Non-functional Requirements in Publish/Subscribe Systems

Von der Fakultät Informatik, Elektrotechnik und Informationstechnik der Universität Stuttgart zur Erlangung der Würde eines Doktors der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Muhammad Adnan Tariq

aus Faisalabad, Pakistan

Hauptberichter:	Prof. Dr. rer. nat. Dr. h. c. Kurt Rothermel
Mitberichter:	Prof. Dr. -Ing. Ralf Steinmetz
Tag der mündlichen Prüfung:	28. June 2013

Institut für Parallele und Verteilte Systeme (IPVS)
der Universität Stuttgart
2013

Acknowledgments

First and foremost, thanks to Almighty Allah S.W.T. for endowing me the strength, the courage and the perseverance for achieving this milestone in life.

I would like to thank my advisor and mentor Prof. Dr. Kurt Rothermel for giving me the opportunity to work in his research group. His critical questions and guidance played an essential role in the success of work done in this thesis. Special thanks to my project supervisor Dr. Boris Koldehofe for his constant support, encouragement and technical assistance during my doctoral studies.

My sincere thanks are also given to Prof. Dr. Ralf Steinmetz for kindly serving on my PhD committee and reviewing this thesis.

I cherish my colleagues in the Distributed Systems Department and within BW-FIT SpoVNet project, for countless interesting discussions on research and friendly working environment. Gerald. G. Koch especially has helped me tremendously by collaborating with me on many research papers. Many other current and former colleagues deserve special mention such as Frank Dürr, Andreas Benzing, Stefan Föll, Faraz Ahmed Memon, Bilal Hameed, Marco Völz, Stamatia Rizou, Björn Schilling, Beate Ottenwälder, Ben Carabelli, Christian Hübsch, Philipp Schaber, Damian Philipp, Harald Weinschrott, Lars Geiger and Imran Ahmed Khan. I am sincerely thankful to the people mentioned above and to the unmentioned ones who helped me in any way.

Last but not the least, I would like to thank the Banden-Wüttemberg Stiftung GmbH for their financial support through the SpoVNet project, which enabled the research work done in this thesis.

Finally, I dedicate this thesis to my mother Afzala Naheed, my father M. Tariq Bashir, my grandfather Bashir Hussain, my daughter Asbah Adnan, my brother M. Salman Tariq, my sisters Wajeeha Rahat and Madiha Awais. The contributions from all these people go much beyond from what is visible in this thesis.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

ALM	Application Layer Multicast
APSP	All Pairs Shortest Path
AS	Autonomous System
BDH	Bilinear Diffie-Hellman
CAN	Content Addressable Network
CLIO	Cross Layer Information Overlay
CORBA	Common Object Request Broker Architecture
CP-ABE	Ciphertext Policy Attribute-based Encryption
CPU	Central Processing Unit
DBDH	Decisional Bilinear Diffie-Hellman
DCOM	Distributed Component Object Model
DHT	Distributed Hash Table
DoS	Denial of Service Attack
GB	Gigabyte
GHz	Gigahertz
IBE	Identity-based Encryption
ICMP	Internet Control Message Protocol
IP	Internet Protocol
ISP	Internet Service Provider
IT	Information Technology
KB	Kilobyte
LAN	Local Area Network
LCC	Linear Capacity Constraints
MBMT	Maximum Bandwidth Multicast Tree
MCPO	Multicast/Multi-peer Overlays
MPLS	Multi-protocol Label Switching
MRCT	Minimum Routing Cost Spanning Tree
ms	Millisecond

MSMT	Maximum Stress Multicast Tree
MST	Minimum Spanning Tree
NAT	Network Address Translation
OCT	Optimum Communication Spanning Tree
OSPF	Open Shortest Path First
P2P	Peer-to-Peer
PBC	Pairing-based Cryptography
PCA	Principal Component Analysis
PKI	Public Key Infrastructure
QoS	Quality of service
RAM	Random Access Memory
RDP	Relative Delay Penalty
RMI	Remote Method Invocation
RPC	Remote Procedure Call
RTT	Round Trip Time
SNMP	Simple Network Management Protocol
SPT	Shortest Path Tree
SSL	Secure Socket Layer
SVD	Singular Value Decomposition
TLS	Transport Layer Security
TTL	Time to Live

Abstract

Content-based publish/subscribe has gained high popularity for large-scale dissemination of dynamic information in the form of *events* from publishers to subscribers in a decoupled fashion. Yet, it is highly challenging to achieve scalability without sacrificing the expressiveness of subscriptions (user queries) in such systems, especially in a peer-to-peer (P2P) environment where subscribers and publishers are also responsible for forwarding events by forming an overlay network. Moreover, the support for non-functional requirements such as quality of service (e.g., end-to-end delay, bandwidth etc.) and security (e.g., authentication, confidentiality etc.) instigate many open research questions. The main advantage of publish/subscribe – its inherent decoupling – turns to be a major obstacle in the fulfilment of non-functional requirements with conventional methods.

Therefore, the goal of this thesis is to develop methods and algorithms that i) enable scalable dissemination of events, ii) support different quality of service aspects, and iii) provide basic security mechanisms, in a P2P content-based publish/subscribe system. In particular, the following contributions are made in this thesis.

As a first contribution, we propose a distributed algorithm to disseminate events in a publish/subscribe system respecting the subscriber-defined QoS constraints in terms of delay requirements and available bandwidth. In addition, the second contribution focuses on minimizing the overall resource usage, i.e., bandwidth consumption and processing load on peers (publishers and subscribers), in a publish/subscribe system. This contribution develops an efficient and scalable method to reduce the rate of events that peers receive and forward though lacking subscription (i.e., *false positives*) by means of subscription clustering, using the techniques from spectral graph theory.

The first two contributions target overlay-level methods to provide efficient dissemination of events and quality of service. The third contribution, however, proposes underlay-aware methods that explicitly take into account the properties of the underlying physical network and its topology, to construct an efficient publish/subscribe routing overlay with low *relative delay penalty* and low *stress* on the physical links.

Finally, as our last contribution, we present novel methods to provide authentication of publishers and subscribers as well as confidentiality and integrity of events using the techniques from pairing-based cryptography. Additionally, an algorithm is developed to preserve the weak subscription confidentiality in the presence of interest clustering of subscribers.

Zusammenfassung

Inhaltsbasierte Publish/Subscribe Verfahren sind ein weit verbreitetes Werkzeug zur Verteilung dynamischer Informationen in Form von Ereignissen, die eine hohe Entkopplung zwischen Anbietern (Publishern) und Subskribenten (Subscriber) ermöglicht. Die Sicherstellung von Skalierbarkeit ohne starke Einschränkungen bei der Ausdrucksstärke der Anfragen von Subskribenten stellt hohe Anforderungen insbesondere an hochgradig verteilte (Peer-to-Peer) Umgebungen, in denen sowohl Publisher als auch Subscriber die Weiterleitung von Ereignissen in einem Overlaynetz organisieren. Ferner erfordert die Sicherstellung nichtfunktionale Eigenschaften wie Dienstgüte (z.B. Ende-zu-Ende Verzögerung oder Bandbreitenbeschränkungen) und Sicherheit (z.B. Authentizität und Vertraulichkeit) die Lösung vieler offener Fragestellungen. Denn der Hauptvorteil der inhärenten Entkopplung von Publishern und Subscribern stellt gleichzeitig auch ein großes Hindernis zur Erfüllung nichtfunktionaler Eigenschaften dar, welches mit Hilfe herkömmlicher Methoden nicht effizient beherrschbar ist.

Das Ziel dieser Dissertation ist daher die Entwicklung von Methoden und Algorithmen, die i) eine skalierbare Verteilung von Ereignissen, ii) verschiedene Dienstgütekriterien und iii) grundlegende Sicherheitsmechanismen in hochgradig verteilten (Peer-to-Peer) Inhaltsbasierten Publish/Subscribe Systemen ermöglichen. Insbesondere folgende Kernbeiträge konnten im Rahmen der Dissertation geleistet werden:

Im ersten Beitrag wird ein Verteilter Algorithmus zur Weiterleitung von Ereignissen vorgeschlagen, der Subscriber spezifische Dienstgüteanforderungen hinsichtlich Ende-zu-Ende Verzögerung und verfügbare Bandbreite erfüllt. Des weiteren konzentriert sich der zweite Beitrag auf die Minimierung der Gesamtressourcen, indem für jeden Peer der Aufwand zum Weiterleiten und Empfangen unnötiger Nachrichten ("False Positives") minimiert wird. Dazu wurde ein auf der spektralen Graphentheorie basierender verteilter Ansatz eines Clusterverfahrens konzipiert, der auch für große Mengen an Subscribern effizient Cluster und geringe Kosten für das Weiterleiten garantiert.

Während die beiden ersten Beiträge sich ausschließlich auf die Organisation von Overlaynetzen fokusieren, werden im dritten Beitrag auch Methoden zur Anpassung von Publish/Subscribe Systemen entwickelt, die Wissen über die zugrunde liegende physische Netz und dessen Topologie miteinbeziehen. So können entsprechende Overlaytopologien zu erzeugen, die nur eine geringe zusätzliche Verzögerung und Bandbreitennutzung relativ zum möglichen Optimum auf den physischen Verbindungen aufweisen. Abschließend wird im letzten Beitrag der Dissertation ein Verfahren basierend auf Identitätsbasierter Kryptographie vorgestellt, dass sowohl Vertraulichkeit, Integrität als auch Authentizität für Ereignisse mit Hilfe eines skalierbaren Schlüsselmanagement sicherstellt. Zusätzlich werden im Rahmen des Verfahrens Algorithmen vorgestellt, die ein schwaches Vertraulichkeitskriterium für die Anfragen eines Subscribers sicherstellen.

Chapter 1

Introduction

Over the last decade, a plethora of new Internet-scale applications and services has emerged. These applications share many characteristics such as, i) hundreds of geographically distributed components (or entities), ii) exchange of high volume information content between the components in a timely manner, and iii) dynamism in terms of number of components and communication relations between them [EFGK03,PC05]. Examples of such newly emerged applications and services include news distribution [CLS03], service discovery [CDN08, KLO06], stock exchange [EFGK03, Bet00, TIB01], networked games [KTKR10, BRS02], electronic auction [SM00], person tracking [PSB03], fraud detection [Sch96], workflow management [CDNF01], road traffic management [MC02], supply chain management [SKBB09], environmental monitoring [SCW10, WXWD03], network monitoring [PJL00, MLJ10], and others.

The communication architectures and middleware infrastructures based on a request/response style of interaction (such as RPC [TA90], RMI [Sun04], DCOM [EE98] and CORBA [OMG11] etc.) have certainly reached limitations to fulfil the requirements and challenges of newly emerged applications, and a more flexible communication paradigm is needed [Que08, EFGK03]. In the recent past, the publish/subscribe paradigm was introduced as a promising alternative for building large-scale distributed applications because of its ability to support decoupled and asynchronous one-to-many and many-to-many communication interactions [Tar06].

In a publish/subscribe system, the communication is driven by the content of the information rather than the identities of producers (*publishers*) or consumers (*subscribers*). Publishers inject information into the system in the form of *events* without the knowledge of relevant set of subscribers. Likewise, subscribers specify their interest in certain events by issuing *subscriptions* without the need to know the set of publishers. The publish/subscribe infrastructure acts as a mediator, it receives the events injected by publishers and notifies all the subscribers interested in those events. Figure 1.1 depicts the interactions of publishers and subscribers in a publish/subscribe

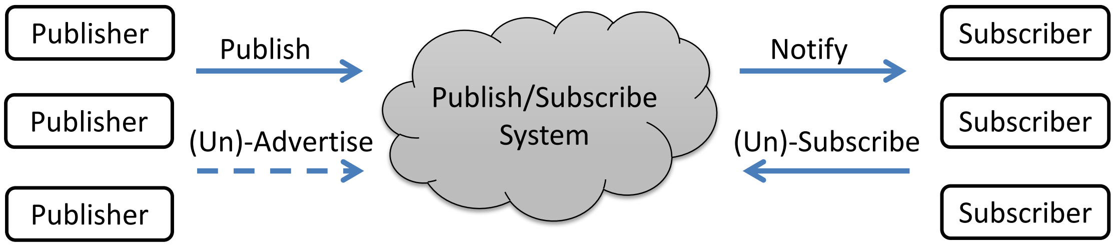


Figure 1.1: Interactions in a publish/subscribe system.

system. The publish/subscribe paradigm allows publishers to optionally express their intent to publish a particular kind of information by means of *advertisements*.

Different ways to declare interest in events have produced many variants of publish/subscribe systems [EFGK03]. The most expressive and powerful variant is content-based, which allows subscribers to define complex filtering criteria on the content of events via subscriptions. In a content-based system, events are structured as a set of attribute/value pairs, and subscriptions are expressed as conjunction of elementary constraints over the value of one or more attributes. For instance, an event might have the following content: {stock = "NASDAQ", company = "IBM", price = 300$, volume = 10,0000} and hence would match a subscription such as this: {stock = "NASDAQ" $\wedge$ price $<$ 400$ }.

An important concern in a content-based publish/subscribe system is to match the incoming events against the available subscriptions and deliver them to the relevant subscribers. To perform matching and delivery of events in an efficient and scalable manner, large-scale publish/subscribe systems are mostly organized as a network of event matchers and routers running on top of an overlay network [CRW01]. In general, two main types of overlay infrastructures are used: broker infrastructures and peer-to-peer (in short P2P) infrastructures.

Broker infrastructures rely on a network of dedicated servers (termed as *brokers*) to match and route events on behalf of publishers and subscribers [BFPB10]. Usually this requires maintenance of large routing state at each broker, e.g., by forwarding subscriptions between the brokers in the overlay network. The incoming events are routed from the publishers towards the interested subscribers by matching against the routing state maintained on the intermediate brokers in the path (cf. Figure 1.2). Typical examples of content-based publish/subscribe systems utilizing broker infrastructure are Siena [CRW00, CRW01], PADRES [JCL+10, Che11], Jedi [CDNF01], Gryphon [BSB+02, BCM+99], and Rebeca [MFB02, Müh02, FCMB06].

More recent systems are P2P-based [GSAA04, TBF+03, Pie04, VRKS06, BDFG07, BGKM07], whereby the participants of a publish/subscribe system not only act as publishers or subscribers but also contribute in forwarding events by organizing themselves in a broker-less event matching and forwarding overlay. The P2P-based (or

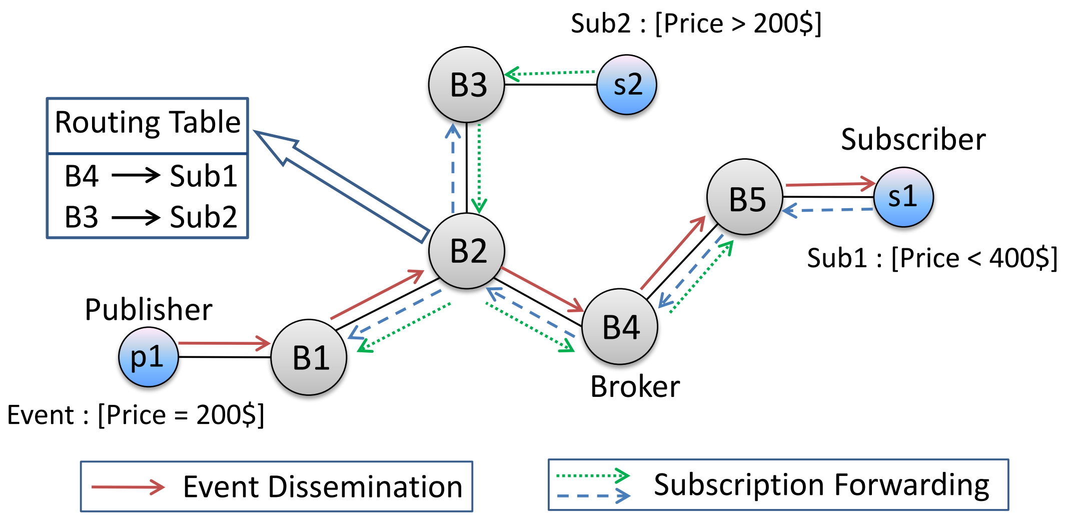


Figure 1.2: Subscription forwarding and event dissemination in broker networks.

in other words broker-less)* publish/subscribe systems improve on broker networks by avoiding fixed infrastructures and providing high decentralization, but also poses many new challenges. More precisely, in a broker-less publish/subscribe system it is still a challenge to efficiently match and route events to the relevant subscribers, while guaranteeing the expressiveness of subscriptions as well as preserving the scalability of the system [CM07]. Moreover, support for quality of service requirements such as end-to-end delays, bandwidth etc., and provision of security mechanisms such as authentication, confidentiality etc., incite many open research questions.

1.1 Research Statement and Contribution

This thesis focuses on three problem areas in broker-less content-based publish/subscribe systems.

1.1.1 Efficient Dissemination of Events

In a broker-less publish/subscribe system, all peers (publishers and subscribers) contribute in forwarding events (cf. Figure 1.3). Therefore, the rate of events that peers receive and forward without a matching subscription (*false positives*) is an important factor for the resource overhead in such a system. In particular, false positives waste system resources by increasing network bandwidth utilization and inducing extra processing load on the peers in an overlay network [Que08]. For instance, flooding (with

*In this thesis, we use the terms P2P-based or broker-less publish/subscribe interchangeably.

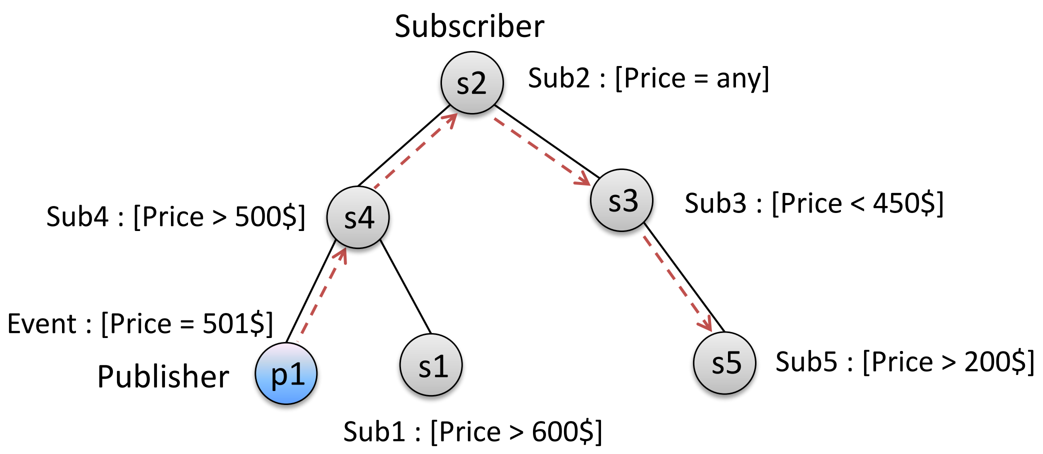


Figure 1.3: Event dissemination in a broker-less publish/subscribe system.

local filtering) can be used to deliver events to all interested subscribers in the system. However, it induces large number of false positives and is clearly not scalable.

An efficient publish/subscribe system should minimize false positives to ensure scalability w.r.t. both the number of peers (publishers and subscribers) in the system and the rate of published events [Que08, CM07]. Minimizing false positives in a broker-less content-based publish/subscribe system is very challenging due to the lack of global knowledge, high dynamicity of subscribers, expressiveness of content-based subscriptions, and continuously evolving event traffic. A number of approaches have been proposed in literature to reduce the effect of false positives in content-based systems. However, these approaches either limit the expressiveness of subscriptions to predefined numeric attributes [VRKS06, BFPB10, AGD+06, CM07, RLW+02] or assume the presence of a broker network [CJ11, BBQV07, CS04, MSRS09, BCM+99].

1.1.2 Provision of Quality of Service (QoS)

Many modern Internet applications such as distributed online games [KTKR10, MTKE11, BRS02] require support for quality of service (QoS) like timely and bandwidth efficient delivery of events. Moreover, various business applications rely on events issued by devices such as sensor networks and RFID readers, or by other applications like data warehouses and workflow machines. These events are used to detect complex situations [Luc01, KKR10], such as detection of fire or demand of stock items in the inventory etc. Delayed events are not desirable in such applications, as it may lead to wrong decisions [KKR08]. Therefore, beyond the basic function of disseminating events, publish/subscribe needs to address quality of service (QoS) metrics such as delay and bandwidth.

To address QoS in a publish/subscribe system two major challenges have to be addressed. First, given individual QoS requirements (such as delay requirements), how can such requirements be satisfied in a decoupled environment of publish/subscribe without compromising the scalability of the system? Second, how to adapt the publish/subscribe overlay according to the availability and variations in the resources of the underlying physical network [MKSB07]? For instance, if two overlay paths in the publish/subscribe system are mapped to a common underlying physical link, then the capacity/bandwidth of the underlay link is shared between the overlay paths. Increase in traffic on one overlay path may reduce the bandwidth available to the other path resulting in delayed events.

The advantage of publish/subscribe – its inherent decoupling – turns to be a major obstacle in the fulfilment of QoS requirements with conventional measures. Especially, in content-based publish/subscribe, a subscriber may receive events from many different publishers and each subsequent event from one publisher may be delivered to a completely different set of subscribers. Most of the content-based publish/subscribe systems therefore provide inadequate support for QoS [MKSB07]. In particular, existing approaches that address QoS aspects related to delay and bandwidth in publish/subscribe systems suffer from major limitations as a result of i) relying on the presence of a static broker network [WCLW06], ii) assuming the availability of global knowledge [CJ11, MSRS09], or iii) placing restrictions on the content-based model [GKK+11, CAR05, PAc+06].

1.1.3 Provision of Basic Security Mechanisms

In addition to QoS, publish/subscribe should provide supportive mechanisms to fulfil the basic security demands of modern large-scale distributed applications such as authentication, confidentiality and integrity.

Traditional security mechanisms to provide authentication, confidentiality and integrity cannot be directly applied in a content-based publish/subscribe system. For instance, end-to-end authentication of publishers and subscribers, using mechanisms such as *public key infrastructure* (PKI), conflicts with the loose coupling between publishers and subscribers. Publishers must maintain the public keys of the interested subscribers in order to encrypt events. Subscribers on the other hand, must know the public keys of all the relevant publishers in order to verify the authenticity of the received events. Likewise, traditional mechanisms to provide confidentiality by encrypting the whole event message conflicts with content-based routing.

The content-based publish/subscribe paradigm therefore possess new challenges to route encrypted events to subscribers without knowing their subscriptions and to allow subscribers and publishers to authenticate each other without knowing each other. Similar to QoS, such security issues are also not properly addressed in literature. Existing approaches towards secure publish/subscribe systems mostly rely on the presence of a traditional static broker network [RR06, BESP08, IRC10, IRC12, SLI11, CGB10].

These approaches either address security under restricted expressiveness, e.g., by using only keyword matching for routing events [SL05, SÖM09], or rely on a network of (semi-)trusted brokers that are in charge at providing content-based matching and routing [OP01, Pie04, Khu05, PEB07].

1.1.4 Contributions

The main contributions of the work performed as a part of this PhD thesis are the following:

1. We propose a P2P-based system to satisfy the individual delay requirements of subscribers in the presence of bandwidth constraints. The proposed system allows subscribers to dynamically adjust the granularity of their subscriptions according to their available bandwidth and delay requirements. Subscribers maintain the publish/subscribe overlay in a decentralized manner, exclusively establishing connections that satisfy their individual delay requirements, and that provide messages exactly meeting their subscription granularity [TKK^{+}10, TKK^{+}11, KTKR10].
2. We present distributed spectral cluster management, a method which adapts the techniques from spectral graph theory to work in distributed settings. The proposed method is applied to content-based publish/subscribe to i) significantly reduce the cost for event dissemination by clustering subscribers exploiting the similarity of events, ii) preserve the expressiveness of the subscription language, and iii) perform robustly in the presence of workload variations [TKKR12].
3. We develop a novel approach that exploits the knowledge of event traffic, user subscriptions, and the topology of the underlying physical network to perform efficient routing in a publish/subscribe system. In particular, methods are developed to infer the underlay topology among subscribers and publishers in a distributed manner. The information of the topology and the proximity between the subscribers to receive similar events is then used to construct a routing overlay with low message overhead, low relative delay penalty and low stress on the physical links [TKR13].
4. We develop a P2P-based system to maintain individual probabilistic delay bounds in a highly dynamic environment for a large number of subscribers. The proposed system observes the behaviour of the communication links and then provides delay bounds with a probabilistic reliability derived from the observations. A subscriber-centric adaptive algorithm is developed to ensure satisfaction of probabilistic delay requirements of subscribers and to maintain high bandwidth (event) dissemination trees. Additionally, scalability is ensured by a publisher-centric clustering of the dissemination trees embedded in the publish/subscribe overlay network [TKKR09].

5. We propose methods to provide authentication, confidentiality and integrity in a broker-less content-based publish/subscribe system. The authentication of publishers and subscribers as well as the confidentiality and integrity of events is ensured, by adapting the pairing-based cryptography techniques to the needs of a publish/subscribe system. Furthermore, an algorithm to cluster subscribers according to their subscriptions preserves a weak notion of subscription confidentiality. The proposed methods provide fine grained key management, and the cost for encryption, decryption, and routing is in the order of subscribed attributes [TKAR10].

Clearly, the above contributions target non-functional requirements in a publish/subscribe system from different aspects. The first contribution addresses satisfaction of subscriber-defined QoS constraints, whereas the second contribution concentrates on the minimization of overall resource usage in a publish/subscribe system. These first two contributions mostly focus on overlay-level methods to provide efficient dissemination of events and QoS. The third and fourth contributions, however, explicitly take into account the properties of the underlying network and/or its topology. The third contribution develops methods that consider the topology of the underlying network to optimize QoS metrics such as delay and bandwidth, in a publish/subscribe system. Likewise, the fourth contribution targets provision of probabilistic bounds to subscribers by accounting for the delay distribution of communication links in the underlying network. Finally, the last contribution focuses on the methods to provide security. It is important to mention here that because of the space constraints the fourth contribution [TKKR09] is not included in this thesis book.

1.2 Background: Spontaneous Virtual Networks

This thesis has been carried out within the SpoVNet (Spontaneous Virtual Networks) research project [Net12, WBH+08, BMHW11], which is funded by *Baden-Württemberg Stiftung GmbH* under the initiative BW-FIT (Baden-Württemberg Support Programme for Information Technology).

The aim of SpoVNet is to develop an overlay-based middleware which enables the spontaneous creation and deployment of application- and network-oriented services on top of heterogeneous networks. More specifically, the SpoVNet approach leads to a system that provides functionality for new communication abstractions, e.g., group communication, event-based communication etc., and addresses non-functional requirements, e.g., QoS support, security methods etc. This facilitates the development of future distributed applications, which cannot be supported by the current Internet due to its complexity and heterogeneity of its networks [Spo07].

The SpoVNet architecture is depicted in Figure 1.4. It consists of three elementary layers: the base, service, and application layer. These layers are separated by logical

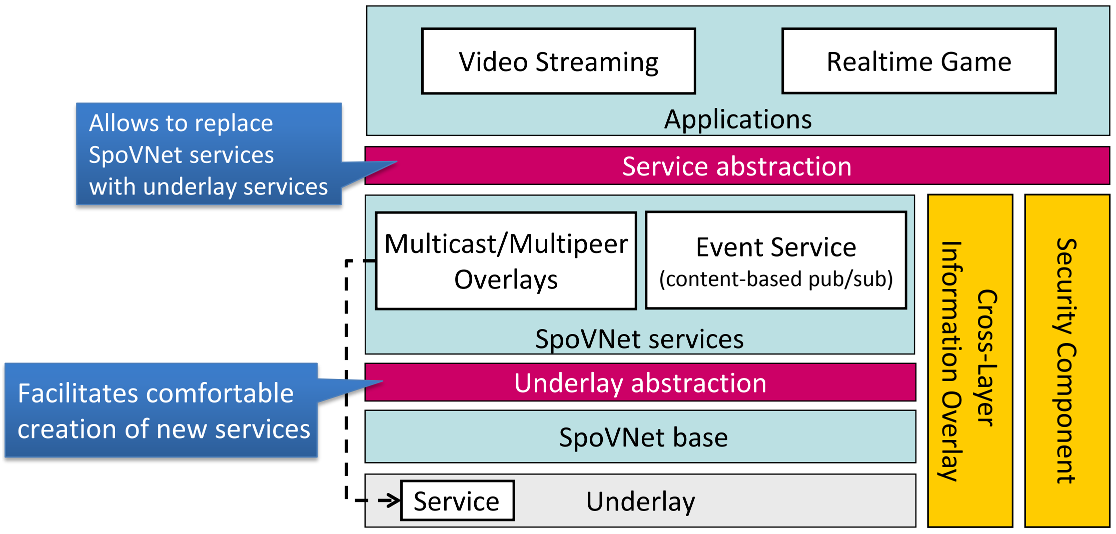

Video Streaming
Realtime Game
Applications
Allows to replace SpoVNet services with underlay services
Service abstraction
Multicast/Multipeer Overlays
Event Service (content-based pub/sub)
SpoVNet services
Cross-Layer Information Overlay
Security Component
Facilitates comfortable creation of new services
Underlay abstraction
SpoVNet base
Service
Underlay

Figure 1.4: SpoVNet Architecture [Spo07].

abstractions to hide the internal workings and complexity of lower layers from the higher layers.

The lowest abstraction (termed as *Underlay abstraction* [BHMW08]) eases the creation of overlay-based services by introducing an underlay-specific functionality layer called SpoVNet Base. It hides the details of underlay protocols by providing seamless connectivity (between the distributed instances of the services running in SpoVNet) in spite of mobility, multi-homing, heterogeneity of access technologies, and presence of middle-boxes (e.g., firewalls, NAT gateways etc.). The *Service abstraction* is made up of well-defined and persistent interfaces exposed by the overlay-based services running on top of underlay abstraction. The service abstraction enables a seamless transition from current to future generation networks by allowing functionality provided by the SpoVNet services (to realized the exposed interfaces) to be replaced by the network functionality without changing the applications [WBH$^+$08].

To support application-specified QoS requirements, it is important for the overlay-based SpoVNet services to obtain information about the relevant underlay properties, e.g., delay, bandwidth, loss rate etc., and adapt to the dynamic changes in those properties. SpoVNet addresses this by a special architectural component called *Cross-Layer Information Overlay* (CLIO) [HHNL09], which provides abstract information about the static and dynamic properties of the underlay.

Two overlay-based services are developed as a part of SpoVNet architecture to provide advance communication abstractions to the applications. The first service, *Multicast/Multi-peer overlays* (MCPO) [HW09] provides group communication. The second service (termed as *Event service*) supports event-based communication [TKKR09] with in-network detection and composition of complex events [KKR10].

The work in this thesis has contributed towards the communication part of event service, which provides the semantics of a content-based publish/subscribe system. Many of the concepts presented in this thesis are successfully applied to support the game application [MTKE11] in the SpoVNet project [KTKR10].

1.3 Structure of the Thesis

The rest of the thesis is structured as follows. Chapter 2 addresses the satisfaction of subscriber-defined delay requirements in the presence of bandwidth constraints. Chapter 3 presents distributed spectral cluster management, a method to minimize overall resource usage in a publish/subscribe system by clustering subscriptions according to recently matched events. Chapter 4 describes the underlay-aware methods to minimize the cost of routing events in terms of delay and message overhead. Security mechanisms to provide authentication and confidentiality in a broker-less content-based publish/subscribe system are described in Chapter 5. Finally, the summary of our contributions and an outlook on possible future work are given in Chapter 6.

Chapter 2

Subscriber-defined QoS Constraints

2.1 Introduction

The evolution of publish/subscribe has followed two main objectives, namely an increased decentralization and an increased orientation on the participants' specific needs. Former static broker-based architectures [CRW01, BSB$^+$02, CDNF01] were overcome by decentralized systems [AGD$^+$06, AT06, YH07, CF05] where publishers and subscribers contribute as peers to the dynamic maintenance of the publish/subscribe system and where they perform the dissemination of events collectively. Specific needs of subscribers were met by the transition from topic-based and channel-based publish/subscribe [CMTV07,BBQ$^+$07] to content-based publish/subscribe [CRW01,CS05].

There is still potential for the adaptation of publish/subscribe to peer-specific needs. For instance, many current systems assume that all subscribers expect the same quality of service (QoS) for their requested events. In fact, for many real-world settings, events are of different importance to individual subscribers which can therefore subscribe with different QoS requirements. Consider, for example, meteorological sensor information such as temperature and wind fields. The data itself is relevant for a large number of application entities such as news agencies, traffic monitoring, energy management, and rescue services. However, while local rescue services need to react fast and cannot tolerate large transmission delays, other recipients like a weather forecast service which has a large prediction window do not have that strict delay requirements. Accounting for individual QoS requirements is promising to better utilize the system's resources. Again, resources such as bandwidth should be considered peer-specific constraints for the maintenance of the system rather than system constants.

Considering peer-specific constraints in publish/subscribe systems is severely complicated by its inherent decoupling. Therefore, in literature, only few approaches have addressed QoS for publish/subscribe. Solutions supporting message delay bounds either assume static topologies [WCLW06] or rely on complex management protocols such as advertisement and subscription forwarding to manage end-to-end state information with respect to each publisher [CAR05, TKKR09]. Peer-specific resource contribution and its inter-dependencies to user-specific delay requirements have not been discussed yet in literature.

In this chapter, we present a broker-less content-based publish/subscribe system which satisfies the peer's individual message delay requirements and supports system stability by accounting for resources contributed by individual peers [TKK+10, TKK+11]. Subscribers arrange in an overlay so that subscribers with tight delay requirements are served before subscribers with looser ones. Peers contribute some of their bandwidth on receiving and forwarding events which do *not* meet their own subscriptions (*false positives*) in exchange for an increased opportunity to satisfy their individual delay requirements. Therefore, peers with tight delay requirements also significantly contribute to the stability of the publish/subscribe system, while they are still in control of their individual permissible ratio of false positives and thus can consider their bandwidth constraints. The evaluations demonstrate the viability of the proposed system under practical workloads and dynamic settings.

2.2 System Model and Problem Formulation

We consider a broker-less content-based publish/subscribe system consisting of an unbounded set of peers. Peers leave and join the system at arbitrary time, and they can fail temporarily or permanently. The peers act as publishers and/or subscribers which connect in an overlay and forward events to relevant subscribers. The set of overlay connections of a peer s can be classified into incoming connections $F_{in}(s)$ and outgoing connections $F_{out}(s)$. We support event forwarding using an out-degree constraint m. It obliges peer s to be ready to forward received messages up to m times ($|F_{out}(s)| \leq m$). Rate $R(s)$ of events received over connections in $F_{in}(s)$ is therefore constrained: it must not consume more than a fraction $\frac{B(s)}{m+1}$ of the available bandwidth $B(s)$.*

The basis for all events and subscriptions is the event space denoted by Ω. It is composed of a global ordered set of d distinct attributes (A_i): $\Omega = \{A_1, A_2, \ldots, A_d\}$. Each attribute A_i is characterized by a unique *name*, its *data type* and its *domain*. The data type can be any ordered type such as integer, floating point and character strings. The domain describes the range $[L_i, U_i]$ of possible attribute values.

The relations between events, subscriptions and advertisements can be demonstrated by modelling Ω geometrically as a d-dimensional space so that each dimension represents an attribute. A publisher's advertisement is a sub-space of that space, and a published

*Bandwidth estimation tools such as pathchar [Jac97], clink [Dow99b, Dow99a], pchar [Mah00] or nettimer [LB01], can be used to measure the available bandwidth.

event is a single point Ω in the space. A subscription is a hyper-rectangle in Ω. An event is *matched* by a subscription iff the point Ω defined by the event is located within the hyper-rectangle defined by the subscription. A subscription f_1 is *covered by* a subscription f_2, denoted as $f_1 \prec f_2$, iff the hyper-rectangle of f_1 is enclosed in the hyper-rectangle of f_2.

Apart from that, we allow a subscriber s to specify the delay $\Delta(s)$ that it is willing to tolerate when receiving events from any of its relevant publishers.

In the publish/subscribe system described above, a peer clearly has two concerns. The first is to receive all relevant messages in compliance with its delay requirements. The second is, for the sake of saving bandwidth, to receive and forward only messages that exactly match the peer's subscription.

More precisely, let S be a set of subscribers and P_S the set of publishers that publish events matching the subscriptions of S. Let $\mathcal{E}^O$ denotes the set of all overlay links and $path(p,s) = \{\langle p, i_1\rangle, \langle i_1, i_2\rangle, ..., \langle i_m, s\rangle\} \subseteq \mathcal{E}^O$ defines the set of overlay links on the path from a publisher $p \in P_S$ over intermediate nodes i_j to a subscriber $s \in S$. The delay on this path is defined as $D(p,s) = \sum_{\langle i,j\rangle \in path(p,s)} d^O(i,j)$, where $d^O(i,j)$ denotes the link delay on a link $\langle i,j\rangle \in \mathcal{E}^O$. The objective is to maintain the publish/subscribe overlay network in the presence of *dynamic* sets of publishers P and subscribers S, so that

1. the delay constraints of maximum number of subscribers are satisfied with respect to the sets of their relevant publishers, i.e., the number of satisfied subscribers is maximized (ideally, in the presence of sufficient resources, $\forall s \in S, \forall p \in P_S : D(p,s) \leq \Delta(s)$), and
2. each subscriber can dynamically adjust the rate of false positives it receives so that its bandwidth constraints are not violated, i.e., $\frac{B(s)}{m+1} \geq R(s)$.

Our approach can work with any monotonically increasing delay metric. However, for simplicity, in our algorithm description we use the hop count as delay metric, i.e., $D(p,s) = |\{\langle i,j\rangle \in \mathcal{E}^O | \langle i,j\rangle \in path(p,s)\}|$. Section 2.7 briefly highlights the changes necessary to make the approach work with round trip time (RTT) as a delay metric.

2.3 Approach Overview

Meeting the objectives presented in Section 2.2 amounts to finding a trade-off between two contradicting goals: to minimise resource usage by avoiding false positives (i.e., a subscriber s receives and therefore forwards only messages that match its own subscription), and to ensure scalability by balancing the contribution of the peers according to their available resources.

Fulfilling the first goal affects the scalability of the overall system especially in the presence of out-degree constraints. In the content-based model, subscriptions often

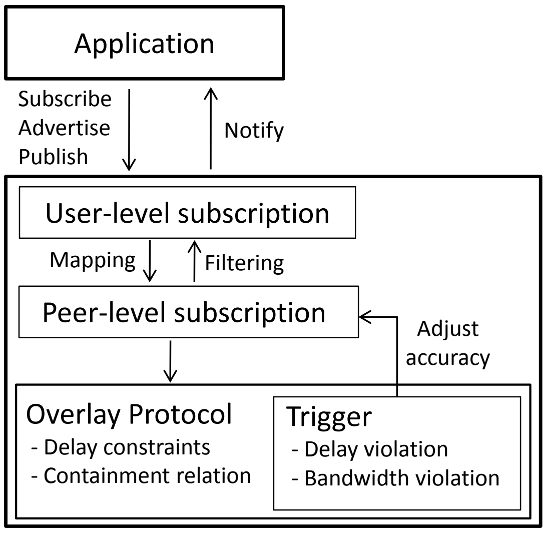


Figure 2.1: Overall architecture.

intersect with each other rather than being in a containment relationship. Hence, the complete removal of false positives may require subscribers to maintain large number of incoming connections in order to cover their subscriptions [Que08]. Therefore, false positives cannot be completely avoided and peers need to contribute resource in terms of false positives to ensure scalability. However, allowing individual peers to induce false positives by arbitrarily coarsening their subscriptions without any regularity is unrewarding due to the fact that coarser subscriptions may still intersect instead of being in a containment relationship.

We therefore propose to coarsen subscriptions systematically by distinguishing between two levels of subscriptions: user-level and peer-level, as shown in Figure 2.1. The user-level subscription represents the original subscription as defined by the application. The peer-level subscription is an approximation of the user-level subscription and defines which events a peer actually receives.

2.3.1 Peer-level Subscription

The peer-level subscription is created by spatial indexing [GG98, MJ07]. The event space is divided into regular sub-spaces which serve as enclosing approximations for user-level subscriptions. The sub-spaces are created by recursive binary decomposition of the event space Ω. The decomposition procedure divides the domain of one dimension after the other and recursively starts over in the created sub-spaces.

Figure 2.2 visualizes the decomposition process with the aid of a binary tree. Each tree level represents one step of the recursive process, starting with the root where Ω is still undivided. The first partition of Ω is created by dividing it in the first dimension and

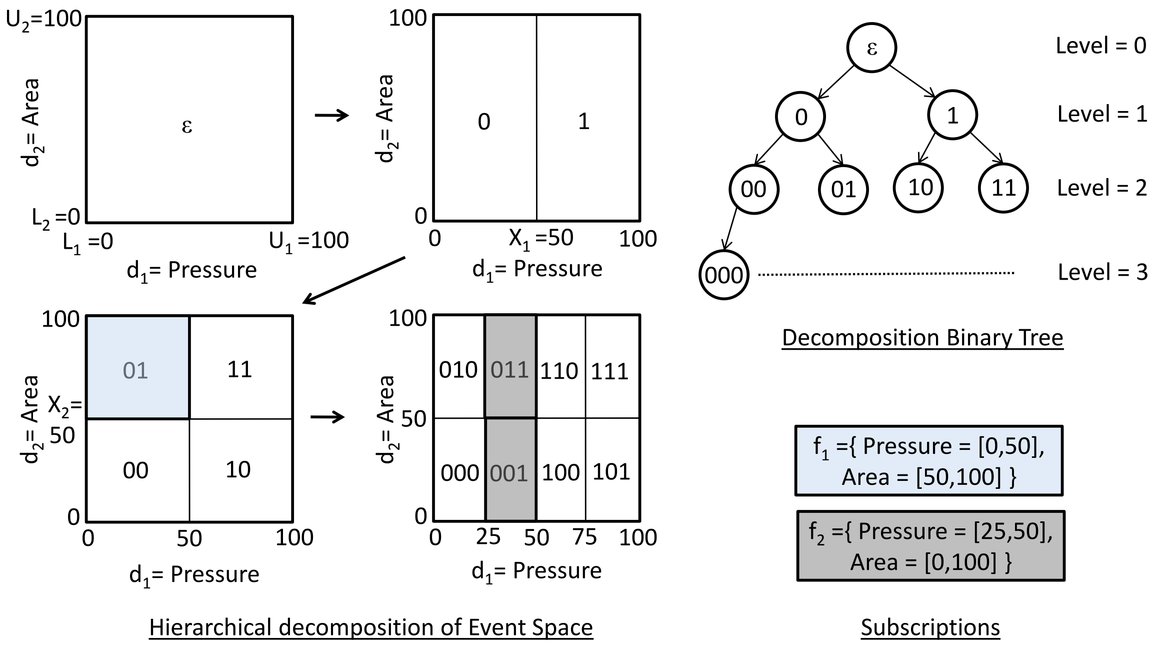


Figure 2.2: Event space decomposition using spatial indexing.

creates two sub-spaces. In one sub-space, the domain of the first dimension is $[L_1, X_1]$ and in the other sub-space, it is $(X_1, U_1]$. The break line X of a domain always equals $\frac{L+U}{2}$ so that the sub-spaces cover equally sized domains of the divided dimension.

Sub-spaces can be identified by *dz-expressions* [OS90]. A dz-expression is a bit-string of "0"s and "1"s, which is empty (ϵ) for Ω. Each time a sub-space is divided, its dz-expression is inherited as prefix for the dz-expressions of the newly created sub-spaces. The new sub-space with the lesser domain of the divided dimension appends "0" to the prefix and the one with the higher domain appends "1". Sub-spaces and their dz-expressions generated by this process have the following properties.

- The size of a sub-space is related to the length of its dz-expression $|dz|$. A dz-expression with smaller length represents a bigger sub-space.
- A sub-space represented by dz-expression dz_1 is *covered by* the sub-space represented by dz_2 iff dz_2 is a prefix of dz_1. More precisely, the covering relation $\prec$ on dz-expressions is defined as follows. Let $dz[0, i]$ be the prefix of dz's bit string that is $i + 1$ bits long. Then

$$dz_1 \prec dz_2 \Leftrightarrow (|dz_1| \geq |dz_2|) \wedge (dz_1[0, |dz_2| - 1] = dz_2)$$

The peer-level subscription of a peer p can be composed of several sub-spaces and is therefore represented by a set of dz-expressions denoted by $DZ(p)$ with $DZ(p) = \{dz_i \mid i \geq 1\}$. For instance, in Figure 2.2, the accurate mapping of $f_2 = \{Pressure =$

$[25, 50] \wedge Area = [0, 100]\}$ requires two sub-spaces in its peer-level subscription. The mapping is $f_2 \longmapsto \{001, 011\}$.

If the mapping between the subscriptions at user-level and peer-level is identical, the peer will only receive events matching its user-level subscription. In general, however, a peer can coarsen its peer-level subscription in a regular manner so that additional events can occupy a share of its bandwidth. For example, f_2 in Figure 2.2 can be coarsened by mapping it to the sub-space 0, i.e., $f_2 \longmapsto \{0\}$.

The regularity of sub-spaces created by spatial indexing is advantageous due to the fact that overlapping sub-spaces are always in a containment relationship, which can be directly mapped to the overlay structure, as discussed in Section 2.4. Additionally, subscriptions can be coarsened or refined in a regular manner. This lesser degree of freedom in the selection of false positives helps to take into account the rate and distribution of events in the decomposition process. In particular, it allows anticipated bandwidth estimation of the sub-spaces that may be included in coarser subscription, as detailed in Section 2.5.

2.4 Publish/Subscribe Overlay Protocol

In this section, we describe the organization and maintenance of the publish/subscribe overlay in the presence of subscriber-specified delay requirements and bandwidth constraints. Moreover, we present an algorithm to route events in the proposed publish/subscribe overlay network.

2.4.1 Overlay Organization

The organization of the publish/subscribe overlay should provide the following properties.

- *Property 1:* Each subscriber peer q receives all and only events matching its peer-level subscription $DZ(q)$. Therefore, peers are organized based on the $\prec$ relation between their dz-expressions, as shown in Figure 2.3.
- *Property 2:* Peers are placed in the overlay network according to their delay requirements. Therefore, peers with tighter delay requirements are closer to their relevant publishers.

Each peer q maintains a set of incoming $F_{in}(q)$ and outgoing connections $F_{out}(q)$. The following two properties ensure that the bandwidth constraints of that peer are considered in a proper way.

- *Property 3:* A peer q receives no duplicate events, i.e., each point in its peer-level subscription is covered exactly once by the subscriptions of subscribers or by all of

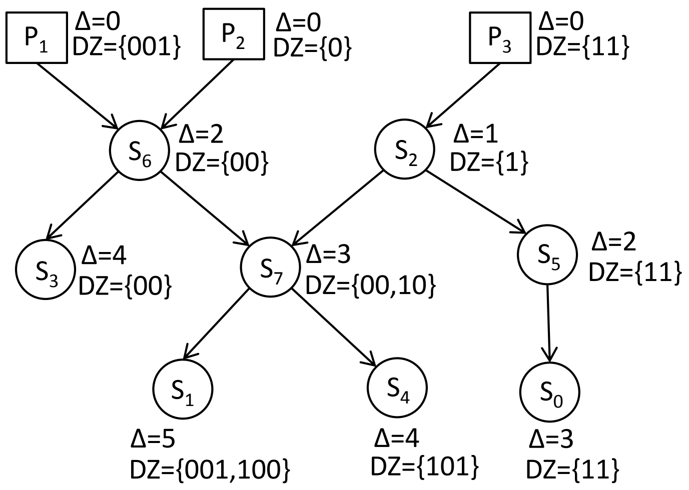


Figure 2.3: Publish/Subscribe overlay with delay requirements and $m = 2$.

the relevant publishers, in $F_{in}(q)$. The relevant publishers are the ones who publish events matching the peer-level subscription of q. For example, in Figure 2.3 subscriber s_6 is connected to the relevant publishers p_1 and p_2 to receive all the events matching its peer-level subscription.

- *Property 4:* A peer q forwards each received event up to m times, i.e., $|F_{out}(s)| \leq m$. In other words, each point in the peer-level subscription of q can be covered up to m times by the subscriptions of peers in $F_{out}(q)$.

2.4.2 Overlay Maintenance

Subscribers maintain the overlay in a decentralized manner by connecting and disconnecting other peers. In particular, subscribers satisfy their peer-level subscriptions and delay requirements by connecting to subscribers or publishers that have covering subscriptions and tighter delay requirements. Thereby, subscribers just rely on the subscription and the delay constraints of the peers they are connecting to, and on the fact that these in turn connect to suitable peers.

2.4.2.1 Data Structures

Here, we briefly describe the main data structures utilized by the peers to maintain the proposed publish/subscribe overlay.

Peer View: Each subscriber maintains a peer view *pView* that caches information about peers (in the system) which have covering subscriptions. The information maintained by *pView* is utilized by the subscribers to find suitable parents, i.e., peers with covering subscriptions and tighter delay requirements, to connect in the overlay network.

Each item in *pView* stores the identifier of a peer q, its peer-level subscription $DZ(q)$ and its delay requirement $\Delta(q)$. Moreover, each item is associated with an age counter, which is used to remove stale entries from *pView*.

List of Relevant Publishers: Besides maintaining *pView*, each subscriber q also keeps a list of those publishers which are relevant because they have covering advertisements, i.e., publish events matching the peer-level subscription of q. The list is represented by *pubList* and allows subscribers to connect to the relevant publishers (cf. Section 2.4.2, *Property 3*). Moreover, *pubList* is used to expedite the inclusion of new publishers in the overlay network during event dissemination (cf. Section 2.4.4).

An entry in *pubList* contains the identifier of a relevant publisher p and its peer-level advertisement $DZ(p)$. Similar to *pView*, an age counter is attached to each entry in *pubList*.

Incoming and Outgoing Connections: A peer q also stores data that refers to its connections in $F_{in}(q)$ and $F_{out}(q)$. Connections in $F_{in}(q)$ supply q with event messages that cover $DZ(q)$ once (cf. Section 2.4.2, *Property 3*). Therefore, if q has multiple parents, it has to split its $DZ(q)$ into distinct sub-sets (i.e., sub-sets of dz-expressions) and subscribe to each parent with a different sub-set, respectively. These sub-sets are denoted by $\{dz(a_i) \mid a_i \in F_{in}(q)\}$ and are stored with the respective $F_{in}(q)$ connections to q's parents.

$F_{out}(q)$ connections are used to forward events to children. Children do not necessarily subscribe the entire $DZ(q)$. Therefore, q stores the $dz(a_i)$ of the respective child with each $a_i \in F_{out}(q)$ and forwards only events matched by $dz(a_i)$ to that child. Moreover, the delay requirement $\Delta(a_i)$ of each child a_i is also stored with the respective F_{out} connection.

2.4.2.2 Maintenance of pView and pubList

For the satisfaction of its subscription, a subscriber peer p needs to discover suitable parent(s) (i.e., a subscriber or relevant publishers) for each of its dz_i in $DZ(p)$. Furthermore, dynamic conditions such as churn, failures and changes in the delay requirements may require previously suitable parent(s) to be replaced. Therefore, each subscriber maintains a peer view *pView* and a publisher list *pubList* that store information about the subscribers and the publishers which are relevant because they have covering subscriptions and advertisements respectively.

To maintain *pView* and *pubList*, subscribers need to receive DZ (i.e., peer-level subscription or advertisement) and Δ (i.e., delay requirement) of a random subset of peers and decide whether they qualify as potential parents. The information about the random subset of peers (in the system) is gathered by employing an epidemic protocol [JVG+07, EGH+03, Kol03].

More precisely, each subscriber peer p periodically runs the *updatePviewAndPubList* procedure (cf. Algorithm 1, *lines 1-8*) and receives a partial membership view (*mList* in the Algorithm 1) of the system. That is, the given subscriber p receives information about a subset of all peers in the system. For each peer q in *mList*, subscriber p decides whether q qualifies to be included in $pView$ (or *pubList*). Peer q is added to $pView$ or *pubList* only if a dz-expression (dz) in the peer-level subscription (or advertisement) of q has a containment relationship with one of the dz-expressions in $DZ(p)$. Finally, the aged entries, i.e., peers without an information update for a long time, are removed from $pView$ and *pubList* to ensure resilience against system dynamics such as churn, failures and changes in subscriptions (or advertisements).

2.4.3 Management of F_{in} and F_{out} Connections

Periodically, each peer p runs the *connectionManagement* procedure (cf. Algorithm 1, *lines 9-15*) to check whether each dz_i in $DZ(p)$ is covered either by the subscription of a subscriber or by all of the relevant publishers in $F_{in}(p)$. Here, by relevant we imply those publishers which provide the events matching dz_i. In case a dz_i is not covered, the *findBestParent* routine selects a parent from $pView$, whose subscription covers dz_i. Peer p sends a connection request to this potential parent once it is selected.

Connection Request: Upon reception of a connection (CONNECT) request from a peer p, the potential parent q will normally acknowledge the connection, but it will reject the request if $\Delta(p) \geq \Delta(q)$ does not hold. In this case, q sends a hint about the most suitable parent for p according to q's knowledge.

Accepting peer p as a child may violate the out-degree constraints of the peer q. In this case, the *peersToDisconnect* routine prepares the disconnection from the child with the most selective subscription. Ties between the candidate children to disconnect are resolved by selecting the one with a lower delay requirement. The disconnection strategy ensures that the peers with less selective (or coarse) subscriptions and tight delay requirements are placed higher in the overlay network, i.e., placed near the relevant publishers (cf. Section 2.5). If peer p is chosen for disconnection, it will receive a hint (POTENTIALPARENT) message instead of a connection acknowledgement.

Upon reception of a hint (POTENTIALPARENT) message, a peer will add the hint to its $pView$ and consider it as a potential parent in its next iteration of the *connectionManagement* procedure.

Connection Acknowledgement: Upon reception of an acknowledgment (ACK) message, a peer p ensures that its peer-level subscription is covered exactly once by parent subscribers (cf. Algorithm 1, *lines 30-35*). This ensures that p's bandwidth is not wasted in receiving duplicate events. For sub-spaces of p's subscription that cannot be covered by parent subscribers, coverage must be accomplished by connecting to *all* relevant publishers. Thus, for each of such sub-spaces that are only covered by one or more publishers, p continues to search for relevant publishers or subscribers.

Algorithm 1 Publish/subscribe overlay maintenance

```
1: procedure updatePviewAndPubList do
2:   for all $q \in mList$ do // Uniform peer sample derived from epidemic protocol
3:     if $\exists dz_p \in DZ(p), \exists dz_q \in DZ(q) : dz_p \prec dz_q \vee dz_q \prec dz_p$ then // $p$ & $q$ have covering subscriptions
4:       if $\Delta(q) == 0$ then // $q$ is publisher so add to publisher list
5:         $pubList = pubList \cup q$
6:       else
7:         $pView = pView \cup q$
8:   Increase age and remove old elements from $pView$ and $pubList$
9: procedure connectionManagement do
10:   while true do
11:     updatePviewAndPublisherList()
12:     if $\exists dz_i \in DZ(p) | dz_i$ is not covered then
13:       $parent =$ findBestParent$(pView, dz_i)$
14:       $pView = pView - parent$
15:       trigger Send(CONNECT, p, parent, $dz_i$ , $\Delta(p)$ )
16:  upon event Receive(CONNECT, p, q, $dz(p)$, $\Delta(p)$) at peer $q$ do
17:   if $\Delta(p) \geq \Delta(q)$ then
18:     $F_{out}(q) = F_{out}(q) \cup p$
19:     if $|F_{out}(q)| > m$ then
20:       $peer[] =$ peersToDisconnect()
21:       for all $t \in peer$ do
22:         $parent =$ findBestParent$(pView \cup F_{out}(q), dz(t))$
23:         trigger Send(DISCONNECT, t)
24:         trigger Send(POTENTIALPARENT, t , parent)
25:       if $p \notin peer$ then
26:         trigger Send(ACK,q)
27:   else // $\Delta(p) < \Delta(q)$
28:     $parent =$ findBestParent$(pView \cup F_{in}(q), dz(p))$
29:     trigger Send(POTENTIALPARENT, p, parent)
30:  upon event Receive(ACK, q) at peer $p$ do
31:   $F_{in}(p) = F_{in}(p) \cup q$
32:   $iCon = \{dz(a_i) : a_i \in F_{in}(p) \wedge \Delta(a_i) \neq 0\}$ // current list of subscriptions from non-publisher parents
33:   Remove all $dz$ from $iCon$ which are covered by $dz(q)$ // $DZ(p)$ should be covered exactly once
34:   for all $a \in F_{out}(p) : \Delta(a) = \Delta(p) = \Delta(q) \wedge dz(a) = dz(p) = dz(q)$ do
35:     trigger Send(DETECTCYCLE, $p$, $dz(p)$, $\Delta(p)$)
36:  upon event Receive(DETECTCYCLE, originator, $dz(q)$, $\Delta(q)$) at peer $p$ do
37:   if originator = p then
38:     trigger Send(DISCONNECT, q) // cycle is present through parent $q$
39:   else
40:     if $\Delta(q) = \Delta(p) \wedge dz(p) = dz(q)$ then // delay constraints and subscription are same
41:       for all $a \in F_{out}(q) : \Delta(a) = \Delta(p) \wedge dz(a) = dz(p)$ do
42:         trigger Send(DETECTCYCLE, originator, $dz(p)$, $\Delta(p)$)
```

Accepting connection from a parent may introduce a cycle in the system. To preserve acyclic topology, we employ a strategy similar to the edge-chasing deadlock detection algorithm [Mos85, Kna87, MM84]. In particular, whenever a peer p establishes an incoming connection, a control (CYCLEDETECT) message is passed down to the peers in $F_{out}(p)$. Peers always connect to the parents with covering subscriptions and thereby cycles can only occur between the peers with similar subscription and delay constraints. This fact is used to reduce the overhead of control messages by forwarding them to only those peers, which have same subscription and delay constraints. The

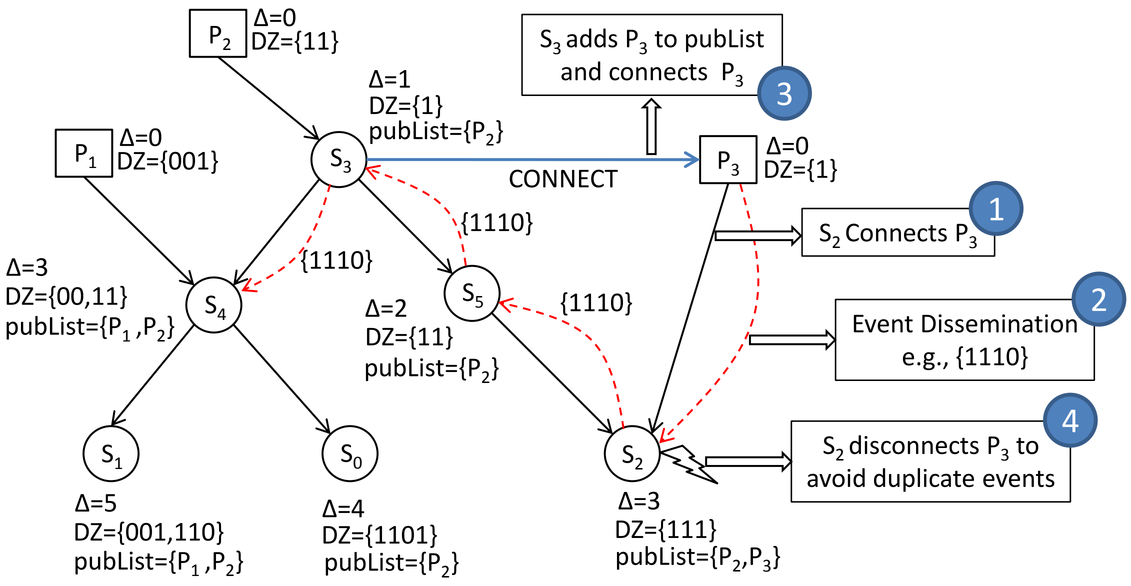

P₂ Δ=0 DZ={11}
P₁ Δ=0 DZ={001}
S₃ Δ=1 DZ={1} publList={P₂}
S₃ adds P₃ to pubList and connects P₃
3
CONNECT
P₃ Δ=0 DZ={1}
{1110}
S₂ Connects P₃
1
S₄ Δ=3 DZ={00,11} pubList={P₁,P₂}
S₅ Δ=2 DZ={11} pubList={P₂}
Event Dissemination e.g., {1110}
2
S₂ disconnects P₃ to avoid duplicate events
4
S₁ Δ=5 DZ={001,110} pubList={P₁,P₂}
S₀ Δ=4 DZ={1101} pubList={P₂}
S₂ Δ=3 DZ={111} pubList={P₂,P₃}

Figure 2.4: Placement of new publisher in publish/subscribe overlay network.

reception of the control message by the originator signals the presence of a cycle and the connection to the respective parent is disconnected (cf. Algorithm 1, *lines 36-42*).

2.4.3.1 Placement of Publishers

Similar to subscriptions, an advertisement of a publisher is represented by a set of dz-expressions (DZ). Each publisher p periodically communicates its $DZ(p)$ to a random subset of other peers (using epidemic protocol), which decide whether to include this information into their *pubList*, as mentioned in Section 2.4.2.2. This allows automatic discovery and inclusion of the publishers in the overlay network as a result of connection requests (CONNECT) from subscribers. More precisely, a subscriber q that is unable to find the parent subscribers for its peer-level subscription sends connection requests to the publishers in its *pubList*. This happens when subscriber q has very tight delay requirement which can only be satisfied by direct connections to the publishers. Similarly, if a subscriber q discovers a newly arrived publisher p (in its *pubList*) from which it is not receiving events via $F_{in}(q)$ connections, it sends connection request to p. This expedites the inclusion of new publishers in the publish/subscribe overlay network, as described later in Section 2.4.4. For instance, in Figure 2.4 a newly arrived publisher p_3 is initially connected to subscriber s_2 and eventually gets connected by the top most subscriber s_3 in the dissemination tree. Figure 2.4 is also described in more detail in Section 2.4.4.

On receiving connection requests, publishers maintain their outgoing connections (F_{out}) similar to subscribers by following the *lines 18-26* of Algorithm 1. More precisely, a

Algorithm 2 Event dissemination

```
1:  upon event Receive( EVENT , msg, publisher, p, q ) at peer $p$ do
2:   if $q \in F_{out}(p)$ then
3:      if dz(publisher) is not covered by any peer in $F_{in}(p)$ then // dz(publisher) $\not\prec \{\bigcup_{a_i \in F_{in}(p)} dz(a_i)\}$
4:         trigger Send(CONNECT, $p$, publisher, $dz(p)$, $\Delta(p)$ )
5:   $D(msg) = D(msg) + d^O(q, p)$
6:   if $q \in F_{out}(p) \vee q ==$ publisher then
7:      for all $a_i \in F_{in}(p) : \Delta(a_i) \neq 0 \wedge msg.dz \prec dz(a_i)$ do
8:         trigger Send(EVENT, msg, publisher, $a_i$, $p$)
9:   if $q \in F_{out}(p) \vee q \in F_{in}(p)$ then
10:     for all $a_i \in F_{out}(p) : a_i \neq q \wedge \Delta(a_i) \neq 0 \wedge msg.dz \prec dz(a_i)$ do
11:        trigger Send(Event, msg, publisher, $a_i$, $p$)
```

publisher p can accept connection request (CONNECT) from any peer q because the delay constraint of q can never be violated. But accepting q as a child may infringe the out-degree constraints of the publisher p. In this case, publisher p runs *peersToDisconnect* procedure to select child with the most selective subscription and a large Δ for disconnection, similar to subscribers. However unlike subscribers, publisher p cannot provide hint of a potential parent to the disconnected child due to the fact that *pView* is not maintained by the publishers.

2.4.4 Event Dissemination

On reception of an event, a peer p runs Algorithm 2. Usually, a peer receives events from F_{in} and forwards them to all connections in F_{out} with a dz that covers the event. However, a newly arrived publisher may not be initially visible in the *pubList* of all relevant peers, i.e., peers which require events from the (newly arrived) publisher to satisfy their peer-level subscriptions. Therefore, the new publisher may be connected by the peers down in the dissemination tree. For example, in Figure 2.4 new publisher p_3 is initially visible only in the *pubList* of subscriber s_2 and is therefore connected to s_2. To handle such cases, event dissemination requires an additional rule: Each event received from a publisher or from F_{out} connections is forwarded to all other matching F_{in} and F_{out} connections. For instance, in Figure 2.4 an event {1110} from publisher p_3 is forwarded to the F_{in} connection of subscriber s_2.

If peer p receives an event from its F_{out} connections that originates from a publisher, and the publisher's advertisement is not completely covered by the dz of p's parents (i.e., p is the top most peer), then p sends a connection request to the publisher. For example, on receiving the event {1110} from the child subscriber s_5, the top most subscriber s_3 in the dissemination tree of Figure 2.4 establishes the connection with publisher p_3. This strategy speeds up the placement of new publishers in the overlay network and allows the peers with the most tight delay constraints to connect directly to publishers. It is worth noting that once the subscriber s_2 in Figure 2.4 starts receiving events published by p_3 from its F_{in} connection, it disconnects from p_3 to avoid duplicates.

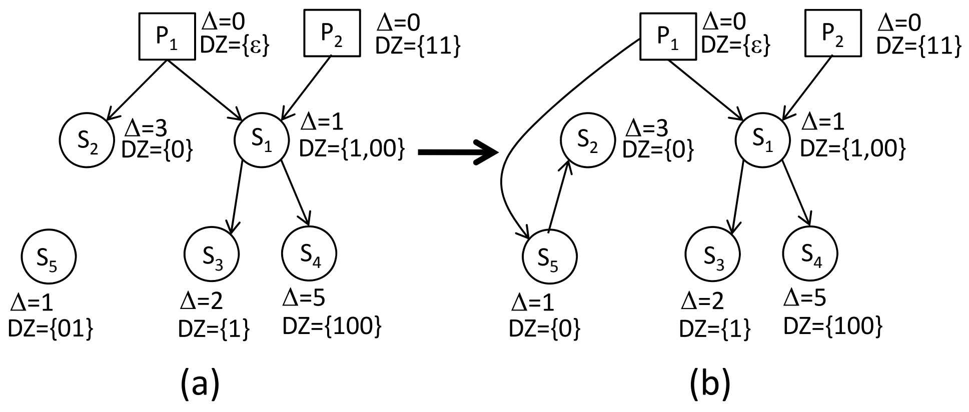


Figure 2.5: Example Scenario with $m = 2$.

If the delay requirements of peer p and its parents are strictly ordered then $\Delta(p)$ will always be satisfied as long as the delay constraints of the parents are not violated. However, if both p and the parent have same delay requirements then $\Delta(p)$ may be violated. In this case, peer p disconnects from its parent if the delay encountered by the received event violates $\Delta(p)$.

2.5 Triggers for Change in Accuracy

Until now we have described the organization and maintenance of the publish/subscribe overlay in the presence of subscriber-specified delay requirements. Nevertheless, we need additional mechanisms to ensure the scalability of the scheme. Sometimes a peer cannot find any potential parent to satisfy its delay constraints. In Figure 2.5(a), for instance, subscriber s_5 has a rather selective subscription and tight delay requirements. If publisher p_1 cannot accommodate more children, then s_5 can only connect to s_2 according to Algorithm 1. However, doing so violates the delay constraints of s_5. In this case, s_5 can coarsen its peer-level subscription according to its bandwidth constraints in order to be placed between p_1 and s_2. This is possible because the overlay maintenance strategy places subscribers with less selective subscriptions higher in the dissemination graph (cf. Algorithm 1, *lines 16-26*). Therefore, subscribers can improve the probability to satisfy their delay requirements by agreeing to a coarser subscription, as shown in Figure 2.5(b). Similarly, if changes in the event rate lead to violation of the bandwidth constraints, a subscriber (with a coarser subscription) increases the accuracy of its peer-level subscription to reduce false positives (cf. Algorithm 3, *lines 4-6*). Methods to refine or coarsen a peer-level subscription according to the bandwidth constraints are detailed in Section 2.5.1.

In addition to individual delay requirements, coarser subscriptions increase the overall satisfaction of subscriptions. Peers (subscribers or publishers) with out-degree con-

Algorithm 3 Triggers for change in accuracy

```
1: upon event TimeOut do
2:   if $\exists dz_i \in DZ(p) | dz_i$ is not covered then
3:     reduce accuracy of peer-level subscription by coarsening
4: upon event BandwidthViolated do
5:   increase accuracy of peer-level subscription accordingly
6:   remove subscribers in $F_{out}(p)$ which are not covered by $DZ(p)$.
```

straints cannot satisfy a large number of fine-grained disjoint subscriptions. However, disjoint subscription *hotspots* are common when accessing content with Internet-like popularity. In that case, even a small number of coarser subscriptions allow subscribers to find intermediary parent peers that satisfy their subscriptions, instead of being rejected by an overloaded publisher with exhausted outgoing connections (cf. Section 2.8.2).

In the following sections, we describe the mechanisms to adjust the accuracy of the mapping between user-level and peer-level subscriptions according to subscriber-specific bandwidth constraints.

2.5.1 Accuracy of Subscription Mapping

A subscriber can reduce the accuracy of the peer-level subscription by using a coarser mapping $\xmapsto{c}$ from the user-level subscription f to a smaller set of coarser dz-expressions DZ^C. Reduced accuracy causes false positives and increases bandwidth usage. Therefore, a condition for selecting a $\xmapsto{c}$ mapping on peer s is that the reduction of accuracy does not violate the peer's bandwidth constraint $B(s)$. The subscriber can ensure this by iteratively selecting another coarse mapping, thereby refining or coarsening individual dz-expressions and thus controlling the overall rate of received events.

The bandwidth usage induced by each dz-expression depends on the rate of events matched by the expression. Therefore, for each sub-space represented by a dz-expression in DZ^C, the subscriber continuously monitors the event rates in the sub-space that is divided once less (DZ^{-1}) and in the sub-spaces that are divided once more (DZ^{+1}). The latter can be calculated by counting the received messages, while the event rate in the coarser sub-space is estimated by means of statistical aggregation [JKS04]. The estimation of the event rate in the coarser sub-space relies on the measurements of other subscribers that are currently subscribed to the coarser sub-space or a part of it. For instance, event rate in the coarser sub-space 1 can be estimated from the event rates monitored by other subscribers in sub-spaces 10 and 11. The epidemic protocol employed in Section 2.4.2 to maintain *pView* and *pubList* can be easily extended to receive event rates monitored by other subscribers. To be more precise, subscribers periodically exchange event rates experienced by their currently subscribed sub-spaces along with their peer-level subscriptions and delay requirements using epidemic protocol. On receiving information from epidemic protocol, a subscriber updates its estimate of event

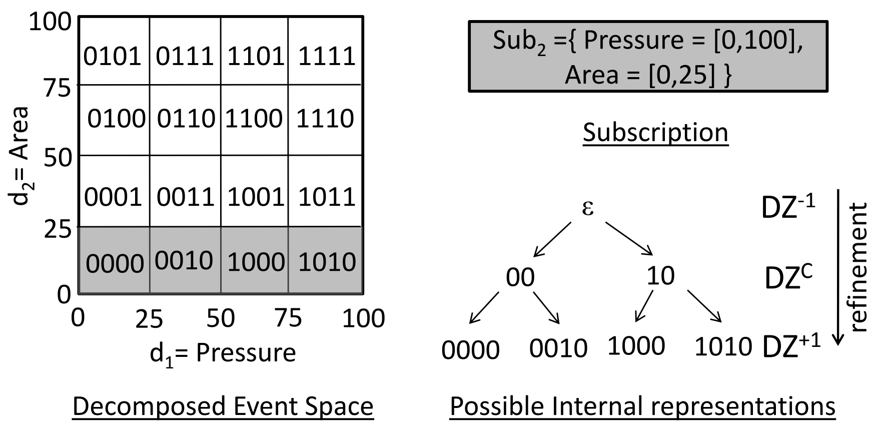


Figure 2.6: Subscriber-defined Accuracy.

rates in the coarser sub-spaces, i.e., DZ^{-1}, along with the adaptation of *pView* and *pubList*, as mentioned in Section 2.4.2.2. Statistical properties provided by epidemic protocols [JVG+07, EGH+03, Kol04] ensure that a peer will eventually learn the event rate in a coarser sub-space if there exists subscribers (in the system) for that sub-space or parts of it. Nevertheless, event rate in a sub-space which is currently not subscribed (in parts or as a whole) by any subscriber in the system can only be measured by actually subscribing to that sub-space.

Figure 2.6 shows the possible mappings from a user-level subscription. If the subscription is currently mapped to $DZ^C = \{00, 10\}$ then the subscriber keeps track of the event rates in the sub-spaces $DZ^{-1} = \{\epsilon\}$ and $DZ^{+1} = \{0000,\ 0010,\ 1000,\ 1010\}$. If there is a high rate of false positives in a sub-space of the current peer-level subscription, the subscriber will drop it and select the relevant of the finer sub-spaces from DZ^{+1} instead. Similarly, the subscriber can select one sub-space from DZ^{-1} instead of multiple previous enclosed sub-spaces and receive additional false positives.

Special rules apply to new subscribers that do not know about event rates yet and cannot decide about the permissible degree of coarseness of their mapping. A new subscriber initially selects a parameter θ that may for example represent the number of incoming connections that the subscriber can maintain. Then it approximates the accurate subscription DZ^A by at most θ dz-expressions in DZ^C. This initial coarse mapping has the following properties:

1. $|DZ^C| \leq \theta$
2. $\forall dz_a \in DZ^A\ \exists dz_c \in DZ^C : dz_a \prec dz_c$

A coarse transformation of a user-level subscription into θ dz-expressions using breadth-first exploration of the binary tree of dz-expressions (cf. Figure 2.2) is given in Algorithm 4. The algorithm generates less than θ dz-expressions if the terminal dz-expressions generated by the splitting of a sub-space into two smaller sub-spaces can

Algorithm 4 Generation of θ *dz*-expressions

```
Require: User-level subscription $f$
Ensure: Generation of at most $\theta$ dz-expressions
 1: $DZ^C \leftarrow \emptyset$ // Final holder of dz-expressions
 2: $L^d$ $d$-dimensional vector with lower bounds for each dimension
 3: $U^d$ $d$-dimensional vector with upper bounds for each dimension
 4: Queue q $\leftarrow \{dz = \epsilon, L^d, U^d\}$
 5: while $!q.empty()$ do
 6:     subSpace $\leftarrow$ q.dequeue()
 7:     i $\leftarrow |subSpace.dz|\ mod\ d$ // Next dimension to decompose/split.
 8:     x $\leftarrow \frac{subSpace.L_i + subSpace.U_i}{2}$ // mid-point of the subSpace along the dimension $i$.
 9:     if $\forall_j (f.L_j \leq subSpace.L_j \wedge f.U_j \geq subSpace.U_j)$ then
10:         $DZ^C \leftarrow subSpace.dz$ // completely covered
11:     else if $(sub.L_j < x \wedge sub.U_j > x) \wedge (|q| + |DZ^C| < \theta)$ then // $f$ occupies both sides of midpoint.
12:         q $\leftarrow \{subSpace.dz \| 0, L^d, U^d(U_i = x)\}$
13:         q $\leftarrow \{subSpace.dz \| 1, L^d(L_i = x), U^d\}$
14:     else if $f.L_j < x \wedge f.U_j < x$ then
15:         q $\leftarrow \{subSpace.dz \| 0, L^d, U^d(U_i = x)\}$
16:     else if $f.L_j > x \wedge f.U_j > x$ then
17:         q $\leftarrow \{subSpace.dz \| 1, L^d(L_i = x), U^d\}$
18:     else
19:         $DZ^C \leftarrow subSpace.dz$
20: Merge terminal dz-expressions
```

be merged to a single dz-expression without affecting the accuracy of the subscription approximation.

2.6 Optimized Spatial Indexing

For an event space with a large set of attributes, the number of dz-expressions for an accurate subscription representation can be very large. As described in Section 2.5.1, a coarse subscription mapping reduces the number of dz-expressions. However, it induces false positives and hence its applicability depends on the bandwidth constraints of the subscriber.

A simple modification in the representation of dz-expressions can reduce their number without changing their accuracy. A *dz-expression* is redefined to include the wild-card $*$ which stands for "0 and 1". Two dz-expressions that differ in only one place can be combined by replacing this place by "$*$". For example, the subscription in Figure 2.6 can be represented by one dz-expression "$*0*0$".

Dz-expressions of that form are created by a modified spatial indexing mechanism. The decomposition procedure works mainly as before. Only if the subscription covers the complete domain of the dimension to be divided, then instead of creating two dz-expression for the smaller sub-spaces (ending with 0 and 1), $*$ is added to the dz-expression.

The containment relationship defined on dz-expressions as well as the subscription mapping and bandwidth estimation mechanisms work with the modified technique. Furthermore, the modification allows subscribers to define constraints only on a subset of attributes in the event space.

2.7 Dealing with Real Delay Values

So far, we assumed that subscribers are arranged in the overlay network according to their delay requirements, i.e., a peer p connects to a parent q only if $\Delta(q) \leq \Delta(p)$. However, connecting subscribers by their delay requirements may prevent the satisfaction of their subscriptions in some scenarios. Consider an example with two subscribers, s with $\Delta(s) = 5$ and q with $\Delta(q) = 6$, and a publisher p. The actual link delays between them are $d^O(q, s) = d^O(s, q) = 2$, $d^O(p, q) = 3$ and $d^O(p, s) = 5$. The delay constraints of q cannot be satisfied if the subscribers are connected according to their delay requirements: $p \xrightarrow{5} s \xrightarrow{2} q$. However, placing q closer to the publisher results in the satisfaction of both subscribers: $p \xrightarrow{3} q \xrightarrow{2} s$.

In general, the maximum experienced delay $D(q) = \max \{D(p, q) \mid p \in P_q\}$ of a subscriber q to its relevant publishers P_q can be lower than its delay constraint $\Delta(q)$. Another subscriber s can exploit this and connect to q if $D(q) + d^O(q, s) < \Delta(s)$ even if $\Delta(q) \geq \Delta(s)$. The presented system can be easily extended to consider the experienced maximum delay while deciding the child connections (cf. Algorithm 1).

The presented improvement, however, may lead to a violation of $\Delta(s)$ if $D(q)$ increases. Such violations can be easily detected during the dissemination of events (cf. Section 2.4.4). More precisely, end-to-end delay experience by an event e is updated on each overlay hop during its dissemination (cf. Algorithm 1, *line 5*). A subscriber s on receiving event e from its parents q increments the end-to-end delay $D(e)$ of the event by adding the overlay link delay between q and s, i.e., $D(e) = D(e) + d^O(q, s)$. If the delay experienced by the received event violates $\Delta(s)$, subscriber s disconnects from its parent.

2.8 Performance Evaluation

In this section, we evaluate the performance of the presented algorithms according to the following criteria: i) convergence to subscription and delay constraint satisfaction, ii) control overhead, iii) adaptability to dynamic conditions, iv) scalability in terms of number of peers and attributes, and v) effect of bandwidth consumption and out-degree constraints on the satisfaction of subscribers.

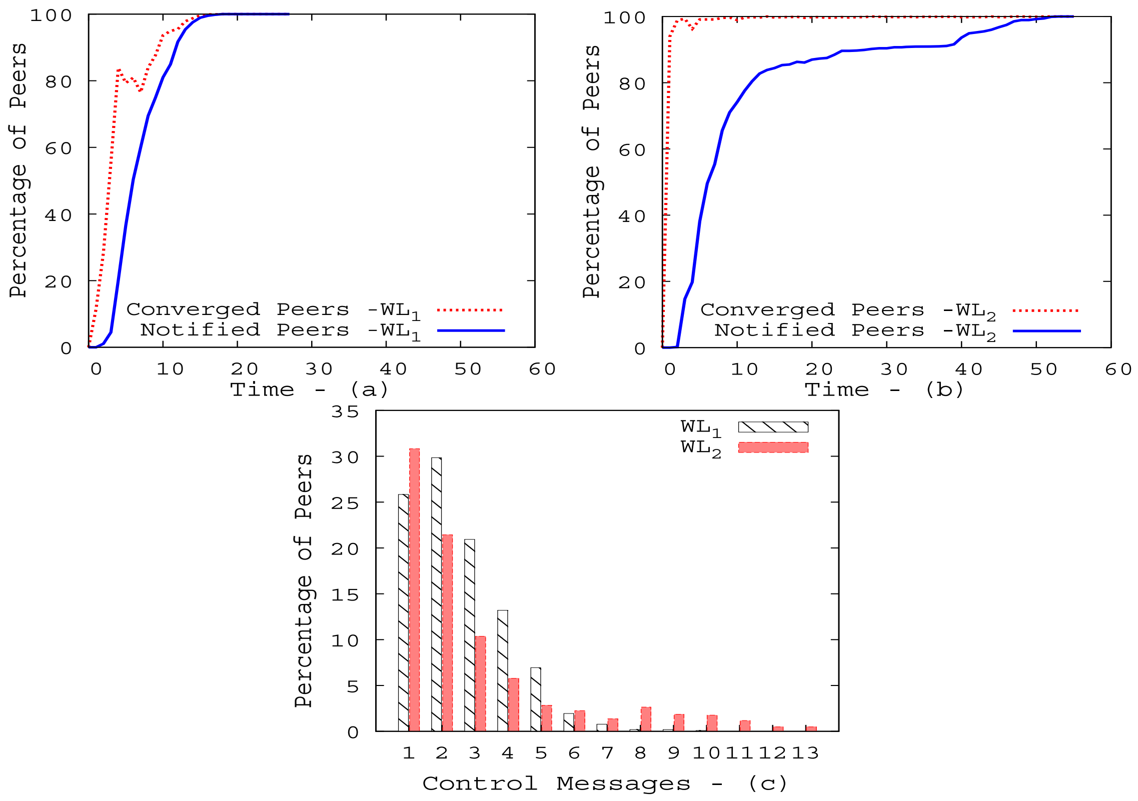


Figure 2.7: Convergence to subscription and delay constraint satisfaction.

2.8.1 Experimental Setup

Simulations are performed using PeerSim [JMJV], a large-scale P2P discrete event simulator. Each peer relies on *Gossip-based peer sampling service* [JVG+07] to maintain its partial view (*pView*) of 5% other peers in the system. Simulations are performed for up to $N = 7000$ peers. Unless otherwise stated, out-degree constraints of the peers are chosen as $m = log_2(N)$. The event space has up to 10 different attributes. The data type of each attribute is Integer[†], and the domain of each attribute is the range $[1, 128]$. We evaluated the system performance under uniform (WL$_1$) and skewed (WL$_2$) subscription workloads; and with skewed and uniform event distributions. Skew is simulated using the widely used 80%-20% Zipfian distribution with $4 - 6$ hot spots.

We use the following performance metrics in our evaluations:

1) Percentage of converged peers: The fraction of peers out of the total population which have found a suitable set of parents that cover their subscription.

[†]Spatial indexing can work with any ordered data type with a known domain [MJ07]. Evaluation results are not sensitive to the choice of data type and therefore, similar to [MJ07] only integer data types are considered.

2) Percentage of notified peers: The fraction of peers which are receiving events from all the relevant publishers without violating their delay constraints.

3) Construction time: Logical time needed to complete the construction of the overlay topology.

2.8.2 Convergence

In this experiment, moderate delay requirements are assigned to the peers such that convergence can be achieved. Figures 2.7(a)-(b) show the construction time for the overlay topology. For all of the workloads, the percentage of notified peers is always less than that of converged peers until 100% convergence is achieved. The reason is that the peers opportunistically connect to other peers in order to cover their subscriptions and satisfy their delay constraints. Therefore, during the evolution of the overlay topology, many separate isolated groups of peers may exist. Some of these groups may not have found a connection to the relevant publishers. Eventually, all the groups converge to one overlay topology.

The overlay construction time for workload WL_2 is higher due to the fact that there is very little overlap between the subscriptions of peers assigned to different hotspots. This results in subscribers with coarser subscriptions occupying all the places near the publishers, forwarding events that only correspond to a portion of the event space. Therefore, the subscribers with finer subscriptions (to uncovered portions of the event space) have to increase their subscription to compete with subscribers with coarser subscriptions.

Figure 2.7(c) shows the control overhead incurred by the peers in order to find suitable parents. It shows the percentage of the affected peers as a function of the number of connection request messages sent by them. It is clear from the figure that for WL_2 workload higher percentage of peers directly connects to the appropriate parent. This is due to high similarity between the subscriptions that belong to the same hotspot. However, some peers incur more control overhead to find suitable parents (i.e., spread of control overhead distribution is wider in case of WL_2) because of very little overlap between the subscriptions of peers assigned to different hotspots as discussed above.

2.8.3 Adaptability to Dynamic Conditions

First, we study the dynamic resilience of the system in the presence of continuously joining and leaving subscribers. The percentage of churn is relative to the total of all peers in the system. For instance, for a total number of 1000 peers, a churn of 2.5% means that in each time step, 25 online peers leave and the same number of new peers with different subscriptions and delay requirements join the system. Figures 2.8(a)-(b) show the percentage of converged and notified peers for different percentages of churn

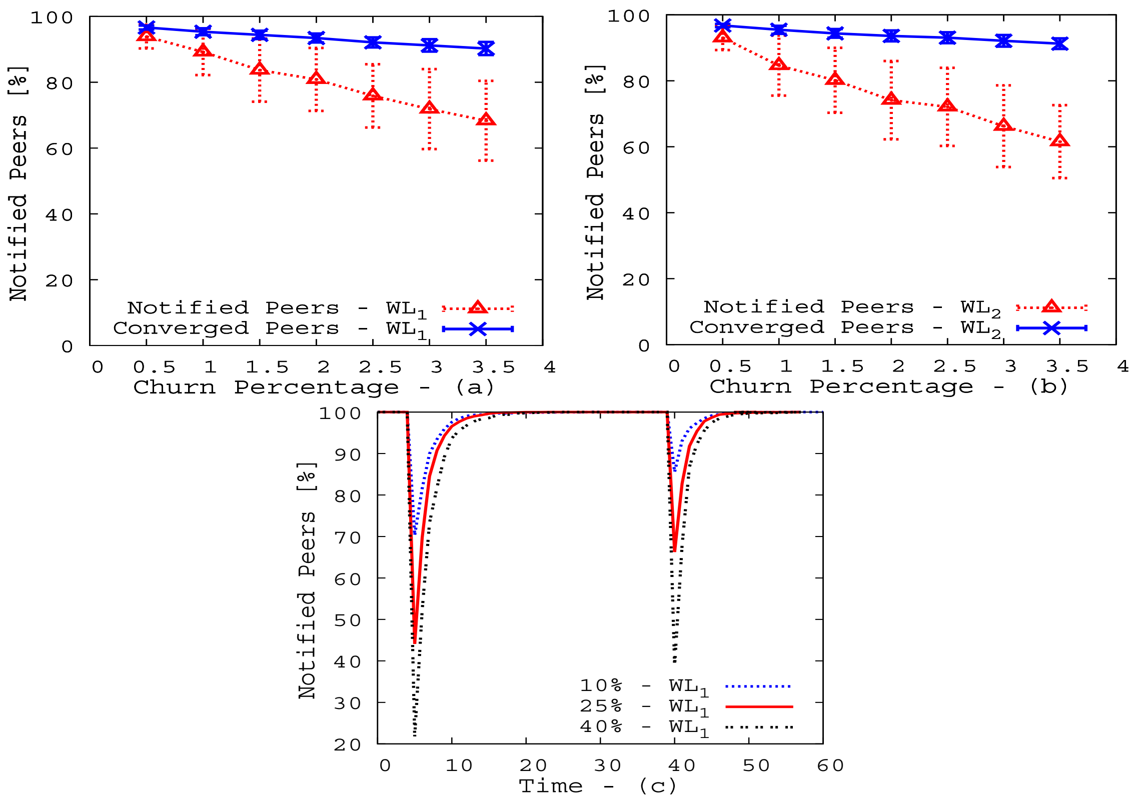


Figure 2.8: Adaptability to continuous and rapid churn.

along with the standard deviation. The reason for the gradual degradation in the percentage of notified and converged peers is the fact that a high churn rate increases the probability that peers placed near the publishers leave the system, affecting the delay constraint satisfaction of all their descendant subscribers. The lower percentage of notified peers in WL_2 (in comparison to WL_1) is due to the fact that the subscription distribution is skewed and some subscribers may need to increase their subscriptions as discussed in the convergence evaluations.

Next, we evaluate the dynamic resilience of the system to sudden massive churn. Once the system converged to a stable state, massive churn is introduced in the system, i.e., 10%, 25% and 40% of the online peers leave and an equal number of peers join the system. Figures 2.8(c) shows that the system can tolerate and recover from massive occurrences of churn. The reason for sudden degradation in the percentage of notified peers is the same as in the case of continuous churn. A large percentage of churn increases the probability that peers with tight delay constraints and coarser subscriptions, and thus are ancestor to many other peers, may leave.

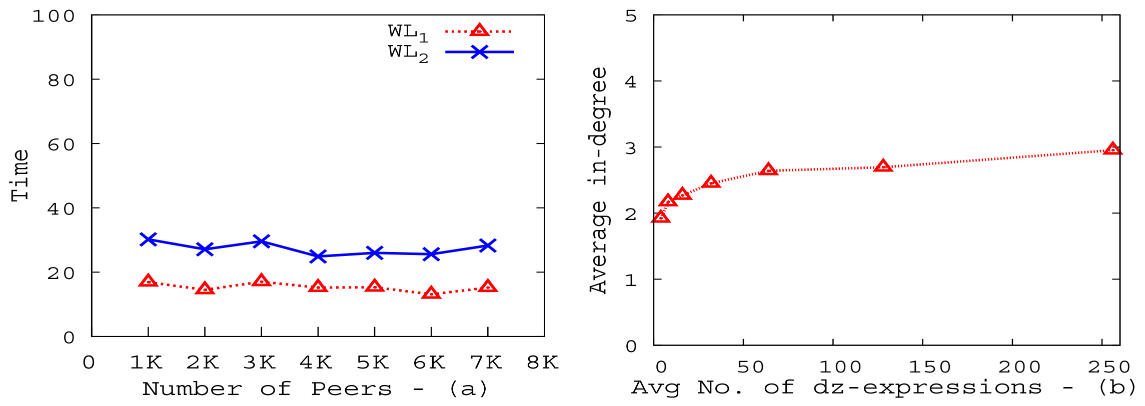


Figure 2.9: Scalability w.r.t. no. of peers and attributes in the system.

2.8.4 Scalability

First, we study the scalability with respect to the number of peers in the system. In all the experiments the out-degree constraints are chosen as $log_2(N)$ of the total number of peers N. Figure 2.9(a) shows that up to 7000 peers the overlay construction time almost stays the same. Furthermore, the overlay construction time for WL_2 is in general higher because some subscribers may need to increase their subscriptions as discussed in the convergence evaluations.

Next, we study the effect of the number of attributes in the event space on the system's scalability. The number of dz-expressions needed for the accurate representation of a user-level subscription generally increases with the number of attributes. A peer maintains a suitable parent for each of its dz-expressions. Therefore, we study the effect of an average increase in the number of dz-expressions on the average in-degree for WL_1 as shown in Figure 2.9(b). The averages are taken over all the peers in the system. The results show a slight increase in the average in-degree with the number of dz-expressions, i.e., increasing the average number of dz-expressions from 4 to 256 increases the average in-degree by just 1.2 to 3.1.

2.8.5 Effect of Bandwidth and Out-degree Constraints

In this experiment, we study the impact of bandwidth and out-degree constraints on the satisfaction of delay requirements of subscribers. Two scenarios are evaluated: one where the subscribers are assigned moderate delay requirements (S_1) and the other with tight delay requirements (S_2). In both the scenarios, delay requirements of all the subscribers cannot be satisfied without inducing false positives. All the subscribers are assigned the same bandwidth constraints, specified in terms of allowed false positives as a percentage of the overall event rate. For example, 3.1% of allowed false positives mean that subscribers can increase their subscription till they are receiving 3.1% of

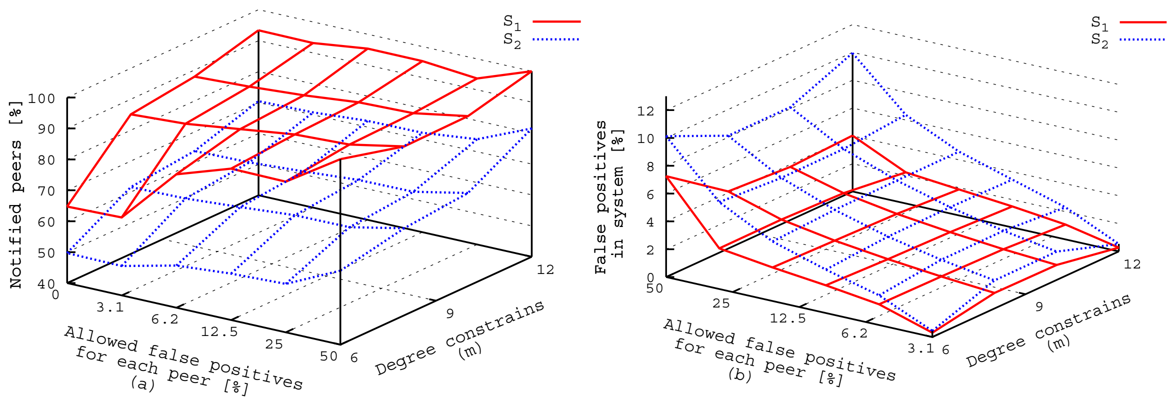


Figure 2.10: Satisfaction of delay requirements in the presence of constraints.

overall events in the system as false positives.

Figure 2.10(a) shows the percentage of notified peers for different percentages of allowed false positives and out-degree constraints. Figure 2.10(b) depicts the actual percentage of false positives in the system for the scenarios S_1 and S_2. The figures show that it is of advantage to increase the rate of false positives since it raises the overall percentage of satisfied subscribers. In case of S_1 with $m = 6$, only 65% of subscribers are notified in the absence of false positives. However, allowing peers to receive up to 6.2% of overall events as false positives increases the percentage of notified peers by 27.6% to 83% with only 1.2% increase in the overall rate of false positives in the system. Moreover, the figures show that the out-degree constraints have more profound impact on the satisfaction of subscribers especially in the presence of tight delay requirements. For example, in scenario S_2 with $m = 6$, even when the subscribers are allowed to increase their false positives up to 50% of overall event rate, only 64% peers are notified, mainly due to the lack of available places to connect (in the overlay network). However, increasing the value of m to 12, raises the percentage of notified peers by 28% to 82%. Figure 2.10(b) also shows that the overall rate of false positives in the system is higher for the scenario S_2. The reason is that the delay requirements of subscribers in S_2 are very tight and that it is not possible to satisfy all of them. In this case, the unsatisfied subscribers coarsen their subscriptions to get a better place in the overlay. However, as all the subscribers have similar bandwidth constraints and there are limited places to satisfy delay requirements, coarsening subscriptions does not give them any competitive advantage. It just raises the overall rate of false positives.

2.9 Related work

Related work to the contributions in this chapter can be classified into the following two areas.

2.9.1 P2P-based Publish/Subscribe Systems

A number of content-based publish/subscribe systems without broker infrastructure has been proposed in the past. The efficiency of event routing in such systems is very sensitive to the organization of subscriber and publisher peers in an overlay network. Typically, two approaches are used for organizing peers. The first approach uses *distributed hash table* (DHT) based overlays, such as Chord [SMLN+03], CAN [RFH+01], Pastry [RD01] etc., to arrange peers. The content-based matching and filtering is implemented as a separate layer on top of DHTs [PB02, TBF+03, MJ07, GSAA04, AT06, YH07]. The efficiency of this approach is restricted as the overlay network is oblivious to the dynamics in the upper content-based layer [JPMH07].

The second approach organizes peers in semantic communities according to the similarities between their subscriptions, i.e., subscribers matching similar events are placed in close proximity in an overlay network [AGD+06, VRKS06, CF05, APBSV10, BFG07, PRGK09]. Such a semantic organization of peers is particularly good in reducing false positives and therefore many recent systems have adopted this approach.

Sub-2-Sub [VRKS06] arranges subscribers with intersecting subscriptions into rings. The events are only disseminated in relevant rings, thus completely avoiding false positives. The number of rings depends on run-time interactions between the subscribers that are active in the system and are impossible to limit. Thus, even for a moderate number of subscribers, the number of rings may quickly grow to a very large number, limiting the scalability of the system [Que08].

DR-tree [BFPB10, APBSV10] extends R-tree [Gut84] to support content-based routing in a distributed fashion. DR-tree inherits beneficial properties of R-tree such as a low rate of false positives and a logarithmic event dissemination time. However, the system induces higher load on the peers close to the root of the DR-tree: Peers closer to the root experience more false positives without any possibility to change their rate dynamically. Additionally, to reduce the overall rate of false positives, subscribers are always inserted from the root.

In the work of Chand et al. [CF05], containment relations between the subscriptions of subscribers are directly mapped to a tree structure. This results in as many trees as there are subscriptions that are not contained in any other subscription [BDFG07]. To avoid this problem, the authors propose a proximity metric to arrange subscribers in a tree (instead of using the containment relations). However, event dissemination on the tree constructed using proximity metric results in false negatives, i.e., subscribers may not receive events matching their subscriptions.

Similarly, DPS [ADG+04] builds a separate tree for each attribute of the event space, and a subscriber can join any tree for which it has specified an attribute filter (in its subscription). A publisher publishes an event on all the trees associated with the attributes in the event. This results in a large number of unnecessary messages on each tree, especially the undesired messages grow with the increase in number of attributes in the event space.

Apart from the stated drawbacks, existing systems only focus on the overall reduction of false positives without taking into account the heterogeneity of subscribers in terms of QoS requirements to better utilize resources in a publish/subscribe system. In an environment, where the capabilities of subscribers vary widely, it is important that individual subscribers have the flexibility to locally adjust the accuracy of received event messages according to their constraints.

2.9.2 Provision of QoS Requirements

Behnel et al. [BFM06] define a common meaning of different QoS metrics in the context of publish/subscribe. Although it is easy to add QoS semantics into subscriptions (IndiQos [CAR05]), only few systems actually cope with satisfying QoS requirements in a loosely coupled environment of publish/subscribe [MKSB07].

Wang et al. [WZC+12] focus on providing QoS support on a single publish/subscribe broker running on a commodity hardware with multi-core processors. In particular, they propose intra-core scheduling and inter-core message dispatching mechanisms to perform event matching in an efficient manner and thereby to avoid the violation of delay requirements. However, satisfaction of delay requirements in the presence of a network of brokers (i.e., distributed publish/subscribe system) is not addressed. A similar work by Wang et al. [WCLW06] targets to achieve bounded delays on event delivery in a broker network, by employing message scheduling strategies at each broker. However, the authors assume a static broker topology and the system is unable to provide QoS bounds in the presence of dynamic conditions.

IndiQoS [CAR05] addresses satisfaction of individual delay and bandwidth requirements of subscribers, but only provides limited form of event routing termed as type-based routing. Moreover, IndiQoS uses network-level QoS architectures such as *Integrated services* [BCS94] and *Differentiated services* [BBC+98] to reserve resources along the communication links and guarantee end-to-end QoS. Resource reservation protocols are not available on a global scale, which limits the applicability of the IndiQoS system to heterogeneous network environments.

Some of the problems stated above are addressed by the work of Tariq et al. [TKKR09]. Instead of relying on reservation protocols, it provides probabilistic delay bounds. Moreover, the expressiveness of content-based routing is preserved. The overall approach is to cluster subscribers into groups according to the available publishers in the system. Within each group, subscription and advertisement forwarding is then used to maintain probabilistic delay bounds. However, the clustering strategy is only sketched.

Systems from the domain of *Application Layer Multicast* (ALM) also address satisfaction of QoS requirements [PWF07]. For instance, LagOver [DSF07] addresses individual delay requirements of peers in the presence of out-degree constraints. It constructs a single-source multicast tree, where each peer in the tree is interested in the same information and is placed according to its desired delay. This is not sufficient to handle the complexity introduced by content-based publish/subscribe.

Finally, some existing publish/subscribe systems address QoS as a system properly rather than requirements from the individual subscribers. For instance, Majumder et al. [MSRS09] address construction of low stretch spanning tree to route events in a broker network. Discussion about such systems will be presented in Section 4.7.

2.10 Summary

In this chapter, we have shown how the individual delay requirements of a large dynamic set of subscribers in a content-based publish/subscribe system can be satisfied without violating their bandwidth constraints. In particular, subscribers are given the flexibility to define their permissible rate of false positives according to their individual bandwidth constraints. Additionally, we propose a subscriber-driven decentralized algorithm to connect publishers and subscribers in an overlay network according to their delay requirements so that subscribers with tight delay requirements are located closer to the relevant publishers. The evaluation shows that the proposed algorithm converges to the satisfaction of subscriber-specific delay constraints even in a very dynamic setting.

Chapter 3

Subscription Clustering

3.1 Introduction

Rather than accounting for individual QoS requirements and resource constraints (cf. Chapter 2), the contributions in this chapter target scenarios where the main emphasis is to minimize the overall resource consumption in a publish/subscribe system. For instance, Green IT initiative by many enterprises focuses on reducing the operational cost of the IT infrastructures and lowering the carbon footprint [CJ11, Ent09]. Publish/Subscribe is increasingly being used as a communication paradigm in many of these enterprises. For example, Google uses GooPS – a topic-based publish/subscribe system – to exchange information across Google services and applications residing across different data centres [LHLJ12, Taj11]. Microsoft employs WSP (Web Solutions Platform) event system – a content-based publish/subscribe – to provide very fast communication infrastructure across $8,000$ servers in Microsoft Bing [Ham12]. Similarly, Yahoo developed YMB (Yahoo Message Broker) – a topic-based publish/subscribe system – to manage replication in a massive geographically distributed database of its web applications [Taj11, CRS$^+$08]. Therefore, in-line with the Green IT initiative, minimizing resource consumption in publish/subscribe systems is an interesting research problem, as recently pointed out by Cheung et al. [CJ11].

False positives serve as one of the major contributors to the resource consumption in publish/subscribe systems, as they increase network bandwidth utilization and induce extra processing load on the peers in an overlay network [Que08]. However, false positives cannot be avoided completely without sacrificing the scalability of a publish/subscribe system [CS04]. The interest of subscribers can be highly diverse and in the worst case generate $2^N - 1$ distinct subscriber groups out of N subscribers.

Subscription clustering is one promising way to reduce the effect of false positives by grouping subscribers with similar subscriptions in a limited number of clusters such that the event dissemination within each cluster can be very efficient w.r.t. the number of false positives [CM07, PC05, MSRS09].

Subscription clustering in a publish/subscribe system is complicated due to the lack of global knowledge, high dynamicity of subscribers and continuously evolving event traffic. Existing approaches to clustering mostly consider the structural (absolute) similarity between the subscriptions such as the area occupied by the intersection of two subscriptions [BFPB10, APBSV10, TKK+10, VRKS06, CM07]. These approaches restrict the expressiveness of the content-based model to predefined numeric attributes and lack mechanisms to further reduce false positives by considering the similarities between subscriptions according to the current event traffic. Only a handful of approaches take into account the current event load of the system to create clusters of subscribers. These approaches either assume global knowledge [PC05, MSRS09, CJ11] or rely on the presence of a network of reliable brokers [BBQV07].

This chapter presents an efficient and scalable approach to cluster management coping with high workload variations [TKKR12]. We study its effectiveness in the context of broker-less publish/subscribe systems. The approach preserves the expressiveness of the content-based model and can work with any method to estimate similarities between subscriptions. Techniques from spectral graph theory are used to perform subscription clustering, because these techniques are proved to be more accurate in finding clusters than traditional mechanisms such as k-means [Lux07]. This work is the first to study spectral clustering in the context of content-based publish/subscribe systems. Furthermore, the presented approach can be seen as a general framework to perform distributed spectral clustering and can be applied to other areas, such as document clustering, image segmentation or data mining etc. Spectral clustering in a distributed setting is highly challenging and to the best of our knowledge has not been previously addressed in literature.

3.2 System Model and Problem Formulation

Similar to Section 2.2, we consider a content-based publish/subscribe system without broker infrastructure. Publishers and subscribers contribute as peers to the maintenance of the system. Publishers disseminate events into the system, while the subscribers specify filtering criteria for the selection of desired events using (possibly many) *subscriptions*.

To avoid dependency on any particular subscription language and to preserve expressiveness, we consider a generic content-based model [Tar08]. A subscription f is a stateless boolean function that accepts an event e as an argument. An event e matches a subscription f if $f(e) = true$. A *containment relation* can be defined on subscriptions. Let E_{f_1} and E_{f_2} denote the set of events matching the subscriptions f_1 and f_2 respectively. Then f_1 is said to be *covered* by the subscription f_2, denoted by $f_1 \prec f_2$,

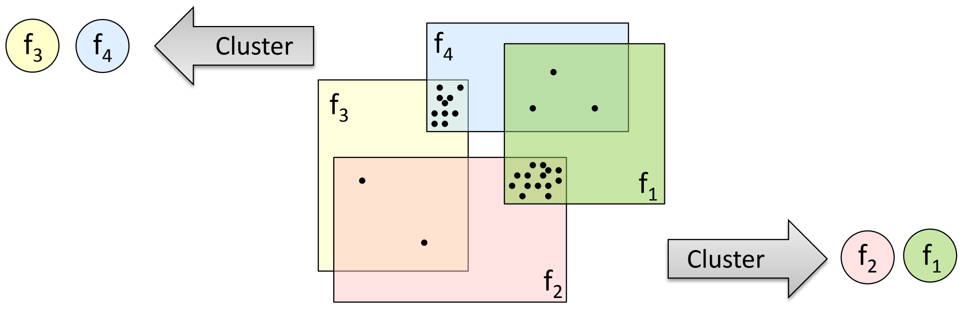


Figure 3.1: Clustering according to the current event load of the system.

iff $E_{f_1} \subseteq E_{f_2}$ holds. Similarly, two subscriptions f_1 and f_2 are said to be overlapping, denoted by $f_1 \simeq f_2$, iff $E_{f_1} \cap E_{f_2} \neq \emptyset$.

In a broker-less publish/subscribe system as described above, the main concern is to reduce the cost of event dissemination by avoiding *false positives* (events a peer is required to forward while they do not match any of the peer's subscriptions). Subscription clustering is an effective approach for this purpose. It partitions the subscriber peers with similar subscriptions into a small number of groups such that the event dissemination within each group incurs few false positives.

To perform clustering, usually the absolute (structural) similarity between the subscriptions, such as the overlap relation ($\simeq$) defined on E_f, is considered to calculate their closeness in receiving the same events. However, the overlap relation does not take into account the number of events recently matched by the subscriptions for calculating the similarities. For instance, a low event rate causes little false positives in a cluster of two similar, yet not identical subscriptions, while with a high event rate, the same cluster encounters a large number of false positives, so that the two subscriptions had better not be clustered. In Figure 3.1, subscriptions f_1 and f_4 should be placed in the same cluster according to the absolute (structural) similarity between them, however, considering the current event load of the system placing f_1 and f_2 in a cluster is more beneficial in reducing the false positives. Therefore, to achieve good clustering that minimizes false positives, the current event load of the system should be considered.

In more detail, let $\overline{E}^t_f$ be the set of last ψ events matched by the subscription f before a given time t. Let $\text{sim}(i, j, t)$ be a function that defines the similarity between two arbitrary subscriptions f_i and f_j at time t using their recently matched events, i.e., $\overline{E}^t_{f_i}$ and $\overline{E}^t_{f_j}$.

The set of N subscriptions in the system $\Pi = \{f_1, ..., f_N\}$ and the pairwise similarities between them can be represented by a *similarity graph.* A similarity graph denoted by $G^S = (\Pi, \mathcal{E}^S, t)$ is a weighted undirected graph, where $\mathcal{E}^S$ is the set of edges connecting distinct subscriptions with non-zero similarity values, i.e., $\mathcal{E}^S = \{\varepsilon^s_{i,j} : f_i \neq f_j \wedge \text{sim}(i, j, t) > 0\}$. The similarities between subscriptions and hence the similarity graph change over time, however, for the clearness in presentation we will omit time t in the following. Using the adjacency matrix of the similarity graph, which is a symmetric matrix we denote by $\mathfrak{W} \in \mathbb{R}^{N \times N}$, we can define the *weighted degree* $\mathfrak{d}$ of a subscription f_i as the sum of its pairwise similarities with all other subscriptions in the system, i.e., $\mathfrak{d}_i = \sum_{j=1}^{N} \mathfrak{w}_{i,j}$. The degree matrix denoted by $\mathfrak{D} \in \mathbb{R}^{N \times N}$ is a diagonal matrix with degrees $(\mathfrak{d}_1, ...\mathfrak{d}_N)$ of all subscriptions on the diagonal. Let Π_1, Π_2 denote two groups of vertices in G^S. The inter-group weight or the sum of edge weights between the vertices of Π_1 and Π_2 can be defined as $\mathfrak{W}(\Pi_1, \Pi_2) = \sum_{i \in \Pi_1, j \in \Pi_2} \mathfrak{w}_{i,j}$. Furthermore, the degree of a group Π_1 is defined as $\mathfrak{d}_{\Pi_1} = \mathfrak{W}(\Pi_1, \Pi)$.

Given a dynamic set of subscribers and continuously evolving similarity graph, our objective is to maintain $k_{\mathcal{S}}$ disjoint clusters of subscriptions $(\Pi_1, ...\Pi_{k_{\mathcal{S}}})$ in a publish/subscribe system so that,

1. inter-cluster weights are minimized, i.e., intersection (or commonality) between the events in different clusters is minimized.
2. clusters are balanced in terms of their size to ensure even event dissemination load on each cluster.

In the presence of only first criterion (i.e., minimization of inter-cluster weights), the subscription clusters can be efficiently created by solving the minimum cut problem [Lux07]. However, in practice minimum cut does not lead to satisfactory results because of its tendency to create singleton clusters, i.e., clusters consisting of only individual subscriptions. The second criterion is added to overcome this problem and create clusters with sufficiently large number of subscriptions.

3.3 Centralized Subscription Clustering

In this section, we describe our approach to maintain the subscription clusters in a centralized fashion. Similar to other state of the art centralized approaches [RLW+02, PC05, MSRS09], the proposed centralized approach has potential to be used as a competent method to perform subscription clustering in a content-based publish/subscribe system. In particular, our contributions in this section are twofold. First, we select different spectral methods (from the literature) that can effectively solve the subscription clustering problem. Second, we show the capability of the selected methods to produce clusters with improved quality in comparison to the related approaches.

3.3.1 Subscription Clustering using Spectral Methods

In the following, we describe the spectral methods and main steps needed to perform subscription clustering.

3.3.1.1 Similarity Function

In order to perform clustering, the first step is to quantify the similarities between the subscriptions by defining an appropriate similarity function sim(i, j). Intuitively, the similarity between the two subscriptions increases with the increase in overlapping events and decreases with the non-overlapping event traffic. Based on this intuition, we propose to use the Jaccard* (in short Jac) similarity function [CC00], which is defined as a ratio of overlapping event sets matched by the subscriptions to the union of their overlapping and non-overlapping event sets, i.e., Jac $= \frac{|\overline{E}^t_{f_i} \cap \overline{E}^t_{f_j}|}{|\overline{E}^t_{f_i} \cup \overline{E}^t_{f_j}|}$. Similarities are assigned in the range $[0, 1]$, where 0 means complete disjointness.

3.3.1.2 Clustering as Graph Partitioning

Once the similarity function is in place and the similarity graph is calculated, the subscription clustering is performed by partitioning the similarity graph into $k_{\mathcal{S}}$ disjoint sub-graphs. We select two graph partitioning objective functions which fulfil our requirements of strongly connected and balanced clusters:

$$\text{Ratio Association} = \max_{\Pi_1, \ldots \Pi_{k_{\mathcal{S}}}} \sum_{i=1}^{k_{\mathcal{S}}} \frac{\mathfrak{W}(\Pi_i, \Pi_i)}{|\Pi_i|} \tag{3.1}$$

$$\text{Normalized Cut} = \min_{\Pi_1, \ldots \Pi_{k_{\mathcal{S}}}} \sum_{i=1}^{k_{\mathcal{S}}} \frac{\mathfrak{W}(\Pi_i, \Pi \setminus \Pi_i)}{\mathfrak{d}_{\Pi_i}} \tag{3.2}$$

The *Ratio Association* (in short RAssoc) objective [DGK07] aims to maximize the association between the cluster members (intra-cluster), whereas the *Normalized Cut* [DGK07] (in short NCut) tries to minimize the association between different clusters (inter-cluster). Both objective functions strive to create clusters that are balanced in terms of the number of vertices and the degree (edge weights) respectively. Both objective functions are NP-hard to solve in the discrete domain. The relaxed versions of these objective functions can be solved by *spectral analysis* of the similarity graph.

3.3.1.3 Spectral Analysis

The spectral clustering consists of two main steps. In the first step, discrete graph partitioning problem is relaxed to obtain the global optimum in the continuous domain

*We have evaluated many different similarity functions such as Russell [CC00], Dice [CC00], Simpson [CC00] and Simple matching [CC00]. However, the results of the Jaccard function were superior in consistency. Therefore, in this chapter we only focus on the properties of this function.

Algorithm 5 Centralized spectral clustering steps

```
1: Calculate the similarities between $N$ subscriptions
       Generate similarity graph $G^S$
       Apply Gaussian function on $G^S$ (NCut)
           $\varepsilon^s_{i,j} = \exp(\frac{-\|\mathrm{sim}(i,j)\|}{\sigma^2})$
2: Perform eigenvector decomposition
       $Y = \Lambda_{k_\mathcal{S}+}\mathfrak{V}^T_{k_\mathcal{S}+} : \overline{\mathfrak{W}} = \mathfrak{U}\Lambda\mathfrak{V}^T$ (RAssoc)
       $Y = \mathscr{Q}^T_{k_\mathcal{S}-} : (\mathfrak{D} - \mathfrak{W})\mathfrak{q} = \lambda\mathfrak{D}\mathfrak{q}$ ( NCut)
3: Apply k-means clustering algorithm
       Cluster $(Y_i)_{i=1,..N} \in Y$ into $k_\mathcal{S}$ clusters
```

by solving a trace optimization problem [KCS11, Lux07]. The solution to trace optimization problem leads to the eigen decomposition [Jol86, CC00] of the similarity graph (or some variant of the similarity graph as described later). The eigen decomposition assigns an eigenvector to each vertex of the similarity graph (i.e., subscriptions). As these eigenvectors take arbitrary values in $\mathbb{R}$, post processing (of the eigenvectors) is required to obtain discrete partitions (or clusters). Therefore, in the second step a k-means algorithm [KMN+02, HK07, DGK09] is used to get discrete partitions from the real value eigenvectors.

Intuitively, the first step (eigen decomposition) embeds the vertices of a similarity graph (i.e., the subscriptions) in a low-dimensional space such that the dimensionality of the low-dimensional space is equal to the number of desired clusters, i.e., $k_\mathcal{S}$. The geometrical coordinates of each vertex (of the similarity graph) in the low-dimensional embedded space is provided by the eigenvector assigned to it. This step (i.e., first step of spectral clustering) is termed as *dimensionality reduction* or *embedding* and is beneficial in reducing the effect of noise and extracting the main features of the data to cluster. This step depends on the partitioning objective function.

The Ratio Association function performs linear dimensionality reduction using *Principal Component Analysis* (PCA) [Jol86, Hot33]. PCA performs dimensionality reduction by means of *singular value decomposition* (SVD) [EY39, KL80], i.e., $\overline{\mathfrak{W}} = \mathfrak{U}\Lambda\mathfrak{V}^T$, where $\overline{\mathfrak{W}}$ is a centred matrix obtained by subtracting the mean of the similarity matrix $\mathfrak{W}$ from its columns, $\Lambda = \{\lambda_i, ..., \lambda_N\}$ is a diagonal matrix of eigenvalues, and $\mathfrak{V}$ and $\mathfrak{U}$ are the matrices of right and left singular vectors respectively. The $k_\mathcal{S}$-dimensional coordinates are obtained as $\Lambda_{k_\mathcal{S}+}\mathfrak{V}^T_{k_\mathcal{S}+}$, where the subscript $k_\mathcal{S}+$ identifies that only the $k_\mathcal{S}$ largest non-zero eigenvalues and corresponding eigenvectors are used.

As an alternative, Normalized Cut embeds the coordinates in non-linear space which only preserves the geometry of the local neighbourhood. To model neighbourhood, the pairwise similarities between the vertices in the similarity graph are weighted by Gaussian kernel, i.e., $\varepsilon^s_{i,j} = \exp(\frac{-\|\mathrm{sim}(i,j)\|}{\sigma^2})$, where σ controls the width of the neighbourhood. The low-dimensional coordinates correspond to the $k_\mathcal{S}$ eigenvectors ($\mathscr{Q}^T_{k_\mathcal{S}-}$) with smallest non-zero eigenvalues (represented by subscript $k_\mathcal{S}-$) obtained by solving the generalized eigensystem. Algorithm 5 shows the overall steps needed to perform spectral clustering.

3.3.2 Effectiveness of Spectral Clustering

We show the effectiveness of our approach w.r.t. the quality of the identified clusters and reduction of false positives as compared to two related approaches.

3.3.2.1 Experimental Setup

Experiments are performed using PeerSim [JMJV]. We assume a content-based schema with up to 5 integer attributes, where the domain of each attribute is in the range [0,30]. Our approach, however, is not limited to a certain number, type or domain of attributes. Experiments are performed on two different models for the distributions of subscriptions and events. The uniform model (WL_1) generates random subscriptions/events independent of each other. The interest popularity model (WL_2) chooses five hotspot regions around which subscriptions/events are generated using the widely used zipfian distribution. For the experiments, up to $32,000$ subscriptions and $6,000$ events are used. Each event matches 5% of subscriptions and each subscription maintains a list of 20 most recently matched events.

We compare our work with two widely used related approaches. The first approach [CM07] (denoted as CK) clusters subscriptions according to the coordinates of their centroid using a k-means algorithm. The second approach (denoted as OV) is a representative of many prominent publish/subscribe systems [BFPB10, RLW+02]. It considers the absolute overlap between the subscriptions as a measure of their similarity and uses k-means or R-tree [YB07] algorithms to cluster them.

3.3.2.2 Quality of Clusters

The employed spectral clustering mechanisms provide no guarantee on the quality of the solution [Lux07]. We therefore evaluate the quality of generated clusters by measuring their *Entropy* [CKVW06, Sha48, ZK04] and *Accuracy* [CKVW06, ST00, FSS10]. The entropy specifies the disorder within the clusters, indicated by the distribution of ideal subscription clusters on the generated clusters, whereas accuracy measures the extent to which subscriptions are assigned to the correct clusters.

More precisely, for a given set of N subscriptions, let the correct (or optimal) classification of subscriptions into clusters be $\hat{\Pi_1}, ...\hat{\Pi_{k_S}}$. We refer to each $\hat{\Pi_i}$ as a class or an ideal cluster. Let $\Pi_1, ...\Pi_{k_S}$ be the clusters generated by performing spectral clustering. The entropy [CKVW06, Sha48, ZK04] of a cluster Π_j generated by spectral clustering

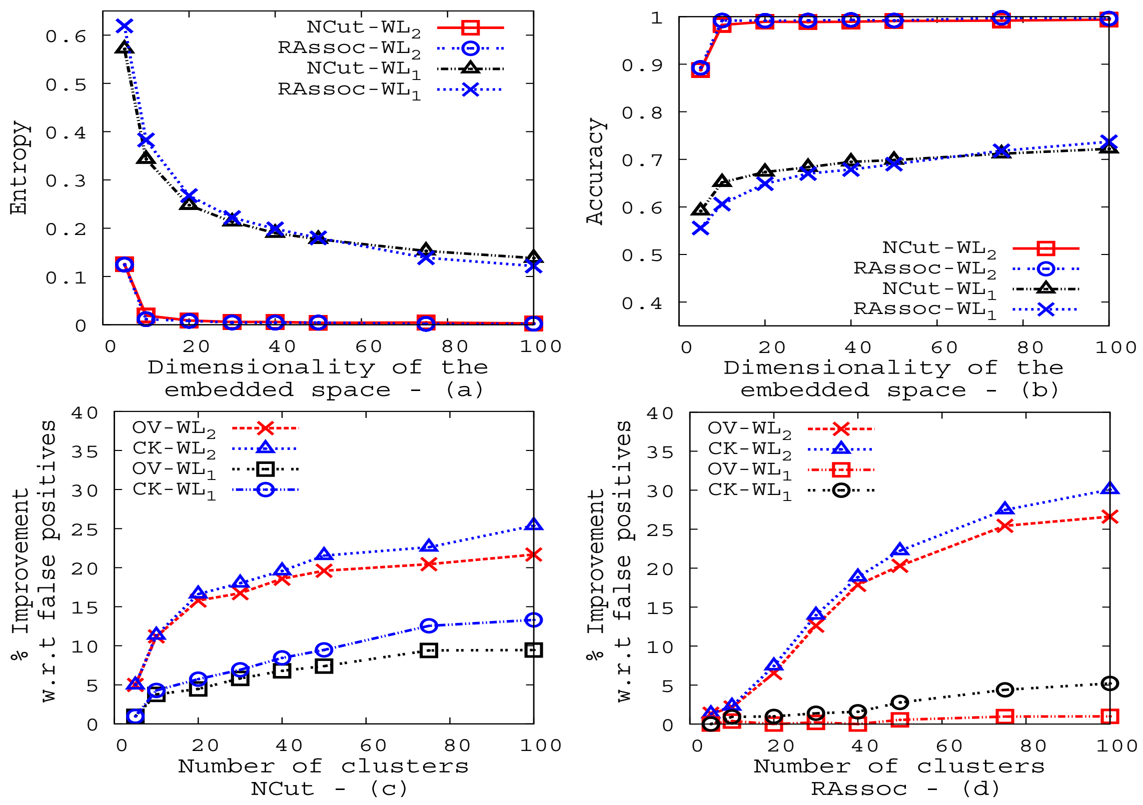


Figure 3.2: Evaluations for centralized spectral clustering.

is calculated as:

$$\mathrm{Et}(\Pi_j) = -\frac{1}{\log k_{\mathcal{S}}} \sum_{i=1}^{k_{\mathcal{S}}} (\frac{|\hat{\Pi}_i \cap \Pi_j|}{|\Pi_j|}) \log (\frac{|\hat{\Pi}_i \cap \Pi_j|}{|\Pi_j|})$$

where $|\hat{\Pi}_i \cap \Pi_j|$ is the number of subscriptions of class i in (generated) cluster j. The overall entropy of $k_{\mathcal{S}}$ clusters is the weighted sum of individual cluster entropies, i.e.,

$$\text{Entropy} = \sum_{j=1}^{k_{\mathcal{S}}} \frac{|\Pi_j|}{N} \ \mathrm{Et}(\Pi_j)$$

Similarly, the accuracy [CKVW06, ST00, FSS10] can be defined as:

$$\text{Accuracy} = \sum_{j=1}^{k_{\mathcal{S}}} \frac{|\Pi_j|}{N} \ \mathrm{Ac}(\Pi_j) \ , \ \ \mathrm{Ac}(\Pi_j) = \max_{i=1}^{k_{\mathcal{S}}} (\frac{|\hat{\Pi}_i \cap \Pi_j|}{|\Pi_j|})$$

Entropy and accuracy obtain scores in the range of [0,1]. A lower entropy and higher accuracy score imply better clustering.

Figures 3.2(a) and (b) show that for both of the alternative partitioning functions (RAssoc and NCut) in Algorithm 5, the quality of clusters improves with the increase in the dimensionality of the embedded space. It is shown that sufficiently good quality clusters can be obtained with just 20-dimensional space. Moreover, the proposed mechanisms perform reasonably well for both workloads (WL_1 and WL_2). In case of WL_2, which is the more realistic of the two workloads, clusters of almost all the subscriptions are identified correctly. However, the quality is not at its optimum for WL_1 because the subscriptions are uniformly distributed and some subscriptions may not share event traffic with any other subscription, making it hard to cluster them.

3.3.2.3 Reduction in False Positives

The reduction in the false positives is measured as the percentage improvement to the related centroid (CK) and overlap (OV) based clustering approaches. In all the experiments, subscriptions are generated using only uniform (WL_1) distribution, whereas events follow uniform (WL_1) and zipfian (WL_2) distribution. In these experiments we focus on studying the benefit (in terms of reduction is false positives) of clusters created according to the likeliness of subscriptions to receive similar event traffic in comparison to the related (CK and OV) approaches which do not take into account the event traffic. Generating subscriptions according to the zipfian distribution (WL_2) gives CK and OV approaches an unfair edge because of the presence of inherent clusters in the subscription workload, i.e., hotspot regions around which subscriptions are generated. Therefore, to quantify the actual benefit of creating clusters according to the event traffic based similarities only uniform subscription workload is used.

Figures 3.2(c) and (d) show that in the case of zipfian event distribution (WL_2) there is a considerable (up to 30%) improvement in reducing false positives in comparison to both the related approaches. However, for uniformly distributed event traffic the improvement drops to just $6-9\%$ for NCut and under 5% for RAssoc, which is predictable as there is no advantage of taking into consideration the event load based similarity in comparison to the absolute similarity metrics. Figures 3.2(c) and (d) indicate that RAssoc is better at reducing false positives in case of zipfian event distribution, whereas NCut performs better in case of uniform event workload.

3.3.3 Properties of Centralized Clustering

In practice, subscriptions of peers as well as event traffic change dynamically and therefore, over time the previously calculated clusters may become suboptimal. In order to adapt to the changes, a central coordinator periodically collects information about the events matched (in the recent time window) by the subscriptions of the peers and repeats the clustering process.

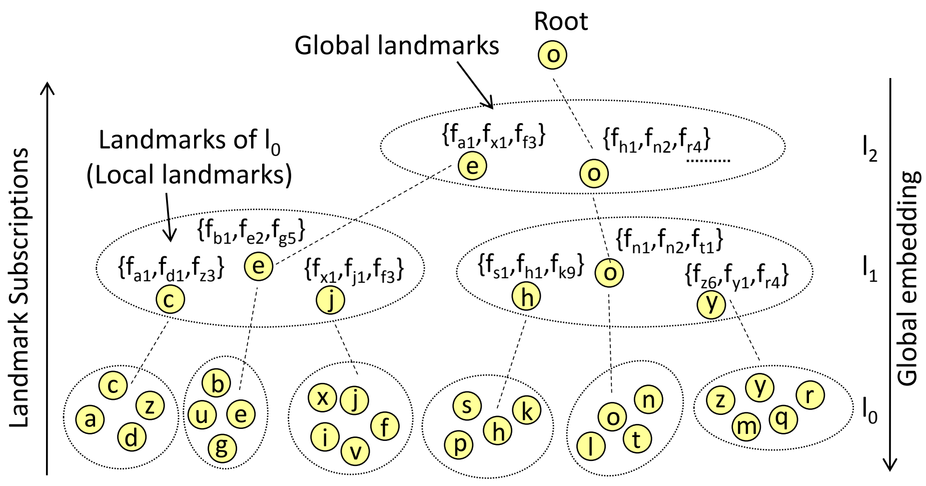

Root
Global landmarks
Landmarks of l_0
(Local landmarks)
$\{f_{a1}, f_{x1}, f_{f3}\}$
$\{f_{h1}, f_{n2}, f_{r4}\}$
$\{f_{b1}, f_{e2}, f_{g5}\}$
$\{f_{a1}, f_{d1}, f_{z3}\}$
$\{f_{x1}, f_{j1}, f_{f3}\}$
$\{f_{n1}, f_{n2}, f_{t1}\}$
$\{f_{s1}, f_{h1}, f_{k9}\}$
$\{f_{z6}, f_{y1}, f_{r4}\}$
l_2
l_1
l_0
Landmark Subscriptions
Global embedding

Figure 3.3: Hierarchical organization of subscribers.

The centralized subscription clustering approach, although still widely researched in literature [RLW+02, PC05, MSRS09], has some scalability issues. First, periodic fetching of the event histories, calculation of clusters and distribution of cluster membership information incurs significant overhead (in terms of bandwidth and processing resources) for the central coordinator. Second, the centralized spectral clustering is very expensive in terms of time and memory requirements. The calculation of the similarity graph encounters quadratic computational and memory requirements. Similarly, the calculation of $k_{\mathcal{S}}$ eigenvectors (of a dense matrix) involves $O(N^3)$ operations [FBCM04].

3.4 Distributed Subscription Clustering

In the previous section, we have described an approach to create and maintain the subscription clusters in a centralized fashion. In particular, we have identified centralized spectral methods that can solve the subscription clustering problem and have shown the effectiveness of those methods to produce good quality clusters in comparison to the related approaches.

Now, our aim is to develop an efficient and scalable approach to perform spectral clustering in a distributed fashion that can compute clusters with accuracy closely approximating the accuracy of the proposed centralized solution. In the subsequent sections, we will first describe the organization of the subscriber peers in an overlay (cf. Section 3.4.1) and afterwards present methods to perform dimensionality reduction (cf. Section 3.4.2) and k-means algorithm (cf. Section 3.4.3) in a distributed manner, exploiting the overlay organization.

3.4.1 Organization of Subscriber Peers

Subscriber peers are arranged into a multilevel hierarchy of small manageable groups as shown in Figure 3.3. The levels are numbered sequentially, with zero being the lowest level (denoted as l_0). All subscribers participate in the lowest level (l_0) groups such that the similarity graph formed by the subscriptions of the members of a l_0 group is connected.[†]

Each group selects a *coordinator* peer that joins the higher level group. A group at level l with coordinator p is denoted as G_p^l and the number of subscriptions maintained by this group is denoted by $|G_p^l|$. In the subsequent formulation, the subscript (p) will be dropped if it can be inferred from the context. At each level l, a coordinator p maintains a list of subscriptions (along with their matched events) called landmarks, which are selected uniformly at random from the members of group G_p^l. The landmarks participate in the higher-level group to create the similarity graph. For example, in Figure 3.3, coordinator c maintains three landmark subscriptions (from the peers a, d and z) which become part of the level l_1 group. The landmarks are used during distributed dimensionality reduction and for an accurate calculation of low-dimensional embedding the number of landmarks should be higher than the dimensions ($k_{\mathcal{S}}$) of the embedded space. This condition is necessary because to obtain $k_{\mathcal{S}}$-dimensional coordinates (of subscriptions) in the embedding space the eigen decomposition of the similarity matrix must result in $k_{\mathcal{S}}$ non-zero eigenvalues (cf. Section 3.3.1). However, the number of non-zero eigenvalues computed as a result of eigen decomposition (of similarity matrix) depends on the rank of the similarity matrix. In case the number of landmarks is less than $k_{\mathcal{S}}$ the resultant similarity matrix will be rank deficient and thus $k_{\mathcal{S}}$-dimensional coordinates cannot be obtained for the landmarks.

Maintaining a hierarchical overlay network as described above in dynamic conditions is a well-researched topic [BBK02, BFPB10]. For this reason, we will not discuss the maintenance algorithms in this chapter, but rather focus on the more challenging issue of performing distributed spectral clustering.

3.4.2 Dimensionality Reduction

To perform dimensionality reduction (low-dimensional embedding, (cf. Section 3.3.1) in a distributed manner, three main issues should be considered. First, the resulting low-dimensional space should be consistent so that the coordinates of different subscriptions can be compared. Second, the embedding should adapt to reflect dynamic changes in the similarities between the subscriptions. Third, the error induced due to the distributed calculation of the embedding should be small.

[†]A subscriber with completely dissimilar (disjoint) subscriptions may participate in multiple lowest level groups.

Our approach for distributed dimensionality reduction addresses the above issues and comprises two steps: i) separate (local) embedding of a small subset of subscriptions in low-dimensional space and, ii) transformation of these independently generated local embeddings into a globally unified coordinate system.

Each group in the hierarchical organization separately calculates low-dimensional coordinates of its subscriptions. The local coordinates (embeddings) are calculated in isolation from each other and therefore use different coordinate systems. As a consequence, similarities between the subscriptions with different local embeddings are not preserved, and thus the k-means algorithm cannot be applied. To overcome this problem, local embeddings are transformed (projected) into a globally unified coordinate system. The landmarks of the highest level (root) group define the basis of the global coordinate system. The global basis is progressively projected to adjust the local embeddings level by level (by traversing the hierarchy) until the lowest level groups are reached.

Each group G^l at level l maintains a *projection* matrix w.r.t. the parent group at level $l+1$. The projection matrix is used to transform the local coordinates of the group G^l to global coordinates as follows:

$$Y_{G^l} = \mathfrak{M}_{G^l} X_{G^l} \tag{3.3}$$

where $Y_{G^l} \in \mathbb{R}^{k_S \times |G^l|}$ are the projected (global) coordinates of the members of group G^l, $X_{G^l} \in \mathbb{R}^{k_S \times |G^l|}$ are the local coordinates and $\mathfrak{M}_{G^l} \in \mathbb{R}^{k_S \times k_S}$ (in short $\mathfrak{M}$) is the projection matrix. The calculation of local coordinates and the projection matrix depends on the properties of the low-dimensional space (i.e., linear or non-linear) as well as on the partitioning objective function. In the following, we will describe mechanisms for both the objective functions (RAssoc and NCut).

3.4.2.1 Ratio Association (Linear Embedding)

Again, the low-dimensional coordinates are calculated using PCA [CC00] to compute an explicit linear mapping between the original space (similarities between subscriptions) and the embedded space. In our distributed organization, we avoid the use of a global projection matrix in favour of separate matrices for each group which are maintained based only on the similarities between the groups' member subscriptions and local landmarks.

Local Embedding: The local coordinates of the landmark subscriptions ($X_{G^l}^{LM}$) maintained by the coordinator p of a group G^l are obtained by using the standard technique (cf. Algorithm 5, *line 2*). The local coordinates of all other subscriptions (X_{G^l}) of the group G^l are calculated w.r.t. these landmarks. Let $\mathfrak{W}_{G^l \rightarrow LM} \in \mathbb{R}^{|LM| \times |G^l|}$ denote the similarities between the subscriptions of G^l and the landmarks. The local coordinates

X_{G^l} can be calculated using Nyström Approximation [FBCM04] as follows:

$$X_{G^l} = \mathfrak{U}_{LM}^T \mathfrak{W}_{G^l \to LM}$$

Global Embedding: In order to convert the local coordinates (X_{G^l}) to the global coordinates (Y_{G^l}), a projection matrix is calculated by the coordinator. The landmarks maintained by the coordinator of a group G^l also participate in the parent group G^{l+1}. The projection matrix is calculated as a basis change matrix between the coordinates of the landmarks at two levels by solving a linear least square problem, i.e.,

$$\min \; ||(X_{G^l}^{LM})^T \mathfrak{M} - (Y_{G^{l+1}}^{LM})^T||$$

The matrix $(X_{G^l}^{LM})^T$ can be decomposed into $\mathfrak{U}\Lambda\mathfrak{V}^T$ using singular value decomposition (SVD) (cf. Section 3.3), where $\mathfrak{U} \in \mathbb{R}^{|LM| \times k_{\mathcal{S}}}$, $\Lambda \in \mathbb{R}^{k_{\mathcal{S}} \times k_{\mathcal{S}}}$ and $V \in \mathbb{R}^{k_{\mathcal{S}} \times k_{\mathcal{S}}}$. Thus the projection matrix is calculated as follows:

$$\begin{aligned} \mathfrak{U}\Lambda\mathfrak{V}^T \mathfrak{M} &= (Y_{G^{l+1}}^{LM})^T \\ \mathfrak{M} &= \mathfrak{V}\Lambda^\dagger \mathfrak{U}^T (Y_{G^{l+1}}^{LM})^T \end{aligned}$$

If the matrix $X_{G^l}^{LM}$ is rank deficient then Λ has some diagonal elements that are zero and cannot be inverted. Hence, the Moore-Penrose pseudo-inverse [CC00] $\Lambda^\dagger$ is used in the calculations.

3.4.2.2 Normalized Cut (Non-linear Embedding)

In the case of Normalized Cut, the dimensionality reduction step embeds the coordinates in a non-linear space. The reduced space does not preserve the global structure, it rather captures the geometries at a local neighbourhood in the similarity graph. Let y_i be the $k_{\mathcal{S}}$-dimensional coordinates associated with the i^{th} subscription, then the dimensionality reduction minimizes the following cost function [Lux07]: $\Phi = \sum_{i,j=1}^{N} \mathfrak{w}_{i,j} || \frac{y_i}{\sqrt{\mathfrak{d}_i}} - \frac{y_j}{\sqrt{\mathfrak{d}_j}} ||_2^2$. The cost function ensures that neighbours with larger weights in the similarity graph stay close in the low-dimensional space.

Local Embedding: Let $\mathfrak{W}_{G^l}$ denote the similarities between the subscriptions of group G^l, and $\mathfrak{D}_{G^l}$ is the corresponding degree matrix (cf. Section 3.2). The local coordinates (X_{G^l}) of the subscriptions of G^l can be calculated by performing generalized eigen decomposition and selecting the $k_{\mathcal{S}}$ smallest non-zero eigenvectors (cf. Algorithm 5, *line 2*).

Global Embedding: The projection matrix of a group G^l is calculated by minimizing the cost function of Normalized cut, i.e.,

$$\Phi = \sum_{i \in G^{l+1}} \sum_{j \in G^l} \mathfrak{w}_{i,j} ||(\frac{y_i}{\sqrt{\mathfrak{d}_i}} - \frac{\mathfrak{M} x_j}{\sqrt{\mathfrak{d}_j}})||^2 \tag{3.4}$$

Equation 3.4 calculates the projection matrix $\mathfrak{M}$ such that the error between the sum of squared distances of the projected coordinates of the subscriptions of G^l and subscriptions of the parent group G^{l+1} is minimized. In particular, to minimize Φ we set the first-order derivative of Equation 3.4 to zero so that the projection matrix $\mathfrak{M}$ can be obtained as:

$$\mathfrak{M} = Y_{G^{l+1}} \mathfrak{D}^{-\frac{1}{2}}_{G^{l+1}} \mathfrak{W}^T \mathfrak{D}^{-1} \mathfrak{D}^{\frac{1}{2}}_{G^l} X^{\dagger}_{G^l}$$

where $\mathfrak{W} \in \mathbb{R}^{|G^l| \times |G^{l+1}|}$ is the weight matrix which specifies similarities between the subscriptions of the groups G^{l+1} and G^l, and $\mathfrak{D} \in \mathbb{R}^{|G^l| \times |G^l|}$ is the degree matrix obtained by summing the row of weight matrix $\mathfrak{W}$. Moreover, $\mathfrak{D}^{\frac{1}{2}}_{G^l} \in \mathbb{R}^{|G^l| \times |G^l|}$ and $\mathfrak{D}^{\frac{1}{2}}_{G^{l+1}} \in \mathbb{R}^{|G^{l+1}| \times |G^{l+1}|}$ represent the degree matrices of the groups G^l and G^{l+1} respectively.

Once the projection matrix $\mathfrak{M}$ is calculated, Equation 3.3 can easily convert local low-dimensional coordinates into global coordinates.

3.4.3 Distributed K-means Algorithm

After low-dimensional embedding, the next step is to partition the subscriptions into clusters by performing a k-means algorithm. We propose two methods to obtain subscription clusters. Both methods benefit from the hierarchical organization of subscriber peers (cf. Section 3.4.1).

3.4.3.1 Sampling Method

In the hierarchical organization, landmark subscriptions maintained by a parent group are representative of the subscriptions in the sub-tree. Similarly, subscriptions of the root group represent the uniform sample of all the subscriptions in the system. Therefore, in the sampling method only the subscriptions maintained by the coordinator of the root group take part in the creation of k-means clusters. The rest of the subscriptions in the system are only placed in the created clusters without changing them, i.e., changing their centres.

Figure 3.4 shows the distributed k-means clustering using sampling method. The coordinator of the root group performs the k-means algorithm on its landmark subscriptions using their global coordinates. The resultant cluster centres are treated as the centres of the global clusters and distributed in the system. Finally each subscription (in the system) is placed in the global cluster whose centre is nearest to its global coordinates (cf. Section 3.5).

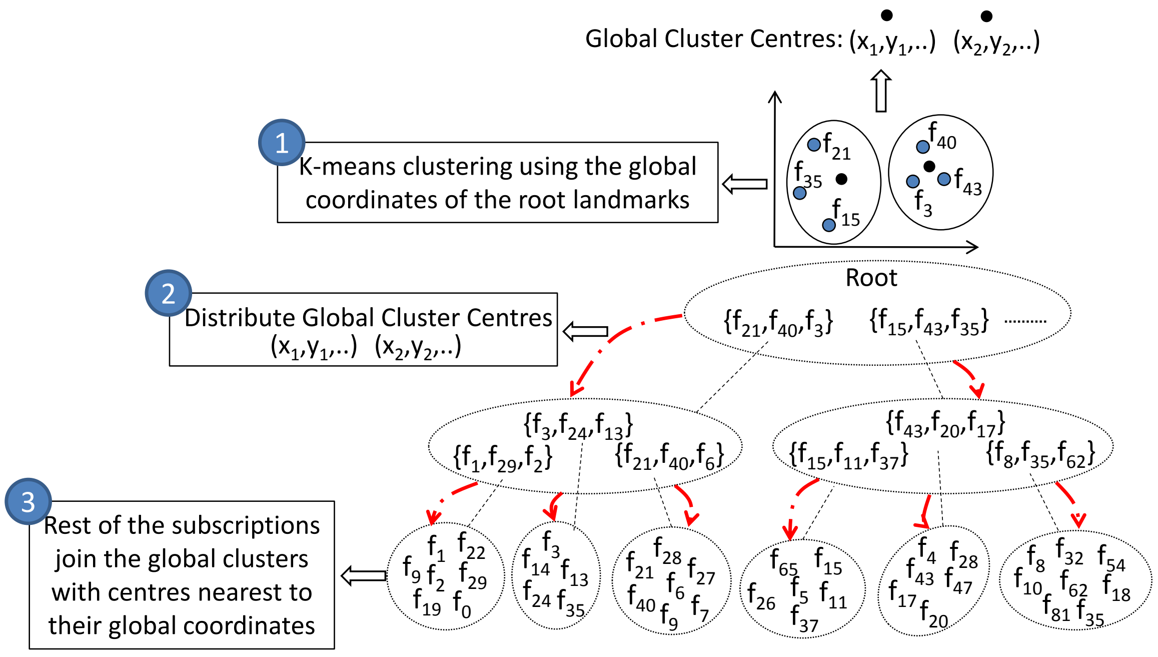

Global Cluster Centres: (x1,y1,..) (x2,y2,..)
1
K-means clustering using the global coordinates of the root landmarks
2
Distribute Global Cluster Centres (x1,y1,..) (x2,y2,..)
Root
{f21,f40,f3} {f15,f43,f35}
{f3,f24,f13} {f1,f29,f2} {f21,f40,f6}
{f43,f20,f17} {f15,f11,f37} {f8,f35,f62}
3
Rest of the subscriptions join the global clusters with centres nearest to their global coordinates

Figure 3.4: Distributed k-means clustering using sampling method.

3.4.3.2 Hierarchical Method

This method is adapted from the HP2PC [HK07] approach. In HP2PC each base (l_0) group in the hierarchy independently performs k-means algorithm to calculate local clusters. The locally calculated clusters are then merged (level by level) by their parent (higher level) groups until the global clusters are obtained at the root. We extended the HP2PC approach to enable the selection of good initial cluster centres to perform k-means at each l_0 group. The k-means algorithm is very sensitive to its initialization and therefore, selection of good initial cluster centres is important for the generation of good quality clusters.

The method works in two phases namely *Selection* and *Merge*. The selection phase provides good initial cluster centres for the k-means clustering performed at each l_0 group. In the hierarchical organization subscriptions of a parent group are representative of the subscriptions in the sub-tree, therefore cluster centres obtained by clustering subscriptions of a parent group provide good estimate of the initial centres for the child groups. The selection phase starts at the root and propagates down the hierarchy (cf. Algorithm 6, *lines 5-10*). While the root uses random initialization, each subsequent group uses the cluster centres calculated by its parent group to initialize its k-means clustering and then forwards the adjusted centres to its child groups.

The merge phase is initiated by the l_0 groups (cf. Algorithm 6, *lines 7-8*). During the merge phase, each intermediate parent group at level l combines the clustering information from the child groups and forwards it to the parent at level $l + 1$ (cf.

Algorithm 6 Hierarchical k-means algorithm

```
1: $Y_M$ represents $k_\mathcal{S}$-dimensional global coordinates of cluster members
2: $y_c$ represents $k_\mathcal{S}$-dimensional global coordinates of the cluster centre
3: $CL = \{y_c, Y_M, nCM\}$ // Entry for each cluster
4: $CS = \bigcup_{i=1}^{k_\mathcal{S}} (CL_i \setminus CL_i.Y_M)$ // Set of $k_\mathcal{S}$ clusters without membership information
5:    upon receive KMEAN_SELECT($p$, $l$, $CS_p$) at peer $q$ do
6:      $cSet = \{CL\}$ = centralizedKmean( $CS_p$ )
7:      if $G_q^l$ is leaf group then
8:        send (KMEAN_MERGE, $q$, $l+1$, $CS_q = \{CL_i \in cSet : CL_i \setminus CL_i.Y_M\}$) to parent group at $l+1$
9:      else // Send to all child groups
10:       $\forall p_i \in G_q^l$, send ( KMEAN_SELECT, $q$, $l-1$, $CS_q = \{CL_i \in cSet : CL_i \setminus CL_i.Y_M\}$ )
11:   upon receive KMEAN_MERGE($p$, $l$, $CS_p$) at peer $q$ do
12:     $CS = CS \cup CS_p$
13:     if $CS$ is updated for all child groups then
14:       $CS_q$ from Selection phase provides initial centres
15:       repeat // perform weighted k-means
16:         for all $cs \in CS$ do
17:           c = $min_{c_m \in CS_q}$ $|cs.y_c - c_m.y_c|$
18:           $LS_c+ = cs.y_c \times cs.nCM$ // Weighted linear sum of global coordinates of the cluster centres
19:           $c.nCM+ = cs.nCM$
20:         Recalculate cluster centres $\forall c \in CS_q$,i.e., $\frac{LS_c}{c.nCM}$
21:       until No change in the cluster centres of $CS_q$
22:     if $G_q^l$ is not a root cluster then
23:       send(KMEAN_MERGE, $q$, $l+1$, $CS_q$) to parent group at $l+1$
```

Algorithm 6, *lines 11-23*). Parents do not have the actual membership information of the clusters from the child groups. Therefore, the clusters from the child groups are treated as $k_\mathcal{S}$-dimensional points and the weighted k-means algorithm is performed such that the weight of each cluster point is determined by the number of members (i.e., subscriptions) associated with it (cf. Algorithm 6, *lines 15-21*). Finally at the end of the merge phase, the root obtains the information about the global clusters.

3.5 Cluster Creation and Maintenance

Once the global clusters are obtained at the root by means of distributed k-means algorithm, the information about the new cluster centres is distributed among the subscribers by transferring this information along the hierarchical organization (cf. Section 3.4.1). In addition, bootstrap peers to join the clusters are announced along with the cluster centres. On reception of this information, each subscriber peer joins the cluster Π_j whose centre is nearest to the global coordinates of its subscription f (i.e., $|y_f - y_{\Pi_j}|$ is minimum, where y_f and y_{Π_j} are $k_\mathcal{S}$-dimensional coordinates associated with the subscription f and the centre of cluster Π_j respectively). If the subscriptions of a subscriber are dissimilar and match different set of events, then the subscriber may join more than one cluster. Each cluster is maintained separately. Existing techniques such as IP multicast [Dee88] or Application layer multicast (ALM) [BBK02, RKCD01, HASG07] can be used to disseminate events within each cluster. Furthermore, additional mechanisms such as subscription forwarding [CS04, BBQV07, JE11, CRW04] can be performed within each cluster to even further reduce the rate of false positives.

P2P systems are very dynamic in nature and therefore, it is infeasible to completely recalculate the clusters for every minor change in the set of subscriptions. Thus, upon arrival of a new subscription f_n in the system, its subscriber s_n first contacts the coordinator p of its group G^l in the hierarchical organization to obtain the global coordinates of the subscription f_n (i.e., y_{f_n}) by means of the locally maintained projection matrix $\mathfrak{M}_{G^l}$. For instance, in the case of RAssoc the global coordinates of subscription f_n is calculated locally by the coordinator p by performing the following two steps:

$$\begin{aligned} x_{f_n} &= \mathfrak{U}^T_{LM} \mathfrak{W}_{f_n \to LM} && \text{(1. Local embedding)} \\ y_{f_n} &= \mathfrak{M}_{G^l} X_{f_n} && \text{(2. Global embedding)} \end{aligned}$$

where $\mathfrak{U}^T_{LM} \in \mathbb{R}^{|k_S| \times |LM|}$ is the matrix of left singular vectors obtained by the SVD of landmarks, $\mathfrak{M}_{G^l} \in \mathbb{R}^{k_S \times k_S}$ is the projection matrix, $X_{f_n} \in \mathbb{R}^{|k_S| \times 1}$ is the column vector of the local coordinates of subscription f_n, and $\mathfrak{W}_{f_n \to LM} \in \mathbb{R}^{|LM| \times 1}$ is the column vector of similarities between the subscription f_n and the landmarks. Once the global coordinates of subscription f_n are obtained, the subscriber s_n joins the cluster whose centre is nearest to the global coordinates of f_n. Similarly, to counter the effect of minor changes in the events' distribution, the coordinator p of each group G^l locally updates the local X_{G^l} and global Y_{G^l} coordinates of its member subscriptions (i.e., subscriptions maintained by the members of the group G^l) using the (locally maintained) projection matrix $\mathfrak{M}_{G^l}$. More precisely, coordinator p of group G^l collects the events recently matched by the member subscriptions and computes the similarity matrix $\mathfrak{W}_{G^l}$. Afterwards, local coordinates X_{G^l} of the member subscriptions are calculated by performing generalized eigen decomposition for NCut (cf. Section 3.4.2.2, *Local Embedding*) or singular value decomposition (and Nyström approximation) for RAssoc (cf. Section 3.4.2.1, *Local Embedding*). Finally, the local coordinates X_{G^l} are converted to global coordinates Y_{G^l} by using the projection matrix (i.e., $Y_{G^l} = \mathfrak{M}_{G^l} X_{G^l}$) and newly computed global coordinates are distributed to the group members. The new global coordinates of a subscription may change its nearest cluster centre; the subscriber of the corresponding subscription then joins the new cluster.

The dynamic changes in the subscriptions, as well as the event workload, can accumulate over time, so that the current set of clusters becomes suboptimal. In order to adapt to the subscription and event workload changes, the coordinator of the root group periodically starts the clustering process by sending a control message towards the base l_0 groups. The clustering process comprises two steps: i) embedding step, i.e., recalculation of projection matrices and global coordinates (cf. Section 3.4.2) and, ii) k-means step, i.e., creation of new clusters using distributed k-means algorithm (cf. Section 3.4.3).

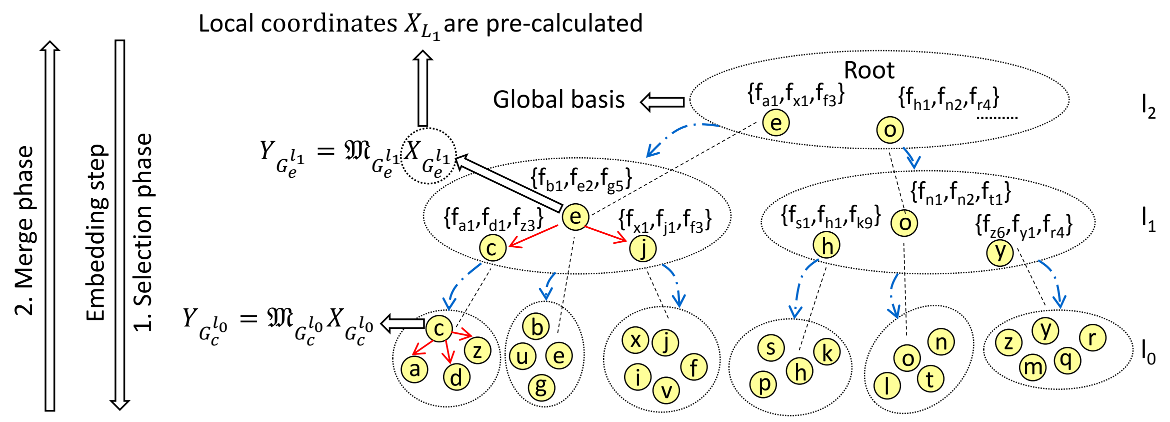

Local coordinates X_{L_1} are pre-calculated
Root
Global basis
$\{f_{a1},f_{x1},f_{f3}\}$
$\{f_{h1},f_{n2},f_{r4}\}$
l_2
$Y_{G_e^{l_1}} = \mathfrak{M}_{G_e^{l_1}} X_{G_e^{l_1}}$
$\{f_{b1},f_{e2},f_{g5}\}$
$\{f_{a1},f_{d1},f_{z3}\}$
$\{f_{x1},f_{j1},f_{f3}\}$
$\{f_{s1},f_{h1},f_{k9}\}$
$\{f_{n1},f_{n2},f_{t1}\}$
$\{f_{z6},f_{y1},f_{r4}\}$
l_1
$Y_{G_c^{l_0}} = \mathfrak{M}_{G_c^{l_0}} X_{G_c^{l_0}}$
l_0
2. Merge phase
Embedding step
1. Selection phase

Figure 3.5: Clustering process using hierarchical k-means method.

Embedding Step: On receiving the control message, coordinator p of each group G^l (on the path towards the base l_0 groups) updates the projection matrix $\mathfrak{M}_{G^l}$ and recalculates the global coordinates Y_{G^l} of its member subscriptions using Equation 3.3, as shown in Figure 3.5. The calculation of the projection matrix is very fast as each group accommodates only a small percentage of the overall subscriptions in the system (cf. Section 3.6). Furthermore, to expedite the embedding step the local coordinates (i.e., local embedding)[‡] X_{G^l} are pre-calculated asynchronously, independent of the control message. Finally, each member peer q in the group G^l receives the newly calculated global coordinates of the subscriptions maintained by it.

K-means Step: To speed up the clustering process, the k-means step is performed simultaneously along with the embedding step. In case sampling method is used to create k-means clusters, only the subscriptions maintained by the coordinator of the root group take part in the creation of clusters (cf. Section 3.4.3.1) and therefore, information about the (global) clusters is distributed along with the control message for the embedding step. In contrast, the hierarchical k-means algorithm works in two phases, i.e., *Selection* and *Merge* (cf. Section 3.4.3.2). The selection phase is performed simultaneously along with the embedding step, whereby the coordinator p of each group G^l on receiving control message from its parent group, i) updates its projection matrix and calculates global coordinates of members subscriptions as mentioned above, ii) uses the cluster centres calculated by its parent group to initialize its k-means clustering on newly created global coordinates, and iii) forwards the adjusted cluster centres to its child groups along with the control message. Subsequently, the merge phase is started by the coordinators of the base l_0 groups and propagates towards the root group, as shown in Figure 3.5.

Finally, it is important to mention that installing new clusters in the system incurs

[‡]The calculation of the embedding (cf. Step 2, Algorithm 5) is the most expensive operation (cf. Section 3.6.4).

control overhead and therefore, once the (global) clusters are obtained at the root group (as a result of sampling method or merge phase of hierarchical method), the coordinator of the root group compares the centres of the newly obtained clusters with the centres of the currently deployed clusters. Only if the new centres deviate more than a predefined threshold allows, new clusters will be installed, i.e., subscribers receive information about the new clusters and join their subscriptions one by one with the cluster whose centre is nearest to the subscription's global coordinates.

3.6 Performance Evaluation

We evaluate four aspects of our distributed approach: i) accuracy of the distributed dimensionality reduction, ii) quality of clusters created by the distributed k-means algorithm, iii) effectiveness of the overall distributed approach to reduce the cost of event dissemination under dynamically changing workload and in comparison to the related P2P-based approach, and iv) scalability of the cluster management in terms of computational time and load on peers.

The experimental setup is the same as described in Section 3.3.2. We modified the NICE protocol [BBK02] to use it for managing the hierarchical organization of subscribers. The number of peers in the experiments ranges from 1000 to 2000, with different percentages of churn. Moreover, Lvs denotes the number of levels in the hierarchical organization, LM denotes the percentage of subscriptions selected as landmarks in each group, and Gps denotes the number of l_0 groups.

3.6.1 Accuracy of Distributed Dimensionality Reduction

Two separate metrics are adopted to measure the accuracy of linear (RAssoc) and non-linear (NCut) dimensionality reduction. The *stress* [TC03, CC00] metric evaluates the quality of linear embedding by calculating the sum of squares of relative errors in the similarities of the subscriptions between the original space and the low-dimensional embedded space, i.e.,

$$\text{Stress} = \sum_{i\in\Pi}\sum_{j\in\Pi}\left(\frac{\mathfrak{w}_{i,j} - \bar{\mathfrak{w}}_{i,j}}{\mathfrak{w}_{i,j}}\right)^2$$

where $\bar{\mathfrak{w}}_{i,j}$ is the similarity between the subscriptions i and j in the low-dimensional space. The linear embedding is accurate if the value of stress is zero.

To measure the quality of non-linear embedding, we evaluate *trustworthiness* [VK06, VK01, FSS10] and *continuity* [VK06, VK01, FSS10] of the neighbourhood relationship between the original and the embedded space. More precisely, let $rk^O(i,j)$ denotes the rank of a subscription i, when the subscriptions are ordered according to their

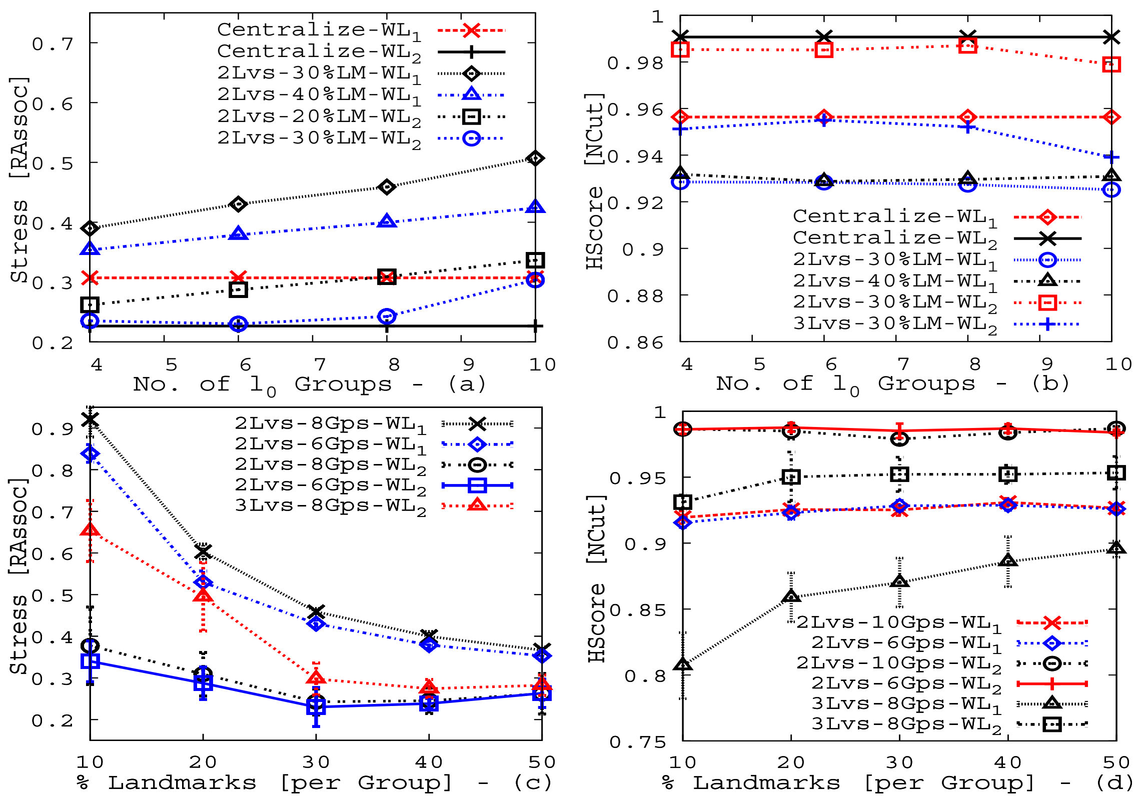


Figure 3.6: Effect of no.of l_0 groups and landmarks on the accuracy of embedding.

similarities from j in the original space. Likewise, $rk^E(i,j)$ defines ordering in low-dimensional embedded space. Moreover, let $Neigh^O_\eta(i)$ and $Neigh^E_\eta(i)$ denote the set of subscriptions which are η nearest neighbours of the subscription i in the original and the embedded space respectively. The trustworthiness (denoted as Tw) measures how far the neighbourhood in the embedded space differs to the original neighbourhood and is defined as:

$$\mathrm{Tw}(\eta) = \frac{2}{N\eta(2N - 3\eta - 1)} \sum_{i=1}^{N} \sum_{j \in Neigh^E_\eta(i)} (rk^O(i,j) - \eta)$$

The continuity (denoted as Cn) quantifies the deviation of the original neighbourhood from the embedded space neighbourhood and is calculated as:

$$\mathrm{Cn}(\eta) = \frac{2}{N\eta(2N - 3\eta - 1)} \sum_{i=1}^{N} \sum_{j \in Neigh^O_\eta(i)} (rk^E(i,j) - \eta)$$

The *harmonic mean* [FSS09] (denoted as HScore) of the trustworthiness and continuity

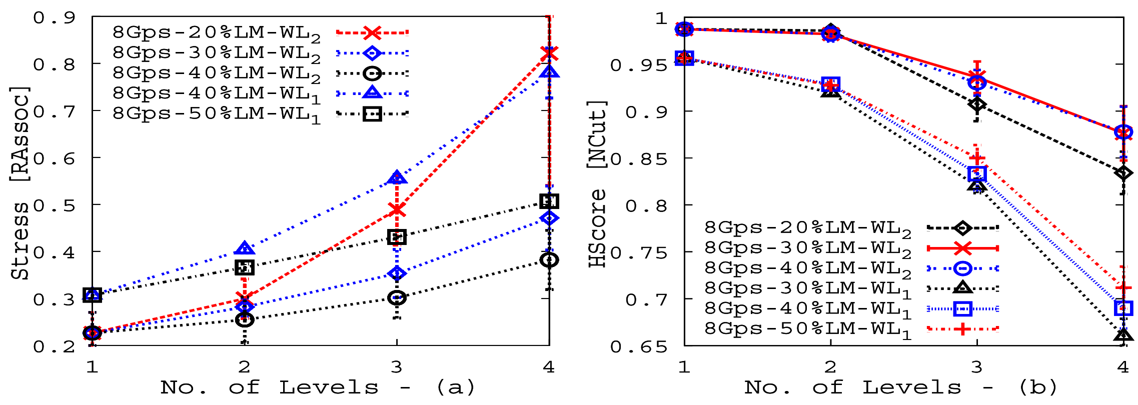


Figure 3.7: Influence of the levels of hierarchy on the embedding accuracy.

is expressed as:

$$\text{HScore} = \frac{2.\text{Tw}(\eta).\text{Cn}(\eta)}{\text{Tw}(\eta) + \text{Cn}(\eta)}$$

Higher values of HScore indicate a better embedding accuracy. The upper bound is 1, which indicates that the η nearest neighbours of every subscription are exactly the same in the original and the embedded space. In our experiments, we used a neighbourhood size of 50.

Figures 3.6(a) and (b) show the effect of the number of l_0 groups on the accuracy of the (linear and non-linear) embedded space in comparison to the centralized approach. All the l_0 groups have the same number of subscriptions. For instance, a total of 8 groups means that each group maintains 12.5% of the overall subscriptions in the system. The following conclusions can be drawn from the figures. First, the accuracy of the distributed approach with two levels of hierarchy closely resembles the accuracy of the centralized method for both types of embeddings. Second, the accuracy decreases slightly with the increase in the number of groups due to the corresponding decrease in the number of subscriptions per group. Third, the centralized and the distributed approaches produce embedding with better accuracy if the subscription workload concentrates on spots of interests (WL_2).

Figures 3.6(c) and (d) show the accuracy of the embedding as a function of the percentage of landmark subscriptions per group. In general, the accuracy increases with the increase in the percentage of landmarks. This behaviour is more significant with the increase in the levels of the hierarchy because each additional intermediate level decreases the overall percentage of subscriptions managed by the higher level groups by a factor of the number of subscriptions managed in the current level and the percentage of allowed landmarks in each group. Therefore lower percentages of landmarks result in significantly smaller numbers of subscriptions managed by the higher level groups. For instance, in case of 50% landmark subscriptions per group, the root of a 3-level

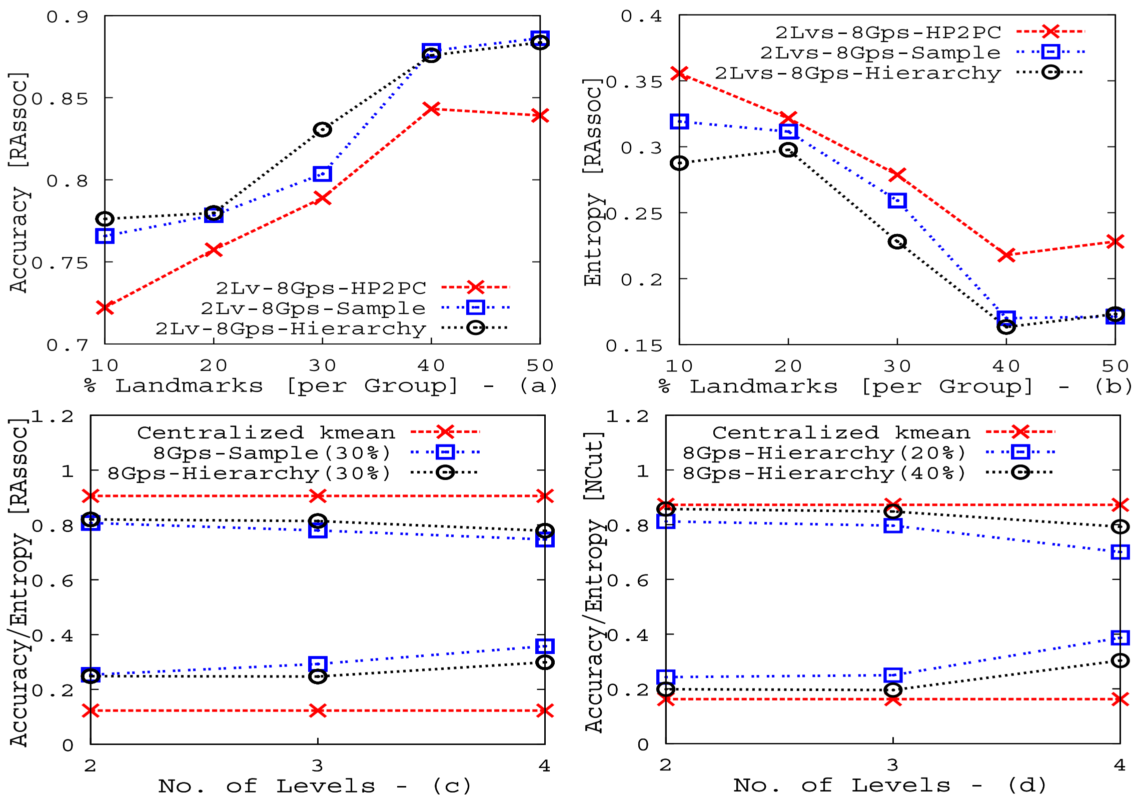


Figure 3.8: Quality of k-means clustering.

hierarchy manages 25% subscriptions, whereas with 10% landmarks, it manages just 1% subscriptions, decreasing the accuracy of the overall embedding. Figures 3.7(a) and (b) show that the accuracy decreases with the increase in the levels of the hierarchy. However up to 4 levels of hierarchy the embedding can be performed with reasonable accuracy. We argue that a 4-level system with 30% landmark subscriptions per group is very promising to perform good low-dimensional embedding. First of all the accuracy is only slightly decreased as compared to the centralized approach. Second, a 4-level system can handle subscriptions in a scalable manner. For instance, in the evaluated scenario, the groups at the intermediate levels manage 30% and 9% of subscriptions respectively, and the root group only manages 2.7% subscriptions. In Section 3.6.4, we show that 4-level system with only 25% landmark subscriptions can easily scale well beyond $32,000$ subscriptions.

3.6.2 Effectiveness of Distributed K-means Algorithm

We compare the quality of clusters obtained by the two proposed approaches, *Sampling method* (Sample) and *Hierarchical method* (Hierarchy), with the related distributed clustering approach HP2PC [HK07]. The quality of clusters is evaluated by *Entropy*

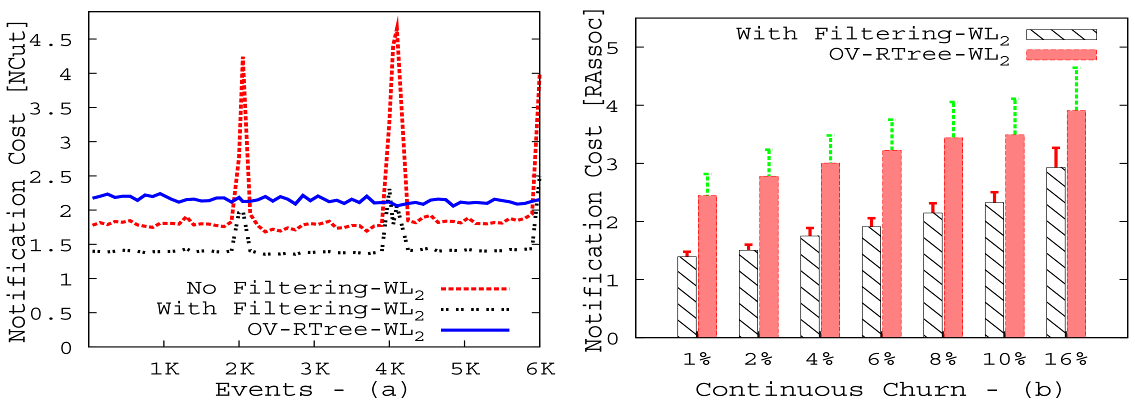


Figure 3.9: Evaluations for distributed clustering approach.

and *Accuracy* (cf. Section 3.3.2). Figures 3.8(a) and (b) show the values of entropy and accuracy versus the percentage of landmark subscriptions. As expected, the quality of clusters improves with the higher percentage of landmarks due to the corresponding increase in the accuracy of the low-dimensional embedded space. Moreover, the figures depict that the selection phase (cf. Section 3.4) is beneficial in improving the quality of clusters by providing a good estimate of the initial cluster centres. Figures 3.8(c) and (d) evaluate the effect of the number of levels on the quality of clusters. The trend shows that the quality of clusters gradually degrades with increasing levels of the hierarchy. There are many reasons for such a degradation: i) decrease in the accuracy of the low-dimensional embedding (cf. Figures 3.7(a) and (b)), ii) less accurate estimation of the initial cluster centres due to the decrease in the percentage of landmarks maintained by the root, and iii) loss of cluster quality at higher levels of hierarchy as a result of the merging of clustering information from the lower level groups [HK07].

3.6.3 Effectiveness of Overall Distributed Approach

We evaluate the effectiveness of our proposed clustering approach w.r.t. i) the cost of event dissemination, ii) the adaptability to dynamic changes in the event workload and iii) the resilience to peer churn. Similar to the work of Baldoni et al. [BBQV07], we define *notification cost* as the ratio of the complete traffic generated in the system to the number of subscriptions matched by the disseminated events. The traffic includes all the control overhead due to hierarchical organization, distributed embedding, k-means algorithm, maintenance of dissemination structure for each cluster and subscription forwarding. The notification cost expresses the efficiency of the event dissemination such that lower cost means higher efficiency. We compare two variants of our proposed approach (with and without event filtering in the clusters) with an overlap metric (OV) based approach. The non-filtering variant uses multicast to disseminate events within each cluster, whereas the filtering variant uses subscription forwarding to further re-

Parameter	Value
Dim. of embedded space	50
% of landmarks	25%
No. of clusters	50
Max. levels of hierarchy	4
Max. no. of l_0 Gps	64
Clustering method	Hierarchical

Table 3.1: Parameters for scalability evaluations.

duce false positives. Both variants use a two-level hierarchical organization to perform dimensionality reduction and calculate subscription clusters. The OV-RTree approach implements a distributed R-tree [BFPB10, YB07], such that non-leaf nodes can have maximum of 4 children. Figure 3.9(a) shows the notification cost during the evolution of the system as more and more events are disseminated. The measurements are taken after every 50 events. To evaluate adaptability, the hotspots of the event distribution are completely changed after every 2,000 events. The figure shows that immediately after the change in event workload, the notification cost of the proposed variants rises significantly for a small transient period, mainly because of the control overhead to calculate and install new clusters. However, the new clusters obtained after the transient period provide almost identical performance increase. Moreover, the figure depicts that the OV-RTree approach shows almost unchanged performance and is not responsive to the changes in the workload. Nevertheless, the proposed filtering and non-filtering variants shows 34% and 18% improvement in the cost of event dissemination (accounting all the control overhead associated with the P2P-based implementation of the approach) respectively in comparison to the OV-RTree approach.

Figure 3.9(b) shows the average notification cost experienced by the OV-RTree and our proposed approach in the presence of continuously arriving and leaving subscriptions. The churn percentage is relative to the total number of peers in the system. For instance, for a total of 2,000 peers, a churn of 10% means that in each time step, 200 online peers leave the system and the same number of new peers joins the system. The churn percentages of 10% and 16% can be seen as worst case scenario that puts the system under very stressing condition. The figure shows that the performance of both the approaches degrades gracefully with the increase in continuous churn. However, our proposed approach is more resilient to churn than OV-RTree, the reason being that the OV-RTree degrades over time (especially, if the newly arriving subscriptions are not added from the root) resulting in a higher rate of false positives and thus higher notification cost. In our proposed system, the periodic recalculation and installation of new clusters avoids the gradual degradation and thus the rate of false positives remains low over time.

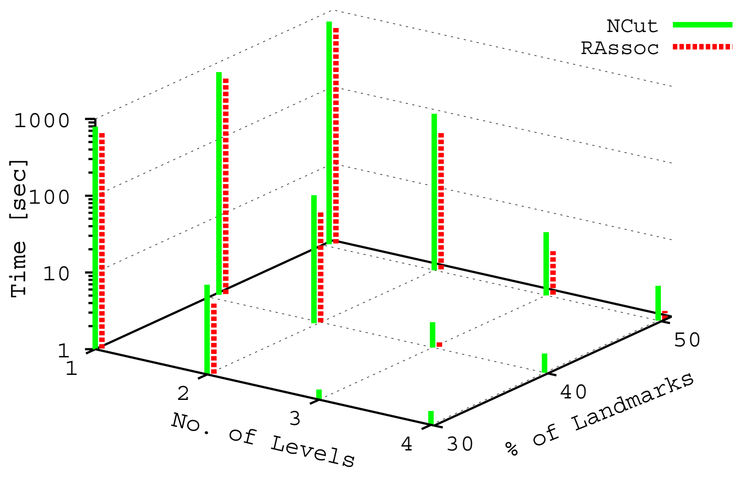


Figure 3.10: Time to perform dimensionality reduction.

3.6.4 Scalability of Cluster Management

In the previous section, we have shown the scalability of the system in terms of the cost of event dissemination under dynamically changing workload. In this section, we evaluate the scalability of our proposed mechanisms w.r.t. i) the overall time to cluster increasingly large number of subscriptions, and ii) the computational load on the peers participating in the hierarchical organization. Furthermore, we compare the computational times and the quality of clusters obtained by the linear and non-linear methods.

The matrix algebra needed to perform clustering is implemented using the JAMA library [HMW$^+$05], a linear algebra package for Java. All the measurements are made on a 2.7 GHz Intel core i7 CPU with 4GB RAM, running a 64 bit Windows 7 operating system. The delays between the communication links are chosen rather conservatively in the range [224 ms, 384 ms].

Figure 3.10 shows the cost in terms of time to perform the dimensionality reduction[§] for different percentages of landmark subscriptions and levels of hierarchy (one level means the centralized approach). For a hierarchy with 2 or more levels, the time is measured from the instant the root starts the dimensionality reduction process till the low dimensional coordinates of all the subscriptions are calculated. Thus the overall cost includes the time to perform local and global embeddings as well as communication delays between the peers along the longest (or the slowest) branch of the hierarchy. The number of subscriptions is fixed to 2,000 for this experiment. In the figure, some values for linear embedding (RAssoc) are not visible because of their smaller magnitude. The figure shows that the distributed approach drastically decreases the time to perform

[§]In this experiment, we solely focus on dimensionality reduction because it is computationally the most expensive operation.

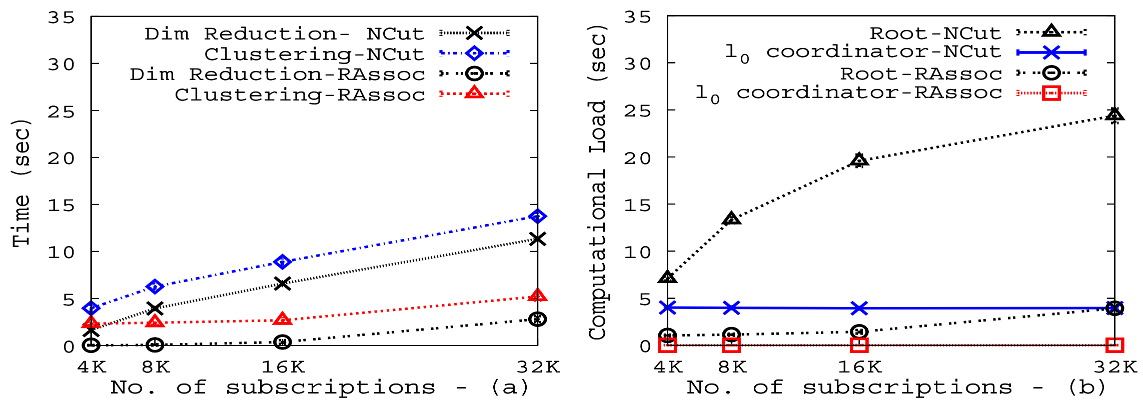


Figure 3.11: Scalability in terms of clustering time and computational load on peers.

dimensionality reduction in comparison to the centralized case. For instance, even for 2-level hierarchy with 50% landmarks the calculation time is decreased by 86% and 90% for linear (RAssoc) and non-linear (NCut) methods respectively. More than 99% decrease in time can be achieved with 4-level hierarchy. Moreover, the Figure 3.10 shows that the calculation time in general decreases with a decreasing percentage of landmark subscriptions and with an increasing number of levels of the hierarchy. Because of the decrease in landmarks and increase in levels, this reduction in time comes with a slight decrease in accuracy (cf. Section 3.6.1). However, with up to 4 levels of hierarchy and with only 25% landmark subscriptions, the clustering can be performed with reasonable accuracy (cf. Figures 3.12(a) and (b)). Another important observation is that the linear embedding (RAssoc) is less expensive compared to the non-linear embedding (NCut). For instance, in the centralized scenario RAssoc takes approximately 18% less time. The computationally intense generalized eigensystem problem is mainly responsible for the additional time in case of NCut. The difference in time (to perform dimensionality reduction) between RAssoc and NCut widens with the increase in levels of hierarchy. The reason is that in RAssoc, only landmark subscriptions are used to calculate the projection matrix, whereas in NCut, the projection matrix is calculated by performing least square optimization w.r.t. all the subscriptions in the parent group (cf. Section 3.4.2).

Figure 3.11(a) shows the overall time to perform dimensionality reduction and clustering in a distributed manner for different numbers of subscriptions. The parameters used in the experiment are specified in Table 3.1. The time is measured from the instant the root starts the clustering process till the clusters are calculated and clustering information is distributed among subscriber peers. This does not include the time to perform local embeddings because of the fact that the local embeddings are not calculated during the clustering (and projection) process (cf. Section 3.5). Figure 3.11(a) shows that even for computationally expensive NCut, up to 32,000 subscriptions can be clustered in less than 15 seconds including the communication delays. On the other

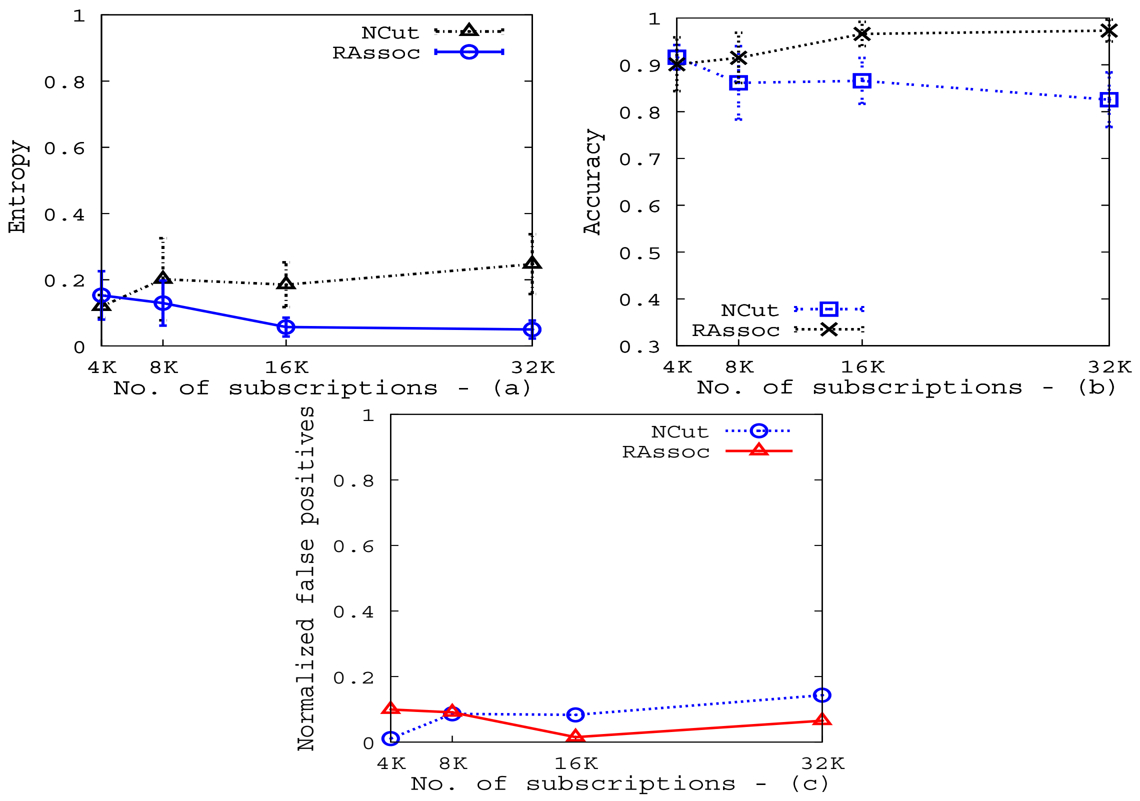


Figure 3.12: Quality of clusters created during the evaluations of Figure 3.11.

hand, clustering using RAssoc is very efficient and takes approximately 5 seconds. The time efficiency of RAssoc is due to the computationally less expensive eigen decomposition problem and the use of only landmark subscriptions for the calculation of the projection matrix, as mentioned above. Moreover, Figure 3.11(a) depicts that the distributed k-means clustering itself is computationally inexpensive and most of the clustering time comprises communication delays during the merge and selection phases (cf. Section 3.4.3), and the distribution of the cluster centres among subscribers.

Figure 3.11(b) shows the computational load experienced by different peers participating in the hierarchical organization w.r.t. the total number of subscriptions to cluster. The computational load of a peer is calculated by adding up times to perform local embedding, projection (global embedding) and clustering at each level of the hierarchy where the peer participates. In particular, the computational load of the root which participates at all levels of the hierarchy is compared with the load on the coordinators of lowest-level (l_0) groups. Figure 3.11(b) shows that the l_0 coordinators experience a computational load of less than 5 seconds in the case of NCut, whereas the load is negligible for RAssoc. Moreover, the load for l_0 coordinators stays constant with the number of subscriptions. This behaviour occurs because the peers can scale up the hierarchy to manage only a moderate number of subscriptions e.g., by dividing a group into two groups, each managing a smaller number of subscriptions[¶]. The root, on the other hand, participates at higher level groups and the number of subscriptions managed at each higher group depends on the subscriptions in the sub-tree. For instance, in case of $4,000$ subscriptions in the system, 25% landmarks and 4-level hierarchy, the highest-level (root) group manages only 63 subscriptions, whereas for $32,000$ system wide subscriptions, the number rises to 500. Therefore, the computational load of the root rises with the increase in the number of subscriptions. We argue that even for $32,000$ subscriptions, the computational load of the root is negligible in case of RAssoc and is easily manageable (under 25 seconds) for NCut. However, if the reduction in the load of the root (or the coordinators at the intermediate levels) is still desirable, there are two possible strategies that can be employed. First, the requirement that all the embedding calculations are performed by the coordinator of a group can be easily removed by allowing the coordinator to assign another participant peer from the group to perform calculations[‖]. This way the root can offload its computational load to different peers at different levels of the hierarchy. Second, the computationally expensive local embedding calculations can be optimized to reduce their time. The local embeddings are calculated by the coordinator peers in a centralized fashion (cf. Section 3.4.2) and therefore the optimizations available for reducing the time of centralized spectral methods [NXC$^+$10, DGK07] can be directly applied.

Figures 3.12(a) and (b) show the quality of the clusters created during the evaluations of Figure 3.11(a). It is clear from the figures that the quality of clusters obtained by NCut decreases slightly with the increase in the number of subscriptions. NCut preserves the geometries at local neighbourhood and is sensitive to the weights between the neighbours in the similarity graph (cf. Equation 3.4). With the increase in the number of subscriptions, the number of groups in the hierarchical organization also increases (to scale with the computational load as mentioned above) and therefore the neighbourhood information across the groups at the same level degrades, resulting in some loss of cluster quality. However, the quality of clusters obtained by NCut can be enhanced by increasing the percentage of landmark subscriptions. In the case of RAssoc, the quality of the clusters improves with the number of subscriptions. This is because RAssoc does not preserve the geometries at the neighbourhood but rather captures the global structure and therefore benefits from the number of subscriptions used as landmarks at the root. An increase in the number of subscriptions in the system results in a large number of landmarks managed by the root and thus in an improvement in the quality of clusters.

Finally, Figure 3.12(c) shows the quality of the created clusters w.r.t. the false positives in the system. The figure displays similar behaviour as depicted by Figures 3.12(a) and

[¶]Dynamic group management in hierarchical overlay networks is a well-researched topic [BBK02, BFPB10]. In our evaluations, a modified version of the NICE protocol [BBK02] is used for managing hierarchy.

[‖]The selection of another peer can be done in a random fashion or by considering some appropriate performance metrics.

(b). The evaluated scenario shows that NCut, which is computationally more expensive, is slightly better at reducing false positives for a moderate number of subscriptions, whereas RAssoc, which is computationally less expensive, performs relatively better in the presence of a very large number of subscriptions.

3.7 Related Work

Related work to our contributions in this chapter can be divided into two categories: i) subscription clustering in publish/subscribe systems, and ii) clustering using spectral methods.

3.7.1 Publish/Subscribe Systems

In the past few years, many content-based publish/subscribe systems have been proposed with scalability as the main design criterion [PC05, BBQV07, CS04, CJ11, CS05, JE11, CM07, BKR08, SMK12]. In order to achieve scalability, a large number of unnecessary events (false positives) are clearly undesirable and should be avoided [Que08]. Traditional systems [CRW01, TE04, LMJ08, CDNF01, FCMB06] rely on subscription forwarding and filtering mechanisms to restrict the dissemination of events to those parts of the overlay network where there are interested subscribers. However, these mechanisms introduce extra overhead and are not very efficient if subscribers that are interested in similar events are widely dispersed in the network [Que08].

Clustering of subscribers has been identified as a promising technique to achieve scalability [BFPB10, RLW+02, CM07, PC05, MSRS09, CJ11, Tar08, GaR03, VRKS06, LHLJ12]. Clustering can be achieved in two ways, either based on similarity of subscriptions or by partitioning the event space [WQA+02].

Kyra [CS04] partitions the event space into clusters, brokers are assigned to each cluster and subscriptions are moved to relevant brokers. Each cluster maintains a separate tree and performs filter-based event routing similar to Siena [CRW01]. However, the approach is not dynamic, and when a broker joins or leaves, the whole partitioning needs to be recomputed [ADG+04]. Additionally, no event partitioning criterion is specified.

Cheung el al. [CJ11] address efficient utilization of resources to reduce the IT operational costs for the enterprises with Green IT initiatives. They propose three subscription assignment strategies to minimize the total number of allocated brokers in a publish/subscribe system. One of the proposed strategies, namely CRAM (Clustering with Resource Awareness and Minimization), is capable of clustering subscriptions with similar interests. However, CRAM experiences longer computational time compared to prior subscription clustering approaches. Moreover, Cheung et al. only focus on a centralized and static scenario.

Riabov et al. [RLW$^+$02] propose offline approaches to group subscribers into a limited number of multicast channels using the techniques from data mining such as k-means, forgy k-means, pairwise grouping and approximate pairwise grouping. Similarly, Semcast [PC05] groups subscribers into semantic multicast channels for disseminating events, but this requires central coordination.

Our work differs from the current state of the art clustering approaches in four important ways. First, most of the existing approaches either assume point (topic-based) subscriptions or only consider the structural (absolute) similarities between the subscriptions. For instance, [PRGK09, GCV$^+$10, BBQ$^+$07, CMTV07, MNOP10, MZV07, WKM00] address clustering in topic-based publish/subscribe systems. Similarly, [APBSV10, RLW$^+$02, VRKS06, CM07] assume a predefined set of numeric attributes restricting the expressiveness of the subscription model. Second, the current load of the system in terms of event traffic is mostly neglected (especially in P2P-based systems) [BFPB10, CM07, VRKS06] and thus the efforts to place two subscriptions in a cluster are wasted if they will not be matched by any event [Que08]. Third, the approaches which take into account the current event load of the system are centralized [RLW$^+$02, PC05, MSRS09, CJ11, LHLJ12]. They assume the presence of a central coordinator which keeps track of the changing subscriptions and event load in the system. Furthermore, these approaches assume the presence of a dedicated network of brokers [CJ11, MSRS09, BBQV07, BCM$^+$99], which is fundamentally different from the P2P architecture of our work. Fourth, spectral clustering in the context of publish/subscribe systems has not been addressed previously in the literature.

3.7.2 Spectral Clustering

Spectral clustering has been shown to be more effective than traditional mechanisms such as k-means [Lux07]. However, nearly all existing spectral approaches are centralized and hence they cannot be directly applied in a distributed setting.

Dhillon et al. [DGK07] discuss the mathematical equivalence between weighted kernel k-means and spectral clustering. They developed a centralized multilevel approach to perform clustering using different graph partitioning objective functions without eigenvector computation. Ning et al. [NXC$^+$10] propose an incremental but centralized algorithm to project new samples (or data points) in non-linear space without recalculating the whole embedding.

Fowlkes et al. [FBCM04] reduced the computational overhead of Normalized cut by first solving the problem for a small random subset of data points and then extrapolating this solution to the full data set. However, the orthogonalization of eigenvectors requires complex calculations, which cannot be performed in a distributed manner.

Our approach for non-linear embedding is inspired from the work of Fang et al. [FSS10]. Similar to our work, Fang et al. perform non-linear dimensionality reduction by minimizing the least square error. However, Fang et al. approach is offline and centralized, which cannot be directly applied in distributed settings. Moreover, Fang et al. consider Ratio Cut [DGK07] as a graph partitioning objective function, which results in a different eigenvalue problem in comparison to the Normalized Cut and Ratio Association objective functions addressed in this work. Nevertheless, none of the spectral clustering approaches is targeted to handle continuously evolving workloads as is the case in P2P-based systems.

3.8 Summary

In this chapter, we have presented an approach to perform spectral clustering in a distributed manner. Although we focused on subscription clustering in a content-based publish/subscribe system, the proposed methods are general and can be applied to other areas such as document clustering, image segmentation or data mining. We have identified different spectral methods to perform subscription clustering and adapted them to work in distributed settings. In particular, we have developed mechanisms to perform linear and non-linear dimensionality reduction as well as k-means clustering using the hierarchical overlay organization. The evaluations show that our distributed mechanisms can i) drastically reduce the time by $86\% - 99\%$ to perform clustering, ii) effectively identify good quality clusters, and iii) significantly reduce the cost of event dissemination ($18\% - 34\%$) in a content-based publish/subscribe system.

Chapter 4

Underlay Topology Awareness

4.1 Introduction

In the previous chapters, we focus on overlay-level mechanisms that use the information about the event traffic and user subscriptions to organize peers in semantic (or interest) communities, i.e., peers matching similar events are placed in close proximity in an overlay network. Such a semantic organization of peers reduces the pure forwarders, i.e., the peers which participate in forwarding an event without a matching subscription, and as a consequence results in a decrease of overall message overhead as well as an improvement in the average number of overlay hops to deliver events. However, the overlay-level mechanisms alone may not provide desired benefits without the knowledge of the underlying physical network topology (in short underlay). For instance, two apparently independent overlay paths may share common underlying physical links [JTC08a] and therefore, the selection of overlay paths solely based on the semantic similarities between the peers may lead to multiple copies of the same messages on the shared underlay links resulting in higher message overhead (bandwidth utilization) and higher end-to-end delays, as illustrated in the following simple example.

Example 1 Figure 4.1(a) shows a small scenario where five subscribers and a publisher are connected to different routers in an IP (*Internet Protocol*) network. The subscribers have non-identical but overlapping subscriptions to a numeric attribute A and thus receive intersecting (or overlapping) sets of event messages. The overlay in Figure 4.1(b) organizes the subscribers according to the containment relationship between their subscriptions without the knowledge of the underlying network. Clearly, no extra message overhead is incurred (to deliver the events received by each subscriber) at an overlay-level, each subscriber only receives the matching events and forwards a subset of those events to its child subscribers. However, the communication cost in

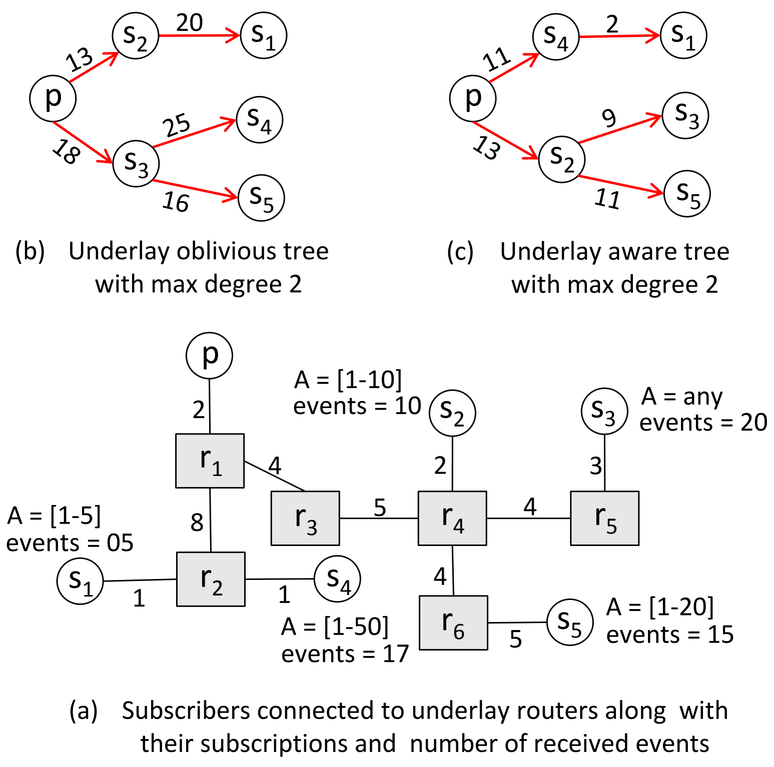

p
13
s2
20
s1
18
s3
25
s4
16
s5
(b) Underlay oblivious tree with max degree 2
p
11
s4
2
s1
13
s2
9
s3
11
s5
(c) Underlay aware tree with max degree 2
p
2
r1
4
8
A = [1-10]
events = 10
s2
2
r3
5
r4
4
r5
3
s3
A = any
events = 20
A = [1-5]
events = 05
s1
1
r2
1
s4
A = [1-50]
events = 17
4
r6
5
s5
A = [1-20]
events = 15
(a) Subscribers connected to underlay routers along with their subscriptions and number of received events

Figure 4.1: An example illustrating the benefits of underlay awareness.

terms of number of packets (or messages) travelled on each underlay link is high (i.e., 327 messages are forwarded in total) because overlay paths induce duplicate messages on common underlay links. Moreover, the delay penalty, i.e., delay in comparison to the unicast delay from the publisher, for individual subscribers is very high e.g., s_4 experience 3.9 times more delay. In contrast, the overlay in Figure 4.1(c) takes into account both the underlay topology and the containment relation between the subscriptions for its organization. Here, the subscriber s_2 has to receive and forward additional events to satisfy the subscriptions of s_3 and s_5 resulting in an overhead of 10 overlay-level messages. However, the delay penalty of subscribers is reduced to at most 1.2 and the communication cost is lowered to 246 underlay messages, which clearly shows that the semantic arrangement alone is not always beneficial in reducing message overhead and delay.

In this chapter, we present a novel scheme that exploits the knowledge of event traffic, user subscriptions, and the router-level topology of the underlying network to construct an efficient routing overlay, which minimizes the overall cost of disseminating events in a content-based publish/subscribe system [TKR13]. Our main contributions can be summarized as follows. First, we develop methods to discover the underlay topology among the peers participating in a publish/subscribe system. The proposed methods incur low overhead by reducing the number of measurements to discover the topology and maintain the topology information in a distributed manner by constructing a *Topology Discovery Overlay* (in short TDO), whereby peers are connected such that the overlap between the underlay routes (mapped by the overlay links between the peers) is minimized (cf. Section 4.4). Second, we propose different strategies to construct a cost efficient publish/subscribe routing overlay on a TDO, taking into account the underlay topology as well as the event traffic matched between the subscriber peers (cf. Section 4.5). Finally, through extensive evaluations, we show that the proposed approach performs well on Internet-like topologies, by leading to substantially low routing cost in terms of message overhead and end-to-end delays (cf. Section 4.6).

4.2 System Model and Problem Formulation

We consider a content-based publish/subscribe system consisting of a set of N distinct peers. The peers have unique identifiers and are connected to the underlying (IP) network at different points through access links. The underlay is modelled as an undirected graph $G^U = (\mathcal{V}^U, \mathcal{E}^U)$, where $\mathcal{V}^U = (\mathcal{R} \cup \mathcal{P})$ represents the set of routers $\mathcal{R}$ and peers $\mathcal{P}$, and $\mathcal{E}^U \subseteq (\mathcal{P} \times \mathcal{R}) \cup (\mathcal{R} \times \mathcal{R}) \cup (\mathcal{R} \times \mathcal{P})$ represents the set of physical links. Each link $\varepsilon^u_{i,j} = (i, j) \in \mathcal{E}^U$ from $i \in \mathcal{V}^U$ to $j \in \mathcal{V}^U$ is associated with a delay value $d^U(\varepsilon^u_{i,j})$.

The overlay network is the virtual topology induced by the peers on the underlay. It can be modelled as a graph $G^O = (\mathcal{P}, \mathcal{E}^O)$, where $\mathcal{E}^O \subseteq \mathcal{P} \times \mathcal{P}$ is the set of overlay links*. An overlay link is the point-to-point connection between two peers p and q. It is mapped to an *underlay route* consisting of a sequence of physical routers $r_i \in \mathcal{R}$ and physical links determined by IP routing, i.e., $\langle p, q\rangle = \{(p, r_i), (r_i, r_j).....(r_m, q)\}$. The delay of an overlay link $\langle p, q\rangle$ corresponds to the unicast path delay from peer p to q, i.e., $d^O(p, q) = \sum_{\varepsilon^u \in \langle p,q\rangle} d^U(\varepsilon^u)$. Note that the IP network routing is usually asymmetric [JTC08a] and therefore, the underlay route from a peer p to another peer q (i.e., $\langle p, q\rangle$) may not be the reverse of the route followed by $\langle q, p\rangle$. The methods developed in this chapter, work with asymmetry by treating $\langle p, q\rangle$ and $\langle q, p\rangle$ as two separate overlay links. However, for the ease of presentation, we assume that the underlay routes are symmetric and $d^O(p, q) = d^O(q, p)$. A prefix relation can be defined between peers w.r.t. the overlay links from a common ancestor peer. Let $\langle p, q\rangle = \{(p, r_{q,1}), (r_{q,1}, r_{q,2}), ..., (r_{q,j}, q)\}$ and $\langle p, s\rangle = \{(p, r_{s,1}), (r_{s,1}, r_{s,2}), ..., (r_{s,j+1}, s)\}$ represent overlay links of p with q and s respectively. Then $\langle p, q\rangle$ is said to be a prefix of $\langle p, s\rangle$ w.r.t. p, denoted by $q \rightsquigarrow^p s$, iff $r_{q,i} = r_{s,i}, \forall i \in \{1, ..., j\}$ holds. Moreover, the neighbourhood of a peer p in the overlay network is defined as the set of peers to which this peer has direct point-to-point connections (overlay links) and is denoted by $NG(p)$.

Peers act as publishers and/or subscribers, and run a content-based publish/subscribe protocol to exchange events. The publish/subscribe overlay is maintained as a spanning

*Some overlay links may not be possible to establish due to administrative or security reasons.

tree T of G^O, i.e., $T = (\mathcal{P}, \mathcal{E}^O_T)$, where $\mathcal{E}^O_T \subseteq \mathcal{E}^O$. In the following, we will refer to T as publish/subscribe overlay or tree. A *path* in T connects a peer p to another peer q over intermediate peers and is defined by a set of overlay (tree) links, i.e., $path(p, q) = \{\langle p, p_i\rangle, \langle p_i, p_j\rangle, ..., \langle p_m, q\rangle\}$. Furthermore, each path is associated with a delay value $D(p, q) = \sum_{\langle i,j\rangle \in path(p,q)} d^O(i, j)$. The dissemination of an event over T induces *routing cost* in terms of delay and message overhead. The routing cost is influenced by three factors, i) rate of false positives, ii) stress on physical links, and iii) relative delay penalty.

False positives measure the excess bandwidth consumption (in terms of extra messages) induced by the publish/subscribe tree T during the dissemination of events. They are defined as the rate of events that peers receive and forward without a matching subscription. Let E_ξ be the set of last ψ events published in the system. Each event $e \in E_\xi$ published by a peer p_e is delivered to a set of matching (subscriber) peers S_e by traversing only those links in T that lie on the paths between p_e and S_e. The *routing load* of a link $\langle i, j\rangle \in \mathcal{E}^O_T$, denoted by $\Upsilon(i, j)$, is defined as the number of overlay paths that pass through the link for the dissemination of all the events in E_ξ, i.e., $\Upsilon(i, j) = \sum_{e \in E_\xi} |\{q \in S_e : \langle i, j\rangle \in path(p_e, q)\}|$. Ideally to avoid false positives, the routing load on each link should be induced by the dissemination of only those events that are matched by the subscriptions of both of its adjacent peers.

The *Stress* of a physical link (i, j) is defined as the number of identical copies of a message sent over that link. Similar to false positives, stress measures the excess bandwidth usage and is influenced by the organization (topology) of peers in the publish/subscribe tree T. Ideally, a message should traverse each physical link at most once.

Relative delay penalty (in short RDP) measures the additional delay introduced by the publish/subscribe tree T on the delivery of an event from a publisher to all relevant subscribers. It is defined as the ratio of the delay experienced when sending events using the overlay to the delay experienced when sending events using the direct unicast path in the underlay [JMWB02]. The RDP value of 1.0 means that the delay at the overlay and the underlay is exactly same.

Given a dynamic set of (subscriber and/or publisher) peers $\mathcal{P}$ and continuously evolving event traffic E_ξ, our objective is to maintain a publish/subscribe tree T such that

1. the stress induced by the publish/subscribe tree on the links in the underlay is minimized, and
2. the cost for disseminating events is minimized in terms of false positives and delay, i.e., $\text{cost}(T, \mathcal{P}) = \min \sum_{\langle p,q\rangle \in \mathcal{E}^O_T} \Upsilon(p, q)\ d^O(p, q)$.

It is interesting to observe that the above problem statement involves three contradicting goals, i.e., i) to lower RDP, ii) to minimize stress, and iii) to reduce false positives. For instance, lowering RDP may increase stress on some links in the underlay. Consider a mesh overlay where all the peers have point-to-point connections (overlay links) with each other. In this case, RDP between all pairs of peers is 1.0, however, the stress on the physical links close to the peers is very high [JMWB02]. Similarly, organizing subscribers (according to the similarity of their received events) to avoid false positives conflicts with the other two goals (cf. Example 1).

So far in our discussion, we have only considered the maintenance of a single routing tree T to distribute events to all subscribers in a publish/subscribe system. However, it is interesting to point out that the cost to disseminated events (in terms of false positives) can be lowered by partitioning subscriptions into multiple clusters (according to their affinity to receive similar events) and maintaining a separate tree to distribute events in each cluster, as mentioned in the previous chapter (cf. Chapter 3). In the presence of such multiple (clusters) trees, the overall cost for routing events in a publish/subscribe system is given as: $\text{cost}(T, \mathcal{P}) = \min \sum_{i=1}^{k_S} \text{cost}(T_{\Pi_i}, \mathcal{P})$, where $T = \{T_{\Pi_1}, ..., T_{\Pi_{k_S}}\}$ represents the set of trees that are associated with the subscription clusters $(\Pi_1, ..., \Pi_{k_S})$. The approach presented in this chapter is independent of the fact that whether subscription clustering is employed or not. Therefore, in the rest of the chapter we assume the presence of a single cluster.

4.3 Approach Overview

Meeting the objectives presented in Section 4.2 amounts to finding a trade-off between three contradicting goals: i) to lower the relative delay penalty (RDP), ii) to reduce the rate of false positives, and iii) to minimize stress on underlay links. We therefore propose to address the problem by decomposing it into two layers, the topology discovery layer and the routing layer, as shown in Figure 4.2.

The topology discovery layer focuses on minimizing RDP and link stress without taking into account the subscriptions of peers and the event traffic matched by them. This layer maintains a topology discovery overlay network (TDO), which connects all the participants of a publish/subscribe system. In general, TDO can use any overlay link from $\mathcal{E}^O = \mathcal{P} \times \mathcal{P}$. However, to lower the delay penalty and limit the duplicate packets on the underlay, only those overlay links are selected, which minimize the overlaps between the mapped underlay routes, i.e., minimize sharing of underlay links or in other words connect peers according to their location in the underlay topology (G^U). Figure 4.2 shows that the peers are arranged in TDO such that the underlay routes mapped by the corresponding overlay links have minimum overlap. For instance, $q \rightsquigarrow^p s$ and therefore $path(p, s)$ (overlay path from p to s) in TDO passes through q. Connecting subscribers according to the underlay topology may induce higher stress on the last mile links e.g., (s, r) has a stress of 3. To alleviate this problem, peers actively monitor the bandwidth of their last mile links[†] and impose limits on their

[†]Bandwidth estimation tools such as pathchar [Jac97], clink [Dow99b,Dow99a], pchar [Mah00] or nettimer [LB01], can measure the available bandwidth of individual underlay links between any source

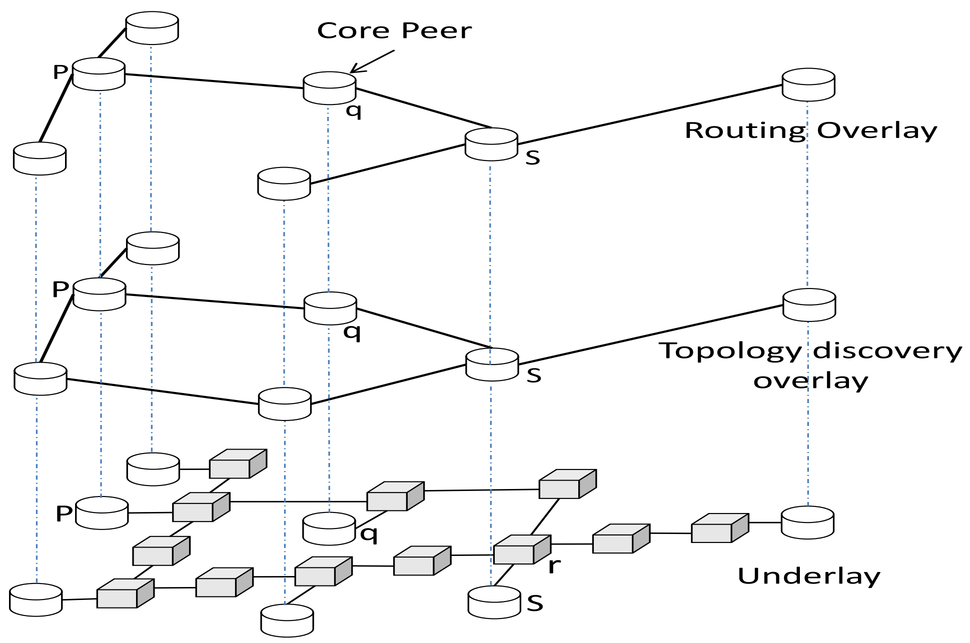


Figure 4.2: Decomposition into different layers.

degrees (neighbours in TDO).

In general, organizing peers in TDO according to the underlay topology requires tools (or techniques) to infer the underlay route among all pairs of peers in the system. A number of underlay route inference tools and techniques are discussed in Section 4.4.3. However, these tools are expensive in terms of time and control traffic. Therefore, the topology discovery layer employs methods to limit the underlay route inferences between the peers without much degradation in the quality of TDO in terms of minimizing stress and RDP.

The routing layer on the other hand runs on top of the TDO and maintains a spanning tree to distribute events (using a subset of overlay links from the TDO), as shown in Figure 4.2. To reduce the cost of event routing, the selection of links from the TDO is based on end-to-end delays between the peers as well as event traffic consumed or produced by them. In particular, a core-based approach is employed, whereby a small set of peers that experience higher routing load and have low delay paths (in the TDO) act as *cores*. The remaining non-core peers connect to their closest cores by using the lowest cost paths (in terms of dissimilarity of received events and end-to-end delays) in the TDO. The cost of event routing is sensitive to the selection of cores and therefore, various core selection strategies with different performance benefits are developed with complexity ranging from the use of global knowledge to only local neighbourhood-based

and destination on the internet. These tools can be employed to monitor the bandwidth of last mile links. Other bandwidth estimation tools maintained by MLab [Too12] and CAIDA [fIDA12] projects can also be used.

voting mechanisms.

In the subsequent sections, we first describe the construction of TDO (cf. Section 4.4) and afterwards present the maintenance of a cost efficient publish/subscribe routing tree on the TDO (cf. Section 4.5).

4.4 Topology Discovery Overlay

In this section, we tackle the problem of constructing a topology discovery overlay (TDO) with low stress and RDP. In particular, we will describe two approaches to maintain the TDO in a dynamic manner, the landmark approach and the random walk approach. The landmark approach extends the work of Kwon et al. [KF05] on underlay-aware single source multicast to support multiple sources in a scalable manner, whereas the random walk approach is more sophisticated, it overcomes the limitations of the landmark approach and addresses the trade-off between stress and RDP.

4.4.1 Landmark Approach

The landmark approach is inspired from the work of Kwon et al. [KF05], which addresses underlay-aware overlay creation with respect to only a single source. However, in a publish/subscribe system, every peer can be a publisher and a subscriber at the same time, and hence the TDO should reflect (discover) the underlay topology w.r.t. all the peers participating in the system. For this reason, the direct use of the approach of Kwon et al. [KF05] is not suitable and would result in an overhead of N^2 underlay route inferences, i.e., detection of router-level underlay path between all pairs of peers.

It is a known fact that the router-level network forms a sparse graph [FFF99] and therefore, underlay route inference between all pairs of peers is not necessary to construct a TDO that reflects a highly accurate underlay topology among publishers and subscribers [JTC08a]. For this reason, the landmark approach selects a small set of $k_{\mathcal{L}}$ peers as pivots (or landmarks), i.e., $k_{\mathcal{L}} \ll N$, and the underlay topology between the peers is discovered only with respect to the selected landmarks.

The set of landmarks is fixed and globally known to all peers in the system. Moreover, landmarks in the set are selected uniformly and independently from each other. Each peer individually picks a random number ρ in $[0, 1]$ and decides to become landmark if $\rho < \frac{k_{\mathcal{L}}}{N}$.[‡] To estimate the total number of peers N in a distributed and scalable manner, a gossip-based aggregation algorithm [JKS04, ZLP09] is used. The same algorithm serves the purpose of distributing the set of landmarks among other peers.

[‡]The accuracy of the discovered topology depends on the selection of landmarks. In general, the landmarks should be well distributed to discover high number of underlay links and routers. We used a lightweight selection method which gives reasonable performance, however, more expensive methods such as k-means, spectral clustering or maximum distance can also be employed [MSS06].

The TDO is maintained as a virtual forest of $k_{\mathcal{L}}$ logical trees, where each tree is associated with a landmark. Each landmark acts as the root of its associated tree while all other peers (including other landmarks) join the tree. In other words every peer in the system joins the tree of each landmark.

4.4.1.1 Joining a Landmark Tree

In order to connect to a tree associated with a landmark s_k, a newly arriving peer s_n sends a connection request to the root of the tree (i.e., s_k). On receiving the request, the root s_k discovers the underlay route mapped by the overlay link $\langle s_k, s_n \rangle$ by using one of the underlay route inference techniques described later in Section 4.4.3. Once the underlay route for $\langle s_k, s_n \rangle$ is discovered, the peer s_n is placed in the tree such that the newly discovered underlay route has minimum overlap with the underlay routes of other members of the tree. This is accomplished by comparing the underlay route of $\langle s_k, s_n \rangle$ with the routes of the neighbours (children) of the current peer at each level of the tree and performing one of the following three mutual exclusive actions [KF05] until the desired parent is reached.

Let s_t be the current peer in the tree which is processing the connection request from s_n. The algorithm deals with three possible cases as follows:

- *Case 1:* $\exists p \in NG(s_t) : p \rightsquigarrow^{s_k} s_n$

  In this case, the connection request is forwarded to peer p as it shares a part of the underlay route from s_k to s_n and hence is a more appropriate parent (for s_n) than s_t. For instance, if $\langle s_k, p \rangle = \{(s_k, r_i), (r_i, r_j), (r_j, p)\}$ and $\langle s_k, s_n \rangle = \{(s_k, r_i), (r_i, r_j), (r_j, r_o), (r_o, s_n)\}$, then placing s_n as a child of p in the tree reduces overlapping among the underlay links.

  It is important to mention here that every parent s_t maintains information about the underlay route of each of its children p on the landmark tree from the root s_k, i.e.,$\langle s_k, p \rangle$. When a join request from a new peer s_n arrives, this information is used to determine the prefix relation between p and s_n w.r.t. the root s_k, i.e., $p \rightsquigarrow^{s_k} s_n$.

- *Case 2:* $\exists p \in NG(s_t) : s_n \rightsquigarrow^{s_k} p$

  In this case, the underlay route of s_n is a prefix of the route mapped by $\langle s_k, p \rangle$ and hence, s_n should become a child of s_t and adopt p as its own child.

- *Case 3:* Neither Case 1 nor Case 2 evaluates to true.

  If no prefix relationship can exist between the underlay routes of s_n and the neighbours of s_t, then s_n joins s_t as a child.

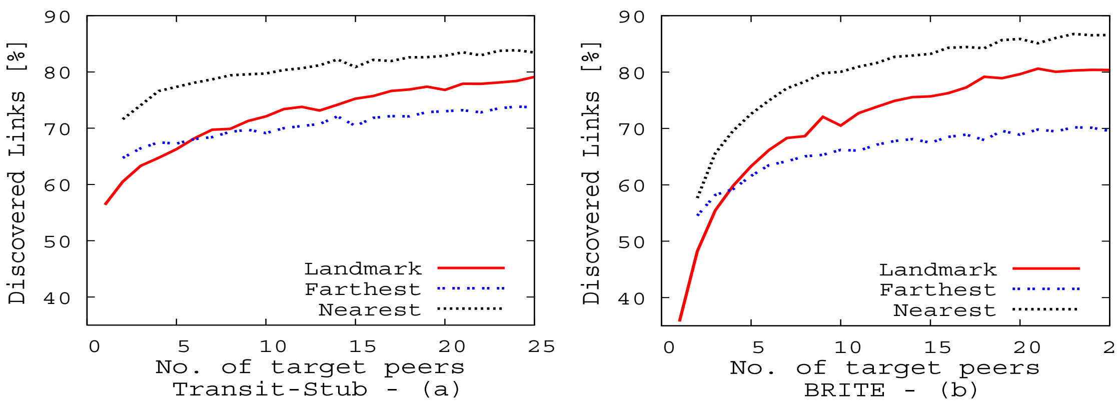


Figure 4.3: Selection of target peers vs. percentage of underlay links discovered.

4.4.2 Random Walk Approach

The TDO maintained by the landmark approach reflects the underlay topology among the participating peers with reasonable accuracy (cf. Figure 4.3). However, there are some drawbacks associated with the landmark approach. First, the stress induced by the TDO on the underlay links close to the landmark peers is high.[§] Second, the landmark approach is not very resilient against churn and the failure of an existing landmark requires all the peers to join a new landmark tree.

The above drawbacks can be avoided, if the landmarks are not fixed. Each peer joining TDO is allowed to infer underlay routes to $k_{\mathcal{L}}$ different peers (termed as *target peers*) and use the inferred routes to find a suitable position (to connect) in the TDO. An important consideration in this regard is how each peer selects its target peers? Clearly, the selection of target peers affects the discovery of the underlay topology (among the peers participating in the TDO) and thereby influences the stress and RDP. Figures 4.3(a) and (b) show the impact of the selection of target peers on the accuracy of discovered underlay topology, i.e., percentage of links discovered in the underlay. The figures depict the percentage of the discovered underlay links on GT-ITM Transit-Stub [ZCB96] and BRITE Multi-level [MLMB01] topologies, for two target peer selection criteria and the landmark approach. The nearest and the farthest criteria make use of round trip delays for the selection of target peers. The figures show that selecting $k_{\mathcal{L}}$ peers with the lowest delays as targets consistently performs better than the landmark approach. The farthest target selection criterion on the other hand, performs worst because the peers with high delays are usually in different Autonomous systems (AS). Inferring underlay routes between the peers in different AS discover the same backbone links repeatedly and consequently the TDO may results in higher stress on the underlay links as well as an increase in RDP [JTC08b]. The links within

[§]For instance, in a much skewed underlay topology, all the peers in the system may be directly connected to a landmark peer.

Algorithm 7 Random walk approach

```
1:  upon event Receive(RANDWALK, $p$, $D(p,q)$, $k_{\mathcal{L}}$ , TTL) at peer $q$ do
2:    TTL = TTL −1
3:    selectionCriteria()
4:    if $|NG(q)| > \ell(q)$ then
5:       peerToRemove = $\{s_1 : \forall a \in NG(q) \;\; d^O(q,s_1) > d^O(q,a)\}$
6:       trigger Send(DISCONNECT, peerToRemove)
7:    if $k_{\mathcal{L}} > 0 \wedge \text{TTL} > 0$ then
8:       peers[] = randomlySelectNeighbours ($NG(q), \sigma \times |NG(q)|$)
9:       for all $t \in$ peers do
10:          delay= $D(p,q) + d^O(q,t) - 2\times$ last mile delay of $q$
11:          trigger Send(RANDWALK, $t$, delay, $\lfloor \frac{k_{\mathcal{L}}}{|peers|} \rfloor$)
12: procedure selectionCriteria do
13:    switch (heuristic) do
14:      case DWorst:
15:         delay = $\{d^O(q,s_2) : \forall a \in NG(q) \;\; d^O(q,s_2) > d^O(q,a)\}$
16:      case DBest:
17:         delay = $\{d^O(q,s_1) : \forall a \in NG(q) \;\; d^O(q,s_1) < d^O(q,a)\}$
18:      case MLink:
19:         delay = $D(p,q)$ // Obtained in the RANDWALK message
20:   if $d^O(q,p)$ <delay then
21:      $k_{\mathcal{L}} = k_{\mathcal{L}} - 1$
22:      Infer underlay route: $\langle q,p\rangle = \{(q,r_i),(r_i,r_{i+1}),....,(r_j,p)\}$ using tools from Section 4.4.3.
23:      if $\exists s_1 \in NG(q) : s_1 \rightsquigarrow^q p$ then // Case 1
24:         trigger Send(CONNECT, $s_1$, $p$, $\langle q,p\rangle$)
25:      else if $\exists s_1 \in NG(q) : p \rightsquigarrow^q s_1$ then // swap $s_1$ and $p$, Case 2
26:         $NG(q) = \{NG(q) \setminus s_1\} \cup p$
27:         trigger Send(NEWPARENT, $s_1$, $p$)
28:         trigger Send(DISCONNECT, $s_1$)
29:      else // Case 3
30:         $NG(q) = NG(q) \cup p$
31:   if $p \in NG(q)$ then
32:      trigger Send(ACK, $p$)
```

a single AS are more diverse and therefore, peers with low delays should be preferred as targets[¶].

The evaluations in Figures 4.3(a) and (b) use global knowledge to select the peers with lowest delays. However, finding $k_{\mathcal{L}}$ nearest peers of a particular peer accurately in a distributed system is very expensive [CCRK04]. For this reason, our distributed approach does not focus on finding $k_{\mathcal{L}}$ nearest peers, but rather proposes three simple heuristics to select the target peers using random walk on the TDO.

A newly arriving peer p joins the TDO by sending the connection request to any existing peer. The connection request follows the protocol mentioned in Section 4.4.1.1 to find an appropriate place (neighbours in the TDO) for the new peer. Once joined, the peer p initiates a random walk (RANDWALK message) on the TDO. The number of peers visited by the random walk is controlled by the Time-to-Live (TTL) value and the neighbourhood selectivity factor (σ). The TTL value determines the number of forwarding hops, while the selectivity factor ($0 < \sigma \leq 1$) specifies the percentage of

[¶]Similar conclusions are drawn by the work of Jin et al. [JTC08b].

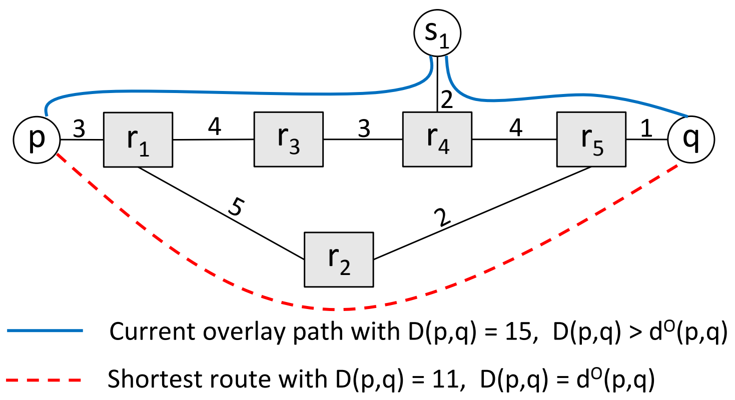

s1
p
r1
r3
r4
r5
q
r2
3
4
3
4
1
2
5
2
Current overlay path with D(p,q) = 15, D(p,q) > dO(p,q)
Shortest route with D(p,q) = 11, D(p,q) = dO(p,q)

Figure 4.4: MLink heuristic.

the neighbours to be included in the random walk at each forwarding hop.

Upon the reception of a random walk (RANDWALK) message from a peer p, the peer q employs one of the following three heuristics to determine its feasibility as a target to infer the underlay route to p (cf. Algorithm 7, *lines 13 - 19*):

- *DWorst Heuristic:* This heuristic selects q as a target if the unicast path delay from q to p (i.e., $d^O(q,p)$) is lower than the delay $d^O(q,a)$ to any existing neighbour a of q.[‖]
- *DBest Heuristic:* This heuristic is more restrictive than DWorst and selects q as a target only if $d^O(q,p)$ improves on the existing neighbour of q with the lowest delay.
- *MLink Heuristic:* This heuristic selects q as a target if some underlay links between p and q with lower delays are not discovered yet, i.e., not mapped by the overlay paths in the TDO. The peers are arranged in the TDO according to the prefix relation between their underlay routes and therefore, existence of undiscovered underlay links between p and q can be checked by comparing the delay on the $path(p,q)$ in the TDO (excluding the last mile delays of the intermediate peers) and the unicast delay $d^O(q,p)$, as shown in Figure 4.4. A smaller unicast delay $d^O(q,p)$ predicts the existence of undiscovered underlay links, which if included in the TDO leads to smaller delay on the $path(q,p)$.

In case peer q is selected as a target, it infers the underlay route of $\langle q,p\rangle$ using tools described in Section 4.4.3, compares it with the routes to other neighbours, and follows one of the three mutually exclusive cases mentioned in Section 4.4.1.1 (cf. Algorithm 7, *lines 22 - 30*). Following case 2 or 3 results in the acceptance of peer p as a neighbour of q. However, accepting peer p as a neighbour may violate the degree constraints (i.e., $\ell(q)$) imposed by peer q (to avoid bandwidth bottleneck on the last

[‖]The unicast delay $d^O(q,p)$ can be measured using *ping* tool, while the unicast delay to the neighbours $NG(q)$ is already known by each peer (as a result of previous join processes).

mile link) and therefore, an existing neighbour with the highest delay may be selected for disconnection (cf. Algorithm 7, *lines 4 - 6*)** Finally, the random walk message is forwarded to $\sigma \times |NG(q)|$ randomly selected neighbours along with the number of remaining targets to be selected. (cf. Algorithm 7, *lines 7 - 11*).

4.4.3 Techniques for Underlay Route Inference

Until now, we assume the availability of some tool to infer the underlay route between the peers. In this section, we give an overview of different techniques and tools which can be used for this purpose.

In general, the techniques to infer the underlay route (or topology) between a pair (or group) of peers mainly fall in two categories [ZP11]: tomography-based and router-assisted. The tomography-based techniques only discover the logical topology between the peers, induce high computation and communication overhead, and have limited accuracy due to certain assumptions about the statistical properties of the underlying network [ZP11, JTC08a]. We therefore propose to use router-assisted techniques. These techniques take advantage of the routers' response to the Internet Control Message Protocol (ICMP) based probe messages to infer the underlay route between the pair of peers. In particular, a peer can find the underlay route to other peers using tools, such as *traceroute*, *tracepath*, *tcptraceroute* etc. These tools have been extensively used for underlay topology discovery [KF05, DF07, WCVY03, SMWA04]. Recently, many new tools are proposed to overcome the limitations (e.g., due to load balancing in ISPs [KKSB07] or Multi-protocol Label Switching – MPLS [RVC01]) and increase the efficiency (especially the probe redundancy) of the standard tools [DRFC06, ACO+06]. A problem with router-assisted techniques is that roughly 1/3 of routers (termed as *anonymous routers*) do not respond to ICMP messages [KF05, ZP11, JTC08a]. These anonymous routers appear distinct in different underlay routes and therefore, inflate the discovered underlay topology [JTC08a]. Several merging techniques [JYCW06, GS08] are proposed to detect and collapse the anonymous occurrences of the same router. All these techniques are orthogonal to our work in this paper and can be integrated. Moreover, a reasonable percentage of routers respond to ICMP messages [KF05, ZP11, JTC08a] and thereby the inferred underlay routes (with distortions induced by anonymous routers) can still suffice to arrange peers in the TDO minimizing RDP and stress.

Another possibility is to rely on the data available from the internet topology discovery projects [SMW02, HPMC02, fIDA12]. Topology servers such as OSPF [SGG+02, GSB+11] can also be used to obtain ISP level (intra-domain) underlay routes by using simple network management protocol (SNMP) [CFSD90]. Moreover, as pointed

**To avoid partitions in the TDO, each peer additionally maintains a small set of overlay links to distant peers. Our evaluations show that maintaining a single link to a distant peer is adequate for this purpose.

out by Kwon et al. [KF05], the periodic logs of router configuration can be employed. However, underlay route information obtained from these techniques may not be very accurate due to dynamic nature of the internet [ZP11].

4.5 Event Routing

Until now, we have described the maintenance of the topology discovery layer. In this section, we address the problem of constructing a cost efficient routing tree T on a TDO, additionally taking into account the end-to-end delay and the event traffic matched between the subscriber peers.

The basic idea of our approach is to reduce the distance (delay) between the peers that consume or produce similar events by placing them nearby in the routing tree T (or in other words, to place peers matching dissimilar events away from each other). Therefore, the first step is to quantify the (dis)similarities between the peers to produce or consume similar events. We used the Jaccard function[††] defined in Section 3.3.1, for this purpose. Let $\overline{E}_{p_i}$ and $\overline{E}_{p_j}$ denote the events matched (or produced) from the set E_ξ by the peers p_i and p_j, respectively. Then the similarity between the peers p_i and p_j is calculated as: $\text{sim}(p_i, p_j) = \frac{|\overline{E}_{p_i} \cap \overline{E}_{p_j}|}{|\overline{E}_{p_i} \cup \overline{E}_{p_j}|}$. Accordingly, the dissimilarity between the peers can be derived as: $\text{dsim}(p_i, p_j) = [1 - \text{sim}(p_i, p_j)] \times \beta$. The dissimilarity values are normalized in the range $[0, \beta]$, where 0 means that peers consume/publish exactly same events and β is a constant that penalizes the dissimilarities.

Consequently, each overlay link in the TDO is weighted according to the delay (induced by the underlay) as well as the dissimilarity of its adjacent peers p_i and p_j to receive/publish same events, i.e., $w(p_i, p_j) = d^O(p_i, p_j) + \text{dsim}(p_i, p_j)$. Similarly, the weight of a path in the TDO, i.e., $path(p_i, p_n)$, is defined as: $W(p_i, p_n) = \sum_{\langle i,j\rangle \in path(p_i,p_n)} w(i, j)$. The links with smaller weights $w(p_i, p_j)$ should be preferred in T to reduce the routing cost. However, taking into account the weights of the links alone, for instance, by connecting peers in a minimum (weighted) spanning tree (MST), may results in a very high routing cost (cf. Section 4.6.3). The reason is that the structure of a tree, i.e., organization of peers, influences the routing load (defined in Section 4.2) on each (overlay) link and is therefore crucial for achieving lower routing cost. For instance, a tree where all peers are connected in a line induces higher load on the links $O(N^3)$ in comparison to the routing load (on the links) in a star tree with one internal peer $O(N^2)$ [WC04]. Similarly, source-based trees (i.e., separate shortest weighted path tree for each publisher peer) are not desirable as they impose serious scalability issues in terms of control overhead and size of the routing tables.

[††] The Jaccard function assigns similarities in the metric space [Cla06], i.e., similarities are non-negative, symmetric and obey triangular inequality.

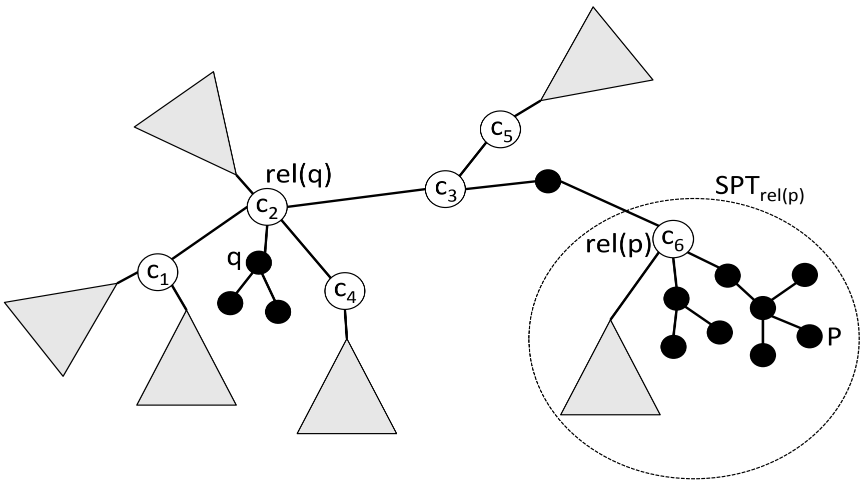


Figure 4.5: Core-based tree for event routing.

We therefore employ a core-based approach to build the publish/subscribe routing tree T. The approach selects a small set of peers, denoted by $\mathcal{C}$, as cores. In general, any peer in the system can be selected as a core. However, only those $k_{\mathcal{C}}$ peers which have low delay paths in the TDO and participate in the dissemination of many events are selected as cores. The core peers are connected with each other using the minimum weighted paths in the TDO. A separate shortest path tree (using link weights) is maintained by each core peer c, denoted as SPT_c. Each non-core peer p selects one of the core peers as its *relay*, denoted as $rel(p)$, such that $W(p, rel(p)) = min_{c \in \mathcal{C}}\, W(p, c)$, and joins the shortest path tree rooted at $rel(p)$ (i.e., $SPT_{rel(p)}$). A path in T between two peers p and q with different relays is composed of three sub-paths: from p to $rel(p)$, from $rel(p)$ to $rel(q)$ and, finally from $rel(q)$ to q, as shown in Figure 4.5.

Clearly, to realize core-based approach three main issues should be addressed: i) selection of good candidate peers to act as cores (cf. Section 4.5.1), ii) discovery and connection to the closest relays by non-core peers (cf. Section 4.5.2) and, iii) maintenance of the routing tree T in the presence of dynamics that arise due to changes in the event traffic and the subscriptions as well as churn/failures of the core and the non-core peers (cf. Section 4.5.3).

4.5.1 Core Selection

The core selection in a distributed environment mandates: i) election of a leader to decide the set of cores ($\mathcal{C}$) and, ii) a structure (e.g., tree) to communicate between the leader and the peers in the system. For this purpose, the peers maintain a spanning tree, denoted as $T^{\mathcal{C}}$, on the TDO such that the root of the spanning tree acts as a leader[‡‡]. Maintaining a distributed spanning tree in dynamic conditions is a well-researched topic [Awe87, CCK88]. For this reason, we will not discuss the maintenance algorithm in this chapter.

In the following, we present various core selection strategies with the variations in selection criteria, required knowledge and performance under different workload scenarios.

4.5.1.1 Strategies Based on Global Knowledge

These strategies assume that the leader has knowledge about the shortest (weighted) paths between all pair of peers in the system (All pairs shortest path – APSP). To acquire APSP knowledge at the leader we used the algorithm of Kanchi et al. [KV04]. The Kanchi's algorithm works on $T^{\mathcal{C}}$ and has a complexity of $O(N)$ message overhead and $O(N^2)$ message size. Once APSP information is available at the leader, two different core selection strategies can be employed.

Maximum Path Count (MaxPath): This strategy selects the peers with higher routing load as cores. In particular, first $k_{\mathcal{C}}$ highest loaded peers which participate in the dissemination of most events, i.e., a number of shortest paths pass through the peers, are selected as cores.

Shortest Path Cost (SPath): This strategy prefers those peers as cores, whose shortest path trees result in lower routing cost. More precisely, for each peer p, the weights along the shortest paths to all other peers are summed (i.e., $\sum_{\forall q \in \mathcal{P}} W(p,q)$) and the first $k_{\mathcal{C}}$ peers with the lowest values are selected as cores.

4.5.1.2 Strategies Based on Local Knowledge

Maintenance of APSP information (required by the strategies based on global knowledge) utilizes messages of very large size, i.e., $O(N^2)$, especially on the links near the root of $T^{\mathcal{C}}$ [KV04]. To overcome the problem, we employ voting-based mechanism where each peer votes for potential candidates to be selected as cores according to its local knowledge. In particular, each peer either votes for itself or its neighbour. The votes are aggregated towards the root (leader) of $T^{\mathcal{C}}$. To reduce the message size, a small (constant) number of core candidates with higher votes are kept at each aggregation step (each level of $T^{\mathcal{C}}$). Finally, the root selects the $k_{\mathcal{C}}$ peers with the highest votes as cores. Two different voting strategies are considered.

Closest Neighbour (CNeigh): This strategy promotes peers which are connected by low delay links and receive events similar to their neighbours (in the TDO) as cores. In this strategy, each peer votes for its neighbour with the lowest weight (i.e., neighbour with low delay and receiving similar events).

[‡‡]The problems of finding a spanning tree and electing a leader are closely related, and can be solved simultaneously [Awe87].

Algorithm 8 Core discovery and connection

```
1: $\forall_{c\in\mathcal{C}}\ w(c) = \infty$ // Initialization performed at each peer $q$
2:   upon event Receive(CORE_SPT, $c_i$, $W(c_i, q)$, $p$ ) at peer $q$ do
3:     $w(c_i) = W(c_i, q)$
4:     $Rcv(c_i) = Rcv(c_i) \cup p$ // msg for core $c_i$ is received from $p$
5:     if $\forall_{t\in\mathcal{C}}\ w(c_i) < w(t)$ then
6:       trigger Send(JOIN_CORE, $c_i$, $p$ )
7:       for all $v \in NG(q) : v \notin Rcv(c_i)$ do
8:         weight = $W(c_i, q) + w(q, v)$
9:         trigger Send(CORE_SPT, $c_i$, weight, $q$ , $v$)
```

Selection Potential (SPotent): This strategy is inspired from the centralized heuristic developed by Campos et al. [CR08] for the construction of a minimum cost multicast tree. In this strategy, each peer votes for itself by measuring its potential to be selected as a core. The selection potential of a peer p is determined by considering three characteristics: i) number of neighbours in the TDO (i.e., $NG(p)$) , ii) sum of weights to all the neighbours (i.e., $\sum_{q\in NG(p)} W(p, q)$) and, iii) worst delay to a neighbour (i.e., $D^W = \{d^O(p, q) : \forall a \in NG(p)\ \ d^O(p, q) > d^O(p, a)\}$). More precisely, the selection potential of a peer p is calculated as follows[§§]:

$$NG(p) + \frac{NG(p)}{\sum_{q\in NG(p)} W(p, q)} + \frac{1}{D^W}$$

4.5.2 Core Discovery and Connection

Once the core peers are selected and distributed by the leader (root of $T^{\mathcal{C}}$), the next step is to connect all the non-core peers to their relays (i.e., closest cores) using minimum weighted paths in the TDO. To accomplish this, each core c_i initiates the construction of a shortest path tree (SPT_{c_i}) by sending CORE_SPT message to its neighbours in the TDO. Upon the reception of a CORE_SPT message for SPT_{c_i} from a peer p, the peer q checks whether the minimum weight path to c_i improves on the weight of the shortest paths to other cores. If there is no improvement then the message is dropped from further forwarding (to reduce control overhead), otherwise the peer q joins SPT_{c_i} by sending join (JOIN_CORE) message to its parent p on SPT_{c_i}. Moreover, the CORE_SPT message for SPT_{c_i} is propagated to only those neighbours that have not received it yet (cf. Algorithm 8).

To avoid partitions in T, all the cores should also be connected with each other. During the core selection, the leader selects a core with the highest votes (similarly highest load or lowest cost in the case of MaxPath and SPath respectively) as the root core. All other cores connect to the shortest path tree of the root core similar to the non-core peers, as described in Algorithm 8.

[§§] Note that the inverse of last two terms, i.e., sum of weights to all the neighbours and worst delay to a neighbour, assures that the peer with the higher value is better.

4.5.3 Handling Dynamics

P2P systems are very dynamic in nature and therefore, complete recalculation of routing tree for every minor change in the event traffic or the set of peers ($\mathcal{P}$) is not feasible.

A newly arrived peer requests its neighbours on the TDO to forward the CORE_SPT messages and joins the SPT of the closest core (cf. Algorithm 8, *lines 1 - 6*). Similarly, disconnections (due to leaves/failures) of existing core and non-core peers are handled locally. A peer p on discovering the disconnection of parent q (which may be core itself) on $SPT_{rel(p)}$, joins the SPT of the second closest core. However, if the information about the minimum weight paths to other cores is not available (due to pruning of CORE_SPT message in Algorithm 8), the peer p requests its neighbours on the TDO to forward the CORE_SPT messages and joins similar to the arrival of new peer[¶¶]. Moreover, the CORE_SPT message of the selected core is forwarded to the child peers (on the SPT of previous $rel(p)$) to detect cycles and update weights to the new core.

The dynamic changes in the subscription and event workload as well as changes in the TDO due to arrivals/disconnections of peers, accumulate over time, so that the current routing tree T becomes suboptimal. In order to adapt to the changes, the leader of $T^{\mathcal{C}}$ periodically starts the core selection process and a new routing tree T is constructed. The time period for the construction of new routing tree T is a system parameter, which is specified by the administrator of the system.

4.6 Performance Evaluations

In this section, we evaluate the performance of the topology discovery layer and the event routing layer under different physical network topologies, dynamically changing workload, and in comparison to different baseline and related approaches.

4.6.1 Experimental Setup

Experiments are performed using PeerSim [JMJV]. Physical network topologies are generated using BRITE [MLMB01] and GT-ITM [CDZ] tools. BRITE generates a non-hierarchical Waxman topology [Wax88] (in short Wax) with 1640 routers. GT-ITM generates two layer hierarchy of transit and stub domains [ZCB96] (in short TS) comprising 1584 routers. Up to $N = 1024$ peers are used in the experiments, which are connected to random routers (stub domain routers for TS topology).[***] The last mile

[¶¶]In case TDO is partitioned due to disconnection, the peer p first initiates a join request on the TDO (cf. Section 4.4.2) and joins the routing tree T only afterwards.

[***]The accuracy of the topology discovery increases with the increase in the number of peers (cf. Section 4.6.2.3) and therefore, to mimic conservative (strict) settings moderate number of peers are used in the experiments.

delays (between the peers and their access routers) are set to be $5-10\%$ of the average delay between all the routers in the system. To avoid bandwidth bottlenecks on the last mile links, the limits on the degree constraints of the peers are chosen as $log_2 N$ (denoted as LG1), $2 \times log_2 N$ (denoted as LG2) and $\sqrt{N}$ (denoted as SQ). Moreover, NC represents the scenario where no degree constraints are imposed on the peers. For all the experiments, each peer performs only a single random walk. Unless otherwise stated, the neighbourhood selectivity factor is set to 0.1 and the number of target peers are chosen as 10.

The content-based schema contains up to 5 integer attributes, where the domain of each attribute is in the range $[0, 10]$. We would like to stress that in our system, similarities between the peers are calculated according to the recently matched events and therefore, results presented in this chapter are of general validity and are not limited to a certain number, type or domain of attributes. Experiments are performed on two different models for the distributions of subscriptions and events. The uniform model (in short WL_1) generates random subscriptions/events independent of each other. The interest popularity model (in short WL_2) chooses five hotspot regions around which subscriptions/events are generated using the widely used zipfian distribution [CMTV07]. For the experiments, up to 5000 events are used and each event matches 5% of subscriptions.

4.6.2 Performance of Topology Discovery Overlay (TDO)

We evaluate the effectiveness of the TDO to discover underlay topology, minimize stress and lower RDP w.r.t. i) number of target peers, ii) size of overall peers participating in the system and, iii) neighbourhood selectivity factor. Moreover, we analyse the control overhead induced by the TDO and its resilience to peer churn.

We compare our work with two baseline approaches. The first approach maintains TDO by inferring underlay routes to $k_{\mathcal{L}}$ arbitrary target peers (chosen randomly) and is denoted as *TarRand*. The second approach (denoted as *DOnly*) organizes peers in the TDO solely based on their end-to-end delays to the target peers without using the underlying topology information and is a representative of many systems [PWF07].

4.6.2.1 Influence of Target Peers

Figures 4.6(a) and (b) show the percentage of discovered underlay links versus the number of target peers ($k_{\mathcal{L}}$), for different (target) selection heuristics and degree constraints. As expected, the percentage of discovered underlay links increases by relaxing the degree constraints, for both types of topologies and all selection heuristics. Moreover, DBest heuristic performs slightly better by discovering higher percentage of underlay links. It is also worth noting that DBest and DWorst heuristics show similar results in

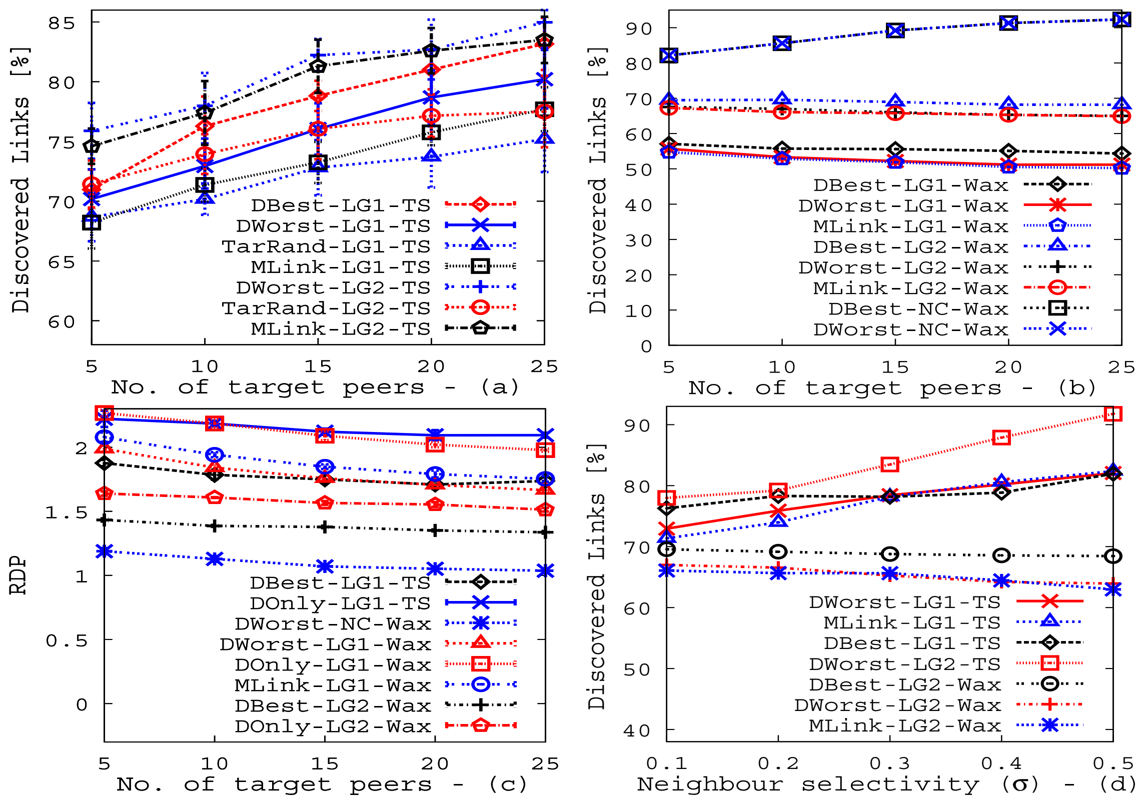


Figure 4.6: Influence of number of target peers and selectivity factor.

the absence of degree constraints. TarRand on the other hand performs poor, which shows that the proposed heuristics are beneficial for the selection of good target peers. Figures 4.6(a) and (b) depict that for transit-stub topologies, the percentage of discovered underlay links improves by increasing the number of target peers under all degree constraints. But in contrast, the percentage of discovered links decreases very slightly in the presence of degree constraints, for waxmann topologies. An explanation for such behaviour lies in the structure of the topologies. Transit-stub (TS) topologies have hierarchical structure with higher diversity of links within each stub domain and all the inter-stub domain links are channelled through the few backbone links in the transit domain. Peers connected to the routers in the same stub domain have low delays between them (due to underlay routes of shorter length) and therefore, increasing the number of target peers results in the discovery of more intra-stub domain links (as the proposed selection heuristics favour targets with low delays) improving the overall percentage of discovered links. Waxman topologies (Wax) on the other hand are flat and the underlay links are uniformly distributed among the routers, i.e., multiple underlay paths may exist between distant peers. Increasing the number of target peers in case of waxmann topologies results in an improved discovery of links between nearby peers in the underlay topology (similar to TS), but at the same time some links to the distant peers are disconnected due to degree constraints, slightly decreasing the overall percentage. However, in the absence of any degree constraints, the percentage of discovered underlay links improves with the increase in the number of target peers similar to TS, as shown in Figure 4.6(b). Nevertheless, the slight loss in accuracy (for Wax topologies) in the presence of degree constraints does not affect the corresponding RDP and Stress. Figure 4.6(c) plots RDP achieved by the TDO for different numbers of target peers and selection heuristics. RDP is measured as the ratio of delay experienced between all pairs of peers on the TDO to the delay experienced between them using the direct unicast paths in the underlay. It is clear from the figure that for Wax topologies, RDP decreases slightly with the increase in the number of target peers. Moreover, RDP is lowered quiet significantly by relaxing the degree constraints as expected. For instance, in the absence of degree constraints and 10 target peers, the RDP is 1.19 for Wax topologies. Furthermore, DOnly approach which does not utilize topology information results in higher RDP.

4.6.2.2 Impact of Neighbourhood Selectivity Factor

Figure 4.6(d) shows the influence of neighbourhood selectivity factor (σ) on the discovery of underlay links. Selectivity factor controls the spread of random walk. Higher selectivity values indicate that the peers nearby (in close neighbourhood) in the TDO are visited more by the random walk to find the target peers. The figure depicts that the percentage of discovered links increases with the increase in selectivity factor for TS topologies. This is because the peers are organized in the TDO according to their location in the underlay and therefore, with higher selectivity values peers connected to the same stub domain are mostly chosen as targets, which results in the discovery of greater number of new underlay links due to higher diversity of intra-stub domain links in TS topologies. However, the percentage of discovered links decreases slightly in the case of Wax topologies. This behaviour is due to the random nature of Wax topologies, as described in the previous section.

4.6.2.3 Effect of Number of Peers Participating in the System

Figures 4.7(a) and (b) illustrate the effectiveness of TDO to lower RDP and minimize stress, for different size of peers participating in the system. It is clear from the Figure 4.7(a) that RDP decreases gradually with the increase in the number of peers. The reason is that the percentage of access routers (i.e., routers directly attached to the peers through the last mile links) increases with the increase in the number of peers and therefore, the organization of peers in the TDO resembles underlay topology more accurately, enabling messages to follow overlay paths which are similar to the underlay routes (except the last mile links to intermediate peers in the TDO), decreasing the delay penalty. For the same reason, only moderate number of peers (up to 1024)

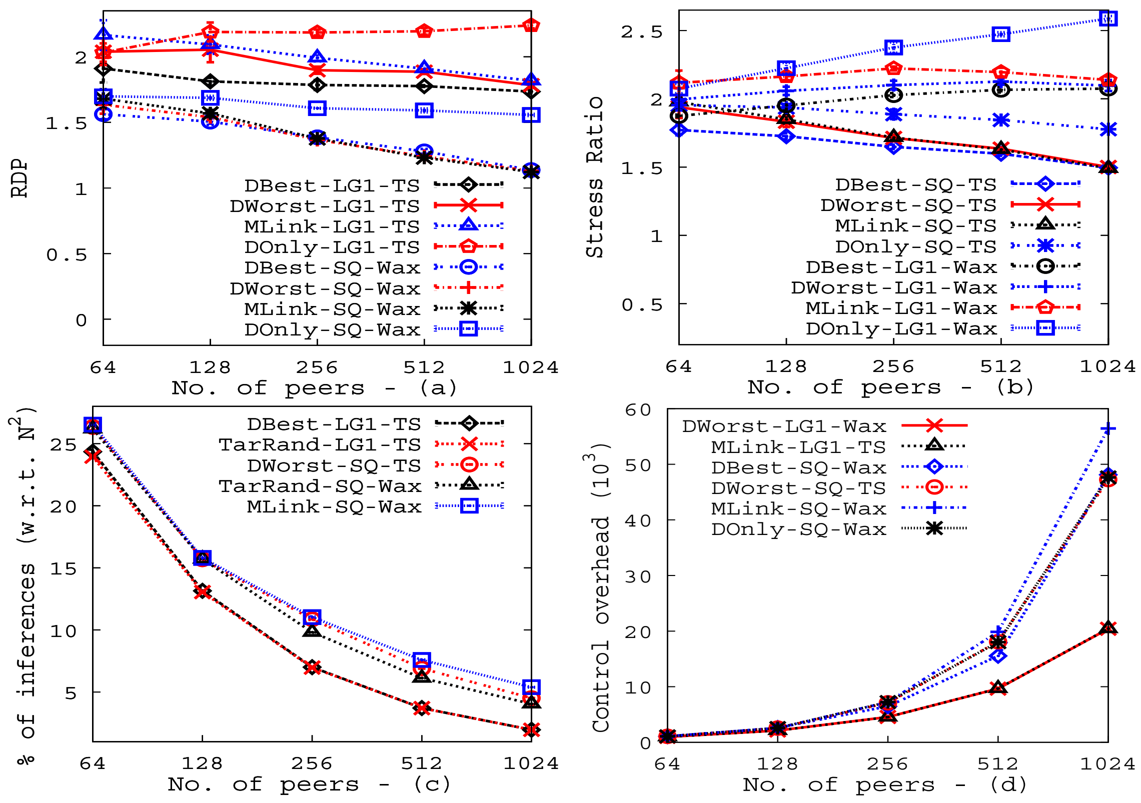


Figure 4.7: Impact of varying number of peers participating in the system.

are used in the evaluations. Figure 4.7(b) shows the impact of the number of peers (participating in the system) on stress. Clearly, the total stress introduced by the TDO increases in proportion to the number of participating peers, since more identical messages traverse physical links when more peers join the TDO. In order to compare the effectiveness of the TDO to reduce stress across different peer sizes, Figure 4.7(b) plots the ratio of total stress introduced (as a result of communication between all pairs of peers) by sending messages on the TDO to the stress introduced when the messages are routed directly on the underlay. The lower value of stress ratio represents higher effectiveness in minimizing stress. Figures 4.7(a) and (b) show that the performance of all three selection heuristics becomes almost similar with the increase in the number of peers and the relaxation of degree constraints. The DOnly approach which does not take into account the underlay topology experiences up to 39% higher RDP and 30% rise in stress. For instance, in the presence of 1024 peers and SQ degree constraints on Wax topologies, DOnly induces $3,703,702$ more messages on the physical links in comparison to MLink.

Figure 4.7(c) shows the influence of the number of peers on the control overhead due to the inference (discovery) of underlay routes between the peers. Underlay route inference techniques are very expensive in terms of computation and communication

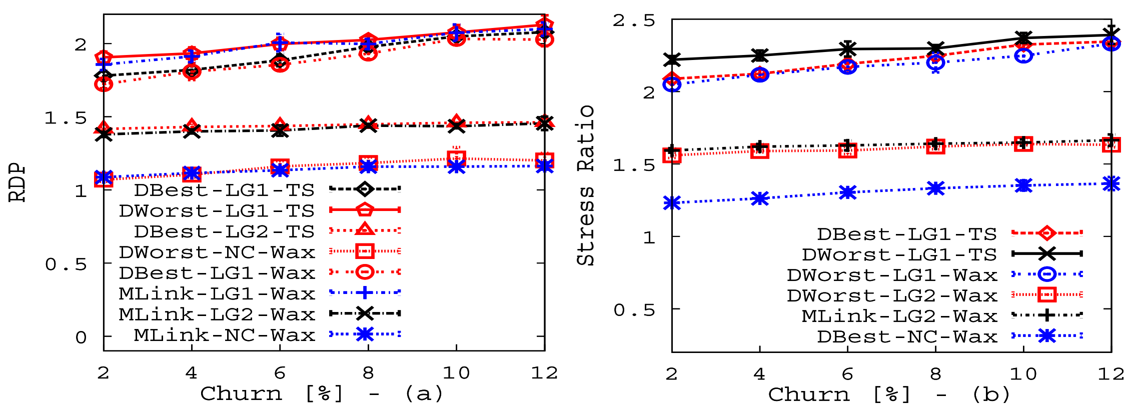


Figure 4.8: Adaptability of TDO to continuous churn.

cost (cf. Section 4.4.3) and therefore, number of underlay route inferences should be minimized. The figure plots the percentage of underlay route inferences in comparison to the naive approach, whereby underlay routes are discovered between all pairs of peers. The percentage of inferences decreases with the increase in the number of peers mainly because the number of targets is kept constant. Figure 4.7(d) displays the control overhead due to peer joins during the construction of the TDO. In general, MLink performs slightly more underlay route inferences and produces higher control overhead. This is because in MLink target peers are selected according to the difference between the delays on the paths in the TDO (excluding the last mile delays) and the unicast delays. This may results in TDO connections between arbitrary peers which have high delays or are not located nearby in the underlay topology and hence, resolves in comparatively higher number of disconnections (e.g., when a peer with lower delay sends join message, the neighbour with the highest delay is disconnected), producing slightly more underlay route inferences and join messages.

4.6.2.4 Adaptability to Dynamics

We next quantify the resilience of the system in the presence of continuously joining and leaving peers. The churn percentage is relative to the total number of peers in the system. For instance, for a total of 500 peers, a churn of 10% means that in each time step almost 50 online peers leave the system and same number of new peers (connected to different routers with different delays) join the system. The churn percentages of 10% and 12% can be seen as worst case scenarios that put the system under very stressing conditions. Figures 4.8(a) and (b) plot RDP and stress achieved by TDO in the presence of different percentages of churn. The figures show that the ability of the TDO to lower RDP and minimize stress degrades slightly with increasing churn. Moreover, the degradation is more prominent in the presence of tighter degree constraints. The reason for the slight degradation (in RDP and stress) is that during

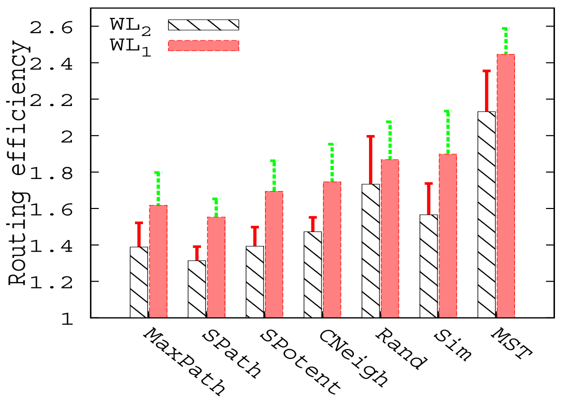


Figure 4.9: Routing efficiency of core selection strategies.

the experiments only the peers joining and leaving the system (as a result of churn) are allowed to perform random walk to find target peers and improve their position in the TDO. The random walks by the rest of the peers are not permitted to simulate conservative conditions and hence, overtime the organization of peers in the TDO does not resemble their positions in the underlay topology accurately resulting in higher RDP and stress. Further experiments show that if other peers are also allowed to perform random walks periodically the degradation can be avoided.

4.6.3 Performance of Event Routing

We evaluate three aspects of our event routing approach: i) capability of the proposed core selection strategies to reduce cost of event dissemination (routing cost) in comparison to the baseline approaches, ii) impact of increase in number of cores on the routing cost and the control overhead and, iii) adaptability to dynamic changes in the workload.

We compare our work against three baseline approaches: i) random (*Rand*), ii) similarity-based (*Sim*), and iii) delay-based (*MST*). The first two are core-based approaches. Rand selects k_C random peers as cores, whereas Sim selects cores solely based on the similarity between the peers to produce/consume similar events. The third approach MST maintains a minimum delay spanning tree for the routing of events. Moreover, we implement an optimal routing (*OPT*) algorithm that uses separate shortest path routing trees for the dissemination of each published event [MSRS09].

4.6.3.1 Core Selection Strategies

Figure 4.9 shows the routing efficiency of the proposed core selection strategies and the baseline approaches in comparison to OPT. More precisely, the efficiency of a routing approach (or a strategy) is defined as the ratio of the event routing cost (cf. Section 4.2) incurred by the approach (or the strategy) to the cost of routing events using OPT. To measure the routing cost around 2000 events are disseminated in the system. Clearly, a lower value for routing efficiency means better approach. The figure depicts that SPath performs better than all other strategies mainly because it uses the global knowledge. However, interestingly SPotent strategy which only exploits the local knowledge to select cores, performs almost similar to the global knowledge based MaxPath strategy. Moreover, all the approaches perform consistently better in case the subscription/event workload follows interest popularity model (i.e., WL_2). This is because the subscriptions associated with an interest hotspot consume similar events and placing such subscriptions nearby in the routing tree limits event dissemination to a certain region saving routing cost, whereas for uniformly distributed workload the improvement from nearby placement of the subscriptions consuming similar events is relatively small. For the same reason, Sim achieves better efficiency than Rand for zipfian workload (WL_2), whereas both Sim and Rand perform almost same in the case of uniform workload (WL_1). Another interesting observation is that MST performs even worse than Rand. This is because MST only considers the TDO links with lowest delays, while Rand additionally takes into account the even traffic consumed by the peers (for connecting them to the closest cores) to further reduce the cost of event dissemination.

4.6.3.2 Impact of Core Size

Figure 4.10(a) depicts the influence of the number of cores on the communication cost to forward subscriptions as well as route events from the publishers to the interested subscribers. The trend shows that the cost increases with the increase in the number of cores. This is because all the cores in $\mathcal{C}$ are connected with each other through the shortest weighted path tree rooted at the core c_h with the highest votes (cf. Section 4.5.2). The SPT_{c_h} may not represent shortest (weighted) paths between all pairs of cores in $\mathcal{C}$ and therefore, the communication (such as event dissemination or subscription forwarding) between two peers p and q with different relays (cores) incurs higher cost due to higher weight (delay and dissimilarity in event traffic) on the path between $rel(p)$ and $rel(q)$. Certainly, the communication cost can be decreased, if the cores are connected with each other through shortest weighted paths. Figure 4.10(b) illustrates this point by showing that in the presence of APSP (All pairs shortest weighted paths) between the cores, the communication cost decreases significantly with the increase in the number of cores. Clearly, this reduction in cost comes at the expense of additional control overhead due to the maintenance of $k_{\mathcal{C}}$ shortest weighted path trees. It is worth mentioning that the cores represent only a small fraction of peers in the system (i.e., $k_{\mathcal{C}} \ll N$) and therefore, control overhead is much lower in comparison to the maintenance of APSP w.r.t. all peers in the system. Figure 4.10(c) depicts the control efficiency of the proposed core selection strategies w.r.t. different

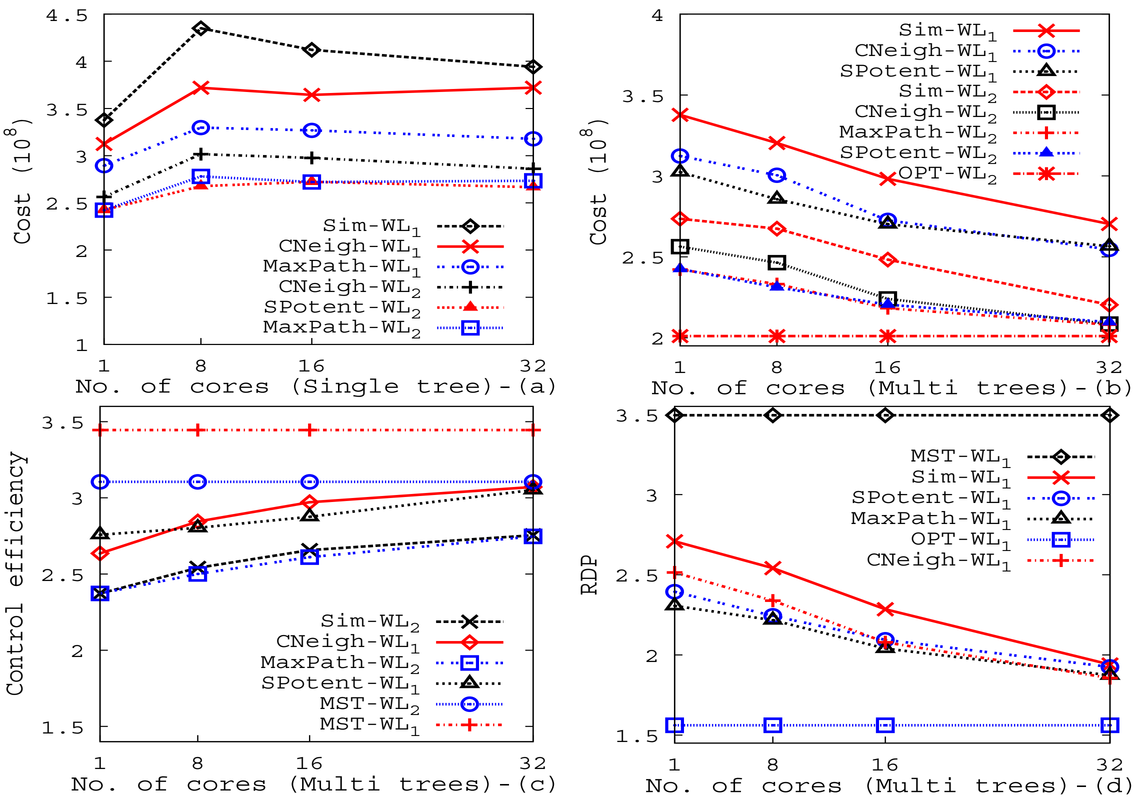


Figure 4.10: Influence of varying number of cores.

core sizes. The control efficiency is defined as the ratio of the complete message traffic generated in the system to the number of matching events received by the peers (i.e., events which are matched by the subscriptions of the peers). The traffic includes all the control overhead induced in the system during leader election, selection of cores, maintenance of SPTs to connect peers and cores in the routing overlay, subscription forwarding and event dissemination. The control efficiency of 1.0 indicates ideal system, whereby events are directly delivered to all the peers with matching subscriptions without incurring any unnecessary message overhead. Figure 4.10(c) shows that the control efficiency decreases (or in other words more control overhead is generated) with the increase in core size, mainly due to the maintenance of k_C shortest path trees. However, the control efficiency of the proposed core selection strategies is still better than MST approach. This is because MST approach does not take into account the event traffic consumed/produced by the peers during the construction of routing overlay and therefore, experiences high control overhead during subscription forwarding and event dissemination. Figure 4.10(d) depicts that RDP experienced by the routing overlay lowers with the increase in core size, as expected. Interestingly, MST approach although maintaining low delay links experiences significantly high RDP e.g., 93% increase in RDP in comparison to CNeigh strategies with 32 cores. This is due to the

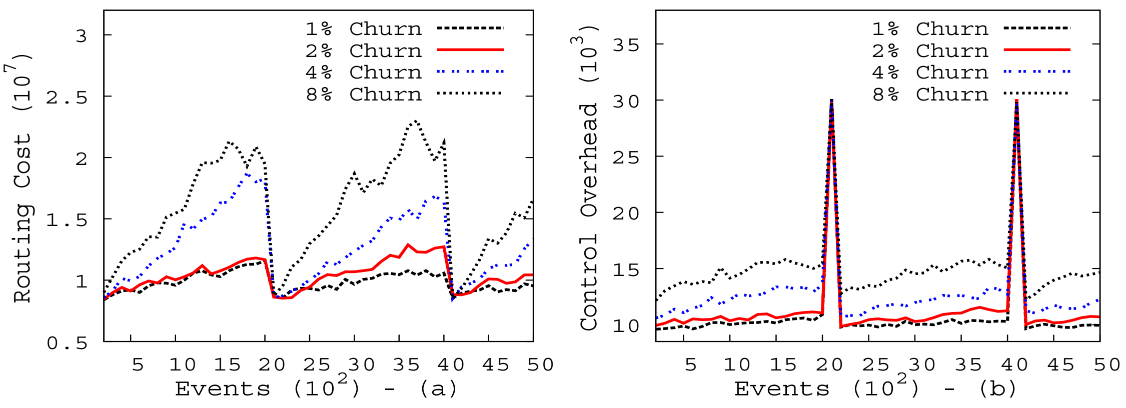


Figure 4.11: Adaptability of routing overlay to continuous churn.

fact that the selection of minimum delay links alone does not guarantee low end-to-end delays between the peers [WC04]. Finally, Figures 4.10(b) and (d) show that SPotent outperforms CNeigh for smaller number of cores, while the later performs slightly better with the increase in core size.

4.6.3.3 Adaptability to the Changes in Workload

Figures 4.11(a) and (b) show the behaviour of the system in the presence of continuously arriving and leaving subscribers. The churn is introduced in the system after the dissemination of every 100 events. The percentage of churn is relative to the total number of peers in the system, as described in Section 4.6.2. During the experiment, SPotent is used as the core selection strategy and subscriptions/events workload is generated using zipfian distribution (WL_2). Nevertheless, similar trends are observed for other core selection strategies and uniformly distributed subscriptions/events workload (WL_1).

Figure 4.11(a) shows that the cost of event routing increases as more and more events (up to 2000) are disseminated in the system. This is because the joins and leaves of peers (due to churn) are only handled locally to reduce the control overhead (cf. Section 4.5.3) and as a consequence, the routing overlay degrades overtime resulting in higher routing cost. Figure 4.11(b) shows the control overhead in terms of the number of overall messages in the system. A slight increase in the control overhead for higher percentages of churn is due to the local handling of peer dynamics, as mentioned above. After every 2000 events, complete recalculation of the routing overlay is initiated by the leader, which significantly lowers the routing cost, as shown in Figure 4.11(a). Moreover, Figure 4.11(b) depicts that immediately after the leader initiated recalculation of the routing overlay, the control overhead rises considerably for a small transient period. However, the new routing overlay obtained after the transient period consumes almost identical control traffic.

4.7 Related Work

The related work to our contributions in this chapter can be classified into three areas: i) publish/subscribe systems, ii) topology-aware overlay networks, and iii) spanning tree optimizations in graph theory.

4.7.1 Publish/Subscribe Systems

In the recent past, several content-based publish/subscribe systems have been proposed with the aim to provide communication efficient routing of events from the publishers to the subscribers. Many of these systems [BFPB10, VRKS06, LHLJ12, GaR03] focus on minimizing bandwidth usage by clustering subscribers according to their interests, without taking into account the properties of the underlying physical network. As shown in Example 1, these systems could be suboptimal with respect to the communication overhead and end-to-end delays. We omit the details of these clustering systems here, since they have been discussed comprehensively in the Section 3.7. Few systems [MSRS09, JPMH07, PAc$^+$06, YAY11] consider underlay related QoS metrics such as end-to-end delay, data rate, loss rate etc., to optimize publish/subscribe overlay for efficient routing of events.

Jaeger et al. [JPMH07] address organization of brokers in an overlay network to achieve better (event routing) efficiency on both, the content-based and the network layer. They prove that finding the optimal broker network is NP-hard even in static settings and develop a distributed heuristic that continuously adjusts the position of brokers in an overlay network based on the i) communication cost of the underlay links, ii) processing cost of the brokers, and iii) event traffic matched between the brokers. The heuristic relies on the local knowledge available to each broker and runs in two phases. In the *evaluation phase*, each broker independently decides about the possibility of adding or removing a link in the overlay network, whereas the *consensus phase* is required to implement the decisions made during the evaluation phase in a distributed manner. The proposed heuristic however, assumes that the brokers cannot fail or leave the system, e.g., during the consensus phase, which is not true for P2P-based environments.

Majumder et al. [MSRS09] propose an approach that constructs multiple trees to distributed events in a content-based publish/subscribe system, in an efficient and scalable manner. Subscriptions are partitioned into groups by using a greedy clustering strategy and a separate tree is maintained for each subscription group. The construction of a tree is formulated as a generalized Steiner tree problem and an approximation algorithm is developed to build trees with communication cost at most poly-logarithmic factor of the optimum. However, the proposed approach assumes the availability of the content-based workload (subscriptions and matched events) and the properties of the underlying network at a central coordinator, which hinders its applicability in P2P-based publish/subscribe systems.

Similarly, XPort [PAc+06] targets the construction of an event distribution tree that can be optimized according to the application-defined metrics. Applications specify optimization metrics as a combination of network and processing costs, e.g., minimize average path delay to the root. XPort develops a two level aggregation framework to calculate the (event distribution or routing) cost of the tree for a given optimization metric. The first level computes the local (network and/or processing) cost of each node (or broker) in the system. The second level calculates the overall cost of the tree as an aggregation e.g., sum, average, min, variance etc., of the local node costs. Subsequently, XPort provides a set of transformation rules, which iteratively adapts the structure of the tree to converge to a minimal cost configuration. The transformation rules however, can only work for a single source publish/subscribe system.

Finally, some existing publish/subscribe systems [SB07,SDB08,ECG09] address reliable delivery of events by explicitly taking into account the router-level topology of the underlying network. Generally, these systems rely on the redundancy in the underlay paths between publishers and subscribers to provide resilience against the network failures. In contrast to our work, these systems assume that the topology information is somehow available. Moreover, the QoS metrics such as delay and bandwidth which are focused in this chapter are not addressed.

4.7.2 Topology-aware Overlay Networks

Previous research in the area of *Application Layer Multicast* (ALM) has shown that the knowledge of the underlying (router-level) network topology is beneficial to achieve low physical link stress, low RDP and high bandwidth data dissemination [JTC08a, JYCW07, JWC05, KF02, CN05, PWF07, ZL08]. Likewise, Abboud et al. [AKG+09] highlight the impact of underlay-awareness on different performance aspects of P2P overlay networks.

Zhu et al. [ZLP09] address the problem of constructing a high bandwidth overlay for ALM. The links in an overlay network are usually correlated and share the capacities (bandwidth) of the underlying physical links. The authors introduce an overlay model (named as linear capacity constraints – LCC) that incorporates the capacity correlations between the overlay links to identify hidden shared bottlenecks [ZL08]. They prove that the problem of constructing a high bandwidth tree using LCC model is NP-hard and propose a distributed heuristic, which incrementally improves the bandwidth of the ALM tree by replacing the lower bandwidth overlay links with the higher bandwidth links.

Similarly, Jin el al. [JYCW07] also target the problem of maximizing the bandwidth for P2P video streaming. In particular, two closely related problems are addressed, i.e., the maximum bandwidth multicast tree (MBMT) and the minimum stress multicast tree (MSMT). The authors prove that both the problems are NP-hard and propose approximation algorithms. However, the approximation algorithms are centralized and assume the availability of the complete knowledge about the router-level topology of the underlying network.

FAT (FAST Application-layer Tree) [JWC05] uses underlay route inference tools such as traceroute, to discover router-level underlay topology and build a multicast tree on top of discovered topology to achieve high bandwidth and low RDP. A heuristic, named Max-Delta, is employed to discover the underlay topology in an efficient and scalable manner. The Max-Delta heuristic utilizes network coordinate systems such as Vivaldi [DCKM04], Lighthouses [PCW$^+$03] or ICS [LHC03], to detect undiscovered links in the underlay and thereby avoids unnecessary overhead by performing only those traceroute measurements, which increase the accuracy of the discovered underlay topology, i.e., increase the discovery of new underlay links. For its correct functioning, Max-Delta heuristic requires global knowledge about the already discovered underlay topology and the network coordinates of the peers, which hinders its scalability to large scenarios. MLink heuristic proposed in this chapter is adopted from the Max-Delta heuristic. In particular, we modified the Max-Delta to operate in a completely decentralized environment and without the use of network coordinates.

4.7.3 Graph Theory

A huge amount of graph theory literature is available on spanning tree related optimization problems [WC04] such as Minimum Routing Cost Spanning Tree (MRCT) [CR08] or Optimum Communication Spanning Tree (OCT) [Hu74]. The event routing problem (cf. Section 4.2) addressed in this chapter is a generalization of MRCT problem and can be mapped to an OCT problem.

The most comprehensive work in this context is performed by Wu el al. [WCT00b, Wu02, WCT00a, Wu04]. The authors develop several approximation algorithms for MRCT and OCT problems. However, the main motivation of their work is to prove the existence of several approximation ratios, rather than to develop efficient algorithms.

Wong et al. [Won80] develop a 2-approximation algorithm for the MRCT problem. The Wong's algorithm computes shortest path tree for all vertices (or nodes) in the input graph and selects the tree with the lowest cost as MRCT. Similarly, Grout et al. [Gro05] propose a greedy algorithm for building the MRCT that minimizes the number of relay vertices by promoting the vertices with higher degree (higher adjacent edges) to be included in the spanning tree. The Grout's algorithm however, only gives good result for homogeneous graphs where all the edges have unit distance (or weight). Most recently, Campos et al. [CR08] propose a centralized heuristic, which modifies the Prim's well known minimum spanning tree (MST) algorithm to construct MRCT. The Prim's algorithm uses only distances between the vertices in the graph to construct

MST. The authors change the Prim's algorithm to use three parameters, i) degree of a vertex, ii) sum of adjacent edge distances and iii) maximum adjacent edge distance, for the selection of vertices to be included in the spanning tree.

Nevertheless, all these MRCT approaches cannot be applied for event routing in a content-based publish/subscribe system, because their focus is to minimize the pairwise distances between the vertices in the input graph without any consideration to the traffic requirements between those vertices. Moreover, these approaches are centralized and are not targeted to handle continuously evolving workload as is the case in P2P-based systems.

4.8 Summary

In this chapter, we have presented an approach that exploits the knowledge of event traffic, user subscriptions and the router-level topology of the underlying physical network to achieve scalable and communication efficient dissemination of events in a content-based publish/subscribe system.

For this purpose, we have developed methods to discover underlay topology between subscribers and publishers in the system. To reduce discovery overhead and maintain topology information in a distributed manner, our proposed methods construct a *Topology Discovery Overlay* (TDO), whereby peers are connected according to the overlapping in the underlay routes.

Afterwards, the information of the discovered topology and the proximity between the peer to receive or produce similar events is used to build publish/subscribe routing overlay. In particular, we have proposed different core selection strategies (exploiting global and local knowledge) to facilitate the construction of a communication efficient event routing overlay on top of TDO.

The evaluations show that for Internet-like topologies, the proposed topology discovery methods are capable of modelling an underlay in an efficient and accurate manner. For instance, the baseline approach which does not take into account the underlay topology experiences up to 39% rise in RDP and 30% higher stress on the physical links. Moreover, the proposed core-based approach yields a significant reduction in event routing cost in comparison to the state of the art. For example, the routing cost is lowered up to 34% and 30% for the SPotent and the CNeigh strategies with single cores respectively, as compared to the widely used minimum spanning tree (MST) based routing approach. Nevertheless, the routing cost is further decreased up to 47% and 49% for the two respective strategies (SPotent and CNeigh) in the presence of 32 cores.

Chapter 5

Security

5.1 Introduction

In this chapter, we focus on providing the basis security mechanisms such as access control, confidentiality and integrity in a content-based publish/subscribe system.

Access control in the context of publish/subscribe system means that only authenticated publishers are allowed to disseminate events in the network and those events are only delivered to authorized subscribers. Similarly, the content of events should not be exposed to the routing infrastructure and a subscriber should receive all relevant events without revealing its subscription to the system. These security issues are not trivial to solve in a content-based publish/subscribe system and new mechanisms are needed to route encrypted events to subscribers without knowing their subscriptions and to allow subscribers and publishers authenticate each other without knowing each other.

In the past, most research has focused on providing expressive and scalable publish/subscribe systems. So far little attention has been given to security issues. Existing approaches towards secure publish/subscribe systems mostly rely on the presence of a traditional static broker network [RR06, BESP08, IRC10, SLI11, CGB10, PEB07, NSB12]. These either address security under restricted expressiveness by providing security for topic-based publish/subscribe [SL05, SÖM09] or rely on a network of trusted brokers [Pie04, OP01, PEB07, Khu05]. Few recent systems target security in untrusted broker environments [SLI11, CGB10]. However, as mentioned by Nabeel et al. [NSB12], these support weaker security and provide inaccurate event routing due to limited ability of brokers to perform content-based matching. Furthermore, existing approaches use coarse grain epoch-based key management and cannot provide fine grain access control in a scalable manner [SL05, RR06, SL07]. Nevertheless, security in broker-less publish/subscribe systems, where the subscribers are clustered according to their subscriptions have not been discussed yet in literature.

In this chapter, we present a new approach to provide authentication, confidentiality and integrity in a broker-less publish/subscribe system [TKAR10]. Our approach allows subscribers to maintain credentials according to their subscriptions. Private keys assigned to the subscribers are labelled with the credentials. A publisher associates each encrypted event with a set of credentials. We adapted Identity-based encryption mechanisms [BSW07, GPSW06], i) to ensure that a particular subscriber can decrypt an event only if there is a match between the credentials associated with the event and the key and, ii) to allow subscribers to verify the authenticity of received events. Furthermore, we address the issue of subscription confidentiality in the presence of semantic clustering of subscribers. A weaker notion of subscription confidentiality is defined and a secure overlay maintenance protocol is designed to preserve the weak subscription confidentiality. Finally, the evaluations demonstrate the viability of the proposed security mechanisms.

5.2 System Model and Background

5.2.1 Content-based Publish/Subscribe

For the routing of events from publishers to the relevant subscribers we use the content-based data model, which is similar to that defined in Section 2.2. The *event space*, denoted by Ω, is composed of a global ordered set of d distinct attributes (A_i): $\Omega = \{A_1, A_2, \ldots, A_d\}$. Each *attribute* A_i is characterized by a unique *name*, its *data type* and its *domain*. The data type can be any ordered type such as integer, floating point and character strings. The domain describes the range $[L_i, U_i]$ of possible attribute values. A *subscription filter* f is a conjunction of predicates, i.e., $f = \{Pred_1 \wedge Pred_2 ... \wedge Pred_j\}$. $Pred_i$ is defined as a tuple (A_i, Op_i, v_i), where Op_i denotes an operator and v_i a value. The operator Op_i typically includes equality and range operations for numeric attributes and prefix/suffix operations for strings. An *event* consists of attributes and associated values. An event is *matched* against a subscription f (and subsequently delivered to the subscriber), if and only if the values of attributes in the event satisfy the corresponding constraints imposed by the subscription. The subscription containment relationship can be defined similar to Section 3.2. Let E_{f_1} and E_{f_2} denote the sets of events matching the subscriptions f_1 and f_2, respectively. Then f_1 is said to be *covered* by the subscription f_2, iff $E_{f_1} \subseteq E_{f_2}$ holds.

We consider publish/subscribe in a setting where there exists no dedicated broker infrastructure. Publishers and subscribers contribute as peers to the maintenance of a self-organizing overlay structure. Peers can join the overlay by contacting an arbitrary peer and thereafter subscribe and publish events. In order to authenticate publishers we use the concept of *advertisements* in which a publisher announces beforehand the set of events which it intends to publish.

5.2.2 Attacker Model

Our attacker model is similar to the commonly used *honest-but-curious* model [SL06, SL07, SÖM09]. There are two types of entities in the system: publishers and subscribers. Both types of entities are computationally bounded. Moreover, all the peers (publishers or subscribers) participating in the publish/subscribe overlay network are honest and do not deviate from the designed protocol, i.e., they route the events according to the protocol and do not drop events or forward them in a wrong manner [SÖM09]. Likewise, authorized publishers only disseminate valid events in the system. However, malicious publishers may masquerade the authorized publishers and spam the overlay network with fake and duplicate events. We do not intend to solve the digital copyright problem. Therefore we assume that authorized subscribers do not reveal the content of successfully decrypted events to other subscribers.

Subscribers are however curious to discover the subscriptions of other subscribers and published events to which they are not authorized to subscribe. Similarly, curious publishers may be interested to read events published in the system. Furthermore, passive attackers outside the publish/subscribe overlay network can eavesdrop the communication and try to discover content of events and subscriptions.

Finally, we assume presence of secure channels for the distribution of keys from the key server to the publishers and subscribers. A secure channel can be easily realized by using transport layer mechanisms such as *Transport Layer Security* (TLS) [DR08, TP11] or *Secure Socket Layer* (SSL) [FKK11]. No other requirement is placed on the underlying network infrastructure regarding confidentiality, authenticity or integrity of exchanged data.

5.2.3 Security Goals and Requirements

The proposed secure publish/subscribe system has fundamentally two set of goals, i.e., security goals and scalability goals.

Authentication of Publishers and Subscribers: In order to avoid non-eligible publications only authorized publishers should be able to publish events in the system. Similarly, subscribers should only receive those messages to which they are authorized to subscribe.

Confidentiality and Integrity of Events: The events should only be visible to authorized subscribers and should be protected from illegal modifications.

Confidentiality of Subscriptions: The authorized subscribers should receive events matching their interests without revealing their subscriptions. Subscription integrity mandates that a subscriber should receive all those events which match its subscription and which are published by authorized publishers. In this chapter, we do not intend to solve event drop-based denial of service (DoS) attacks and therefore, we only focus on providing subscription confidentiality.

Apart from the security goals, the secure publish/subscribe system should scale with the number of subscribers in the system. Three aspects are important to preserve scalability: i) the number of keys to be managed and the cost of subscription should be independent of the number of subscribers in the system, ii) the key server and subscribers should maintain small and constant numbers of keys per subscription, iii) the re-keying overhead should be minimized without compromising the fine grained access control.

5.2.4 Drawbacks of Traditional Security Mechanisms

Authentication using existing public key encryption mechanisms such as PKI allows a party to encrypt data to a particular user. Senders and receivers are strongly coupled, i.e., before a sender can encrypt a message, the receiver must generate a public/private key pair, sign its public key by a certificate authority and communicate it to the sender. Furthermore, PKI is inefficient for a large number of subscribers as each event needs to be encrypted with each subscriber's individual public key. Therefore, a mechanism is needed to enable any pair of users to securely communicate and verify each other's signatures without the need to exchange private or public keys.

It is very hard to provide subscription confidentiality in a broker-less publish/subscribe system, where the subscribers are arranged in an overlay network according to the containment relationship between their subscriptions. In this case, regardless of the cryptographic primitives used, the maximum level of attainable confidentiality is very limited. The limitation arises from the fact that a parent can decrypt every event it forwarded to its children. Therefore, mechanisms are needed to provide a weaker notion of confidentiality.

5.2.5 Identity-based Encryption

While a traditional PKI infrastructure requires to maintain for each identity a private/public key pair which has to be known between communicating entities to encrypt and decrypt messages, *Identity-based encryption* [Sha84, BF03] provides a promising alternative to reduce the amount of keys to be managed.

In Identity-based encryption (IBE), any valid string which uniquely identifies a user can be the public key of the user. A key server maintains a single pair of public and private master keys. The master public key can be used by the sender to encrypt and send the messages to a user with any identity, e.g., an email address. To successfully decrypt the message, a receiver needs to obtain a private key for its identity from the key server. Figure 5.1 shows the basic idea of using Identity-based encryption.

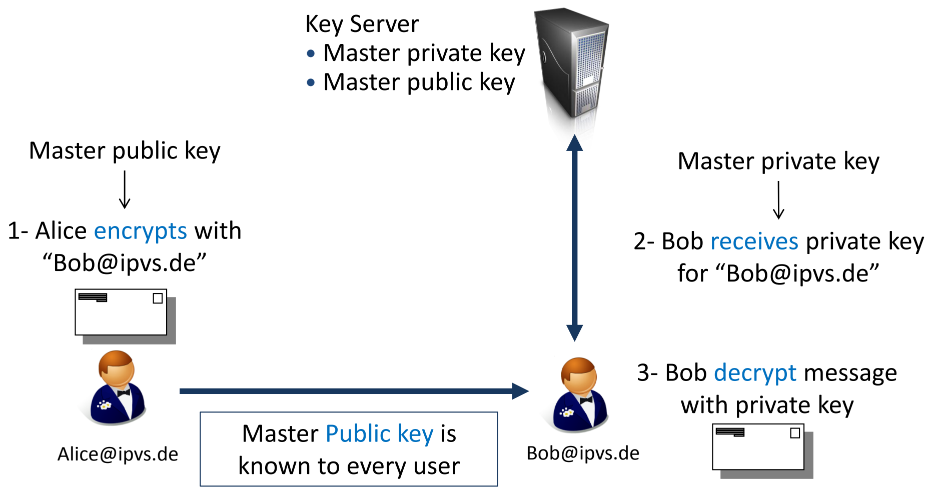

Key Server
• Master private key
• Master public key
Master public key
1- Alice encrypts with "Bob@ipvs.de"
Master private key
2- Bob receives private key for "Bob@ipvs.de"
3- Bob decrypt message with private key
Alice@ipvs.de
Master Public key is known to every user
Bob@ipvs.de

Figure 5.1: Identity-based encryption.

We want to stress here that although Identity-based encryption at the first glance, appears like a highly centralized solution, its properties are ideal for highly distributed applications. A sender needs to know only a single master public key in order to communicate with any identity. Similarly, a receiver only obtains private keys for its own identities. Furthermore, an instance of central key server can be easily replicated within the network. The replicas can function independently without having to interact with each other. This enables key servers to be created on demand for load balancing and reliability. Finally, a key server maintains only a single pair of master keys and therefore, can be realized as a smart card, provided to each participant of the system. It is interesting to note that in case of replicated key servers, changing master key pair requires one secure transfer to each replica and therefore, the cost is in order of the number of replicas. Smart cards on the other hand should be physically replaced. However their use is still practical as the master key pair is usually secure and is seldom changed.

Although Identity-based encryption has been proposed in 1984 [Sha84], only recently *pairing-based cryptography* has laid the foundation of a practical implementation of *Identity-based* encryption [BF01]. The main idea behind pairing-based cryptography is to establish a mapping between two cryptographic groups. This allows reduction of one problem in one group to a different usually easier problem in another group. The mapping between cryptographic groups is achieved by means of *bilinear maps*, a technique we also apply subsequently for establishing the basic security mechanisms in the publish/subscribe system and therefore, introduce here the main properties.

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be cyclic group of order q, where q is some large prime. A bilinear map is a function $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ that associates a pair of elements from $\mathbb{G}_1$ to elements in $\mathbb{G}_2$. A bilinear map satisfies the following conditions:

1. *Bilinearity:* $\hat{e}(u^x, v^y) = \hat{e}(u^y, v^x) = \hat{e}(u, v)^{xy}$, for all $u, v \in \mathbb{G}_1$ and $x, y \in \mathbb{Z}$.
2. *Non-degeneracy:* $\hat{e}(u, v) \neq 1$, for all $u, v \in \mathbb{G}_1$.
3. *Computability:* $\hat{e}$ can be efficiently computed.

Suitable bilinear maps can be constructed from elliptic curves such as $y^2 = x^3 + 1$ over finite field F_q. The following properties of bilinear maps can be easily verified. For all $u, v, h \in \mathbb{G}_1$ and $x, y, a, b \in \mathbb{Z}$.

- $\hat{e}(u, v)^x.\hat{e}(u, v)^y = \hat{e}(u, v)^{x+y}$
- $\hat{e}(u, v)^{a+b} = \hat{e}(u, v)^a.\hat{e}(u, v)^b$
- $\hat{e}(u.h, v) = \hat{e}(u, v).\hat{e}(h, v)$

5.3 Approach Overview

For providing security mechanisms in publish/subscribe we rely on the methods of Identity-based encryption and adapt it in order to support many-to-many interactions between subscribers and publishers. Publishers and subscribers interact with a key server by providing *credentials* to the key server. In turn they receive keys which fit the expressed capabilities in the credentials and thus can subsequently be used to decrypt and sign relevant messages in the content-based publish/subscribe system. In this case we say the credential is *authorized* by the key server.

Consequently, a credential consists of two parts: first a binary string which describes the capability of a peer in publishing and receiving events, and second a proof of its identity. The latter is used for authentication against the key server and verification whether the capabilities match the identity of the peer. While this can happen in a variety of ways, e.g., relying on a challenge/response scheme, hardware support, etc., we pay attention mainly at expressing the capabilities of a credential, i.e., how subscribers and publishers can *create* a credential. This process needs to account for the many possibilities to partition the set of events expressed by an advertisement or subscription and exploits overlaps in subscriptions and publications. Subsequently, we use the term credential only for referring to the capability string of a credential.

The keys allotted to publishers and subscribers, and the ciphertexts are assigned (or encoded with) credentials. In particular, the Identity-based encryption ensures that a particular key can decrypt a particular ciphertext only if there is a match between the credentials of the ciphertext and the key. Publishers and subscribers maintain separate private keys for each authorized credential.

The public keys are generated by a string concatenation of an credential, an epoch for key revocation, a symbol $\in \{SUB, PUB\}$ distinguishing publishers from subscribers, and some additional parameters described in Section 5.5. The public keys can be easily

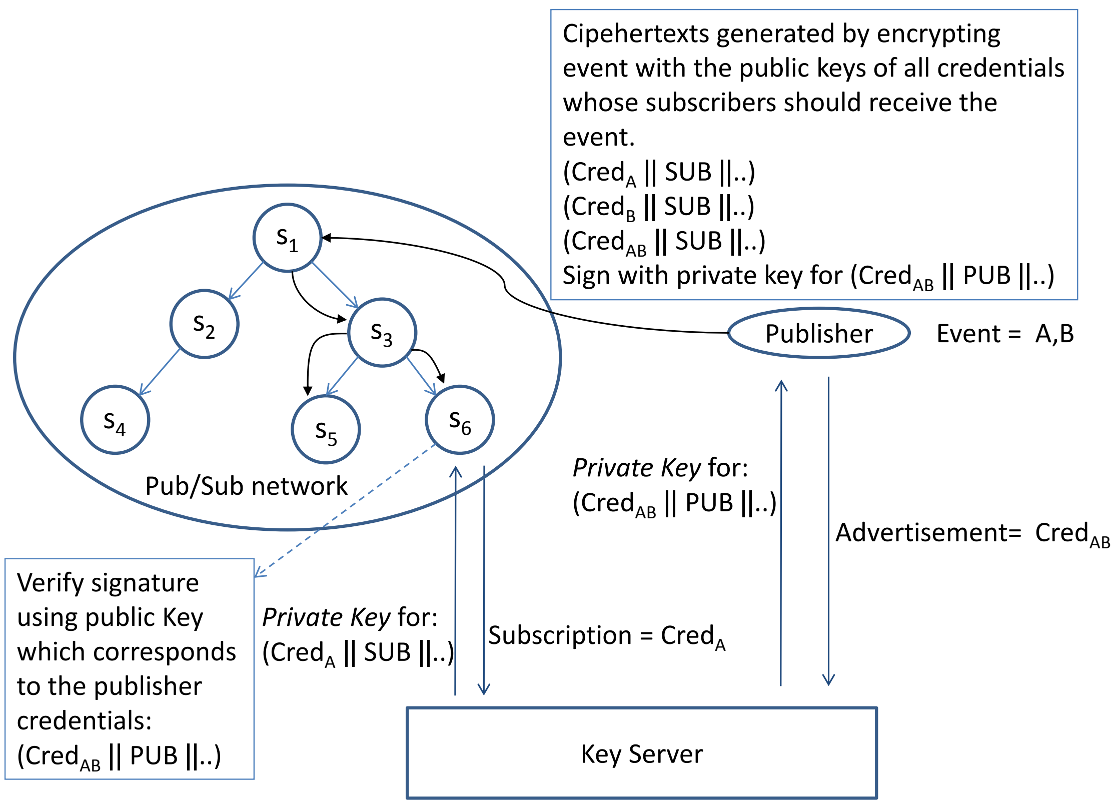

Cipehertexts generated by encrypting event with the public keys of all credentials whose subscribers should receive the event.
(Cred$_A$ || SUB ||..)
(Cred$_B$ || SUB ||..)
(Cred$_{AB}$ || SUB ||..)
Sign with private key for (Cred$_{AB}$ || PUB ||..)
s_1
s_2
s_3
s_4
s_5
s_6
Pub/Sub network
Publisher
Event = A,B
Private Key for: (Cred$_{AB}$ || PUB ||..)
Advertisement= Cred$_{AB}$
Verify signature using public Key which corresponds to the publisher credentials: (Cred$_{AB}$ || PUB ||..)
Private Key for: (Cred$_A$ || SUB ||..)
Subscription = Cred$_A$
Key Server

Figure 5.2: Approach overview: Publisher has credentials to publish events with two attributes A and B, Subscriber s_6 has credentials to receive events with attribute A.

generated by any peer without contacting the key server or other peers in the system.[*] Similarly, encryption of events and their verification using public keys do not require any interaction.

Due to the loose coupling between publishers and subscribers, a publisher does not know the set of relevant subscribers in the system. Therefore, a published event is encrypted with the public key of all possible credentials, which authorizes a subscriber to successfully decrypt the event. The ciphertexts of the encrypted event are then signed with the private key of the publisher, as shown in Figure 5.2.

The overlay network is organized according to the containment relationship between the subscriptions. Subscribers with coarser subscriptions are placed near the root and forward events to the subscribers with less coarser subscriptions. To maintain such a topology each subscriber should know the subscription of its parent and child peers. When a new subscriber arrives, it sends the connection request along with its subscrip-

[*]The authenticity of the public keys is guaranteed implicitly as long as the transport of the private keys to the corresponding users are kept secure.

tion to a random peer in the overlay network. The connection request is forwarded by possibly many peers in the overlay network before it reaches the right peer to connect. Each forwarding peer matches the subscription in the request with the subscription of its parent and child peers to decide the forwarding direction. Maintaining a relationship between subscriptions clearly contradicts subscription confidentiality. Therefore, we proposed an approach to ensure a weaker notion of subscription confidentiality in Section 5.6.

This section has introduced only the main steps in establishing security. The subsequent sections discuss i) how to create credentials systematically to support scalability (cf. Section 5.4), ii) how the authentication of publishers and subscribers as well as confidentiality and integrity of events is achieved (cf. Section 5.5), and iii) how the publish/subscribe overlay is maintained and a weak notion of subscription confidentiality is preserved (cf. Section 5.6).

5.4 Creation of Credentials

In the following, we will first describe the creation of credentials for numeric and string attributes. Further extensions to handle complex subscriptions are discussed subsequently.

5.4.1 Numeric Attributes

The event space, composed of d distinct numeric attributes can be geometrically modelled as a d-dimensional space such that each attribute represents a dimension in the space (cf. Section 2.2). Subscriptions and advertisements are represented by hyper-rectangles in the space, whereas published events represent points.

With the spatial indexing approach, the event space is hierarchically decomposed into regular sub-spaces, which serve as enclosing approximations for the subscriptions and advertisements, as detailed in Section 2.3. The decomposition procedure divides the domain of one dimension after the other and recursively starts over in the created sub-spaces. For example, a 2-dimensional event space is first divided in dimension d_1 into two sub-spaces, then in dimension d_2 into four sub-spaces, then again in dimension d_1 into eight sub-spaces and so on. In general, the number of finest granularity (smallest) sub-spaces created by decomposing a d-dimensional space is equal to $\prod_{i=1}^{d} \mathfrak{Z}_i$, where $\mathfrak{Z}_i = \frac{U_i - L_i}{Granularity(i)}$ and $Granularity(i)$ defines the smallest addressable value of the attribute A_i.

Subscription or advertisement of a peer can be composed of several sub-spaces. A credential is assigned for each of the mapped sub-space. An event can be approximated by the smallest (finest granularity) sub-space that encloses the point represented by it. To deliver the encrypted event a ciphertext must be generated for each sub-space that encloses the event so that the peer whose subscription mapped to any of these sub-spaces should be able to successfully decrypt the event. An event dz_e matches (is enclosed in) a sub-space dz_s, if dz_e is covered by dz_s. In general, the number of sub-spaces matched by an event dz_e is in the order of $\log_2(\prod_{i=1}^{d} \mathbb{Z}_i)$ and is equal to $|dz_e| + 1$. For example, an event 0010 is matched by the five sub-spaces $0010, 001, 00, 0$ and ϵ.

For an event space with a large set of numeric attributes, the number of mapped sub-spaces and therefore, credentials for a subscription can be very large. This affects the scalability of the system in terms of space required to store the keys, computational cost at the key server and communication cost between the key server and the peers. There are three possibilities to mitigate this problem. The first possibility is to allow a coarsening of individual subscriptions by mapping to a smaller set of bigger sub-spaces. For example, f_2 in Figure 2.2, originally mapped to the sub-spaces $\{000, 010\}$, can be coarsened by mapping to the sub-space 0 (cf. Section 2.3). The coarsened sub-space encloses more events and therefore, a peer can decrypt more events than authorized by the original subscription. The second possibility is to reduce the granularity of each attribute but it affects the expressiveness of all the subscriptions. For example, if the finest addressable value of a numeric attribute is 8 then range such as $[1, 5]$ is not possible.

A third and better possibility to reduce the number of credentials is to decompose the domain of each attribute into sub-spaces separately. The spatial indexing procedure is the same as above; however, in this case a separate decomposition tree is built for each attribute. Each peer receives credentials separately for each attribute in its subscription. The number of credentials maintained for each subscription or advertisement is bounded by $\sum_{i=1}^{d} \log_2(\mathbb{Z}_i)$. Similarly, the number of sub-spaces matched by an event are $\sum_{i=1}^{d} \log_2(\mathbb{Z}_i)$. In our system, we used the third approach.[†]

5.4.2 String Attributes

The above spatial indexing technique can work with any ordered data type with a known domain. String attributes usually have a maximum number of characters. This allows them to have known bounds. They can be linearised by hashing or other linearisation mechanisms and thus can also be indexed [TKK+11, MJ07].

Credentials for more expressive string operations such as prefix matching can be generated using a *trie*. A trie is an ordered data structure for storing strings in a way that allows fast string lookup and prefix matching. Each node in the trie is labelled

[†]Similar strategy is employed by Shi et al. [SBC+07] to provide multi-dimensional range queries over encrypted data. However, their cryptographic methods are more costly than those proposed in this chapter (cf. Section 5.8).

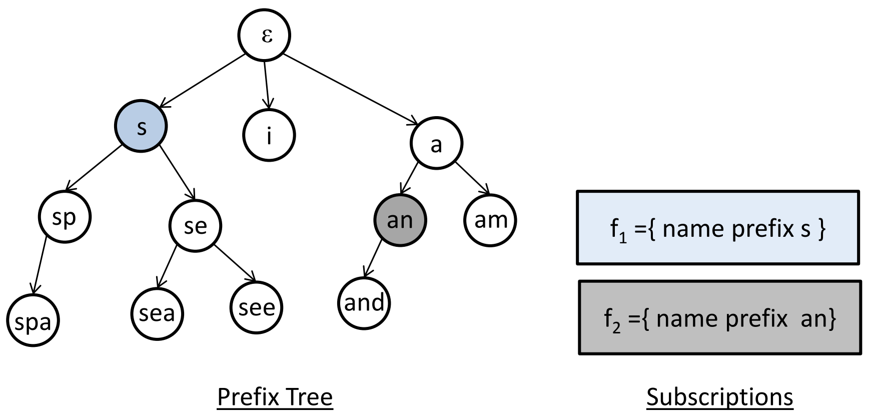


Figure 5.3: Prefix matching.

with a string, which serves as a common prefix to all its descendants, as shown in Figure 5.3. Each peer is assigned a single credential, which is the same as its subscription or advertisement. Events correspond to the leaf nodes of the trie. To deliver an encrypted event a ciphertext must be generated with the label of each node in the path from the leaf to the root of the trie, so that a peer whose subscription matches any of the labels should be able to successfully decrypt the event. In general, the number of nodes on the longest path from a leaf to the root of a trie associated with a string attribute A_i is equal to $\mathfrak{L}_i$, where $\mathfrak{L}_i$ is the length of the longest label assigned to a leaf node. Similar mechanism can be used to generate credentials for suffix matching. Likewise, credentials can also be generated for other attribute types such as concept hierarchies [SLI11], however, they will not be discussed in this chapter.

5.4.3 Complex Subscriptions

For a complex subscription with predicates on different attributes, a subscriber receives separate credentials and thus keys for each attribute. Using these keys, a subscriber should be able to successfully decrypt any event with the corresponding attributes if it is authorized to read the values associated with the attributes. Any cryptographic primitive can be easily used for this purpose. For example, similar to PSGuard [SL06], the keys for individual attributes can be XORed together and hashed to get the actual key for decrypting the event.

In a content-based publish/subscribe system, a subscription defines a conjunction on predicates. An event matches a subscription if and only if all of the predicates in the subscription are satisfied. To ensure event confidentiality, a subscriber must not be able to successfully decrypt any event which matches only parts of its subscriptions. However, assigning keys for individual attributes and XOR based decryption does not prevent this behaviour. For example, consider a subscriber with two subscriptions $f_1 = \{Area = [10, 20] \wedge location = Stuttgart\}$ and $f_2 = \{Area =$

$[40, 80] \wedge \mathit{location} = \mathit{London}\}$. If the credentials and therefore keys are assigned for individual attributes then the subscriber can also decrypt the events matching the subscriptions $f_3 = \{\mathit{Area} = [10, 20] \wedge \mathit{location} = \mathit{London}\}$ and $f_4 = \{\mathit{Area} = [40, 80] \wedge \mathit{location} = \mathit{Stuttgart}\}$. Although he is not authorized to read events matching the subscriptions f_3 and f_4. To properly ensure event confidentiality, all the keys associated with a subscription should be bound together, so that keys associated with different subscriptions should not be combined together.

5.5 Publisher/Subscriber Authentication and Event Confidentiality

In this section, we will describe the construction of security methods to achieve authentication of publishers and subscribers as well as confidentiality and integrity of events.

One naive solution would be to directly use the techniques from PKI by assigning public/private key pair to each credential. Publishers and subscribers can contact key server to obtain the public/private key pairs that corresponds to their credentials. However, PKI does not provide a mechanism to bind together the public/private key pairs associated with the same subscription (cf. Section 5.4.3) and therefore, cannot be used.

The security methods described in this section are built upon Attribute-based encryption [BSW07, GPSW06], which is a general form of Identity-based encryption. In particular, we adapted the Ciphertext-policy Attribute-based encryption (in short CP-ABE) scheme proposed by Benthencourt et al. [BSW07] to our requirements. More precisely, our modifications, i) allow publishers to sign and encrypt events at the same time by using the idea of Identity-based signcryption proposed by Yu et al. [YYSZ09], ii) enable efficient routing of encrypted events (from publishers to subscribers) by using the idea of searchable encryption proposed by Boneh et al. [BCOP04] and, iii) allow subscribers to verify the signatures associated with all the attributes (of an event) simultaneously. Our modifications do not change the basic structure of the CP-ABE scheme and preserve the same security strength, as discussed in Section 5.5.6.

5.5.1 Security Parameters and Initialization

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ denote the bilinear groups of prime order q, i.e., $|\mathbb{G}_1| = |\mathbb{G}_2| = q$, $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ denotes an admissible bilinear map and g denotes a generator in $\mathbb{G}_1$. Moreover, let $H_1 : \{0,1\}^* \rightarrow \{0,1\}^{n_u}$, $H_2 : \{0,1\}^* \rightarrow \{0,1\}^{n_m}$, $H_3 : \{0,1\}^* \rightarrow \mathbb{G}_1$, and $H_4 : \mathbb{G}_2 \rightarrow \{0,1\}^{\log q}$ designate collusion resistant cryptographic hash functions.

The initialization algorithm i) chooses $\alpha, \varphi \in \mathbb{Z}_q$, ii) computes $g_1 = g^\alpha$ and $h = g^\varphi$, iii) chooses $g_2, u', m' \in \mathbb{G}_1$, and iv) selects vectors $\vec{u} = (u_i)$ and $\vec{m} = (m_i)$ of length n_u and

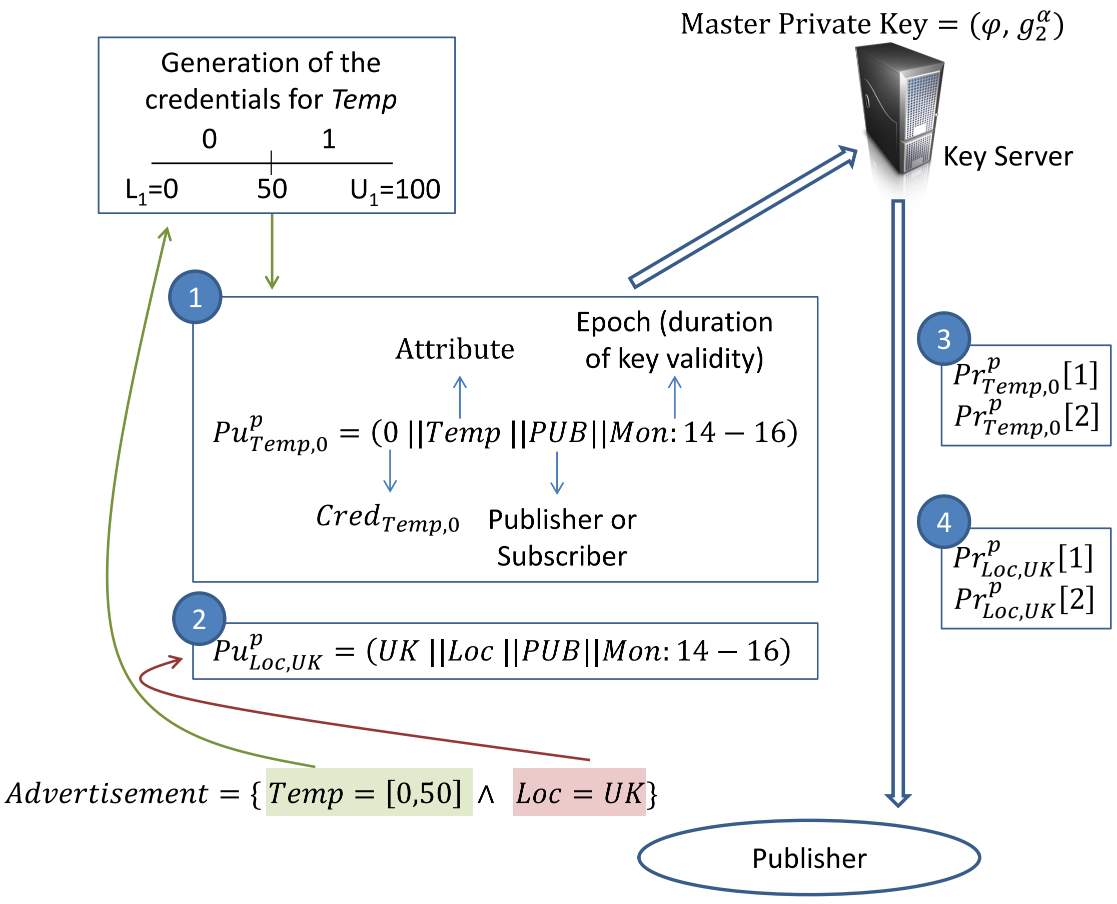

Master Private Key = (φ, g_2^α)
Key Server
Generation of the credentials for *Temp*
0
1
L_1=0
50
U_1=100
1
Attribute
Epoch (duration of key validity)
$Pu^p_{Temp,0} = (0 \,||Temp\, ||PUB||Mon: 14-16)$
$Cred_{Temp,0}$
Publisher or Subscriber
2
$Pu^p_{Loc,UK} = (UK\, ||Loc\, ||PUB||Mon: 14-16)$
3
$Pr^p_{Temp,0}[1]$
$Pr^p_{Temp,0}[2]$
4
$Pr^p_{Loc,UK}[1]$
$Pr^p_{Loc,UK}[2]$
$Advertisement = \{Temp = [0,50] \wedge Loc = UK\}$
Publisher

Figure 5.4: Key generation for publishers.

n_m respectively with every element chosen uniformly at random from $\mathbb{G}_1$. The *Master Public Key* MPu is published as:

$$\text{MPu} = (\hat{e}, g, g_1, g_2, h, u', m', \vec{u}, \vec{m})$$

This master public key is known to every peer in the system and is used for encryption and signature verification. The *Master Private key* MPr is (φ, g_2^α), and is only known to the key server. The master private key is used for generating private keys for publishers and subscribers.

5.5.2 Key Generation for Publishers

Before starting to publish events, a publisher contacts the key server along with the credentials for each attribute in its advertisement. If the publisher is allowed to publish events according to its credentials, the key server will generate separate private keys for each credential. Let $Cred_{i,j}$ denotes the credential with label j for the attribute A_i, e.g., $Cred_{Temp,0}$ represents credential 0 of attribute $Temp$, as shown in Figure 5.4.

Public Key: The public key of a publisher p for credential $Cred_{i,j}$ is generated as:

$$Pu^p_{i,j} := (Cred_{i,j} \ \ || \ A_i \ || \ PUB \ || \ Epoch)$$

where the symbol $PUB \in \{SUB, PUB\}$ indicates that the public key is associated with a publisher and the $Epoch$ specifies the time window of key validity. As mentioned later in Section 5.5.3, a different symbol SUB is used in the public keys for subscribers to ensure that the keys used for the verification of valid events (i.e., authentication of publishers) are different from the ones used to provide event confidentiality.

Private Keys: The key server will generate the corresponding private keys as follows. For each credential $Cred_{i,j}$ and a publisher p, let $v_p = H_1(Pu^p_{i,j})$ be a bit string of length n_u and let $v_p[k]$ denotes the kth bit. Let $\Gamma_{i,j} \subseteq \{1, 2, \dots, n_u\}$ be the set of all k for which $v_p[k] = 1$. The key server chooses $\gamma_{i,j} \in \mathbb{Z}_q$ at random and computes:

$$Pr^p_{i,j} := \left(g_2^\alpha \left(u' \prod_{k \in \Gamma_{i,j}} u_k \right)^{\gamma_{i,j}}, g^{\gamma_{i,j}} \right) =: \left(Pr^p_{i,j}[1], Pr^p_{i,j}[2] \right)$$

Figure 5.4 shows the generation of public and private keys for a publisher.

5.5.3 Key Generation for Subscribers

Similarly, to receive events matching its subscription, a subscriber contacts the key server and receives the private keys for the credentials associated with each attribute A_i.

Public Key: In case of subscribers, the public key for a credential $Cred_{i,j}$ is given as:

$$Pu^s_{i,j} := (Cred_{i,j} \ || \ A_i \ || \ SUB \ || \ Epoch)$$

Private Keys: The private keys are generated as follows. The key server chooses $\gamma_s \in \mathbb{Z}_q$ at random. The same γ_s is used for all credentials associated with a subscription. For each credential $Cred_{i,j}$, it calculates $\Gamma_{i,j}$ similar to the publisher's case (cf. Section 5.5.2), chooses $\gamma_{i,j} \in \mathbb{Z}_q$ and computes:

$$Pr^s_{i,j} := \left(g_2^{\gamma_s} \left(u' \prod_{k \in \Gamma_{i,j}} u_k \right)^{\gamma_{i,j}}, g^{\gamma_{i,j}}, H_3 \left(u' \prod_{k \in \Gamma_{i,j}} u_k \right)^{\varphi} \right) =: \left(Pr^s_{i,j}[1], Pr^s_{i,j}[2], Pr^s_{i,j}[3] \right)$$

Furthermore, a credential independent key $Pr^s[4] = g_2^{\frac{\gamma_s + \alpha}{\varphi}}$ is generated. Later we will see that γ_s along with $Pr^s[4]$ is needed to bind the keys/credentials of a subscription together.

It is worth mentioning that the key $Pr^s_{i,j}[3]$ (unlike other keys, i.e., $Pr^s_{i,j}[1]$, $Pr^s_{i,j}[2]$ and $Pr^s[4]$) is not used to decrypt events. In our construction, both the content and the credentials of an event are encrypted to ensure confidentiality, as described in Section 5.5.4. This makes it infeasible to route events from publishers to subscribers in a publish/subscribe overlay network. The $Pr^s_{i,j}[3]$ keys assigned to a subscription f provide peers in the publish/subscribe overlay network the ability to make routing decisions by testing (without decrypting an event) whether the credentials associated with an encrypted event match the credentials of the subscription f (cf. Section 5.5.5). In other words, $Pr^s_{i,j}[3]$ keys enable credential search on encrypted events.

5.5.4 Publishing Events

To ensure confidentiality a publisher p encrypts an event message before publishing it. Moreover, to permit subscribers to verify the authenticity and integrity of the encrypted event message, it is signed using the private keys of the publisher p. In the following, we show the (cryptographic) steps performed by the publisher p to encrypt and sign an event message M.

5.5.4.1 Encryption

To encrypt an event message M, a publisher p chooses $b_i \in \mathbb{Z}_q$ at random for each attribute A_i of the event, such that $b = \sum_{i=1}^{d} b_i$.[‡] These random values ensure that only the subscribers who have matching credentials for each of the attributes should be able to decrypt the event. Furthermore, the publisher generates a fixed length random key SK for each event.

More precisely, the publisher uses parameters from the master public key MPu and generates the ciphertexts for the event as follows.

- *Step 1:* Compute:

  $$CT_1 = \hat{e}(g_1, g_2)^b SK, \quad CT_2 = h^b \quad \text{and} \quad CT_3 = \text{BlockCipher}(Msg||0^*)^{SK}$$

  where $Msg = (M, \{Pu^p_{i,j}\})$ defines a record that includes, i) the actual event message M and, ii) the public keys of the credentials which authorize the publisher p to send the event.

  The cost of asymmetric encryption generally increases with the size of the plaintext. Therefore, only a fixed length random key SK is encrypted using the private keys of publisher. The record Msg is encrypted with a symmetric encryption algorithm such as AES [MVM09], Serpent [KKS00] or Triple DES [BB12], using key SK.

[‡]Lagrange interpolating polynomial mechanism [GPSW06, CHH10] can also be used in place of randomization.

During decryption, a subscriber does not know about the credentials with which the event is encrypted and cannot tell in advance whether he is authorized to read the event message. Therefore, to enable the subscribers to detect the successful decryption of events, *Msg* is appended with a predefined number of zeros ($Msg||0^*$). Alternatively, hash of *Msg*, i.e., $H_2(Msg)$, can be included in the ciphertext to serve the same purpose.

Moreover, *Msg* includes the public keys of the publisher in addition to the event message M, as mentioned earlier. The inclusion of these public keys increases the efficiency of signature verification process (cf. Section 5.5.5.2) at the expense of a small increase in the ciphertext size (one public key per attribute).

- *Step 2:* For each attribute A_i, compute $CT_i = g^{b_i}$. The CT_i ciphertexts along with $CT^{'}_{i,j}$ (created in *Step 3*) are used for the routing of encrypted events. In particular, the ciphertexts CT_i and $CT^{'}_{i,j}$ are constructed according to the searchable encryption scheme of Boneh et al. [BCOP04] and hence enable credential search on encrypted events. Peers in the publish/subscribe overlay network route encrypted events by matching credentials hidden in $Pr^s_{i,j}[3]$ keys of subscriptions (cf. Section 5.5.3) against the credentials encoded in these ciphertexts, as described in Section 5.6.4.

- *Step 3:* For each attribute of the event, a ciphertext should be created for every credential that matches the value associated with that attribute, so that a subscriber with any of these credentials should be able to decrypt the event. For example, in case of a numeric attribute with value mapped to 0000, a ciphertext should be disseminated for the credentials 0000,000,00 and 0.

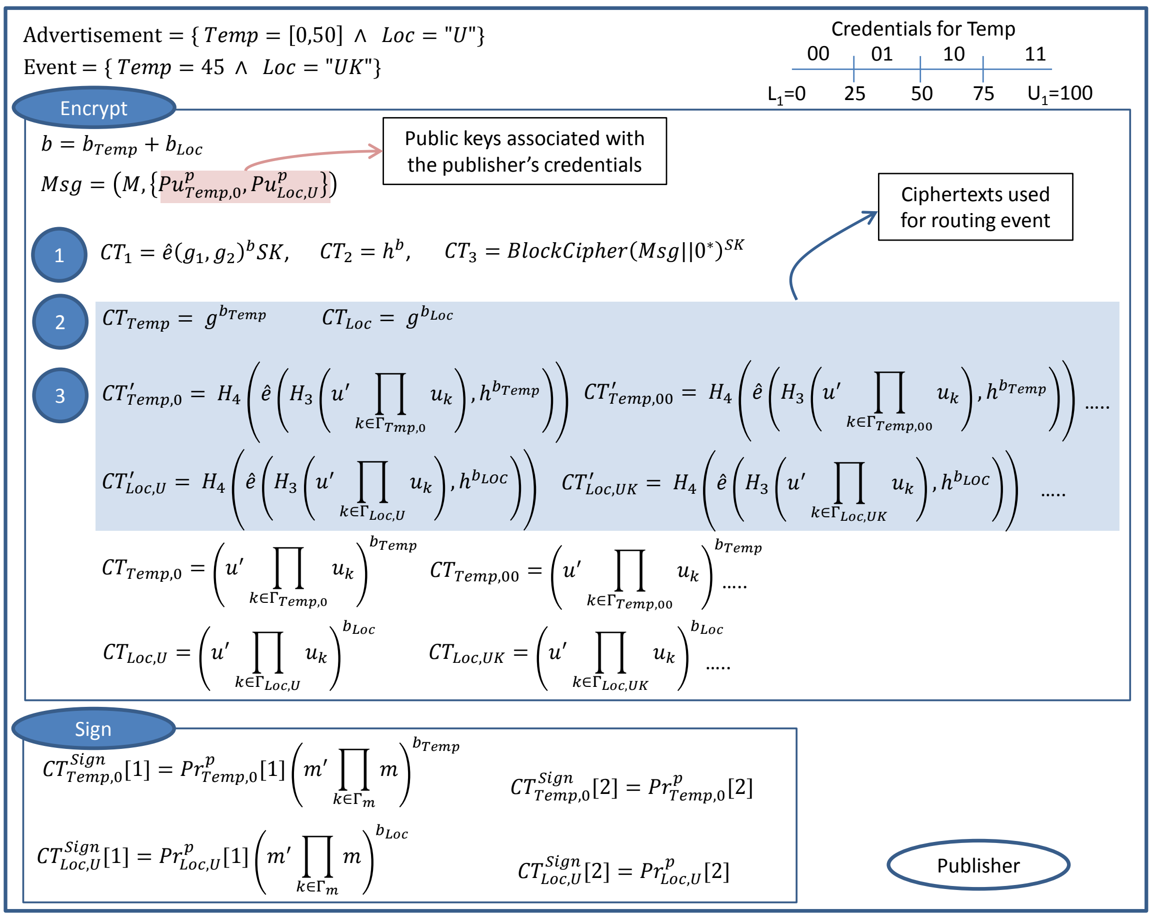
  For each credential $Cred_{i,j}$ that matches the value of the attribute A_i, compute $CT_{i,j} = \left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)^{b_i}$ and $CT^{'}_{i,j} = H_4\left(\hat{e}\left(H_3\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right), h^{b_i}\right)\right)$, where $\Gamma_{i,j}$ is calculated, as described in Section 5.5.3.

  The ciphertexts are ordered according to the containment relationship (in descending order) between their associated credentials, e.g., for the above example the order is $[CT_{i,0}, CT_{i,00}, CT_{i,000}, CT_{i,0000}]$.

5.5.4.2 Signature

Finally, the publisher p signs the ciphertexts using its private keys. It computes $v_m = H_2(M)$ a bit string of length n_m. Let $v_m[k]$ denotes the kth bit and $\Gamma_m \subseteq \{1, 2, \ldots, n_m\}$ be the set of all k for which $v_m[k] = 1$. For each attribute, the credential $Cred_{i,j}$ that authorizes the publisher p to send the corresponding attribute value, p computes:

$$CT^{sign}_{i,j}[1] = Pr^p_{i,j}[1]\left(m' \prod_{k \in \Gamma_m} m_k\right)^{b_i} \text{ and } CT^{sign}_{i,j}[2] = Pr^p_{i,j}[2]$$



Advertisement = { $Temp$ = [0,50] ∧ Loc = "U"}
Event = { $Temp$ = 45 ∧ Loc = "UK"}
Credentials for Temp
00
01
10
11
L_1=0
25
50
75
U_1=100
Encrypt
$b = b_{Temp} + b_{Loc}$
$Msg = (M, \{Pu^p_{Temp,0}, Pu^p_{Loc,U}\})$
Public keys associated with the publisher's credentials
Ciphertexts used for routing event
1
$CT_1 = \hat{e}(g_1, g_2)^b SK, \quad CT_2 = h^b, \quad CT_3 = BlockCipher(Msg||0^*)^{SK}$
2
$CT_{Temp} = g^{b_{Temp}} \quad CT_{Loc} = g^{b_{Loc}}$
3
$CT'_{Temp,0} = H_4\left(\hat{e}\left(H_3\left(u' \prod_{k \in \Gamma_{Tmp,0}} u_k\right), h^{b_{Temp}}\right)\right) \quad CT'_{Temp,00} = H_4\left(\hat{e}\left(H_3\left(u' \prod_{k \in \Gamma_{Temp,00}} u_k\right), h^{b_{Temp}}\right)\right)$ .....
$CT'_{Loc,U} = H_4\left(\hat{e}\left(H_3\left(u' \prod_{k \in \Gamma_{Loc,U}} u_k\right), h^{b_{Loc}}\right)\right) \quad CT'_{Loc,UK} = H_4\left(\hat{e}\left(H_3\left(u' \prod_{k \in \Gamma_{Loc,UK}} u_k\right), h^{b_{Loc}}\right)\right)$ .....
$CT_{Temp,0} = \left(u' \prod_{k \in \Gamma_{Temp,0}} u_k\right)^{b_{Temp}} \quad CT_{Temp,00} = \left(u' \prod_{k \in \Gamma_{Temp,00}} u_k\right)^{b_{Temp}}$ .....
$CT_{Loc,U} = \left(u' \prod_{k \in \Gamma_{Loc,U}} u_k\right)^{b_{Loc}} \quad CT_{Loc,UK} = \left(u' \prod_{k \in \Gamma_{Loc,UK}} u_k\right)^{b_{Loc}}$ .....
Sign
$CT^{Sign}_{Temp,0}[1] = Pr^p_{Temp,0}[1]\left(m' \prod_{k \in \Gamma_m} m\right)^{b_{Temp}} \quad CT^{Sign}_{Temp,0}[2] = Pr^p_{Temp,0}[2]$
$CT^{Sign}_{Loc,U}[1] = Pr^p_{Loc,U}[1]\left(m' \prod_{k \in \Gamma_m} m\right)^{b_{Loc}} \quad CT^{Sign}_{Loc,U}[2] = Pr^p_{Loc,U}[2]$
Publisher

Figure 5.5: An exemplary event encrypted and signed by a publisher.

The credentials $Cred_{i,j}$ used for signatures are same to those included in CT_3, i.e., the public keys (of the publisher p) associated with these credentials are already included in Msg.

Figure 5.5 shows the steps followed and the ciphertexts generated by a publisher during the encryption and signature of an exemplary event.

5.5.5 Receiving Events

On receiving the ciphertexts from a publisher p, a subscriber s tries to decrypt the event using its private keys. If the decryption is successful, the subscriber s then checks the authenticity of the decrypted event by verifying the signatures (associated with the event) using the public keys of the publisher p. In the following, we show the cryptographic steps performed by the subscriber s to decrypt the event message M and verify its signatures.

5.5.5.1 Decryption

The $CT_{i,j}$ ciphertexts for each attribute are strictly ordered according to the containment relation between their associated credentials (cf. Section 5.5.4.1, *Step 3*), therefore a subscriber only tries to decrypt the ciphertext whose position coincides with the position of its credential in the containment hierarchy of the corresponding attribute. The position of a credential can be easily determined by calculating its length. For example, for a numeric attribute, credential 0000 occupies 4^{th} position in the containment hierarchy, i.e., after 0,00 and 000. Subscribers decrypt the event in the following manner.

- *Step 1:* The symmetric key SK is retrieved from the ciphertext CT_1 by performing the following pairing-based cryptographic operations.

$$DT = \frac{\left(\prod_{i=1}^{d} \frac{\hat{e}\left(Pr^{s}_{i,\tau_i}[1], CT_i\right)}{\hat{e}\left(Pr^{s}_{i,\tau_i}[2], CT_{i,\tau_i}\right)}\right) CT_1}{\hat{e}\left(CT_2, Pr^{s}[4]\right)} = SK \tag{5.1}$$

  where τ_i is the credential assigned to the subscriber for the attribute A_i.[§] As mentioned above, for each attribute A_i only the ciphertext that corresponds to the credential assigned to the subscriber is used during decryption, i.e., CT_{i,τ_i}. The security intuition as well as the correctness of the operations performed by Equation 5.1 are discussed in Section 5.5.6.

- *Step 2:* Symmetric key SK is then used to recover $Msg = (M, \{Pu^{p}_{i,j}\})$ from CT_3. The successful decryption of *Msg* is detected by looking for predefined number of zeros appending the *Msg* record or verifying the hash of *Msg*, i.e., $H_2(Msg)$.

5.5.5.2 Verification

A subscriber will only accept the message if it is from an authorized publisher. To check the authenticity of an event, subscribers use the master public key (MPu) and perform the following steps.

- *Step 1:* Compute: $VT_L = \hat{e}\left(\prod_{i=1}^{d} CT^{sign}_{i,j}[1], g\right)$, where $\prod_{i=1}^{d} CT^{sign}_{i,j}[1]$ represents the product of all received $CT^{sign}_{i,j}[1]$ ciphertexts.

- *Step 2:* Compute: $VT_{R1} = \prod_{i=1}^{d} \hat{e}\left(g_1, g_2\right)$.

[§]A subscriber might have many credentials for a single attribute, e.g., $\log_2(\mathfrak{Z}_i)$ in the worst case for a numeric attribute A_i. Our overlay topology maintenance (cf. Section 5.6.3) and event dissemination (cf. Section 5.6.4) mechanisms ensure that a subscriber knows the exact credential needed to decrypt the event.

- *Step 3:* Compute: $VT_{R2} = \hat{e}\left(\prod_{i=1}^{d}\left(u'\prod_{k\in\Gamma_{i,j}} u_k\right), \prod_{i=1}^{d} CT_{i,j}^{sign}[2]\right)$, where $\prod_{i=1}^{d}\left(u'\prod_{k\in\Gamma_{i,j}} u_k\right)$ represents the product of all $Pu_{i,j}^{p}$ in CT_3 and $\prod_{i=1}^{d} CT_{i,j}^{sign}[2]$ is the product of all received $CT_{i,j}^{sign}[2]$ ciphertexts.
- *Step 4:* Compute: $VT_{R3} = \hat{e}\left(m'\prod_{k\in\Gamma_m} m_k, \prod_{i=1}^{d} CT_i\right)$.

The received event is authentic if the following identity holds.

$$VT_L = VT_{R1} \times VT_{R2} \times VT_{R3}$$

Remember, the ciphertext CT_3 contains the public keys of the credentials which authorize the publisher to send the event. If no such public keys are included in CT_3, then the subscriber should try to authenticate the event by checking for all possible credentials which a publisher might hold to publish the event. For example, an event with a single numeric attribute and a value mapped to 0000 can be published by the publisher with credentials 0000,000,00 or 0. Similarly, for prefix matching an event with label "sea" can be published by the publisher with credentials "sea", "se" or "s". Therefore, in the absence of public keys of credentials in CT_3, our approach is as follows: a subscriber checks the authenticity of the event for each attribute A_i separately[¶], by verifying that for one of the possible credentials $Cred_{i,\tau}$ the following identity holds:

$$\hat{e}\left(CT_{i,\tau}^{sign}[1], g\right) = \hat{e}\left(g_1, g_2\right)\ \hat{e}\left(u'\prod_{k\in\Gamma_{i,\tau}} u_k, CT_{i,\tau}^{sign}[2]\right)\ \hat{e}\left(m'\prod_{k\in\Gamma_m} m_k, CT_i\right) \tag{5.2}$$

In this case, the total verification cost is $\sum_{i=1}^{d}\log_2(\mathfrak{Z}_i)$ and $\sum_{i=1}^{d}\mathfrak{L}_i$ for numeric and string attributes respectively. It is interesting to note that the separate verification for each attribute can be performed, even if the public keys are included in CT_3, however, the combined approach as described above is better because (bilinear) pairing operations are much more expensive than multiplications. The combined approach performs only four pairing operations, whereas $3d + 1$ pairing operations are needed for separate verification.

Table 5.1 shows the asymptotic performance of the proposed security methods. For numeric attributes, the number of security parameters in master public key MPu is constant, i.e., $O(1)$, whereas the cost to sign, decrypt and verify an (encrypted) event increases with the number of attributes in the system, i.e., $O(d)$. It is worth noting that the linear verification cost (i.e., $O(d)$) is only achievable if CT_3 contains the public keys of the credentials which authorize the publisher to send the event, as mentioned above. Moreover, the number of private keys assigned to a subscription, the number of

[¶]Combined verification, as mentioned above, is not feasible here and will result in a cost of $\prod_{i=1}^{d}\log_2\mathfrak{Z}_i$ and $\prod_{i=1}^{d}\mathfrak{L}_i$ for numeric and string attributes respectively.

Table 5.1: Cost of security methods.

	Public params	Private keys	Ciphertext Size	Encryption cost	Decryption cost	Sign cost	Verification cost
Numeric	$O(1)$	$O\left(\sum_{i=1}^{d} \log_2 \mathfrak{Z}_i\right)$	$O\left(\sum_{i=1}^{d} \log_2 \mathfrak{Z}_i\right)$	$O\left(\sum_{i=1}^{d} \log_2 \mathfrak{Z}_i\right)$	$O(d)$	$O(d)$	$O(d)$
String	$O(1)$	$O\left(\sum_{i=1}^{d} \mathfrak{L}_i\right)$	$O\left(\sum_{i=1}^{d} \mathfrak{L}_i\right)$	$O\left(\sum_{i=1}^{d} \mathfrak{L}_i\right)$	$O(d)$	$O(d)$	$O(d)$

ciphertexts generated by the publisher (during encryption and signature) and the cost to encrypt an event increases with the number of attributes in the system as well as the domain and the granularity of each attribute (cf. Section 5.4.1), i.e., $O\left(\sum_{i=1}^{d} \log_2 \mathfrak{Z}_i\right)$.

In the case of string attributes, the size of master public key, decryption cost, signature cost and verification cost is same as that of numeric attributes. However, the private key size, ciphertext size and encryption cost depend on the number of string attributes in the system and the length of the (longest) label assigned to a leaf node of the prefix trie of each attribute (cf. Section 5.4.2), i.e., $O\left(\sum_{i=1}^{d} \mathfrak{L}_i\right)$.

5.5.6 Security Analysis

In this section, we analyse the strength of our proposed security methods. As mentioned earlier, the proposed methods are adapted from the ciphertext-policy attribute-based encryption (CP-ABE) scheme and therefore, their strength essentially depends on the security of CP-ABE. Bethencourt et al. [BSW07] proved CP-ABE scheme to be secure against chosen plaintext attacks under the hardness of Decisional Bilinear Diffie-Hellman (DBDH) problem. Moreover, Bethencourt et al. [BSW07] showed that CP-ABE can also be secured against chosen ciphertext attacks by applying random oracle technique such as Fujisaki-Okamoto transformation [FO99].

In the following, we show that our modifications to the CP-ABE scheme preserve the same security strength.

Binding Ciphertexts of an Event Using Randomness: During encryption our construction adds randomness to the ciphertexts associated with the attributes of an event, i.e., a separate $b_i \in \mathbb{Z}_q$ is used in the generation of ciphertexts for each attribute A_i (cf. Section 5.5.4). Here, we argue that the randomness can only be cancelled out, i.e., the encrypted event can only be successfully decrypted, if the private keys of a subscription possess correct credentials for each attribute of the event. Moreover, privates keys should belong to the same subscription, i.e., keys from different subscriptions cannot be used to decryption an event.

In order to simplify our discussion, we label different parts of Equation 5.1, as follows.

$$DT = \left(DT'' = \frac{DT' = \prod_{i=1}^{d} \frac{\hat{e}\left(Pr^{s}_{i,\tau_i}[1], CT_i\right)}{\hat{e}\left(Pr^{s}_{i,\tau_i}[2], CT_{i,\tau_i}\right)}}{\hat{e}\left(CT_2, Pr^{s}[4]\right)} \right) CT_1$$

To decrypt an event an attacker must clearly recover $\hat{e}(g_1, g_2)^b$ to retrieve SK from CT_1, i.e.,

$$\frac{CT_1 = \hat{e}\left(g_1, g_2\right)^b SK}{\hat{e}\left(g_1, g_2\right)^b} \rightarrow SK \tag{5.3}$$

The group elements g_1 and g_2 used in the (bilinear) pairing $\hat{e}(g_1, g_2)^b$ are public parameters and therefore, the challenge for an attacker is to find b. For this purpose, the attacker must compute DT' by pairing private keys $Pr^{s}_{i,\tau_i}[1]$ and $Pr^{s}_{i,\tau_i}[2]$ assigned to a subscription with credential τ_i for each attribute A_i of the event with the corresponding ciphertexts CT_i and CT_{i,τ_i}, so that the randomness attached to (the ciphertexts of) each attribute during encryption is added up to retrieve $b = \sum_{i=1}^{d} b_i$, i.e.,

$$\begin{aligned}
DT' &= \prod_{i=1}^{d} \frac{\hat{e}\left(Pr^{s}_{i,\tau_i}[1], CT_i\right)}{\hat{e}\left(Pr^{s}_{i,\tau_i}[2], CT_{i,\tau_i}\right)} && (5.4)\\
&= \prod_{i=1}^{d} \frac{\hat{e}\left(g_2^{\gamma_s}\left(u' \prod_{k \in \Gamma_{i,\tau}} u_k\right)^{\gamma_{i,\tau}}, g^{b_i}\right)}{\hat{e}\left(g^{\gamma_{i,\tau}}, \left(u' \prod_{k \in \Gamma_{i,\tau}} u_k\right)^{b_i}\right)} && (5.5)\\
&= \prod_{i=1}^{d} \frac{\hat{e}\left(g_2^{\gamma_s}, g^{b_i}\right)\ \hat{e}\left(\left(u' \prod_{k \in \Gamma_{i,\tau}} u_k\right)^{\gamma_{i,\tau}}, g^{b_i}\right)}{\hat{e}\left(g^{\gamma_{i,\tau}}, \left(u' \prod_{k \in \Gamma_{i,\tau}} u_k\right)^{b_i}\right)} && (5.6)\\
&= \prod_{i=1}^{d} \frac{\hat{e}\left(g_2^{\gamma_s}, g^{b_i}\right)\ \hat{e}\left(u' \prod_{k \in \Gamma_{i,\tau}} u_k, g\right)^{\gamma_{i,\tau}.b_i}}{\hat{e}\left(g, u' \prod_{k \in \Gamma_{i,\tau}} u_k\right)^{\gamma_{i,\tau}.b_i}} = \prod_{i=1}^{d} \hat{e}\left(g_2, g\right)^{\gamma_s.b_i} && (5.7)\\
&= \hat{e}\left(g_2, g\right)^{\gamma_s.\sum b_i} = \hat{e}\left(g_2, g\right)^{\gamma_s.b} && (5.8)
\end{aligned}$$

In case the attacker uses private keys with a wrong credential τ_j for an attribute A_j, he will not be able to recover $\hat{e}(g_2, g)^{\gamma_s.b_j}$ in Equation 5.7 because the term $\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)$ cannot be cancelled out and thus $\hat{e}(g_2, g)^{\gamma_s.b}$ cannot be correctly computed in Equation 5.8. Therefore, DT' can only be computed correctly if the private keys with correct credentials for each attribute of the event are available.

Moreover, the desired value b obtained by performing the cryptographic steps of $DT^{'}$ is blinded by some random γ_s and cannot be readily used to retrieve SK (cf. Equation 5.3). To detach γ_s, the attacker must compute $DT^{''}$, i.e.,

$$DT^{''} = \frac{\hat{e}\left(g_2, g\right)^{\gamma_s.b}}{\hat{e}\left(CT_2, Pr^s[4]\right)} \tag{5.9}$$

$$= \frac{\hat{e}\left(g_2, g\right)^{\gamma_s.b}}{\hat{e}\left(h^b, g_2^{\frac{\gamma_s+\alpha}{\varphi}}\right)} = \frac{\hat{e}\left(g_2, g\right)^{\gamma_s.b}}{\hat{e}\left(g^{\varphi.b}, g_2^{\frac{\gamma_s+\alpha}{\varphi}}\right)} = \frac{\hat{e}\left(g_2, g\right)^{\gamma_s.b}}{\hat{e}\left(g, g_2\right)^{\gamma_s.b+b.\alpha}} \tag{5.10}$$

$$= \frac{1}{\hat{e}\left(g, g_2\right)^{b.\alpha}} = \frac{1}{\hat{e}\left(g^{\alpha}, g_2\right)^{b}} = \frac{1}{\hat{e}\left(g_1, g_2\right)^{b}} \tag{5.11}$$

It is infeasible for an attacker to use credentials (private keys) assigned to different subscriptions in order to recover $\hat{e}(g_1, g_2)^b$ from $DT^{''}$. This is because during key generation private keys associated with different credentials of a subscription are tied together by adding some random value γ_s (cf. Section 5.5.3). If private keys from different subscriptions (that may or may not belong to the same subscriber) are used together $DT^{'}$ cannot be computed correctly and thus γ_s cannot be cancelled out during the cryptographic steps of $DT^{''}$ (cf. Equation 5.10), preventing the decryption of the event.

Finally, it is important to mention here that the original CP-ABE scheme targets embedding complex access control policies in the ciphertext. The policies are described by a monotonic access tree, where the nodes of the tree are composed of threshold gates (such as AND, OR etc.) and the leaves describe attributes. The CP-ABE scheme therefore uses Lagrange interpolation polynomial to bind access tree associated with a policy in the ciphertext. Our construction borrows the randomization principle from CP-ABE to bind together the ciphertexts associated with different attributes of an event, however, we refrain from using Lagrange polynomial to avoid complexity.

Credential Search on Encrypted Events: In our construction, we integrate the public key encryption with keyword search scheme (also known as searchable encryption scheme) of Boneh et al. [BCOP04] to perform efficient routing of encrypted events, as described later in Section 5.6.4. The scheme of Boneh et al. is already proved to be secure against chosen plaintext (i.e., keyword) attacks in the random oracle model under the hardness of Bilinear Diffie-Hellman (BDH) problem [BCOP04]. However, what is left is to show that the combination of CP-ABE and searchable encryption scheme of Boneh el al. is also secure.

First, we argue that the searchable encryption scheme does not undermine the security strength of the (modified) CP-ABE scheme. Separate keys (i.e., $Pr^s_{i,j}[3]$) are employed to perform credential search on encrypted events. These keys cannot be used by an attacker to decrypt events in the system because they are mapped to a random element of bilinear group $\mathbb{G}_1$ as a result of hash function H_3. Likewise, information about the exact credentials of an encrypted event is neither revealed from the ciphertexts $CT^{'}_{i,j}$ and CT_i nor from the unsuccessful matches with the credentials of the subscriptions. This is because an attacker (or a peer in the overlay network) can only detect that a credential τ_i associated with an attribute A_i of the encrypted event is same as the credential of the subscription if the result of the cryptographic operation $H_4\left(\hat{e}\left(Pr^s_{i,\tau_i}[3], CT_i\right)\right)$ is equal to the ciphertext $CT^{'}_{i,\tau_i}$ of the event, i.e.,

$$\begin{aligned} \text{Test}(Pr^s_{i,\tau_i}[3], CT_i, CT^{'}_{i,\tau_i}) &= H_4\left(\hat{e}\left(Pr^s_{i,\tau_i}[3], CT_i\right)\right) && (5.12)\\ &= H_4\left(\hat{e}\left(H_3\left(u'\prod_{k\in\Gamma_{i,j}} u_k\right)^{\varphi}, g^{b_i}\right)\right) && (5.13)\\ &= H_4\left(\hat{e}\left(H_3\left(u'\prod_{k\in\Gamma_{i,j}} u_k\right), g^{\varphi.b_i}\right)\right) && (5.14)\\ &= H_4\left(\hat{e}\left(H_3\left(u'\prod_{k\in\Gamma_{i,j}} u_k\right), h^{b_i}\right)\right) = CT^{'}_{i,\tau_i} && (5.15) \end{aligned}$$

Equation 5.15 clearly shows that if the credentials of the encrypted event and the subscription do not match then $CT^{'}_{i,\tau_i}$ cannot be recovered and hence the credential of the encrypted event cannot be revealed. However, successful recovery of $CT^{'}_{i,\tau_i}$ indicates that the credential of the encrypted event is same as that of the subscription.

Second, we claim that our construction preserves the strength of the searchable encryption scheme of Boneh et al. [BCOP04] for two reasons: i) the original scheme of Boneh et al. is incorporated in our construction without modifications, i.e., $Pr^s_{i,\tau_i}[3]$ keys and ciphertexts CT_i and $CT^{'}_{i,\tau_i}$ hold the same structure, and ii) the random exponential b_i employed by the ciphertexts CT_i and $CT^{'}_{i,\tau_i}$ cannot be revealed during the decryption of events as per CP-ABE scheme.

Identity-based Signcryption: In our construction, we incorporate the Identity-based signcryption scheme of Yu et al. [YYSZ09] to allow publishers to sign and encrypt events at the same time. Yu et al. [YYSZ09] already proved that their Identity-based signcryption scheme is secure against adaptive ciphertext attacks under DBDH and is unforgeable under the computational Diffie-Hellman assumption. In the following, we show that the combination of CP-ABE and signcryption of Yu et al. is also secure.

We argue that the signcryption does not undermine the security strength of the (modified) CP-ABE scheme. More precisely, to provide signcryption two additional ciphertexts, i.e., $CT^{sign}_{i,j}[1]$ and $CT^{sign}_{i,j}[2]$ are included in our construction. These signcryption ciphertexts enclose private keys of publishers, e.g., $CT^{sign}_{i,j}[2]$ is equal to $Pr^p_{i,j}[2]$. However, the private keys embedded in these signcryption ciphertexts cannot be utilized by an attacker to decrypt events in the system due to the fact that the public keys encoded in the private keys of the publishers use a different symbol, i.e., PUB (cf. Section 5.5.2) and therefore the term $\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)$ cannot be cancelled out to recover $\hat{e}(g_2, g)^{\gamma_s . b}$ in Equation 5.8. Moreover, these signcryption ciphertexts clearly do not reveal any information about the key parameters of the system as long as the security strength of the original signcryption scheme of Yu et al. [YYSZ09] is preserved.

Likewise, it is easy to realize that our construction upholds the strength of the signcryption scheme of Yu et al. [YYSZ09], mainly for two considerations. First, private keys of publishers enclosed by the signcryption ciphertexts are generated using the mechanism proposed by Yu et al. Second, ciphertexts utilized during the verification of signatures hold the same structure (as suggested by Yu et al.) and the random exponential in those ciphertexts (i.e., b_i in $CT^{sign}_{i,j}[1]$ and CT_i) cannot be revealed during decryption as per CP-ABE scheme. For the same reasons, separate verification of signatures for each attribute of an event by Equation 5.2 is valid and secure, as proved by Yu et al.

Finally, we show the correctness of our approach to verify the signatures associated with all the attributes of an event simultaneously. In this case, a received event is considered to be published by an authorized publisher if the identity $VT_L = VT_{R1} \times VT_{R2} \times VT_{R3}$ holds (cf. Section 5.5.5.2). The correctness of the verification process can be established as follows.

$$\begin{aligned}
VT_L &= \hat{e}\left(\prod_{i=1}^{d} CT^{sign}_{i,j}[1], g\right) = \hat{e}\left(\prod_{i=1}^{d} g_2^{\alpha}\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)^{\gamma_{i,j}}\left(m' \prod_{k \in \Gamma_m} m_k\right)^{b_i}, g\right) \\
&= \hat{e}\left(\prod_{i=1}^{d} g_2^{\alpha}, g\right) \times \hat{e}\left(\prod_{i=1}^{d}\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)^{\gamma_{i,j}}, g\right) \times \hat{e}\left(\prod_{i=1}^{d}\left(m' \prod_{k \in \Gamma_m} m_k\right)^{b_i}, g\right) \\
&= \prod_{i=1}^{d} \hat{e}\left(g_2^{\alpha}, g\right) \times \prod_{i=1}^{d} \hat{e}\left(\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right)^{\gamma_{i,j}}, g\right) \times \hat{e}\left(\left(m' \prod_{k \in \Gamma_m} m_k\right)^{\sum_{i=1}^{d} b_i}, g\right) \\
&= \prod_{i=1}^{d} \hat{e}\left(g_2, g^{\alpha}\right) \times \hat{e}\left(\prod_{i=1}^{d}\left(u' \prod_{k \in \Gamma_{i,j}} u_k\right), \prod_{i=1}^{d} g^{\gamma_{i,j}}\right) \times \hat{e}\left(m' \prod_{k \in \Gamma_m} m_k, g^{b}\right) \\
&= VT_{R1} \times VT_{R2} \times VT_{R3}
\end{aligned}$$

5.5.7 Rekeying

When a subscriber arrives or leaves the system, the keys of all the subscribers with the corresponding credentials should be changed in order to provide forward and backward secrecy. The forward secrecy ensures that a subscriber who left the system cannot

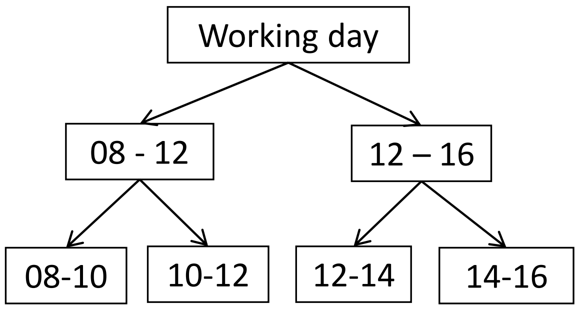


Figure 5.6: Fine grain key management within an epoch.

decrypt future encrypted events, whereas the backward secrecy guards that a newly arrived subscriber cannot decrypt (encrypted) events published in the past. Changing the keys for each incoming or outgoing subscriber impacts the scalability of the system in terms of communication overhead and load on the key server. Therefore, our approach is to use periodic rekeying by dividing the system time into epochs.

The length of the epoch is directly related to the overhead incurred due to rekeying. Big length epochs are used because they require less frequent rekeying, however, they can only provide coarse grain access control. Our approach enables fine grain access control within big length epochs by the addition of a time attribute in the system. The time attribute specifies credentials of peers to receive or send the events at finer granularity within each epoch. For example, in Figure 5.6 the epoch length of one working day is hierarchically divided into smaller time slots. Only the peers with the credentials of a time slot can receive or send the event in that time slot. The time attribute is treated like any other attribute in the system. Each encrypted event contains the ciphertexts for each credential which authorizes a subscriber to receive events in the current time slot. Similarly, for successful decryption of an event, a subscriber should use the private key that corresponds to the required time credential. The construction of our security methods guarantee that events cannot be retrieved without having required time keys. Furthermore, the time keys from different subscriptions cannot be combined to retrieve unauthorized events.

Of course, the flexibility of having fine grain access comes at the cost of managing more keys. However, the overhead is small compared to the granularity of access, e.g., access control at the granularity of a second for an epoch of one month, requires a subscriber or a publisher to maintain 22 decryption keys in the worst case.

Each subscriber or publisher contacts the key server at the end of the epoch and receives the private keys for the new epoch. It is interesting to note that in case of smart cards, private keys are generated locally, without incurring any communication overhead. Otherwise, the overhead for regenerating private keys and securely communicating them to respective publishers or subscribers is shared between the replicas of the key server.

5.6 Subscription Confidentiality

In the previous section, we have proposed security methods to achieve authentication of publishers and subscribers as well as confidentiality and integrity of events. Moreover, we have analysed the performance and security strength of the proposed methods.

Now we focus on the goal of achieving subscription confidentiality. The subscription confidentiality in a broker-less environment is highly influenced from the organization of publishers and subscribers in the overlay network. Therefore, in the subsequent sections, we first introduce broker-less publish/subscribe overlay network (cf. Section 5.6.1) and define weaker notion of subscription confidentiality (cf. Section 5.6.2). Afterwards, we present secure overlay maintenance (cf. Section 5.6.3) and event dissemination protocols (cf. Section 5.6.4) to achieve the proposed weaker subscription confidentiality.

5.6.1 Publish/Subscribe Overlay

The publish/subscribe overlay is a virtual forest of logical trees, where each tree is associated with an attribute (cf. Figure 5.7). A subscriber joins the trees corresponding to the attributes of its subscription. Similarly, a publisher sends an event on all the trees associated with the attributes in the event.

Within each attribute tree, subscribers are connected according to the containment relationship between their credentials associated with the attribute. The subscribers with coarser credentials (e.g., the ones mapped to coarser sub-spaces in case of numeric attributes) are placed near the root of the tree and forward events to the subscribers with finer credentials. A subscriber with more than one credential can be handled by running multiple virtual peers on a single physical node, each virtual peer maintaining its own set of tree links. For example, in Figure 5.7 the subscriber s_3 has two credentials $\{000, 010\}$ and is connect to two places in the tree.

In order to connect to an attribute tree, a newly arriving subscriber s_n sends the connection request along with its credential to a random peer s_r in the tree. The peer s_r compares the request credential with its own; if the peer's credential covers the request credential and the peer can accommodate more children, it accepts the connection. Otherwise, the connection request is forwarded to all the children with covering credentials and the parent peer with the exception of the peer, from which it was received. In this way, the connection request might be forwarded by many peers in the tree before it reaches the suitable peer with covering credential and available connection. Figure 5.7 shows the path followed by a request from a subscriber s_n until it reaches the desired parent subscriber.

The drawback of maintaining separate trees for each attribute is that the subscribers also receive events that match only a part of their subscription (false positives). However, it cannot affect event confidentiality because false positives cannot be decrypted without having required credentials for each attribute.

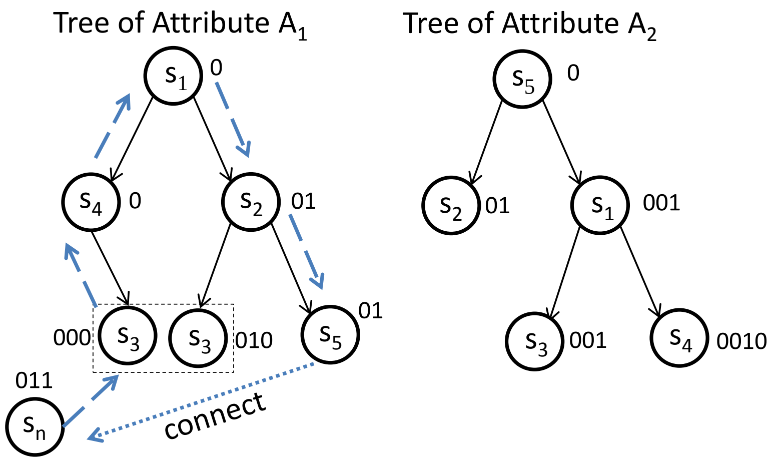


Figure 5.7: Publish/Subscribe system with two numeric attributes.

It is worth mentioning here that the publish/subscribe overlay network presented in Chapter 2 also organizes subscribers according to the containment relationship between their subscriptions. However, we decided against using the publish/subscribe overlay organization from Chapter 2 because it enables subscribers to coarsen their credentials (peer-level subscriptions) according to their delay requirements and bandwidth constraints. The coarsening of credentials violates event confidentiality because a subscriber whose credentials are coarsened can receive and decrypt unauthorized events (i.e., events not matching its original credentials). Note that the secure overlay maintenance algorithm presented later in Section 5.6.3 mandates that only the subscribers with valid private keys for their credentials are placed in the attribute trees and therefore coarsening of credentials is only possible if the corresponding private keys (from the key server) are available. For the same reason, the publish/subscribe overlay presented in this section prohibits satisfaction of individual delay requirements of subscribers in certain scenarios. For instance, tight delay requirements of a subscriber s with finer credentials may not be satisfied. This is because the subscriber s (with finer credentials) may require to coarsen its credentials to be placed near the publishers in the attribute trees, which is not allowed to ensure event confidentiality.

5.6.2 Weak Subscription Confidentiality

The broker-less publish/subscribe overlay introduced in the previous section organizes peers (i.e., subscribers) according to the containment relationship between their subscriptions (i.e., credentials). To maintain such an overlay network each peer should know the subscriptions (credentials) of its children as well as its parent(s). This makes provision of strong subscription confidentiality infeasible. We therefore propose a weaker notion of subscription confidentiality. The weaker confidentiality limits the information leaked/inferred about the subscriptions of subscribers to the bare minimum required for the proper functioning and maintenance of the publish/subscribe overlay.

Definition 5.6.1 *Let s_1 and s_2 denote two subscribers in a publish/subscribe system which both possess credentials for an attribute A_i.* Weak subscription confidentiality *ensures that at most the following information can be inferred about the credentials of the subscribers:*

1. *The credential of s_1 is either coarser or equal to the credentials of s_2.*
2. *The credential of s_1 is either finer or equal to the credentials of s_2.*
3. *The credential of s_1 and s_2 are not in any containment relationship.*

It is worth noting that broker-based publish/subscribe systems also rely on containment relationships between the subscriptions to perform efficient content-based routing. This implies that even though the subscriptions are encrypted, an untrusted broker can easily determine the relationships between the subscriptions of different subscribers [RR06], e.g., a broker can detect that the subscription of s_1 is covered by (or coarser than) the subscription of s_2. Therefore, we argue that the Definition 5.6.1 is consistent with the subscription security model used by the untrusted broker-based publish/subscribe systems in the literature.

5.6.3 Secure Overlay Maintenance

In the following, we propose a secure protocol to maintain the desired publish/subscribe overlay topology without violating the weak subscription confidentiality. For simplicity and without loss of generality, here we discuss the overlay maintenance with respect to a single tree associated with a numeric attribute A_i and each of the subscribers owns a single credential.

The secure overlay maintenance protocol is based on the idea that in the tree subscribers are always connected according to the containment relationship between their credentials, e.g., a subscriber with credential 00 can only connect to the subscribers with credentials 0 or 00. Therefore, the connection (or join) request from a newly arrived subscriber s should only be deciphered by those (existing) subscribers in the tree whose credentials cover the credential of s. The rest of the subscribers in the tree even though participate in the maintenance protocol, i.e., forwarding of join request to find an appropriate position for s in the tree, cannot gain information about the credential of s.

More precisely, a new subscriber s generates a fixed length random key SW and encrypts it with the public keys $Pu^s_{i,j}$ of all credentials that cover its own credential, e.g., a subscriber with credential 00 will generate ciphertexts by applying the public keys

$Pu^s_{i,0}$ and $Pu^s_{i,00}$. The generated ciphertexts are added to a *connection request* (CR) and the request is forwarded to a random peer in the tree. A connection is established if the peer can decrypt any of the ciphertexts using its private keys. To enable peers to detect the successful decryption of any of the ciphertexts in the CR, either the SW is appended with predefined number of zeros, i.e., $(SW||0^*)$ or hash of SW is included.

It is worth noting that a random key SW is used in the creation of connection request (CR) instead of any deterministic string, e.g., identity of the subscriber s. This is because an attacker can try to encrypt the known deterministic string using the public keys of different credentials and compare the generated ciphertexts with the ciphertexts in the connection request (CR) of the subscriber s to guess about the actual credential of s. Moreover, we will discuss later that the use of a single random key SW during the creation of CR is not enough to ensure security.

Filling the Security Gaps: By looking at the number of ciphertexts in the connection request a peer can detect the credential of the joining subscriber s. For example, a subscriber with credential 00 can only connect to 0 or 00 and therefore, a connection request will have two ciphertexts, whereas the connection request for 000 will have three ciphertexts. In the worst case, a subscriber has a credential of the finest granularity. This can be covered by $\log_2(\mathfrak{Z}_i)$ and $\mathfrak{L}_i$ other credentials for the numeric and the string attribute A_i respectively (cf. Section 5.4). Therefore, a connection request contains in the worst case that many ciphertexts. To avoid any information leak, ciphertexts in the connection request are always kept in $O(\log_2\ \mathfrak{Z}_i)$ ($O(\mathfrak{L}_i)$ for prefix matching) by adding random ciphertexts if needed. Furthermore, the ciphertexts are shuffled to avoid any information leak from their order.

The use of a single random key SW for the generation of ciphertexts in CR is not enough to ensure security and can leak information about the credential of the joining subscriber s. This is because a peer which has successfully decrypted one of the ciphertexts in CR of s can encrypt the (retrieved) SW using the public keys of different credentials (i.e., $O(\log_2\ \mathfrak{Z}_i)$ and $O(\mathfrak{L}_i)$ public keys for numeric and string attribute A_i respectively) and compare the generated ciphertexts with the ciphertexts in CR to determine the random ciphertexts in CR and thus can easily guess the credential of the joining subscriber s. For example, a peer who successfully recovered SW from the CR of a subscriber with credential 00, can easily detect that the ciphertext in CR that corresponds to the credential 000 is random by encrypting SW with the public key $Pu^s_{i,000}$ and comparing the resultant ciphertext with the ciphertexts in CR. Therefore, a different random key SW is used for the generation of each ciphertext in the CR to avoid any information leak.

Finally, to avoid an attacker to generate arbitrary connection request messages and try to discover the credential of other peers in the system, the connection request is signed by the key server. This step needs to be performed only once, when a newly arrived subscriber authorizes itself to the key server in order to receive private keys for its credentials.

Algorithm 9 Secure overlay maintenance protocol at peer s_q

```
1:  upon event Receive(CR of $s_{new}$ from $s_p$) do
2:    if $decrypt\_request(CR) == SUCCESS$ then
3:      if degree($s_q$) == available then // can have child peers
4:        connect to the $s_{new}$
5:      else
6:        forward CR to $\{child\ peers\ and\ parent\} - s_p$
7:    if $decrypt\_request(CR) == FAIL$ then
8:      if $s_p$ == parent then
9:        Try to swap by sending its own CR to the $s_{new}$.
10:     else
11:       forward to parent
```

Overall Algorithm: The secure overlay maintenance protocol is shown in Algorithm 9. In the algorithm, the procedure *decrypt_request* tries to decrypt one of the ciphertexts in the connection request message.

A peer s_q on receiving a connection request (CR) from a newly arrived subscriber s_{new} first runs *decrypt_request* procedure. Upon the successful execution of the procedure, peer s_q can except the new subscriber s_{new} as a child. However, due to degree constraints s_q may not be able to accommodate s_{new} and forward the CR request to its parent and children on the attribute tree (cf. *Algorithm* 9, *lines 1-6*).

In case the procedure *decrypt_request* fails, the connection request is only forwarded to the parent. This is because peers are maintained in the tree according to the containment relationship between their credentials and therefore, if a peer fails to decrypt a CR its children would not be successful either.

Moreover, a child peer s_q receives CR (of subscriber s_{new}) from the parent s_p only if the parent cannot accommodate more children. If s_q cannot be the parent of s_{new}, i.e., s_{new}'s credential is coarser than that of s_q, then it tries to swap its position with s_{new} by sending its own connection request (cf. *Algorithm* 9, *lines 7-9*). However, if none of the children of parent s_p can connect or swap with s_{new}, then there is no containment relationship between the credentials of the children and s_{new}. In this case, parent s_p connects the new subscriber s_{new}, but tries to find a suitable position for one of its existing children s_e down in the tree by forwarding the CR of s_e as mentioned above.

5.6.4 Secure Event Dissemination

In this section, we describe mechanisms to route events in the publish/subscribe overlay network without violating the weak subscription confidentiality.

In a tree-based publish/subscribe overlay network, events are usually disseminated from the root of the tree towards the leaf peers. Peers at each level of the tree except the lowest level, i.e., leaf peers, know the subscriptions of their children. Upon receiving an event, a non-leaf peer forwards the event to each child whose subscription matches the event. This event dissemination strategy cannot be directly applied in our system because a parent peer should know the subscriptions (i.e., credentials) of its children which violates the weak subscription confidentiality.

Cryptographic methods such as encrypted search [IRC10, WCEW02] can be employed to allow a parent peer to check whether an encrypted event matches a hidden subscription (credential) of a child peer without revealing any information. However, even these cryptographic methods are not adequate to ensure weak subscription confidentiality. This is because the secure overlay maintenance protocol (cf. Section 5.6.3) ensures that the peers are arranged in a tree according to the containment relationship between their credentials, i.e., credential of a parent peer covers the credentials of its children. Therefore, a parent peer can decrypt every event, which it forwards to the children. Regardless of the cryptographic methods, a parent can eventually discover the credentials of its children, e.g., by maintaining history of the events forwarded to each child. Two mechanisms can be used to avoid this problem.

5.6.4.1 One Hop Flooding

In one hop flooding (OHF), a parent assumes that the children have the same credentials as its own and forwards each successfully decrypted event to all of them. In turn the children forward each event which is successfully decrypted to all of their children and so on.

The detailed mechanism works as follows. To publish an event, a publisher forwards the ciphertexts of each attribute to the root of the corresponding attribute tree. All the ciphertexts of an event are labelled with a unique value such as the sequence number of the event. This helps subscribers to identify all the ciphertexts of an event (though the ciphertexts for each attribute are received on a separate tree).

Upon receiving ciphertexts on the tree associated with an attribute A_i, a subscriber may not be able to decrypt the event because the ciphertexts for other attributes of the event may not be received at that time. However, the subscriber should still decide whether to forward the received ciphertexts to its children on the attribute tree of A_i. This is accomplished by using the ciphertexts CT_i and $CT^{'}_{i,j}$ generated by the publisher of the event during encryption, as mentioned in Section 5.5.4. More precisely, the subscriber checks whether it can generate ciphertext $CT^{'}_{i,j}$ by i) performing bilinear pairing of CT_i with the private key $Pr^{s}_{i,\tau_i}[3]$ of the credential $Cred_{i,\tau_i}$ associated with the overlay connection from which the ciphertexts are received,[‖] and ii) applying the hash function H_4.

The decision to forward ciphertexts associated with an attribute A_i to the children can be described as:

$$\text{Routing Decision} = \begin{cases} \text{forward} & \text{if } H_4\left(\hat{e}\left(Pr^{s}_{i,\tau_i}[3], CT_i\right)\right) = CT^{'}_{i,\tau_i} \\ \text{drop} & \text{otherwise} \end{cases} \tag{5.16}$$

[‖]A subscriber maintains separate connection for each credential (cf. Section 5.6.1).

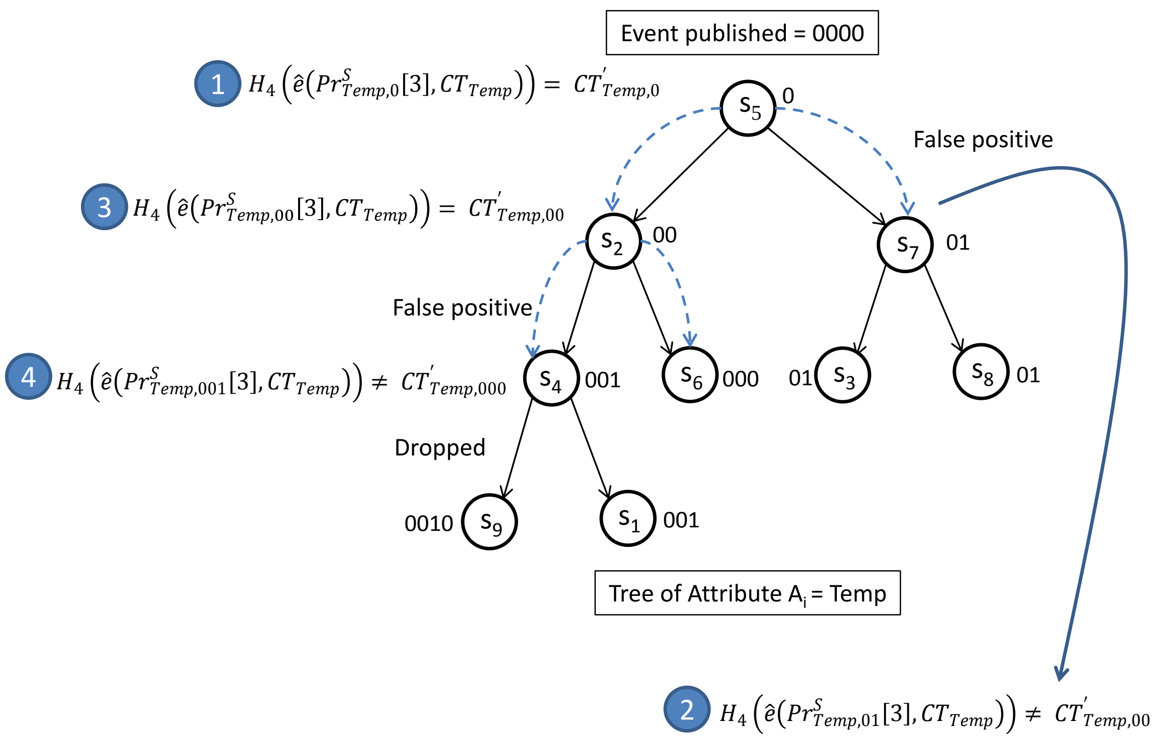

Event published = 0000
1 $H_4\left(\hat{e}(Pr^S_{Temp,0}[3], CT_{Temp})\right) = CT'_{Temp,0}$
3 $H_4\left(\hat{e}(Pr^S_{Temp,00}[3], CT_{Temp})\right) = CT'_{Temp,00}$
4 $H_4\left(\hat{e}(Pr^S_{Temp,001}[3], CT_{Temp})\right) \neq CT'_{Temp,000}$
2 $H_4\left(\hat{e}(Pr^S_{Temp,01}[3], CT_{Temp})\right) \neq CT'_{Temp,00}$
False positive
False positive
Dropped
S5 0
S2 00
S7 01
S4 001
S6 000
01 S3
S8 01
0010 S9
S1 001
Tree of Attribute A_i = Temp

Figure 5.8: Example of one hop flooding (OHF) strategy.

The successful generation of CT'_{i,τ_i} means that the credential $Cred_{i,\tau_i}$ assigned to the subscription of the subscriber matches one of the credentials in the ciphertexts of the event for attribute A_i and therefore, the ciphertexts should be forwarded to all children. However, in the case of failure, the ciphertexts are dropped from further forwarding. It is worth noting that the correctness and the security strength of the cryptographic operations performed in Equation 5.16 is already discussed in Section 5.5.6.

Once the ciphertexts of all the attributes in the event are received, the subscriber tries to decrypt the whole event and verify its authenticity using the cryptographic methods described in Section 5.5.

Problem arises when the credential of a parent on an attribute tree is same as that of its child. In this case, the child can successfully generate CT'_{i,τ_i} for (the ciphertexts of) every event forwarded by the parent and thus can infer that the credential of the parent is same as its own. To prevent this information leak, OHF strategy enforces random link padding, i.e., each parent sends dummy ciphertexts (that cannot be decrypted by its children) according to the randomly generated schedule such as poison distribution [WMS08].

Figure 5.8 shows secure event routing using one hop flooding (OHF) strategy on an attribute tree associated with a numeric attribute.

5.6.4.2 Multi-credential Routing

In one hop flooding (OHF) strategy, a child may have finer credential than its parent on an attribute tree and thus may receive many false positives.

Multi-credential routing (MCR) strategy targets reduction in false positives by enabling parents to forward only those events on each attribute tree that match the credentials of their children. However, the parents are only informed about the hidden credentials of their children and events are forwarded by matching credentials encoded in the ciphertexts against the hidden credentials of the children by using Equation 5.16. In particular, every child subscriber s on an attribute tree A_i informs each parent p about the private key $Pr^{s}_{i,\tau_i}[3]$ of the credential $Cred_{i,\tau_i}$ associated with the overlay connection to p.[**] Upon receiving an event on an attribute tree A_i, a parent p forwards the event to a child s if Equation 5.16 verifies that one of the credentials under which the event is encrypted matches the credential of the private key $Pr^{s}_{i,j}[3]$ submitted by the child s.

Although the actual credentials of children are hidden from the parent peers by the use of $Pr^{s}_{i,j}[3]$ keys. Nevertheless, the hidden credentials (i.e., $Pr^{s}_{i,j}[3]$ keys) are not adequate to ensure weak subscription confidentiality. This is because a parent decrypts every event which it forwards to its children and therefore, can eventually discover their credentials. For example, a parent peer with credential 0 (i.e., authorized to decrypt events encrypted with credential 0) can maintain a list of all events forwarded to a child and can easily discover that the credential of the child is 01 if the Equation 5.16 only allows forwarding of the events encrypted with credential 01.

To preserve weak subscription confidentiality, subscribers divide the original credential(s) for each attribute of their subscriptions into a number of fine granular credentials and $Pr^{s}_{i,j}[3]$ key for each (fine granular) credential is forwarded to a separate parent in the corresponding attribute tree. For example, credential 1 for a numeric attribute can be divided into three credentials 10, 110 and 111, and a separate parent connection can be maintained (by forwarding $Pr^{s}_{i,j}[3]$ key) for each credential (obtained as a result of division). This enables that the exact credential(s) of an attribute of a subscription cannot be determined unless multiple parents (with knowledge about the individual credentials) collude with each other. To ensure that a subscriber always connects to a distinct parent for each of its credentials, techniques such as broadcast revocation can be used [LSW10]. It is also important to mention that the subscribers cannot generate $Pr^{s}_{i,j}[3]$ keys for the fine granular credentials obtained as a result of dividing the original credential(s) and therefore, should contact the key server for the creation of $Pr^{s}_{i,j}[3]$ keys. This step can be performed at the same time when a new subscriber authorizes itself to the key server.

[**]The $Pr^{s}_{i,\tau_i}[3]$ keys of subscribers cannot be used to decrypt events. These keys are only created to facilitate the routing of encrypted events from publishers to subscribers (cf. Section 5.5.3).

A subscriber maintains atleast $k_{\mathcal{D}}$ credentials for each attribute of its subscription.[††] The parameter $k_{\mathcal{D}}$ can be defined by the system or selected by each subscriber independently depending on its confidentiality requirements. The complete subscription of the subscriber cannot be determined unless $d.k_{\mathcal{D}}$ parents collude with each other.

Problem arises when a subscriber already holds the finest granularity credential for an attribute which cannot be further divided. This problem can be addressed in two ways. First, by decreasing the expressiveness of subscriptions, i.e., restricting the finest (or the smallest) value addressed by the credentials assigned to an attribute. For instance, the smallest addressable value of a numeric attribute can be restricted to 4, so that a credential (assigned to the attribute) can be further divided into $k_{\mathcal{D}} = 4$ credentials, each addressing the value at the granularity of 1. Second, OHF strategy can be utilized for hiding the finest granularity credentials, whereby a subscriber does not reveal the ($Pr^s_{i,j}[3]$ key of) finest credential to its parent and thus receives all the events matched by the credential of the parent.

5.6.5 Analysis of Subscription Confidentiality

In this section, we establish that the proposed publish/subscribe overlay network preserves weaker notion of subscription confidentiality (cf. Definition 5.6.1) by showing the correctness of secure overlay maintenance (cf. Section 5.6.3) and secure event dissemination (cf. Section 5.6.4) algorithms. Moreover, we investigate traffic analysis [PW85, KAP02] and timing [Ray01] attacks on subscription confidentiality.

5.6.5.1 Correctness

For simplicity and without loss of generality, we discuss the correctness of the proposed publish/subscribe overlay maintenance and event dissemination algorithms w.r.t. a single credential associated with an attribute A_i. In particular, we show that the proposed algorithms preserve weak subscription confidentiality by arguing that the information revealed about the credential of a subscriber associated with an attribute A_i is limited to that permitted by Definition 5.6.1.

For an attribute A_i, let $\mathbb{S}_{\preceq}$ be the set of peers in the system whose credentials cover the credential of the subscriber s_1. Let $\mathbb{S}_{\succeq}$ denote the set of subscribers whose credentials are covered by the credential of the subscriber s_1 and $\mathbb{S}_{\not\succ}$ denote the set of subscribers whose credentials have *no* containment relation with the credential of the subscriber s_1. The information revealed about the credential of s_1 to the peers participating on the tree associated with the attribute A_i can be described in three lemmas.

Lemma 1 *Any peer $s' \in \mathbb{S}_{\preceq}$ can infer that the credential of s_1 is either finer or equal to its own credential.*

[††]A subscriber already maintains $\log_2(\mathbb{Z}_i)$ credentials for an attribute A_i in worst case.

Clearly, the information about the credential of s_1 can be revealed to a peer $s^{'}$ either from the connection request (CR) messages of the overlay maintenance protocol or during the routing (dissemination) of events.

Information Revealed by Overlay Maintenance: To determine the exact credential of s_1 from its CR message, a peer $s^{'} \in \mathbb{S}_{\preceq}$ should successfully decrypt all valid ciphertexts (i.e., ciphertexts which are not generated randomly) in the CR (of s_1) so that the random ciphertexts are identified (cf. Section 5.6.3). Identity-based encryption (IBE), however, ensures that $s^{'}$ can only decrypt that ciphertext in the CR which is encrypted using the public key ($Pu^{s}_{i,j}$) corresponding to its own credential (i.e., $s^{'}$ possesses the private keys to decrypt the ciphertext).

Moreover, ciphertexts in CR are encrypted using different random keys (SW) and reordered by shuffling. This ensures that the exact credential of s_1 cannot be revealed to $s^{'}$ either from the discovery of a random key (SW) from the successful decryption of one of the ciphertexts in CR or from the position of ciphertexts (associated with different credentials) in CR, as discussed in Section 5.6.3.

Likewise, the requirement that CR should be signed from the key server abstains $s^{'}$ from generating false (engineered) connection request (CR) messages and forwarding them to s_1 in order to discover its exact credential.

In conclusion, $s^{'}$ cannot determine the exact credential of s_1 from the CR messages forwarded by the overlay maintenance protocol. Any peer $s^{'} \in \mathbb{S}_{\preceq}$ can only decrypt one of the ciphertexts in the CR of s_1, and can only guess that the credential of s_1 is either finer or equal to its own credential.

Information Revealed by Event Dissemination: The overlay maintenance algorithm ensures that peers are arranged in the tree associated with the attribute A_i according to the containment relationship between their credentials and therefore, s_1 can only connect (in the tree) as a child of a peer $s^{'} \in \mathbb{S}_{\preceq}$ in order to receive the events matching its credential.

Two different strategies can be used to forward the events received by the peer $s^{'}$ (from the parent on the tree) to its child s_1, as mentioned in Section 5.6.4. In one hop flooding (OHF) strategy, $s^{'}$ forwards all the events which match its own credential to the child s_1. Therefore, $s^{'}$ cannot infer more information except that the credential of s_1 is either finer or equal to its own credential.

In multi-credential routing (MCR) strategy, only a part of the actual (or exact) credential of s_1 can be revealed to the parent $s^{'}$, i.e., a credential with finer granularity can be revealed. This restricts $s^{'}$ from guessing any information about the exact credential of s_1. The exact credential of s_1 may be finer, equal or coarser than the credential of $s^{'}$. In order to detect the exact credential of s_1, the peer $s^{'}$ should collude with other parents of s_1 on the tree to discover the rest of the finer credentials that make up the credential of s_1.

Finally, it is worth noting that subscription confidentiality in general decreases with the decrease in granularity of the credential of $s^{'}$, i.e., $s^{'}$ with the credential of finest granularity can determine the subscription of peer s_1 with certainty. The multi-credential routing (MCR) strategy is better in mitigating this information leak in comparison to the one hop flooding (OHF) strategy.

Lemma 2 *Any peer $s^{''} \in \mathbb{S}_{\succeq}$ can infer that the credential of s_1 is either coarser or equal to its own credential.*

Information Revealed by Overlay Maintenance: In this case, the credential of peer $s^{''}$ is finer than the credential of s_1. This restricts $s^{''}$ from successfully decrypting any ciphertext in the CR of s_1 because the CR contains valid ciphertexts only for those credentials which cover the credential of s_1. Moreover, the construction of CR message forbids $s^{''}$ from inferring any information about the credential of s_1, as described earlier for Lemma 1. Hence, $s^{''}$ can only guess from the CR message of s_1 that the credential of s_1 is either coarser or is not in a containment relationship to its own credential.

On the contrary, s_1 can decrypt one of the ciphertexts included in the CR message of $s^{''}$ and therefore, s_1 can be selected as a parent of $s^{''}$ in the tree by the overlay maintenance algorithm. In this case, $s^{''}$ can infer that the credential of its parent s_1 is either coarser or equal.

Information Revealed by Event Dissemination: According to the overlay maintenance algorithm, $s^{''}$ can only connect in the tree as a child of s_1 to receive events.

In one hop flooding (OHF) strategy, $s^{''}$ receives all those events that match the credential of s_1. If all the events received from s_1 also match the credential of $s^{''}$ then $s^{''}$ can easily deduce that the credential of s_1 is same as its own. However, due to random link padding, i.e., dummy events forwarded by s_1 along with the actual event traffic, $s^{''}$ can only guess that the credential of s_1 is either coarser or same as its own credential.

Similar to Lemma 1, $s^{''}$ cannot determine the exact credential of s_1 in the presence of multi-credential routing (MCR) strategy, unless it (i.e., $s^{''}$) collude with other peers.

Again it is interesting to point out that subscription confidentiality decreases with the increase in granularity of the credential of $s^{''}$, i.e., $s^{''}$ with the most coarse credential can exactly determine the subscription of s_1. For example, if $s^{''}$ has subscribed to the whole domain of the attribute and the secure overlay maintenance protocol allows it to connect to s_1, then s_1 has also subscribed for the whole domain. As before, multi-credential routing (MCR) strategy can be employed in place of OHF to mitigate this information leak.

Lemma 3 *Any peer $s^{'''} \in \mathbb{S}_{\nsucc}$ on the tree associated with the attribute A_i can infer that credential of s_1 is not in any containment relationship with its own credential.*

Information Revealed by Overlay Maintenance: Similar to Lemma 2, $s^{'''}$ cannot decrypt any ciphertext included in the CR of s_1. This is because the credentials of $s^{'''}$ and s_1 are not in a containment relationship and therefore, CR (of s_1) does not contain a ciphertext encrypted using the public key that corresponds to the credential of $s^{'''}$. Hence, $s^{'''}$ can atmost infer that the credential of s_1 is not in a containment relationship to its own credential.

Information Revealed by Event Dissemination: The overlay maintenance algorithm ensures that $s^{'''}$ is not connected to s_1 as a parent or a child in the tree associated with the attribute A_i. Therefore, $s^{'''}$ neither forwards nor receives events from s_1 and can only infer that the credential of s_1 has no containment relationship to its own credential.

5.6.5.2 Attacks on Subscription Confidentiality

Here, we present traffic analysis [PW85, KAP02] and timing attacks [Ray01] on subscription confidentiality and describe possible remedies against those attacks.

Traffic Analysis Attack: The communication patterns arise from forwarding connection request (CR) messages and routing events.[‡‡] An attacker with access to the communication links (or channels) can deduce the containment relations between the credentials of different (subscriber) peers by observing these communication patterns. It is important to stress here that the containment relations alone do not reveal much information about the actual credentials of the (subscriber) peers, unless a considerable percentage of malicious subscribers either disclose their credentials to the attacker or collude together to instantiate an attack (cf. Section 5.7.3).

The traffic analysis attack is avoided in our system by making the routing of events and CR messages indistinguishable by altering the bit sequence of each event/CR message at the intermediate hops on its path (in the overlay network) from the source to the destination(s). The bit sequence can be easily altered by enabling point-to-point encryption between the pair of peers directly connected via overlay links, for instance, by using transport layer mechanisms such as TLS [DR08, TP11] or SSL [FKK11].

Timing Attack: Even though the events and the CR messages are made indistinguishable by altering their bit sequence (as mentioned in the traffic analysis attack), an advanced attacker might still be able to distinguish between different events (or CR messages) by taking into account the delays on the communication links. For instance, given the set of events published into the overlay network along with their arrival times and the set of events received by the subscribers along with their reception time, an

[‡‡]Such communication patterns also arise in the broker network, i.e., path followed by an event from the publisher through intermediate brokers to the relevant subscribers. However, existing solutions toward secure publish/subscribe systems [RR06, IRC10, SL07] do not address the information leak due to these communication patterns.

Encryption(E)	10KB/sec
Decryption(D)	10KB/sec
Signature(S)	158 sign/sec
Verification(V)	52 verify/sec

Table 5.2: Throughput of cryptographic primitives.

attacker might be able to correlate the events in the two sets by using network delay information. Such an attack is referred as *timing attack* in literature [Ray01].

A possible remedy against timing attack is to enforce constant or random link padding [SBS02, Ray01]. The constant link padding scheme maintains constant flow of event messages on each link in the overlay network, whereas the random padding scheme sends event messages according to a randomly generated schedule [WMS08]. The one hop flooding (OHF) strategy employs random link padding scheme (cf. Section 5.6.4.1) and therefore, timing attacks are not possible. Likewise, random link padding scheme can also be utilized for the multi-credential routing (MCR) strategy.

Apart from using the link padding scheme, anonymization techniques such as a mix network [BFK01] or onion routing [DMS04] can also be employed to protect the privacy of peers in the overlay network and thus avoid the information leak by means of timing attack. However, anonymization of all the overlay links (i.e., anonymizing the participant peers of each overlay link) is not necessary. Our evaluations show that in the absence of random link padding scheme and in the presence of significantly large percentage of malicious subscribers (i.e., 25%), almost 100% subscription confidentiality can be achieved in a publish/subscribe system with 8 attributes by anonymizing the participants of only 20% links in the overlay network (cf. Section 5.7.3).

5.7 Performance Evaluations

Similar to EventGuard [SL05], we evaluate our solution in two aspects: i) quantifying the overhead of our cryptographic primitives, and ii) benchmarking the performance of our secure publish/subscribe system. Moreover, we analyse the attacks on subscription confidentiality, as described in Section 5.6.5.2.

5.7.0.3 Experimental Setup

Simulations are performed using PeerSim [JMJV], a large-scale P2P discrete event simulator. Simulations are performed for up to $N = 2,048$ peers. The out-degree constraints of the peers are always chosen as $\log_2(N)$. Similar to EventGuard [SL05], the delays between the communication links are chosen in the range [24ms, 134ms].

Operation	Time(ms)
Encryption(E)	$6.9 + d \times 5.4$
Signature(S)	$d \times 6.32$
Decryption(D)	$6.2 + d \times 6.1$
Verification(V)	$19.3 + d \times 0.001$

Table 5.3: Computation times (in ms) from the perspective of publishers and subscribers.

The event space has up to $d = 16$ different attributes. The data type of each attribute is Integer, and the domain of each attribute is the range $[1, 16]$. We evaluate the system performance under uniform subscription and advertisement workloads; and with a uniform event distribution. The number of malicious peers vary between 5% and 25%. For a multi-credential enabled event dissemination approach (MCR), every peer maintains atleast three credentials (i.e., $k_{\mathcal{D}} = 3$) for each of its subscribed attributes.

The security mechanisms are implemented by the Pairing-based cryptography (PBC) library [Lyn10]. The implementation uses a 160-bit elliptic curve group based on the supersingular curve $y^2 = x^3 + x$ over a 512-bit finite field.

5.7.1 Performance of Cryptographic Primitives

In this section, we measure the computational overhead of our security methods. All of our measurements were made on a 2.00 GHz Intel Centrino Duo with 2GB RAM, running Ubuntu 9. Table 5.2 shows the throughput of the cryptographic primitives to perform encryption, decryption, signature and verification. All reporting values are averaged over 1000 measurements. In our system, pairing-based encryption is used to encrypt a random key SK, which is later used to decrypt the actual event using symmetric encryption (cf. Section 5.5.4). Therefore, the message size is kept 128 bytes as this key length is good enough for most symmetric encryption algorithms. Table 5.3 shows the computational overhead (in ms) from the perspective of publishers and subscribers in our system. In general, the cost of verification is high due to the fact that it involves the computationally expensive bilinear pairing operations. This justifies our proposal to send the public keys of the publisher's credentials along with the ciphertext (cf. Section 5.5.4) and to perform combined verification of the signatures of all the attributes at the same time (cf. Section 5.5.5). The other possibility is to verify the signature of each attribute separately, however as mentioned in Section 5.5.5 combined approach (i.e., verification at the same time) involves only four bilinear pairing operations, whereas $3d + 1$ pairing operations are required for separate verification.

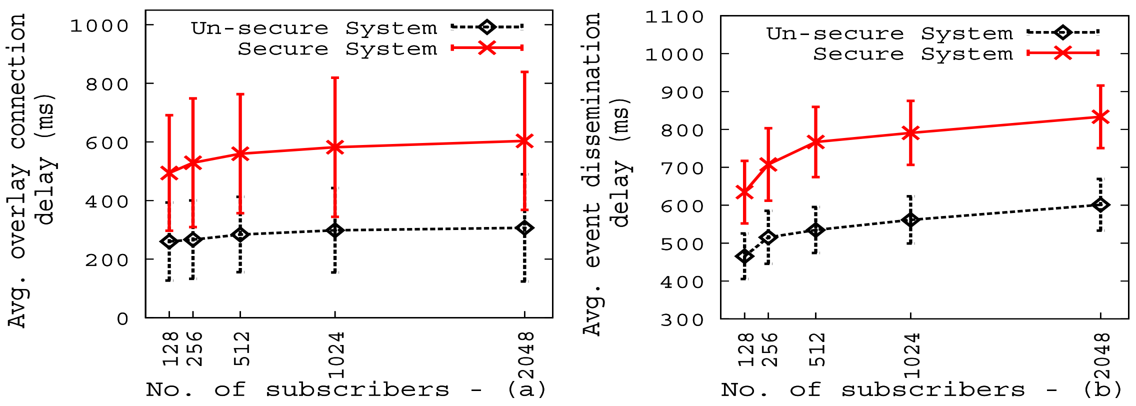


Figure 5.9: Performance of publish/subscribe system.

5.7.2 Performance of Publish/Subscribe System

We evaluate the performance of our system w.r.t. the overlay construction time and the event dissemination delays.

In Figure 5.9(a), we measure the average delay experienced by each subscriber to connect to a suitable position in an attribute tree. Delay is measured from the time a subscriber sends connection request message to a random peer in the tree till the time the connection is actually established. The evaluations are performed only for a single attribute tree. Figure 5.9(a) shows that the average connection time (delay) increases with the number of peers in the system because of the increase in the height of the attribute tree (each new hop increases the network delay as well as time to apply security methods). However, the corresponding increase in the connection time (delay) is small due to the fact that the overall out-degree also increases with the number of peers, resulting in only a small increase in the height of tree. Furthermore, Figure 5.9(a) shows that there is an overhead of approximately 230-300 ms due to security mechanisms. Our evaluations with higher number of attributes indicate that the average connection delay experienced by a subscriber is independent to the number of attributes. This is because each attribute tree is created in parallel and a subscriber sends connection request to connect multiple attribute trees at the same time.

Figure 5.9(b) measures the average time needed by the event to be disseminated to all the relevant subscribers in the system. For each subscriber, the time is measured from the dissemination of the event by the publisher till it is successfully decrypted and verified by the subscriber. For the experiment, 160 publishers are introduced in the system and each published 10 events. Figure 5.9(b) shows that the average time to disseminate an event increases with the number of peers in the system because of the increase in number of the relevant subscribers as well as the height of the dissemination tree. Similar to the previous results, there is an overhead of approximately 150-250 ms due to security mechanisms. Furthermore, Figure 5.9(b) displays that the variation

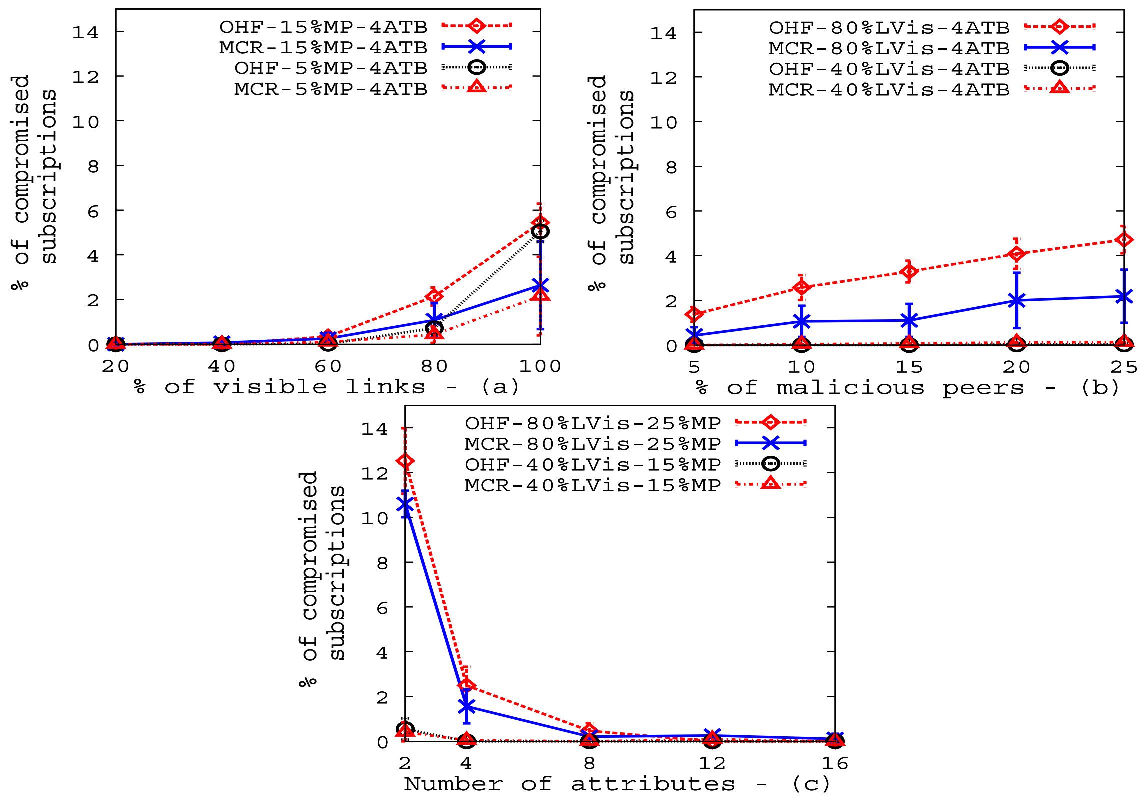


Figure 5.10: Analysis of attacks on subscription confidentiality.

in the event dissemination times (i.e., standard deviation) is less than the variation in the overlay construction times (depicted in Figure 5.9(a)) because the overall number of subscribers receiving each published event is approximately same due to uniform subscription workload.

5.7.3 Analysis of Subscription Confidentiality

As mentioned in Section 5.6.5.2, random link padding scheme can be used to avoid traffic analysis and timing attacks on subscription confidentiality. The link padding however is expensive in terms of communication overhead as it requires dissemination of dummy traffic on the overlay links. Therefore, in this section we analyse the attacks on subscription confidentiality in the absence of link padding, i.e., OHF and MCR event dissemination strategies are used without random link padding scheme.

In particular, Figure 5.10 analyses the subscription information revealed to an attacker (or a group of attackers) as a result of observing the communication patterns that arise because of the secure overlay maintenance protocol and the event dissemination. In the figure, *LVis* denotes the visibility of the communication links (or channels) to the attacker, *MP* denotes the percentage of malicious peers, and *ATB* denotes the number of attributes in the content-based model.

In general, the subscription information revealed to an attacker depends on the following four factors.

Visibility of the Communication Links: An attacker should monitor the traffic on the communication links (or channels) to infer the containment relations between the credentials of different subscriber peers. However, in reality an attacker may not be able to monitor the traffic on all the communication links. A communication link is considered invisible to an attacker (or in other words not monitored by an attacker) for two reasons: i) it belongs to a different administrative domain not under the influence of the attacker, or ii) the communication end-points (i.e., the pair of peers using the communication link) are anonymized (hidden), by using the techniques mentioned in Section 5.6.5, making the traffic monitoring on the link useless. Figure 5.10(a) shows that in the presence of a fixed percentage of malicious peers that reveal their subscription credentials to the attacker, the percentage of compromised subscriptions (i.e., the subscriptions with all the credentials revealed to the attacker) increases with the increase in the percentage of links monitored by the attacker. Note that in the Figure 5.10(a), the link visibility of 100% along with 15% malicious peers mimics very hostile environment. Moreover, the percentage of compromised subscriptions does not include the subscriptions revealed by the malicious peers.

Percentage of Malicious Peers: As mentioned in Section 5.6.5, the containment relationship itself does not reveal much information about the actual credentials of the (subscriber) peers unless a considerable percentage of malicious subscribers either reveal their own credentials to the attacker or collude together to instantiate an attack. Figure 5.10(b) shows that the percentage of compromised subscriptions increases slightly with the increase in the number of malicious peers. Again, $10\% - 25\%$ peers revealing their subscription credentials to an attacker represent very conservative (strict) settings.

Number of Credentials: Increasing the number of credentials maintained by the subscribers decreases the likelihood for an attacker to identify the exact subscriptions. For the same reason, the MCR approach, where each subscriber maintains atleast 3 credentials per attribute, performs better than OHF, as shown in Figures 5.10(a) and (b).

Number of Attributes: Finally, the number of attributes in the content-based model influences the percentage of compromised subscriptions, as shown in Figure 5.10(c). The figure shows that for a moderate scenario, where 40% communication links are monitored by the attacker and 15% peers maliciously reveal their subscription credentials, almost 100% subscription confidentiality can be achieved in the presence of just 3 to 4 attributes.

5.8 Related Work

Related work to the contributions in this chapter can be classified into the following two areas.

5.8.1 Publish/Subscribe Systems

Over the last decade many content-based publish/subscribe systems [CRW01,FCMB06, AGD$^+$06, BKR09, CJ09, JCL$^+$10, SMK12, BGKM07] have evolved. Most systems focus on increasing scalability by reducing the cost of subscription forwarding and event matching. Only a few systems have addressed security issues in a content-based publish/subscribe system. Wang et al. [WCEW02] investigate the security issues and requirements that arise in an internet-scale publish/subscribe system. They concluded that due to loose coupling between publishers and subscribers, many security issues cannot be directly solved by the current technology and require further research.

Hermes [Pie04] proposes a security service that uses role-based access control to authorize subscribers as well as to establish trust in the broker network. Visibility of events and subscriptions to a broker depends on its (broker) trust level. Similarly, Pesonen et al. [PEB07] address the role-based access control in multi-domain publish/subscribe systems by the use of a decentralized trust management.

Opyrchal et al. [OP01] leverage concepts from secure group-based multicast techniques for the secure distribution of events in a publish/subscribe system. They show that previous techniques for dynamic group key management fail in a publish/subscribe scenario, since every event potentially has a different set of interested subscribers. To overcome the problem, they propose a key caching technique. However, brokers are assumed to be completely trustworthy.

Eventguard [SL05, SLI11] provides six guards/components to protect each of the five major publish/subscribe operations (subscribe, unsubscribe, advertise, unadvertise, publish) and routing. It uses ElGamal for encryption and signatures. However, it only supports topic-based routing through the direct use of pseudo random functions.

PSGuard [SL06, SL07] addresses scalable key management in a content-based system by using hierarchical key derivation to associate keys with subscriptions and events. Key derivation algorithms are similar to our credential creation approach, however, our cryptographic primitives and methods to ensure security are completely different. In particular, PSGuard does not address the issues related to the secure routing of events and subscription confidentiality. Moreover, event confidentiality is not properly ensured in case of complex subscriptions, i.e., the keys associated with the predicates in a complex subscription are not bind together as mentioned in Section 5.4.3.

Another drawback with the existing solutions is their assumption about the presence of a broker network [RR06, BESP08, IRC10, SLI11, CGB10, PEB07, NSB12, Pie04, OP01, Khu05]. These solutions are not directly applicable to peer-to-peer environments where subscribers are clustered according to their interests.

5.8.2 Pairing-based Cryptography

The recent progress of pairing-based cryptography motivates many applications built upon Identity-based encryption. Attribute-based encryption [BSW07, GPSW06], is a general form of Identity-based encryption. It allows for a new type of encrypted access control, where the access control policies are either embedded in the user private keys or in the ciphertexts.

Recently, Ion et al. [IRC10, IRC12][§§] propose an approach to enable the confidentiality of events and subscriptions in a content-based publish/subscribe system by using attribute-based encryption. The proposed approach assumes the presence of a network of brokers and is not applicable to our scenario. Moreover, it mainly focuses on the issue of matching encrypted events against encrypted subscriptions without taking into account the degradation in the subscription confidentiality as a result of the communication patterns between publishers and subscribers being observed by an attacker.

Moreover, Shi et al. [SBC+07] and Boneh et al. [BW07] address complex queries such as conjunction, subset and range queries, over encrypted data using Identity-based encryption. In particular, our approach to generate logarithmic ciphertexts and keys for numeric attributes is inspired from the work of Shi et al. [SBC+07] on multi-attribute encryption for range queries. However, both Shi et al. [SBC+07] and Boneh et al. [BW07] address the problem from a pure cryptographic perspective and their proposed solutions are not practical in our scenario. To be more precise, the number of public parameters, encryption cost and ciphertext size for range queries, in the cryptographic construction of Boneh et al. [BW07], increase with the number of attributes (dimensions) and domain of each attribute (number of points in each dimension), i.e., $O(\sum_{i=1}^{d} \mathfrak{Z}_i)$. Similarly, the decryption cost of Shi et al. [SBC+07] is exponential in the number of attributes $O(\prod_{i=1}^{d} \log_2 \mathfrak{Z}_i)$. Therefore, instead of using their cryptographic methods, we derived our methods directly from the attribute-based encryption. Furthermore, the work by Shi et al. [SBC+07] and Boneh et al. [BW07] is not targeted towards content-based systems and does not address the issues related to verification of event authenticity, subscription confidentiality and secure event routing.

5.9 Summary

In this chapter, we have presented a new approach to provide authentication and confidentiality in a broker-less content-based publish/subscribe system. The approach is

[§§]The approach proposed by Ion et al. [IRC10] is published after the publication of the work conducted in this chapter [TKAR10].

highly scalable in terms of number of subscribers and publishers in the system and the number of keys maintained by them. In particular, we have developed mechanisms to assign credentials to publishers and subscribers according to their subscriptions and advertisements. Private keys assigned to publishers and subscribers, and the ciphertexts are labelled with credentials. We adapted techniques from Identity-based encryption, i) to ensure that a particular subscriber can decrypt an event only if there is a match between the credentials associated with the event and its private keys and, ii) to allow subscribers to verify the authenticity of received events. Furthermore, we developed a secure connection protocol and proposed two event dissemination strategies to preserve the weak subscription confidentiality in the presence of semantic clustering of subscribers.

The evaluations demonstrate that supporting security in a broker-less publish/subscribe system is feasible w.r.t. i) throughput of the proposed cryptographic methods, ii) publish/subscribe overlay construction time, and iii) event dissemination delays. We additionally evaluate traffic analysis and timing attacks on subscription confidentiality in the absence of random link padding scheme. The evaluations depict that the multi-credential enabled event dissemination (MCR) approach is more resilient to the attacks on subscription confidentiality in comparison to OHF approach. Moreover, the evaluations show that in the presence of a significantly large percentage of malicious subscribers (i.e., 25%) almost 100% subscription confidentiality can be achieved in a publish/subscribe system with 8 attributes by hiding the communication of only 20% of the links (e.g., by employing anonymization techniques) in the overlay network.

Chapter 6

Summary and Future Work

In this chapter, we summarize the main contributions of this thesis and present a brief discussion on possible future work.

6.1 Summary

Content-based publish/subscribe has gained high popularity for large-scale dissemination of dynamic information. In a content-based system, publishers and subscribers are loosely coupled and exchange information in the form of events. This loose coupling is traditionally ensured by intermediate routing over a broker network. In more recent systems, publishers and subscribers organize themselves in a broker-less routing infrastructure, forming an event forwarding overlay. In a broker-less publish/subscribe system, it is very challenging to efficiently route the events from the publishers to the relevant subscribers, while guaranteeing the expressiveness of content-based subscriptions as well as preserving the scalability of the system. Moreover, support for quality of service, e.g., end-to-end delay, bandwidth etc., and provision of security mechanisms, e.g., authentication, confidentiality etc., instigate many new research challenges. In this context, following are the main contributions and results of this thesis.

- We propose a publish/subscribe system to satisfy the individual delay requirements of a large dynamic set of subscribers without violating their bandwidth constraints (cf. Chapter 2). The proposed system allows subscribers to adjust the granularity of their subscriptions according to their available bandwidth using the spatial indexing method. This i) enables subscribers to improve their individual chance to satisfy their delay requirements, and ii) significantly reduces the complexity in maintaining the publish/subscribe overlay. Subscribers maintain the overlay in a decentralized manner, exclusively establishing connections that satisfy their individual delay requirements, and that provide subscribers only with events which exactly match their subscription granularity. Thus, a subscriber receives no false positives other than those which it is controlling by the subscription granularity.

Our evaluations show that the proposed publish/subscribe system converges to the satisfaction of subscriber-specified delay requirements in very dynamic settings and scales well with the size of the system. Moreover, the evaluations illustrate that allowing subscribers to contribute some of their bandwidth on receiving and forwarding events which do not match their own subscriptions has the benefit of increasing the overall percentage of subscribers whose delay requirements are satisfied. More importantly, the benefit out weights the cost and comes with only a small increase in the overall rate of false positives in the system. For instance, in the scenario where the subscribers have moderate delay requirements, the percentage of satisfied subscribers increases from 65% to 88% with only 2.5% increase in the overall rate of false positives in the system.

- We present an approach to reduce the effect of false positives in a publish/subscribe system by means of subscription clustering, i.e., grouping subscribers with similar subscriptions in a limited number of clusters such that the event dissemination within each cluster is very efficient w.r.t. the number of false positives (cf. Chapter 3). The subscription clustering is formulated as a graph partitioning problem and methods from the spectral graph theory are identified to solve the problem. The centralized spectral methods are very expensive in terms of computation times and memory requirements. Therefore, the centralized spectral methods are adapted to work in a distributed manner. In particular, efficient and scalable methods (and algorithms) are developed to perform linear and non-linear dimensionality reduction as well as k-means clustering, using the hierarchical overlay organization. Finally, mechanisms are presented to create and maintain clusters of subscribers using the developed distributed spectral methods in a highly dynamic P2P-based system.

  Our evaluations show the effectiveness of the proposed centralized spectral methods to identify good quality subscription clusters in comparison to the centroid- [CM07] and overlap-based [BFPB10, RLW+02] clustering approaches used by the state of the art broker-less publish/subscribe systems. Moreover, the evaluations illustrate the capability of the proposed distributed methods to i) drastically lower the time by $86\% - 99\%$ to perform clustering in comparison to the centralized spectral methods, ii) effectively create good quality clusters with accuracy closely approximating the accuracy of the centralized solution, and iii) significantly reduce the cost of event dissemination (in terms of overall number of messages) in a content-based publish/subscribe system, i.e., up to $18\% - 34\%$ improvement in comparison to a widely used P2P-based approach.

- We introduce a novel scheme that exploits the knowledge of the event traffic, user subscriptions and the router-level topology of the underlying network to construct an efficient content-based routing overlay that i) minimizes the stress on the underlay links, and ii) reduces the cost for disseminating events in terms of false positives and end-to-end delays (cf. Chapter 4). The proposed scheme works in two layers, the topology discovery layer and the routing layer. The topology discovery layer employs methods to discover underlay topology between publishers and subscribers in the system in a distributed manner and with low overhead. The routing layer then uses the information of the discovered underlay topology and the proximity between the subscribers to receive similar events to build publish/subscribe routing overlay, as core-based trees. In particular, different core selection strategies with variations in the selection criteria and the performance benefits are proposed, to facilitate the construction of an efficient routing overlay.

Our evaluations show that for Internet-like topologies, the topology discovery layer is capable of lowering physical link stress and reducing relative delay penalty (RDP). For instance, the baseline approach which does not take into account the underlay topology experiences up to 39% rise in RDP and 30% higher stress on the physical links. Moreover, the proposed core-based strategies reduce the cost to disseminate events by up to 49% in comparison to the widely used minimum spanning tree (MST) based routing approach.

- Finally, we present an approach for achieving authentication of publishers and subscribers as well as ensuring confidentiality of events and subscriptions in a broker-less content-based publish/subscribe system (cf. Chapter 5). The proposed approach is scalable in terms of number of subscribers and publishers in the system and the number of keys maintained by them. In particular, methods are developed to create credentials for publishers and subscribers according to their subscriptions and advertisements. Privates keys assigned to publishers and subscribers, and the ciphertexts are labelled with credentials. Techniques from Identity-based encryption are adapted, i) to ensure that a particular subscriber can decrypt an event only if there is a match between the credentials associated with the event and its private keys, and ii) to allow subscribers to verify the authenticity of received events. Moreover, a weaker notion of subscription confidentiality is defined and a secure overlay maintenance protocol is designed to preserve the weak subscription confidentiality in the presence of semantic clustering of subscribers.

  Our evaluations show that providing security is affordable w.r.t. i) throughput of the proposed cryptographic methods, and ii) delays incurred during the construction of the publish/subscribe overlay and the event dissemination. Furthermore, we evaluate attacks on the subscription confidentiality. In particular, we analyse the subscription information revealed to an attacker (or a group of attackers), as a result of observing the communication patterns that arise because of the secure connection protocol and the event dissemination. The evaluations depict that in a publish/subscribe system with 8 attributes and in the presence of a significantly large percentage of malicious subscribers, i.e., 25%, almost 100% subscription confidentiality can be achieved by hiding the communication of only 20% of the links (e.g., by employing anonymization techniques) in the overlay network.

In conclusion, this thesis extends the state of the art by contributing methods and algorithms that provide scalable dissemination of events and fulfil diverse range of non-functional requirements for the applications using content-based publish/subscribe system.

6.2 Future Work

There exists several ways to extend the work in this thesis or in general to perform future research in the area of content-based publish/subscribe. However, in the following, we focus only on the most promising research direction, i.e., software-defined content-based networking.

In Chapter 4, we have shown that organizing peers in a publish/subscribe overlay without the knowledge of the underlying network may results in higher message overhead (bandwidth utilization) and higher end-to-end delays, since such an organization may leads to the forwarding of same messages over the same physical links multiple times. Utilizing methods that explicitly take into account the topology of the underlying physical network to optimize the publish/subscribe overlay are beneficial, as shown in Section 4.6. However, these methods come with an additional cost to infer the underlay topology from the overlay topology.

Therefore, it would be highly attractive to implement content-based routing directly on the network layer. Since changes to existing standard network protocols and hardware seemed to be unrealistic (as the slow support of the highly anticipated IPv6 standard shows), current research refrains from network layer implementations. However, the advent of new networking technologies, namely, software-defined networking and network virtualization, have a potential to dramatically change the picture.

For instance, the adoption of the OpenFlow standard [Com12] in state of the art switches supports a decoupling of the control and data (forwarding) plane. In future content-based publish/subscribe systems, it will be possible to configure the forwarding tables of switches directly and "on-the-fly" using a controller process implemented in software and running on an external host. Currently, this technology is already heavily used to configure communication flows in datacentres or campus networks [AFRR+10]. The OpenFlow standard also gained strong support from network operators and hardware manufacturers. Companies such as IBM, NEC and HP already offer first OpenFlow switches.

Similarly, the advent of virtual networking will allow applications to access and configure a network of virtual routers that in the future even will spread over multiple ISPs. Software-defined networking using OpenFlow will also here be the paradigm to flexibly configure the behaviour of switches, which then can be mapped to their physical counterparts without loss in performance.

These trends will indeed allow to define a general purpose middleware for content-based publish/subscribe that results in comparable performance to network layer implementations, i.e., publish/subscribe systems supporting line-rate message forwarding, microseconds switching delay, and close to optimal bandwidth efficiency not only with respect to the overlay network but also the underlay network topology.

To support our claims, we recently present reference architecture to realize content-based publish/subscribe using OpenFlow specifications [KDTR12, KDT13]. Furthermore, we sketch how the methods (in particular, spatial indexing and subscription clustering methods) proposed in this thesis can be used to perform content-based routing in the reference architecture. However, the concrete realization of these methods in the reference architecture and their evaluations is still a future work.

Bibliography

[ACO$^+$06] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, 2006.

[ADG$^+$04] E. Anceaume, A. K. Datta, M. Gradinariu, G. Simon, and A. Virgillito. DPS: self-* dynamic reliable content-based publish/subscribe system. Technical Report 1665, IRISA, France, 2004.

[AFRR$^+$10] Mohammad Al-Fares, Sivasankar Radhakrishnan, Barath Raghavan, Nelson Huang, and Amin Vahdat. Hedera: dynamic flow scheduling for data center networks. In *Proceedings of the 7th USENIX conference on networked systems design and implementation (NSDI)*, 2010.

[AGD$^+$06] E. Anceaume, M. Gradinariu, A. K. Datta, G. Simon, and A. Virgillito. A semantic overlay for self- peer-to-peer publish/subscribe. In *Proceedings of the 26th IEEE international conference on distributed computing systems (ICDCS)*, Washington, DC, USA, 2006. IEEE Computer Society.

[AKG$^+$09] Osama Abboud, Aleksandra Kovacevic, Kalman Graffi, Konstantin Pussep, and Ralf Steinmetz. Underlay awareness in p2p systems: Techniques and challenges. In *IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, 2009.

[APBSV10] Luciana Arantes, Maria Gradinariu Potop-Butucaru, Pierre Sens, and Mathieu Valero. Enhanced DR-Tree for low latency filtering in publish/subscribe systems. In *Proceedings of the 24th IEEE international conference on advanced information networking and applications (AINA)*, 2010.

[AT06] Ioannis Aekaterinidis and Peter Triantafillou. PastryStrings: A comprehensive content-based publish/subscribe DHT network. In *Proceedings of* *the 26th IEEE international conference on distributed computing systems (ICDCS)*, 2006.

[Awe87] B. Awerbuch. Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems. In *Proceedings of the ACM symposium on theory of computing (STOC)*, 1987.

[BB12] William C. Barker and Elaine B. Barker. SP 800-67 Rev. 1. Recommendation for the triple data encryption algorithm (TDEA) block cipher. Technical report, National Institute of Standards & Technology, 2012.

[BBC$^+$98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated service. *RFC 2475*, December 1998.

[BBK02] Suman Banerjee, Bobby Bhattacharjee, and Christopher Kommareddy. Scalable application layer multicast. *ACM SIGCOMM Computer Communication Review*, 4:205–217, 2002.

[BBQ$^+$07] Roberto Baldoni, Roberto Beraldi, Vivien Quema, Leonardo Querzoni, and Sara Tucci-Piergiovanni. TERA: topic-based event routing for peer-to-peer architectures. In *Proceedings of the inaugural international conference on distributed event-based systems (DEBS)*, 2007.

[BBQV07] Roberto Baldoni, Roberto Beraldi, Leonardo Querzoni, and Antonino Virgillito. Efficient publish/subscribe through a self-organizing broker overlay and its application to SIENA. *The Computer Journal*, 50:444–459, 2007.

[BCM$^+$99] Guruduth Banavar, Tushar Deepak Chandra, Bodhi Mukherjee, Jay Nagarajarao, Robert E. Strom, and Daniel C. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *Proceedings of the 19th IEEE international conference on distributed computing systems (ICDCS)*, 1999.

[BCOP04] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques on Advances in cryptology (EUROCRYPT)*, 2004.

[BCS94] R. Braden, D. Clark, and S. Shenker. Integrated services in the Internet architecture: an overview. *RFC 1633*, June 1994.

[BDFG07] Silvia Bianchi, Ajoy Datta, Pascal Felber, and Maria Gradinariu. Stabilizing peer-to-peer spatial filters. In *Proceedings of the 27th international conference on distributed computing systems (ICDCS)*, Washington, DC, USA, 2007. IEEE Computer Society.

[BESP08] Jean Bacon, David M. Eyers, Jatinder Singh, and Peter R. Pietzuch. Access control in publish/subscribe systems. In *Proceedings of the second international conference on distributed event-based systems (DEBS)*, pages 23–34, New York, NY, USA, 2008. ACM.

[Bet00] Katherine Betz. A scalable stock web service. In *Proceedings of the international workshop on parallel processing (ICPP)*, 2000.

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In *Proceedings of the international cryptology conference on advances in cryptology*, 2001.

[BF03] Dan Boneh and Matt Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, 32:586–615, 2003.

[BFG07] Silvia Bianchi, Pascal Felber, and Maria Gradinariu. Content-based publish/subscribe using distributed R-Trees. In *Proceedings of the 13th international conference on parallel computing (Euro-Par)*, 2007.

[BFK01] Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web MIXes: A system for anonymous and unobservable Internet access. In *Proceedings of the international workshop on designing privacy enhancing technologies: design issues in anonymity and unobservability*, 2001.

[BFM06] Stefan Behnel, Ludger Fiege, and Gero Muehl. On quality-of-service and publish-subscribe. In *Proceedings of the 26th international conference on distributed computing systems workshops*. IEEE Computer Society, 2006.

[BFPB10] Silvia Bianchi, Pascal Felber, and Maria Gradinariu Potop-Butucaru. Stabilizing distributed R-Trees for peer-to-peer content routing. *IEEE Transactions on Parallel and Distributed Systems*, 21:1175–1187, 2010.

[BGKM07] Sebastien Baehni, Rachid Guerraoui, Boris Koldehofe, and Maxime Monod. Towards fair event dissemination. In *Proceedings of the 27th International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2007.

[BHMW08] Roland Bless, Christian Hübsch, Sebastian Mies, and Oliver Waldhorst. The underlay abstraction in the spontaneous virtual networks (SpoVNet) architecture. In *Proceedings of 4th EuroNGI conference on next generation Internet networks (NGI)*, 2008.

[BKR08] Jorge A. Briones, Boris Koldehofe, and Kurt Rothermel. SPINE: Publish/subscribe for wireless mesh networks through self-managed intersecting paths. In *Proceedings of the 8th IEEE international conference on innovative internet community systems (I2CS)*. IEEE Computer Society, June 2008.

[BKR09] Jorge A. Briones, Boris Koldehofe, and Kurt Rothermel. Spine : Adaptive publish/subscribe for wireless mesh networks. *Studia Informatika Universalis*, 7:320 – 353, 2009.

[BMHW11] R. Bless, C. Mayer, C. Hübsch, and O. Waldhorst. *Future Internet Services and Service Architectures*, chapter SpoVNet: An Architecture for

Easy Creation and Deployment of Service Overlays, pages 23–47. River Publishers, 2011.

[BRS02] Ashwin R. Bharambe, Sanjay Rao, and Srinivasan Seshan. Mercury: a scalable publish-subscribe system for Internet games. In *Proceedings of the 1st workshop on Network and system support for games (NetGames)*, 2002.

[BSB$^+$02] Sumeer Bhola, Robert E. Strom, Saurabh Bagchi, Yuanyuan Zhao, and Joshua S. Auerbach. Exactly-once delivery in a content-based publish-subscribe system. In *Proceedings of the international conference on dependable systems and networks (DSN)*, pages 7–16, Washington, DC, USA, 2002. IEEE Computer Society.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the IEEE symposium on security and privacy*, Washington, DC, USA, 2007. IEEE Computer Society.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *Proceedings of the 4th theory of cryptography conference (TCC)*, 2007.

[CAR05] Nuno Carvalho, Filipe Araujo, and Luis Rodrigues. Scalable QoS-based event routing in publish-subscribe systems. In *Proceedings of the 4th IEEE international symposium on network computing and applications.* IEEE Computer Society, 2005.

[CC00] Trevor F. Cox and M.A.A. Cox. *Multidimensional Scaling, Second Edition.* Chapman and Hall/CRC, 2000.

[CCK88] C. Cheng, I. Cimet, and S. Kumar. A protocol to maintain a minimum spanning tree in a dynamic topology. *ACM SIGCOMM Computer Communication Review*, 18:330–337, 1988.

[CCRK04] Manuel Costa, Miguel Castro, Antony Rowstron, and Peter Key. PIC: Practical Internet coordinates for distance estimation. In *Proceedings of the 24th international conference on distributed computing systems (ICDCS)*, 2004.

[CDN08] Gianpaolo Cugola and Elisabetta Di Nitto. On adopting content-based routing in service-oriented architectures. *Information and Software Technology*, 50:22–35, 2008.

[CDNF01] Gianpaolo Cugola, Elisabetta Di Nitto, and Alfonso Fuggetta. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Transactions on Software Engineering*, 27:827–850, 2001.

[CDZ] Kenneth L. Calvert, Matthew B. Doar, and Ellen W. Zegura. Modeling topology of large internetworks. http://www.cc.gatech.edu/projects/gtitm/.

[CF05] Raphaël Chand and Pascal Felber. Semantic peer-to-peer overlays for publish/subscribe networks. In *Proceedings of the 11th international Euro-Par conference on parallel processing*, pages 1194–1204, 2005.

[CFSD90] J. D. Case, M. Fedor, M. L. Schoffstall, and J. Davin. Simple network management protocol (SNMP). *RFC 1157*, 1990.

[CGB10] Sunoh Choi, Gabriel Ghinita, and Elisa Bertino. A privacy-enhancing content-based publish/subscribe system using scalar product preserving transformations. In *Proceedings of the 21st international conference on database and expert systems applications: Part I*, 2010.

[Che11] Alex King Yeung Cheung. *Resource Allocation Algorithms for Event-Based Enterprise Systems*. PhD thesis, Department of Electrical and Computer Engineering, University of Toronto, 2011.

[CHH10] Chun-I CFan, Ling-Ying Huang, and Pei-Hsiu Ho. Anonymous multireceiver identity-based encryption. *IEEE Transactions on Computers*, 59:1239–1249, 2010.

[CJ09] Alex Cheung and Hans-Arno Jacobsen. Publisher relocation algorithms for minimizing delivery delay and message load. Technical report, University of Toronto, 2009.

[CJ11] Alex King Yeung Cheung and Hans-Arno Jacobsen. Green resource allocation algorithms for publish/subscribe systems. In *Proceedings of the 31st international conference on distributed computing systems (ICDCS)*, 2011.

[CKVW06] David Cheng, Ravi Kannan, Santosh Vempala, and Grant Wang. A divide-and-merge methodology for clustering. *ACM Transactions on Database Systems*, 31:1499–1525, 2006.

[Cla06] Kenneth L. Clarkson. Nearest-neighbor searching and metric space dimensions. In *Nearest-neighbor methods for learning and vision: theory and practice*, pages 15 – 59. MIT Press, 2006.

[CLS03] Mao Chen, Andrea LaPaugh, and Jaswinder Pal Singh. Content distribution for publish/subscribe services. In *Proceedings of the ACM/IFIP/USENIX 2003 international conference on middleware*, 2003.

[CM07] Emiliano Casalicchio and Federico Morabito. Distributed subscriptions clustering with limited knowledge sharing for content-based publish/subscribe systems. In *Proceedings of the 6th IEEE international symposium on network computing and applications (NCA)*, 2007.

[CMTV07] Gregory Chockler, Roie Melamed, Yoav Tock, and Roman Vitenberg. SpiderCast: a scalable interest-aware overlay for topic-based pub/sub communication. In *Proceedings of the inaugural international conference on distributed event-based systems (DEBS)*, pages 14–25. ACM, 2007.

[CN05] Yi Cui and Klara Nahrstedt. High-bandwidth routing in dynamic peer-to-peer streaming. In *Proceedings of the ACM workshop on advances in peer-to-peer multimedia streaming*, 2005.

[Com12] ONF Market Education Committee. *Software-defined Networking: The New Norm for Networks.* Open Networking Foundation, 2012.

[CR08] Rui Campos and Manuel Ricardo. A fast algorithm for computing minimum routing cost spanning trees. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 52:3229–3247, 2008.

[CRS$^+$08] Brian F. Cooper, Raghu Ramakrishnan, Utkarsh Srivastava, Adam Silberstein, Philip Bohannon, Hans-Arno Jacobsen, Nick Puz, Daniel Weaver, and Ramana Yerneni. PNUTS: Yahoo!'s hosted data serving platform. *Proceedings of the VLDB Endowment*, pages 1277–1288, 2008.

[CRW00] Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf. Achieving scalability and expressiveness in an Internet-scale event notification service. In *Proceedings of the 19th annual ACM symposium on principles of distributed computing (PODC)*, 2000.

[CRW01] Antonio Carzaniga, David S. Rosenblum, and Alexander L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19:332–383, 2001.

[CRW04] Antonio Carzaniga, Matthew J. Rutherford, and Er L. Wolf. A routing scheme for content-based networking. In *Proceedings of the 23th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*. IEEE, 2004.

[CS04] Fengyun Cao and Jaswinder Pal Singh. Efficient event routing in content-based publish-subscribe service networks. In *Proceedings of the 23th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*. IEEE, 2004.

[CS05] Fengyun Cao and Jaswinder Pal Singh. MEDYM: match-early with dynamic multicast for content-based publish-subscribe networks. In *Proceedings of the ACM/IFIP/USENIX 2005 international conference on middleware*, pages 292–313, New York, NY, USA, 2005. Springer-Verlag.

[DCKM04] Frank Dabek, Russ Cox, Frans Kaashoek, and Robert Morris. Vivaldi: a decentralized network coordinate system. *ACM SIGCOMM Computer Communication Review*, 34:15–26, 2004.

[Dee88] S. E. Deering. Multicast routing in internetworks and extended LANs. *ACM SIGCOMM Computer Communication Review*, 18:55–64, 1988.

[DF07] Benoit Donnet and Timur Friedman. Internet topology discovery: A survey. *IEEE Communications Surveys and Tutorials*, 9:2–15, 2007.

[DGK07] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1944–1957, 2007.

[DGK09] Souptik Datta, Chris Giannella, and Hillol Kargupta. Approximate distributed k-means clustering over a peer-to-peer network. *IEEE Transactions on Knowledge and Data Engineering*, 21:1372–1388, 2009.

[DMS04] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th conference on USENIX security symposium - volume 13*, 2004.

[Dow99a] Allen B. Downey. Clink. http://allendowney.com/research/clink/, 1999.

[Dow99b] Allen B. Downey. Using pathchar to estimate Internet link characteristics. *ACM SIGCOMM Computer Communication Review*, 29:241–250, 1999.

[DR08] T. Dierks and E. Rescorla. The transport layer security (TLS) protocol version 1.2. *RFC 5246*, August 2008.

[DRFC06] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Deployment of an algorithm for large-scale topology discovery. *IEEE Journal on Selected Areas in Communications*, 24:2210–2220, 2006.

[DSF07] Anwitaman Datta, Ion Stoica, and Mike Franklin. LagOver: Latency gradated overlays. In *Proceedings of the 27th international conference on distributed computing systems (ICDCS)*. IEEE Computer Society, 2007.

[ECG09] Christian Esposito, Domenico Cotroneo, and Aniruddha Gokhale. Reliable publish/subscribe middleware for time-sensitive Internet-scale applications. In *Proceedings of the 3rd ACM international conference on distributed event-based systems (DEBS)*, 2009.

[EE98] Guy Eddon and Henry Eddon. *Inside distributed COM.* Microsoft Press, 1998.

[EFGK03] Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys*, 35:114–131, 2003.

[EGH$^+$03] P. Th. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec. Lightweight probabilistic broadcast. *ACM Transactions on Computer Systems*, 21:341–374, 2003.

[Ent09] Symantec Enterprise. Green IT report - regional data United States and Canada. http://www.scribd.com/doc/15860957/2009-Green-IT-Report-Final, May 2009.

[EY39] C. Eckardt and G. Young. A principal axis transformation for non-hermitian matrices. *Bull. Amer. Math Soc.*, 45:118–121, 1939.

[FBCM04] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:214–225, 2004.

[FCMB06] Ludger Fiege, Mariano Cilia, Gero Muhl, and Alejandro Buchmann. Publish/Subscribe grows up: Support for management, visibility control, and heterogeneity. *IEEE Internet Computing*, 10:48–55, 2006.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the Internet topology. *ACM SIGCOMM Computer Communication Review*, 29:251–262, 1999.

[fIDA12] CAIDA: The Cooperative Association for Internet Data Analysis. http://www.caida.org/home/, 2012.

[FKK11] A. Freier, P. Karlton, and P. Kocher. The secure sockets layer (SSL) protocol version 3.0. *RFC 6101*, August 2011.

[FO99] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, 1999.

[FSS09] Haw-ren Fang, Sophia Sakellaridi, and Yousef Saad. Multilevel nonlinear dimensionality reduction for manifold learning. Technical report, Minnesota Supercomputer Institute, University of Minnesota, 2009.

[FSS10] Haw-ren Fang, Sophia Sakellaridi, and Yousef Saad. Multilevel manifold learning with application to spectral clustering. In *Proceedings of the 19th ACM conference on information and knowledge management (CIKM)*, 2010.

[GaR03] Mário Guimarães and Luís Rodrigues. A genetic algorithm for multicast mapping in publish-subscribe systems. In *Proceedings of the 2nd IEEE international symposium on network computing and applications*, 2003.

[GCV$^+$10] Sarunas Girdzijauskas, Gregory Chockler, Ymir Vigfusson, Yoav Tock, and Roie Melamed. Magnet: practical subscription clustering for Internet-scale publish/subscribe. In *Proceedings of the 4th ACM international conference on distributed event-based systems (DEBS)*, 2010.

[GG98] Volker Gaede and Oliver Günther. Multidimensional access methods. *ACM Computer Survey*, 30:170–231, 1998.

[GKK+11] Shuo Guo, Kyriakos Karenos, Minkyong Kim, Hui Lei, and Johnathan Reason. Delay-cognizant reliable delivery for publish/subscribe overlay networks. In *Proceedings of the 31st international conference on distributed computing systems (ICDCS)*, 2011.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on computer and communications security (CCS)*, New York, NY, USA, 2006. ACM.

[Gro05] Vic Grout. Principles of cost minimisation in wireless networks. *Journal of Heuristics*, 11:115–133, 2005.

[GS08] Mehmet Hadi Gunes and Kamil Sarac. Resolving anonymous routers in Internet topology measurement studies. In *Proceedings of the 27th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*, 2008.

[GSAA04] Abhishek Gupta, Ozgur D. Sahin, Divyakant Agrawal, and Amr El Abbadi. Meghdoot: Content-based publish/subscribe over P2P networks. In *Proceedings of the 5th ACM/IFIP/USENIX international conference on middleware*, pages 254–273. Springer-Verlag New York, Inc., 2004.

[GSB+11] Mukul Goyal, Mohd Soperi, Emmanuel Baccelli, Gagan Choudhury, Aman Shaikh, Hossein Hosseini, and Kishor Trivedi. Improving convergence speed and scalability in OSPF: A survey. *IEEE Communications Surveys & Tutorials*, 14:443–463, 2011.

[Gut84] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the ACM SIGMOD international conference on management of data*, 1984.

[Ham12] Keith Hamilton. Web solutions platform (WSP) event system. In *Proceedings of the 6th ACM international conference on distributed event-based systems (DEBS)*, 2012.

[HASG07] Mojtaba Hosseini, Dewan Tanvir Ahmed, Shervin Shirmohammadi, and Nicolas D. Georganas. A survey of application-layer multicast protocols. *IEEE Communications Surveys and Tutorials*, 9:58–74, 2007.

[HHNL09] Dirk Haage, Ralph Holz, Heiko Niedermayer, and Pavel Laskov. CLIO - a cross-layer information service for overlay network optimization. In *Proceedings of the 16th ITG/GI conference on kommunikation in verteilten systemen (KiVS)*, 2009.

[HK07] K. Hammouda and M. Kamel. HP2PC: Scalable hierarchically-distributed peer-to-peer clustering. In *Proceedingsof SIAM international conference on data mining (SDM)*, 2007.

[HMW$^+$05] Joe Hicklin, Cleve Moler, Peter Webb, Ronald F. Boisvert, Bruce Miller, Roldan Pozo, and Karin Remington. JAMA library. http://math.nist.gov/javanumerics/jama/, 2005.

[Hot33] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[HPMC02] Bradley Huffak, Daniel Plummer, David Moore, and k. Claffy. Topology discovery by active probing. In *Proceedings of the symposium on applications and the Internet (SAINT) workshops*, 2002.

[Hu74] T. C. Hu. Optimum communication spanning trees. *SIAM Journal of Computing*, 3:188 – 195, 1974.

[HW09] Christian Hübsch and Oliver P. Waldhorst. Enhancing application layer multicast solutions by wireless underlay support. In *Proceedings of the 16th ITG/GI conference on kommunikation in verteilten systemen (KiVS)*, 2009.

[IRC10] Mihaela Ion, Giovanni Russello, and Bruno Crispo. Supporting publication and subscription confidentiality in pub/sub networks. In *Proceedings of 6th International ICST Conference on Security and Privacy in Communication Networks (SecureComm)*, 2010.

[IRC12] Mihaela Ion, Giovanni Russello, and Bruno Crispo. Design and implementation of a confidentiality and access control solution for publish/subscribe systems. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 56:2014–2037, 2012.

[Jac97] V. Jacobson. Pathchar. http://www.caida.org/tools/utilities/others/pathchar/, 1997.

[JCL$^+$10] Hans-Arno Jacobsen, Alex King Yeung Cheung, Guoli Li, Balasubramaneyam Maniymaran, Vinod Muthusamy, and Reza Sherafat Kazemzadeh. The PADRES publish/subscribe system. In *Principles and Applications of Distributed Event-Based Systems*. IGI Global, 2010.

[JE11] K. R. Jayaram and Patrick Eugster. Split and subsume: Subscription normalization for effective content-based messaging. In *Proceedings of the 31st international conference on distributed computing systems (ICDCS)*, 2011.

[JKS04] Mark Jelasity, Wojtek Kowalczyk, and Maarten van Steen. An approach to massively distributed aggregate computing on peer-to-peer networks. In *Proceedings of the 12th workshop on parallel, distributed and network-based processing (PDP)*. IEEE Computer Society, 2004.

[JMJV] Márk Jelasity, Alberto Montresor, Gian Paolo Jesi, and Spyros Voulgaris. PeerSim: A peer-to-peer simulator. http://peersim.sourceforge.net/.

[JMWB02] Sushant Jain, Ratul Mahajan, David Wetherall, and Gaetano Borriello. Scalable self-organizing overlays. Technical Report UW-CSE 02-02-02, University of Washington, 2002.

[Jol86] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[JPMH07] Michael A. Jaeger, Helge Parzyjegla, Gero Muehl, and Klaus Herrmann. Self-organizing broker topologies for publish/subscribe systems. In *Proceedings of the 2007 ACM symposium on applied computing (SAC)*, pages 543–550. ACM, 2007.

[JTC08a] Xing Jin, Wanqing Tu, and S. H. Gary Chan. Scalable and efficient end-to-end network topology inference. *IEEE Transactions on Parallel and Distributed Systems*, 19:837–850, 2008.

[JTC08b] Xing Jin, Wanqing Tu, and S.-H. Gary Chan. Traceroute-based topology inference without network coordinate estimation. In *Proceedings of IEEE international conference on communications (ICC)*, 2008.

[JVG$^+$07] Márk Jelasity, Spyros Voulgaris, Rachid Guerraoui, Anne-Marie Kermarrec, and Maarten van Steen. Gossip-based peer sampling. *ACM Transactions on Computer Systems*, 25, 2007.

[JWC05] Xing Jin, Yajun Wang, and S.-H. Gary Chan. Fast overlay tree based on efficient end-to-end measurements. In *Proceedings of IEEE international conference on communications (ICC)*, 2005.

[JYCW06] Xing Jin, W. P.K. Yiu, S. H.G. Chan, and Yajun Wang. Network topology inference based on end-to-end measurements. *IEEE Journal on Selected Areas in Communications*, 24:2182–2195, 2006.

[JYCW07] Xing Jin, W. P.K. Yiu, S. H.G. Chan, and Yajun Wang. On maximizing tree bandwidth for topology-aware peer-to-peer streaming. *IEEE Transactions on Multimedia*, 9:1580–1592, 2007.

[KAP02] Dogan Kedogan, Dakshi Agrawal, and Stefan Penz. Limits of anonymity in open environments. In *Revised Papers from the 5th International Workshop on Information Hiding (IH)*, pages 53–69, 2002.

[KCS11] E. Kokiopoulou, J. Chen, and Y. Saad. Trace optimization and eigenproblems in dimension reduction methods. *Numerical Linear Algebra with Applications*, 18:565–602, 2011.

[KDT13] Boris Koldehofe, Frank Dürr, and Muhammad Adnan Tariq. Event-based systems meet software-defined networking. In *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS)*, 2013.

[KDTR12] Boris Koldehofe, Frank Dürr, Muhammad Adnan Tariq, and Kurt Rothermel. The power of software-defined networking: Line-rate content-based routing using OpenFlow. In *Proceedings of the 7th international ACM middleware for next generation Internet computing (MW4NG) workshop of the 13th international middleware conference*, 2012.

[KF02] Minseok Kwon and Sonia Fahmy. Topology-aware overlay networks for group communication. In *Proceedings of the 12th international workshop on network and operating systems support for digital audio and video (NOSSDAV)*, 2002.

[KF05] Minseok Kwon and Sonia Fahmy. Path-aware overlay multicast. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 47:23–45, 2005.

[Khu05] Himanshu Khurana. Scalable security and accounting services for content-based publish/subscribe systems. In *Proceedings of the ACM symposium on applied computing*, 2005.

[KKR08] Gerald G. Koch, Boris Koldehofe, and Kurt Rothermel. Higher confidence in event correlation using uncertainty restrictions. In *Proceedings of the 28th international conference on distributed computing systems workshops (ICDCSW)*, 2008.

[KKR10] Gerald G. Koch, Boris Koldehofe, and Kurt Rothermel. Cordies: expressive event correlation in distributed systems. In *Proceedings of the 4th ACM international conference on distributed event-based systems (DEBS)*, 2010.

[KKS00] Tadayoshi Kohno, John Kelsey, and Bruce Schneier. Preliminary cryptanalysis of reduced-round serpent. In *AES candidate conference*, 2000.

[KKSB07] Srikanth Kandula, Dina Katabi, Shantanu Sinha, and Arthur Berger. Dynamic load balancing without packet reordering. *ACM SIGCOMM Computer Communication Review*, 37:51–62, 2007.

[KL80] V. Klema and A. Laub. The singular value decomposition: Its computation and some applications. *IEEE Transactions on Automatic Control*, 25:164–176, 1980.

[KLO06] John Keeney, David Lewis, and Declan O'Sullivan. A proactive approach to semantically oriented service discovery. In *Proceedings of the 2nd workshop on innovations in web infrastructure (IWI)*, 2006.

[KMN$^+$02] Tapas Kanungo, David M. Mount, Nathan S. Netanyahu, Christine D. Piatko, Ruth Silverman, and Angela Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:881–892, 2002.

[Kna87] Edgar Knapp. Deadlock detection in distributed databases. *ACM Computing Surveys*, 19:303–328, 1987.

[Kol03] Boris Koldehofe. Buffer management in probabilistic peer-to-peer communication protocols. In *Proceedings of the 22nd IEEE Symposium on Reliable Distributed Systems (SRDS)*, pages 76 – 85, 2003.

[Kol04] Boris Koldehofe. Simple gossiping with balls and bins. *Studia Informatica Universalis*, 3:43 – 60, 2004.

[KTKR10] Gerald G. Koch, M. Adnan Tariq, Boris Koldehofe, and Kurt Rothermel. Event processing for large-scale distributed games. In *Proceedings of the 4th international conference on distributed event-based systems (DEBS)*, 2010.

[KV04] Saroja Kanchi and David Vineyard. An optimal distributed algorithm for ALL-pairs shortest-path. *International Journal on Information Theories and Applications*, 11:141–146, 2004.

[LB01] Kevin Lai and Mary Baker. Nettimer: a tool for measuring bottleneck link, bandwidth. In *Proceedings of the 3rd conference on USENIX symposium on Internet technologies and systems*, 2001.

[LHC03] Hyuk Lim, Jennifer C. Hou, and Chong-Ho Choi. Constructing Internet coordinate system based on delay measurement. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, 2003.

[LHLJ12] Wei Li, Songlin Hu, Jintao Li, and Hans-Arno Jacobsen. Community clustering for distributed publish/subscribe systems. In *Proceedings of IEEE international conference on cluster computing*, 2012.

[LMJ08] Guoli Li, Vinod Muthusamy, and Hans-Arno Jacobsen. Adaptive content-based routing in general overlay topologies. In *Proceedings of the 9th ACM/IFIP/USENIX international conference on middleware*, 2008.

[LSW10] Allison Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *Proceedings of the IEEE symposium on security and privacy*, 2010.

[Luc01] David C. Luckham. *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems.* Addison-Wesley Longman Publishing Co., Inc., 2001.

[Lux07] Ulrike Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:394–416, 2007.

[Lyn10] Ben Lynn. The pairing-based cryptography (PBC) library. http://crypto.stanford.edu/pbc/, 2010.

[Mah00] Bruce A. Mah. Pchar. http://kb.pert.geant.net/PERTKB/PcharTool, 2000.

[MC02] René Meier and Vinny Cahill. STEAM: Event-based middleware for wireless ad hoc network. In *Proceedings of the 22nd international conference on distributed computing systems (ICDS)*, 2002.

[MFB02] Gero Muehl, Ludger Fiege, and Alejandro P. Buchmann. Filter similarities in content-based publish/subscribe systems. In *Proceedings of the international conference on architecture of computing systems (ARCS)*, 2002.

[MJ07] Vinod Muthusamy and Hans-Arno Jacobsen. Infrastructure-less content-based publish/subscribe. Technical report, Middleware Systems Research Group, University of Toronto, 2007.

[MKSB07] Shruti P. Mahambre, Madhu Kumar S.D., and Umesh Bellur. A taxonomy of QoS-aware, adaptive event-dissemination middleware. *IEEE Internet Computing*, 11:35–44, 2007.

[MLJ10] Vinod Muthusamy, Haifeng Liu, and Hans-Arno Jacobsen. Predictive publish/subscribe matching. In *Proceedings of the 4th ACM international conference on distributed event-based systems (DEBS)*, 2010.

[MLMB01] Alberto Medina, Anukool Lakhina, Ibrahim Matta, and John Byers. BRITE: Universal topology generation from a user"s perspective. Technical report, Boston University, 2001.

[MM84] Don P. Mitchell and Michael J. Merritt. A distributed algorithm for deadlock detection and resolution. In *Proceedings of the third annual ACM symposium on Principles of distributed computing (PODC)*, 1984.

[MNOP10] Miguel Matos, Ana Nunes, Rui Oliveira, and José Pereira. StAN: exploiting shared interests without disclosing them in gossip-based publish/subscribe. In *Proceedings of the 9th international conference on peer-to-peer systems (IPTPS)*, 2010.

[Mos85] J. E.B. Moss. *Nested transactions: an approach to reliable distributed computing.* Massachusetts Institute of Technology, 1985.

[MSRS09] Anirban Majumder, Nisheeth Shrivastava, Rajeev Rastogi, and Anand Srinivasan. Scalable content-based routing in pub/sub systems. In *Proceedings of the 28th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*, 2009.

[MSS06] Yun Mao, Lawrence K. Saul, and Jonathan M. Smith. IDES: An Internet distance estimation service for large networks. *IEEE Journal on Selected Areas in Communications*, 24:2273–2284, 2006.

[MTKE11] Philip Mildner, Tonio Triebel, Stephan Kopf, and Wolfgang Effelsberg. A scalable peer-to-peer-overlay for real-time massively multiplayer online games. In *Proceedings of the 4th international ICST conference on simulation tools and techniques*, 2011.

[Müh02] Gero Mühl. *Large-scale content-based publish/subscribe systems.* PhD thesis, Department of Computer Science, Darmstadt University of Technology, 2002.

[MVM09] Frederic P. Miller, Agnes F. Vandome, and John McBrewster. *Advanced Encryption Standard.* Alpha Press, 2009.

[MZV07] Tova Milo, Tal Zur, and Elad Verbin. Boosting topic-based publish-subscribe systems with dynamic clustering. In *Proceedings of the ACM international conference on management of data (SIGMOD)*, 2007.

[Net12] SpoVNet: Spontaneous Virtual Networks. http://www.spovnet.de/, 2012.

[NSB12] Mohamed Nabeel, Ning Shang, and Elisa Bertino. Efficient privacy preserving content based publish subscribe systems. In *Proceedings of the 17th ACM symposium on access control models and technologies*, 2012.

[NXC$^+$10] Huazhong Ning, Wei Xu, Yun Chi, Yihong Gong, and Thomas S. Huang. Incremental spectral clustering by efficiently updating the eigen-system. *Pattern Recognition (Elsevier)*, 43:113–127, 2010.

[OMG11] OMG: Object Management Group. Common object request broker architecture specification. http://www.omg.org/spec/CORBA/, 2011.

[OP01] Lukasz Opyrchal and Atul Prakash. Secure distribution of events in content-based publish subscribe systems. In *Proceedings of the 10th conference on USENIX security symposium (SSYM)*, Berkeley, CA, USA, 2001. USENIX Association.

[OS90] Yutaka Ohsawa and Masao Sakauchi. A new tree type data structure with homogeneous nodes suitable for a very large spatial database. In *Proceedings of the 6th international conference on data engineering*, pages 296–303, Washington, DC, USA, 1990. IEEE Computer Society.

[PAc$^+$06] Olga Papaemmanouil, Yanif Ahmad, Uğur Çetintemel, John Jannotti, and Yenel Yildirim. Extensible optimization in overlay dissemination trees. In *Proceedings of the ACM SIGMOD international conference on management of data*, 2006.

[PB02] Peter R. Pietzuch and Jean Bacon. Hermes: A distributed event-based middleware architecture. In *Proceedings of the 22nd international conference on distributed computing systems, workshop (ICDCSW).* IEEE Computer Society, 2002.

[PC05] Olga Papaemmanouil and Ugur Cetintemel. SemCast:: Semantic multicast for content-based data dissemination. In *Proceedings of the 21st international conference on data engineering (ICDE)*, pages 242–253, Washington, DC, USA, 2005. IEEE Computer Society.

[PCW+03] Marcelo Pias, Jon Crowcroft, Steve R. Wilbur, Timothy L. Harris, and Saleem N. Bhatti. Lighthouses for scalable distributed location. In *Proceedings of the 2nd international workshop on peer-to-peer systems (IPTPS)*, 2003.

[PEB07] Lauri I. W. Pesonen, David M. Eyers, and Jean Bacon. Encryption-enforced access control in dynamic multi-domain publish/subscribe networks. In *Proceedings of the inaugural international conference on distributed event-based systems (DEBS)*, 2007.

[Pie04] Peter Pietzuch. *Hermes: A Scalable Event-Based Middleware.* PhD thesis, University of Cambridge, Feb 2004.

[PJL00] Louis Perrochon, Eunhei Jang, and David C. Luckham. Enlisting event patterns for cyber battlefield awareness. *DARPA Information Survivability Conference and Exposition*, 2, 2000.

[PRGK09] Jay A. Patel, Étienne Rivière, Indranil Gupta, and Anne-Marie Kermarrec. Rappel: Exploiting interest and network locality to improve fairness in publish-subscribe systems. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 53:2304–2320, 2009.

[PSB03] Peter R. Pietzuch, Brian Shand, and Jean Bacon. A framework for event composition in distributed systems. In *Proceedings of the ACM/IFIP/USENIX international conference on middleware*, 2003.

[PW85] Andreas Pfitzmann and Michael Waidner. Networks without user observability -- Design options. In *Proceedings of a workshop on the theory and application of cryptographic techniques on Advances in cryptology (EUROCRYPT)*, pages 245–253, 1985.

[PWF07] Gabriel Parmer, Richard West, and Gerald Fry. Scalable overlay multicast tree construction for media streaming. In *Proceedings of the international conference on parallel and distributed processing techniques and applications (PDPTA)*, 2007.

[Que08] Leonardo Querzoni. Interest clustering techniques for efficient event routing in large-scale settings. In *Proceedings of the 2nd international conference on distributed event-based systems (DEBS).* ACM, 2008.

[Ray01] Jean-François Raymond. Traffic analysis: protocols, attacks, design issues, and open problems. In *Proceedings of the international workshop on designing privacy enhancing technologies: design issues in anonymity and unobservability*, 2001.

[RD01] Antony I. T. Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In

Proceedings of the 18th IFIP/ACM international conference on distributed systems platforms, pages 329–350. Springer-Verlag, 2001.

[RFH$^+$01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A scalable content-addressable network. *ACM SIGCOMM Computer Communication Review*, 31:161–172, 2001.

[RKCD01] Antony I. T. Rowstron, Anne-Marie Kermarrec, Miguel Castro, and Peter Druschel. SCRIBE: The design of a large-scale event notification infrastructure. In *Proceedings of 3rd international workshop on networked group communication (NGC)*, pages 30–43. Springer-Verlag, 2001.

[RLW$^+$02] Anton Riabov, Zhen Liu, Joel L. Wolf, Philip S. Yu, and Li Zhang. Clustering algorithms for content-based publication-subscription systems. In *Proceedings of the 22nd international conference on distributed computing systems (ICDCS)*, Washington, DC, USA, 2002. IEEE Computer Society.

[RR06] Costin Raiciu and David S. Rosenblum. Enabling confidentiality in content-based publish/subscribe infrastructures. In *Proceedings of the 2nd IEEE/CreatNet international conference on security and privacy in communication networks (Securecomm)*, 2006.

[RVC01] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. *RFC 3031*, January 2001.

[SB07] Madhu Kumar SD and Umesh Bellur. A distributed algorithm for underlay aware and available overlay formation in event broker networks for publish/subscribe systems. In *Proceedings of the 27th international conference on distributed computing systems workshops (ICDCSW)*, Washington, DC, USA, 2007. IEEE Computer Society.

[SBC$^+$07] Elaine Shi, John Bethencourt, T-H. Hubert Chan, Dawn Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In *Proceedings of IEEE symposium on security and privacy*, pages 350–364, Washington, DC, USA, 2007. IEEE Computer Society.

[SBS02] Rob Sherwood, Bobby Bhattacharjee, and Aravind Srinivasan. P5: A protocol for scalable anonymous communication. In *Proceedings of the IEEE symposium on security and privacy*, 2002.

[Sch96] Scarlet Schwiderski. *Monitoring the behaviour of distributed systems.* PhD thesis, University of Cambridge, Computer Laboratory, 1996.

[SCW10] A. J. Stanford-Clark and G. R. Wightwick. The application of publish/subscribe messaging to environmental, monitoring, and control systems. *IBM Journal of Research and Development*, 54:396–402, 2010.

[SDB08] Madhu Kumar S. D and Umesh Bellur. Availability models for underlay aware overlay networks. In *Proceedings of the 2nd international conference on distributed event-based systems (DEBS)*, 2008.

[SGG$^+$02] A. Shaikh, M. Goyal, A. Greenberg, R. Rajan, and K. K. Ramakrishnan. An OPSF topology server: Design and evaluation. *IEEE Journal on Selected Areas in Communications*, 20:746–755, 2002.

[Sha48] Claude E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 1948.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO on advances in cryptology*, pages 47–53, 1984.

[SKBB09] Kai Sachs, Samuel Kounev, Jean Bacon, and Alejandro Buchmann. Performance evaluation of message-oriented middleware using the SPECjms2007 benchmark. *Performance Evaluation*, 66:410–434, 2009.

[SL05] Mudhakar Srivatsa and Ling Liu. Securing publish-subscribe overlay services with EventGuard. In *Proceedings of the 12th ACM conference on computer and communications security (CCS)*, pages 289–298, New York, NY, USA, 2005. ACM.

[SL06] Mudhakar Srivatsa and Ling Liu. Scalable access control in content-based publish-subscribe systems. Technical report, Georgia Institute of Technology, 2006.

[SL07] Mudhakar Srivatsa and Ling Liu. Secure event dissemination in publish-subscribe networks. In *Proceedings of the 27th international conference on distributed computing systems (ICDCS)*, 2007.

[SLI11] Mudhakar Srivatsa, Ling Liu, and Arun Iyengar. EventGuard: A system architecture for securing publish-subscribe networks. *ACM Transactions on Computer Systems*, 29, 2011.

[SM00] Dawn Song and Jonathan Millen. Secure auctions in a publish/subscribe system. http://www.csl.sri.com/users/millen/papers/dcca8.ps, 2000.

[SMK12] Stephan Schnitzer, Hugo Miranda, and Boris Koldehofe. Content routing algorithms to support publish/subscribe in mobile ad hoc networks. In *Proceedings of the 5th Workshop on Architectures, Services and Applications for the Next Generation Internet (WASA-NGI)*, 2012.

[SMLN$^+$03] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, and Hari Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for Internet applications. *IEEE/ACM Transactions on Networking*, 11:17–32, 2003.

[SMW02] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring ISP topologies with rocketfuel. *ACM SIGCOMM Computer Communication Review*, 32:133–145, 2002.

[SMWA04] Neil Spring, Ratul Mahajan, David Wetherall, and Thomas Anderson. Measuring ISP topologies with rocketfuel. *IEEE/ACM Transactions on Networking*, 12:2–16, 2004.

[SÖM09] Abdullatif Shikfa, Melek Önen, and Refik Molva. Privacy-preserving content-based publish/subscribe networks. In *Emerging challenges for security, privacy and trust, volume 297 of IFIP advances in information and communication technology*, 2009.

[Spo07] SpoVNet Consortium. SpoVNet: An architecture for supporting future Internet applications. In *Proceedings of 7th Würzburg workshop on IP: joint EuroFGI and ITG workshop on visions of future generation networks*, 2007.

[ST00] Noam Slonim and Naftali Tishby. Document clustering using word clusters via the information bottleneck method. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, 2000.

[Sun04] Sun Microsystems. Java remote method invocation specification. http://java.sun.com/j2se/1.5/pdf/rmi-spec-1.5.0.pdf, 2004.

[TA90] B. H. Tay and A. L. Ananda. A survey of remote procedure calls. *ACM SIGOPS Operating Systems Review*, 24:68–79, 1990.

[Taj11] Naweed Tajuddin. Techniques for overlay design of content-based publish/subscribe systems. Master's thesis, Department of Electrical and Computer Engineering, University of Toronto, 2011.

[Tar06] Sasu Tarkoma. *Efficient Content-based Routing, Mobility-aware Topologies, and Temporal Subspace Matching.* PhD thesis, University of Helsinki, Faculty of Science, Department of Computer Science and Helsinki Institute for Information Technology, 2006.

[Tar08] Sasu Tarkoma. Dynamic content-based channels: meeting in the middle. In *Proceedings of the 2nd international conference on distributed event-based systems (DEBS)*, 2008.

[TBF$^+$03] Wesley W. Terpstra, Stefan Behnel, Ludger Fiege, Andreas Zeidler, and Alejandro P. Buchmann. A peer-to-peer approach to content-based publish/subscribe. In *Proceedings of the 2nd international workshop on distributed event-based systems (DEBS)*, New York, NY, USA, 2003. ACM.

[TC03] Liying Tang and Mark Crovella. Virtual landmarks for the Internet. In *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement (IMC)*, 2003.

[TE04] Peter Triantafillou and Andreas Economides. Subscription summarization: A new paradigm for efficient publish/subscribe systems. In *Proceedings of the 24th international conference on distributed computing systems*, 2004.

[TIB01] TIBCO. TIBCO software chosen as infrastructure for NASDAQ's supermontage.

http://www.tibco.com/company/news/releases/2001/press388.jsp, 2001.

[TKAR10] Muhammad Adnan Tariq, Boris Koldehofe, Ala' Altaweel, and Kurt Rothermel. Providing basic security mechanisms in broker-less publish/subscribe systems. In *Proceedings of the 4th ACM international conference on distributed event-based systems (DEBS)*, 2010.

[TKK$^+$10] Muhammad Adnan Tariq, Gerald G. Koch, Boris Koldehofe, Imran Khan, and Kurt Rothermel. Dynamic publish/subscribe to meet subscriber-defined delay and bandwidth constraints. In *Proceedings of the 16th international conference on parallel computing (Euro-Par)*, 2010.

[TKK$^+$11] Muhammad Adnan Tariq, Boris Koldehofe, Gerald G. Koch, Imran Khan, and Kurt Rothermel. Meeting subscriber-defined QoS constraints in publish/subscribe systems. *Concurrency and Computation: Practice and Experience*, 23:2140–2153, 2011.

[TKKR09] Muhammad Adnan Tariq, Boris Koldehofe, Gerald.G Koch, and Kurt Rothermel. Providing probabilistic latency bounds for dynamic publish/subscribe systems. In *Proceedings of the 16th ITG/GI conference on kommunikation in verteilten systemen (KiVS)*. Springer, 2009.

[TKKR12] Muhammad Adnan Tariq, Boris Koldehofe, Gerald G. Koch, and Kurt Rothermel. Distributed spectral cluster management: A method for building dynamic publish/subscribe systems. In *Proceedings of the 6th ACM international conference on distributed event-based systems (DEBS)*, 2012.

[TKR13] Muhammad Adnan Tariq, Boris Koldehofe, and Kurt Rothermel. Efficient content-based routing with network topology inference. In *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS)*, 2013.

[Too12] MLab: Measurement Lab Tools. http://www.measurementlab.net/measurement-lab-tools, 2012.

[TP11] S. Turner and T. Polk. Prohibiting secure sockets layer (SSL) version 2.0. *RFC 6176*, March 2011.

[VK01] Jarkko Venna and Samuel Kaski. Neighborhood preservation in nonlinear projection methods: An experimental study. In *Proceedings of the international conference on artificial neural networks*, 2001.

[VK06] Jarkko Venna and Samuel Kaski. Local multidimensional scaling. *Neural Networks - Special issue: Advances in self-organizing maps*, 19:889–899, 2006.

[VRKS06] Spyros Voulgaris, Etienne Rivière, Anne-Marie Kermarrec, and Maarten van Steen. Sub-2-Sub: Self-organizing content-based publish and subscribe for dynamic and large scale collaborative networks. In *Proceedings of the 5th international workshop on peer-to-peer systems (IPTPS)*, 2006.

[Wax88] Bernard M. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6:1617–1622, 1988.

[WBH$^+$08] Oliver P. Waldhorst, Christian Blankenhorn, Dirk Haage, Ralph Holz, Gerald G. Koch, Boris Koldehofe, Fleming Lampi, Christoph P. Mayer, and Sebastian Mies. Spontaneous virtual networks: On the road towards the Internet's next generation. *it - Information Technology*, 50:367–375, 2008.

[WC04] Bang Ye Wu and Kun-Mao Chao. *Spanning Trees and Optimization Problems.* Chapman and Hall, 2004.

[WCEW02] C. Wang, A. Carzaniga, D. Evans, and A. Wolf. Security issues and requirements for Internet-scale publish-subscribe systems. In *Proceedings of the 35th annual Hawaii international conference on system sciences (HICSS).* IEEE Computer Society, 2002.

[WCLW06] Jinling Wang, Jiannong Cao, Jing Li, and Jie Wu. Achieving bounded delay on message delivery in publish/subscribe systems. In *Proceedings of the international conference on parallel processing*, pages 407–416, Washington, DC, USA, 2006. IEEE Computer Society.

[WCT00a] Bang Ye Wu, Kun-Mao Chao, and Chuan Yi Tang. Approximation algorithms for some optimum communication spanning tree problems. *Discrete Applied Mathematics*, 102:245–266, 2000.

[WCT00b] Bang Ye Wu, Kun-Mao Chao, and Chuan Yi Tang. Approximation algorithms for the shortest total path length spanning tree problem. *Discrete Applied Mathematics*, 105:273–289, 2000.

[WCVY03] Daniel G. Waddington, Fangzhe Chang, Ramesh Viswanathan, and Bin Yao. Topology discovery for public IPv6 networks. *ACM SIGCOMM Computer Communication Review*, 33:59–68, 2003.

[WKM00] Tina Wong, Randy z Kat, and Steven McCanne. An evaluation of preference clustering in large-scale multicast applications. In *Proceedings of the 19th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*, 2000.

[WMS08] Wei Wang, Mehul Motani, and Vikram Srinivasan. Dependent link padding algorithms for low latency anonymity systems. In *Proceedings of the 15th ACM conference on computer and communications security*, 2008.

[Won80] Richard T. Wong. Worst-case analysis of network design problem heuristics. *SIAM Journal of Algebraic and Discrete Methods*, 1:51–63, 1980.

[WQA$^+$02] Y.M. Wang, L. Qiu, D. Achlioptas, G. Das, P. Larson, and H.J. Wang. Subscription partitioning and routing in content-based publish/subscribe systems. In *Proceedings of the 16th international symposium on distributed computing (DISC)*, 2002.

[Wu02] Bang Ye Wu. A polynomial time approximation scheme for the two-source minimum routing cost spanning trees. *Journal of Algorithms*, 44:359–378, 2002.

[Wu04] Bang Ye Wu. Approximation algorithms for the optimal p-source communication spanning tree. *Discrete Applied Mathematics*, 143:31–42, 2004.

[WXWD03] Qiang Wang, Jun-gang Xu, Hong-an Wang, and Guo-zhong Dai. Adaptive real-time publish-subscribe messaging for distributed monitoring systems. In *Proceedings of the 2nd IEEE international workshop on intelligent data acquisition and advanced computing systems: technology and applications*, 2003.

[WZC$^+$12] Zhaoran Wang, Yu Zhang, Xiaotao Chang, Xiang Mi, Yu Wang, Kun Wang, and Huazhong Yang. Pub/Sub on stream: a multi-core based message broker with QoS support. In *Proceedings of the 6th ACM international conference on distributed event-based systems (DEBS)*, 2012.

[YAY11] Albert Yu, Pankaj K. Agarwal, and Jun Yang. Subscriber assignment for wide-area content-based publish/subscribe. In *Proceedings of the IEEE 27th international conference on data engineering (ICDE)*, 2011.

[YB07] Eiko Yoneki and Jean Bacon. eCube: hypercube event for efficient filtering in content-based routing. In *Proceedings of the OTM confederated international conference on "on the move to meaningful Internet systems": CoopIS, DOA, ODBASE, GADA, and IS - volume part II*, 2007.

[YH07] Xiaoyu Yang and Yiming Hu. A DHT-based infrastructure for content-based publish/subscribe services. In *Proceedings of the 7th IEEE international conference on peer-to-peer computing (P2P)*, pages 185–192, Washington, DC, USA, 2007. IEEE Computer Society.

[YYSZ09] Yong Yu, Bo Yang, Ying Sun, and Sheng-lin Zhu. Identity based signcryption scheme without random oracles. *Computer Standards & Interfaces*, 31:56–62, 2009.

[ZCB96] Ellen W. Zegura, Kenneth L. Calvert, and Samrat Bhattacharjee. How to model an internetwork. In *Proceedings of the 15th IEEE international conference on computer communications, joint conference of the IEEE computer and communications societies (INFOCOM)*, 1996.

[ZK04] Ying Zhao and George Karypis. Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine Learning*, 55:311–331, 2004.

[ZL08] Ying Zhu and Baochun Li. Overlay networks with linear capacity constraints. *IEEE Transactions on Parallel and Distributed Systems*, 19:159–173, 2008.

[ZLP09] Ying Zhu, Baochun Li, and Ken Qian Pu. Dynamic multicast in overlay networks with linear capacity constraints. *IEEE Transactions on Parallel and Distributed Systems*, 20(7):925–939, 2009.

[ZP11] Xian Zhang and Chris Phillips. A survey on selective routing topology inference through active probing. *IEEE Communications Surveys & Tutorials*, 2011.