Exploiting Node Mobility for Coordinating Data Usage in Crisis Scenarios

Giovanni Russello[1] Enrico Scalavino[2]

[1] Create-Net, via alla Cascata 56D, Povo, Trento, Italy - 38123
giovanni.russello@create-net.org
[2] Imperial College London, Computing Department,
180 Queens Gate, London SW7 2RH, UK
escala@imperial.ac.uk

Abstract. Controlling data usage in a crisis scenario is a particularly acute issue due to the sensitive nature of the data being distributed. Typical access control mechanisms rely on a centralised architecture where access policies are stored and evaluated. However, such architecture is not practical in a crisis scenario due to the absence or the impossibility of using a wide-area communication infrastructure. In this paper, we introduce xDUCON a framework for enforcing usage control policies that relies on the Shared Data Space (SDS) abstraction for the coordination of policy evaluation and enforcement. Policies are represented as tuples to facilitate their propagation across the available infrastructure.

1 Introduction

In facing an emergency situation, several organisations are called on the scene of the accident. Emergency services, local authorities, health bodies and the environment agency are typically involved directly in the rescue operations. Data can be collected on the fly by the responders of each organisation and other data can be supplied from the organisation control centres. Ancillary organisations can be involved in the emergency by providing services and information. For instance, a surveillance company with a CCTV network installed into different buildings could provide real-time data on the crisis.

In order to effectively complete the rescue mission decisions must be taken in short time. Reading the actual situation requires the organisations and companies to share the information that they are responsible for. However, the sensitive nature of data being distributed requires that its confidentiality is protected. Data can contain victims' private information whose disclosure is governed by legislation such as the data protection act in the UK. Moreover, misuse of the information could have other negative effects. For instance, if media networks were allowed to broadcast any information general panic may spread further complicating the rescue operations.

In this context, data needs to be distributed across several organisational boundaries making the management of information release a fine act of balancing two opposite needs: on the one hand, the "need-to-know" principle dictates that

data should be released only to the smallest set of interested parties that have a specific need to access the information; on the other hand, the "need-to-share" principle dictates that the distribution of data should reach the largest set of entities to increase the benefit of shared knowledge. Shared information is often sensitive to each partner therefore it is necessary that rules for controlling its dissemination and usage are in place and enforced by all the involved parties.

Retaining control of the data being disseminated is a problem that was initially studied by Graubart in [5]. Most of the available solutions rely on mechanisms that are based on centralised architectures where connectivity is always available. As discussed in [15], the available enforcing mechanisms are based on hard-coded policies located in gateways to control the data dissemination and usage. As a result, policies are statically bound to the location where the gateway resides are too cumbersome to be used scenarios where no assumption can be made on the locations where the policy evaluation and enforcement take place. For instance, during a crisis rescuers have to operate in hostile environments with limited or intermittent connectivity. In such a scenario, the enforcement of usage control policies cannot rely on the availability of centralised repositories where the policies to be enforced are stored and evaluated.

In this paper, we present a high-level description of a framework for coordinating the control of information based on the model of the Shared Data Space (SDS) named xDUCON. Because the policy enforcement model is based on the SDS abstraction it is possible to specify and enforce usage control policies for dynamic network scenarios (such as the case in a crisis). In particular, in this paper we provide some preliminary details on how xDUCON can leverage the movements of rescuers that operate in the incident area and that can physically carry information rather then relying on traditional network communications. Each device where the xDUCON framework is deployed has its local SDS that can interact with the SDSes of other devices within communication range. Because data and policies are represented as tuples they can transparently be propagated through the different nodes representing the rescues and be enforced as required. This enables a localised evaluation and enforcement of usage control policies that is dynamically adjusted to the dissemination of the data.

This paper is organised as follows. In Section 2, we present a crisis management scenario discussing some of the motivations that drive our research. Section 3 describes the details of xDUCON and how usage control policies are defined. In Section 4, xDUCON is compared to related work. We conclude in Section 5 providing some highlights of future research.

2 A Crisis Scenario

In this section, we introduce a typical scenario of a crisis where different organisations, including some companies not directly involved with the rescue operations (e.g. utilities and transport providers), are called to gather and share information on the crisis.

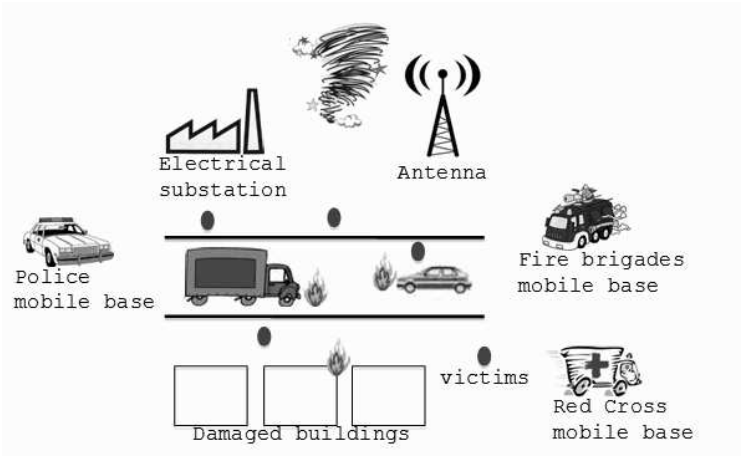

Fig. 1. A possible scenario of a crisis management.

The scenario is based on the absence of any type of communication facility and network. The events are described in Figure 1. Consider a twister raging over a town. Local antennas as well as buildings are hit and damaged. Cars and trucks driving in a main street begin to swerve causing chaos and accidents. People are left injured on the street or inside their vehicles. Some of the cars catch fire and flames rapidly reach stalls on the street side and buildings. Meanwhile, the twister also hits a nearby electrical substation actually stopping the power supply to most of the town's centre.

Police, Fire Brigades and Red Cross, are called to intervene on the scene. Vehicles of different types are sent to the area and some of them can act as mobile bases. These mobile bases can have powerful means of communications (such as satellite up-links) and can be connected to remote centres. During the operations, information gathered includes: 1) information on the victims; 2) the plant of the nearby electrical facilities and network; 3) information on the damaged buildings, their internal structure and the companies working inside, such as working days and times and personnel; 4) information on the progress of the rescue operations. Some of the data must be collected in the field while other can be provided by external organisations (e.g. the electricity provider and the companies with branches in the town) and sent to the rescue organisations' mobile bases.

In this context, there are two main requirements that need to be fulfilled: (i) on the one hand, rescuers need to quickly access the data for carrying out their tasks; (ii) on the other hand, data confidentiality needs to be protected against misuse and unnecessary accesses. Neglecting the first requirement can have a high cost even in terms of human life if the rescue operations were not effective, while neglecting the second requirement could have negative side-effects, for example in terms of legal actions ensued by victims or companies whose sensitive information was revealed.

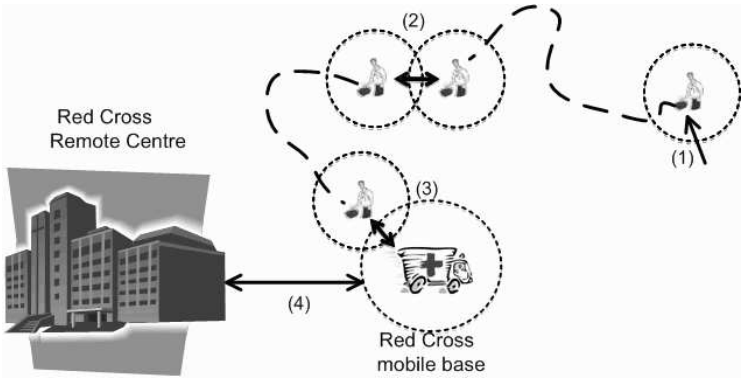

Fig. 2. The Red Cross mobile base aggregates data from individual responders from the disaster area and provides connections to the Red Cross remote centre.

Responders carry mobile devices (e.g. PDAs and smartphones) with limited communication range. Although the mobile bases have powerful means of communication it is not always possible to be connected to each rescuer's device due to physical barriers (e.g., rescuers can be operating underground or in a tunnel). However, it is possible to exploit close-range communications between devices to propagate the data and let the movements of the rescuers carry the data around the crisis area.

Figure 2 shows how the mobility of the devices can be used for the data dissemination process between nodes. Rescuers devices store data collected in different locations of the deployment area (1) and upon encountering other rescuers this data can be transferred to their devices (2). The rescuers can also get close to the local mobile bases and forward their data (3). At the same time, the data issued by external organisations (e.g. information related to the electrical network) are sent to the mobile bases from the organisation remote centre (4). This data is propagated to the rescuers' devices in the communication range of the mobile base and then disseminated to other rescuers operating in the disaster area but outside the communication range.

In disseminating data during a crisis, the set of entities that can have access to the data evolves with the evolving of the situation. Entities' rights to access a document should be dynamically evaluated and changed according to the actual situation. However, traditional access control schemes cannot be applied since no central server can be reached for the evaluation of the appropriate access policies. Moreover, each organisations might have a specific set of policies that can be available to its own units but not to the units of other organisations.

What is needed is a framework supporting the dissemination of the data along with the policies that govern the accesses. This would support a localised evaluation and enforcement of policies as well as the availability of such policies

where they are actually needed. The description of the approach that we propose for satisfying those requirements is the focus of the following section.

3 Approach

Our approach makes use of the Shared Data Space (SDS) abstraction for the dissemination of data and policies during a crisis scenario. The framework proposed here exploits the *referential-decoupling* and *time-decoupling* properties of the SDS model in order to make them available through *where* and *when* needed. Data and policies are represented as tuples that are not statically bound to the location where they have been generated. Tuples are generally agnostic of the entities' locations that are going to use them and applications can exchange tuples without the need of explicitly synchronising their executions. Moreover, the temporal-decoupling property allows the separation of the policy evaluation process from the activation process providing a more flexible enforcement model that suits well the needs of a crisis scenario.

3.1 xDUCON Framework

Our approach named xDUCON is inspired by the SDS abstraction where distributed applications coordinate their execution through a tuple space and the coordination language Linda [4]. In xDUCON, a SDS is used for coordinating the dissemination, evaluation and enforcement of usage control policies for the entities that operate in a disaster area. Figure 5 provides an overview of the deployment of the xDUCON framework in two hand-held devices: one device belongs to a paramedic and the other to a policeman. The SDS implementation used in the xDUCON framework is called *xDSpace*[3].

The unit of data that is contained in a SDS is called a *tuple*. A tuple is a ordered sequence of named fields with a value associated with it. Tuples are inserted using the `put` operation. Tuples are retrieved using an associative method requiring a matching template and using two operations: the `read(Tmpl)` operation retrieves a copy of a matching tuple; using a `take(Tmpl)` operation removes the matching tuple from the space. In their basic form, retrieval operations are blocking that is these operations will block the execution thread until a matching tuple is found in the space. In xDSpace retrieval operations can take an extra parameter `timeout`: this parameter if specified determines the amount of time (in milliseconds) that the operation is blocked waiting for a matching tuple. If a tuple is not found when the timeout expires then a `null` value is returned.

The programming model of xDSpace provides also some built-in functionalities. For instance, the operation `update(Tmpl,T)` atomically performs an update by substituting a tuple matching the template `Tmpl` with the tuple `T`. In certain cases, it is necessary to retrieve all the tuples matching a given template. As addressed by Rowstron and Wood in [11], it is not possible to perform such an

[3] Cross Domain Space

operation using the basic operations because the associative method used for retrieving tuples is non-deterministic. To circumvent this limitation, xDSpace supports the bulk versions of the `read` and `take` operations, respectively named `readb(Tmpl)` and `takeb(Tmpl)`. Another built-in functionality in xDSpace is the `notifyOn(Tmpl)` operation that returns every new instance of a tuple matching `Tmpl`. In this way, an application is notified every time a new tuple is inserted or updated (since an `update` is seen as a `take` followed by a `put`).

3.2 The xDSpace Abstraction

In xDSpace, tuples are used for representing *Subjects*, *Targets* and *xDPolicies*[4]. Subject tuples represent entities that perform actions on targets. Each member of the two organisations that is assigned to the crisis is represented by a subject tuple active in the xDSpace deployed in the corresponding device. Target tuples represent resources accessed by subjects. A subject acts on resources through the applications deployed on the device. For instance, an application deployed on a paramedic's device can be used for retrieving the medical data of the assisted victim. As the paramedic measures the vital signs of the victim, the new data can be inserted in the medical report. As such, a subject (through her application) can act both as provider and consumer of target tuples. The execution of actions that subjects (through their applications) perform on Target tuples are governed by xDPolicy tuples stored in the xDSpace. The policies impose restrictions on the use and distribution of the data collected and disseminated in the crisis scenario.

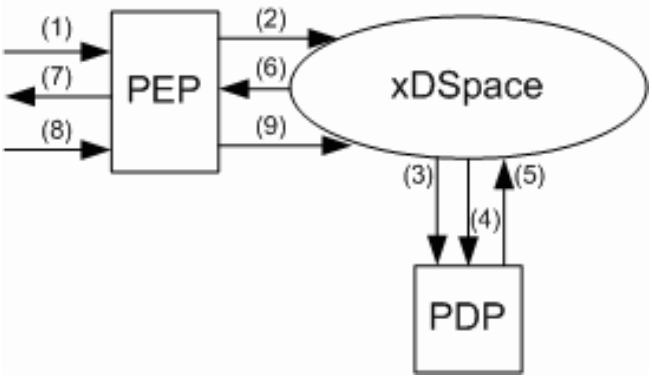

Fig. 3. Coordinating the interaction between the PEP and PDP through xDSpace.

In the xDUCON framework, the policy evaluation is carried out at the Policy Decision Point (PDP) while the enforcement of the decision that resulted from the policy evaluation is carried out at the Policy Enforcement Point (PEP).

[4] Cross Domain Policies

The PDP and PEP interact through the xDSpace as shown in Figure 3. When a request from an application comes to the PEP (1), a *request tuple (RqT)* is inserted in the space (2). A RqT is specified as `(subjectT, targetT, action, reqId)` tuple, where `subjectT` and `targetT` specify the subject and target tuples, `action` is the action that the subject is executing on the target, and `reqId` is a unique id to identify the request. The RqT is captured by the PDP (3) that retrieves from the space all xDPolicies matching the values of the first three fields in the RqT (4). The matching xDPolicies are evaluated and as a result the action could be granted or denied. Let us assume that for our case here the action is granted, then the PDP inserts a *decision tuple (DT)* in the space to communicate the PEP to grant the user's request (5). The DT is specified as `(reqId, decision)` where `(reqId)` is the id of the requesting tuple RqT; the field `decision` is used for the decision of the policy evaluation and it might take the values `GRANTED` or `DENIED`. The PEP retrieves the DT (6) and enforces the decision taken by the PDP by allowing the user to execute the action (7). When the user completes the action the rights to execute the action can be released (8) and the PEP inserts a *release tuple (RlT)* in the space (9). The RlT is specified as `(reqId)` that is the id of the request that was used for requesting the user's rights. Informing the PDP of the release of the rights is important in our framework for two reasons. First, xDUCON supports an ongoing evaluation model where the rights granted to a user can be revoked if certain conditions are no longer valid. When a RlT is inserted, the PDP can deallocate the resources used for monitoring the ongoing policy conditions. The second reason is that xDPolicies can also specify post-release actions to be executed when the user release the rights. The triggering of the execution of such actions is the insertion of a RlT.

3.3 Dissemination Model

In the scenario, it is assumed that the agencies and organisations involved in the crisis management have established already several *Data Sharing Agreements* (DSAs) with each others. Assuming the existence of a single universal DSA that covers all the aspects and requirements for the data dissemination is very unlikely since there will be many overlapping requirements. Moreover, some of the organisations involved are commercial organisations that have requirements on their data that are dictated by economical reasons. Therefore, we assume the existence of several DSAs between each agencies and organisations. The structure of the DSA contains a section for the legal aspects of data sharing with references to the relevant legislation. Another section of the DSA focuses on more concrete aspects such as what data should be shared, to whom the data is going to be released, and what data is expected from the other parties involved in the DSA. Moreover, requirements on how the data should be used and under which conditions are also specified in this section.

This latter section of the DSA is used for gathering the information necessary to xDUCON. In particular, it is necessary to be able to classify the data that is shared during a crisis, to identify the entities with which the data is going to be shared and to identify the protection requirements. At the time when the

mobile bases of each agency is deployed in the crisis area, each mobile base has access to the necessary information extracted from the DSA agreed with the other agencies. In particular, each mobile base deploys on the devices carried by the personnel belonging to the same agency the following information:

- a *Dissemination Table* (DT) containing the data types to be disseminated and the lists of entities to which a specific data type should be provided;
- the set of xDPolicies that control access to the data types;
- the credential signed with the private key of the agency to be used for authenticating the personnel;
- the public key of the other agencies involved in the crisis management and that is used to authenticate personnel belonging to other agencies.

For instance, let us consider the case shown in Figure 4. When the RC mobile base reaches the disaster area, the medical personnel is debriefed before acting in the area. During this time, the devices carried by each medical personnel are connected with the mobile base that deploys the xDPolicies into the xDSpace and the DT, credential and public keys of the other organisations involved (such as Police) into the *Dissemination Manager*.

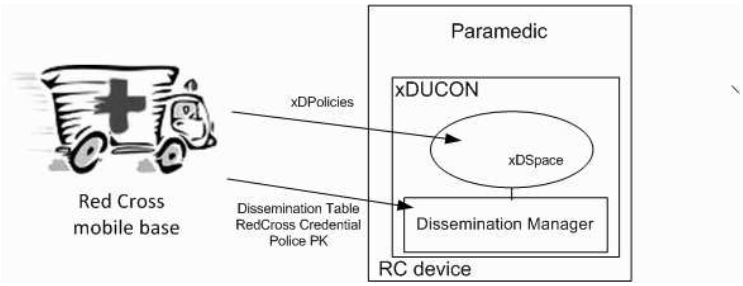

Fig. 4. Deployment of the DSA information from a Red Cross Mobile Base to a handheld device of a paramedic.

When the paramedic's device comes into a communication range with other devices carried by rescuers from other organisations, the Dissemination Manager on each device will try to authenticate the other and exchange data. As an example, let us consider the case of the paramedic's device coming into communication range within a Police Officer's device as shown in Figure 5. The Dissemination Manager on each device start exchanging credentials and use the respective public key to authenticate each other (1). After the authentication phase, the Dissemination Managers check their DT to decide which data type must be sent to the other device. In the case shown on Figure 5, the Dissemination Manager of the Red Cross (RC)'s device collects the target tuple representing the victim reports and together with the appropriate xDPolicies forwards them to the Police device.

xDSpace supports a two-side enforcement model: xDPolicies can be enforced both at the release-time and access-time of a target tuple. For the case of our scenario, the release of the victim report by the paramedic in the area is evaluated against a set of xDPolicies that prevent the provider to include in the report information that is not of interest for the Police Force. Similarly, when a policeman accesses the report another set of policies might be enforced to limit the number of accesses that can be executed by the Police Force. An example of a xDPolicy will be presented in the following section.

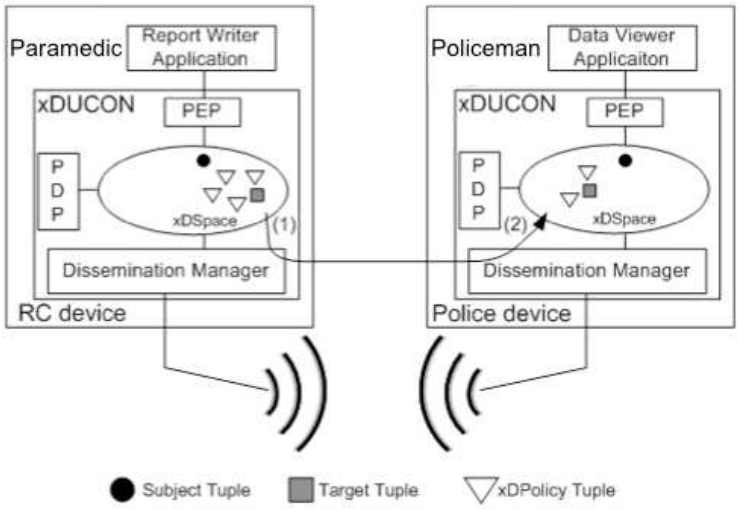

Fig. 5. Exchange of information between a Red Cross device and a Police device through the xDUCON framework.

3.4 xDPolicies

An xDPolicy is represented by a tuple specified as `xDPolicyTuple(SbjT,TargT,Action,Body)`. The `Body` of a `xDPolicyTuple` is evaluated by the PDP when a request tuple for a subject `SbjT` to execute an `Action` on target `TargT` is retrieved by the PDP. The `Body` comprises four different parts shown as follows:

```
REQUEST
  statement
GRANTED
  statement
RELEASED
  statement
REVOKED
  statement
```

PDP executes each part of a xDPolicy body depending on which phase the policy evaluation and enforcement is. The `REQUEST` part is evaluated when the request comes in and usually returns a `DT` tuple with a decision that can grant or deny the execution of the action. The `GRANTED` part is evaluated during the access. This part is used for checking that the ongoing conditions are satisfied while the access rights are used by the subjects. Moreover, some ongoing actions can be executed to update fields of the subject and/or target tuples. When the subject releases the access rights, then the `RELEASED` part is executed. Usually some post-actions are executed here for updating access counters of target and subject tuples. Finally, if some of the ongoing conditions evaluated in the `GRANTED` part are not satisfied, the access rights can be revoked. In this case the `REVOKED` part is executed.

xDPolicy 1 *As an example of an xDPolicy let us consider the case of a police officer accessing a patient record received from a paramedic. The police officer is represented by a* police officer tuple (POT) *while the patient's record is represented by a* patient record tuple (PRT)*. The policy verifies that a pre-condition is satisfied in order to execute the action. The condition checks that the patient record has not been accessed more than 5 times. If this is the case, before granting the rights for executing the action, the policy executes some pre-actions. First of all, the tuple field representing the access counter is increased. For auditing reasons, a* log tuple (LogT) *is inserted in the space containing the police officer id, the record id, the type of access, and the date of the access. Finally, the request is granted by inserting a reply tuple. The policy is specified as follows:*

```
POT(Id,rank)
PRT(recordId,medicalData,numOfAccesses)
xDPolicy POT, PRT, read
  REQUEST
    IF PRT.numOfAccesses<=5
      DO PRT.numOfAccesses++
          space.put(LogT(POT.Id,PRT.recordId,read,date())
          space.put(DT(RqT.reqId,GRANT)
```

4 Related Work

xDUCON focuses on three research themes: the enforcement of usage control policies, the use of the SDS model in dynamic networks, and finally on the man-

agement of a crisis scenario. Therefore, in the following we compare xDUCON with the most prominent research efforts for each theme.

xDUCON policy model is inspired by the UCON_{ABC} model described in [9]. UCON_{ABC} model enables the dynamic revocation of access rights even during ongoing accesses and takes into account the mutability issue providing per-, on-, and post-actions that are used for updating the attributes of subjects and targets. However, the xDUCON framework provides an enforcement model that is closer to an actual implementation than the UCON_{ABC} model.

As for industrial approaches, the eXtensible rights Markup Language (XrML) [16] and the eXtensible Access Control Markup Language (XACML) [1] are worth mentioning. Both languages are XML schema-based and provide a transactional mechanism for access control. Although both languages lack the capability of expressing ongoing conditions for long-lived accesses it is still possible to use XACML in combination with a PEP that requires re-requesting access permission for a resource at periodic intervals. xDUCON provides a more elegant approach where the SDS abstraction coordinates the interaction between the PEP and PDP to revoke the access rights when it is required.

In literature, there are several examples of policy-based approach for access control. For instance, in Ponder2 [12] authorisation policies can be specified based on (subject,target,action)-triple. Moreover, conditions can be specified that need to be satisfied before the access is made. The Ponder2 authorisation policies do not support ongoing conditions for evaluating long-lived access. Therefore, the model does not support the revocation of access rights but it is up to the subject to release them. Ponder2 supports Event-Condition-Action (ECA) policies that specify actions to be executed when a specific event is raised and a given condition is true. This mechanism could be used for revoking the access rights to subject during a long-lived session but again a modified PEP is required to support the revoking action. However, these policies cannot be directly used for updating the subject and target attributes as consequence of an access because they are completely independent of the authorisation model of Ponder2.

The UCON_{ABC} model also supports *obligations* similarly to the concept defined by Bettini et al. in [3]. An obligation is interpreted as an action that has to be fulfilled after the authorisation has been granted. Other models, as the one presented in [6], have introduced the term *provision* to indicate actions to be executed before the authorisation can be granted. In xDUCON, obligations can be captured as conditions defined in the xDPolicies. The fulfilment of an obligation can be represented by the presence of a corresponding obligation tuple in the space. During evaluation of a xDPolicy, the PDP can look up for the obligation tuple and therefore verify the fulfilment of the obligation.

The Cross-Domain Language (CDL) described by Thomas and Tsang in [15] provides a framework for on-demand and rapid sharing of sensitive data. The language is specifically targeted to document release across different domains and it is centred on an information-flow rather than a typical subject-object view of authorisations. The language provides constructs for specifying sender-side and receiver-side policies. It supports regrading operations for sanitising

document content. Moreover, it supports the specification of obligations to be fulfilled before or after the release of a document. At this stage, the language still lacks the capability to specify ongoing access control and actions for the mutability issue.

Lime [10] is one of the first approaches to exploit the SDS model for the interaction in a dynamic network setting. Each mobile device sports a private SDS that is merged with the SDS of other devices in range to be used as a common interaction space to exchange tuples. The approach used in TOTA [8] exploits the dissemination of tuples in a network to carry contextual information. TOTA approach is not only limited to networks of mobile nodes but can be generalised to any peer-to-peer network. Although xDUCON is presented here in a mobile context, it can be deployed in any type of network. The basic idea of mediating the evaluation and enforcement of policies through the SDS is that we can exploit the uncoupling propriety of tuples: policy tuples can be disseminated together with the data tuples to be available where they are needed. Worth of mentioning in this context is the approach presented in CAST [14]. Here, the SDS model is used to coordinate the execution of workflows that are distributed on mobile devices in the presence of scarce network availability. This is very close to the approach presented here where xDUCON is used for coordinating the enforcement of usage control policies.

To circumvent the issues related to limited connectivity typical of a crisis scenario, several studies have been conducted to demonstrate the feasibility of opportunist networks (oppnets) [2]. In an oppnet, the devices deployed in the area are used for carrying the data and to share it with other peer nodes when in communication range. However, data protection schemes adopted in oppnets are intended to protect the confidentiality of data shared between a specific source and a specific sink [7].

This work stems from the idea presented in [13] where an encryption and policy evaluation scheme for data disseminated in crisis scenarios is proposed. However, we here use xDUCON as the enforcement and coordination layer for data and policy dissemination.

5 Conclusions and Future Research

In this paper, we presented xDUCON a framework for the specification and enforcement of usage control policies. The framework is based on the concept of the SDS, where tuples are used for representing subjects, targets, and policies. The data and policies are not statically bound to a specific location but can be easily propagated to any locations. This feature suits well the needs of a decentralised dissemination and enforcement of policies for controlling the accesses to data gathered and distributed during the management of a crisis scenario. In this context, it is not always possible to rely on a wide-area communication facility because the infrastructure could have been damaged or not present at all.

As future work, we are currently working on an implementation of the framework to further validate our ideas. Moreover, we are also planning to add some cryptographic capabilities to the xDSpace. Currently the protection of the data is assumed to be managed by the correct enforcement of the policies associated with it. However, if such a mechanism is compromised then it is possible to access the data. Adding cryptographic capabilities to the framework would further protect the data confidentiality despite the presence of a compromised node.

6 Acknowledgments

We acknowledge financial support from the EC Consequence project (Grant Agreement 214859). We are grateful to the reviewers for their detailed and constructive comments that helped improve this paper.

References

1. eXtensible Access Control Markup Language (XACML) Specification 2.0. Available at http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml. 2005.
2. N. Aschenbruck, E. Gerhards-Padilla, M. Gerharz, M. Frank, and P. Martini. "Modelling mobility in disaster area scenarios." In *Proceeding of the 10-th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 4–12, 2007.
3. C. Bettini, S. Jajodia, X. Wang, and D. Wijesekera. "Obligation Monitoring in Policy Management." In *POLICY 02: Proceedings of the Third IEEE International Workshop on Policies for Distributed Systems and Networks*. 2002.
4. D. Gelernter. "Generative Communication in Linda." *ACM Trans. Prog. Lang. Syst.*, 7(1):80–112, 1985.
5. R. Graubart. "On the need for a third form of access control." In *12th National Computing Security Conferece*, pp. 296–304, Baltimore, MD, USA, 1989.
6. M. Kudo and S. Hada. "XML document security based on provisional authorization." In *Proceedings of the ACM Conference on Computer and Communications Security*. 2000.
7. L. Lilien, Z. Huma Kamal, V. Bhuse, and A. Gupta. "Opportunistic Networks: The Concept and Research Challenges in Privacy and Security" ." In *Proc. of Intl. Workshop on Research Challenges in Security and Privacy for Mobile and Wireless Networks (WSPWN 2006)*, Miami, March, 2006.
8. M. Mamei, F. Zambonelli, and L. Leonardi. "Tuples On The Air: A Middleware for Context-Aware Computing in Dynamic Networks." In *Proceedings of the 23rd International Conference on Distributed Computing Systems (ICDCSW '03)*, p. 342, Washington, DC, USA, 2003.
9. J. Park and R. Sandhu. "The UCON_{ABC} usage control model". In *ACM Trans. Inf. Syst. Secur.*, 7(1):128–174, 2004.
10. G. P. Picco, A. L. Murphy, and G.-C. Roman. "Lime: Linda Meets Mobility." In *Proc.* 21st Int'l Conf. on Software Engineering (ICSE'99), ACM Press, pp. 368-377, Los Angeles (USA), D. Garlan and J. Kramer, eds., May 1999.
11. A. Rowstron and A. Wood. "Solving the Linda Multiple rd Problem". In *Coordination Languages and Models*, pp. 357–367, 1996.

12. G. Russello, C. Dong, and N. Dulay. "Authorisation and conflict resolution for hierarchical domains". In *POLICY 07: Proceedings of the Eighth IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 201–210, Bologna, Italy 2007.
13. E. Scalavino, G. Russello, R. Ball, V. Gowadia, E. C. Lupu. "An Opportunistic Authority Evaluation Scheme for Data Security in Crisis Management Scenarios". To appear in *5th ACM Symposium on Information, Computer and Communication Security (ASIACCS)*, Beijing, China, April 2010.
14. R. Sen, G. Hackmann, M. Haitjema,G. Roman, and C. Gill "Coordinating Workflow Allocation and Execution in Mobile Environments." In *Proceeding of the 9th Int'l Conf. on Coordination Models and Languages*,pp. 249–267, Paphos, Cyprus, June 2007.
15. R. Thomas, G. Russello, S.Tsang. "Realizing the CDL Cross-Domain Language in the Ponder2 Policy Framework: Experiences and Research Directions." In *IEEE International Symposium on Policies for Distributed Systems and Networks, 2009* pp.76–83, London, U.K., 2009.
16. X. Wang, G. Lao, T. DeMartini, H. Reddy, M. Nguyen, and E. Valenzuela. "XrML–eXtensible rights Markup Language." In *XMLSEC: Proc. ACM workshop on XML security*, pages 71–79, New York, NY, USA, 2002. ACM.