Distributed Knowledge Management for Autonomous Access Control in Computer Networks[1]

Alexandr Seleznyov and Stephen Hailes
Department of Computer Science, University College London
Gower Street, London, WC1E 6BT, UK
Email: {A.Seleznyov, S.Hailes}@cs.ucl.ac.uk

Abstract

This work discusses a conceptual model for automatic acquisition and processing of knowledge about users and devices in computer networks. It employs autonomous agents for distributed knowledge management and integrates them into an autonomic middleware component. Agents grouped into distributed communities act as mediators between users, devices, and network resources. Communicating between each other they make decisions on whether a certain user or device can be given access to a requested resource. In other words, agents in our system perform user/device authentication, authorisation, and maintenance of user credentials.

Keywords: trust, access control, knowledge management, autonomous agents, autonomic computing, middleware, pervasive computing.

1. Introduction

Our society has become increasingly dependent on the rapid access to and processing of information. Increased connectivity not only provides access to a larger number of more varied resources more quickly than ever before, it also gives an access path to resources from virtually anywhere on the network. The number of cross-domain applications is therefore increasing and the domains themselves have become less internally coherent in terms of the levels of trustworthiness one might expect from their users.

Although security mechanisms, such as PKI, already exist and are being used for authentication and access control, they require significant amounts of manual intervention; PKI costs are dominated by the management of policies and procedures [1]. Thus, as the number of network entities grows, the cost of maintaining proper policies and procedures has become unreasonably high, leading to the failure of PKI as a viable technology for general use.

There is growing acceptance that centralised authorisation and authentication architectures will never have the semantic richness adequately to describe the uncertainties in trust that are inevitable in realistic current and near future deployments. Instead, there is a pressing need for distributed generic, autonomous (and hence autonomic), trust management systems in which entities are capable of taking greater responsibility for their own protection. Such approaches are particularly important as enablers for promising technologies such as pervasive computing.

In this paper, we present our considerations for building a resilient, dependable and fault-tolerant access control system that is usable in environments with the degree of heterogeneity likely to be found in pervasive systems. In reality, this can only be achieved by making the system context aware and adaptive, both of which mean a move further from the simple certainties of PKI. In order to ensure that the complexities of such systems do not overwhelm either the user or the application developer, and in order to ensure the greatest degree of application independence, we believe that the correct place for this authorisation system is at the middleware level.

Our Autonomic Distributed Authorisation Middleware system (ADAM) [2] relies on facilitating self-protection through the collection and collation of the data that results from the dynamic web of interactions between network entities. Distributed knowledge acquisition and management is used to authenticate a user, reason about her credibility, and establish an appropriate trust level authorising (or denying) access to requested resources.

The reminder of the paper is organised as follows: In section 2 we review the basis on which our model is constructed. In section 3 we discuss conceptual model of knowledge management used by ADAM. Section 4 discusses different contexts of information, and, finally, section 5 concludes this paper.

[1] This work was supported by BT Labs, Martlesham, as part of the UCL@Adastral.Park MARS project.

2. Premises

There are four logical components that must be taken into account while making an access control decision (Figure 1): sets of principals, resources, actions, and policies.

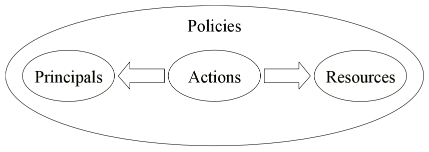

Figure 1. Trust relation's components

The first set defines network entities that can request a resource or service. It consists of users, devices, and applications/processes. The second set represents network resources or services that may be requested. When a user requests access to a resource it is necessary to determine (i) whether the user is who they claim to be (authentication) (ii) whether this user is eligible to access the requested resource and what kind of actions she can perform on it (authorisation). In this paper, we focus on authorisation.

In order to respect the dynamicity of changing context in pervasive environments, it is essential to ensure that authorisation decisions are made at the granularity of requests for individual actions. Thus, in our model, each resource may advertise sets of actions that are permissible at any given point in time to given classes of user. Overall control of this process of trust establishment and maintenance is determined by local policies – the fourth component.

Our approach in ADAM can be differentiated from other trust management systems thus: (i) It is designed to facilitate automatic trust establishment and maintenance between entities situated in different network domains, which provides the flexibility necessary to allow it to function in a pervasive environment. (ii) The model allows each user to have multiple electronic identities. (iii) It only authorises, it does not authenticate; the authentication task is delegated to other components. (iv) Finally, instead of identity-based authorisation, we authorise each action.

3. Knowledge Management in Distributed Environments

Most current models of access control require a statically defined set of rules to match against requests. Often, a central authority disseminates access lists or/and resources themselves have local authorisation rules. However, both approaches have a common problem – they use static information for which update mechanisms are slow and cumbersome.

In pervasive and mobile environments, greater dynamicity and greater decentralisation are required in view of the uncertainties associated with trustworthiness of entities and the rapidity of context change. Decentralisation can be achieved by utilising dynamically acquired information about interactions with other network entities in the decision to trust or not to trust a given principal. However, given the scale of the environment, it is essential to examine the form such interactions might take to determine whether (and how) it is practicable to exchange this information.

It is fortunate for the decentralised approach that interactions between entities are non-random in nature. The physical structures of computer networks suggest that interactions between network entities form a small world network of preferable attachment [3] and this partially defines (and limits) the way that participants interact. Likewise, social cohesiveness is also a factor that limits the range of interactions and thus has the potential to be exploited both in managing the complexity of decentralised information gathering and in allowing behavioural stereotyping. A collected history of interactions is composed of a set of observations, and thus has two problems: (i) this history could become very extensive over time, (ii) there are no universally accepted semantics for observations.

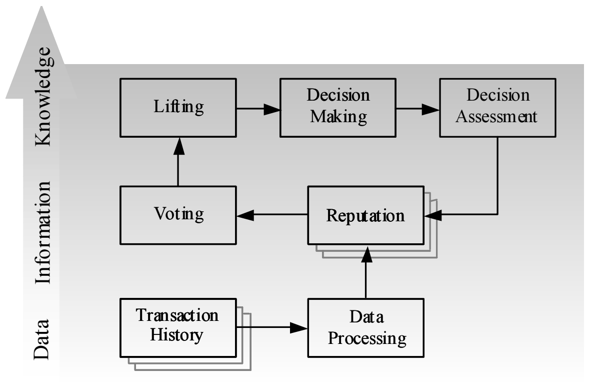

Figure 2. Information flow in ADAM

In ADAM, we collect this history and generalise it, to build knowledge about users. This knowledge is used as a credential whenever a user requests a network resource. In other words, trust decisions are based on *knowledge* built by fusing *information* obtained from heterogeneous distributed sources, which is, in turn, based on *data* obtained from direct observations (see Figure 2). Fortunately, the basic model of decision support – observe, orient, decide, act (OODA) – may be adapted here [4]. Agents *observe* histories of transactions (data), on which basis local reputations (information) are formulated. The *orient* functions include gathering different referrals and combining them (knowledge). To build this knowledge, the information must be correlated and lifted from one context to another since, to be able to *decide* how to proceed with the user request, it is necessary to compare observations based on different sets of parameters, taken from different environments (domains). After the decision has been made, agents *act*, translating their decision into a set of authorisation restrictions, providing access, and monitoring resource manipulations within the given access restrictions.

ADAM's agents do not form a fixed structure. They cooperate by dynamically forming different communities. By analogy with social communities [5] we identify, from an information flow perspective, two types of agent groupings: *groups of practice* and *groups of interest*.

Groups of practice consist of agents resident in the same knowledge domain, sharing common practice or a long-term interest. Groups of practice have a relatively long lifespan and they exist as long as common practice or interest exists. For example, people who work together on particular projects or problems constitute a single knowledge domain. Agents resident within this type of domain are connected by strong ties [3]. Whilst groups of practice do not often undergo dramatic changes in their structure of shared knowledge, they are not static. Groups of practice adapt to environmental changes by changing membership and by updating common knowledge over time. Agents in a group of practice provide the means to support learning for recently created agents by sharing knowledge.

Agents involved in distributed knowledge management form groups of interest when processing a user request to access a resource. A group of interest exists only for as long as is required to process a user request. The only interest of the group is the collection of referrals from internal (intra-domain) and external (inter-domain) sources. Whatever the relationship between groups of interest and knowledge, agents are connected by weak ties [3] inside a group of interest.

Separation into groups of practice and interest is context-driven and is conditioned by the boundaries of knowledge domains. It allows the superposition of structure on an otherwise intractable problem, providing scalable and rapid access control mechanisms. It also solves some compatibility problems – it is possible to use different knowledge management techniques within different groups of practice as long as standardised protocols are used inside groups of interest. This means that it is possible to implement local knowledge management more efficiently, using techniques that perform better on this type of data in this context.

4. Modelling Information and its Context for Access Control

Figure 3 outlines the process of decision making in ADAM. When a user requests an action, a secure channel is established between the user terminal and user agent using a session key. In Figure 3 it is possible to see that user sends *activation* and gets *acknowledgement*, after which she can request network services.

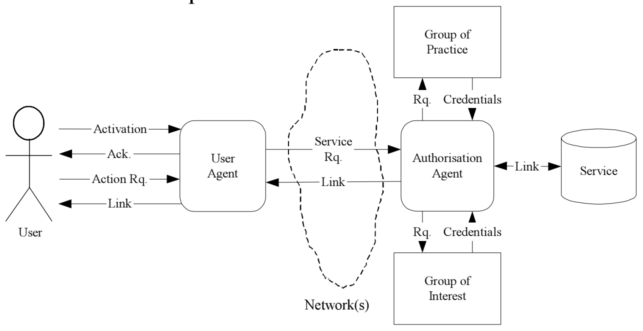

Figure 3. Decision-making process

Once a transaction is requested, the user agent connects to the service and passes the user's request. An authorisation agent is created to handle the request. It analyses the available user and connection data and returns information about service availability and requirements for a successful service request under current conditions. The user agent may, in turn, either provide the required information or ask the user to provide it.

At this point, the authorisation agent needs to ascertain whether the user is who they claim to be; a task that it delegates to third trusted parties. After the user's identity is confirmed, the authorisation agent must authorise the requested transaction. The main duty of this agent is to decide whether the requested transaction will lead to an event perceived to be beneficial (or neutral) to the resource, Va^+, or harmful, Va^- [6]. The relative strength of Va^- and Va^+ in arriving at a decision is defined by an agent's attitude towards risk, which is formed with help of local policy during initialisation. According to Deutsch [6] a trusting choice will occur if the following holds:

$$Va^+ \times S.P.^+ > Va^- \times S.P.^- + K \qquad (1)$$

where $S.P.^+$ is the subjective probability of attaining Va^+, and $S.P.^-$ of attaining Va^-. K is a "security level", which is defined by policy. The values of $S.P.^+$ and $S.P.^-$ have to be determined and analysed by authorisation agent.

For simplicity and speed, the request for referrals is first sent to the group of practice and, if there is insufficient information returned, members of the group of interest are asked to provide referrals. User agents may be resident in a different network from that in which the requested resource is situated (and the corresponding authorisation agent is resident). In this case, the user agent has no knowledge of local policy (access rules) nor of resource state (context) that would allow it to formulate an opinion about whether the request should be satisfied. An authorisation agent protecting a resource is, however, aware of the local policy. It translates a reputation query outcome (information about trustworthiness of the user's identity provided by an authentication agent) into authorisation restrictions with help of local policy. In other words, a decision about the user request is a result of negotiation between user and authorisation agents.

Once the requested action has been performed, the authorisation agent evaluates its experience in terms of Va^+ and Va^-, and disseminates this before being destroyed.

4.1 Contextual Trust

In our architecture, the assessment of user reputation is made according to information gathered from different sources situated in different knowledge domains. Each knowledge domain represents a group of practice that has own distinct interests, policies, and priorities. This means that each recommender agent bases its recommendation on some local information (possibly a history of transactions), the nature of which will be different in different domains. In addition to this, recommendations are biased by the domain's practices. We use the TLA+ logic [7] to specify referrals. This logic allows the specification and checking of models of concurrent systems, and provides the ability to reason under uncertainty.

Trust cannot necessarily be generalised [8]. Thus, we define and use the notion of context to describe circumstances under which trust relationships or recommendations are valid. In other words, *context of recommendation* and *context of trust* are used to describe them more precisely, and to establish and express the boundaries of their existence.

Context is an integral feature of any kind of information or knowledge [9]. The same information object can have different meanings and different names in two different contexts and be meaningless in a third. Thus, "*the information contained in a context is dependent on that context*" [9]. Fortunately, a range of different decontextualisation and lifting techniques have been developed and used in AI [10].

In order to be able to use information from different networks, we need to model it. In other words, we need to define different the types of data we expect to be available to our agents, describe it in such a way that we capture its meaning and make it understandable and useful for the agents. In ADAM there are numerous entities involved in information gathering and processing. Thus, there is a problem inherent in the data modelling task - differences of opinions (or even terminology) between agents. These are caused by differences of environments in which the agents are resident and the agents' perception of these environments. Thus, it is necessary to ensure that assertions made in different knowledge domains with regard to user reputation are held in the local context of requested resource where the decision has to be made. Determination of information's context and treatment of it as a separate object allows ADAM's agents to deal with the inevitable uncertainty and inconsistency in the information base.

In most cases, the presence of inconsistency renders information meaningless. One method of dealing with inconsistency in data is by restoring its consistency before reasoning. However, this is not always possible; moreover, some important elements of knowledge may disappear. Inconsistent or contradictory information can be represented in the same information base as long as it is treated in different contexts [9]. Knowing this, it may sometimes be useful to retain inconsistent information [11], especially in the environment in which ADAM operates. Depending on the context of a request, one of the contradictory pieces of information would be chosen for reasoning.

Currently, there is no generic definition of context. Different approaches define context for their own purposes, such as: lexical analysis [12], AI [13], mobile computing [15], etc. ADAM uses AI techniques and, therefore, should be able to use the definition of context established for AI [13]. However, in artificial intelligence, contexts have primarily been introduced as means of partitioning knowledge into manageable sets [14]. In ADAM, the grouping of information by context is not desirable. On the contrary, we need to combine pieces of information from different contexts by creating relative interpretations in a single context for all such pieces. Thus, in ADAM, *context* is any knowledge about the environment from which information or data was extracted. It is a set of assumptions about the environment formalised as an abstract object, relative to which the description of objects is given. In ADAM, it is not possible to separate knowledge from its context; knowledge exists only in the form of *relative interpretations*.

There are several elements on which context can be based [9], categorised as temporal, spatial, functional, and structural. Below we define them for ADAM and explain how they affect knowledge acquisition.

The first factor is *temporal*. The network environment changes over time and, therefore, knowledge about networks and their entities must be updated to reflect the changes. In other words, information may be correct in one temporal state of a knowledge domain and incorrect or meaningless in another. The temporal aspect of context raises a very important question of how to establish a tradeoff between accuracy of information and its freshness. ADAM assesses users' reputations according to the history of transactions. The longer the history, the more accurate the resulting knowledge, and the more confidence ADAM can have in its decisions. On the other hand, older transactions may have been performed in a context that is no longer extant, which may result in inadequate decisions. Thus, to avoid these situations, the length of the history of transactions must be carefully tailored.

Another vital factor in context formation is *spatial.* As mentioned above, ADAM agents function in a range of networks in which the same object may be described differently, information meaning may change, etc. It is crucial to preserve an object's meaning when transferring it from one context (domain) to another. In ADAM, the *functional* factor is aligned with the spatial because, by definition, we divide domains according to their practices.

The last main factor that may affect context formation is *structural*. If we have a complex object we may consider it from different points of view according to its main structural elements. Each point of view may constitute a different context.

In addition to the above-mentioned factors, there are two levels of abstraction for which context is involved: resource and domain. The domain context is determined by the main practice of the domain. Likewise, each resource inside the domain has its own, narrower, context that depends on the nature of information and services the resource provides. Figure 4 shows the two context factors.

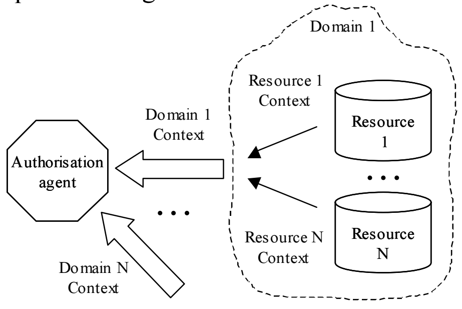

Figure 4. Resources' and domains' contexts factors

As can be seen from Figure 4, a model that takes into account both context factors would be most accurate and flexible. However, it is complex, since it requires two-step context combination process. In ADAM, we use both context factors, with some simplifications. For direct observations and recommendations, agents know and use contexts of the resources to which they are sending requests for referrals. If, however, an authorisation agent needs to ask some domain about a reputation, it has no knowledge of the concrete resources inside the domain. The domain itself identifies the sources and combines the information obtained from them into a trust value.

5. Conclusions

This paper presents a conceptual model of a knowledge management approach aimed at automation of the trust establishment process. The model relies upon two groups of agents: mobile user agents protecting user interests and authorisation agents protecting network resources. The access control decisions are results of negotiations between them. In this paper we also identified many problems that must be solved before the proposed model can be used efficiently in real life.

6. References

[1] Rothke, B. Security Strategies for E-Companies: an insiders view, Information Security Magazine. 2001.

[2] Seleznyov, A., Hailes, S. A Conceptual Access Control Model Based on Distributed Knowledge Management, To appear in Proceedings of 18th International Conference on Advanced Networking and Applications, Japan, 2004.

[3] Buchanan, M. Nexus: Small Worlds and the Groundbreaking Science of Networks, W.W. Norton & Company, 2002.

[4] Bass, T. Intrusion Detection Systems and Multisensor Data Fusion, *Communications of the ACM*, Vol. 43, Num. 11, pp 99 - 105, 2000.

[5] Fischer, G., Ostwald, J. Knowledge Management: Problems, Promises, Realities, and Challenges, *IEEE Intelligent Systems*, Vol. 16, Num. 1, pp. 60-72, 2001.

[6] Deutsch, M. The Resolution of Conflict, Yale University Press, New Haven, 1973.

[7] Lamport, L. Specifying Concurrent Systems with TLA+. *Calculational System Design*, (Eds.) Broy, M. and Steinbrüggen, R., ISBN: 90 5199 459 1, 1999.

[8] Barber B. Logic and Limits of Trust, New Jersey:Rutgers University Press, 1983.

[9] Theodorakis, M. Contextualization: An Abstraction Mechanism for Information Modelling, PhD, Department of Computer Science, University of Crete, 2001.

[10] Norrie, M., Wunderli, M. Coordination system modelling, Proceedings of the 13th International Conference on The Entity Relationship Approach, Manchester, 1994.

[11] Gabbay, D., Hunter, A. Making Inconsistency Respectable: Part 2 Meta-level handling of inconsistency, Symbolic and Qualitative Approaches to Reasoning and Uncertainty (ECSQARU'93), LNCS, pp. 129-136, 1993.

[12] Buvac S. Resolving Lexical Ambiguity using a Formal Theory of Context, in Van Deemter and Peters (Eds.) Semantic Ambiguity and Underspecification, CSLI Publications, Stanford, 1996.

[13] McCarthy, J. Generality in artificial intelligence, *Communications of the ACM*, 30(12), pp. 1030-1035, 1987.

[14] Hendrix, G. Encoding Knowledge in Partitioned Networks. In Nicolas Findler, eds., *Associative Networks*. New York: Academic Press, 1979.

[15] Julien, C., Roman, G.-C., and Huang, Q., "Declarative and Dynamic Context Specification Supporting Mobile Computing in Ad Hoc Networks," Technical Report WUCSE-03-13, Washington University, CS Department, St. Louis, Missouri.