Trust Management and Whether to Delegate

Simon N. Foley

Department of Computer Science,
University College, Cork, Ireland.
s.foley@cs.ucc.ie

Abstract. Trust Management systems use security credentials to determine whether action requests are authorized. This paper examines how they might be used in determining whether requests for delegation are safe.

1 Introduction

Delegation certificates/credentials specify delegation of authorization between public keys. When an application receives a request to execute a particular action, then authentication notwithstanding, the application may use the credentials to determine whether the request is authorized. Trust management systems such as KeyNote [3] and SPKI [4] provide assistance to applications in making these decisions. For example, an application can request authorization judgments using the KeyNote API without having to be aware of the details of how to interpret credentials or the details of the underlying security policy. As an engineering paradigm, the desirability of such separation of concerns is well understood, as it can lead to applications that are easier to develop, understand and maintain.

In addition to judgments on authorization, applications may also desire assistance in determining whether it is safe to delegate authorization for some action to a public key, that is, to write and sign a new credential. As with authorization judgments, it is preferable that the details of these delegation judgments are separated from the application. In this paper we consider how a trust management system might provide such assistance.

The problem to be discussed is depicted in Figure 1. The owner of public key

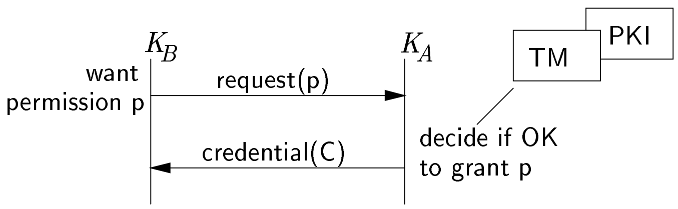


Fig. 1. Request for Delegation of Authorization

K_B requests authorization permission p from an application that owns public key K_A. K_A may use credentials presented by K_B to determine whether it is safe to sign a credential C delegating permission p to K_A (principles represented by their public key).

Section 2 discusses the trivial case where the credentials presented prove that K_A has already been delegated authorization p; writing a new credential is simply certificate reduction. Section 3 considers the case where K_A uses additional 'local' rules to determine if delegation is safe. When handling a request, K_A may be willing to offer an alternative, but similar, credential to K_A. Section 4 considers how techniques from similarity-based retrieval for case-based reasoning systems might be used to support flexibility in requests.

2 Defacto Delegation and Reduction

A simple model is used to represent delegation of authorization between public keys. A signed credential, represented as $\{\!|\, K_B, p \,|\!\}_{K_A}$, indicates that K_A delegates to K_B, the authorization permission p. Permissions are structured in terms of lattice $(PERM, \leq, \sqcap)$, whereby given $p \leq q$ means that permission q provides no less authorization than p. A simple example is the powerset lattice of $\{\mathsf{read}, \mathsf{write}\}$, with ordering defined by subset, and greatest lower bound defined by intersection.

Given a credential $\{\!|\, K_B, q \,|\!\}_{K_A}$ and $p \leq q$ then there is an implicit delegation of p to K_B by K_A, written as $(\!|\, K_B, p \,|\!)_{K_A}$. Two reduction rules follow.

$$\frac{\{\!|\, K_B, p \,|\!\}_{K_A}}{(\!|\, K_B, p \,|\!)_{K_A}} \qquad \frac{(\!|\, K_B, p \,|\!)_{K_A}; p' \leq p}{(\!|\, K_B, p' \,|\!)_{K_A}}$$

If delegation is regarded as transitive, and if K_A delegates p to K_B, and K_B delegates p to K_C, then it follows that K_A implicitly delegates p to K_C.

$$\frac{(\!|\, K_C, p \,|\!)_{K_B}; (\!|\, K_B, p' \,|\!)_{K_A}}{(\!|\, K_C, p \sqcap p' \,|\!)_{K_A}}$$

This corresponds to SPKI certificate reduction [4] (greatest lower bound is equivalent to SPKI tuple intersection). It is not unlike a partial evaluation over a collection of KeyNote credentials, resulting in another credential. At this point, we do not consider permissions that cannot be further delegated.

These rules are used by a trust management system to determine whether a request for an action (permission) is authorized. If the local policy specifies that K_A is authorized for p, and delegation $(\!|\, K_B, p \,|\!)_{K_A}$ can be deduced, then it follows that K_B is authorized for p.

The trust management system can be used to provide guidance when writing new credentials. From Figure 1, if the owner of K_A can derive $(\!|\, K_B, p \,|\!)_{K_A}$ from the credentials presented by K_B or via a PKI, then it can safely write and sign a new credential $\{\!|\, K_B, p \,|\!\}_{K_A}$ for K_B. Alternatively, K_A may write a weaker credential $\{\!|\, K_B, q \,|\!\}_{K_A}$ for the highest $q \leq p$ for which $(\!|\, K_B, q \,|\!)_{K_A}$ can be deduced.

We call this *defacto* delegation as it does not confer any additional authorization on the requester. Advantages of this kind of delegation have been discussed in [1]. It can be used to reduce the size of delegation certificate chains to be stored and/or presented; this may be useful for limited devices such as PDAs, and so forth. It can also provide for a degree of anonymity: long delegation chains may reveal sensitive information on how authority was acquired.

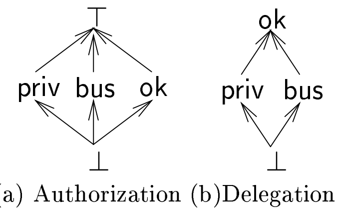
3 Dejure Delegation and Reduction

One difficulty with defacto delegation is that it does not help a principle to decide whether it is safe to confer new permissions on a requester. For example, a customer opening a new bank account must present an appropriate identity credential before a credential for transacting on an account can be issued. We call this *dejure* delegation: by providing appropriate credentials the requester can acquire additional authorization; authorization that is not defacto.

The principle and/or application system making a delegation decision requires guidelines on when it is safe to do a dejure delegation. For the purposes of this paper we make the simplifying assumption that these guidelines can be encoded as additional reduction 'rules' specific to the application.

Example 1 A communications company with public key K_C provides private (priv) and business (bus) services. Private customers must provide a certificate signed by a recognized introduction agency K_I. Business customers must provide a credit-worthiness certificate from rating agency K_R.

Alice (K_A) and Bob (K_B) have certificates $\{\!|\, K_A, \mathsf{ok} \,|\!\}_{K_I}$ and $\{\!|\, K_B, \mathsf{ok} \,|\!\}_{K_R}$, respectively. For the purposes of authorization testing, permissions priv, bus and ok are considered disjoint (Figure 2(a)) during reduction.



(a) Authorization (b)Delegation

Fig. 2. Permission Orderings

Alice requests private customer authorization priv and presents $\{\!|\, K_A, \mathsf{ok} \,|\!\}_{K_I}$. For the purposes of dejure delegation, K_C is willing to trust K_I as an introducer. In effect, delegation of ok (by K_I) implies priv. For flexibility, delegation of ok

(by K_R) implies bus and priv. This is characterized by providing additional rules specific to the application and modifying the permission ordering (Figure 2(b)).

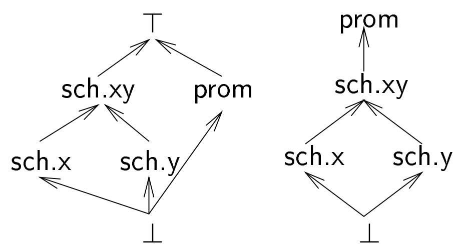
$$\frac{}{(\!| K_I, \mathsf{priv} |\!)_{K_C}} \qquad \frac{}{(\!| K_R, \mathsf{priv} |\!)_{K_C}} \qquad \frac{}{(\!| K_R, \mathsf{bus} |\!)_{K_C}}$$

Just as reduction can be used by an application to determine defacto delegation, it can also be used for dejure delegation. For example, by reduction, it is safe for K_C to write and sign $\{\!| K_B, \mathsf{bus} |\!\}_{K_C}$ since we have $(\!| K_R, \mathsf{bus} |\!)_{K_C}$, $\{\!| K_B, \mathsf{ok} |\!\}_{K_R}$ and $\mathsf{bus} \sqcap \mathsf{ok} = \mathsf{bus}$. $\triangle$

The advantages of characterizing dejure delegation as reduction is that the trust management system remains responsible for trust calculations, not the application. It can answer two kinds of query. The first is the 'normal' authorization query based on the presented credentials and defacto permission ordering. The second type of query is for defacto or dejure delegation (Figure 1). In this case reduction also uses the additional credential rules and dejure permissions. We assume that the defacto permission orderings are a subset of the dejure orderings.

Example 2 WebCom [8] provides architectural support to servers scheduling mobile computation and services to client systems. KeyNote [3] provides trust management: server credentials prove authorization (to clients) to schedule certain services; client credentials prove (to servers) the right to receive certain services [5]. Servers may 'promote' [7] clients to become servers in their own right. A client must present credentials proving its authorization to be promoted; if satisfied, a server then delegates service scheduling authority to the promoted client. An arbitrary client does not have defacto authority to schedule; it must acquire it from an authorized server.

Servers and clients trust a common authority K_W for authority to schedule and to be promoted. For example, authority for scheduling services x and y is granted to K_S as $\{\!| K_S, \mathsf{sch.xy} |\!\}_{K_W}$. Authority to be promoted is granted to K_C as $\{\!| K_C, \mathsf{prom} |\!\}_{K_W}$. Figure 3(a) defines the permission ordering used when carrying out a defacto reduction.



(a) Authorization (b) Delegation

Fig. 3. Permission Orderings

A server is willing to delegate scheduling authority to a client only if the client presents a credential with the promotable permission. In this special case the server is willing to recognize promotion as a delegation of scheduling authority. This is reflected by additional delegation rules that are used for determining dejure delegation, along with the existing credentials and the revised ordering in Figure 3(b).

$$\frac{}{(\!|\, K_W, \mathsf{sch.x} \,|\!)_{K_S}} \qquad \frac{}{(\!|\, K_W, \mathsf{sch.y} \,|\!)_{K_S}}$$

Thus, the dejure reduction to $(\!|\, K_C, \mathsf{sch.x} \,|\!)_{K_S}$ means that server K_S may promote K_C to schedule x services and write the corresponding certificate. △

4 Similarity and Delegation

A request for particular authorization may fail because it was not expressed in precisely the right way. For example, a request for a service might be rejected, despite the requester having dejure credentials for a similar service. Rather than rejecting such requests, it may be preferable to offer a credential with a similar authorization. For example, Alice likes the service that Bob's mobile phone credential provides, takes a copy, and requests a similar credential.

Supporting imprecision for retrieval systems has been considered in the literature on similarity-based retrieval for case-based reasoning systems [10]. For the purposes of this paper we will use a very simple notion of similarity. Define a similarity measure over permissions: $p \sim q$ returns a value indicating the degree of similarity between permissions p and q, ranging from 0 (not similar) to 1 (identical). When K_B requests permission p from K_A, then K_A may issue $\{\!|\, K_B, q \,|\!\}_{K_A}$ where q has the highest $p \sim q$ value such that $(\!|\, K_B, q \,|\!)_{K_A}$.

Example 3 Consider Example 1. An extended service permission (t, x) specifies that the quality of service for $t \in \{\mathsf{priv}, \mathsf{bus}\}$ is value $x \in [1, \mathsf{max}]$. The permission ordering from Figure 2(b) is extended such that $(t, x) \leq (u, y)$ iff $t = u$ and $x \leq y$. Suppose that the following dejure delegation rules are specified.

$$\frac{}{(\!|\, K_R, (\mathsf{bus}, x) \,|\!)_{K_C}} \quad [x \in \{5, 3\}]$$

$$\frac{}{(\!|\, K_R, (\mathsf{priv}, x) \,|\!)_{K_C}, (\!|\, K_I, (\mathsf{priv}, x) \,|\!)_{K_C}} \quad [x \in \{3, 1\}]$$

Reduction satisfies Bob's request for a credential with permission $(\mathsf{bus}, 3)$. Bob must phrase his request exactly: a request for precisely $(\mathsf{bus}, 1)$ will be rejected, even though a similar and possibly acceptable credential could be written, that is, $\{\!|\, K_B, (\mathsf{priv}, 1) \,|\!\}_{K_C}$.

Table 1 defines a possible similarity measure. Similarity is primarily based on the difference between the quality values. However, a weighting is used to

$\sim$	ok	(bus, y)	(priv, y)
ok	1	0	0
(bus, x)	0	$\frac{\text{max}-\lvert x-y\rvert}{\text{max}}$	$0.8 \times \frac{\text{max}-\lvert x-y\rvert}{\text{max}}$
(priv, x)	0	$0.8 \times \frac{\text{max}-\lvert x-y\rvert}{\text{max}}$	$\frac{\text{max}-\lvert x-y\rvert}{\text{max}}$

Table 1. Permission Similarity Measure

give preference to the same service when the difference is small. Suppose that the maximum quality value is 5. Bob's request for permission (bus, 1), retrieves (priv, 1) with the highest similarity measure (0.8) when compared with (bus, 1). A request for (bus, 2) retrieves (bus, 3) (measure 0.8) as the better match since both (priv, 1) and (priv, 3) gives measure 0.64 when compared with (bus, 2). $\triangle$

In the previous example the similarity measure is applied only to the reduction result. How the measure might be used during a reduction requires further study: direct application may have a cascading effect along delegation chains making the measure ineffective.

5 Conclusion

This paper considers how a trust management system might be used by an application to determine whether it is safe to do defacto or dejure delegation. When making a decision, similarity-based retrieval techniques provide for a degree of imprecision that may be useful when it is not clear beforehand precisely what form the delegation should take.

Delegated trust is treated as transitive, as exemplified by the reduction rule above. While it is straightforward to extend it to include a 'no-delegate' credential flag, it remains quite a rigid notion of trust. It would be interesting to consider how other notions and/or measures of trust might be applied to authorization credentials. For example, a PGP certificate [11] provides a degree of trust field, indicating the extent to which the signer trusts a key to vouch for the identity of others. Applying this notion to an authorization credential may require reduction to find multiple independent delegation chains from K_A to K_B before writing a new credential $\{\!|\, K_B, p \,|\!\}_{K_A}$. A variety of other measures for trust have been proposed [2, 9, 6]. One avenue worth pursuing would be to use the insurance certificate logic in [6] to form the basis of dejure rules that specify what kind, and how much, insurance is necessary before delegation should be done.

Acknowledgment This work was supported in part by a grant from Enterprise Ireland National Software Directorate. The example on promotion and delegation in WebCom (Example 2) grew from useful discussions with colleagues John Morrison and Thomas Quillinan.

References

1. T. Aura and C Ellison. Privacy and accountability in certificate systems. Technical Report HUT-TCS-A61, Helsinki University of Technology, Laboratory for Theoretical Computer Science, 2000.
2. T. Beth, M. Borcherding, and H. Klein. Valuation of trust in open networks. In *European Symposium on Research in Computer Security*, LNCS 875. Springer Verlag, 1994.
3. M Blaze et al. The keynote trust-management system version 2. September 1999. Internet Request For Comments 2704.
4. C Ellison et al. Spki certificate theory. September 1999. Internet Request for Comments: 2693.
5. S.N. Foley, T.B. Quillinan, J.P. Morrison, D.A. Power, and J.J. Kennedy. Exploiting KeyNote in WebCom: Architecture neutral glue for trust management. In *Proceedings of The Fifth Nordic Workshop on Secure IT Systems*, Reykjavik, Iceland, Oct 2001.
6. J.K. Millen and R.N. Wright. Reasoning about trust and insurance in a public key infrastructure. In *Computer Security Foundations Workshop*. IEEE Computer Society Press, 2000.
7. John P. Morrison and David A. Power. Master promotion and client redirection in the webcom system. In *PDPTA, Las Vegas USA*, 2000.
8. J.P. Morrison, D.A. Power, and J.J. Kennedy. A Condensed Graphs Engine to Drive Metacomputing. Proceedings of the international conference on parallel and distributed processing techniques and applications (PDPTA '99), Las Vagas, Nevada, June 28 - July1, 1999.
9. M.K. Reiter and S.G. Stubblebine. Authentication metric analysis and design. *ACM Transactions on Information and System Security*, 2(2):138–158, 1999.
10. I Watson and F Marir. Case based reasoning review. *The Knowledge Engineering Review*, 9(4), 1994.
11. P. Zimmermann. *The Official PGP Users Guide*. MIT Press, 1995.