Secure Web service discovery: overcoming challenges of ubiquitous computing

Slim Trabelsi
Institut Eurecom
Slim.Trabelsi@eurecom.fr

Jean-Christphe Pazzaglia
SAP Labs France
Jean-Christophe.Pazzaglia@sap.com

Yves Roudier
Institut Eurecom
Yves.Roudier@eurecom.fr

***Abstract*—** *Dynamic and self-organizing systems like those found in ubiquitous computing or semantic web based scenarios raise numerous challenges regarding trust and privacy. Service discovery is a basic feature of SOA deployment in such systems, given that entities need to locate services they can describe but that they do not necessarily know. PKI based solutions to securing this mechanism, which require a preliminary key distribution, are therefore rendered awkward and contrived. In contrast, the new concept of Attribute Based Encryption, derived from Identity Based Encryption schemes, makes it possible to create secret communication channels with unknown services based solely on some attributes that are part of their description and in a decentralized fashion, that is, without the introduction of any additional trusted third party like a registry. This paper discusses how such a scalable solution to enabling secure and decentralized discovery protocols can be implemented and put to use. After reviewing the security properties that are expected, the paper then goes on to detail how to extend the WS-Discovery Web Service protocol with such mechanisms. Preliminary experimental results based on an implementation of this extended protocol are finally presented.*

***Index Terms*—Secure Service discovery, Attribute Based Encryption, WS-Discovery, Web Services, Service Oriented Architectures, Ubiquitous Computing.**

1. INTRODUCTION

Discovery mechanisms have long been recognized as a fundamental need in distributed architectures. This can readily be seen with the deployment of the DNS system on the Internet for instance. The development of mobile devices, local and personal area networks, and home automation has prompted the availability of nearby device discovery protocols and mechanisms in the recent years. The Bluetooth discovery protocol or UPnP are good instances of this trend. The advent of ubiquitous computing is today blurring traditional boundaries illustrated by both approaches to discovery. This new computing paradigm, which poses context-awareness at the central design issue, does not draw a real separation between devices in a local area network and remote servers. Instead, Service Oriented Architectures (SOA) are used as a unifying concept for both types of resources, as can be seen with Jini [5] or Web Services and their extensions.

SOA was originally (in particular as envisioned in Jini) intended to enable access to applications running on nearby devices in a dynamic fashion. This programming style promotes the use of loosely coupled and highly interoperable applications to overstep the limitations of traditional distributed component solutions (like CORBA [16]). A *Service*, the building block of SOA solutions, encapsulates a set of related business functions within a container and gives access to these functions through standardized interfaces. Orchestration techniques were later developed in order to enable the dynamic composition of basic and interchangeable services, thus providing the basis for more complex interactions.

Discovery therefore becomes of strategic importance in SOA stacks and this importance is growing proportionally with the dynamic organization of the environment: while a typical intranet implementation may rely on a basic discovery strategy (e.g. naming service in CORBA) or may even not strictly require it (e.g. predefined set of known services), Internet-wide and above all ubiquitous computing applications face a set of challenges with respect to discovery. In such applications, the discovery strategy should cope with the heterogeneity of services and platforms from a technical perspective (e.g. take into account bandwidth, energy savings …), with the complex semantics of service descriptions (e.g. resorting to terminology- or ontology-based descriptions), with the scalability of the solution (from a few PDAs to several thousand sensors and servers).

Since the emergence of SOA based programming, different works (UDDI [2], WS-Discovery [3], OWL-S [4]) have aimed at solving many of these discovery issues. Still, only very few of them put security and trust as primary concerns, even though these issues are essential to the successful deployment of ubiquitous computing solutions: for instance, services should prevent critical information that may make them vulnerable to attackers from being disclosed inconsiderately; a user should also protect his privacy and make sure that personal information do not get disclosed to rogue entities when he performs a service lookup. This paper aims at contributing to the security of discovery protocols through the protection of their lookup and response messages with identity-based encryption techniques. Section 2 first lists the

security objectives of this work. The principles of our proposal for securing service discovery are detailed in Section 3. The extension of an existent service discovery protocol, WS-Discovery, is then discussed in Section 4, where the introduction of security mechanisms is also evaluated. This section also presents preliminary results of a prototype implementation. We finally compare the approach presented in this paper with related work and draft future directions for investigation.

2. SECURITY REQUIREMENTS OF SERVICE DISCOVERY

One distinguishes the service requester from the service provider as the main players in discovery protocols; still some protocols feature a registry based service discovery, but this paper does not address this type of discovery mechanism. Discovery is very often performed at the initiative of the service requester (e.g. lookup model) but can also be initiated by the service provider (e.g. advert model). The specificity of discovery is that these players, who are likely from different administrative domains, are by definition initially unaware of their respective existence and security policies. The following requirements make it necessary to answer relatively original threats in this context:

- Client and service authentication: the very objective of service discovery is to communicate with previously unknown entities that provide specific functionalities. Open discovery services therefore require that the first message sent (lookup or advert) be in clear, also meaning that the content of the message can be accessed. Without the means to authenticate clients and servers, service discovery makes the implementation of a man-in-the-middle attack possible [15], a malicious entity being able to wrongly answer a discovery message. Registry based discovery schemes obviously make it much simpler than infrastructure-less ones to perform secure discoveries, since the registry is the only element which the client needs to identify and which the client should be identified to, yet at the price of additional infrastructure deployment requirements. This paper instead concentrates on peer-to-peer discovery.
- Privacy: the discovery initiator takes a more important risk than the other party since it does not control which entities will receive the discovery message, nor the potential usage of the information embedded in his request message. The information disclosed by client requests is likely to reveal a subset of the intentions of the service requester. An attacker may try to gather profiles of users of the service discovery mechanism based on the information carried by discovery messages as well as subsequent messages generated by the actual access to the service and expose some more information (host name, Certificate, Credentials …). The correlation of such data with discovery related information is particularly worrying from a privacy protection perspective.
- Access control: since client/service authentication is problematic in the initial discovery phase, traditional service oriented architectures do not support access control during discovery. Service providers would ideally advertise their services exclusively to authorized users, even though this objective is in practice difficult to achieve in a pervasive environment. Still, disclosing the description of a service to any requester potentially increases the risk that a malicious client or malware take advantage of this knowledge and of service vulnerabilities to gain an unauthorized access.
- Availability: denial of service (DoS) is an attack to the availability of resources preventing the authorized access to a system resource or delaying system operations and functions. Openly exposing service descriptions during discovery enables attackers to exploit vulnerabilities by creating specially crafted messages for the server or by the registry. Notably, registries clearly constitute a single point of failure and therefore are particularly sensitive to brute force DoS attacks. Peer-to-peer discovery on the contrary is expected to increase the availability of services on the whole.

The main mechanism proposed in this paper to answer a part of these requirements is the protection of confidential and private information exchanged during the discovery process through encryption. This especially makes it possible for the server to restrict the discovery of its profile to a specific user group, and for the user to let details of his discovery requests visible only to a restricted number of servers.

3. SECURING SERVICE DISCOVERY – PRINCIPLE

3.1. Profiles and attributes

In a SOA architecture, every entity (client/service) exposes some information about itself through one or more profiles. This information is useful for the users to distinguish between the different entities of the system. Just like an identity card contains particular characteristics of its owner like his name, his age, his home address, profiles characterize an entity through the enumeration of attributes.

Profiles can be used by services in order to announce themselves and can be published in a public repository accessible to all users. During a service discovery process, the server publishes its service profile (service description). The attributes contained in this profile can be useful for the user to select the service he wants to contact.

Depending on the technology used for the service deployment, profiles will take different forms: service description in the Web Service framework may for instance take either the form of a WSDL [5] profile consisting in an XML-based file, or of a DAML [6] profile made of an OWL-S based semantic web description. In WS-Discovery, the service profile is composed of two strings (Type and Scope). Similarly in CORBA, the description of the service is limited

to a name within a context; this name is contained in a naming graph where each name is associated with a reference to the service. In Jini, the service registers its serialized proxy object together with a set of relevant attributes (Entry) that may be later use during discovery. The client profile can be described in a certificate that contains some indications about his identity and public key (X.509 Certificates) and also some other attributes like the roles, the rights, or the delegations (Eureca [7], X.509 Attribute certificates).

As can be seen, all the attributes related to the description of users and services can be used to distinguish between the different entities involved in the system and also to improve the knowledge about the surrounding environment.

3.2. Security Mechanism Overview

Identity based encryption (IBE) [21],[8] makes it possible for a user to encrypt a message or a document using the identity of its intended recipient as a public encryption key, and without the need for the public key certificate of that recipient. In a variant of this technique, one defines an identity as a set of attributes instead of the name of an entity. Assuming each attribute has precise semantics and that all communicating parties share a common understanding regarding these attributes and their values, the identity based encryption mechanism can be used in an "attribute-based encryption" (ABE) fashion [13],[22].

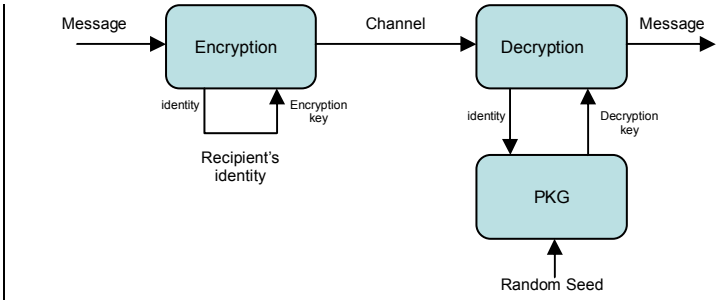

Figure 1: Identity Based Cryptosystem [21]

In the ABE approach, an entity can encrypt a message or a document for one or more entities endowed with the same characteristics that are captured by a certain set of common attributes. In order to decrypt such message, the recipient has to use private key corresponding to its attributes (identity for IBE). This key can be generated by a Public Key Generator (PKG) for entities that can prove the authenticity of their attributes (by providing an attribute Certificate like Eureca).

In the service Discovery process, the client's lookup request contains a set of properties (considered as attributes) that must be satisfied by the requested service. These properties are usually related with the service functionality (like the type, service provider, input or output parameters…), yet non-functional requirements such as a certification, recommendations, or the membership to some trust domains may be also expressed in the lookup request. Assuming that a user encrypts a message using some of these properties as public key attributes, decryption depends on the ownership of the correct credentials. For example, with this scheme and assuming that a client is looking for a printer service belonging to the Eurecom domain, the client will encrypt his message with the public key associated with string `{printer | Eurecom}`, which encodes type and scope attributes. The client will then broadcast this message to all surrounding services. Listening to the broadcast channel will not be sufficient to interpret the lookup request and only services that have the right attributes and therefore the associated private keys will be able to read the content of the message. This solution makes it possible to perform the discovery stage in a secure manner: the client performs a service discovery by broadcasting its request; at the same time, he imposes an implicit access control to the content of his request for printer services belonging to the Eurecom domain.

3.3. A Secure Discovery Use Case

This section introduces a scenario that illustrates the requirement for attribute based encryption between entities involved. Let us suppose that an airline company offers wireless services during flights (news, e-mail, movies, duty-free shopping …). Depending on the class of his seat, each passenger will have different access privileges to these services.

The shopping service would be accessible to all passengers without any restriction. Every passenger sending a service discovery request containing the { shopping } keyword with a laptop will receive a response containing the details of where and how to access the digital shopping mall (Figure 2). Of course, it is assumed that this service location needs not be protected from other passengers since it is publicly accessible. This does not preclude subsequent requirements for access control to the duty-free shopping service, for instance because one has to check the validity of the passenger's credit card number before agreeing to some transaction.

Figure 2: Discovering shopping service in insecure mode

Passengers in business and first class may also get access to their e-mail. The response sent by the e-mail service will need to ABE-encrypted in order to restrict its access to business and first class passengers only (Figure 3). The response may contain credentials in order to enable the passengers to access the service. All other passengers should be unable to locate the service, much less gain access to it.

Figure 3: Discovering restricted e-mail services

Passengers in first class may also request a *premium movie* service that lets them access to recent movies. The premium service should only be accessible in presence of an adult passenger, in particular in order to protect children against offensive or violent contents. A service discovery request for the premium movie should thus contain the requesting passenger's age for instance, yet such personal information should remain as confidential as possible. Encrypting the service discovery request to render it accessible by the premium service only would be enough to protect the passenger's privacy. However, to cope with requirements regarding access control to the service description, the discovery response will also need to be encrypted according to requester's age, so that the location of the movie service will be known only to adult passengers traveling in first class (Figure 4).

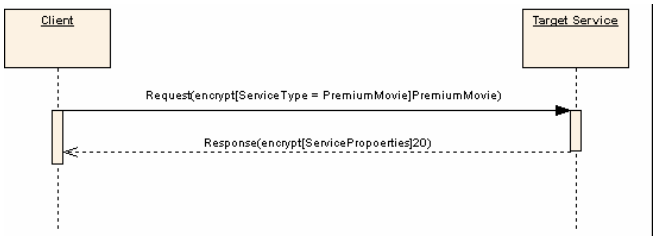

Figure 4: Discovering restricted movie service with privacy protection

In all these examples, the assurance that some trusted authority did grant attributes to a service according to an agreed upon taxonomy is all a passenger needs to protect the privacy of his lookup messages. The same holds true with respect to the granting of attributes to passengers. In all these scenarios, identities (or attributes) therefore are central to the referencing of services or passengers.

4. EXTENDING THE WS-DISCOVERY PROTOCOL WITH ATTRIBUTE BASED ENCRYPTION

4.1. WS-Discovery

Web Services Dynamic Discovery (WS-Discovery [3]) is a technical specification that defines a multicast discovery protocol to locate services connected to a network. Each service provider announces itself (by sending a "Hello" message) through the multicast group to expose the services that can provide. Each user that is looking for a service propagates its query (by sending a "Probe" message) through the multicast and only the concerned service must make a response (by sending a "Probe Match" message). As we mentioned previously the default matched attributes are the Type and the Scope of the service, obviously other attributes and meta-data information can also be added.

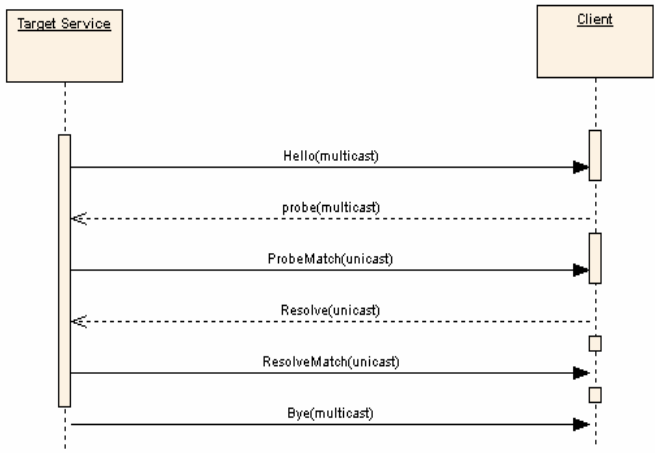

Figure 5: WS-Discovery protocol message sequence

Because the WS-Discovery protocol is based on multicast, the discovery scope may be restricted to local subnets. For this reason and to scale to a large number of endpoints, the specification defines multicast suppression behavior if a discovery proxy (DP) is available on the network. By listening for these announcements, clients detect discovery proxies (that are assimilated to the other services) and switch to use a discovery proxy-specific protocol. However, if a discovery proxy is unresponsive, clients revert to use the standard protocol. These discovery proxies can communicate between each others in order to extend the discovery scope to the others subnets. This feature enables to migrate smoothly between carefully-managed and ad hoc networks.

The WS-Discovery specification does not suggests securing the discovery process but it recommends the usage of a compact signature format to secure the exchanged messages. In this case each entity has the possibility to verify the signature of the message sender. This signature protects against the message modifications the replay, the spoofing.

Signature verification is obviously insufficient to protect users (servers and clients) since a valid signature only assess that the message content has not been altered without presuming of the level of trust of the issuer. Moreover, the content of the message is not confidential and there is no guarantee against the disclosure of private information. For example a malicious server can publish fake services with a valid signature or listen to request messages in order to collect valuable information.

4.2. Applying the identity based encryption

The goal of our solution is to protect the sensitive information contained within the WS-Discovery messages. To reach this objective, we applied ABE-mechanism to the principal messages exchanged during the discovery phase. The attributes used to encrypt the data are precisely the attributes enabling to describe a service:

- ***Type***: An identifier of the service endpoint (logical name describing the capability of the service. Ex: Printer, TV …)

- ***Scope***: An extensibility point that may be used to organize the services into logical groups (Ex: for the printer service the scope could be Color, Black & White …)

Assuming now that these two attributes identify the service, they can then be used to protect the client's probe messages by encrypting other attributes of the message[1]:

```
<s:Header ... >
      <a:Action ... >
      http://schemas.xmlsoap.org/ws/2
004/10/discovery/Probe
      </a:Action>
      <a:MessageID>
         xs:anyURI
      </a:MessageID>
      <a:ReplyTo>
         endpoint-reference
      </a:ReplyTo>
      <a:To>
         xs:anyURI
      </a:To>
      ...
</s:Header ... >
<s:Body ... >
      <d:Probe ... >
            <d:Types>
                Printer
            </d:Types>
            <d:Scopes >
                Eurecom
           </d:Scopes >

            ...
      </d:Probe>
  </s:Body>
```

Figure 6: Probe Message

In this example, the ***Probe*** message (Figure 6) content could be self-encrypted using the attributes (Ex: $Encrypt[Attribute]_{Attribute}$) in order to hide the mandatory services' attributes requested by the user (the requested service type and the EndpointReference of the requester). This guarantee that only the service that holds the private keys corresponding to these attributes are able to decrypt and process the Probe message. Of course these private keys should be provided by a trusted PKG only to trusted services. The PKG should therefore verify the credentials exhibited by the service, this can done using existing PKI infrastructures and a specific X509v3 profile. The profile should be tuned to capture the attributes describing the service.

Now let's focus on the service's response, the ***ProbeMatch*** message (Figure 7) that must also be protected especially since the content of the message provides a set of attributes offering a precise description of the service (location, address, URI).

```
[<d:ProbeMatch ... >
      <a:EndpointReference>
            <a:Address>
            uuid:98190dc2-0890-4ef8-
   ac9a-5940995e6119
     </a:Address>
      <a:EndpointReference>
      <d:Types>
            Printer
      </d:Types>
      <d:Scopes>
            Eurecom
      </d:Scopes>
      <d:XAddrs>
         http://printer.eurecom.fr/
      </d:XAddrs>
      <d:MetadataVersion>
                  75965
      </d:MetadataVersion>
...
</d:ProbeMatch>]
```

Figure 7: Probe Match Message

All these attributes could be encrypted using a unique identifier of the user that requested the service. In order to avoid carrying extra information, the `endpoint-reference` information contained in the ReplyTo tag (Figure 7) of the Probe message's header, can be used as the identifier of the user:

In this case the encryption action will be notated $Encrypt[\Pr obeMatch]_{endpoint-referece}$ and only the owner of this `endpoint-reference` that holds the appropriate private key corresponding to this identifier is able to decrypt the Probe Match message. As described previously, this private key can be provided by a PKG relying on existing PKI infrastructure.

[1] In the messages, the **s** namespace refers to soap (http://www.w3.org/2003/05/soap-envelope) , the **d** namespace refers to WS-Discovery (http://schemas.xmlsoap.org/ws/2005/04/discovery/) and **d** namespace refers to WS-Addressing (http://schemas.xmlsoap.org/ws/2004/08/addressing).

4.3. Experiments and preliminary results

In order to evaluate the efficiency of this new secure service discovery model, we developed a Java implementation of the WS-Discovery protocol combined with Voltage IBE toolkit [19]. The Voltage IBE toolkit (C library) provides a high level interface for an easy integration with any application.

The XML message generation and processing functions of the WS-Discovery protocol was modified as follows:

- During the Probe message generation the clear text `String` contained in the `Type` tag is replaced by the encrypted text (Figure 8), then the protected probe message is broadcasted to the entire multicast group. In this example, the type Printer is encrypted.

```
<s:Body>
      <d:Probe>
        <d:Types>
          (Encrypt[Printer]{Printer|Eurecom})
        </d:Types>
      </d:Probe>
  </s:Body>
```

Figure 8: Encrypted Probe Message

- All services listening to the multicast group will receive the message but only the intended recipients will be able to decrypt the content of the `Type` tag in order to process the received message. The encrypted text will be extracted from the `Type` tag by using the secret key corresponding to the appropriate type of service (Figure 9), thus making it possible to retrieve the clear text (Printer).

```
<d:ProbeMatch>
      <a:EndpointReference>
            <a:Address>
                  uuid:dc1c483f-8bc8-
48b9-9e34-e6546645c2ec
            </a:Address>
      </a:EndpointReference>
            <d:Types>
                  Printer
            </d:Types>
            <d:MetadataVersion>
                  1
            </d:MetadataVersion>
```

Figure 9: Clear Probe Massage

Our early experiments show that the application of the IBE mechanisms to the WS-Discovery protocol adds a negligible extra processing time:

- workstation specifications:
  - Virtual Machine: VMware 1.0.1
  - OS: Fedora Core 5 with a Linux 2.6.x kernel i686
  - CPU: Mobile Intel ® Pentium ® 4 CPU 1.70 GH
  - Physical memory 512 MB
- System setup: generate the system parameters milliseconds + the time generate the private key related to the Printer type = 1370 milliseconds
- Encrypt the Printer text (client part): 1090 milliseconds
- Decrypt the Printer text (server part): 790 milliseconds

If we suppose that the system setup step can be avoided (each has a local storage for the system parameter and the private key) the additional extra-time generated by the IBE application is approximately equal to 1880 milliseconds (encryption + decryption). This value does not affect the usual sequencing of the WS-Discovery message exchange protocol. This extra time is not problematic due the fact that WS-Discovery protocol uses the UDP messaging format imposing a retransmission delay of 5 seconds in order to ensure the good reception of a message in spite of some packet loss risks. This delay covers the extra time generated by the decryption process.

We are actually working on a 100% Java solution by integrating the Identity Based Encryption JCE Provider [9] with our Java WS-Discovery Protocol implementation.

4.4. Restrictions on attributes Typing

The attribute encoding is the fundamental mechanism for selecting the potential recipients of a lookup message or response. The attribute value used should be precise enough to restrict the broadcast of the message borne information and not too restrictive at the same time in order to limit performance issues. Indeed, performing a lookup on a single attribute means that the recipient will have to try and decrypt the chosen field (e.g. Type of the service) with every attribute value that he may have registered with the PKG (nothing prevents a service from being registered under several types for instance, e.g. printer, color-printer, etc.). Every such different value for a private key will for instance increase the response time of the server in the lookup discovery model in a linear fashion. Performing a lookup on a combination of multiple attribute values would mean an even bigger combinatorial for the decryption phase. Practical limits of the approach will have to be experimented: on one hand, protocol timeouts are the upper bound regarding the lookup that can be performed (e.g. in the order of 5 s for WS-Discovery); on the other hand, the affordable time for performing a discovery is much less for such a system to remain usable (probably a maximum search time of a few seconds).

Another possibility that we can consider to optimize the key combination, is the usage of Policy Based Encryption mechanism [18] (PBE). This mechanism is an extension of the ID-based cryptography that allows the encryption of a data

according to a policy, and only the entity that fulfills this policy is able to decrypt this data. The advantage of using this mechanism is the possibility to use conjunctions and disjunctions to make an efficient combination of the encryption/decryption attributes. Unfortunately, there is no available implementation of this cryptographic scheme.

4.5. Key Revocation

Identity Based cryptosystems do not really have a key revocation mechanism: if a key is corrupted, all the entities related to the CA have to change the system parameters. The solution to limit the impact of this issue therefore consists in defining a key validity expiration date. Compared with public key certificate schemes in which the identifier is a random value revocable and replaceable over time, an IBE public key identifier is a unique name that cannot be revoked. In contrast, as proposed in [8], it can be extended: we can for instance concatenate to this name with a variable value like a date, e.g., {Printer | 2006}.

4.6. WS-Discovery Proxies

We can extend our solution to the discovery proxies (described in the section four) by making some modification to the original protocol. In order to retrieve a service, the client can choose to contact a Discovery Proxy by multicasting (Figure 10) a *Probe* message encrypted using the identifier {DiscoveryProxy} as encryption key. And only certified proxies are able to decrypt the message using the associated private key provided by the PKG.

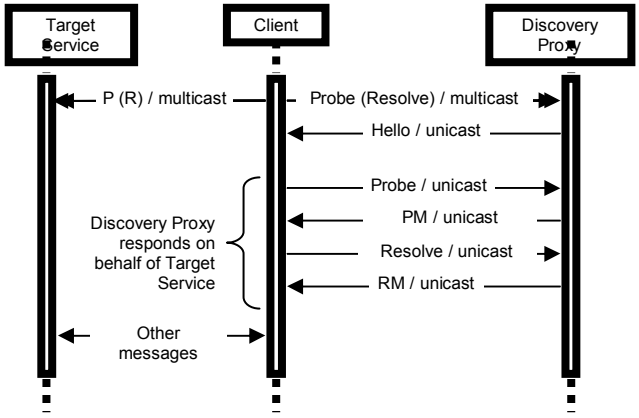

Figure 10: WS-Discovery sequence messages including a discovery proxy

4.7. Private Key Generation: Online Vs Offline

The private key generation is problematic for two reasons. First, How can the user contact the Private Key Generator (PKG) in a secure manner (secure key distribution)? Second, what are the appropriate keys required to decrypt the messages (key generation)? One of the interesting features of IBE and ABE systems is the choice of the key distribution mechanism, for which there exist three possible architectures:

- Online PKG: there exists a key server that is permanently running and reachable by the users in a secure manner. The user sends the server the requested attributes, and depending on his profile, the server generates and sends the private key corresponding to these attributes. It is a simple solution, yet inapplicable to ad-hoc environments non permanently connected to a network infrastructure.
- Offline PKG: the key server generates all the private key corresponding to the attributes contained in the user profile. These keys are stored into a protected directory in the user's laptop. The user will choose among his key collection the appropriate private key corresponding to the expected attributes of the recipient. This architecture makes it difficult however to change the system parameters (profiles, attributes, keys …) or to manage the revocation of keys.
- Embedded PKG: the key server is implemented in a tamper-resistant hardware like a smartcard provided by the certification authority. This solution represents an ideal compromise between the two precedent solutions for ubiquitous computing applications. Using an embedded PKG the user that needs to get a private key does not have to contact in a secure manner a distant PKG. In fact in order to protect the sensitive information (private key, certificate …) during the key exchange, the two involved entities have to setup a secure channel. The second advantage of using an embedded PKG, is the permanent availability, the user does not have to be permanently connected to a network to reach a distant PKG. But the weakness of this solution concerns the Attribute certificate revocation management. Because of the off-line status of the PKG, it is not possible to verify if the attribute certificate provided by the key requester is revoked by the CA.

5. RELATED WORK

In a precedent work we [10] made an analysis on some threats in SOA attacks that use a registry supported discovery. We also detailed some specific discovery security issues and we proposed a modification on the usual message exchange protocol provided to secure the discovery process. This modification is based on a trusted infrastructure for the discovery including a registry that has the task to establish a trust relationship between the different actors of the system. This security model provides mechanisms that should be used during service discovery in order to protect not only the server, but also the service requestor regarding security and privacy.

One of the first approaches dealing with secure service discovery was proposed by [20]. This architecture relies on an additional component, called Service Discovery Service (SDS), which plays the role of a secure information repository (secure registry). This SDS helps clients and servers to set up a trust relationship and secure channels between each another:

it provides authentication, access control, encryption, signature verification, and privacy protection using a PKI. The SDS multicasts its public key certificate in order to allow the encryption of the service publication and of the client request. A server has also the possibility to restrict the discovery of its services to some specific clients by associating an access control list with the published service profile; a component of the SDS called the capability manager will use this list to enforce access control. The SDS, which has to be permanently available, and which has to decrypt all messages coming from clients and servers surrounding it therefore constitutes a single point of failure. It could create a bottleneck, and some malicious user could attack the SDS by sending fake encrypted messages. In order to encrypt the exchanged messages, the SDS uses a hybrid public/symmetric key system. For every service it has to memorize, a temporary symmetric key, that could affect the performance of the SDS if there is a large number of services around. Trust establishment between the SDS and other entities is limited to a simple verification of the SDS public certificate validity.

[11] also proposes an architecture for securing service discovery. In this work, components share a multicast address that will be used to bootstrap communication. Directories (i.e. registries) use this multicast address to periodically announce their unicast address and certificate. Proxies are used to protect the servers by handling the registration, authentication, authorization, and key management for them. Entities in the system set up a session using hybrid encryption. Due to the number of proxies (one by service) and the PKI infrastructure used to secure the communication, the model proposed is however likely to generate an important message overhead. Contrary to the claimed objective of this work to address pervasive systems, services are likely to be static, whereas the approach we advocate only requires the local availability of a fixed registry (each client potentially being a server for other clients).

Privacy issues are addressed by [12] in which the authors propose the use of Bloom filters to protect the client and server personal information (identity, certificates, attributes…). Membership tests are performed between the directory and the client using generated Bloom filters in order to authenticate themselves. The participating entities must agree beforehand on specific hash functions in order to use these Bloom filters, yet this issue is not resolved but through a static agreement. In comparison, the present paper's approach to privacy relies on the knowledge of predefined attributes corresponding to specific services.

The notion of “attribute-based encryption” is introduced in [13] where the authors are proposing a new approach of the IBE mechanism usage in which identity is considered as a set of descriptive attributes. The authors propose to encrypt and decrypt data using the biometric attributes of a user (iris scan, finger print …). The claimed advantage of this approach is that the user does not need a certificate but only his biometric signature in order to prove his identity and to obtain the private key used for decryption.

As mentioned in the previous section, the possibility to use a smartcard as a PKG seem to be the more practical solution in order to generate and distribute in a secure manner the appropriate private keys. This kind of embedded PKG was already developed [14] as a Java Card applet enabling secure key generation.

6. Conclusion

This paper discussed specific security and privacy issues of peer-to-peer service discovery mechanisms. A solution based on a particular type of Identity Based encryption called Attributed based Encryption was used to add security during the service discovery process by protecting the user’s requests and restricting the access to the discovery of a service. This solution does not need to rely on a trusted third party in order to perform the matchmaking function. This solution's main limitation is the possibility for a malicious user to perform a DoS attack against a service provider, a problem which we are currently working on.

We are implementing a secure extension of the WS-Discovery protocol as part of the developed within the European project MOSQUITO [17] middleware. We are also planning on experimenting peer-to-peer discovery with an embedded PKG as an alternative solution with less constraints regarding infrastructure deployment.

Attribute Based Encryption makes it possible to both encrypt for and describe the semantics of a service, which is essential for accurately yet privately answering a request. Contrary to PKIs, IBE public keys (a string) identify services with human understandable semantics, even though they should be shared by all parties. Using ABE further permits combining different attributes to describe some complex service profiles or service types. The use of ABE may also prove beneficial in more open scenarios in which services are likely to be described using various knowledge representations, like for instance controlled vocabularies or ontologies. We are investigating how to combine reasoning about how service profiles match with such tools together with encryption, even though this may require tradeoffs regarding the privacy of service lookup.

Acknowledgment

The authors wish to thank Cedric Roux from Institut Eurecom for his important contribution in developing the java version of the WS-Discovery Protocol

Bibliography

[1] SUN Microsystems, Jini Specificationshttp://java.sun.com/products/jini/
[2] OASIS, “UDDI”, http://www.uddi.org
[3] WS-Discovery Specifications http://msdn.microsoft.com/ws/2005/04/ws-discovery/
[4] D. Martin et al, “Bringing Semantics to Web Services: The OWL-S Approach”, Proceedings of the 1st SWSWPC, USA 2004.
[5] WSDL specifications http://www.w3.org/TR/wsdl
[6] DARPA Agent Markup Language (DAML) http://www.daml.org/

[7] S. Crosta, J C. Pazzaglia, H. Schottle “Modelling and Securing European Justice Workflows“, 6th Information Security Solutions Europe (ISSE 2005) Security Conference 27-29 September 2005 - Budapest, Hungary
[8] D. Boneh and M. Franklin, “Identity-based encryption from the weil pairing“, Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology, pages 213–229. Springer-Verlag, 2001.
[9] A. Duffy and T. Dowling, “An Object Oriented Approach to an Identity Based Encryption Cryptosystem”, 8th IASTED International Conference on Software Engineering and Applications, 2004.
[10] xxxxx, xxxx and xxxxx “Enabling Secure Discovery in a Pervasive Environment” 3rd International Conference on Security in Pervasive Computing (SPC 2006) – York – UK – April 2006
[11] F. Zhu, M. Mutka, and L. Ni, “Splendor: A secure, private, and location-aware service discovery protocol supporting mobile services”, in Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (Percom’03). IEEE Computer Society, Mar. 2003, pp. 235–242.
[12] F. Zhu, M. Mutka, L. Ni “Prudent exposure: A private and user centric service discovery protocol” Proceedings of the 2nd IEEE International Conference on Pervasive Computing and Communications (PerCom’04) Orlando, USA, 2004
[13] A. Sahai and B. Waters, “Fuzzy Identity-Based Encryption”, Advances in Cryptology-Eurocrypt'05.LNCS 3494, pp. 457-473, Springer, 2005.
[14] A. Duffy and T. Dowling, "Java Card Key Generation for Identity Based Systems", NUI Maynooth Department of Computer Science, Technical Report Series, 2005.
[15] M. Ghader et al “Secure resource and service discovery in personal networks” Wireless World Research Forum Meeting #12, Canada, 4-5 Nov 2004
[16] CORBA http://www.corba.org/
[17] MOSQUITO Project IST 004636 https://www.mosquito-online.org/
[18] W. Bagga, R. Molva, “Policy-based cryptography and applications”, FC' 2005, 9th International Conference on Financial Cryptography and Data Security, 28 February-03 March 2005, Roseau, The Commonwealth of Dominica - Also published in LNCS Volume 3570
[19] Voltage IBE Toolkit http://www.voltage.com/ibe_dev/
[20] S.E. Czerwinski et al, “An Architecture for a Secure Service Discovery Service” , In Proceedings of MobiCom '99, Seattle, WA, August 1999
[21] A. Shamir, “Identity-based cryptosystems and signature schemes", in Advances in Cryptology Crypto '84, Lecture Notes in Computer Science, Vol. 196, Springer-Verlag, pp. 47{53, 1984
[22] V Goyal, et al, “Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data”, Proceedings of 13th ACM Conference on Computer and Communications Security (CCS 2006), Alexandria, USA, October 2006