Published in the Proceedings of the IASTED International Conference on Communications, Internet, and Information Technology, CIIT 2007, Banff, Alberta, Canada, June, 2007, pp575-122.

Intra-Federation Credential Negotiation Based on Individualized Release Strategy

Javed I. Khan
Kent State University
233 MSB, Kent, OH 44242 USA
javed@kent.edu

Kailas B. Bobade
Kent State University
252 MSB, Kent, OH 44242 USA
kbobade@kent.edu.

Manas Hardas
Kent State University
252 MSB, Kent, OH 44242 USA
Mhardas@kent.edu

ABSTRACT
In federated world like Shibboleth the release policy- how a member's personal information to be divulged is set by their home organization. The member has little say in it. In this paper we present an alternate framework where members can specify and personalize their own attribute release policy. Such personalization is however non-trivial. As opposed to simple request-reply based communication, such personalization inherently necessitates a mechanism of negotiation for which we present a new federated negotiation enabled framework.

KEY WORDS
Privacy, Federation, Negotiation, Authentication.

1. Introduction

A *Federation* is an association of *organizations* where organizations offer their members each other's services and thus members can enjoy limitless new services. To ensure seamless integration, federation establishes circle of trust so that member information issued by home organization, with which member is registered, is considered valid in rest of organizations. In a federation *service provider* organization provides resource to all federated members subject to *Access Control Policy (ACP)* [1, 3]. When a member needs access to the resources, *service provider* organizations retrieve necessary attributes of that member by querying member's home organization. One such open standard based system is the Shibboleth [4]. Using Shibboleth an organization could tap into already existing massive user-base of other organizations and federated members get benefit of ***single sign-on.*** In federation, *service provider* organizations receive member's attributes issued by her home organization using shibboleth system. So shibboleth has claimed to increase privacy. In a way it is true, but private person is actually absent from such system so does individualized privacy. Currently, both in Shibboleth and in private arrangements- the exchange policy is set by organizations- not by individuals. Individuals have very little knowledge- least any say, in how their information is being released by their home organization. However there are additional aspects of privacy. Alan Westin [5] has defined privacy as: "The right of individuals to determine for themselves when, how and to what extent information about them is communicated to others." There could be various scenarios where a home organization has to disclose its member's private information by her acquiescence. For this consider a federation of Universities and Companies.

In such federation students will apply for jobs in companies to schedule an interview. Companies need information like GPA, Transcript, and SSN etc from students. University could release student's information as per her acquiescence. But students often prefer to release same information like Transcript to companies under different conditions like company offering job in operating system or software engineering. Same could be true for the companies. To provide such facility, it is imperative to have Individualized Policies for federated members. Here, *Individualized policy* is a course of action created according to the specifications of an individual to determine information release decisions in context of service provider's offer.

Such Individualized Policies inherently triggers exchange of information between two or more parties. This process of requesting and releasing information to reach an agreeable objective is called as *negotiation* [2]. In this paper, we present a novel formalization of negotiation process and then discuss negotiation agent capable of negotiating with its peer in another organization both acting on behalf of the individual members.

2. Negotiation Process Model

In negotiation, every resource has a *requestor* (R), and an *owner* (O). Generally, negotiation starts with the release request of a target resource from the owner. Here, we define the states via which these resources are released in negotiation by the state diagram given in figure1.

In this model a released resource transits through the states namely *requested, offered*, and *accepted* during negotiation. When a party first requests a resource, it is placed into requested state. When the other party decides to release it, it is brought to the offered state. When the requesting party finally accepts, it is moved into accepted state. However, depending on the complexity of release policies a negotiation may transit through additional states namely *pending, not_available, denied, available, extraneous*, and *rejected*. If the second party's release

Published in the Proceedings of the IASTED International Conference on Communications, Internet, and Information Technology, CIIT 2007, Banff, Alberta, Canada, June, 2007, pp575-122.

policy requires some other resource before the requested resource can be released then the original request moves to pending state. If the second party does not have the requested resource then it is placed into not_available state. If it has but would like to deny, then it moves to denied state. There is also a state called available which is used in proxy negotiation where both parties move their releasable resources to *available* state until negotiation becomes successful. Once both parties are sure that successful negotiation is possible, they will exchange values of resources which are in *available* state. If any party finds some error with the value of released resource then the resource is *rejected* state.

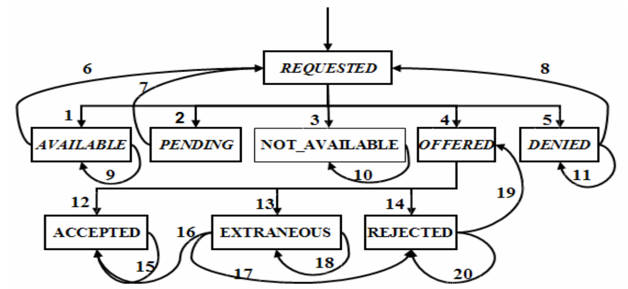


Figure 1: A) State Transition Diagram representing states of negotiation of an attribute. Numbers 1-20 indicates transition between states. B) States namely *Requested, Available, Pending, Offered* and *Denied* are considered for protocol and algorithm given in this paper. Protocol and algorithm are extensible for rest of the states.

We now explain the possible transitions by modeling a simple negotiation between Alice and Bob.

Alice's side: Alice starts negotiation by requesting resource 'A' from Bob. At this time, state of negotiation of resource 'A' is ***requested*** as shown in figure 1.

Bob's side: Once Bob receives request for 'A', as per release policy, he will bring state of negotiation of that attribute from *requested* to one of the five states namely *available*, *pending, not_available, offered,* and *denied* as explained next. 1) If access control policy of resource 'A' is satisfied, then state of negotiation transits from ***requested*** to ***offered*** as show by number 4. This transition means Bob will release actual resource immediately unlike transition from ***requested*** to ***available.*** 2) If release policy of resource 'A' is satisfied, but negotiation type is proxy then state of negotiation transits from ***requested*** to ***available*** as show by number 1. Here, Bob will not release actual resource until he makes sure that successful negotiation is possible. 3) If Bob requires a resource from Alice to release 'A', then state of negotiation transits from ***requested*** to ***pending*** shown by number 2. 4) If resource 'A' is not available with Bob, then state of negotiation transits from ***requested*** to ***not_available*** as shown by number 3. 5) If resource 'A' is available, but Bob does not want to release it, then the state of negotiation transits from ***requested*** to ***denied*** shown by number 5.

Alice's side: When Alice receives response from Bob, it will bring state of negotiation of 'A' to *accepted, extraneous*, *rejected* or *requested* state as explained next. 1) If this is the resource Alice wanted, then it will go to ***accepted*** state and negotiation will terminate. But if Alice has some alternative resource to request, then it will request new resource from Bob. 2) If Bob has offered something which Alice did not request and doesn't want, then state of negotiation transits from ***offered*** to ***extraneous*** state. 3) If there is an error in the resource format or credibility, then Alice will reject it and place it into ***rejected*** state.

3. Resource Exchange Protocol

This section provides a protocol for bi-partite negotiation which proceeds through number of steps between the parties. The core steps in a negotiation are session initiation, request, analysis of request and decision, counter request and decision at the other side, and so on, and at the end closure. An ongoing negotiation may or may not result in a deal. But the decision and analysis algorithm must continue until a resolution- deal or no-deal is achieved.

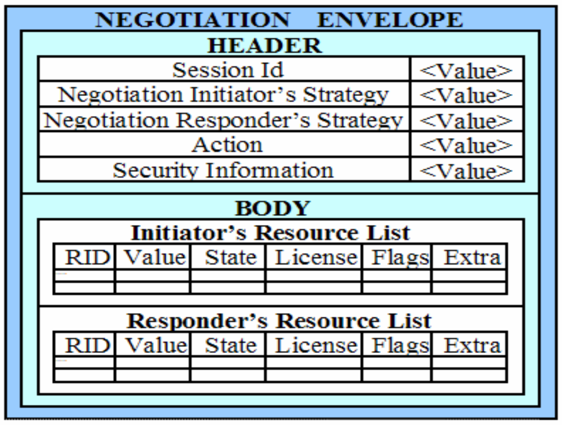


Figure 2: Negotiation Message

The core component of any protocol is (a) messaging structure and (b) resolution process. We have designed negotiation message to store all necessary state information about the resources so participants don't have to store ongoing negotiation information. The message, as shown in figure 2, has two main compartments namely header and body. Header carries session information and contains Action, Session Id, Initiator Strategy, Responder Strategy, and Security Information fields. Strategy fields carry negotiation strategy chosen by Negotiation Initiator and Negotiation Responder. This field is useful when two parties negotiate for negotiation strategy. Security Information carries encrypted identities of participant organizations, and members. Action field contains one of the messages namely Advertisement, Solicitation, Greeting, Strategy, Negotiation, Deal, No_Deal, Adieu, Error. Advertisement message includes target resources (R) in the message and then waits for the other party to reciprocate by sending solicitation (advertisement) message. Greeting message is used to invite other party for a negotiation session by offering initial identities and expect same from other party. Strategy message is used to decide negotiation strategy between two parties. Deal message indicates successful negotiation and No_Deal indicates unsuccessful negotiation. Adieu message indicates reporting about negotiation session. Error message indicates any unexpected error in negotiation.

Body has two parts namely Initiator's Resource List (IRL) and Responder's Resource List (RRL). Each item in these lists has the following fields. Resource Descriptor (RID) is the name of resource. This uniquely identifies the resource. The Value field contains the value or an URL to the value of the resource. Then there are three flags each one byte. TYPE, LOC and FORM. TYPE of a listed resource can have one of the following values Personal (P), Credential (C), Attribute (A) or Information (I). The LOC field tells where the credential value is available. It can be disclosed in the message (M), or in the private space of the Initiators (I), or of Responder (R). If the value of LOC flag is M then the value or location is disclosed in the VALUE field. If it is disclosed in the message, then the FORM field indicates if the VALUE field is Text (T) or URL (U). Finally, the License field contains the granted disclosure policy for the resource. State field contains current state of negotiation of a resource during negotiation process. Extra field carries state information about negotiation resolution process of every resource involved in negotiation, if necessary.

4. Negotiation Enabled Framework

This section presents an agent based negotiation enabled framework in federation. We consider a setup where every member in federation is owner of the resources such as files, attribute etc. Owners can specify their preferences and choices regarding their resource release policies.

Main components of framework are Policy Vault (PV), Resource Vault (RV), and Negotiation Agent (NA). Negotiation Agent has three subsystems Policy Resolver, Session Manager and Message Handler.

4.1 Resource Vault

Resource Vault is the cache system for the resources where each resource has a unique ID. We classify four kinds of resources mostly based on their source of authentication and privacy requirements. A resource can be a 1) *personal information* (P) (such as display-name, legal-name, email, social-security number) refers to the resources generated by owner and has strong privacy requirement, 2) *credentials* (C) (such as credit card, signed-certificate, transcript, credit-report) refers to the resources generated by third parties thus might be subject to additional verification, 3) *attributes* (A) (such as position title, age-group, medical-coverage) refers to the information vouched by the home organization, 4) *other-piece-of-information* (I) which can be released and exchanged subject to negotiation (such as news, weather, joke-of-the-day). All resources are owned by the members and treated in the same way in the negotiation process but particular piece of information may appear in different categories in different organizations.

4.2 Policy Vault

The *Policy Vault* stores the release policies for each of owner's resource. Release policy is expressed by a set of rules and a rule is represented in the disjunctive normal form (DNF) as R1 $\leftarrow$ $C_1 \vee C_2 \ldots C_j$, where, clauses $C_1 = R_1 \wedge R_2 \ldots\ldots R_K$, which means resource R_1 will be released when either of clauses C_1, C_2, or C_j is satisfied. Here, clause C_1 requires all resources R_1, R_2 and R_K from the other side.

Owner can specify two rules for each resource. Negotiation Agent will select appropriate rule based on whether owner is Negotiation Initiator (NI) or Negotiation Responder (NR). Owner can also specify License offered to requester of the resource. License decides whether requester can Cache (H), Store (S), and Forward (F) requested resource.

4.3 Negotiation System

Now, let's understand interaction of these components in brief. When negotiation begins, *Policy Resolver* will select policy to request and release attributes. These set of resources are send to *Session Manager* to add information like timestamp, information about opposite party etc. Session manager adds necessary session information and send that to Message Handler. *Message Handler* generates message from received resources and other information to send it to negotiation counterpart.

On opposite party's side, *Message Handler* accepts incoming message and disassembles it to get all attributes, session information, and sender's details. This information is then forwarded to *session Manger* for verification. Once verified, it forwards only received attributes to *Policy Resolver*. *Policy Resolver* chooses policy from *Policy Vault* and resolves all requested attributes, and selects releasable attributes from the *Resource Vault*. Resolution process also decides which attributes to request from opposite party. All releasable attributes along with opposite party's attribute from incoming message gets forwarded to *Session Manager.*

5. Stateless Eager Attribute Negotiation Algorithm (SEAN)

SEAN, shown in table 3, accepts rlist and P^R as input, where rlist is resources available in IRL and RRL. P^R is policy of Negotiation Responder assuming that algorithm is executing at Negotiation Responder's end. Same algorithm also executes at Negotiation Initiator's end. Each time it executes it returns one of the three states of negotiation namely DEAL, NO-DEAL, or NEGOTIATION i.e. negotiation is continuing. If the negotiation is continuing, algorithm also returns three things namely a set of newly *offered* resources (NEW_RELEASE), a resource (NEW_REQUEST) which has been counter requested, and name of a resource which is currently pending (NEW_PENDING). This is a

```
SEAN_state_machine(rlist, P^R) {
1. OLD_REQUEST = {A    rlist: A. state = REQUESTED}
2. OLD_RELSET = {A    rlist: A. state = OFFERED}
3. OLD_PENDING = {A    rlist: A .state = PENDING}
4. nego_state = CONTINUE
5. If (A ∈rlist: {A is service request} and {A    OLD_RELSET})
6. nego_state = DEAL
7. return ({Ø},{Ø},{Ø},nego_state)
8. Else  {
9.    If (A in OLD_PENDING )      /*For Initiator only*/
10.      NEW_REQUEST = A
11.   If (A in OLD_REQUEST)       /*For Responder only*/
12.      RELSET = SEAN_rule_resolver (A, rlist, RULE[ A])
13. NEW_PENDING ∈ (A    OLD_REQUEST: {RELSET= Ø})
14. NEW_RELSET = RELSET
15. If ((A is service request) and (A    NEW_RELEST))
16. For (each A in P^R )
17. RELSET = SEAN_rule_resolver (A, rlist, RULE[A])
18.      NEW_RELSET = (NEW_RELSET) U (RELSET)
19.      NEW_RELSET =(NEW_RELSET) U (OLD_RELSET)
20. If ((OLD_RELSET = = NEW_RELSET) & (This is not start
    of negotiation))
21.      nego_state = NO-DEAL
22.      return ({Ø},{Ø},{Ø},nego_state) }
23. Else
24.      return(NEW_REQUEST,NEW_RELSET,
25. NEW_PENDING, nego_state)
   }
}

SEAN_rule_resolver (A, rlist, Rule) {
1. If ( A can be resolved using rlist and Rule) {A    REESET}      2. Return (RELSET)    }
```

Table 1 : SEAN_state_machine routine which changes negotiation state of attributes and also generates state of negotiation i.e. DEAL, NO_DEAL or NEGOTIATION.

resource which has been requested and responder is currently resolving on its release.

5.1 Execution of Algorithm

Here, we explain three states namely DEAL, NO-DEAL, and NEGOTIATION.

1) When s*ervice provider* releases the requested resource, negotiation becomes successful i.e. DEAL (line 5-7). If OLD_RELSET, which contains all resources offered by the opposite party, contains the initial target then the state becomes DEAL and it returns (({Ø}, {Ø}, {Ø}, DEAL)).

2) On the other hand, NO-DEAL means neither of the parties has anything new to offer and requested resource is still in pending or requested state. At the end of current request (18-21), if there is anything new to offer then it is added to newly released resources set (NEW_RELSET). It is then compared with the old released set (OLD_RELSET). If these two sets are same, then negotiation state becomes NO-DEAL with return parameters (line 22) as (({Ø}, {Ø}, {Ø}, NO-DEAL)). But first request from Negotiation Initiator is exception to above mentioned rule.

In the first request, if Negotiation Initiator doesn't have anything to offer then OLD_RELSET and NEW_RELSET are same. But here negotiation is not necessarily unsuccessful as Negotiation Responder might offer her own resources and thus negotiation can proceed. For this exception line 20 has clause (this is not start of negotiation).

3) If state is neither DEAL nor NO-DEAL, then state becomes NEGOTIATION as explained below.

Requester, along with the request for target resource, offers new resources without any release policy. In this case algorithm checks if requested resource's policy can be satisfied with offered resources

Attribute	Alias	Value	Policy
Company_Name	(I_1)	ABC Inc	~
Job_Location	(I_2)	Xyz, OH	$(R_3 \wedge R_{10} \wedge R_2)$
Job_Title	(I_3)	Soft Engg.	$(R_2 \wedge R_{10})$
Job_Profile	(I_4)	Resp.html	$(R_9 \wedge R_2 \wedge R_{10})$
Salary	(I_5)	50k	$(R_2 \wedge R_7 \wedge R_{10})$
Benefits	(I_6)	Benef.htm	~
401	(I_7)	Yes	$(R_2 \wedge R_{10})$
Bonus	(I_8)	Bns.html	$(R_2 \wedge R_{10} \wedge R_3)$
Visa_sponorship	(I_9)	Yes	~
Start_Date	(I_{10})	MM/DD/YY	R_3

Table 2.1: Access Control Policy of Job Provider i.e. ABC Inc/John to schedule an Interview

Attribute	Alias	Value	Policy
Company_Name	(I_1)	KLM Inc	$(R_2 \wedge R_7 \wedge R_6)$
Job_Location	(I_2)	Xyz, OH	$(R_2 \wedge R_7 \wedge R_8)$
Job_Title	(I_3)	Soft Engg.	$(R_2 \wedge R_7) V (R_2)$
Job_Profile	(I_4)	Resp.html	$(R_2 \wedge R_7 \wedge R_9)$
Salary	(I_5)	50k	$(R_2 \wedge R_7 \wedge R_{10})$
Benefits	(I_6)	Benef.htm	~
401	(I_7)	10k	$(R_2 \wedge R_5)$
Bonus	(I_8)	Bns.html	$(R_2 \wedge R_9 \wedge R_3)$
Visa_sponorship	(I_9)	No	~
Start_Date	(I_{10})	MM/DD/YY	R_{11}

Table 2.2: Access Control Policy of Job Provider i.e. KLM Inc/Bob to schedule an Interview

Attribute	Alias	Value	Policy
Company_Name	(I_1)	CDE Inc	(R_2)
Job_Location	(I_2)	Xyz, OH	$(R_8 \wedge R_6 \wedge R_2)$
Job_Title	(I_3)	Soft Engg.	$(R_2 \wedge R_7)$
Job_Profile	(I_4)	Resp.html	$(R_2 \wedge R_5 \wedge R_8)$
Salary	(I_5)	50k	$(R_7 \wedge R_2 \wedge R_8)$
Benefits	(I_6)	Benef.htm	~
401	(I_7)	Yes	$(R_2 \wedge R_8 \wedge R_3)$
Bonus	(I_8)	Bns.html	$(R_2 \wedge R_3 \wedge R_8)$
Visa_sponorship	(I_9)	Yes	~
Start_Date	(I_{10})	MM/DD/YY	~

Table 2.3: Access Control Policy of Job Provider i.e. CDE Inc/Yang to schedule an Interview

Attribute	Alias	Value	Policy
Interview	(R_1)	Yes	$(I_1 \wedge I_6 \wedge I_5)$
Name	(R_2)	Alice	~
Email	(R_3)	alice@k.edu	$(I_1 \wedge I_3 \wedge I_{10})$
Cell_Phone	(R_4)	123123123	(I_4)
SSN	(R_5)	000000000	$(I_1 \wedge I_3 \wedge I_5)$
School	(R_6)	KSU	(I_3)
Major	(R_7)	Comp-Sci	~
Experience	(R_8)	Expr.html	$(I_1 \wedge I_{10})$
Transcripts	(R_9)	Tran.html	$(I_1 \wedge I_3 \wedge I_4)$
Publications	(R_{10})	Publ.html	~

Table 2.4: Access Control Policy of Job Seeker i.e. Alice to schedule an Interview

Attribute	Alias	Value	Policy
Interview	(R_1)	Yes	$(I_3 \wedge I_1 \wedge I_9)$
Name	(R_2)	Pooja	~
Email	(R_3)	pooja@k.edu	$(I_1 \wedge I_2)$
Cell_Phone	(R_4)	123123123	(I_4)
SSN	(R_5)	000000000	$(I_1 \wedge I_2 \wedge I_5)$
School	(R_6)	KSU	~
Major	(R_7)	Comp-Sci	$(I_4) V (I_3)$
Experience	(R_8)	Expr.html	$(I_5 \wedge I_4)$
Transcripts	(R_9)	Tran.html	$(I_1 \wedge I_3 \wedge I_4)$
Publications	(R_{10})	Publ.html	(I_1)

Table 2.5: Access Control Policy of Job Seeker i.e. Pooja to schedule an Interview

Attribute	Alias	Value	Policy
Interview	(R_1)	Yes	$(I_2 \wedge I_1 \wedge I_{10})$
Name	(R_2)	Sajid	~
Email	(R_3)	sajid@k.edu	$(I_1 \wedge I_3)$
Cell_Phone	(R_4)	123123123	(I_4)
SSN	(R_5)	000000000	$(I_1 \wedge I_2 \wedge I_5)$
School	(R_6)	KSU	(I_1)
Major	(R_7)	Comp-Sci	$(I_1 \wedge I_{11})$
Experience	(R_8)	Expr.html	$(I_5 \wedge I_6) V (I_6)$
Transcripts	(R_9)	Tran.html	$(I_1 \wedge I_3 \wedge I_4)$
Publications	(R_{10})	Publ.html	~

Table 2.6: Access Control Policy of Job Seeker i.e. Sajid to schedule an Interview

A) If policy is satisfied, then resource owner will offer requested resource (11-14). Line 11 checks if 'A' is requested resources and line 12 passes it to SEAN_rule_resolver routine. Line 1 in SEAN_rule_resolver routine determines if 'A' can be offered and at line 3, it is returned using in RELSET. If 'A' is offered, then it becomes part of NEW_RELSET shown by line number 14. In this case, NEW_PENDING will be empty at line 13 because 'A' will either go in offer or pending state from requested. If 'A' is the requested resource, then algorithm won't release rest of the resources and this is done by line number 15.

B) If policy isn't satisfied, then resource owner will offer her unlocked resources to requester and in turn requester will offer her unlocked resources. This way negotiation proceeds (11-18). In this case, at line 13 NEW_PENDING will contain requested resource because its policy is not satisfied. So at line 14, NEW_RELSET would be empty. Line 16-18 will release all possible resources of responder by going through routine SEAN_rule_resolver. These released resources are returned at line number 24. At this time two sets (OLD_RELSET and NEW_RELSET) are not same.

In eager strategy only Negotiation Initiator requests resources so lines 9-10 will handle pending requests for Negotiation Initiator. While only Negotiation Responder has to handle incoming requests so line 11-14 are exclusively for Negotiation Responder.

6. Example of Negotiation

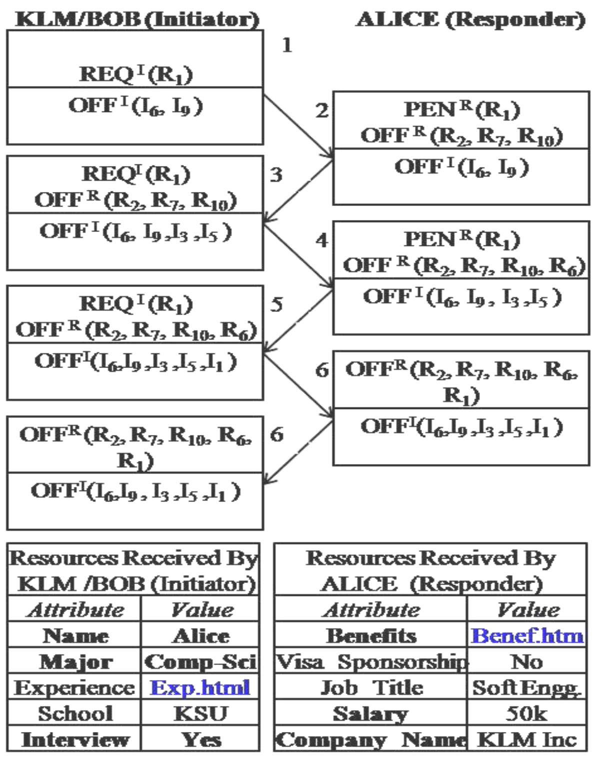
We consider a job seeker-hunter scenario in a complex federated setup consisting of group of Universities and Companies. Students store their individualized policies with their universities and various managers in the companies set up their requirements to search potential employees. In this setup, we consider three students Alice, Pooja and Sajid with home institution KSU and three hiring managers Bob, John, and Yang from KLM Inc, ABC Inc, and DEF Inc respectively with release policies in Table 2.1 to 2.6. Here, we will discuss one negotiation in detail along with negotiation statistics.

Each message carries state information about the resources in an ongoing negotiation. In our notation we group resources against their respective current states in a negotiation as $STATE^P$ ({R_1:V_1}, {R_2:V_2}… (Rn: Vn}). STATE can be the states of a resources specified in figure 1 such as REQ, AVL, PEN, DEN, OFFERED etc. Each argument is a pair where Rn is the ID of the resource and Vn is the special information about the resource. The superscript p represents the party (negotiation initiator (I) or negotiation responder (R)) involved in the latest update of a state. For stateless eager strategy V is always empty. Top portion and bottom portion of each message corresponds to IRL and RRL as shown in figure-2.

6.1 Negotiation to find potential candidate

Negotiation Agent of Bob's company initiates a search for potential candidates and starts negotiation with Alice's agent to schedule an interview as shown in figure 3 A).

Bob's and Alice's access control policies are shown in Table 2.2 and 2.4 respectively. Bob's agent will start negotiation by requesting *Interview* (i.e. REQ^I (R_1)) from Alice's agent and as per eager strategy, Bob's agent will offer attributes (i.e. OFF^I (I_6, I_9)) as shown in message 1.


KLM/BOB (Initiator)
ALICE (Responder)
1
$REQ^I(R_1)$
$OFF^I(I_6, I_9)$
2
$PEN^R(R_1)$
$OFF^R(R_2, R_7, R_{10})$
$OFF^I(I_6, I_9)$
3
$REQ^I(R_1)$
$OFF^R(R_2, R_7, R_{10})$
$OFF^I(I_6, I_9, I_3, I_5)$
4
$PEN^R(R_1)$
$OFF^R(R_2, R_7, R_{10}, R_6)$
$OFF^I(I_6, I_9, I_3, I_5)$
5
$REQ^I(R_1)$
$OFF^R(R_2, R_7, R_{10}, R_6)$
$OFF^I(I_6, I_9, I_3, I_5, I_1)$
6
$OFF^R(R_2, R_7, R_{10}, R_6, R_1)$
$OFF^I(I_6, I_9, I_3, I_5, I_1)$
6
$OFF^R(R_2, R_7, R_{10}, R_6, R_1)$
$OFF^I(I_6, I_9, I_3, I_5, I_1)$

Resources Received By KLM /BOB (Initiator)	
Attribute	*Value*
Name	Alice
Major	Comp-Sci
Experience	Exp.html
School	KSU
Interview	Yes

Resources Received By ALICE (Responder)	
Attribute	*Value*
Benefits	Benef.htm
Visa Sponsorship	No
Job Title	SoftEngg.
Salary	50k
Company Name	KLM Inc

Figure 3: A) Messages exchanged during negotiation between Bob and Alice to schedule an Interview using Eager strategy. Messages are identified by numbers from 1-6. B) At the end of Negotiation, Agent will generate list of resources received for KLM/Bob. C) At the end of Negotiation, Agent will generate list of resources received for Alice.

Once Alice's agent receives message 1, according requested attribute's i.e. *interview* disclosure rule, it will change state of negotiation of *Interview* to pending (i.e. PEN^R (R_1)). Agent also offers attributes (i.e. OFF^R (R_2, R_7, R_{10})), whose disclosure rule is satisfied by Bob's offered attributes (i.e. I_6 and I_9), as shown in message 2. Here notice that, Alice's agent keep Bob's offered attributes as it is from message 1 to message 2. This way negotiation proceeds and at the end of negotiation, agent stores negotiation information in terms of resources released and received as shown in figure 3 B) and 3 C).

6.1.1 Resources Received in Negotiation

At the end of negotiation, agent will generate list of resources received and released for its user. Because of space constraint, we show resources received only by Alice, Pooja, and Sajid in Tables 3.1-3.3.

	RESOURCE DESCRIPTOR	RESOURCE VALUE
KLM/ BOB	**Benefits**	Benef.htm
	Visa Sponsorship	No
	Job Title	Soft Engg.
	Salary	50k
	Company Name	KLM Inc
ABC/ JOHN	**Company Name**	ABC Inc
	Benefits	Yes
	Visa Sponsorship	Yes
	Job Title	Soft Engg.
	Salary	50k
	401	Yes
CDE/ YANG	**Benefits**	Yes
	Visa Sponsorship	Yes
	Start Date	MM/DD/YY
	Company Name	CDE Inc
	Job Title	Soft Engg.
	Job Location	XYZ, PA
	Salary	50k
	Bonus	Bns.html

Table-3.1: It shows resources received by Alice at the end of negotiation from three managers namely Bob, John, and Yang. Alice has explicitly requested **Company_Name**, **Salary** and **Benefits** from companies.

	RESOURCE DESCRIPTOR	RESOURCE VALUE
KLM/ BOB	Benefits	Yes
	Visa_Sponsorship	Yes
	Job_Title	Soft Engg.
	Company_Name	KLM Inc
ABC/ JOHN	**Company_Name**	ABC Inc
	Benefits	Yes
	Visa_Sponsorship	Yes
	Job_Title	Soft Engg
	401	Yes
CDE/ YANG	Benefits	Yes
	Visa_Sponsorship	Yes
	Start_Date	MM/DD/YY
	Company_Name	CDE Inc

Table-3.2: It shows resources received by Pooja at the end of negotiation from three managers namely Bob, John, and Yang. Pooja has explicitly requested **Company_Name**, **Job_Title** and **Visa Sponsorship** from companies.

	RESOURCE DESCRIPTOR	RESOURCE VALUE
KLM/ BOB	Benefits	Benef.htm
	Visa_Sponsorship	No
	Job_Title	Soft Engg.
ABC/ JOHN	**Company Name**	ABC Inc
	Benefits	Yes
	Visa_Sponsorship	Yes
	Job_Title	Soft Engg.
	Salary	50k
	401	Yes
	Job Location	XYZ, CA
	Start Date	MM/DD/YY
CDE/ YANG	Benefits	Benef.htm
	Visa_Sponsorship	Yes
	Start Date	MM/DD/YY
	Company Name	CDE Inc
	Job Location	XYZ , PA

Table-3.3: It shows resources received by Sajid at the end of negotiation from three managers namely Bob, John, and Yang. Sajid has explicitly requested **Company_Name**, **Job_Location** and **Start_Date** from companies.

Negotiating Parties	Negotiation Result	Number of Rules fired in a Negotiation	Number of Messages exchanged in a Negotiation	Number of resources released by both parties		Number of attributes released in a Negotiation
				Necessary for optimum negotiation	Not necessary for optimum negotiation	
ABC-Alice	Success	10	4	7	3	10
ABC-Pooja	Success	9	4	6	3	9
ABC-Sajid	Success	14	6	8	6	14
CDE-Alice	Success	15	6	8	7	15
CDE-Pooja	Fail	7	4	——	—	7
CDE-Sajid	Success	9	4	7	3	10
KLM-Alice	Success	10	6	7	3	10
KLM-Pooja	Success	8	6	7	1	8
KLM-Sajid	Fail	4	4	——	——	6

Table 4 : Statistic of negotiations between three students and three managers using SEAN

6.2 Negotiation Statistic

Table-4 shows statistic based on parameters such as number of messages exchanged, number of rules fired, number of resources requested and released in nine negotiations. Number of messages exchanged, rules fired, and resourced released are counted at the end of successful or unsuccessful negotiation.

7. Conclusion

We have contributed a novel negotiation enabled framework to preserve privacy of federated members and designed protocol as well as Stateless Eager Attribute Negotiation Algorithm. Using SEAN both parties receive each other's attributes - an integral part of negotiation.

The privacy implications of this work are quite interesting. With the systems like Shibboleth, a member is no longer at the mercy of the s*ervice provider* organization about the disclosure policy, but is at the mercy of the on-size-fits-all release policy set by the home organization. Through this work the individual's privacy is further enhanced. A member is now not at the mercy of home organization which acts more as a negotiation agent on behalf of the member- than as her policy setter.

8. References

[1] Kathy Bohrer, Stephen Levy. Individualized Privacy Policy Based Access Control. In *Proceedings 6th International Conference on Electronic Commerce Research (ICECR-6)* October 2003, Dallas, Texas, USA.

[2] W. H. Winsborough, K. E. Seamons, and V. E. Jones. Automated Trust Negotiation. *DARPA Information Survivability Conference and Exposition*, Hilton Head, January 2000.

[3] D.F. Ferraiolo, J. Cugini, D.R. Kuhn. Role Based Access Control: Features and Motivations. *Computer Security Applications Conference* (1995).

[4] Shibboleth. http://shibboleth.internet2.edu/

[5] Alan Westin "Privacy and Freedom" 1970