PAES: Policy-Based Authority Evaluation Scheme

Enrico Scalavino, Vaibhav Gowadia, and Emil C. Lupu

Department of Computing, Imperial College London
{escala,vgowadia,e.c.lupu}@imperial.ac.uk

Abstract. *Enterprise Rights Management (ERM)* systems aim to protect disseminated data even after it has been sent to remote locations. Existing systems are based on common components, have similar functionalities and often have two shortcomings: a centralised architecture and a lack of concern for the trust and privacy of data recipients. To access the data, recipients must present their credentials to a policy evaluation authority, which they cannot choose and may not trust. Furthermore, recipients may be unable to access the data if their connection is intermittent or if they are off-line. To address these limitations, we propose *PAES: a Policy-based Authority Evaluation Scheme*, which combines data protection with a distributed policy evaluation protocol. The result allows us to implement the sticky policies paradigm in combination with trust management techniques. This permits distributing policy evaluation over a flexible set of authorities, simultaneously increasing the resilience of policy enforcement.

1 Introduction

Organisations and individuals gather and exchange data on a daily basis to conduct their business and their life. Information about customers, users, collaborators and competitors is gathered and stored; new data is produced through business activities or received from third parties. This data is often exchanged with clients and collaborators and needs to be adequately protected even after it has been delivered to them.

This problem has been gaining increasing attention in recent years. Controlling data usage after dissemination to remote parties is a challenge which underpins Enterprise Data Sharing, Privacy and Digital Rights Management (DRM). Both academia and industry have proposed several approaches to the problem, which is often referred to in industry as *Enterprise Rights Management* (ERM). Although these approaches vary on aspects such as policy/rights deployment and data protection mechanisms, common design principles and functionalities can be identified. First, data is associated with access rights (or usage control policies) and cryptographically protected before distribution to recipients. Then, a central *trusted authority (TA)*, often the originator himself, responds to access requests, evaluates recipients' rights and issues the decryption keys. A trusted component running on the recipient devices that we will refer to as the *virtual machine (VM)*, ensures that rights are locally enforced. Specific architectures differ in their management of user authentication, policy and rights retrieval, audit of user actions and other tasks [4,16,17,18,23].

Although used in many ERM products and in research, this architecture has several limitations. First, the set of TAs is statically and explicitly defined. Changes in this set,

E. Gudes, J. Vaidya (Eds.): Data and Applications Security 2009, LNCS 5645, pp. 268–282, 2009.

such as adding a new TA or removing an existing one, require defining new agreements, establishing new trust relationships and implementing them in terms of key exchanges, trust records etc. A second issue concerns the availability of the TA: if recipients cannot contact the specified TA, they cannot access the data received. Furthermore, current protocols assume that recipients are willing to present their credentials to the authority designated by the data originator even when credentials may be confidential. A last issue arises in particular in organisational environments where partners do not know the internal structure of the organisation with whom they interact. In such situations, it is difficult for the originator to specify access rules based on internal roles or to specify which entities in the partner organisation are entitled to issue or verify specific credentials. Instead, recipients are typically better placed to choose which authority could better evaluate their credentials amongst a scope of *acceptable* authorities defined by the originator.

To address these issues we propose a *Policy-based Authority Evaluation Scheme (PAES)*. Its design is motivated by the idea that authority to evaluate policies can be granted by other policies associated with them in the same manner as access to data is granted by policies associated with the data. Therefore, in addition to specifying the criteria recipients must meet to access the data, PAES also permits data originators to specify the criteria that authorities must meet to be trusted to evaluate policies. The set of TAs is therefore a dynamic set defined by characterisation and this confers increased flexibility in finding authorities mutually trusted by both data originators and recipients.

In PAES any entity that satisfies the originator's requirements can act as a policy evaluation authority. The same principle can be generalised to attribute and identity certification. In PKI infrastructures, authorities certify the association of an identity (or attribute) to a public key. Users trust certificates issued by authorities but authorities issue certificates according to their own policies and practices. Thus certificates from different authorities have different meanings and varying confidence in the associations they certify. In PAES, authorities are only trusted to evaluate the policies they receive and are designated by other policies. Therefore, authority hierarchies can be built according to the user's policies and the certificates issued carry both the authority signature and references to the policies they are based on. Recipients of a certificate can thus verify whether an authority is part of a specific hierarchy but also whether the policy used by the authority satisfies their requirements. This is possible in so-called policy-based PKIs [13], but the policies included in certificates are still decided by the authorities, not the users.

In this paper, we aim to describe design principles that can lead to more flexible data dissemination control and address the issues discussed above. Therefore, we first present PAES as a general approach before proposing an implementation. The rest of the article is organised as follows: related work is presented in Section 2 whilst Section 3 describes an application scenario providing a context for the examples; Section 4 gives a general overview of our approach while Section 5 describes the protocol and the data protection mechanism we propose to implement PAES. Section 6 introduces a possible policy language to represent PAES policy chains. A general discussion on the proposed solutions is given in Section 7. Finally, conclusions are drawn in Section 8, which also briefly discusses future work.

2 Related Work

Park et al. [18] proposed guidelines for ERM architecture design whose central concepts are the *virtual machine (VM)*, *control set* and *control center*. Virtual machines are software components running on the recipient's devices. Control sets are lists of usage policies or rights the VM enforces. Control centers are similar to what we previously called TAs. Most existing solutions comprise these components and vary on the configuration, responsibilities of these components and their implementation in software or in hardware.

Most ERM products, such as the Authentica [4], Liquid Machines [2] or Microsoft RMS (Rights Management System) [16] originate in industry. MS RMS is perhaps representative of their design. Its architecture is centralised and based on the deployment of publishing servers that broadly correspond to TAs and that issue encryption keys to users authorised to disseminate data and decryption keys to users authorised to access it. Before disseminating data, users *publish* it on a publishing server. The server then creates a *publishing licence* that contains an Access Control List for the data, a reference to the publishing server and the key used to encrypt the data (protected for the server itself). The user can freely distribute the licence to recipients that use it to contact the publishing server and obtain the access rights and decryption keys protected for each recipient in a *use licence*. Other systems differ mainly in the expressiveness of the authorisation policies, the authentication mechanisms employed, the policy deployment methods and the techniques used to store credentials.

Different research proposals have also been made. Yu et al. [24] presented the *Display-Only File Server (DOFS)* where data processing is performed on remote trusted servers and clients only receive snapshots, i.e. partial views of the actual data. The principle behind this design is that the actual data never leaves the trusted server. However, the solution presents similar shortcomings to existing systems as snapshots are only a subset of the original data with different formatting.

More recently, several studies have focused on trusted computing (TC) [22,10] and its applicability to ERM systems. Sandhu et al. [20] defined a family of Policy, Enforcement and Implementation (PEI) frameworks. TC is used to seal the decryption keys for the data into trusted VMs. VMs are then trusted to correctly authenticate recipients and to not disclose the keys. The authors later extended the proposed models allowing user-based access control policies and describing a specific implementation [21].

Casassa Mont introduced the use of sticky policies and of Identity Based Encryption (IBE) [7] [19] to transform policies into encryption keys and bind them to the encrypted data [17]. Decryption keys can then be issued only by a specific TA, chosen at encryption time, if the policies are satisfied. This approach avoids the data publication phase as all users know the TA's public parameters and can encrypt and distribute data. The Attribute-Based Encryption (ABE) scheme [8] [11] can be considered an evolution of IBE use, where decryption keys are generated through any credentials combination that satisfies the policy. With respect to IBE, ABE avoids contacting a TA for policy evaluation and key issuing, i.e. it enables off-line data access.

In contrast to the work described above, our approach aims to apply in ERM concepts derived from trust management frameworks such as the SPKI/SDSI [9], the RT [15] family of languages and SecPAL [6] [5]. More specifically we use the principle

that authority over a decision (e.g. an attribute assignment) can be delegated to entities satisfying specific criteria. However, when used in the context of ERM, the systems mentioned above have a number of shortcomings. First, their evaluation model relies on a central evaluator to which all the credentials must be presented. This extends the problem of credential confidentiality to all the entities in the chain. Second, the policies used for attribute assignment are not decided by the originator who thus loses control over the delegation sequence. In contrast, in PAES: i) entities are only delegated the right to evaluate pre-defined policies, ii) authority over a policy evaluation cannot be further delegated, iii) authority is granted by the positive evaluation of another policy itself evaluated by an authorised entity. Intuitively, this recursive process generates an evaluation/authority chain similar to those introduced by Trust Management systems.

3 Application Scenario

We consider two fictional organisations, the Sacred Heart Hospital (SHH) and the Medical Research Centre (MRC), co-operating to develop a new drug. MRC develops new drug formulations while SHH runs trials with patients and feeds back the results. The process is iterative so trial results affect new drug formulations and information is exchanged between the two partners at each iteration. MRC needs to access the records of patients involved in the trial to assess their response but also to review symptoms, side-effects and potential interference with other medications. SHH needs access to drug formulations to evaluate potential effects on the patients' health. Both SHH and MRC manage confidential data. SHH manages medical records governed by legislation such as HIPAA in US. MRC could abuse access to these records to influence the trials' outcomes or to advertise other drugs. On the other hand, MRC's data on drug formulations and their effects on patients is commercially sensitive. In this context, successes represent a competitive advantage while failures and negative side-effects are a source of embarrassment. Therefore both confidentiality and integrity of the data must be protected regardless of the administrative domain in which the data resides.

Figure 1 shows a particular interaction in this scenario where patients' case histories are distributed to personnel working on the project. In particular we consider SHH (and its doctors) as the data originators and MRC biochemists as the data recipients.

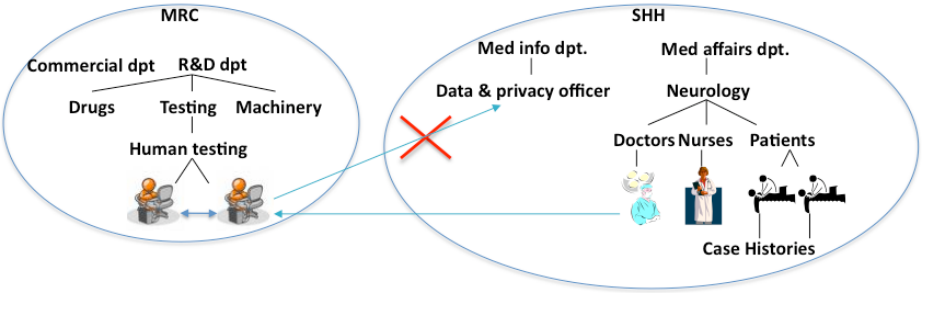

Fig. 1. Cooperative Scenario Example

Existing ERM frameworks (see Section 2) would typically force biochemists to present their credentials to a precise TA in SHH (e.g., the data protection officer) to obtain access rights to the data. This is not suitable for two reasons. First, the SHH data protection officer must authenticate data recipients ensuring their role and participation in the project. For this, it should have knowledge of MRC staff or of MRC authorities on this matter. Second, biochemists must agree to send their credentials (e.g. including on all their skills and competencies) to the data protection officer in SHH that they may not trust with their personal information. Even if the effect of credential disclosure could be mitigated through the use of trust management systems, this solution in not viable in our scenario because of the many shortcomings already described in Section 2. For example, a trust management policy may specify that whoever is trusted as a biochemist by MRC, should have access to the patient case file. The patient however has no control or knowledge of the criteria MRC uses in its trust decisions. MRC might delegate them to a third entity (e.g. an employment agency for scientists, SEA) the patient neither knows nor trusts. When a biochemist, Alice, receives the data her credentials as a biochemist are certified by SEA, which is trusted by MRC, but they must still be presented to an authority in SHH for evaluation. This violates not only Alice's privacy, but also reveals the business relationship between SEA and MRC. Some ERM systems such as MS RMS may allow SHH to define a TA internal to MRC. However, this does not mean that biochemists necessarily trust this TA with their personal information. In fact, they may even not trust TAs defined by MRC itself. Moreover, cross-organisational trust relationships must be set up statically. In case of intermittent or absent connectivity, or unavailability of the designated TA, biochemists cannot access the data.

PAES aims to allow MRC biochemists to be authenticated through any chain of authorities whose root is directly trusted by the data originator and to choose any entity they trust to evaluate them, provided that it meets some specified criteria. We present a more detailed example in Section 4.

4 The Policy Authority Evaluation Scheme

An *access (or usage) control policy* defines a set of criteria that must be satisfied to perform a specific action. Policies expressed and enforced by existing languages and control systems [1,3,12,14] mostly convey this meaning. Therefore, each policy comprises a target object, an action and a set of access conditions and can be defined as:

Definition 1. *A policy p is a tuple (t,a,c) where t is a protected target object, a is an operation that can be executed on the object and c denotes a set of conditions constraining the operation execution.*

Let R_p be the set of entities satisfying policy p, i.e. that can perform action a on object t. Note that the set changes continuously as the conditions in c can be either satisfied or not at different moments in time.

Definition 2. *An Evaluation Authority EA for policy p is an entity trusted to evaluate and enforce p.*

An evaluation authority can be explicitly defined if the policy writer directly knows and trusts it, or can be defined by characterization if the policy writer only knows which characteristics an entity must present to be trusted. Let DEA_p be the set of directly trusted evaluation authorities for a policy p and EA_p contain the set of directly trusted authorities and those defined by characterization. In the following we will use the terms *ea* and *TA* interchangeably. PAES' basic idea can be easily conveyed by adapting definition 2 to definition 1:

Definition 3. *Let p_i and p_j be two policies such that $p_j = (p_i, eval, c)$, where eval represents the policy evaluation action. Let ea be an entity acting in the system where p_i and p_j are defined. Then $ea \in EA_{p_i} \iff ea \in DEA_{p_i} \vee ea \in R_{p_j}$.*

The above definitions allow us to introduce the new concept of *policy chain*:

Definition 4. *A policy chain is a sequence of n policies $p_1 \rightarrow \ldots \rightarrow p_n$ such that $\forall p_i | i = 2 \ldots n : p_i = (p_{i-1}, eval, c)$.*

Thus each policy p_i ($i \geq 2$) is an *authority policy* that specifies the requirements an entity must satisfy to be considered an evaluation authority for policy p_{i-1} and p_1 is the *access control policy* for the data. Policy p_n, i.e. the last policy in the set, is always evaluated by an authority directly trusted by the policy writer (i.e. $EA_{p_n} = DEA_{p_n}$). Figure 2 illustrates the general format of a policy chain and the corresponding sets of evaluation authorities.

According to these definitions, an entity e is an evaluation authority that can vouch for the satisfaction of policy p_i by other entities (i.e. $e \in EA_{p_i}$) , if (i) it satisfies p_{i+1} evaluated by an evaluation authority $ea_j \in EA_{p_{i+1}}$, or (ii) the policy writer directly entitled it, i.e. $e \in DEA_{p_i}$.

When applying policy chains to existing rights management systems the set of authorities is no longer static but automatically changes when entities no longer satisfy the specified requirements or when new ones that do appear. Users (or evaluation authorities) that must be evaluated against a policy can choose their trusted evaluator among those satisfying the higher level policy and those being explicitly listed by the policy

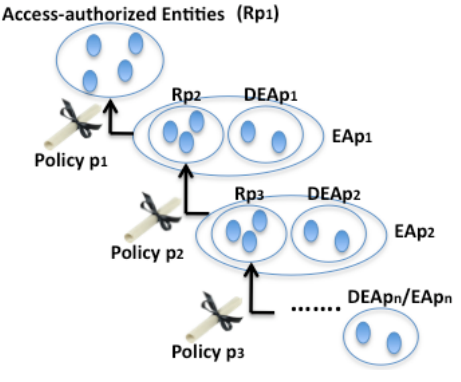

Fig. 2. Policy-defined groups of authorities

writer. They can for example choose the authority they trust the most with their confidential credentials. Moreover, evaluation chains can be built over different paths (i.e. with different authorities). If an authority goes offline or cannot be reached it can be easily replaced by another one satisfying the same requirements. This partly solves network problems, even if it still does not allow recipients to access the disseminated data if their devices are offline or cannot reach any authority.

With respect to trust management schemes such as TPL or RT, the trust relationship linking the policy writer to the data recipient still exists but in a restricted form. Data originators decide upon the policies composing the chain thus effectively controlling the delegation. For example, TPL allows specifying the trusted credentials but not the criteria according to which credentials are issued and who (apart from the policy writer himself) can evaluate if the criteria are met. We discuss other advantages of PAES in the next section where we also present a possible implementation of the scheme.

Figure 3 shows an example chain from the scenario described in Section 3. Each level specifies the entities authorised to evaluate the previous policy, either as an explicit list, a higher-level policy, or both. The figure shows a possible SHH corporate policy governing access to patients' records specifying that only MRC biochemists specialised in stem cells research can access them. SHH lets each patient agree with the policy and choose a set of entities he trusts to evaluate it. The patient considered in the example chooses Bob, one of his closest friends, to determine whether recipients satisfy the requirement. If Bob is a journalist most researchers would not want him to know they work on such a delicate research area. However, the policy also allows biochemists to be evaluated by any research facility working on stem cells research ranked by USNews as one of the best hospitals. Evaluation of whether a research facility satisfies this policy is left to SHH. SHH and the research facility may however be competitors, and the facility may not trust SHH to know what kind of research projects it is carrying out. Therefore the policy also allows research facilities to be evaluated by any publicly recognised organisation giving grants for medical research. Whether a company or organisation satisfies this requirement is directly left to verify to the European Union Research Commission (EURC). Note that SHH and patients can define authorities according to their liking and that they do not need to know in details the structures of the organisations involved but only sufficiently to state their requirements.

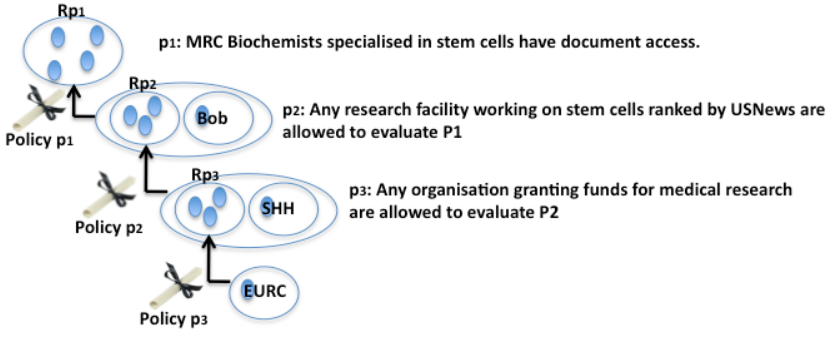

Fig. 3. Policy chain example

In the following section we describe a possible implementation of a data protection mechanism and protocol based on PAES that permits the evaluation and enforcing of policy chains.

5 A Possible Implementation

To implement the principles described above we propose a three phase protocol comprising: (i) the policy chain definition, (ii) the data protection and (iii) the policy chain evaluation. The first two phases concern protecting the resource from being accessed by non-authorised subjects. The third phase concerns policy evaluation and enforcement after the data has been received by recipients. The protocol is intended to be integrated with existing ERM systems.

Before describing the various phases, we introduce the basic notations and definitions necessary to formalise the protocol. Let E = $\{e_1, \ldots, e_e\}$ be the entities exchanging and accessing data and/or evaluating policies. Let P = $\{p_1, \ldots, p_p\}$ be the set of all possible policies. A policy evaluation is a boolean function $eval : E \times E \times P \rightarrow \{true, false\}$ such that $eval(e_l, e_m, p_i)$ denotes that e_m satisfies (or does not satisfy) the conditions imposed by p_i according to the evaluation made by e_l.

K denotes a symmetric encryption key and $\{D\}_k$ the data D encrypted with k. PK_e and PK_e^{-1} denote the private and the public key of an entity $e \in E$, and ID_e e's public key identity certificate (as in any PKI).

5.1 Policy Chain Definition Phase

At first, the data originator specifies a chain of policies $p_1 \rightarrow \ldots \rightarrow p_n$ as defined in Section 4 and an ordered list of sets of entities $\{DEA_{p_1}, \ldots, DEA_{p_n}\}$ such that each entity $ea_{ij} \in DEA_{p_i}$ is directly trusted to evaluate p_i. The specified policy chain is combined with the list to obtain a set of several *policy levels* $1, \ldots, n$ (with $n \geq 1$) where each level i defines a pair (p_i, DEA_{p_i}).

5.2 Data Protection Phase

In the *data protection phase* the data originator generates a symmetric key k_i for each specified level i except level n (the root of the chain is always a set of directly trusted evaluation authorities whose public keys are known by the originator) and a further key k_0. Using these keys the data D is encrypted as described below. The result of the protection phase is a concatenation of message parts $m_0, m_1, \ldots, m_n$ where:

- $m_0 = \{D\}_{k_0}$;
- $m_i = p_i | DEA_{p_i} | \{info_i\}_{k_i} | \{info_i\}_{PK_{ea_{i1}}} | \ldots | \{info_i\}_{PK_{ea_{id}}}$, where d is the cardinality of DEA_{p_i}, $info_i = (H(p_i), k_{i-1})$ for $1 \leq i \leq n$. $H(p_i)$ denotes a secure hash function applied to policy p_i;
- $m_n = p_n | DEA_{p_n} | \{info_n\}_{PK_{ea_{n1}}} | \ldots | \{info_n\}_{PK_{ea_{nd}}}$.

For simplicity we used the DEA_{p_i} to represent the sets of directly trusted authorities. In the actual implementation (see algorithm 1) for each entity $e \in DEA_{p_i}$ we include

the certificate ID_e in message m_i (although for the purpose of the protocol, it would be sufficient to represent each entity with a public key and a URL). In the data package shown above, the first symmetric level key k_0 is used to protect the data while each subsequent key k_i is used to protect k_{i-1} (k_{n-1} is only protected with the public keys of the directly trusted evaluation authorities at level n). A copy of each symmetric level key k_{i-1} is also protected with the public keys of the directly trusted evaluation authorities for p_i. Therefore, to obtain the data protection key k_0 each entity at level i must be authorised by an authority ea_{i+1}. At the root of the chain there is a directly trusted evaluation authority. Hashes of policies are included at each level to ensure policy integrity. Algorithm 1 describes the procedure to protect data according to a defined policy chain.

Input: Data D to be protected with an n-policy chain
$m_0 = \{D\}_{k_0}$;
for *i=1...n-1* **do**
  $info_i = (H(p_i), k_{i-1})$;
  $m_i = p_i | \{info_i\}_{k_i}$;
  foreach $ea_{ij} \in DEA_i$ **do**
    $m_i = m_i \,|\{info_i\}_{PK_{ea_{ij}}} | ID_{ea_{ij}}$;
  end
end
$info_n = (H(p_n), k_{n-1})$;
$m_n = m_n \,| p_n$;
foreach $ea_{nj} \in EA_n$ **do**
  $m_n = m_n \,|\{info_n\}_{PK_{ea_{nj}}} | ID_{ea_{nj}}$;
end
Output: $m_0 \ldots m_i \ldots m_n$

Algorithm 1. Data protection algorithm

The output of the data protection phase for the SHH patient's record in our example would then be:

m_0 $\{Record\}_{k_0}$ |
m_1 p_1(MRC biochemist working on stem cells) | $\{H(p_1), k_0\}_{k_1}$ | $\{H(p_1), k_0\}_{PK_{Bob}}$ | ID_{Bob}
m_2 p_2(Stem cells facility with USnews rank ≤ 10) | $\{H(p_2), k_1\}_{k_2}$, $\{H(p_2), k_1\}_{PK_{SHH}}$ | ID_{SHH}
m_3 p_3(Recognised org. granting funds) | $\{H(p_3), k_2\}_{PK_{EURC}}$ | ID_{EURC}

5.3 Policy Chain Evaluation Phase

The *policy chain evaluation* is a recursive procedure comprising an initial forward process (*policy evaluation*) and a final backward process (*key disclosure*). The policy evaluation finds a chain of entities satisfying the policies at each level. The initiator is the data recipient e_0 that receives the data with the messages $m_0, m_1, \ldots, m_n$ and tries to access D. To do so it first looks for an entity $ea_{1j} \in DEA_{p_1}$ he trusts and that grants him access under p_1. If the search succeeds e_0 sends m_1 to ea_{1j} (in practice: fields $\{info_i\}_{PK_{ea_{ih}}}$ for different authorities ($h \neq j$) are no longer useful and removed). The evaluation authority ea_{1j} can then decrypt and return k_0 with which e_0 can access D. Otherwise, e_0 can choose any entity $e_1 \in E$ he trusts and that grants him access under p_1. Then e_0 removes message m_0 and sends the rest of the messages to e_1. To return k_0, e_1 must first access

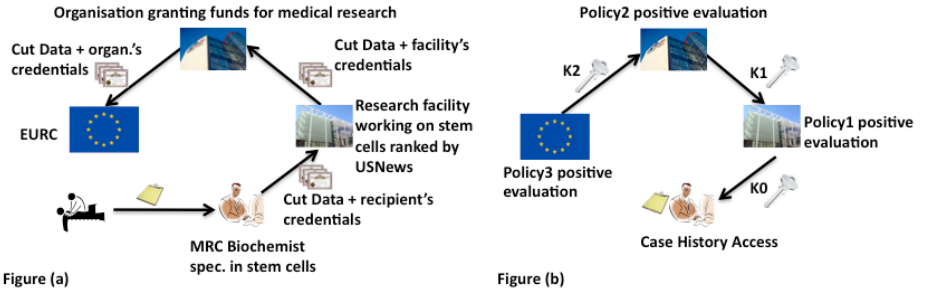

Fig. 4. Policy evaluation recursion

k_1 and therefore iterates the process (i.e. it looks for an entity that positively evaluates it under p_2). The sequence of evaluations terminates whenever an entity $ea_{ij} \in DEA_{p_i}$ returns a positive evaluation. To summarise, the forward iteration process finds a chain of entities $e_0, \ldots, e_h$ (if it exists) s.t. : (i) $1 \leq h \leq n$, (ii) $eval(e_{i+1}, e_i, p_{i+1}) = true$, (iii) e_i trusts e_{i+1} $\forall\, 0 \leq i < h$, and (iv) $e_h \in DEA_h$.

In the key disclosure process each entity e_i (where $i \neq n$) satisfying p_{i+1} is added to the set of evaluation authorities EA_i for policy p_i as it satisfied the criteria to be authorised to evaluate p_i (i.e. p_{i+1}). This is done by allowing each entity to get the symmetric level key necessary to disclose the information at the preceding level (i.e. k_{i-1}). This process is initiated by the last entity in the chain e_h. Entity e_h can directly access the information $info_h = H(p_h), k_{h-1}$ encrypted with its public key. It can then send k_{h-1} to entity e_{h-1} that will disclose k_{h-2} iterating the process. Each entity e_i in the chain sends to entity e_{i-1} the key k_{i-1} to decrypt the information $info_{i-1}$ until $info_0$, i.e. the data, is decrypted. Note that at each backward iteration the hash of each policy p_i is checked.

Figure 4 shows the execution of the two phases in our example. The recipient of a patient's record removes m_0 and sends the data, along with his credentials (certifying that he is a biochemist working for MRC and specialised in stem cells) to a research facility. Since policies are public (see Section 7 for motivation), he should look for a facility running a project on stem cells and ranked by USNews, as specified in the requirements. We call this entity e_1. e_1 verifies that the recipient is a MRC biochemist specialised in stem cells, removes message m_1 and sends the data, along with its credentials, to an organisation that funds research projects. We call this entity e_2. e_2 verifies that e_1 is a facility working on stem cells and ranked by USNews, removes message m_2 and sends the data, along with its credentials, to the European Union Research Commission (e_3). The EURC can directly access $h(p_3)$ and k_2 and verify that e_2 is a company whose research grants are publicly recognised. This concludes the forward process.

During the backward process e_3 sends k_2 to e_2, who can now access and verify the integrity of p_2. If the policy was not tampered with it can send k_1 back to e_1 that can now access and verify the integrity of p_1. If the policy was not tampered with it can send k_0 back to the data recipient that can now access the document.

Each entity trusts the one at the next level to see its credentials and to correctly verify if it satisfies the requirements specified in the policy. Trust however is not transitive since it is strictly related to the evaluation of a policy which involves only two nodes.

Credentials sent from an entity to an authority to be evaluated are in fact not further disseminated and confidential information is accessed only by authorities trusted to do so. Finally, since at each step a part of the data package is removed, entities at each level can only access the information they have been authorised for. With respect to trust management systems, recipients no longer need to gather credentials from every entity involved in the chain. At each step two different entities (evaluator and evaluated entity) perform a part of the protocol and evaluate part of the policy chain. At the end, the data recipient only has to present his own credentials to the chosen authority and does not worry about the other entities being part of the chain.

6 A Policy Language for Policy Chains

We propose a language that can be used to express policy chains, i.e. to combine basic policies in a form that can be interpreted during the PAES data protection phase. Chains expressed in this language convey the same semantic for policies and policy chains as that described in Section 4 but in a more readable syntax. Although the syntax is loosely based on the SecPAL language, the evaluation is performed according to the protocol described in the previous section. A PAES policy chain is specified as a sequence of policies, where each policy is the combination of a *permission* and a *policy statement*:

Subject **says** permission **if** policy statement

A permission represents the assignment of a right to a subject in the form *subject can do action* while a *policy statement* represents the corresponding conditions. Note that the policy-chain language is agnostic to the syntax used to specify policy statements. Any usage control policy language could be used provided that evaluation authorities can interpret it. We express the authorisation to access data as:

PolicyWriter **says** x **can** access data **if** $P_1(x, data)$

where $P_1(x, data)$ is a policy that grants x access to the data. Authority policies are expressed as:

PolicyWriter **says** y **can** say $P_1(x, data)$ **if** $P_2(y, P_1)$

where $P_2(y, P_1)$ is a policy that grants y the right to verify if x satisfies policy P_1. In contrast with SecPAL, note that delegated evaluation rights cannot be delegated further. A policy chain can be simply represented as:

PolicyWriter **says** x **can** access data **if** $P_1(x, data)$,
PolicyWriter **says** x **can** say $P_1(y, data)$ **if** $P_2(x, P_1)$, PolicyWriter **says** ea_{1j} *cansay* $P_1(y, data)$..
PolicyWriter **says** x **can** say $P_2(y, P_1)$ **if** $P_3(x, P_2)$, PolicyWriter **says** ea_{2j} *cansay* $P_2(y, P_1)$..
.....
PolicyWriter **says** $ea_n j$ **can** say $P_n(y, P_{n-1})$..

where the terms $ea_{1j}...ea_{nj}$ are used to indicate the evaluation authorities directly trusted by the policy writer. Using a hypothetical language for policy statements, we can now represent the policy chain and set of directly trusted authorities for our example as:

SHH **says** x **can** access case-history
if (x.role = Biochemist AND x.org = MRC AND x.spec = stem-cells),
SHH **says** Bob **can** say (y.role = Biochemist AND y.org = MRC AND y.spec = stem-cells)
SHH **says** x **can** say (y.role = Biochemist AND y.org = MRC AND y.spec = stem-cells)
if (x.facility = Research AND x.activity = stem-cells AND x.USNewsrank $\leq$ 10),
SHH **says** SHH **can** say (y.facility = Research AND y.activity = stem-cells AND y. USNewsrank $\leq$ 10),
SHH **says** x **can** say (y.facility = Research AND y.activity = stem-cells AND y. USNewsrank $\leq$ 10) **if** (x.role = PublicResFunder),
SHH **says** EURC **can** say (y.role = PublicResFunder)

Note that the example does not specify which entities must issue the required credentials (e.g. the public recognition as research funder being issued by the United Nations Institute for Training and Research, UNITAR, or USNews signing a rank certificate). This is typically addressed in the language adopted for policy statements.

7 Discussion

PAES represents a generalisation of the traditional ERM protocol. The originator (or an authority he trusts) could in fact be included as first directly trusted authority in the chain or second for collaboration between two domains (the first one being the authority in the partner domain), thus implementing the traditional ERM protocol.

Limitations. A limitation of PAES is that when one of the policies is evaluated negatively, the entity under evaluation must choose a different evaluator or return back to the previous entity a negative response. This also happens if no trusted evaluators can be found. Therefore, a policy chain evaluation may return a negative result even if the entity under evaluation actually satisfies the specified requirements. The flexibility that PAES offers to receive a negative response and try a different evaluation authority also raises a scalability issue. If n entities are required to evaluate a policy at each level of a chain spanning l levels, in the worst case the number of messages exchanged would be $O(n^l)$. However, we must consider that: i) an evaluation chain can be shorter than the corresponding policy chain (e.g. if the chosen evaluation authority is directly trusted by the policy writer), ii) policy recipients can filter out the entities they do not trust or that do not probably satisfy the chain requirements, iii) policy chains for most application scenarios are very short. Note that PAES allows the originator to specify his/her own policies for ascertaining his/her trust at each level. Leaving the policy specification to the data originator places a burden on him/her, but in practice many users are likely to use similar policies. Furthermore, as our example shows, corporate or regulatory policies can be directly used while the originator can simply integrate them with "shortcuts", i.e. with the specification of authorities he trusts to perform the evaluations.

Although the PAES approach offers greater flexibility, data recipients are still limited in their choice of evaluation authorities to those acceptable to the originator as defined in the policies. A negotiation process between data recipients and originators could offer more flexibility but would not be feasible in a data dissemination environment where users may not know each other and may be online at different times. Moreover, this may require the data to be repackaged for each negotiating recipient and would result

in a set of disseminated packages containing the same information but being controlled by different rules.

Possible Attacks. PAES aims to protect the confidentiality of disseminated data but remains prone to an attack that does not affect data confidentiality but the system availability, i.e. a denial-of-service attack (DOS). PAES policies are attached to the data as clear text so that they can be evaluated before the data is sent to the upper level. This is done to avoid the construction of a chain that may fail in the backward process. Only policies' hashes are protected to verify policy integrity. If a policy is corrupted then its chain can be destroyed (as well as its copies already disseminated) and the resource consumption would correspond only to the cost of an evaluation chain construction. No protected information would be disclosed to non-authorised entities. However, an attacker may disseminate many tampered policy chains in the network, thus causing the consumption of a large amount of resources (an evaluation chain must be constructed to discover every tampered package). The best solution to mitigate DOS attacks is that of identifying the packages' sources. This can be done for example by having data originators sign all the policies in the chain. Therefore authorities could verify the signature before executing the protocol and, if the policies have been tampered with, they could simply include the originator's key in a blacklist or revocation list.

Applicability. PAES may have different applications. Our work was motivated by ERM systems where our solution allows data originators to define who can access the data before the dissemination, without necessarily knowing the result of the successive access attempts and the set of possible recipients. However, the same protocol could also be used for data dissemination in Peer-To-Peer and ad-hoc networks. Every peer could define the requirements recipients must satisfy to access the data and how the satisfaction of these requirements must be evaluated by other trusted peers. In this sense, a particular application of the protocol could be represented by data exchange through e-mails. In this case PGP (Pretty Good Privacy) keys could be used.

An other interesting application is that of privacy protection. Consider a user accessing a remote service that requires a form to be filled with his private information. Assuming the company issuing the service is willing to provide such protection (e.g. to attract more clients) it could let the user define his own protection policies for the data inserted in the form and protect it before it is sent to the server. This would allow him to specify that data can be accessed only under the national regulations as evaluated by a known and trusted independent authority.

8 Conclusions and Future Work

The fact that different organisations can work on joint projects does not imply that the single individuals and entities of each party directly trust each other. The same can be said for entities working for the same organisation. Existing ERM systems and more generally policy evaluation mechanisms assume instead the existence of such trust relationships and force users to disclose their credentials to unknown entities. PAES fills this gap by allowing data recipients to choose their own trusted policy evaluators among those satisfying criteria defined by the data originator. Also, it enhances existing solutions

by distributing the policy evaluation process, so that the set of evaluation authorities can dynamically change. This is useful whenever some authorities are unknown or unreachable. Finally, PAES implements the Sticky Policy paradigm allowing data originators to distribute data with no need of an online publication phase. We have thus shown that simple extensions to existing policy evaluation systems can bring improvements in terms of flexibility and resiliency of the evaluation process. However, these improvements also have a complexity cost in some cases.

Future work will focus on an evolution of the protocol to deal with threshold policies (i.e. policies that must be evaluated by more than one authority). We will also allow different parts of the same policy (e.g. connected by AND/OR operators) to be evaluated by authorities satisfying different requirements. This means realising policy trees rather than simple chains. We aim to further evaluate the use of signed credentials attesting policy chain evaluations and thus how PAES can be integrated with existing certification systems.

Acknowledgments

We acknowledge financial support from the EC Consequence project (Grant Agreement 214859). We are also grateful to our colleagues G. Russello, L. Mostarda and C. Dong for many useful discussions and constructive comments.

References

1. eXtensible Access Control markup language (xacml) (version 2.0, 2005), http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf
2. Liquid machines and microsoft windows rights management services (rms): End-to-end rights management for the enterprise (2006), http://www.cmdsolutions.com/pdfs/LiquidMachines%20Windows%20RMS%20Business%20White%20Paper%20FINAL%20060213.pdf
3. Ashley, P., Hada, G., Karjoth, S., Powers, C., Schunter, M.: The enterprise privacy authorization language (epal 1.1) - reader's guide to the documentation -. Technical Report 93951, IBM (2003)
4. Authentica. Enterprise rights management for document protection, White Paper (2005)
5. Becker, M., Fournet, C., Gordon, A.: Secpal: Design and semantics of a decentralized authorization language. Technical Report MSR-TR-2006-120, Microsoft (2006)
6. Becker, M., Fournet, C., Gordon, A.: Design and semantics of a decentralized authorization language. In: CSF 2007: Proc. 20th IEEE Computer Security Foundations Symposium, Washington, DC, USA, pp. 3–15 (2007)
7. Boneh, D., Franklin, M.: Identity based encryption from the weil pairing. In: Proc. 21st Annual Int. Cryptology Conference, Santa Barbara, USA, pp. 213–229 (2001)
8. Chase, M.: Multi-authority attribute based encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 515–534. Springer, Heidelberg (2007)
9. Clarke, D., Elien, J.-e., Ellison, C., Fredette, M., Morcos, A., Rivest, R.L.: Certificate chain discovery in spki/sdsi. J. of Computer Security 9 (2001)
10. Felten, E.W.: Understanding trusted computing: Will its benefits outweigh its drawbacks? IEEE Security and Privacy 1(3), 60–62 (2003)

11. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proc. 13th ACM Conf. on Computer and Communications Security, pp. 89–98. ACM, New York (2006)
12. Content Guard. extensible rights markup language (xrml) 2.0 (2001), http://www.xrml.org/
13. Housley, R., Ford, W., Polk, W., Solo, D.: Internet x.509 public key infra'structure certificate and crl profile. Request for Comments: 2459 (1999), www.ietf.org/rfc/rfc2459.txt
14. Iannella, R.: Open digital rights language (odrl), version 1.1. W3c note, World Wide Web Consortium (2002), http://www.w3.org/TR/odrl
15. Li, N., Mitchell, J.-C., Winsborough, W.H.: Design of a role-based trust management framework. In: Proceedings of the 2002 IEEE Symposium on Security and Privacy, pp. 114–130. IEEE Computer Society Press, Los Alamitos (2002)
16. Microsoft. Technical overview of windows rights management services for windows server 2003. White Paper (2005), download.microsoft.com/download/8/d/9/8d9dbf4a-3b0d-4ea1-905b-92c57086910b/RMSTechOverview.doc
17. Mont, M.-C., Pearson, S., Bramhall, P.: Towards accountable management of identity and privacy: Sticky policies and enforceable tracing services. In: DEXA Workshops, pp. 377–382 (2003)
18. Park, J., Sandhu, R.S., Schifalacqua, J.: Security architectures for controlled digital information dissemination. In: 16th An. Computer Security Applications Conf. (ACSAC), New Orleans, USA, p. 224. IEEE Computer Society, Los Alamitos (2000)
19. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
20. Sandhu, R.S., Ranganathan, K., Zhang, X.: Secure information sharing enabled by trusted computing and pei models. In: ASIACCS, pp. 2–12 (2006)
21. Sandhu, R.S., Zhang, X., Ranganathan, K., Covington, M.: Client-side access control enforcement using trusted computing and pei models. J. High Speed Networks 15(3), 229–245 (2006)
22. Schoen, S.: Trusted computing: Promise and risk (2003), http://www.eff.inorg/files/20031001_tc.pdf
23. Avoco Secure. Choosing an enterprise rights management system: Architectural approach (2007), www.windowsecurity.com/uplarticle/Authentication_and_Access_Control/ERM-architectural-approaches.pdf
24. Yu, Y., Chiueh, T.-c.: Enterprise digital rights management: Solutions against information theft by insiders. Research Proficiency Examination (RPE) report TR-169, Department of Computer Science, Stony Brook University (2004)