CSE, TKREC, Telangana, India [1]

Abstract: The cloud computing affords excessive prospects for the necessities of multifarious computing and consistent communication. Categorized by the increasing arrival rate of live content, the emergency applications that stance a great challenge how to publicize large-scale live content to interested users in a scalable and reliable manner. The publish/subscribe (pub/sub) model is broadly used for data dissemination of its capacity of impeccably escalating the system to colossal size. Most event matching services of existing pub/sub systems either lead to low matching throughput when matching a large number of twisted subscriptions when a large number of servers fail. SREM, a scalable and reliable event matching service for content-based pub/sub systems in cloud computing environment to realize low routing latency and consistent links between servers, a distributed overlay Skip Cloud to organize servers of SREM. Through a Partition technique, large-scale skewed subscriptions are charted into several subspaces, which safeguards high matching quantity and delivers multiple candidate servers for each event. Assessing the concert of SREM, 64 servers are organized and millions of live content items are tested in a Cloud Stack test bed. The tentative results shown that the traffic above of routing events in Skip Cloud is at least 60 percent smaller than in Chord overlay, the matching rate in SREM is at least 3.7 times and at most 40.4 times larger than the single-dimensional partitioning technique of Blue Dove based on various parameter settings. SREM facilitates the event loss rate to drop back to 0 in tens of seconds even if a large number of servers fail concurrently.

Keywords: Publish/subscribe, event matching, overlay construction, content space partitioning, cloud computing.

I. INTRODUCTION

Prominence given in serving users to make real-time decisions, data dissemination has become intensely significant in many large-scale emergency applications, such as earthquake monitoring, disaster weather warning, and status update in social networks. Recently, data dissemination in these emergency applications presents a number of fresh trends. One is the rapid growth of live content. For example, Face book users publish over 600,000 pieces of content and Twitter users send over 100,000 tweets on average per minute [1]. The other is the highly dynamic network environment. The measurement studies indicates that most users' sessions in social networks only last several minutes [2]. In emergency scenarios, the sudden disasters like earthquake or bad weather may lead to the failure of a large number of users immediately. These features necessitate the data dissemination system to be scalable and reliable. Firstly, the system must be scalable to support the large amount of live content. The key is to offer a scalable event matching service to filter out insignificant users. Otherwise, the content may have to traverse a large number of impassive users before they reach interested users. Secondly, with the dynamic network environment, it's reasonably needed to provide reliable schemes to keep continuous data dissemination capacity. Or else, the system interruption may cause the live content becomes obsolete content. Driven by these requirements, publish/subscribe (pub/ sub) pattern is widely used to disseminate data due to its flexibility, scalability, and efficient support of complex event processing. In pub/sub systems (pub/subs), a receiver (subscriber) registers its interest in the form of a subscription. Events are published by senders to the pub/sub system.

The system matches events against subscriptions and disseminates them to interested subscribers. Recently, cloud computing provides great opportunities for the applications of complex computing and high speed communication [12], [13], where the servers are connected by high speed networks, and have powerful computing and storage capacities. A number of pub/sub services based on the cloud computing environment have been proposed, such as Move [14], Blue Dove [15] and SEMAS [16]. However, most of them can not completely meet the requirements of both scalability and reliability when matching large-scale live content under highly dynamic environments. This mainly stems from the following facts: 1) Most of them are inappropriate to the matching of live content with high data dimensionality due to the limitation of their subscription space partitioning techniques

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

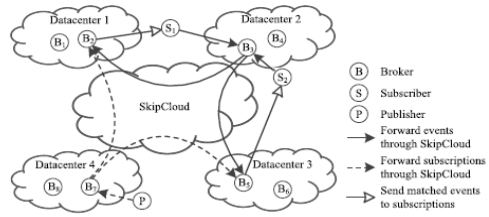


Fig. 1. System framework.

which bring either low matching throughput or high memory overhead. 2) These systems adopt the one-hop lookup technique [17] among servers to reduce routing latency. Let us have a glance on the following contributions: A distributed overlay protocol, called Skip Cloud, to organize servers in the cloud computing environment. Skip Cloud enables subscriptions and events to be forwarded among brokers in a scalable and reliable manner. Also it is easy to implement and maintain. To achieve scalable and reliable event matching among multiple servers, a hybrid multidimensional space partitioning technique, called HP artition came in to existence. It permits similar subscriptions to be separated into the same server and provides multiple candidate matching servers for every event. Additionally, it adaptively assuages hot spots and keeps workload balance among all servers. Extensive experiments based on a Cloud Stack test bed to verify the performance of SREM under various parameter settings.

II. RELATED WORK

A large body of efforts on broker based pub/subs have been proposed in recent years. One method is to organize brokers into a tree overlay, such that events can be delivered to all relevant brokers without duplicate transmissions. Besides, data replication schemes [2] are employed to ensure reliable event matching. For instance, Siena [3] advertises subscriptions to the whole network. When receiving an event, each broker determines to forward the event to the corresponding broker according to its routing table. In Atmosphere [4], it dynamically identifies entourages of publishers and subscribers to transmit events with low latency. It is appropriate to the scenarios with small-scale of subscribers. As the number of subscribers increases, the over-overlays constructed in Atmosphere probably have the similar latency like in Siena. To ensure reliable routing, Kazemzadeh and Jacobsen [5] propose a d-fault-tolerance algorithm to handle concurrent crash failure of up to d brokers. Brokers are required to maintain a partial view of this tree that includes all brokers within distance dþ1.The multi-hop routing techniques in these tree-based pub/subs lead to a high routing latency. Besides, skewed subscriptions and events lead to unbalanced workloads among brokers, which may severely reduces the matching throughput. In contrast, SREM uses Skip Cloud to reduce the routing latency and HP artition to balance the workloads of brokers. Another method is to divide brokers into multiple clusters through unstructured overlays. Brokers in each cluster are connected through reliable topologies. For instance, brokers in Kyra [7] are grouped into cliques based on their network proximity. Each clique divides the whole content space into non-overlapping zones based on the number of its brokers. After that, the brokers in different cliques which are responsible for similar zones are connected by a multicast tree. Thus, events are forwarded through its corresponding multiple tree. Sub-2-Sub [8] implements epidemic-based clustering to partition all subscriptions into disjoint subspaces. The nodes in each subspace are organized into a bidirectional ring. Due to the long delay of routing events in unstructured overlays, most of these approaches are inadequate to achieve scalable event matching. In contrast, SREM uses Skip Cloud to organize brokers, which uses the prefix routing technique to achieve low routing latency. To reduce the routing hops, a number of methods organize brokers through structured overlays which commonly Need QðlogNÞ hops to locate a broker. Subscriptions and events falling into the same subspace are sent and matched on a rendezvous broker. For instance, Pastry String [9] constructs a distributed index tree for each dimension to support both numerical and string dimensions. The resource discovery service proposed by Ranjan et al. [10] maps events and subscriptions into d-dimensional indexes, and hashes these indexes onto a DHT network. To ensure reliable pub/sub service, each broker in Meghdoot [11] has a back up which is used when the primary broker fails. Compared with these DHT-based approaches, SREM ensures smaller forwarding latency through the prefix routing of Skip Cloud, and higher event matching reliability by multiple brokers in each top cluster of Skip Cloud and multiple candidate groups of HP artition. Recently, a number of cloud providers have offered a series of pub/sub services. For instance, Move [14] provides high available key-value storage and matching respectively based on one-hop lookup [17]. Blue Dove [15] adopts a single-dimensional partitioning technique to divide the entire spare and a performance-aware forwarding scheme to select candidate matcher for each event. Its scalability is limited by the coarse-grained clustering technique. SEMAS [16] proposes a fine-grained partitioning technique to achieve high matching rate. However, this partitioning technique only provides one candidate for each event and may lead to large memory cost as the number of data dimensions increases. In contrast, HP artition makes a better trade-off between the matching throughput and reliability through a flexible manner of constructing logical space.

III. SYSTEM DESIGN

3.1 SREM Design

To support large-scale users, we consider a cloud computing environment with a set of geographically distributed data centers through the Internet. Each datacenter contains a large number of servers (brokers), which are managed by a datacenter management service such as Amazon EC2 or OpenStack. All brokers in SREM as the front-end are exposed to the Internet, and any subscriber and publisher can associate to them unswervingly. To accomplish reliable connectivity and

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016

low routing latency, these brokers are connected through an distributed overlay, called Skip Cloud. The entire content space is partitioned into disjoint subspaces, each of which is managed by a number of brokers. Subscriptions and events are dispatched to the subspaces that are overlapping with them through Skip Cloud. Subscriptions and events falling into the same subspace are matched on the same broker. After the matching process completes, events are broadcasted to the corresponding interested subscribers. The subscriptions generated by subscribers S1 and S2 are dispatched to broker B2 and B5,respectively. Upon receiving events from publishers, B2 and B5 will send matched events to S1 and S2, respectively.

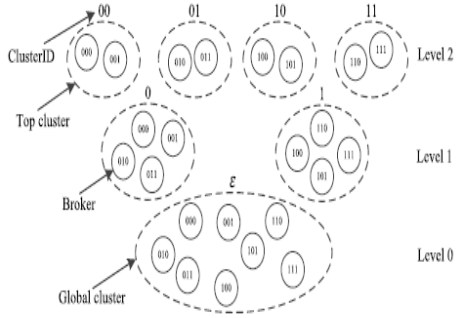


Fig. 1: An example of Skip Cloud with eight brokers and three levels.

Every broker is acknowledged by a binary string. All datacenters due to the various skewed distributions of users' interests. The node failure may lead to unreliable and inefficient routing among servers. To this end, it is organized servers into Skip Cloud to reduce the routing latency in a scalable and reliable manner. Such a framework offers a number of advantages for real-time and reliable data dissemination. First, it allows the system to timely group similar subscriptions into the same broker due to the high bandwidth among brokers in the cloud computing environment, such that the local searching Time can be greatly reduced. Second, since each subspace is managed by multiple brokers, this framework is fault-tolerant even if a large number of brokers crash straightaway. Third, because the data center management service provides scalable and elastic servers, the system can be easily expanded to Internet-scale.

3.2 SKIPCLOUD: Topology Construction

SkipCloud systematizes all brokers into levels of clusters. The clusters at each level of SkipCloud can be treated as a partition of the whole broker set. At the top level, brokers are systematized into numerous clusters whose topologies are complete graphs. Each cluster at this level is called top cluster. It contains a leader broker which produces a unique b-ary identifier with length of m using a hash function (e.g. MD-5). This identifier is called ClusterID. Individually, each broker's identifier is a unique string and shares common prefix of length m with its ClusterID. At this level, brokers in the same cluster are accountable for the same content subspaces, which provides numerous identical candidates for each event. Since brokers in the same top cluster generate frequent communication among themselves, such as updating subscriptions and dispatching events, they are organized into a complete graph to reach each other in one hop. After the top clusters have been well organized, the clusters at the rest levels can be generated level by level. Precisely, each broker decides to join which cluster at every level.

TABLE 1
Notations in SkipCloud

N_b	the number of brokers in SkipCloud
m	the number of levels in SkipCloud
D_c	the average degree in each cluster of SkipCloud
N_c	the number of top clusters in SkipCloud

IV. PROPOSED SYSTEM: INNOVATIVE STRATEGIES

4.1 HPARTITION:

In order to take benefit of multiple distributed brokers, SREM distributes the entire content space among the top clusters of Skip Cloud, so that each top cluster only switches a subset of the entire space and searches a small number of candidate subscriptions. SREM employs a hybrid multidimensional space partitioning technique, called HP artition, to realize scalable and reliable event matching. Generally speaking, HP artition divides the entire content space into disjoint subspaces. Subscriptions and events with overlapping subspaces are dispatched and matched on the same top cluster of Skip Cloud .To keep workload balance among servers, HP artition divides the hot spots into various cold spots in an adaptive manner.

Algorithm 1. Neighbor List Maintenance

Input: *views*: the neighbor lists.
m: the total number of levels in SkipCloud.
cycle: current cycle.

```
1: j = cycle%(m + 1);
2: for each i in [0,m-1] do
3:     update views[i] by the peer sampling
       service based on Cyclon.
4: for each i in [0,m-1] do
5:     if views[i] contains empty slots then
6:         fill these empty slots with other levels'
           items who share common prefix of length i
           with the ClusterID of views[i].
```

Algorithm 2. Prefix Routing

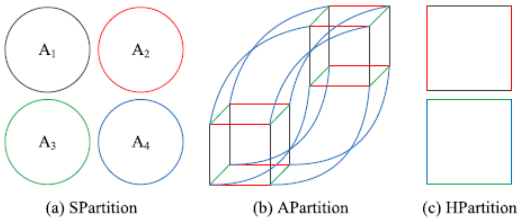
```
1: l = commonPrefixLength(self.ID, event.ClusterID);
2: if (l == m) then
3:     process(event);
4: else
5:     destB ← the broker whose identifier matches
       event.ClusterID with the longest common prefix
       from self.views.
6:     l_max = commonPrefixLength(destB.identifier,
       event.ClusterID);
7:     if (l_max ≤ l) then
8:         destB ← the broker whose identifier is
           closest to event.ClusterID from view[l].
9:         if (destB and myself is in the same cluster of
           level l) then
10:            process(event);
11:    forwardTo(destB, event);
```


DOI 10.17148/IJARCCE.2016.56170

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 5, Issue 6, June 2016



(a) SPartition (b) APartition (c) HPartition

4.2 Adaptive Selection Algorithm

Because of diverse distributions of subscriptions, both HSPartition and SSPartition cannot substitute with each other. HSPartition is striking to divide the hot spots whose subscriptions are uniform dispersed regions. However, it's unsuitable to rift the hot spots whose subscriptions all appear at the same exact point. On the other hand, SSPartition allows to divide any kind of hot spots into multiple subsets even if all subscriptions falls into the same single point. Nevertheless, compared with HSPartition, it has to dispatch an event to multiple subspaces, which brings a higher traffic overhead. To accomplish balanced workloads among brokers, An adaptive selection algorithm to select either HSPartition or SSPartition to assuage hot spots. The selection is based on the similarity of subscriptions in the same hot spot. Specifically, subspace with maximal size of subscriptions in HSPartition. We choose HSPartition as the partitioning algorithm through combining both partitioning techniques, this selection algorithm can alleviate hot spots in an adaptive manner.

V. IMPLEMENTATION

To take advantage of the reliable links and high bandwidth among servers of the cloud computing environment, we choose the CloudStack [24] testbed to design and implement our prototype. To develop the prototype as modular and portable, used ICE [25] as the fundamental communication platform. ICE offers a communication resolution that is laidback to program with, and allows the developers to only focus on their application logic. To evaluate the performance of SkipCloud, it is implemented both SkipCloud and Chord to forward subscriptions and messages. To evaluate the performance of HPartition,
the prototype supports different space partitioning policies. Furthermore, the prototype provides three different message forwarding strategies, i.e, least subscription amount forwarding, random forwarding, and probability based forwarding.

5.1 Parameters and Metrics

A group of virtual machines (VMs) in the CloudStack test bed to evaluate the performance of SREM. Each VM is running in a exclusive physical machine, and we use 64 VMs as brokers. For each VM, it is equipped with four processor cores, 8 GB memory, and is connected to Gigabit Ethernet switches. In Skip Cloud, the number of brokers in each top cluster d is set to 2. To ensure reliable connectivity of clusters, the average degree of brokers in each cluster Dc is 6. In HP artition, the entire subscription space consists of eight dimensions, each of which has a range from 0 to 500. The range of each dimension is cut into 10 segments by default. For each hotspot, the range of its every dimension is cut into two segments iteratively. The performance of SREM through a number of metrics is evaluated in following perspectives. Subscription searching size: The number of subscriptions that need to be searched on each broker for matching a message. Matching rate: The number of matched events per second. Suppose the first event is matched at the moment of T1, and the last one is matched at the moment of T2. Event loss rate: The percentage of lost events in a specified time period.

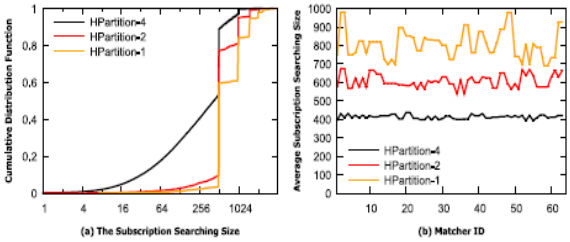


Fig. 2: Distribution of subscription searching sizes

5.3 Forwarding Policy and Workload Balance

5.3.1 Impact of Message Forwarding Policy

The forwarding policy determines each message to be forwarded to which broker, which significantly affects the workload balance among brokers. These policies mainly lies in its better trade-off between workload balance and searching latency. For instance, when we use HPartition- 4 under SkipCloud, the matching rate of the probability based policy are of random policy and least subscription amount forwarding policy, respectively. When we use HPartition-4 under Chord, the corresponding gains of the probability based policy .

5.3.2 Workload Balance of Brokers

We compare the workload balance of brokers under different partitioning strategies and overlays. In the experiment, we use the probability based forwarding policy to dispatch events. One million events are forwarded and matched against 40000 subscriptions in the experiments.

VI. CONCLUSIONS AND FUTURE WORK

SREM, a scalable and reliable event matching service for content-based pub/sub systems in cloud computing environment. SREM attaches the brokers over and done with a scattered overlay Skip Cloud, which certifies reliable connectivity among brokers through its multi-level clusters and brings a low routing latency through a prefix routing algorithm. A hybrid multi-dimensional space partitioning technique, helps out SREM in reaching scalable and balanced clustering of high dimensional twisted subscriptions, and each event is permitted to be matched on any of its candidate servers. Extensive experiments with real deployment based on a Cloud Stack testbed are accompanied, producing results which demonstrate that SREM is effective and practical, and also

presents good workload balance, scalability and reliability under various parameter settings. Although proposed event matching service can competently filter out extraneous users from big data volume, there are still a number of problems need to be solved. Based on this event matching service, it is considered utilizing a cloud-assisted technique to realize a general and scalable data dissemination service over live content with several data sizes.

REFERENCES

[1] Dataperminite. (2014). [Online]. Available: http://www.domo.com/blog/2012/06/how-much-data-is-created-every-minute/
[2] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida, "Characterizing user behavior in online social networks," in Proc. 9th ACM SIGCOMM Conf. Internet Meas. Conf., 2009, pp. 49–62.
[3] A. Carzaniga, "Architectures for an event notification service scalable to wide-area networks," Ph.D. dissertation, Ingegneria Informatica e Automatica, Politecnico di Milano, Milan, Italy, 1998.
[4] P. Eugster and J. Stephen, "Universal cross-cloud communication," IEEE Trans. Cloud Comput., vol. 2, no. 2, pp. 103–116, 2014.
[5] R. S. Kazemzadeh and H.-A Jacobsen, "Reliable and highly available distributed publish/subscribe service," in Proc. 28th IEEE Int. Symp. Reliable Distrib. Syst., 2009, pp. 41–50.
[6] Y. Zhao and J. Wu, "Building a reliable and high-performance content-based publish/subscribe system," J. Parallel Distrib. Comput., vol. 73, no. 4, pp. 371–382, 2013.
[7] F. Cao and J. P. Singh, "Efficient event routing in content-based publish/subscribe service network," in Proc. IEEE INFOCOM, 2004, pp. 929–940.
[8] S. Voulgaris, E. Riviere, A. Kermarrec, and M. Van Steen, "Sub-2-sub: Self-organizing content-based publish and subscribe for dynamic and large scale collaborative networks," Res. Rep. RR5772, INRIA, Rennes, France, 2005.
[9] I. Aekaterinidis and P. Triantafillou, "Pastrystrings: A comprehensive content-based publish/subscribe DHT network," in Proc. IEEE Int. Conf. Distrib. Comput. Syst., 2006, pp. 23–32.
[10] R. Ranjan, L. Chan, A. Harwood, S. Karunasekera, and R. Buyya, "Decentralised resource discovery service for large scale federated grids," in Proc. IEEE Int. Conf. e-Sci. Grid Comput., 2007, pp. 379–387.
[11] A. Gupta, O. D. Sahin, D. Agrawal, and A. El Abbadi, "Meghdoot: Content-based publish/subscribe over p2p networks," in Proc. 5th ACM/IFIP/USENIX Int. Conf. Middleware, 2004, pp. 254–273.
[12] X. Lu, H. Wang, J. Wang, J. Xu, and D. Li, "Internet-based virtual computing environment: Beyond the data center as a computer," Future Gener. Comput. Syst., vol. 29, pp. 309–322, 2011.
[13] S. Voulgaris, D. Gavidia, and M. van Steen, "Cyclon: Inexpensive membership management for unstructured p2p overlays," J. Netw. Syst. Manage., vol. 13, no. 2, pp. 197–217, 2005.
[14] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: A resilient global-scale overlay for service deployment," IEEE J. Sel. Areas Commun., vol. 22, no. 1, pp. 41–53, Jan. 2004.
[15] Murmurhash. (2014). [Online]. Available: http://burtleburtle.net/bob/hash/doobs.html
[16] A.-M Kermarrec, L. Massouli_e, and A. J. Ganesh, "Probabilisticreliable dissemination in large-scale systems," IEEE Trans. Parallel Distrib. Syst., vol. 14, no. 3, pp. 248–258, Mar. 2003.
[17] Y. Wang and S. Li, "Research and performance evaluation of data replication technology in distributed storage systems," Comput. Math. Appl., vol. 51, no. 11, pp. 1625–1632, 2006.

BIOGRAPHIES

Dr V. Goutham is a Professor and Head of the Department of computer Science and Engineering at TKR Engineering College affiliated to J.N.T.U Hyderabad. He received M.Tech from Andhra University and B.Tech from J.N.T.U Hyderabad. He worked for various MNC Companies in Software Testing and Quality as Senior Test Engineer. His research interests are Software Reliability Engineering, software testing, software Metrics, and cloud computing.

Mrs. B.Triveni is working as a Assistant Professor in the Department of computer Science and Engineering at TKR Engineering College affiliated to J.N.T.U Hyderabad.

Ms. B.Navya Department of computer Science and Engineering at TKR Engineering College affiliated to J.N.T.U Hyderabad.