International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 3, Issue 2, February 2016

Guaranteed Event Delivery For Publish/Subscribe Services

[1]Lajitha H, [2] Ratheesh S
[1] PG Scholar, [2] Assistant Professor
[1][2] Department of Computer Science, College of Engineering Perumon
[1] lajithahs@gmail.com [2] ratheeshs2014@gmail.com

Abstract- **The publish/subscribe(pub/sub) is a widespread communication paradigm, which has been paid more attention in distributed applications such as online gaming, social networking , business intelligence because it offers the time, space and synchronization decoupling properties. These applications are typically characterized by a large number of participants which are communicate by exchanging messages on a wide area network (WAN), such as the Internet. However, while this kind of paradigm fits the generic asynchrony and scalability requirements of large scale complex critical infrastructures (LCCIs), it completely or partially lacks the support of quality-of-service (QoS) guarantees. The maximum flow path is needed to achieve maximum data rate/ packet delivery for multicasting to several sinks. The proposed system uses a techniques network coding in a maximum flow network. And also uses a mechanism to choose gossip partners during the recovery phase of the protocol.**

***Index Terms*— Publish, subscribe, gossiping, coding.**

I. INTRODUCTION

In the last few years, there is a need for developing new technologies because of the huge increase in the rate of internet users. There are large numbers of participants scattered across the world that communicate by exchanging messages using internet. The features of the internet change in accordance with the degree of connectivity of the users with it, who are located at different networks such as LANs, WANs and large scale Complex Critical Infrastructures (LCCIs).The communication in a WAN may be affected by the unpredictable behavior of the network, with messages that can be dropped or delayed. Also the LCCIs are the systems which are involved in exchange of a large amount of messages, which required satisfying the dynamic decoupling properties. This demand motivates the designing of a new flexible loosely coupled transmission infrastructures. The Publish / Subscribe paradigm are an effective solution for distributed applications, which act as a messaging middleware for such loosely coupled transmissions that meet the decoupling properties, scalability and asynchrony requirements of those applications.

Earlier systems which implements the publish/subscribe paradigm had certain challenges includes either it provide only a best-effort service or address just a single requirement at a time such as message ordering , delivery of message in bounded time and reliability of transmission. Some applications like messaging, social networking can rely on a best-effort service, while other applications may require several nonfunctional requirements to be met. The messages must be delivered within time and complete if not received in time they are useless. So this paper aims to design a mechanism that provides the timeliness and reliability to the publish/subscribe services over the internet.

II. BACKGROUND

A. Publish/Subscribe paradigm

A publish/subscribe communication system[2] realizes an anonymous interaction among its participants. The node which is creating and publishing an event is called Publisher; other nodes which are receiving that event notification are called Subscribers. Each participant in a pub/sub system can take on the role of the publisher or a subscriber of information depends on the state of the node either receiving or sending the event at that instant. Fig. 1 describes a simple publish/subscribe system with its operations. A publisher dispatches the information by executing the publish( ) operation on the Event Notification Service. The Notification Service dispatches the information submitted by other processes to a subscriber by executing the notify( ) operation on it. Receivers are not directly targeted from publisher but indirectly addressed according to the content of notifications. Subscribers register their interest in events by calling a subscribe() operation on the event notification service, without knowing the effective sources of these events. This subscription information remains stored in the event service and is not forwarded to publishers. The unsubscribe () operation is used for cancel their registration.

Publish / subscribe system with linear network coding With linear network coding, outgoing packets are linear combinations of the original packets, where addition and multiplication are performed over the field.Each packet consists of L bits and "s" consecutive bits of a packet can be considered as a symbol over the field F_2^s [3].
The original packets produced by several publishers are M^1, M^2...,M^n. The coefficients which are used to produce the linear combination are obtained by any of the following way:

- Each node in the network select uniformly at random the coefficients over the field F_2^s [8].
- Certain probability of selecting linearly dependent combinations-probability is related to the field size 2^s.[10]
- Polynomial-time algorithm[9] for multicasting which examines each node of the network, and decides what linear combinations each node performs. So that each node uses fixed linear coefficients and the packets only need to carry the information vecto.

Each linear combination is given by:

$$X = \sum_{i=1}^{n} g_i M^i$$

The summation have been carried out at each symbol position k,

$$X_k = \sum_{i=1}^{n} g_i M_k^i$$

The coefficient (g_1, ..., g_n) are called encoding vector and the encoded data X is called information vector.

For solving a linear system each node stores the encoded vectors with its own original packets, row by row, in a decoding matrix. Initially, the matrix only contains the non-encoded packets issued by this node with the corresponding encoding vectors. When an encoded packet is received, it is inserted as last row into the decoding matrix. The matrix is transformed to triangular matrix using Gaussian elimination. A received packet is added to the matrix if it increases the rank of the matrix else discarded.

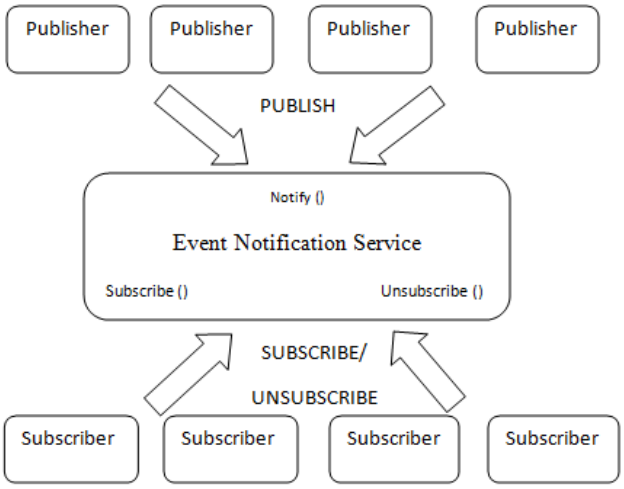

Fig.1: A simple Publish/Subscribe System Publish / Subscribe with Gossiping Gossiping is an epidemic approach in which the event/message transmitted like a contiguous disease spread or diffusion of perfume in the air. It is a probabilistic and fully distributed approach to event routing and achieves high stability under high network dynamics, and scales gracefully to a huge number of nodes[5] .It is performed at the time of recovering any lost packets is found at receiver/subscriber side. In gossiping, each node contacts one or a few nodes in each round, and exchanges information with these nodes. Whenever the node receives a message it stores that in a buffer of size „b", called as fan_in, if the node forwards a message limited number of times "t" to other nodes in the overlay network is called as fan_out.

There are three different types of gossiping algorithms are there at present:

- ***Push:*** immediately on receiving a message the node forwards that to other node.
- ***Pull:*** periodically the identifiers of set of messages received will be sent to other nodes. On recognizing that a message is missed by comparing the set of Ids with local history it makes an explicit pull request for the missed packet.
- ***Push/Pull:*** the node on successfully receiving message diffuses message Id to others. The other nodes which were not received it will make explicit pull request.

B. Ford–Fulkerson algorithm

The Ford–Fulkerson method is an algorithm which computes the maximum flow in a flow network. The idea behind the algorithm is that as long as there is a path from the source (start node) to the sink (end node), with available capacity on all edges in the path, send flow along one of

these paths. Then find another path, and so on. A path with available capacity is called an augmenting path.

Let G=(V,E) be a graph, and for each edge from u to v , let c(u,v) be the capacity and f(u,v) be the flow. The aim is to find the maximum flow from the source to the sink .After every step in the algorithm the following is maintained:

- Flow on an edge doesn't exceed the given capacity of the edge.
- Incoming flow is equal to outgoing flow for every vertex except s and t.

Residual Graph of a flow network is a graph which indicates additional possible flow. If there is a path from source to sink in residual graph, then it is possible to add flow. Every edge of a residual graph has a value called residual capacity which is equal to original capacity of the edge minus current flow. Residual capacity is basically the current capacity of the edge. Residual capacity is 0 if there is no edge between to vertices of residual graph.

The Ford-Fulkerson Algorithm

1. for every edge e, f(e) = 0
2. G_f is residual graph of G with respect to f
3. while G_f has a simple s-t path
   let P be simple s-t path in G_f
   f = augment(f,P)
   Construct new residual graph G_f.

augment(f,P)

let b be bottleneck capacity, i.e., min capacity of edges in P

1. for each edge (u,v) in P
   if e=(u,v) is a forward edge f(e) = f(e) + b
   else (* (u,v) is a backward edge *)
   e = (v,u) (* (v,u) is in G *)
   f(e) = f(e) – b
   return f

Initialize the residual graph as original graph as there is no initial flow and initially residual capacity is equal to original capacity. To find an augmenting path, either do a BFS or DFS of the residual graph. And find out if there is a path from source to sink. To find path which have maximum flow ,create an array. By traversing through this found path and find possible flow through this path by finding minimum residual capacity along the path. Add the found path to the overall flow. To update residual capacities in the residual graph subtract path flow from all edges along the path and add path flow along the reverse edges. There is a need to add path flow along reverse edges because may later need to send flow in reverse direction.

III. RELATED WORK

In this section we study about some of the projects which make use of the publish/subscribe mechanism as major constraint. Reliability for publish/subscribe system can be ensured in JEDI (Java Event-based Distributed Infrastructure by use of TCP connections. While the use of TCP in multicast tree overlays exhibits a low throughput in practical applications. IndiQoS is an architecture for a distributed publish-subscribe message broker that uses a structured overlay network to route events in the broker and match subscriptions with advertisements. It focuses on respecting timeliness in event dissemination. Subscribers specify the latency constraint as an attribute of their subscriptions. Several QoS properties (including message ordering, timeliness, and reliability) are ensured by some publish/subscribe commercial systems, such as Tibco, DDS , and JMS. However, these systems are designed to properly work in small/medium LANs, while their performance poorly degrades over WANs.

IV. PROPOSED SYSTEM

The aim of this paper is needed to enforces both reliability and timeliness for publish/subscribe services over WAN. We create two independent blocks for achieving this aim : one implements a network coding protocol that aims to reduce the delivery time of an event along in a maximum flow network and one implements a gossip-based algorithm to recover from possible event losses[1].
The proposed system designed to reliably and timely deliver events is consist of mainly two phases:

Dissemination: A publisher divides an event into plain packets and find it's linear combination. The original packets along with the linear combination is passed to the ENS .Then ENS find the paths from publishers to the registered subscribers which have the maximum flow and publish the event.
Recovery: The subscribers use a gossip protocol to gather possible lost events. This phase depends on the gossip style in use: If the push or push/pull strategy is enabled, then a node that notifies an event (i.e., it has received enough packets to reconstruct the whole event) disseminates to the other nodes the received packets or the event identifier, respectively. When the pull style is enabled, periodically a node disseminates to other nodes the identifiers of the last notified events.

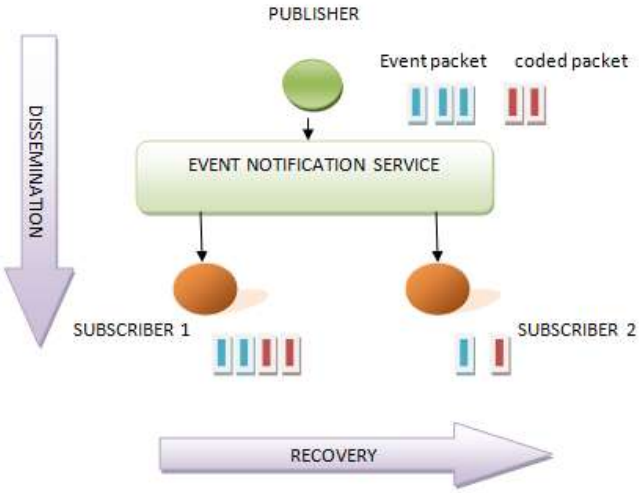

Fig.3 Dissemination and Recovery phase

Dissemination: A publisher p publishes an event and redundant packets on the ENS. Recovery: Based on the specific gossip strategy in use, a subscriber can recover missing packets. Subscriber1 have a sufficient number of packets to reconstruct the original event, while Subscriber 2 needs a gossip round to gather the missing information.

The system maintains four matrices: capacity matrix (used store the capacity of edges path), flow matrix (used to store the flow of edges), path matrix (used to store the edges corresponding to the maximum flow path) and decoding matrix is maintained in the triangular form. Each time a new encoded packet arrives, it is inserted into the matrix only if the new carried coefficients increase the rank of that matrix. Both path matrix and flow matrix are updated dynamically by performing the Ford-Fulkerson algorithm.

In PUBLISH() operation ,the event is fragmented in plain packets . And the coded packets are generated from the linear combinations of the original packets. The original packets along with the linear combination is passed to the ENS .Then ENS find the paths from publishers to the subscribers which have the maximum flow by using Ford-Fulkerson algorithm[Section III].The updated path matrix contains the maximum flow path/min-cut separating source and sink node. ENS send the event along this path.

At the subscriber, decode the packets if they are linearly combined .If the number of receiving a number of packets equal to or higher than is a necessary but not sufficient condition to fully reconstruct an event. Otherwise, use any of the gossip strategies.

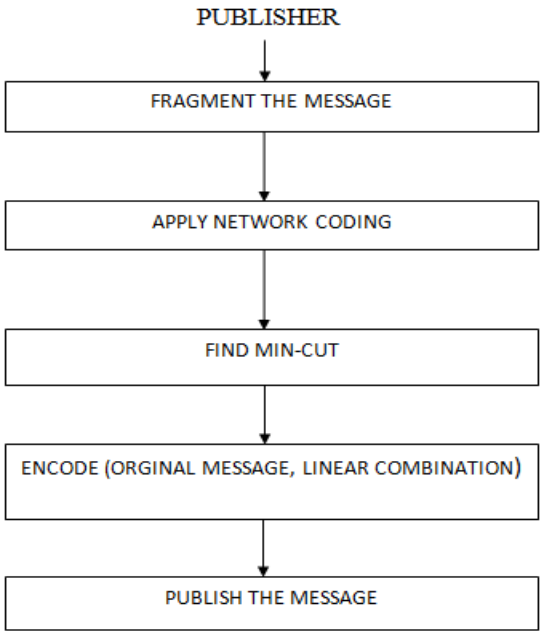

Fig.3 Publish() operation

If a push or pushpull strategy is used, then the subscriber find a set of random nodes currently in the system (these are active subscribers that time).If push strategy is used, the active nodes send the event e to requested node. If a pushpull strategy is used, the node forwards only the identifier of the last received message. By comparing it with the local history to find the missing packet.

Pseudocode for operations at the subscriber

1. incomingpackets ← packet U decode(coded packets)
2. if(incomingpackets >= n)
   then e = reconstructEvent(incomingpackets)
   notify(e);
   endif
3. if(gossip-mode = PUSH or PUSHPULL)
   then contacts ← active subscribers;
   endif
4. if(gossip-mode = PUSH)
   then for each(sj ϵ contacts)
   push(e);
   end if.
5. else if(gossip-mode = PUSHPULL)
   then for each (sj ϵ contacts)
   push(id_e);

V. CONCLUSION

In this paper, we have proposed a strategy that combines coding and gossip for reliable and timely event dissemination over the Internet. In this work we uses an min-cut algorithm along with coding for providing the maximum packet delivery. The coding reduced the number of the retransmissions required there by improves performance of system which achieved the timeliness of event notification services. While performing the publish of message through a maximum flow path also increases the timeliness of the system and reduce the cost of performing the broadcasting operation. In the near future, we plan to extend the contribution of this work by proposing a different mechanism like polynomial time coding instead of random linear network coding. Gossiping improves the reliability and increased the success rate of the nodes.

RFERENCES

[1]. Christian Esposito, Marco Platania, and Roberto Beraldi "Reliable and Timely Event Notification for Publish/Subscribe Services Over the Internet" IEEE/ACM Transactions on Networking, Vol 22,No1,February 2014.

[2]. P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec "The many faces of publish/subscribe," ACM Computing Surveys (CSUR), vol. 35, no. 2, pp. 114-131, June 2003.

[3]. C. Fragouli, J. L. Boudec, and J. Widmer "Network coding: an instant primer,"Computer Communication Review, vol. 36, no.1, p. 63, 2006.

[4]. C. Esposito, S. Russo, R. Beraldi, and M. Platania "On the benefit of network coding for timely and reliable event dissemination in WAN," in Proc. 1st Int. Workshop Netw. Resilience, Oct. 2011, pp. 8489.

[5]. Paolo Costa, Matteo Migliavacca, Gian Pietro Picco, and Gianpaolo Cugola "Introducing Reliability in Content Based Publish/Subscribe through Epidemic Algorithms," 20133 Milano, Italy.

[6]. S. Li, R. Yeung, and N. Cai "Linear network coding," IEEE Trans. Inf. Theory, vol. 49, no. 2, pp. 371-381, Feb. 2003.

[7]. https://en.wikipedia.org/wiki/Ford%E2%80%93Fulkerson_algorithm.

[8]. C. Fragouli, J. Widmer, and J.-Y. L. Boudec. "The benefits of coding over routing in a randomized setting", International Symposium on Information Theory (ISIT), page 442, July 2003.

[9]. P. Sanders, S. Egner, and L. Tolhuizen." Polynomial time algorithms for network information flow," Proc. 15th ACM Symposium on Parallel Algorithms and Architectures, 2003.

[10]. Y. Wu, P. A. Chou, and K. Jain." A comparison of network coding and tree packing". ISIT 2004, 2004.