Risk Assessment in Distributed Authorization

Peter Chapin
Department of Computer Science
University of Vermont
pchapin@cs.uvm.edu

Christian Skalka
Department of Computer Science
University of Vermont
skalka@cs.uvm.edu

X. Sean Wang
Department of Computer Science
University of Vermont
xywang@cs.uvm.edu

ABSTRACT

Distributed authorization takes into account several elements, including certificates that may be provided by non-local actors. While most trust management systems treat all assertions as equally valid up to certificate authentication, realistic considerations may associate risk with some of these elements; some actors may be less trusted than others, some elements may be more computationally expensive to obtain, and so forth. Furthermore, practical online authorization may require certain levels of risk to be tolerated. In this paper, we introduce a trust management logic that incorporates formal risk assessment. This formalization allows risk levels to be associated with authorization elements, and promotes development of a distributed authorization algorithm allowing tolerable levels of risk to be precisely specified and rigorously enforced.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*

General Terms

Security, Languages, Theory

Keywords

Distributed Authorization, Trust Management Logic

1. INTRODUCTION

Trust management systems provide a formal means to specify and enforce distributed authorization policies. From its origins in BAN [7] and ABLP logic [1], research progress in this field now comprises systems such as SDSI/SPKI [16, 9] and RT [13]. The expressiveness and rigor of these systems has become increasingly important to security in modern distributed computing infrastructures, as web-based interactions continue to evolve in popularity and complexity.

Authorization in trust management usually takes into account several facts and assertions, including certificates provided by non-local, untrusted actors. Although e.g. cryptographic techniques provide certain measures of confidence in this setting, not all components of authorization can realistically be used with the same level of confidence; the Pretty Good Privacy (PGP) framework acknowledges this, by including a notion of trustworthiness of certificates. Furthermore, efficient online authorization decisions often require a weakening of ideal security, since the latter may be prohibitively expensive. This weakening may involve the acceptance of assertions that would otherwise be verified, in case lowered confidence levels are more tolerable than the danger of intractability. Thus, many practical distributed authorization decisions include elements of *risk* associated with authorization components, where risk could be associated with trust, or computational cost, or any other practical consideration making some facts more or less risky than others.

A rigorous assessment of authorization should accurately assess risk, but risk in trust management is usually an informal consideration. In this paper, we introduce a trust management logic, called RT^R, that formally incorporates formal risk assessment. The system is a variant of RT [13], and includes an abstract definition of risk, a means to associate risk with individual assertions, and a semantics that assesses risk of authorization by combining the risk of assertions used in authorization decisions. This formalization promotes development of a distributed authorization algorithm allowing tolerable levels of risk to be precisely specified and rigorously enforced.

1.1 Paper Outline

The remainder of the paper is organized as follows. In Sect. 2, an overview of the RT_0 system is given for background. In Sect. 3, we define the syntax and set-theoretic semantics of RT^R, an authorization logic with risk assessment. In Sect. 4, we give a graph-theoretic interpretation of RT^R that is equivalent to the set-theoretic semantics, and show that so-called credential graphs can be automatically reconstructed by a distributed chain discovery algorithm, as an implementation of distributed authorization. In Sect. 5, we discuss some interesting applications motivating the development of RT^R, and we conclude with a summary of the paper and remarks on related work in Sect. 6.

2. OVERVIEW OF RT

Rather than defining a new trust management logic for a

FMSE'05, November 11, 2005, Fairfax, Virginia, USA.
Copyright 2005 ACM 1-59593-231-3/05/0011 ...$5.00.

formalization of risk, we take advantage of the existing RT system [13]. This system combines the strengths of role-based access control with an expressive trust management logic, and enjoys a variety of existing implementation techniques [15]. We believe these features make RT one of the most advanced trust management systems, and an appealing setting for the development of formal risk assessment. The RT role-based trust management system is actually a collection of trust management logics, all of which are variations on a basic logic called RT_0 [13]. Variations include separation of duties and delegation. In this same spirit, we propose a variation on RT_0 to incorporate a formalization of risk assessment, so we briefly review RT_0 here to provide necessary background.

In RT_0, individual actors, or principals, are called *Entities* and are defined by public keys. We let A, B, C, D, E range over entities. Each entity A can create an arbitrary number of *Roles* in a namespace local to the entity, denoted $A.r$. The *RoleExpressions* of RT^R, denoted f, are either entities or roles or constructed from other role expressions by *linking* and *intersection*, as described below. To define a role an entity issues credentials that specify the role's membership. Some of these credentials may be a part of private policy; others may be signed by the issuer and made publically available. The overall membership of a role is taken as the memberships specified by all the defining credentials.

RT_0 provides four credential forms:

1. $A.r \longleftarrow E$

   This form asserts that entity E is a member of role $A.r$.

2. $A.r \longleftarrow B.s$

   This form asserts that all members of role $B.s$ are members of role $A.r$. Credentials of this form can be used to delegate control over the membership of a role to another entity.

3. $A.r \longleftarrow B.s.t$

   This form asserts that for each member E of $B.s$, all members of role $E.t$ are members of role $A.r$. Credentials of this form can be used to delegate control over the membership of a role to all entities that have the attribute represented by $B.s$. The expression $B.s.t$ is called a *linked role.*

4. $A.r \longleftarrow f_1 \cap \cdots \cap f_n$

   This form asserts that each entity that is a member of all role expression forms $f_1, \ldots, f_n$ is also a member of role $A.r$. The expression $f_1 \cap \cdots \cap f_n$ is called an *intersection role.*

Authorization is then cast as a role membership decision: an access target is represented as some role expression f, and authorization for that target for some entity A is equivalent to determining whether A is a member of f. In such a decision, we call f the *governing role.* Authorization always assumes some given finite set of credentials, denoted $\mathcal{C}$. We use $\mathit{Entities}(\mathcal{C})$ to represent the entities used in a particular set of credentials $\mathcal{C}$, and similarly $\mathit{RoleNames}(\mathcal{C})$, $\mathit{Roles}(\mathcal{C})$, etc.

2.1 Example

Suppose a hotel H offers a room discount to certain preferred customers, who are members of $H.\mathit{preferred}$. The policy of H is to grant a discount to all of its preferred customers in $H.\mathit{preferred}$ as well as to members of certain organizations. H defines a role $H.\mathit{orgs}$ that contains the public keys of these organizations. Into that role H places, for example, the key of the AAA, the American Auto Association. These credentials are summarized as follows:

$$H.\mathit{discount} \longleftarrow H.\mathit{preferred}$$

$$H.\mathit{discount} \longleftarrow H.\mathit{orgs}.\mathit{members} \qquad H.\mathit{orgs} \longleftarrow AAA$$

Now imagine that at a later time a special marketing plan is created to encourage travellers to stay at H. A decision is made that all members of the AAA are automatically preferred customers and thus the credential $H.\mathit{preferred} \longleftarrow AAA.\mathit{members}$ is added to the policy.

Finally suppose that Mary is a member of the AAA. She has a credential, $AAA.\mathit{members} \longleftarrow M$, attesting to that fact. By presenting this credential to H's web service Mary can prove in two distinct ways that she is authorized to receive the discount. On one hand she is a member of an organization in $H.\mathit{orgs}$. On the other hand she is, indirectly, a preferred customer of H. Certain practical considerations may motivate H's decision about which "proof" to use. As we will see in Sect. 4, specified risk thresholds in RT^R can steer authorization in the right direction.

3. THE SYSTEM RT^R

The system RT^R is RT_0 extended with a formal definition of risk assessments. In this section we define the syntax and semantics of RT^R, and give some examples of risk-assessed authorization decisions in RT^R. As for RT in [15], we define a set theoretic semantics for RT^R, since this allows an easy correspondence with the graph theoretic characterization of RT^R for distributed chain discovery given in the next section. While a constraint datalog semantics for RT^R– similar to the datalog semantics of RT in [12]– is an interesting possibility, it is beyond the scope of this paper.

3.1 Syntax and Semantics

The system RT^R is defined as a framework, parameterized by a *risk ordering*, which is required to be a complete lattice $(\mathcal{K}, \preccurlyeq)$. We let κ and K range over elements and subsets of $\mathcal{K}$ respectively, and let $\top$ and $\bot$ denote top and bottom. Any instantiation of RT^R is expected to supply a risk ordering with $\preccurlyeq$ decidable, and must also supply an associative, commutative, monotonic *risk aggregation* operation $\oplus$. The relation $\preccurlyeq$ allows risks to be compared, and "greater" and "lesser" risks assessed, while $\oplus$ allows risks to be combined when authorization decisions involve multiple risks. Examples of particular risk orderings are given in Sect. 3.2 and Sect. 5, below.

The basis of risk assessment is the association of risk with individual credentials, since credentials are the fundamental assertions used in authorization decisions. Thus, credentials in RT^R are of the following form:

$$A.r \xleftarrow{\kappa} f$$

where κ is the risk associated with the credential. We leave unspecified the precise mechanism of risk association, though

$$\begin{aligned}
\mathrm{bounds}[\mathrm{rmem}](A.r) &= \bigcup_{A.r \xleftarrow{\kappa} e \in \mathcal{C}} \mathrm{expr}[\mathrm{rmem}](e) \oplus \kappa \\
\\
\mathrm{expr}[\mathrm{rmem}](B) &= \{(B, \bot)\} \\
\mathrm{expr}[\mathrm{rmem}](A.r) &= \mathrm{rmem}(A.r) \\
\mathrm{expr}[\mathrm{rmem}](A.r_1.r_2) &= \bigcup_{(B,\kappa) \in \mathrm{rmem}(A.r_1)} \mathrm{rmem}(B.r_2) \oplus \kappa \\
\mathrm{expr}[\mathrm{rmem}](f_1 \cap \cdots \cap f_n) &= \bigoplus_{1 \le i \le n} \mathrm{expr}[\mathrm{rmem}](f_i)
\end{aligned}$$

Figure 1: RT^R semantic functions

in many cases it is likely that the authorizing agent will automatically assign risk to credentials. In essence, the aggregation of risks associated with credentials used in some authorization decision constitutes the risk of that decision.

Formally, the semantics of RT^R associates risk κ with the membership of entities B in roles $A.r$. Thus, the meaning of roles $A.r$ are finite sets of pairs of the form (B, κ), called *RiskAssessments*; we let R range over such sets. For any $\mathcal{A} \subseteq$ *Entities*, *RiskAssessment*$(\mathcal{A})$ denotes the set of risk assessments R such that $(A, \kappa) \in R$ implies $A \in \mathcal{A}$. Note that any R may associate more than one risk with any entity, i.e. there may exist $(A, \kappa_1), (A, \kappa_2) \in R$ such that $\kappa_1 \neq \kappa_2$. This reflects the possibility of more that one path to role membership, each associated with incomparable risk. Taking the glb of incomparable risks in risk assessments is unsound, since the glb will assess a lesser risk of membership than is in fact possible to obtain through any path.

However, if a risk assessment associates two distinct but comparable risks with a given role membership, the lesser of the two can be taken as representative; in general, risk assessments can be taken as a set of lower bound constraints on risk in authorization. Thus, we define equivalence on risk assessments as follows:

$$R \cup \{(A, \kappa_1), (A, \kappa_2)\} = R \cup \{(A, \kappa_1)\} \qquad \text{where } \kappa_1 \preccurlyeq \kappa_2$$

We call *canonical* those risk assessments R such that there exist no $(A, \kappa_1), (A, \kappa_2) \in R$ where $\kappa_1 \preccurlyeq \kappa_2$, and observe that any equivalence class of risk assessments has a unique canonical form. Furthermore, the canonical representation of any assessment R, denoted $\hat{R}$, is decidable since assessments are finite and $\preccurlyeq$ is decidable. We extend the ordering $\preccurlyeq$ to risk assessments as follows:

$$R_1 \preccurlyeq R_2 \iff \forall (A, \kappa_1) \in \hat{R}_1 . \exists \kappa_2 . (A, \kappa_2) \in \hat{R}_2 \wedge \kappa_1 \preccurlyeq \kappa_2$$

The relation is clearly decidable. We also observe that it is is a partial order:

Corollary 3.1. *The relation $\preccurlyeq$ on risk assessments is a partial order.*

Hereafter we restrict our consideration to canonical risk assessments without loss of generality. We immediately observe the following, which will be useful in the development of a formal semantics for RT^R:

Lemma 1. *For all finite $\mathcal{A} \subset$ Entities, the poset:*

$$(\mathit{RiskAssessment}(\mathcal{A}), \preccurlyeq)$$

is a complete lattice.

Proof. Given $\mathcal{R} \subseteq$ *RiskAssessment*$(\mathcal{A})$. For each $A \in \mathcal{A}$, let $K_A = \{\kappa \mid \exists R \in \mathcal{R}.(A, \kappa) \in R\}$, and let κ_A be the lub of K_A, which must exist since we require risk orderings to be complete lattices. Let $R_{\mathcal{R}} = \{(A, \kappa_A) \mid A \in \mathcal{A}\}$. Clearly $R_{\mathcal{R}}$ is an element of *RiskAssessment*$(\mathcal{A})$, and is a lub of $\mathcal{R}$. The existence of a glb for $\mathcal{R}$ follows dually. □

A notion of aggregation of risk assessments is useful to define:

$$\begin{aligned}
R \oplus \kappa &\triangleq \{(A, \kappa' \oplus \kappa) \mid (A, \kappa') \in R\} \\
R_1 \oplus R_2 &\triangleq \{(A, \kappa_1 \oplus \kappa_2) \mid (A, \kappa_1), (A, \kappa_2) \in R_1 \times R_2\}
\end{aligned}$$

We assert monotonicity of this operation:

Corollary 3.2. *The operation $\oplus$ on risk assessments is monotonic.*

As we will see below, solutions to sets of credentials are functions of type:

$$\mathit{Role} \rightarrow \mathit{RiskAssessment}$$

Letting f and g be functions of this type, we define:

$$f \preccurlyeq g \iff f(A.r) \preccurlyeq g(A.r) \text{ for all roles } A.r$$

Now, we can define the semantics of RT^R, by extending the semantics of RT$_0$ in [15] to assess risk:

Definition 1 (Semantics of RT^R). *Given a set $\mathcal{C}$ of RT^R credentials, the semantics $\mathcal{S}_{\mathcal{C}}$ of $\mathcal{C}$ is a function mapping role expressions to risk assessments. In particular, $\mathcal{S}_{\mathcal{C}}$ is the least function* rmem : *Role $\rightarrow$ RiskAssessment (ordered by $\preccurlyeq$ as above) such that* bounds[rmem] $\preccurlyeq$ rmem, *where* bounds[rmem] *and the auxiliary function* expr[rmem], *mapping role expressions to risk assessments, are defined as in Fig. 1.*

Given any $\mathcal{C}$, we can construct $\mathcal{S}_{\mathcal{C}}$ by a least fixpoint argument, showing that any set of credentials has a solution. The technique follows [15]. The solution is constructed as the limit of the sequence $\{\mathrm{rmem}_i\}_{i \in \mathbb{N}}$, where rmem_i : *Roles*$(\mathcal{C}) \rightarrow$ *RiskAssessment*$(\mathit{Entities}(\mathcal{C}))$ for every i. The sequence is defined inductively by taking $\mathrm{rmem}_0(A.r) = \varnothing$ for every role $A.r$, and letting:

$$\mathrm{rmem}_{i+1}(A.r) = \mathrm{bounds}[\mathrm{rmem}_i](A.r)$$

for every $A.r$. The function relating the values in $\{\mathrm{rmem}_i\}_{i \in \mathbb{N}}$ is monotonic, since $\cup$ and $\oplus$ are monotonic, the latter by definition and Corollary 3.2. Further, the pointwise ordering of functions rmem_i under $\preccurlyeq$ forms a complete lattice, by Lemma 1. Therefore, a least fixpoint of the sequence $\{\mathrm{rmem}_i\}_{i \in \mathbb{N}}$ exists. Let rmem_ω be this fixpoint, and define:

$$\begin{array}{rcll} \mathcal{S}_\mathcal{C}(A.r) & = & \mathrm{rmem}_\omega(A.r) & A.r \in \mathit{Roles}(\mathcal{C}) \\ \mathcal{S}_\mathcal{C}(A.r) & = & \varnothing & A.r \notin \mathit{Roles}(\mathcal{C}) \end{array}$$

It is easily shown that $\mathcal{S}_\mathcal{C}(A.r)$ so defined is a least solution to $\mathcal{C}$ as specified in Definition 1.

3.2 Examples

We now give some examples of risk assessments for authorizations in two different risk models, illustrating applications of the system. Other more complex examples are discussed in Sect. 5.

3.2.1 Bound-of-Risks

In [8], an information flow security model is presented where all static data is assigned to a security class. Security classifications of variables are then assigned based on the combination of security classes of data flowing into those variables, as determined by an abstract program interpretation. Security classes are identified by elements in a complete lattice, where "class-combination" is defined as the lub of combined classes.

We propose that an adaptation of this model is useful in the context of authorization risk assessment. We do not propose an abstract interpretation of authorization, incorporating some form of "may-analysis", but rather a purely dynamic authorization and risk assessment model, so in this sense we differ from the model proposed in [8]. Nevertheless, we may adopt the use of least upper bounds as a "class-combination" mechanism– in our terminology, "risk aggregation"– that assesses the risk of any authorization decision as the least upper bound of risks associated with all credentials used in the decision.

Consider a risk ordering where three classifications $\mathcal{K} = \{\mathit{low}, \mathit{medium}, \mathit{high}\}$ are defined, and the following relations are imposed:

$$\mathit{low} \preccurlyeq \mathit{medium} \preccurlyeq \mathit{high}$$

and $\oplus$ is taken to be the lub operator. Imagine also that an online vendor called *Store* maintains a purchasing policy whereby representatives of the *Acme* corporation have *buyer* power only if they are both employees and official purchasers. Since this policy is maintained locally, it is associated with a *low* risk of usage, hence *Store* could specify:

$$\mathit{Store.buyer} \overset{\mathit{low}}{\longleftarrow} \mathit{Acme.purchaser} \cap \mathit{Acme.employee}$$

Imagine further that *Ed* attempts to make a purchase from *Store*, providing certificates claiming *employee* and *purchaser* status. However, if we assume that these certificates can possibly be faked, or that role membership within the *Acme* corporation has a volatile status, higher risk can be assigned to these certificates:

$$\mathit{Acme.employee} \overset{\mathit{medium}}{\longleftarrow} \mathit{Ed} \qquad \mathit{Acme.purchaser} \overset{\mathit{high}}{\longleftarrow} \mathit{Ed}$$

We also assume that a less risky path of establishing *Ed*'s membership in the *Acme.purchaser* role is through a *manager* certificate obtained directly from *Personnel*, and via *Acme*'s own policy specifying *purchaser* power for all *manager*s:

$$\mathit{Acme.purchaser} \overset{\mathit{low}}{\longleftarrow} \mathit{Personnel.manager}$$

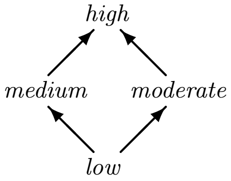
$$\mathit{Personnel.manager} \overset{\mathit{low}}{\longleftarrow} \mathit{Ed}$$

Although using *Ed*'s certificate asserting his membership in the *Acme.purchaser* role will incur a *high* risk, because of the less risky path to this relation, the risk assessment of this set of credentials will find that establishing *Ed*'s membership in the *Store.buyer* role requires a lower bound of *medium* risk. The least solution for all given roles is as follows:

$$\begin{array}{rcl} \mathit{Store.buyer} & : & \{(\mathit{Ed}, \mathit{medium})\} \\ \mathit{Acme.employee} & : & \{(\mathit{Ed}, \mathit{medium})\} \\ \mathit{Acme.purchaser} & : & \{(\mathit{Ed}, \mathit{low})\} \\ \mathit{Personnel.manager} & : & \{(\mathit{Ed}, \mathit{low})\} \end{array}$$

Of course, in certain cases it may be preferable to use the certificate *Ed* provides, instead of going through *Personnel*– if wait times for distributed communication with that node are prohibitively long, for example. However, in this case it should be specified that a *high* level of risk will be tolerated in the credential chain. In Sect. 4 and Sect. 5, we define a technique for credential chain discovery that implements this idea.

Returning to the example, for the purposes of illustration we imagine that the risk ordering is extended with an element *moderate*, that is incomparable with *medium*, inducing the lattice:



We also imagine that *Store* has cached an old certificate, establishing *Ed*'s membership in the *Acme.employee* role with *moderate* risk:

$$\mathit{Acme.employee} \overset{\mathit{moderate}}{\longleftarrow} \mathit{Ed}$$

In this case, since *moderate* and *medium* are incomparable, the risk assessment will reflect that *Ed*'s membership in the *Store.buyer* and *Acme.employee* roles can be established via two paths with incomparable risk:

$$\begin{array}{rcl} \mathit{Store.buyer} & : & \{(\mathit{Ed}, \mathit{medium}), (\mathit{Ed}, \mathit{moderate})\} \\ \mathit{Acme.employee} & : & \{(\mathit{Ed}, \mathit{medium}), (\mathit{Ed}, \mathit{moderate})\} \end{array}$$

Precision and safety in the assessment of minimal risk is not lost by taking the glb of incomparable risk assessments.

3.2.2 Sum-of-Risks

An alternative to the bound-of-risks model is a sum-of-risks model, where credentials are assigned numeric risk values and the total risk for any authorization decision is the sum of all risks associated with the credentials used in the decision. Thus, we take the risk ordering in this model to be the lattice of natural numbers up to ω induced by $\leq$, and we take $\oplus$ to be addition. This model is useful in case risk is considered additive, or in case the number of credentials used in an authorization decision is an element of risk, the more the riskier.

Imagining a similar situation as above, the following risks could be assigned, where 1 is considered “not risky” and 4 is considered “risky”:

$$\mathit{Store.buyer} \xleftarrow{1} \mathit{Acme.purchaser} \cap \mathit{Acme.employee}$$

$$\mathit{Acme.employee} \xleftarrow{3} \mathit{Ed} \qquad \mathit{Acme.purchaser} \xleftarrow{4} \mathit{Ed}$$

$$\mathit{Acme.purchaser} \xleftarrow{2} \mathit{Personnel.manager}$$

$$\mathit{Personnel.manager} \xleftarrow{3} \mathit{Ed}$$

Note that *Ed*’s certificate claiming membership in the role *Acme.purchaser* is still assigned higher risk than both the certificate establishing his *manager* status and the certificate establishing *purchaser* rights for *manager*s. However, the sum-of-risks model will still ascertain that the use of *Ed*’s certificate will be the least risky way to establish his membership in the *Store.buyer* role. The solution of the given credentials will comprise the following risk assessments:

$$\begin{array}{rcl} \mathit{Store.buyer} & : & \{(\mathit{Ed}, \mathit{8})\} \\ \mathit{Acme.employee} & : & \{(\mathit{Ed}, \mathit{3})\} \\ \mathit{Acme.purchaser} & : & \{(\mathit{Ed}, \mathit{4})\} \\ \mathit{Personnel.manager} & : & \{(\mathit{Ed}, \mathit{3})\} \end{array}$$

If a pure count of credentials used in authorization is the basis of risk assessment, this model can be formally obtained in the sum-of-risks model by associating risk 1 with every credential.

4. RT^R CREDENTIAL CHAIN DISCOVERY

In this section we discuss an algorithm for authorization with risk in a distributed environment. Following RT credential chain discovery [15], our technique is to characterize credential sets graph-theoretically, except that our credential graphs are risk-weighted multigraphs, to accommodate risk assessments. Credential graphs are shown to be a full abstraction of solutions as in Definition 1, and the RT^R discovery algorithm is shown to correctly reconstruct credential graphs.

In addition to theoretical correctness, our chain discovery algorithm has two important practical features:

1. The algorithm need not verify a role membership in a risk-optimal fashion, but rather is parameterized by a risk threshold, that is a maximum tolerable risk for role membership verification.

2. The discovery procedure is *directed*, in the sense that it is aborted along search paths whose risk overruns the maximum threshold.

The first feature allows end-users to modulate tolerable levels of risk in authorization. The second feature reaps any efficiency benefits intended by associating risks with credentials, as high risk may be associated with high expense, e.g. if risks are wait times.

4.1 Credential Graphs

We begin by defining an interpretation of credential sets $\mathcal{C}$ as a credential graph. More precisely, sets of credentials are interpreted as a weighted multigraph, where nodes are role expressions, edges are credentials, and weights are risks. Authorization is implemented by determining reachability, via risk weighted paths, where the aggregation of edge risk along the path is the risk of authorization. Reachability is predicated on simple paths, since traversing cycles can only increase risk, and any path with a cycle would otherwise generate an infinite number of risk weighted paths. Allowing the latter would preclude a constructive definition of credential graphs, since chains are distinguished by risk and cycle traversal increases risk monotonically.

Definition 2 (Risk weighted credential chains). *Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a weighted multigraph with nodes $f \in \mathcal{N}$ and edges $f_1 \xrightarrow{\kappa} f_2 \in \mathcal{E}$ weighted by elements κ of a given risk ordering. The pair:*

$$((f_1, \ldots, f_n), \kappa_1 \oplus \cdots \oplus \kappa_{n-1})$$

is a risk weighted path *in $\mathcal{G}$ iff for all $i \in [1..n-1]$, there exists $f_i \xrightarrow{\kappa_i} f_{i+1} \in \mathcal{E}$. A weighted path $((f_1, \ldots, f_n), \kappa)$ is* simple *iff no node is repeated in $(f_1, \ldots, f_n)$. We write $f \xrightarrow{\kappa}\!\!\!\!\rightarrow f'$, pronounced “there exists a credential chain from from f to f' with risk κ”, iff $((f, \ldots, f'), \kappa)$ is a simple risk weighted path. We write $f \xrightarrow{\kappa}\!\!\!\!\rightarrow f' \in \mathcal{G}$ iff $f \xrightarrow{\kappa}\!\!\!\!\rightarrow f'$ holds given $\mathcal{G}$.*

The definition of credential graphs is founded on the definition of risk weighted chains, since edges derived from linked and intersection credentials are supported by them.

Definition 3 (Credential graph). *Given $\mathcal{C}$, its* credential graph *is a weighted multigraph $\mathcal{G}_\mathcal{C} = (\mathcal{N}_\mathcal{C}, \mathcal{E}_\mathcal{C})$, where:*

$$\mathcal{N}_\mathcal{C} = \bigcup_{A.r \xleftarrow{\kappa} e} \{A.r, e\}$$

And $\mathcal{E}_\mathcal{C}$ is the least set of risk-weighted edges satisfying the following closure properties:

1. *If $A.r \xleftarrow{\kappa} e \in \mathcal{C}$ then $e \xrightarrow{\kappa} A.r \in \mathcal{E}_\mathcal{C}$.*

2. *If $B.r_2, A.r_1.r_2 \in \mathcal{N}_\mathcal{C}$ and $B \xrightarrow{\kappa}\!\!\!\!\rightarrow A.r_1$, then $B.r_2 \xrightarrow{\kappa} A.r_1.r_2 \in \mathcal{E}_\mathcal{C}$.*

3. *If $D, f_1 \cap \cdots \cap f_n \in \mathcal{N}_\mathcal{C}$ and for each $i \in [1..n]$ there exists $D \xrightarrow{\kappa_i}\!\!\!\!\rightarrow f_i$, then $D \xrightarrow{\kappa} f_1 \cap \cdots \cap f_n \in \mathcal{E}_\mathcal{C}$, where $\kappa = \kappa_1 \oplus \cdots \oplus \kappa_n$.*

The definition of credential graphs can be made constructive by iterating closure over an initial initial edge set $\mathcal{E}^0_\mathcal{C}$:

$$\mathcal{E}^0_\mathcal{C} = \left\{ A.r \xrightarrow{\kappa} e \mid A.r \xleftarrow{\kappa} e \in \mathcal{C} \right\}$$

In rules (2) and (3), the paths predicating membership in $\mathcal{E}_\mathcal{C}$ are called *support paths*, and the edges are called *derived.* On each iteration, add a new weighted edge according to closure rule (2) or (3). Since $\mathcal{C}$ is finite, and support paths must be simple, the process will reach a fixpoint in a finite number of iterations; this fixpoint is $\mathcal{E}_\mathcal{C}$.

We observe that the characterization of credential sets $\mathcal{C}$ is sound and complete with respect to the set theoretic semantics given in the previous section. These results will form a bridge with the semantics of RT^R for establishing correctness of credential chain discovery. The statement of soundness reflects the fact that while risk assessments of credential sets express minimum risk bounds of role membership, the credential graph does not preclude reachability via paths of higher risk.

Theorem 4.1 (Soundness). *For all $B, A.r$, if $B \xrightarrow{\kappa}\!\!\!\!\!\twoheadrightarrow A.r \in \mathcal{G}_\mathcal{C}$, then $(B, \kappa') \in \mathcal{S}_\mathcal{C}(A.r)$ with $\kappa' \preccurlyeq \kappa$.*

The statement of completeness reflects that any assessed risk is the weight of some related path in the graph:

Theorem 4.2 (Completeness). *For all $A.r$, if $(B, \kappa) \in \mathcal{S}_\mathcal{C}(A.r)$, then $B \xrightarrow{\kappa}\!\!\!\!\!\twoheadrightarrow A.r \in \mathcal{G}_\mathcal{C}$.*

4.2 Backward Chain Discovery Algorithm

As discussed in [15], any role $A.r$ is defined by its credentials. In centralized chain discovery, all credentials are maintained locally by assumption. In distributed chain discovery, some credentials may be stored remotely. *Backwards* chain discovery assumes that the credentials defining a role $A.r$ are obtained through the entity A, so that chains need to be reconstructed "backwards", beginning with the governing role of an authorization decision. We now define a backwards credential chain discovery algorithm checkmem for RT^R, possessing features described at the beginning of Sect. 4. We abstract the details of credential retrieval and risk assignment, other than its "backwards" nature, assuming that remote risk-weighted credentials can always be retrieved on demand (and cached, presumably). While forwards and mixed discovery techniques for RT are also discussed in [15], analogous techniques for RT^R are beyond the scope of this paper. For brevity and clarity in the presentation, we textually describe the algorithm checkmem.

4.2.1 Definition of checkmem

The algorithm checkmem$(A, f, \kappa_{\max})$ reconstructs a proof graph, to check membership of A in role f within a given threshold $\kappa_{\max}$. The algorithm maintains the following mutable datastructures: a solution of type *RoleExpressions* $\rightarrow$ *RiskAssessment*, an association of *solution monitors* with graph nodes, discussed below, and an association of sets of *search risks* with graph nodes. Search risks are the accumulated risks along any discovery path to the given node; it is important to note that search risk associations are different than risk assessments. When a node is first encountered during search, it is added to the queue for future search. No node is added to the queue more than once.

Initially, the solution is a default mapping to the empty risk assessment, and every node is associated with an empty set of solution monitors and search risks. To begin the search, the node f is added to the queue, and associated with the search risk $\bot$.

Nodes are taken from the queue individually for searching, but not indiscriminately; rather, only nodes that have a search risk below the threshold κ are searched. In this way, the algorithm short-circuits discovery along paths that are too risky. Over-threshold nodes are not removed from the queue, since future discovery might find a less risky path to that node. Hence, nodes wait in the queue until they are the next below-threshold node to be searched. The algorithm runs until there are no below-threshold nodes left in the queue, or until a solution for A in f below threshold $\kappa_{\max}$ is found.

Solution monitors propagate solution elements (A, κ) forward along discovered edges, aggregating edge risks as they go; their control flow structure mimics the discovered graph structure. Whenever a monitor notifies a node f to add a solution element (A, κ), if there does not exist $\kappa' \preccurlyeq \kappa$ such that (A, κ') is already in f's solution (in which case we say it is *canonically new*), the solution is added to f, and all of f's solution monitors are applied to the new solution. There are three classes of solution monitors:

1. A *role monitor* for a given role $A.r$ and edge risk κ is a function abstracted on solution elements (B, κ'), that notifies $A.r$ to add $(B, \kappa' \oplus \kappa)$ to its solutions.

2. A *linking monitor* for a given linked role $A.r_1.r_2$ is a function abstracted on solution elements (B, κ), that creates a role monitor for $A.r_1.r_2$ and κ, applies it to each known element of $B.r_2$'s solution, and adds it to $B.r_2$'s solution monitors to propagate solutions yet to be discovered. Finally, given all search risks κ' of $A.r_1.r_2$, $\kappa' \oplus \kappa$ is added to $B.r_2$'s search risks, and $B.r_2$ is added to the queue if it hasn't already been.

3. An *intersection monitor* for a given intersection role $f_1 \cap \cdots \cap f_n$ is a function abstracted on solution elements (B, κ), that creates a role monitor for $f_1 \cap \cdots \cap f_n$ and $\bot$, and applies it to each element (B, κ') in the canonical form of the assessment $R_1 \oplus \cdots \oplus R_n$, where each R_i is the assessment of f_i in the current solution.

Whenever nodes are taken from the queue according to the above described discipline, they are processed depending on their form:

1. To process an entity A, the node A is notified to add $(A, \bot)$ as a solution to itself.

2. To process a role $A.r$, the credentials defining $A.r$ are retrieved. For each such credential $A.r \xleftarrow{\kappa} f$, a role monitor for $A.r$ and κ is created, is applied to all of f's known solutions, and is added to f's solution monitors for propagating solutions still to be discovered. Finally, given all search risks κ' of $A.r$, $\kappa' \oplus \kappa$ is added to f's search risks, and f is added to the queue if it hasn't already been.

3. To process a linked role $A.r_1.r_2$, a linking monitor for $A.r_1.r_2$ is created, is applied to all of $A.r_1$'s known solutions, and is added to $A.r_1$'s solution monitors. Every search risk κ of $A.r_1.r_2$ is added to $A.r_1$'s search risks, and $A.r_1$ is added to the queue if it hasn't already been.

4. To process an intersection role $f_1 \cap \cdots \cap f_n$, an intersection monitor for $f_1 \cap \cdots \cap f_n$ is created, and added to each f_i. Every search risk κ of $f_1 \cap \cdots \cap f_n$ is added to each f_i's search risks, and each f_i is added to the queue if it hasn't already been.

When node processing for an invocation checkmem$(A, f, \kappa_{\max})$ halts, the algorithm returns true iff there exists (A, κ) in the solution of f such that $\kappa \preccurlyeq \kappa_{\max}$.

4.2.2 Properties

Assuming that defining credentials can always be obtained for any role, we assert that checkmem satisfies the following properties, demonstrating that it correctly reconstructs credential graphs. Since credential graphs are full abstractions of the RT^R semantics as discussed in Sect. 4.1, these results demonstrate that checkmem is a correct implementation of RT^R.

THEOREM 4.3 (SOUNDNESS). $\mathrm{checkmem}(A, f, \kappa_{\max})$ *implies* $A \overset{\kappa}{\twoheadrightarrow} f$ *such that* $\kappa \preccurlyeq \kappa_{\max}$.

THEOREM 4.4 (COMPLETENESS). $A \overset{\kappa}{\twoheadrightarrow} f$ *implies that* $\mathrm{checkmem}(A, f, \kappa)$ *holds.*

We also observe that the algorithm terminates, regardless of the given risk threshold. This is because nodes are never visited more than once, and solution monitors will not traverse any graph cycle, and hence are guaranteed to terminate. Solution monitors only propagate canonically new members, but traversal of a cycle necessarily causes a monotonic increase in a solution's risk assessment, hence canonical containment in an existing solution.

THEOREM 4.5 (TERMINATION). *For all* A, f, *and* k, $\mathrm{checkmem}(A, f, \kappa)$ *terminates.*

4.2.3 Discussion

There are two particular instances where the definition of checkmem could be enhanced, for more eager short-circuiting of chain discovery in case risk thresholds are exceeded along discovery paths. First, observe that credentials are retrieved before being checked to see if their risks will force the discovery threshold to be exceeded. However, risks such as expected wait time suggest that it is more practical for credentials to be retrieved after ensuring they won't overrun the threshold. A number of minor variations on checkmem can be imagined that will address this.

A more interesting enhancement is relevant to the propagation of search risks along discovery paths leading from intersection nodes. Observe that from any intersection role $f_1 \cap \cdots \cap f_n$, the search risks of $f_1 \cap \cdots \cap f_n$ are propagated to each f_i. However, this could be a under-approximation of search risks for any given f_i. For example, suppose that A is being checked for authorization and (A, κ) is known to be the only possible assessment of A in f_1's solution. When checking f_n, the search risks of f_n inherited from $f_1 \cap \cdots \cap f_n$ could be aggregated with κ, since κ is certain to be a component risk of any authorization supported by discovery from f_n. A generalization of this idea is beyond the scope of this paper, but is an interesting topic for future work.

5. APPLICATIONS

In this section we discuss interesting applications of RT^R. Details of these applications are avenues for future work; here we describe how RT^R could be used to support trust management systems that incorporate notions of risk.

5.1 Trust but Verify

The Trust but Verify (TbV) framework [17] provides a setting for distributed trust management that takes into account a notion of *trust* for online authorization decisions, backed up by offline *verification*. Many realistic authorization decisions require "softening" of security in the online phase; this amounts to trusting the validity of certain assertions in this phase, that would otherwise be too expensive to verify. However, online trust should be specified so that sound offline verification is well-defined, providing formal certainty that offline verification supports online trust.

Any authorization decision in the TbV framework is abstractly specified as derivability of a target privilege **priv** given a security context s, written $s \vdash \mathbf{priv}$. Any instance of the TbV framework comprises a *trust transformation*, that formalizes the definition of trust in terms of a function, mapping initial security contexts s to contexts $[\![s]\!]$, that contain assertions that are trusted solely for efficient online verification. Furthermore, the trust transformation should be reversible, via an audit technique that is required to reconstruct a security context that is at least as strong as the pre-image s of any trust-transformed security context $[\![s]\!]$. The audit technique is the implementation of offline verification. In [17], the TbV framework is developed using ABLP logic [1]. However, the RT framework is a more modern trust management system, with a variety of implementation techniques and variations [15]. The RT^R variation offers a unique dimension of support for TbV, since trust can be encoded using definitions of risk in RT^R.

The TbV framework is characterized by three conditions, that we recount here. We show how RT^R can be used to instantiate the framework in a system that satisfies these conditions. The first condition requires that authorization decisions are decidable:

CONDITION 1. *Let* s *be an authorization context; then validity of* $s \vdash \mathbf{priv}$ *is decidable.*

In RT^R, authorization decisions are implemented as role membership decisions with an assessed risk, and security contexts are sets of credentials $\mathcal{C}$. That is, if the role $A.r$ represents a target privilege and B is a privilege requester, then authorization amounts to discovery of $B \overset{\kappa}{\twoheadrightarrow} A.r \in \mathcal{G}_{\mathcal{C}}$, where κ must be within a specified threshold.

The second condition specifies that auditing reverses trust transformation, though since trust transformations can be many-to-one, the context returned by auditing need not be the exact preimage of trust transformation:

CONDITION 2. *Let* s *be a trusted context; then success of* $\mathit{audit}(s)$ *entails* $[\![\mathit{audit}(s)]\!] = s$.

The last condition sufficiently strengthens the requirements of auditing to formally establish that any auditing is a sound verification of trust injected by the trust transformation:

CONDITION 3. *Let* s *be a trusted context; then if* $\mathit{audit}(s)$ *succeeds, for all* **priv** *it is the case that* $\mathit{audit}([\![s]\!]) \vdash \mathbf{priv}$ *implies* $s \vdash \mathbf{priv}$.

The condition requires that auditing of a trust-transformed context must reconstruct a context that is at least as strong as the initial context, pre-trust-transformation. In RT^R, since authorization contexts are credentials $\mathcal{C}$, and the authorization decision includes a risk threshold, trust transformation may be implemented as the extension of a credential base $\mathcal{C}$ with additional "riskier" credentials, along with an increase in the tolerable risk threshold in chain discovery. Returning to the example in Sect. 3.2, the initial authorization decision could be to determine $\mathit{Ed} \overset{\kappa}{\twoheadrightarrow} \mathit{Store.buyer}$, where $\kappa \preccurlyeq \mathit{medium}$. An online trust transformation could add Ed's credential $\mathit{Acme.purchaser} \xleftarrow{\mathit{high}} \mathit{Ed}$ to the credential base, and tolerate $\kappa \preccurlyeq \mathit{high}$. Auditing in this case would entail removing Ed's certificate from the credential base, and restoring the risk threshold $\kappa \preccurlyeq \mathit{medium}$. In fact, just raising the risk threshold would be sufficient, since raising the risk threshold would eliminate the possibility of using Ed's certificate in the proof of his membership in $\mathit{Store.buyer}$, and in general trust transformation and auditing could be implemented in RT^R purely by modulation of risk thresholds in chain discovery.

5.2 Cost/Benefit Analysis

Risk in RT^R is defined in an abstract manner. Although the examples in this paper have used atomic risk values, it is possible to define a risk ordering on compound risk values. For example, suppose both levels of "trustability" and expected wait times for retrieval of specific credentials are considered components of risk. The set $\mathcal{K}$ could then contain elements of the form (κ, t), where $\kappa \in \{low, medium, high\}$ as in Sect. 3.2 and t is a wait time represented as an integer, and:

$$(\kappa_1, t_1) \preccurlyeq (\kappa_2, t_2) \iff \kappa_1 \preccurlyeq \kappa_2 \wedge t_1 \leq t_2$$

reflecting that lower wait times, as well as higher confidence in validity, define lower risk. Maximum risk in chain discovery would then specify both a tolerable level of trust, and a tolerable wait time for any particular credential.

This suggests an *interactive* procedure for chain discovery, where the costs of raising the level of one component of risk could be balanced against benefits in another risk dimension. In the above scenario, if chain discovery in some instance fails given a threshold (κ, t), chain discovery could be re-run with a higher threshold, but notice there is a choice of which element(s) of risk to raise. The cost of raising κ can then be balanced against the benefits in the time dimension, by re-running chain discovery with the threshold (κ', t), where $\kappa \preccurlyeq \kappa'$. The opposite is also clearly the case. This cost/benefit analysis would be further enhanced by optimizing chain discovery. The backward chain discovery algorithm presented in this paper ensures that risks are kept below a certain threshold, but does not attempt to optimize risk. By extending chain discovery with optimization techniques, in the presence of compound risk, benefit dimensions could be optimized within a fixed cost dimension. For example, optimal wait times could be sought given a *high* level of trust risk. Development of optimizing algorithms is a topic for future work.

6. CONCLUSION

We now conclude with comments on related work and a short summary of the paper.

6.1 Related Work

Many trust management systems have been developed by previous authors. In such a system resource owners write policy statements using a suitable policy language that describes the attributes of authorized users. When a request is made, the requesting entity provides signed credentials that prove the requester complies with the policy. Proofs are constructed automatically, and implement a formal semantics. Previous systems include BAN [7] and ABLP logic [1], PolicyMaker [6], KeyNote [5], SDSI/SPKI [16], [9], and RT [14], [15], [13], to name a few. However, our focus is not on trust management, but trust management extended with risk assessment.

Proof carrying carrying authorization (PCA) [4, 3] is a framework for specifying and enforcing webpage access policies. It is based on ABLP logic, but includes primitives for detecting timestamp expiration. While this capability reflects some sense of risk assessment, it is not as general as the notion of risk expressed in our system.

In [2], semantics for a number of RT variants are obtained via embedding in constraint datalog. An implementation of "confidence levels", similar to our notion of risk assessment, is suggested via the use of constraints, though not developed in detail. While it is possible that many interesting risk assessment schemes can be defined using RT_1 or RT_2, we believe that defining a new RT variant to explicitly capture the notion of risk assessments is appealing in various respects. In particular, we are able to define risk in a general manner, and isolate issues related to online authorization with components of risk.

Dealing with trustworthiness in distributed systems has been an active research area (see, e.g., [10]). In [11], an algebra is provided for reasoning about trust in certificate chains. Our notion of risk is related to the notion of trust, and some relevant operators of [11] may be directly incorporated into our framework. Comparative expressiveness of risk and trust operators is an interesting research topic, but is beyond the scope of this paper.

6.2 Summary

In this paper we have defined RT^R, a role-based trust management framework with formal risk assessment. This system is a variation on RT [13], and includes the capability to associate credentials with risk, and to assess risk levels of authorization as the aggregated risks of authorization components. Risks are defined in an abstract manner, under the requirement that the set of risks be a complete lattice, with a monotonic aggregation operator. A formal semantics has been given, that associates role membership with risk levels. An algorithm has also been defined for implementation of this semantics, providing an automatic risk assessed authorization procedure. The algorithm is specialized for functionality in a distributed environment, and can be parameterized by risk thresholds, specifying a maximum tolerable risk for authorization. The algorithm is directed, to avoid proof paths whose aggregate risks exceed the given threshold, hence to risk as little as possible during the course of authorization.

7. REFERENCES

[1] M. Abadi, M. Burrows, B. Lampson, and G. Plotkin. A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems*, 15(4):706–734, 1993.

[2] Scot Anderson. Constraint datalog in trust management. Master's thesis, University of Nebraska, 2003.

[3] Andrew W. Appel and Edward W. Felten. Proof-carrying authentication. In G. Tsudik, editor, *Proceedings of the 6th Conference on Computer and Communications Security*, Singapore, November 1999. ACM Press.

[4] Lujo Bauer. *Access Control for the Web via Proof-carrying Authorization.* PhD thesis, Princeton University, 2003.

[5] Matt Blaze, Joan Feigenbaum, John Ioannidis, and Angelos D. Keromytis. *RFC-2704: The KeyNote Trust-Management System Version 2.* IETF, September 1999.

[6] Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized trust management. Technical Report 96-17, DIMACS, June 28 1996.

[7] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[8] D. Denning. A lattice model of secure information flow. In *Communications of the ACM*, pages 236–243. ACM, May 1976.

[9] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen. SPKI certificate theory. RFC 2693, Sept. 1999.

[10] Tyrone Grandison and Morris Sloman. A survey of trust in internet applications. *IEEE Communications Surveys & Tutorials*, 4th Quarter, 2000.

[11] Audun Jøsang. An algebra for assessing trust in certification chains. In J. Kochmar, editor, *Proceedings of the Network and Distributed Systems Security Symposium (NDSS'99)*. The Internet Society, 1999.

[12] Ninghui Li and John C. Mitchell. Datalog with constraints: A foundation for trust management languages. In *Proceedings of the Fifth International Symposium on Practical Aspects of Declarative Languages*, January 2003.

[13] Ninghui Li and John C. Mitchell. Rt: A role-based trust-management framework. In *Proceedings of the Third DARPA Information Survivability Conference and Exposition*, pages 201–212. IEEE Computer Society Press, April 2003.

[14] Ninghui Li, John C. Mitchell, and William H. Winsborough. Design of a role-based trust-management framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 114–130. IEEE Computer Society Press, May 2002.

[15] Ninghui Li, William H. Winsborough, and John C. Mitchell. Distributed chain discovery in trust management. *Journal of Computer Security*, 11(1):35–86, February 2003.

[16] R. Rivest and B. Lampson. SDSI — a simple distributed security infrastructure, 1996. `http://theory.lcs.mit.edu/rivest/sdsi11.html`.

[17] Christian Skalka and X. Sean Wang. Trust but verify: Authorization for web services. In *ACM Workshop on Secure Web Services*, October 2004.