A Negotiation-based Trust Establishment Service for CROWN Grid

Jianxin Li* Jinpeng Huai Li Lin
School of Computer Science and Engineering
Beihang University, Beijing, China
E-mail: {lijx, huaijp,linli}@act.buaa.edu.cn
*Corresponding author

Jie Xu
School of Computer,
University of Leeds, U.K.
E-mail: scsjx@leeds.ac.uk

Abstract: In order to build trust relationship between service requesters and service providers in an open grid computing environment, we design a novel negotiation-based trust establishment service which supports distributed credential chain construction and privacy preservation to enhance the grid security infrastructure. In this service, we develop a novel **CR**edential ch**A**in aware **NE**gotiation strategy (CRANE) for trust establishment on the fly by gradually disclosing credentials according to various access control policies. This strategy can protect sensitive credentials, partial credential chains and sensitive information in access control policies based on two concepts: Soft Protection and Hard Protection. What's more, a credential federation mechanism is designed for this service when the negotiators use heterogeneous security infrastructures, e.g., Kerberos and PKI. Our approach has been successfully implemented as useful components and fundamental security services in the CROWN Grid, and techniques such as trust tickets and policy caching that can greatly increase service efficiency are used. Comprehensive experiments have been conducted, which demonstrate our approach is feasible.

Keywords: Grid Computing; Trust Management; Trust Negotiation; Security Policy; Credential Federation; Privacy

Biographical notes: Jianxin Li received his PhD in Computer Science from Beihang University in 2007. He is an Assistant Professor at the School of Computer, Beihang University. His current research interests include grid computing, trust management.

Jinpeng Huai, PhD, Professor. His current research interests include computer software and theory, network middleware and grid computing, network security.

Li Lin, PhD Candidate. Her current research interests include grid computing, P2P computing, algorithm design and analysis for network resource management.

Jie Xu is Chair of Computing at Leeds University. In 1990 he joined the University of Newcastle upon Tyne and received the PhD degree there in Advanced Fault-Tolerant Software. In 2004 he joined the University of Leeds. His current research interests include computer system fault diagnosis, fault-tolerant software and dependable distributed system.

1 INTRODUCTION

Grid computing promises to empower highly desirable resource sharing and problem solving in dynamic, multi-institutional virtual organizations (Foster, Kesselman et al. 2001; Pearlman, Welch et al. 2001). Grids are becoming a large-scale distributed computing environment, where a potentially unbounded number of users and services without pre-existing trust relationships may be involved. Traditional access control methods based on user identity in a virtual organization are not sufficient since they do not scale as the number of users and services increase dramatically and the behaviours of users and services are highly dynamic. It has been a fundamental but challenging problem to dynamically build mutual trust relationship

between service requester and provider coming from different security domains while preserving their privacy in an open grid environment.

Nowadays, several systems, such as Akenti (Thompson, Essiari et al. 2003), Permis (Chadwick and Otenko 2002) and GridShib (Welch, Barton et al. 2005), are used in grids and support flexible authorization based on user attributes instead of identity. These systems, however, are style of one-shot authorization mechanism. To become a valid user, requesters have to disclose enough information, which often includes their sensitive information. Furthermore, they only provide a coarse-grained approach, mostly is binary (disclose or not), to protect the sensitive information. Therefore, it is important to design a practical trust establishment service supporting collaborations across organizations in open Grid computing environments.

Automated Trust Negotiation (ATN) (Winsborough, Seamons et al. 2000) is an advanced approach for trust establishment across multiple domains. The participants can establish trust through interactively disclosing credentials and access control policies while preserving their privacy. TrustBuilder (Winslett, Yu et al. 2002) and PeerTrust are two representative systems for trust negotiation in use now. However, these systems suffer three major drawbacks. First, TrustBuilder is based on IBM TE (Herzberg, Mass et al. 2000) TPL and X.509 v3 certificate. Since TE does not support flexible delegation statement, it can not express authorization delegation across multiple security domains, which seriously limits the ability of trust establishment over open Internet. Second, PeerTrust (Constandache, Olmedilla et al. 2005) is a logic-based inference engine for trust negotiation. But it is not trivial for ordinary users to edit the logic policies, and meanwhile no corresponding supported services and tools are developed. Third, both TrustBuilder and PeerTrust ignores a condition when two negotiators have different credential formats and heterogeneous security infrastructures e.g., PKI and Kerberos, so it is necessary to design an identity mapping and credential conversion mechanism to solve this problem. Finally, in some scenarios, most entities in the same domain often need to retrieve the same credentials during the different negotiation sessions, so a proxy-based trust establishment mechanism also is needed.

To address the above issues, we develop a practical trust establishment service, which enhances grid security infrastructure and support flexible trust establishment between strangers from different security domains. This approach has been successfully implemented in CROWN (Huai, Hu et al. 2007), and the main contributions of our work are as follows:

1. A negotiation-based trust establishment policy is proposed based on an extended trust management language, which can express authority delegations, attribute parameter constraints and delegation depth constraints. Moreover, a novel trust negotiation strategy, CRANE, is presented to protect sensitive information in a fine-grained sense during credential chain construction. By employing two innovative concepts of *Hard Protection* and *Soft Protection*, CRANE can meet various privacy preservation requirements. In CRANE, a priority ranking method is provided to deal with the issue that multiple credential sets satisfy an access control policy. Using this method, an entity has guiding to disclose their credentials with various tactics.

2. We have designed and implemented a negotiation service complied with `WSRF` specification in the CROWN grid. In this service, two techniques, *Trust Ticket* (Bertino, Ferrari et al. 2004) and *Policy Caching*, are adopted to improve negotiation performance in practice. A credential federation mechanism is designed to support credential conversion between heterogeneous security infrastructures. Especially, this service has been successfully applied to build trust relationship in CROWN remote & hot service deployment (Huai, Sun et al. 2007).

The remainder of this paper is organized as follows. Section 2 presents the negotiation-based trust establishment architecture and Section 3 describes the negotiation strategy *CRANE*. We introduce implementation experience of trust negotiation engine and service in Section 4. The performance evaluation is given in Section 5. We discuss related work in the area of trust negotiation and grid security infrastructure in Section 6. Finally, we conclude the paper in Section 7.

2 POLICY AND CREDENTIAL LANGUAGE

In CROWN, we adopt the trust management language RT_0 and RT_1 (with role parameters) (Li, Winsborough et al. 2003) to depict policy statements.

In RT language, an *entity* is a uniquely identified individual or organization, and we generally use *A*, *B*, and *C*, sometimes with subscripts, to denote entities. A *role* takes the form of an entity followed by a role name, separated by a dot, e.g., *A.r* and *BHU.student.* Additionally, a role name may contain some parameters, e.g. *BHU.student* (*age*, *dept*). Here, we use symbol γ to denote the constraints of role parameters and delegation depth. For instance, we can define γ = (*age*>18 and *dept='CS'*) as a constrain of this role for a role *BHU.student*(*age*, *dept*). The four basic rules (Li, Winsborough et al. 2003) are as follows.

Rule 1: Simple Member Rule

$$A.r(m_1, m_2, \ldots, m_k) \leftarrow B: \gamma$$

It means that $B \in member(A.r)$ where $member(A.r)$ represents a set of entities who are members of *A.r*, m_k is the role parameter.

Rule 2: Role Mapping Rule

$$A.r_0(m_1, m_2, ..., m_k) \leftarrow B.r_1(n_1, n_2, ..., n_s) : \gamma$$

This means $member(A.r_0) \supseteq member(B.r_1)$.

Rule 3: Linking Role Rule

$$A.r_0(m_1, m_2, ..., m_k) \leftarrow B.r_1(n_1, n_2, ..., n_s).r_2(l_1, l_2, ..., l_t) : \gamma$$

This means *A* asserts that an entity has attribute r_0 if there is a *C* such that $C \in member(B.r_1)$ asserts that this entity has attribute r_2.

Rule 4: Roles Intersection Rule

$$A.r_0(m_1, ..., m_k) \Leftarrow B_1.r_1(n_1, ..., n_s) \cap ... \cap B_k.r_k(l_1, ..., l_t) : \gamma,\ k \geq 2.$$

It means $member(A.r_0) \supseteq (member(B_1.r_1) \cap ... \cap member(B_k.r_k))$. A credential *e* can be defined with a signed access rule, and a set of credentials is denoted as symbol *E*. Traditional certificates or attribute credentials only contain entity's identity or attributes, but credentials here can contains attribute mapping and permission delegation rules across multiple domains. Of course, how to store and collect these credentials is a crucial problem, and Li et al. (Li, Winsborough et al. 2003) have designed a distributed credential discovery algorithm.

3 NEGOTIATION STRATEGY: CRANE

The negotiation strategy that controls how and when to disclose information is a key component in ATN (Winsborough, Seamons et al. 2000). We have developed a negotiation strategy CRANE to support dynamic trust establishment while preserving privacy. The aims of CRANE is to construct a credential chain from the requester to the ultimate goal. The credential chain construction requires participants to disclose credentials and access control policies iteratively. In order to preserve privacy related to credential chains, one special feature of CRANE is that not necessary to disclose all details of a credential chain. Hereby, the disclosed credentials that satisfy on one side may not really satisfy the mediator's original policy. That means, even if the credentials of *Alice* satisfy a policy *P* disclosed by *Bob*, they are possible not to satisfy the original policy P_{org} yet because some sensitive information in P_{org} have been filtered.

3.1 Sensitive Information Protection

In CRANE, we coined two sensitive information protection approaches, *Soft Protection* and *Hard Protection*.

- *Soft Protection*: Assigning the access control policy to sensitive credentials, parameters and policies.
- *Hard Protection*: Hiding some sensitive information, such as constraints on role parameters, partial credential chain etc.

In general, the *Hard Protection* approach is used based on backward credential chains construction. Figure 1 illustrates two scenarios.

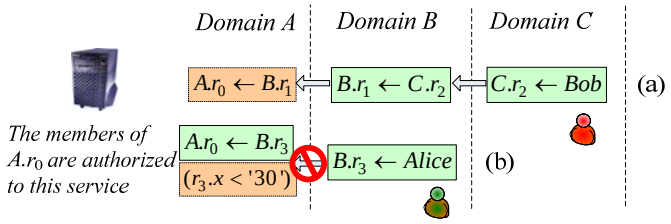

Figure 1 *Credential Chain Construction with Hard Protection*

In scenario (a) , *A* can construct a credential chain $A.r_0 \leftarrow B.r_1 \leftarrow C.r_2$ with a backward fashion, but *A* would not like to disclose it's relation with *B*, and there is no access control policy assigned to these credentials. If TTG approach is employed, the trust could not be established in this scenario because it require participants to disclose their full credential chains. However, if *A* discloses the partial credential chain, the trust still may be established. Hence *A* need not disclose the full credential chains, but just creates a new policy which asks *Bob* to prove he is a member of role $C.r_2$. In scenario (b) , *Hard Protection* approach is used to protect role parameter $r_3.x$. Although a successful credential chain is realized on *Alice*'s side, the trust should not be established since the constraints information in the disclosed policy is incomplete and the final credential verification is failure.

In CRANE, we define a directed acyclic graph, called *TrustGraph*, to construct credential chains and control information disclosure based on above privacy information protection approaches. Furthermore, we define a *SatTree* to maintain a priority on credential sets so as to disclose minimal satisfied credentials every time.

3.2 Trust Graph and SatTree

Since there may be multi-set of credentials that satisfy the special access control policy, *TrustGraph* often contains more than one path of credential chain. Next, we will give the definitions of *TrustGraph* and *SatTree*, and show methods to construct credential chains while protecting sensitive credentials and sensitive information in access control policies.

Definition 1 (Trust Goal): A Trust Goal is a triple tuple $\langle RoleExpr, \gamma, entity \rangle$, and this means it needs to construct a credential chain from an *entity* to role *RoleExpr* such that $entity \in member(RoleExpr)$ and satisfies constraints γ.

According to the categories of *RoleExpr*, there are four types of Trust Goal.

Definition 2 (TrustGraph): *TrustGraph* is a directed acyclic graph $\langle Node_T, Edge_T \rangle$, where $Node_T$ is a set of nodes of *Trust Goals* or entities, and $Edge_T$ is a set of edges adjacent to $Node_T$. It holds following properties:

- $Node_T$ is non-empty and finite;
- The edge in $Edge_T$ has three types: *Credential Edge*, *Composition Edge* and *Special Edge*;
- The leaf node of *TrustGraph* is a *Special Goal* or a *Common Goal*.

As shown in Figure 2 (only *RoleExpr* is showed in a *Trust Goal*), we can see nodes are connected with different edges, where *Credential Edge* denotes there is a credential between two roles. For example, the nodes of a credential $A_1.r_1 \leftarrow A_2.r_2 \cap A_3.r_3$ are connected with a *Credential Edge*. There are also two important auxiliary edges: *Composition Edge* and *Special Edge*. *Composition Edge* is used to connect each role expression in the intersection role part and *Special Edge* is used to connect a *Linking Goal* with a new *Special Goal*.

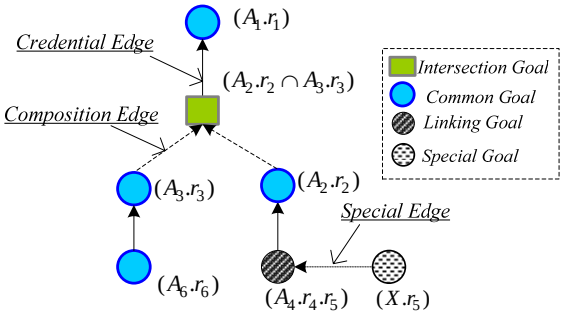

Figure 2 *Nodes and Edges in TrustGraph*

The bidirectional credential chain construction method (Li, Winsborough et al. 2003) is used in *TrustGraph* for different purposes. One is *backward credential chain construction* which determines entities that belong to a specified role, and we use it to create access control policies. The other is *forward credential chain construction* which determines roles to which an entity belongs, and we use it to generate the sets of credentials that satisfied the policies.

Definition 3 (The Minimal Solution of Policy): Let P be an access control policy, and $E = \{e_1, ..., e_n\}, n \in \mathrm{N}$ be a credential set satisfies P, then we call E as a solution of P, denoted as $E \Rightarrow P$. If there is $\forall E' \subset E$, holds $E' \not\Rightarrow P$, then E is a minimal solution of policy P, denoted as $E \Rightarrow_{\min} P$.

To select an appropriate policy solution, we assign a sensitive rank to every credential. Given a credential set E owned by s, $w(E)$ is the total rank of sensitive credentials in E for s, that is $w(E) = \sum rank(e_i)$ where $e_i \in E_{sen} \subseteq E$. The default rank value of a credential is 0, it means that this credential is non-sensitive. If $w(E) = 0$, it means that there is no sensitive credential in E. Moreover, we also use $\delta(E)$ to denote the number of credentials in E, that is $\delta(E) = |E|$.

We denote an access control policy P as a formula in disjunctive normal form $P = (p_1 \vee ... \vee p_n)$. Then, for every atomic policy p_i, we can generate a set of solutions $\{E_{i,1}, ..., E_{i,k}\}$, where $E_{i,j} \Rightarrow_{\min} p_i$, $1 \le j \le k$.

Definition 4 (Binary Relation on Credential Sets): The binary relation $\trianglelefteq$ on credential sets ΣE is defined as follows:

C1) if $w(E_{i,j}) \neq w(E_{k,m}), (E_{i,j} \trianglelefteq E_{k,m}) \Leftrightarrow w(E_{i,j}) < w(E_{k,m})$;

C2) if $w(E_{i,j}) = w(E_{k,m})$ and $\delta(E_{i,j}) \neq \delta(E_{k,m})$, then $(E_{i,j} \trianglelefteq E_{k,m}) \Leftrightarrow \delta(E_{i,j}) < \delta(E_{k,m})$;

C3) if $w(E_{i,j}) = w(E_{k,m})$ and $\delta(E_{i,j}) = \delta(E_{k,m})$, then $(E_{i,j} \trianglelefteq E_{k,m}) \Leftrightarrow (i < k) \vee ((i = k) \wedge (j \le m))$.

Theorem 1: Binary relation $\trianglelefteq$ is a full order on ΣE.

Proof: This proof aims to show that binary relation $\trianglelefteq$ satisfy the properties: reflexive, antisymmetric and transitive.

i) For any $E_{i,j} \in \Sigma E$ such that $w(E_{i,j}) = w(E_{i,j})$ and $w(E_{i,j}) = w(E_{i,j})$, there exists

$$(i = i) \wedge (j = j) \Leftrightarrow (E_{i,j} \trianglelefteq E_{i,j}).$$

It means that $\trianglelefteq$ is reflexive.

ii) For any $E_{i,j}, E_{k,m} \in \Sigma E$, we proceed by contradiction. Let $E_{i,j}$, and $E_{k,m}$ be symmetric, that is $E_{i,j} \trianglelefteq E_{k,m}$ and $E_{i,j} \trianglelefteq E_{k,m}$. if $E_{i,j} \trianglelefteq E_{k,m}$, then there are three cases:
For the case of C1), there is $w(E_{i,j}) \neq w(E_{k,m})$, then

$$w(E_{i,j}) < w(E_{k,m}) \text{ and } w(E_{k,m}) < w(E_{i,j}).$$

It is contradictions!
For the case of C2), there is $w(E_{i,j}) = w(E_{k,m})$ and $\delta(E_{i,j}) \neq \delta(E_{k,m})$, then

$$w(E_{i,j}) < w(E_{k,m}) \text{ and } w(E_{k,m}) < w(E_{i,j}).$$

It is contradictions!
For the case of C3), there is $w(E_{i,j}) = w(E_{k,m})$ and $\delta(E_{i,j}) = \delta(E_{k,m})$, then we have

$$((i < k) \vee ((i = k) \wedge (j \le m))) \wedge ((k < i) \vee ((k = i) \wedge (m \le j))).$$

It is contradictions!

The above proof shows that the symmetric assumption are contradicted in all cases. Hence, the binary relation is antisymmetric.

iii) As the proof procedure of *ii*), we can also prove that $\trianglelefteq$ is transitive.

In summary, from *i*), *ii*) and *iii*), the relation $\trianglelefteq$ on ΣE is a full order. □

Definition 5 (SatTree) Let an access control policy $P = (p_1 \vee ... \vee p_n)$ be a formula in disjunctive normal form. All the minimal solutions of P can be organized as a three-layer tree where the root node is P, the middle nodes are formulas $p_k (1 \le k \le n)$ in conjunctive normal form. The leaf nodes are credentials sets which are arrayed according to full order relation $\trianglelefteq$. We call this tree as *SatTree*.

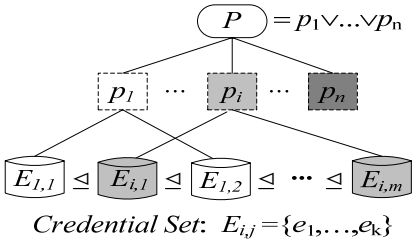

Figure 3 *The SatTree*

Based on a *SatTree*, we can justify whether the policy P is satisfied or not and disclose the appropriate credential set.

Next, we will show how trust is established through an example. As Table 3 shows, both *Alice* and *MedServer* have their own credentials and policies. *Alice* protects credential e_1 with $Rule_1$, and *MedServer* makes authorization decision to its service according to $Rule_2$.

This negotiation includes three rounds of interaction, and the constructed *TrustGraph* and *SatTree* during trust establishment are illustrated in **Figure 4**, $policy_1$ and $policy_2$ are generated dynamically using backward credential chain construction during negotiation

Table 3. *The credentials and access rules in Example* 1

	Alice	*MedServer*
	e_1, $rule_1$	e_2, $rule_2$
	e_1: *BHU.researcher←Alice* e_2: *MA.accredited ←MedServe* e_3: *CHEA.university ←BHU* (In *CHEA's CredMan*)	
		$rule_1$: *e1.access←MA.accredited* $rule_2$:*Medserver.access←CHEA.university.researcher*
		$policy_1$: <*X.researcher ? Alice*> $policy_2$: <*MA.accredited ?MedServer*>

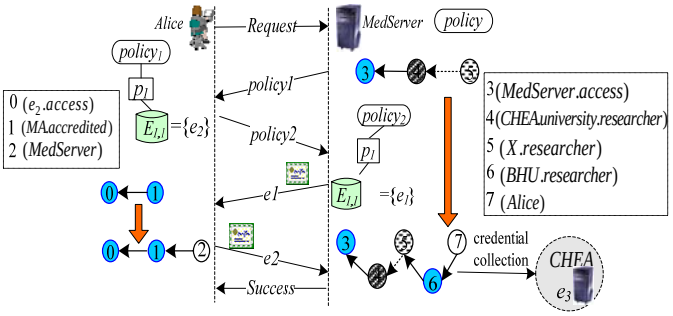

Figure 4 *Negotiation between Alice and MedServer*

4 SYSTEM ARCHITECTURE AND IMPLEMENTATION

We have successfully implemented our designing in the CROWN grid. CROWN aims to promote the utilization of valuable resources and cooperation of researchers nationwide and world-wide. The trust management architecture designed in CROWN provides basic secure communication and policy-based access control based on a series of Web Service security standards. Furthermore, the trust negotiation service is designed, and we have launched some pilot applications using this trust negotiation service (Sun, Zhu et al. 2005).

4.1 CROWN Trust Management Architecture

The overall trust management architecture of CROWN is shown as **Error! Reference source not found.**. We use the term *region* to denote different area with heterogeneous security infrastructures such as PKI or Kerberos; term *domain* to denote an autonomous organization and term *node* to denote a ServiceContainer implemented based on

GT4 WS-Core Container with various new features and extensions, such as remote and hot service deployment, monitoring and management service, etc.

In this architecture, we present a policy-based and fine-grained secure communication mechanism supporting message-level encryption/decryption, signing/verification etc. Moreover, other new security functionalities can be easily integrated with this architecture due to its flexibility. It is highly flexible though configuration, which facilitates administrators to specify fine-gained security policies for any service. In addition, an authentication service and an authorization service can be deployed in every domain, and a credential management service (*CredMan*) as a MyProxy (Basney, Humphrey et al. 2005) replacement in CROWN middleware. *CredMan* service allows users to access their credentials anywhere, anytime, even when they are on a system without grid infrastructure. However, *CredMan* service is implemented as a grid service, and it is decoupled with underlying security mechanisms. On the other hand, MyProxy is heavily coupled with SSL as a session security mechanism and a built-in access control model which is hard coded and inflexible to be extended.

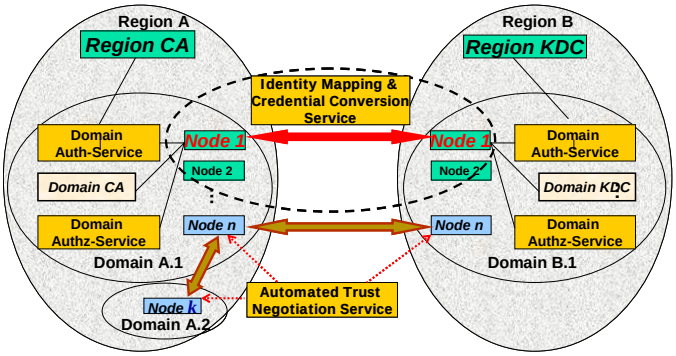

Figure 5 *Trust Management Architecture in CROWN*

ATNService provides trust negotiation between nodes in consideration of privacy protection, it complies with `WS-Trust` specification. In this service, an identity mapping and credential conversion mechanism is designed for nodes from heterogeneous regions. Therefore user with either X.509 certificate or Kerberos ticket can be authenticated via a `WS-SecureConversation` service which conforms to `GSS-API` standard.

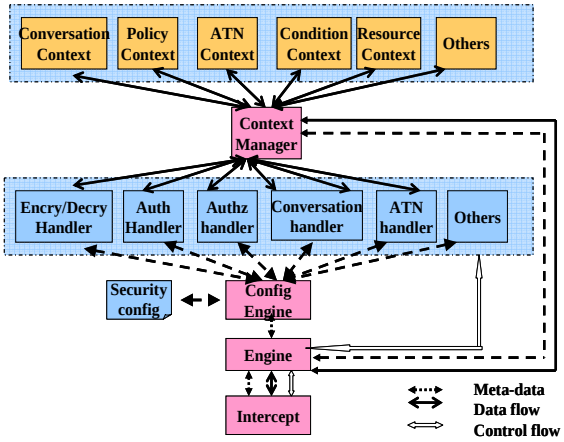

Figure 6 *Security Handlers in CROWN*

Error! Reference source not found. depicts security handlers on the service container layer. This designing is much inspired by Axis. The security processing relies on a configurable security *chains*, and generally two chains, which deal with the request and response messages of service respectively, are configured for every grid service. For the sake of simplicity, there is only a processing chain shown in this figure. Each *handler* in the processing chain is in charge of some specific security functions. The grid services developers, deployers and administrators can customize services protection by merely configuring these chains. Besides the information related to the request or response message, some other information such as session keys, state of automated trust negotiation, properties of resources are also needed by the handlers to finish their processing, and these information is called *security contexts* and is classified and managed by a *context manager*.

4.2 KedFed, ATNEngine and ATNService

Error! Reference source not found. shows structure and data flow among the modules inside *CredFed* implementation. The input of *CredFed* is user's credential, and the output is a new credential in a format different with its input credential. It also demonstrates a procedure of mapping a X.509 credential to a Kerberos credential. First, the input credential is processed by the authentication module, which is realized by secure-conversation mode offered by underlying communication security component of CROWN, to verify whether the user is the real owner of this credential. If so, the credential is then forwarded to identity mapping module, which will map the identity of user to another domain based on mapping policy. Then the new identity will be processed by credential conversion module to generate a new credential for the user. Finally this credential is returned to the user by *CredFed*. In *CredFed*, each module has its corresponding policy that can be customized by an administrator.

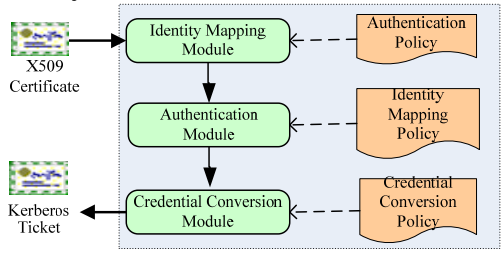

Figure 7 *CredFed Component*

The architecture of the *ATNEngine* is shown in **Figure 8**, and it contains four components.

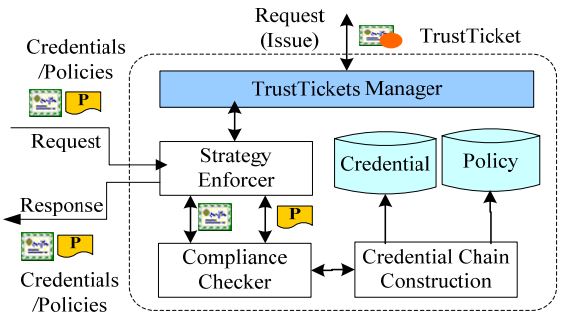

Figure 8 *The architecture of ATNEngine*

- ***Negotiation Strategy Enforcer***: Creating the local session manager and disclosing messages generated with CRANE.
- ***Compliance Checker***: Determining whether local credentials satisfy specified access control policies or getting satisfied credentials for access control policies.
- ***Credential Chain Construction***: In grids, trust establishment often involves finding a credential chain that delegates authority from the resource to the requester. The main function of this component is to collect credentials and construct a credential chain.
- ***Trust Ticket Manager***: Issuing or verifying short-term tickets to avoid re-negotiation when multiple service access requests occur in a short time interval.

The procedure of trust negotiation is shown in Figure 9.

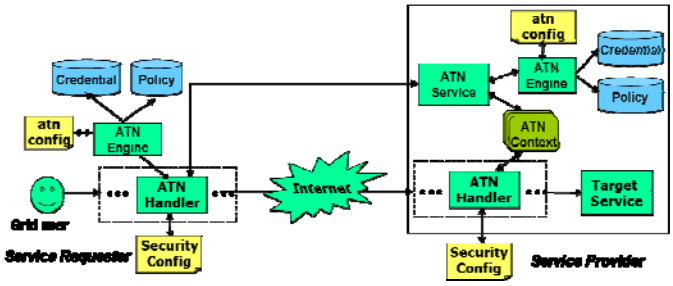

Figure 9 *Request Chain for ATNService and Security Handlers*

As illustrated in Figure 9, a series of procedures are involved in the trust negotiation. When a *Grid user requests* the target service which protected by trust negotiation service, it will firstly initialize an *ATNEngine* through local *ATNHandler*. Upon receiving the negotiation requests from client, the service provider will create an *ATNEngine*, too. The state of negotiation will be stored in *ATNContext*. Then, the two participants may disclose their credentials according to the provider's policy; or policies for sensitive credential. This process will be interacted until a final decision ('success' or 'failure') is reached. If the negotiation succeeds, *ATNService* will return a success status, and the context will be updated accordingly. The requester can insert the session id into SOAP header and sign it before sending to the target service. The target service will verify the authenticity of session id through its *ATNHandler*, and allow the access if the verification succeeds.

Although collaborations between strangers across multiple security domains become possible with the support of trust negotiation service, the performance of service access may be decreased due to additional interactions. To improve performance and applicability of negotiation service, we use trust ticket (Bertino, Ferrari et al. 2004) and policy caching mechanisms.

In CROWN, we also developed an Eclipse Plugin-based console, named *CROWN Launcher* (shown in Figure 10 and Figure 11), which provides a tool to manage credentials, access control policies and config files. In *CROWN Launcher*, client command parameter string can be generated via a wizard, and the negotiation procedure between client and server is displayed on a special GUI view, and we can check detailed information by clicking the policy or credential icon in the view. Moreover, the SOAP messages during negotiation also can be logged.

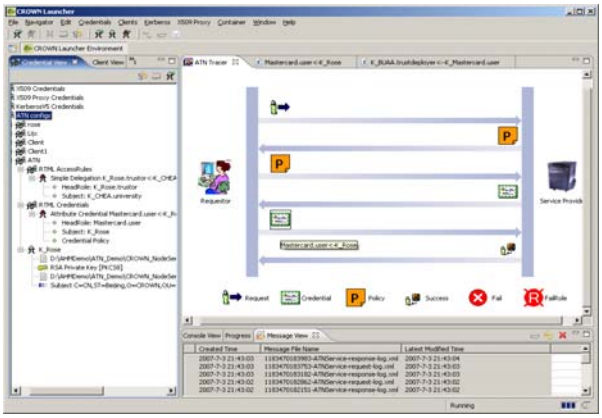

Figure 10 *Screenshots of ATN in CROWN Launcher*

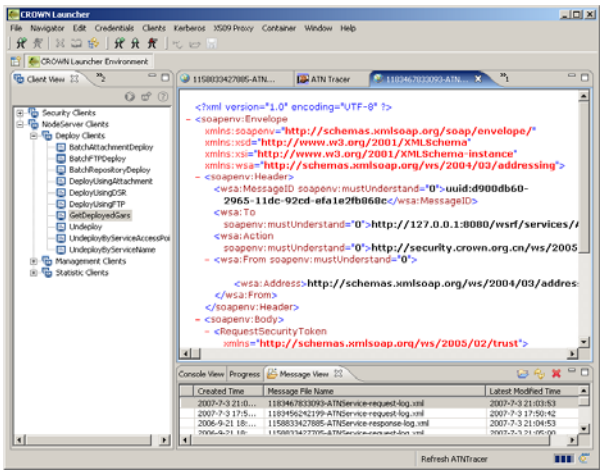

Figure 11 *Screenshots of ATN in CROWN Launcher*

5 EXPERIMENTAL STUDY

We have conducted some experiments in CROWN test bed. CROWN middleware is deployed on a cluster node with Intel Xeon 2.8GHz CPU, 2G RAM, Linux operating systems and 100Mbps Internet connection. We evaluated

our engine and service by some comprehensive experiments and the results indicate it is feasible. We developed the following metrics:

- *Negotiation Response Time*: To measure the response time from messages input to output for the *ATNEngine*.
- *Percentage of Negotiation Processing* (%): To measure the percentage of execution time of an engine during the whole negotiation procedure. The percentage is computed as follows:

$$(\sum t_k^{se} / (t_e - t_0)) \times 100\% ,$$

where $\sum t_k^{se}$ is the total time used during k rounds of strategy enforcing, and t_e is the negotiation accomplishment time and t_0 is the negotiation start time.

- *Service Execution Time*: Assume that there are n requests to the *ATNService* concurrently at t_0, and all sessions finished at t, that means the service execution time is $\Delta t = t - t_0$.

Experiment Group 1: *Launching the services deployed on cluster node, and a requester invokes the negotiation* service, *we only measure the performance of ATNEngine*.

Based on Example 1, we design another two examples: Example 2 (with delegation depth constraints) and Example 3 (with role parameter constraints). In order to make a fair comparison with TrustBuilder, these examples need not to discover credentials from remote CredMans. We run these examples in order and get response time of each negotiation round.

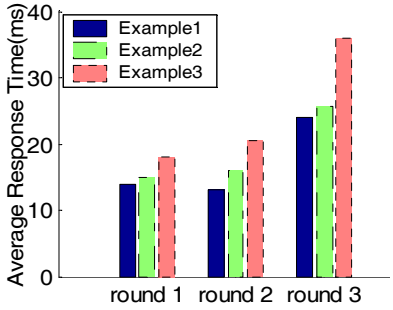

Figure 12 *negotiation rounds vs. average response time*

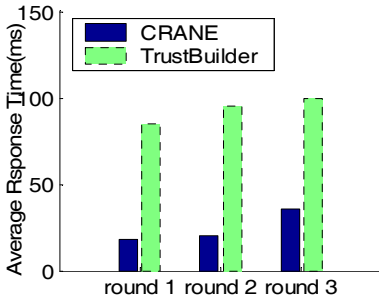

Figure 13 *negotiation rounds vs. average response time*

Figure 12 shows the average negotiation response time of each round. The result indicates that the average response time of Example 3 is some higher than the other two example. As a whole, the time is approximately 15-40 ms in CRANE. In Addition, we designed a similar scenario with Example 1 for TrustBuilder, and get the results. As shown in Figure 13 the response time of TrustBuilder is higher than CRANE.

The time spent in different phases of negotiation service as follows:

Negotiation Processing (ms)	Security Communication (ms)	Others (ms)
51	262	521

The percentage of negotiation processing time is:

$$(\sum t_k^{se} / (t_e - t_0)) \times 100\% = 51/835 = 6\% .$$

As shown in Figure 14, only 6% of time is spent on negotiation but around 31% is spent on security communication, e.g. SOAP message encryption and signature handler processing. The biggest part (around 62%) is used by other processes (e.g., request message parsing) on the side of client and server. The results show that the impact of negotiation processing on the service performance is very small.

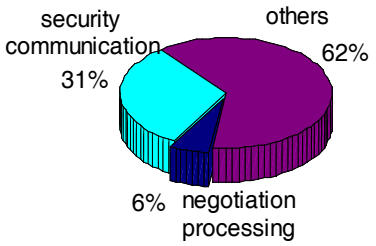

Figure 14 *the percentage of negotiation accomplishment time*

Experiment Group 2: *Launching the services deployed on cluster node, and multiple concurrent requests invoke the same negotiation service, we measure the performance of ATN Service*.

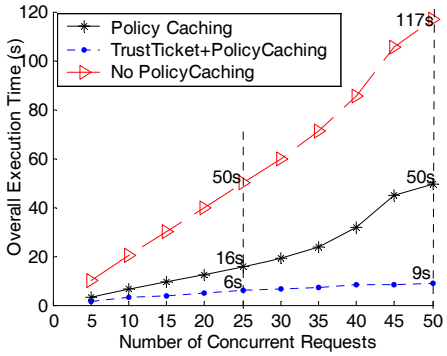

Figure 15 *concurrent requests vs. service execution time*

Figure 15 plots the overall execution time against the number of concurrent requests varying from 5 to 50 with step of 5. As we can see, the overall execution time increases linearly with the increasing number of concurrent requests and the impact of trust ticket and policy caching on service performance is obvious. For instance, when there are 25 concurrent requests, the service execution time is 6s, 16s and 50s respectively, and the time is 9s, 50s and 117s when 50 concurrent requests arrive. This figure shows that strong reducing trend of overall execution time for the trust ticket and policy caching case. In CROWN, we believe that the service can greatly benefits from these mechanisms.

6 RELATED WORK

There are many investigations (Bonatti and Samarati 2003; Yu 2003; Yu and Winslett 2003; Dragoni and Saidane 2008) such as policy graph, policy migration and filtration on trust negotiation regarding to privacy preservation for sensitive credentials and access control policies. Theoretically, all of these technologies can be used in *ATNService* . However, the trust negotiation with the support of distributed credential chain construction is not fully addressed by previous approaches besides TTG. In particular, to the best of our knowledge, the practical implementations and applications on trust negotiation are few. Therefore, we try to give a practical architecture for trust negotiation in Grids.

Trust management (TM) which provides distributed authorization mechanism is a fundamental technology for ATN. Li et al. (Li, Mitchell et al. 2002) presented an expressive trust management language, RT_0, which supports attribute-based delegation and subsumes SDSI/SPKI. TTG is a negotiation protocol designed based on RT_0. TrustBuilder (Winslett, Yu et al. 2002) is a system based on TPL which was developed by IBM Research Lab.

Currently, Grid Security Infrastructure (GSI) (Welch, Siebenlist et al. 2003) is a popular security middleware developed by the Globus project, which has been successfully used in many projects. New XML security specifications, e.g., SAML and XACML are also offered in GT4. The most representative systems for grid authorization are CAS (Community Authorization Server) (Pearlman, Welch et al. 2001), VOMS (2005) and Akenti (Thompson, Essiari et al. 2003; Ryutov, Zhou et al. 2005). CAS allows service providers to delegate some of the authority for maintaining fine-grained access control policies to communities, the VOMS provides a centralized authorization repository at Virtual Organization level for users' attributes, and Akenti makes authorization decision based on user's attributes got from certificates. MyProxy (Basney, Humphrey et al. 2005) allows users to delegate credentials to services acting on their behalf. Although these systems and services are offered, there is much room for further strengthening grid security. Specially, they do not concern much sensitive information protection and assume that attribute credentials are publicly available during the authorization decision. Rather, a negotiation approach is necessary when establishing trust between strangers. GridShib (Welch, Barton et al. 2005) manages attribute release via a web-based user interface, but do not support delegation amongst multiple attribute authorities and user still needs to obtain a valid certificate from a CA. J.Basney et al. (Basney, Nejdl et al. 2004) have also recognized the importance of trust negotiation in grids, and proposed the PeerTrust language based on distributed logic program. Currently, PeerTrust (Constandache, Olmedilla et al. 2005) also makes an effort to integrate with GT4. Moreover, as mentioned in (Winsborough and Li 2002), current approaches make unrealistic simplified assumptions about the language for the credentials expression, and irrational assumption for the credential storage. In fact, for any a potent system, the distributed credential discovery (delegation of authority) should be adequately supported. TTG (Trust Target Graph) (Winsborough and Li 2002), designed based on RT_0 (Role-based Trust Management) language that supports delegation of authority, is a closest work to this paper, but TTG has some limitations in several aspects. First, *role* used in TTG doesn't includes attributes parameter and delegation depth constraints, and TTG does not consider to protect sensitive information within the credential chains and access control policies. Second, TTG does not provide efficient credential disclosure tactics where there is multi-set of credentials satisfy a special policy. Finally, TTG also does not consider heterogeneous security infrastructures, and there is still no implemented system for it even now.

Our approach has some features in common with above systems, but specially extends a expressive trust management language to support flexible decentralized authorization, and provides a practical negotiation-based trust establishment service for entities from different security domains, even they adopts heterogeneous security infrastructures. We believe it is an interesting complement for existing grid security infrastructures when trust is needed between unknown entities.

7 CONCLUSION AND FUTURE WORK

A practical trust establishment service is important to security and trustworthiness of grid computing environments. Our work is an essential complement to existing grid security infrastructures. In our study, we have designed a negotiation-based trust establishment service, and a novel credential chain aware negotiation strategy, CRANE is proposed. Specially, a credential federation mechanism supports identity mapping and credential conversion is presented. All these approaches have been implemented in CROWN. To improve the performance and applicability of *ATNService*, we adopted two additional techniques, trust ticket and policy caching. Our experience also shows initial evidence that it is a viable solution to trust establishment in grid environments.

During the deployment and management of *ATNService*, it always needs professional administrators. Beside, the *ATNService* may become the point of single failure. We are attempting to use Virtual Machine technologies to encapsulate this service, and improve its manageability and reliability.

ACKNOWLEDGEMENT

This work is partially supported by grants from China 863 High-tech Program (No. 2007AA01Z426, 2007AA01Z120, 2007AA010301), China 973 Fundamental R&D Program (No. 2005CB321803) and National Natural Science Funds for Distinguished Young Scholar (No. 60525209). We would

also like to thank Yanmin Zhu for his helpful suggestions.

REFERENCES

Alfieri, R., R. Cecchini, et al. (2005). "From gridmap-file to VOMS: managing authorization in a Grid environment." Future Generation Computer Systems(FGCS), The International Journal of Grid Computing: Theory, Methods and Applications **21**(4): 549-558.

Basney, J., M. Humphrey, et al. (2005). "The MyProxy Online Credential Repository." Software: Practice and Experience **35**(9): 801-816.

Basney, J., W. Nejdl, et al. (2004). Negotiating Trust on the Grid. Proc. of 2nd Workshop on Semantics in P2P and Grid Computing at the Thirteenth International World Wide Web Conference, New York, USA.

Bertino, E., E. Ferrari, et al. (2004). "Trust-X:A peer to peer Framework for Trust Negotiations.".IEEE Trans on Knowledge and Data Engineeing **16**(7): 827-841.

Bonatti, P. A. and P. Samarati (2003). "A Uniform Famework for Regulating Access and Information Release on the Web." Journal of Computer Security **10** (3): 241-271.

Chadwick, D. W. and A. Otenko (2002). The PERMIS X.509 role based privilege management infrastructure Proceedings of the seventh ACM symposium on Access Control Models and Technologies(SACMAT), Monterey, California, USA ACM Press.

Constandache, I., D. Olmedilla, et al. (2005). Policy based dynamic negotiation for grid services authorization. Semantic Web Policy Workshop in conjunction with 4th International Semantic Web Conference, Galway, Ireland.

Dragoni, N. and A. Saidane (2008). A Framework for Dependable Trust Negotiation in Open Environments. Fifth IEEE Workshop on Engineering of Autonomic and Autonomous Systems, Belfast.

Foster, I., C. Kesselman, et al. (2001). "The Anatomy of the Grid: Enabling Scalable Virtual Organizations." International Journal of High Performance Computing Applications **15**(3): 200-222.

Herzberg, A., Y. Mass, et al. (2000). Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers. IEEE Symposium on Security and Privacy (S&P 2000), Oakland, CA,USA.

Huai, J., C. Hu, et al. (2007). "CROWN: A Service Grid Middleware with Trust Management Mechanism." Science in China Series F: Information Sciences **49**(6): 731-758.

Huai, J., H. Sun, et al. (2007). "ROST: Remote and hot service deployment with trustworthiness in CROWN Grid." Future Generation Computer Systems(FGCS), The International Journal of Grid Computing: Theory, Methods and Applications **23**(6): 825-835

Li, N., J. C. Mitchell, et al. (2002). Design of a Role-based Trust Management Framework. Proceedings of 2002 IEEE Symposium on Security and Privacy, Berkeley, California, IEEE Computer Society Press.

Li, N., W. H. Winsborough, et al. (2003). "Distributed credential chain discovery in trust management." Journal of Computer Security **11**(1): 35-86.

Pearlman, L., V. Welch, et al. (2001). A Community Authorization Service for Group Collaboration. the IEEE 3rd International Workshop on Policies for Distributed Systems and Networks, Monteray, California, U.S.A. .

Ryutov, T., L. Zhou, et al. (2005). Adaptive Trust Negotiation and Access Control for Grids. The 6th IEEE/ACM International Workshop on Grid Computing(Grid 05), Seattle, Washington, USA.

Sun, H., Y. Zhu, et al. (2005). Early Experience of Remote and Hot Service Deployment with Trustworthiness in CROWN Grid. Advanced Parallel Processing Technologies, 6th InternationalWorkshop,APPT 2005, Hong Kong, China, Springer.

Thompson, M. R., A. Essiari, et al. (2003). "Certificate-based authorization policy in a PKI environment." ACM Transactions on Information and System Security (TISSEC) **6**(4): 566-588.

Welch, V., T. Barton, et al. (2005). Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Collaboration. In 4th Annual PKI R&D Workshop.

Welch, V., F. Siebenlist, et al. (2003). Security for Grid services. Twelfth International Symposium on High Performance Distributed Computing (HPDC-12), , IEEE.

Winsborough, W. H. and N. Li (2002). Towards Practical Automated Trust Negotiation. Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02) Monterey, CA,USA, IEEE Computer Society, Washington, DC, USA

Winsborough, W. H., K. E. Seamons, et al. (2000). Automated Trust Negotiation. In DARPA Information Survivability Conference and Exposition, IEEE Press.

Winslett, M., T. Yu, et al. (2002). "The TrustBuilder Architecture for Trust Negotiation." IEEE Internet Computing **6(6)**: 30-37.

Yu, T. (2003). Automated Trust Establishment In Open Systems, UIUC. PhD Dissertation.

Yu, T. and M. Winslett (2003). A Unified Scheme for Resource Protection in Automated Trust Negotiation. IEEE Symposium on Security and Privacy, Berkeley, California.