it 5/2009

Security in Multi-domain Event-based Systems

Sicherheit in ereignis-basierten Mehrdomänensystemen

Jean Bacon, David Eyers, Jatinder Singh, University of Cambridge,
Brian Shand, National Health Service, Cambridge,
Matteo Migliavacca, Peter Pietzuch, Imperial College London

Summary Event-based systems give the potential for active information sharing. The event-based paradigm, if used for event transport, provides loose coupling between components, many-to-many communication and mutual anonymity of event producers and event consumers. This communication style has been adopted enthusiastically for convenience of programming; particularly for financial processing, healthcare applications and sensor-based systems. But some data is sensitive, and its visibility must be controlled carefully for personal and legal reasons. Our research projects have explored this space for some time, investigating application domains in which the event-based paradigm is appropriate yet where security is an issue. We discuss security issues for multi-domain, event-based systems, considering the requirements of applications and the risk associated with failure. We provide an overview of the state-of-the-art in secure event-based systems: research already carried out, work in progress and issues still to be addressed. This is of relevance to emerging large-scale systems required by government and public bodies for domains such as healthcare, police, transport and environmental monitoring. ►►► **Zusammenfassung** Ereignis-basierte Systeme ermöglichen den aktiven Austausch von Informationen. Ein ereignis-basiertes Paradigma erlaubt eine lose Kopplung von Komponenten, n-zu-n-Kommunikation und gegenseitige Anonymität von Erzeugern und Konsumierern von Ereignissen. Dieser Kommunikationsstil führt auch zu vereinfachter Programmierbarkeit, insbesondere auf dem Gebiet der finanziellen Datenverarbeitung, im Gesundheitsbereich und für Sensor-basierte Anwendungen. Da jedoch Daten oftmals aus rechtlichen und persönlichen Gründen vertraulich sind, muss deren Sichtbarkeit sorgfältig begrenzt werden. Unsere Forschungsprojekte haben seit einiger Zeit Lösungen auf diesem Gebiet untersucht. Wir beschreiben Sicherheitsfragen auf dem Gebiet von Ereignis-basierten Mehrdomänensystemen, wobei wir die Anforderungen von Anwendungen und die Risiken von Fehlern berücksichtigen. Wir geben einen Überblick über vergangene, gegenwärtige und zukünftige Forschung zum Thema Sicherheit in Ereignis-basierten Systemen. Diese Arbeit ist besonderes relevant angesichts der kommenden großflächigen Systeme, die von staatlichen und öffentlichen Institutionen im Gesundheitswesen, Polizeiwesen, Transportwesen und in der Umweltüberwachung angestrebt werden.

Keywords C.2.4 [Computer Systems Organization: Computer-Communication Networks: Distributed Systems] distributed applications; D.4.6 [Software: Operating Systems: Security and Protection] access controls; C.4 [Computer Systems Organization: Performance of Systems] reliability; security, publish-subscribe networks ►►► **Schlagwörter** Ereignis-basierte Systeme, Sicherheit, Zugriffskontrolle, Mehrdomänensysteme

1 Introduction

In recent years, *event-based* (or event-driven) *architectures* established themselves as one of the most prevalent paradigms for building scalable and flexible applications. They have been successfully applied to many different application areas, including healthcare, finance, transport, supply chain management and public services [3; 14; 15]. In event-based architectures, applications are structured

around *events*, which represent the information flow between a heterogeneous, potentially distributed set of components. Advantages of this approach are that it naturally supports the timely processing of information, decouples application components for increased scalability and easier maintenance and can be implemented efficiently using message-oriented middleware.

At the same time, event-based systems require fundamentally different approaches to security that reflect their heterogeneous and loosely coupled nature [2]. While event-based applications enable timely information sharing within and between autonomous administrative parts of a distributed system, this has to occur in a secure and constrained fashion. We use the term *domain* to describe each of these autonomous administrative entities. The visibility of sensitive data must be carefully controlled for personal and legal reasons. In this paper, we discuss the security requirements of event-based architectures and describe concrete solutions for securing them.

To understand the security requirements of a multi-domain system, we start by considering those for a single domain in Sect. 2, extending our discussion to inter-domain interactions in Sect. 3. From this basis, we describe the issues in composing a *multi-domain* event-based system, throughout which information can be shared actively. Examples of multi-domain systems include a national police force comprising multiple regional forces; branches of an organisation distributed worldwide; a national healthcare service comprising hospitals, clinics, primary care practices etc. Each domain has autonomy over administering resources, principals and roles, and expressing certain policies that have local scope, yet interaction with other domains is essential, often mandated by law or national government.

In terms of solutions for securing event-based systems, we discuss the approaches for secure event type management (Sect. 3.1), controlling client access (Sect. 3.2) and supporting partially untrusted infrastructures (Sect. 3.3). We also describe how information flow in an event-based system can be controlled (Sect. 3.4) and how events can be integrated with databases without compromising security (Sect. 3.5). We finish with application scenarios that illustrate many of the proposed techniques in a real-world setting (Sect. 4).

2 Event-based Architectures

Our research focuses on event-based architectures that span multiple administrative domains, as found in many public service (e. g., healthcare and police) and business applications (e. g., company divisions). We introduce our model of a multi-domain event-based system that we assume throughout the paper and give a brief overview of other related work.

Our previous work, which is summarised below, has described the components of event-based systems: event producers, consumers and brokers; and the different styles of events that have been used in various systems, for example, message queues, topic hierarchies, type and attribute based events [1; 11; 12]. Figure 1 illustrates a multi-domain publish/subscribe system: three domain clouds are shown interacting; the specific application example will be described in Sect. 4. The figure also shows the generic components of domains. Each domain comprises a collection of *event brokers* that manage internal communication between *event publishers* that produce events and *event subscribers* that consume events. Publishers may *advertise* the event types that they are authorised to *publish*. Subscribers *subscribe* to events using subscrip-

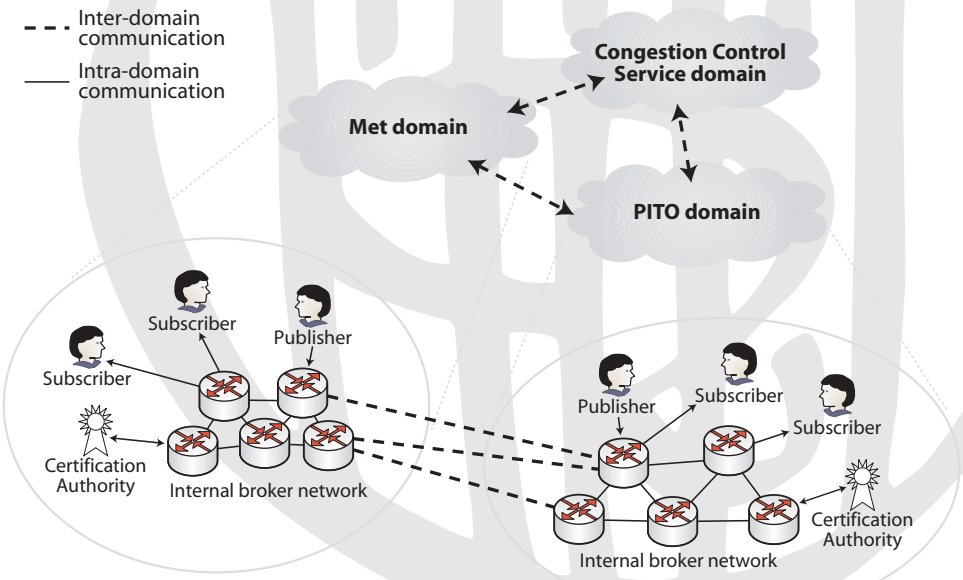

Figure 1 Illustration of a multi-domain event-based system architecture.

tions with filters that describe their event interests. In addition, each domain contains a *certification authority* that is authorised to manage names and access control policy within that domain.

To control access by clients to event-based communication, we take advantage of *role-based access control* (RBAC) [13]. RBAC is a scalable mechanism for managing access control policy. Roles are introduced as an indirection between the principals in a system and the privileges protected by the access control system. To secure event-based communication using RBAC, access control policy specifies the roles that are authorised to advertise, publish and subscribe to the various events defined in the system. The use of RBAC to manage message delivery security is well established, for example in the J2EE support for Java Messaging Service (JMS) security.

For each domain of an event-based system, we assume a management structure responsible for the following:

Administration of principals and roles. Within a domain, *principals* are uniquely identified through secure authentication mechanisms. *Roles* are defined, as is the association of principals and roles. Correct *role activation* by principals, i. e., the acquisition of privileges associated with a given role, is securely enforced.

Event type management. A domain must present a system-wide unique *name* for an event type and its use must be according to policy. A *type owner* is associated with each event type. A type owner registers an event type and is the source of privilege for use of the type, and its evolution.

Per domain authorisation policy specification and enforcement. *Authorisation policy* specifies the privileges of roles, including any context requirements, and this is securely enforced. Only those authorised can publish and subscribe to events, each according to current context.

Interaction management with other domains. We assume that *domain managers* negotiate and specify inter-domain authorised interactions in terms of roles, principals and permissions including context. To effect this management securely, policy must be specified and enforced automatically. A domain may contain a secure server capable of issuing, storing and checking credentials and at least one dedicated and trusted event broker (the "Certification Authority" nodes in Fig. 1).

Related Work

A general overview of the issues in securing a publish/subscribe service is given in [22]. Some approaches place the burden of control on information producers. In *symmetric publish/subscribe* [19], publishers include a filter with their publications. Events are delivered according to the intersection of publication and subscription constraints. In the work of Oprychal et al. [7], *event owners* may conditionally licence event privileges to other principals in the system.

EventGuard [17] and PSGuard [18] propose encryption and key management schemes for publish/subscribe systems which involve clients encrypting messages for transmission through an untrusted broker network. In EventGuard publishers sign events and encrypt them with a publication-specific, random encryption key. The encryption key is then encrypted with a topic-specific key and attached to the event. Event brokers are expected to verify publishers' signatures on each routing hop. In PSGuard a sophisticated key management scheme using key graphs is proposed to define encryption keys according to content-based subscriptions thus avoiding creating a key group for combinations of recipients.

In environments where data is perpetually sensitive (e. g., healthcare), it may be inappropriate to liberally distribute encrypted information. Compromised keys render (historical) events visible. Often, it is important to control the transmission of the information itself, through mechanisms such as RBAC and flow-control policy.

Other approaches allow administrators to define security aspects. A *scope* [4] is a grouping structure that encapsulates a number of components. Scopes are used to limit the visibility of an event to its members, with operations for dealing with external entities. Wun and Jacobsen [23] describe a general policy model, allowing actions to be performed at various stages of the publish/subscribe process. This model can be used to perform security operations.

3 Securing Multi-domain Event-based Systems

As for a single domain, the security of a multi-domain event-based system relies on the confidentiality and integrity of the information being exchanged. In a multi-domain system, security can be provided system-wide if the following requirements are met:

a) Only *authorised subscribers* may receive sensitive events, and publishers must be restricted to *publish only authorised events*. We propose system-wide event type naming and control (Sect. 3.1) integrated with client access control (Sect. 3.2) to address this requirement.

b) *Events* must be *secure in transit*, and must be delivered as required by recipients. Secure event flow control addresses this (Sect. 3.4), coupled with standard point-to-point encryption such as SSL, and our initial assumptions of a secure credential server per domain and at least one trusted event broker per domain.

Since the event-broker network comprises brokers from multiple domains, trust may not be uniform across the broker network. Some domains may be deemed less trustworthy than others, for example in a globally distributed company. A monitoring system may also be in place. Trusted brokers must be able to recognise untrusted brokers (Sect. 3.3). For non-

sensitive or low-priority events, security in transit and reliable delivery may be explicitly relaxed.

c) For each domain, the *domain authorisation policy* must be *correct, and secure* against fraudulent alteration. Policy correctness is beyond the scope of this paper – typically, a formal security audit of the application design and information flows would be required. We support the enforcement of such correct policy, by enabling separation of policy and operational data flow, and use the secure credential servers to provide correct and up-to-date authorisation policy.

d) *End-to-end security analysis* must be supported. We address this in the event space using a consistent event naming and access control model throughout each domain, enabling a straightforward mapping between application security and event security. Section 3.5 illustrates this for databases, frequently an integral part of event-based systems. For external issues, such as physical security and the security implications of paper and physical media, a formal security audit is needed.

With this architectural background and the above assumptions, we can list the challenges in securing multi-domain event-based systems that our techniques below aim to address. These challenges include:

a) non-uniform trust between domains,
b) non-uniform trust of the event-broker infrastructure,
c) enforcing that imported components will perform as specified,
d) special concerns relating to sensitive data that may persist for a human lifetime or longer,
e) how to express and enforce end-to-end communication of data on a need-to-know basis.

Security requirements that are broader in scope than securing an event-based system are not addressed above. These include minimising the impact of a security policy failure, specifying restart procedures including partial recovery, intruder detection and the trade-off between minimising the overhead of enforcing security while demonstrating that legal obligations are met.

3.1 Event Type Management

An important aspect when securing event data is how the event-based system refers to event types uniquely – note that all events are assumed to be typed. We show in [9] how a principal in a domain invokes the event service to create, and thus become the owner of a new event type. The owner also specifies the access rights to the attributes of the event type in terms of roles defined in the creating domain, according to context.

In many cases, event types are defined in a parent domain for use across an entire wide-area system. In other cases, domain managers negotiate the right to use an event type outside the creating domain. We show in [9] how standard certificate chain technology is used to pass the credentials required for using an event type from the creating domain to other domains. Access rights to use the event type in receiving domains are specified in terms of the roles in those domains.

Over time, an event type is likely to evolve through different versions, and there are access control implications of this evolution. Details of how we ensure event-type uniqueness and controlled evolution are given in [8].

3.2 Client Access Control

Principals and roles are administered within a single domain. For inter-domain operation, we assume prior policy negotiation between domain managers about the permitted access of principals and roles to the event system. For both intra- and inter-domain operation, policy must be specified and enforced that describes how clients are permitted to access the event-based infrastructure to advertise, publish and subscribe to events. We describe this client-level access control in detail in [1] including applying different access control restrictions for different attributes; for example, some event fields may need to be removed for some roles, some values may need to be replaced by ball-park ranges for privacy preservation. In some cases, a generic policy may be expressible when events are communicated from one domain to another.

We describe in [1] how such policies may be enforced by the event-based system using RBAC. We assume certificates are signed, issued and checked by a secure server per domain. When a certificate is presented outside its home/issuing domain a check-back is made with the home domain's security service. The certificate can be cached to avoid subsequent inter-domain checking, but a notification must be set up in case the certificate is revoked; another use of an event-based infrastructure.

3.3 Event Transmission Through Untrusted Channels

In multi-domain event-based systems, we cannot assume that all components are equally trusted. If multiple administration domains pool their resources, some event brokers may be fully trusted. For example, brokers owned and managed by police domains may not trust other domains' brokers to handle event data that is not public.

In large, dedicated broker networks, optimised routing schemes are used. Content-based routing is a popular approach for sharing transmission paths close to publishers and fanning out only when close to subscribers. Distributed hash table approaches can manage the joining and leaving of brokers, while maintaining correct routing tables. However, this can bring an untrusted broker into the path of confidential data.

In [9; 10], we discuss how whole-event and attribute-grained data encryption mechanisms can ensure that brokers may decrypt only the data they are trusted to see. An example is described in Sect. 4. Content-based routing is adjusted for use by untrusted brokers, not able to view content. Event brokers must join key groups, and keys must be refreshed when brokers join and leave the group [8].

Through the above examples, we see that some security concerns are addressed by the approaches already outlined. A remaining issue is that some data remains sensitive for periods of a human lifetime or longer. Concerns include that sensitive data may be stored long term in encrypted form or, by authorised parties, in clear. Transmitting encrypted data, and allowing only authorised parties to decrypt it, is insecure long-term. The security of encryption keys of a given length is of limited duration; as computers increase in power, keys will be broken.

We have explored inter-domain transmission of events, end-to-end, on a need-to-know basis [14; 15]. We apply event transformations, for example to remove sensitive fields (relating to specific people) or to obfuscate specific values (replacing them by ranges). Transformations allow a domain to control the information released in the given context. Clearly, the richer the representation of state, the more granular the control. For example, data can be tailored to a specific domain, or set of circumstances, rather than to a particular role. Accountability is improved, as those responsible for data are in control of what they transmit. Transformations facilitate decentralised data management; an approach that mitigates the risks and impacts of a confidentiality breach [5].

3.4 Distributed Event Flow Control

In extremely sensitive scenarios, such as healthcare, restricting delivery of event notifications only to certain clients is not sufficient, as these clients may not be completely trusted to respect policy. This is particularly relevant when the client is an autonomous application, which may be subverted or contain errors. In this case, application components should be sand-boxed by the trusted infrastructure to prevent policy violation.

This approach is similar to *information flow control* systems (IFC) as applied first in programming languages [6], and more recently in operating systems [24]. IFC focuses on securing local applications by taint-tracking data as it flows through application components. This is done by attaching security labels to data: IFC restricts access to and manipulation of these labels. Thus, since components can communicate only through labelled data, the system can provide end-to-end guarantees on data confidentiality and integrity.

In the *SmartFlow* project [16], we explore the use of these techniques to secure event-based systems in a multi-domain environment. We assign security labels to values contained in event attributes and to event processing components that operate on them. For example, a component's confidentiality label records the highest event label delivered to the component. The system ensures that the confidentiality label of a component is a lower bound on the labels of events produced by that component, thus preserving end-to-end system security.

3.5 Database Integration

Active information sharing within and between domains is as likely to be via databases as through direct transmission. The legal requirement for archival storage means that events will be logged in databases. Access to the archive must be controlled as rigorously as to primary data sources. Persistent storage is also required for reliable event delivery. We therefore believe that databases are essential to an event-based architecture.

We have explored the replacement of continuous query support by advertisement and publication of events by databases [20], which also incorporates application-level transactions [21]. Such integration achieves better performance than systems where communication and database service are separate. Database components are subject to RBAC for advertisement and publication and their clients for subscription, as described in Sect. 3.2.

4 Application Scenarios

Next we describe how the techniques for securing event-based systems are applied to particular application scenarios.

Healthcare. Healthcare is highly collaborative, where multiple care providers require notification of health incidents (events). However, providers are responsible for protecting personal information. To support this, we introduce a broker-specific data control layer that interacts with a publish/subscribe service to restrict and transform events according to (environmental and messaging) context [15].

Healthcare is moving towards the provision of care outside of traditional care institutions (e. g., hospitals). Homecare environments are highly data-driven, where principals, from various administrative domains, require real-time data; e. g., to deal with emergency situations. We have considered homecare scenarios [14], where local policy provides principals with only that information *relevant*, in the current context, to their role in the care process.

This scenario shows the need for consistent access control policy, both to protect patient privacy, and to ensure patient safety in an emergency. By restricting access to information throughout the event-based system, we support effective containment of private information, in a way that supports end-to-end security analysis.

UK Police forces. Here we examine in detail a possible, multi-domain, event-based architecture for congestion charging in London.

This scenario, as illustrated in Fig. 2, involves interaction between three different domains: the Metropolitan Police (Met) domain, the Congestion Charge Service (CCS) domain, and the Police Information Technology Organisation (PITO) domain.

The Metropolitan Police Domain administers a set of CCTV cameras that publish events as vehicles are seen

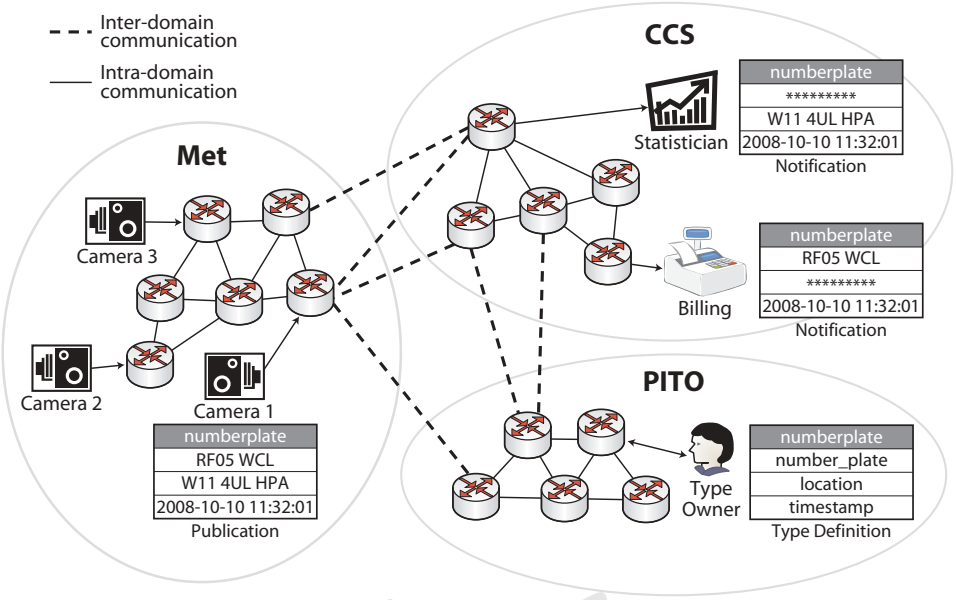

Figure 2 Architecture supporting congestion charging.

to move around the London area. The CCS operates the systems that determine which vehicles to levy charges on each day due to those vehicles travelling through the London congestion charging zone. The PITO domain represents the administrative unit from which national police data standards are managed, and in our context would be the owner of the event types used for inter-domain operation.

Consider the management of NUMBERPLATE events that occur when vehicles are sighted in the congestion zone by the Met domain cameras. The types and attributes of NUMBERPLATE events are controlled by the PITO domain; the Met domain must have been authorised to publish these events by PITO.

A number of different sorts of subscriptions can be justified that have different levels of access to attributes of NUMBERPLATE events. Within the CCS, the billing staff need only be authorised to view information about the numberplate, and the time of its sighting. A statistician measuring the impact of traffic condition changes, however, may be authorised to see the location and the time of sightings, but not the numberplates of the cars themselves. Within the Met domain, it is likely that certain investigations would be assisted by having certain officers subscribe to all of the information about sightings of a particular numberplate.

By allowing fine-grained access control over event attributes to be handled by the middleware, we can more effectively aggregate the dissemination of events. For example, through the use of per-attribute encryption, we can avoid the need to form separate subscriptions to the publisher that are partitioned along the lines of different access control rights. This has the net effect of reducing the number of messages that need to be transmitted.

5 Conclusions

Secure event-based systems need consistent, fine-grained protection over information exchange. Multi-domain systems add complexities, such as trust management within a shared infrastructure. In this paper, we recommend a solution that makes few assumptions about domain management and secure infrastructure. We propose system-wide event naming and access control, explicitly model event security in transit, and separate security policy management from operational data flow, to provide support for end-to-end security analysis in complex event-based systems.

System security depends on the correctness, completeness and enforcement of our assumptions on event-based architectures. Monitoring procedures must encompass not only application-level logging but also the functioning of secure services that manage and control certificates.

Point-to-point communication, rather than content-based routing of sensitive data across administrative domains, avoids the complexity of partially untrusted infrastructures. For the relatively small number of brokers and domains in many public service domains, this is perhaps the best approach.

Our most recent approach using information flow control gives a domain fine-grained control over the transmission and content of sensitive events. Instead of liberally distributing encrypted data, this restricts information flow close to the source to reduce the long-term risk of decryption.

However, the security of event-based systems depends on factors beyond the event processing infrastructure. Formal security audit of an event-based application must include all aspects of event security, physical security and human factors. In particular, the long-term security of sensitive data held by authorised parties must depend on

human as well as automated procedures. Many recent gross privacy violations have resulted from human negligence rather than system incorrectness. Such parties can be reminded automatically of their legal obligations to ensure privacy without imposing an undue burden. Providing comprehensive solutions addressing these issues is an open challenge for future work in this space.

Acknowledgements

This work was supported by the UK Engineering and Physical Sciences Research Council under grants EP/C53718X, EP/F042469 and EP/F044216.

References

[1] J. Bacon, D. M. Eyers, K. Moody, and L. I. W. Pesonen. *Securing publish/subscribe for multi-domain systems*. In: G. Alonso, editor, Middleware'05, LNCS 3790, pp. 1–20, Grenoble, France, Nov 2005.

[2] J. Bacon, D. M. Eyers, J. Singh, and P. R. Pietzuch. *Access control in publish/subscribe systems*. In: Proc. of the 2nd Int'l Conf. on Distributed event-based systems (DEBS'08), pp. 23–34, New York, NY, USA, 2008.

[3] D. M. Eyers, J. Bacon, and K. Moody. *OASIS role-based access control for electronic health records*. In: IEE Proc. on Software, 153(1):16–23, Feb 2006.

[4] L. Fiege, M. Mezini, G. Muhl, and A. P. Buchmann. *Engineering event-based systems with scopes*. In: European Conf. on Object-Oriented Programming (ECOOP), pp. 309–333, 2002.

[5] Markle Foundation. *Achieving electronic connectivity in healthcare*. 2004.

[6] A. C. Myers and B. Liskov. *A decentralized model for information flow control*. In: Proc. of the 16th ACM Symp. on Operating Systems Principles (SOSP'97), pp. 129–142, New York, NY, USA, 1997.

[7] L. Opyrchal, A. Prakash, and A. Agrawal. *Supporting privacy policies in a publish-subscribe substrate for pervasive environments*. In: Journal of Networks, 2(1):17–26, Feb 2007.

[8] L. I. W. Pesonen. *A capability-based access control architecture for multi-domain publish/subscribe systems*. Technical Report 720 and Ph. D. thesis, University of Cambridge, 2008.

[9] L. I. W. Pesonen, D. M. Eyers, and J. Bacon. *Access control in decentralised publish/subscribe systems*. In: Journal of Networks, 2(2):57–67, Apr 2007.

[10] L. I. W. Pesonen, D. M. Eyers, and J. Bacon. *Encryption-enforced access control in dynamic multi-domain publish/subscribe networks*. In: Proc. of the Int'l Conf. on Distributed Event-Based Systems (DEBS'07), pp. 104–115, June 2007.

[11] P. Pietzuch, D. Eyers, S. Kounev, and B. Shand. *Towards a common API for publish/subscribe*. In: Proc. of the Int'l Conf. on Distributed Event-Based Systems (DEBS'07), pp. 152–157, June 2007. Short paper.

[12] P. Pietzuch and J. Bacon. *Hermes: A distributed event-based middleware architecture*. In: Proc. of the 1st Int'l Workshop on Distributed Event-Based Systems (DEBS'02), ICDCS, pp. 611–618, July 2002.

[13] R. Sandhu, E. Coyne, H. L. Feinstein, and C. E. Youman. *Role-Based Access Control models*. In: IEEE Computer 29(2):38–47, 1996.

[14] J. Singh and J. Bacon. *Event-based data dissemination control in healthcare*. In: Revised Selected Papers of the First Int'l Conf. on Electronic Healthcare (eHealth 2008), pp. 167–174, City University, London, Sept 2008. Springer.

[15] J. Singh, L. Vargas, J. Bacon, and K. Moody. *Policy based information sharing in publish/subscribe middleware*. In: Policy 2008, IEEE 9th Int'l Workshop on Policies for Distributed Systems and Networks, pp. 137–144, Palisades, NY, USA, June 2008. IEEE Computer Society.

[16] SmartFlow Project. http://www.smartflow.org. EPSRC Grant EP/F042469/1; Oct 08–Sep 11.

[17] M. Srivatsa and L. Liu. *Securing publish-subscribe overlay services with EventGuard*. In: Proc. of the 12th ACM Conf. on Computer and Communications Security (CCS'05), pp. 289–298, New York, NY, USA, 2005.

[18] M. Srivatsa and L. Liu. *Secure event dissemination in publish-subscribe networks*. In: Proc. of the 27th Int'l Conf. on Distributed Computing Systems (ICDCS'07), p. 22, Washington, DC, USA, 2007.

[19] A. Tomasic, C. Garrod, and K. Popendorf. *Symmetric publish/subscribe via constraint publication*. Technical Report CMU-CS-06-129R, Carnegie Mellon University, 2006.

[20] L. Vargas, J. Bacon, and K. Moody. *Integrating databases with publish/subscribe*. In: Proc. of the 4th Int'l Workshop in Distributed Event-Based Systems (DEBS'05), pp. 392–397, June 2005.

[21] L. Vargas, J. Bacon, and K. Moody. *Transactions in distributed event-based middleware*. In: Proc. of the 3rd Int'l Conf. on Enterprise Computing, E-Commerce, and E-Services (EEE'06), pp. 53–56, June 2006.

[22] C. Wang, A. Carzaniga, D. Evans, and A. Wolf. *Security issues and requirements in internet-scale publish-subscribe systems*. In: Proc. of the 35th Annual Hawaii Int'l Conf. on System Sciences (HICSS'02), p. 303, 2002.

[23] A. Wun and H.-A. Jacobsen. *A policy management framework for content-based publish/subscribe*. In: Middleware'07, LNCS 4834, pp. 368–388, 2007.

[24] N. Zeldovich, S. Boyd-Wickizer, E. Kohler, and D. Mazières. *Making information flow explicit in HiStar*. In: Proc. of the 7th Symp. on Operating Systems Design and Implementation (OSDI'06), pp. 263–278, Berkeley, CA, USA, 2006.

Received: May 15, 2009

Prof. Dr. Jean Bacon, Professor of Distributed Systems, University of Cambridge Computer Laboratory. Leads the Opera research group, with focus on asynchronous middleware for large-scale, widely distributed or geographically concentrated ubiquitous systems. Fellow, IEEE and BCS; member, Governing Body IEEE-CS, 2001-2007; founding EIC, Distributed Systems Online 2000-2008; Editorial Board, IEEE Computer.

Address: Computer Laboratory, University of Cambridge, JJ Thomson Avenue, CB3 0FD Cambridge, UK, e-mail: Jean.Bacon@cl.cam.ac.uk

Dr. David Eyers, Senior Research Associate in the Opera research group working on the SmartFlow project (funded by the UK EPSRC). Research interests include wide-area event-based systems, distributed access control mechanisms, and the interaction between the two. His Ph. D. thesis investigated event-based systems providing support for dynamic access control.

Address: Computer Laboratory, University of Cambridge, JJ Thomson Avenue, CB3 0FD Cambridge, UK, e-mail: David.Eyers@cl.cam.ac.uk

Jatinder Singh, Research student currently completing his Ph. D. thesis entitled "Controlling Publish/Subscribe for Healthcare". His research interests surround data privacy, particularly concerning infrastructure for the wide-scale dissemination of personal information. He has industry experience in designing health and judicial systems and completed his B. Sc. (Hons) and some Law at UWA.

Address: Computer Laboratory, University of Cambridge, JJ Thomson Avenue, CB3 0FD Cambridge, UK, e-mail: Jatinder.Singh@cl.cam.ac.uk

Dr. Brian Shand, Senior Research Fellow at CBCU Research (NHS), focussing on software infrastructure for the next generation of healthcare

applications. He holds a Ph. D. in distributed systems from the University of Cambridge and an M. Sc. in digital image processing from the University of Cape Town, with publications in distributed systems, event-based middleware, disclosure control and trust-based computation.

Address: Clinical and Biomedical Computing Unit (CBCU), National Health Service, Unit C – Magog Court, Shelford Bottom, CB22 3AD Cambridge, UK, e-mail: Brian.Shand@cbcu.nhs.uk

Dr. Matteo Migliavacca, Research Associate at DSE group working on the SmartFlow project after obtaining a Ph. D. degree at Politecnico di Milano with a thesis on Middleware Services for Large Scale Dynamic Distributed Systems. His main research interests are in distributed systems in the areas of routing, adaptivity, context-awareness and programming abstractions.

Address: Department of Computing, Imperial College London, 180 Queen's Gate, SW7 2AZ London, UK, e-mail: migliava@doc.ic.ac.uk

Dr. Peter Pietzuch, Lecturer in the Distributed Software Engineering (DSE) section, Department of Computing at Imperial College London. His research focuses on the design and engineering of scalable, reliable and secure Internet systems, including event processing, peer-to-peer and global sensing applications. He obtained his Ph.D. from Cambridge University and was a post-doctoral fellow at Harvard University.

Address: Department of Computing, Imperial College London, 180 Queen's Gate, SW7 2AZ London, UK, e-mail: prp@doc.ic.ac.uk