Despliegue de políticas condicionadas para la protección de atributos en negociadores móviles

Carles Martínez-García*, Guillermo Navarro-Arribas†, Joaquin Garcia-Alfaro‡,§

* Universitat Autònoma de Barcelona, Edifici Q, 08193, Bellaterra
† Artificial Intelligence Research Institute, 08193, Bellaterra
‡ Universitat Oberta de Catalunya, 08018, Barcelona
§ Institut Telecom, Telecom Bretagne, 35576, Cesson-Sevigne, France

***Resumen*—En este trabajo, presentamos una propuesta para la realización de procesos automáticos de negociación en entornos móviles a través del despliegue de políticas condicionadas. Dichas políticas contienen un conjunto de atributos condicionados por el propio usuario para proteger aquellas informaciones consideradas como sensibles o que potencialmente pudieran violar su privacidad. A medida que avance el proceso de negociación, los atributos servirán de mecanismo de control para concluir las distintas etapas que componen dicho proceso. Presentamos una visión práctica de nuestra estrategia orientada hacia su integración en aplicaciones pervasivas de comercio electrónico basadas en el uso de agentes móviles y políticas definidas según el estándar XACML.**

I. Introducción

Dadas las tendencias actuales en entornos de usuario de aplicaciones móviles y/o pervasivas, aparece con frecuencia un requisito básico: un proceso de negociación automático. El objetivo suele ser, por lo general, cubrir las necesidades de optimización de parámetros de dichos entornos. Algunos ejemplos apropiados podrían ser la optimización de parámetros en escenarios de tipo *roaming* (para conseguir los mejores parámetros de calidad de servicio) o *multi-homing* (para conseguir accesos de tipo *always best connected*). Por supuesto, es importante destacar también la necesidad de negociación con simples propósitos transaccionales (por ejemplo, búsqueda de una oferta, no necesariamente la más barata, en escenarios tradicionales de tipo proveedor-consumidor).

Todos estos procesos de negociación suelen requerir el intercambio de datos de carácter personal (tales como, preferencias de usuario, datos bancarios o histórico de negociaciones previas). Parte de esta información es generalmente vista por el usuario como sensible o privada. De hecho, este tipo de datos puede ser utilizado por los proveedores de manera inapropiada (por ejemplo, utilización ilícita de métodos de *profiling* para garantizar QoS). La necesidad de garantizar una protección adecuada de dichos datos se pone de manifiesto de manera aún más acentuada cuando el proceso de negociación se ejecuta, en nombre del usuario, por entidades *software* autónomas.

Estas entidades suelen ejecutar el proceso de negociación en plataformas distantes, diseminando registros sobre el usuario. Estos registros pueden ser utilizados por un adversario para violar su privacidad. Así, por ejemplo, trabajos como [7] han mostrado que tan sólo un mínimo de conocimiento sobre un individuo, cuya identidad se ha borrado de un proveedor de servicios, puede ser suficiente para identificarlo a partir de un conjunto de datos públicos. Es necesario, pues, introducir un mecanismo de protección adicional en estos entornos.

En este trabajo proponemos un mecanismo sencillo, a la vez que eficiente, para permitir la revelación progresiva de información de carácter personal en entornos pervasivos de negociación automática. Para ello, nos centramos en la realización del proceso por parte de agentes móviles. Estos presentan dos características importantes, movilidad efectiva de código (con todo lo que ello comporta), y capacidad de negociación en nombre del usuario. Así mismo, planteamos un acercamiento a la negociación basado en políticas, de manera similar a [9], [5], [1], donde agentes móviles se utilizan para automatizar la negociación entre clientes y proveedores de servicios. Nuestra propuesta se puede ver como una simplificación de sistemas de *trust negotiation* [8], [13], o como un soporte complementario a estas en entornos de computación móvil en general.

Organización del artículo — El resto del artículo se ha organizado de la siguiente manera: La sección II presenta algunos antecedentes en relación a nuestra propuesta. La sección III define de manera más concreta la motivación de nuestro trabajo y presenta la propuesta haciendo hincapié en su arquitectura y aportando notas de implementación. Finalmente, la sección IV concluye el artículo.

II. Antecedentes

Multitud de técnicas de negociación han sido propuestas y analizadas en el área de la matemática aplicada y de la teoría de juegos [4], [10]. La mayoría de estas técnicas tratan de utilizar modelos formales para definir e interpretar interacciones entre dos o más participantes, en forma de incentivos que conducirán finalmente el proceso de decisión de cada participante. Podemos encontrar, entre dichas soluciones, el estudio de estrategias que garanticen una decisión óptima, en términos económicos, a través de la detección de comportamientos preestablecidos. Estas técnicas han influenciado de manera decisiva la mayoría de soluciones basadas en la utilización de agentes y/o técnicas de computación cooperativa. De hecho,

estas soluciones han sido adaptadas con éxito hacia marcos de trabajo específicos en el área de negociación en aplicaciones de comercio electrónico [9] y redes IP móviles [5], [1]. La utilización de estas técnicas de negociación se prevé de vital importancia para las futuras aplicaciones de la tecnología ubicua, siendo de especial relevancia la necesidad de procesos de negociación que garanticen un intercambio mínimo de datos de carácter personal entre consumidores y proveedores de servicios electrónicos [12].

Cualquier proceso de negociación requiere la realización de una toma de decisiones. Estas decisiones son influenciadas, en gran medida, por las necesidades propias de cada una de las partes involucradas en el proceso. La capacidad de anticiparse a los deseos o motivaciones de un participante puede suponer una clara ventaja para sus oponentes. Por este motivo, la mayoría de sistemas de apoyo a la negociación tratan de anticiparse a los deseos/necesidades de sus oponentes mediante la incorporación de métodos que permitan modelar, y por lo tanto, anticipar, las decisiones de los participantes [11]. Asumamos, por ejemplo, aplicaciones de negociación electrónica de tipo *policy-driven*. Estas aplicaciones acostumbran a conducir el proceso de negociación a partir del intercambio de un conjunto de políticas en formato electrónico. Estas políticas permiten definir, a partir de lenguajes formales basados en lógica de primer orden, por ejemplo, el conjunto de declaraciones que será utilizado en el proceso de negociación. Multitud de lenguajes han sido propuestos en la literatura con el objetivo de formalizar este proceso [2]. Existen también en la literatura métodos de detección que permiten el análisis de dichas políticas para poder determinar, de entre un conjunto de posibles situaciones, aquellas que sean potencialmente más probables para conducir el proceso de negociación hacia un objetivo determinado [10], [11]. La parte más relevante a analizar acostumbra a ser el conjunto de atributos incluidos en la solicitud de ofertas, así como los resultados de la negociación. La figura 1 muestra un ejemplo, inspirado en la familia de protocolos propuestos en [9], donde podemos apreciar la inclusión de un conjunto de atributos que identificarán el objeto asociado al proceso de negociación, así como a las entidades involucradas en dicho proceso.

```
...
<Transaction name="SubmitProposal" ... >
  <Collaboration name="ReachAgreement">
    < InitiatingRole  name="Requester" id="... />
    <RespondingRole name="Responder" id="... />
    < Activity  name="RequesterNegotiation"
                binaryCollaboration="ConductNegotiation"
                fromAuthorizedRole="AgreementRequester"
                toAuthorizedRole="AgreementResponder">
      <Start toState="RequesterNegotiation" ... />
      <Transition  fromState="RequesterNegotiation"
                   toState="RequesterContract"
                   conditionGuard="Success"
                   ... />
      <Failure  fromState="RequesterNegotiation"
                conditionGuard="AnyFailure"
                ... />
      <Success fromState="RequesterContract"
                conditionGuard="Success"
                ... />
      <Failure  fromState="RequesterContract"
                conditionGuard="AnyFailure"
                ... />
    </ Activity >
    <RequestOffer ... >
      <Attribute  name="currency" EUR />
      <Attribute  name="productNumber" 1234−5678 />
      <Attribute  name="productName" Notebook Computer />
      <Attribute  name="productDescription" Mobile ... />
    </RequestOffer>
</Transaction>
...
```

Figura 1. Solicitud de ofertas en una negociación de tipo *policy-driven*.

En el caso de negociaciones donde se requiera el intercambio de datos de carácter personal, ya sea para la identificación del objeto asociado al proceso de negociación, o para identificar a las entidades del proceso, es imprescindible garantizar un proceso de protección apropiado. La simple clasificación e identificación de los datos que requerirán dicha protección puede llegar a ser extremadamente compleja. Esto se pone aún más de relieve si tenemos en cuenta que incluso una dirección IP o un número de teléfono móvil asociado a las entidades u objetos del proceso de negociación pueden ser considerados como datos de carácter personal a proteger. En este sentido, el grupo de trabajo de la unión europea, encargado de regular la ley de protección de datos y vida privada de los ciudadanos, urge en su dictamen presentado en [3] la búsqueda de nuevas soluciones, más allá de la simple ofuscación de datos, para garantizar el derecho a la protección de datos de carácter personal de este tipo de aplicaciones. El objetivo de la propuesta que presentamos a continuación es precisamente iniciar el estudio de nuevas soluciones que puedan tratar esta problemática.

III. Información privada en negociadores móviles

El marco donde se centra nuestra propuesta se caracteriza, por un lado, por ser un entorno distribuido en el cual existen diferentes proveedores de servicios. Por otro lado, diferentes usuarios pretenden acceder a los distintos servicios desplegados. Como paso previo al acceso a los servicios, se establece un proceso de negociación, local al proveedor de servicios, en el cual un agente móvil representa al usuario. Para ello, el agente negociador presenta una política de negociación que contiene información referente a éste. Esta información está contenida en una política XACML (*eXtensible Access Control Markup Language*), estándar de OASIS [6] que proporciona un lenguaje basado en XML muy flexible y expresivo, para especificar políticas de control de acceso, así como el protocolo de petición-respuesta asociado.

Dada la vulnerabilidad que supone que el código móvil se ejecute en una plataforma remota y, a priori, no confiable, hace que la información referente al usuario, contenida en la política de negociación, sea susceptible de ser comprometida. En términos de privacidad, esta vulnerabilidad desautoriza a los agentes móviles a contener una política de negociación con información referente al usuario. No obstante, esta política se hace necesaria para completar el proceso de negociación. Cabe

destacar que cualquier registro de información residual en una plataforma podría ser utilizados por un adversario en beneficio propio. Surge, pues, la necesidad de controlar el filtrado de información de carácter personal. Es más, en ciertos casos en que el usuario no confíe lo suficiente en la plataforma del proveedor de servicios, se puede sacrificar el resultado final de la negociación por la preservación de cierto nivel de privacidad.

Con este propósito, el usuario percibe el riesgo de filtrado de información (invasión de privacidad) como un contexto de la misma negociación. Hay contextos donde el usuario, ya sea por su confianza en el proveedor de servicios o por su necesidad de obtener un resultado mejor en la negociación, puede estar dispuesto a revelar más información inicialmente considerada privada. Así pues, y dentro de cada contexto de privacidad concreto, proponemos que el agente móvil encargado de la negociación lleve una política ad-hoc a la etapa de negociación. Cada política que el agente negociador llevará consigo contiene información diferente según su grado de privacidad. De esta manera, se establece, mediante el contexto, el nivel de privacidad bajo el que debe operar el proceso de negociación. Aunque pueda parecer tedioso el desplegar una política diferente para cada contexto de privacidad, hay que tener en cuenta que la generación de políticas es completamente automática y, por lo general, transparente al usuario.

III-A. Arquitectura

El escenario en el que trabajamos está compuesto por distintas entidades [1]. En primer lugar, un agente móvil de tipo *User Negotiator* (UN), que es enviado por el usuario a la plataforma del proveedor. Su primera tarea es sondear las ofertas de los proveedores (fase de descubrimiento), y la segunda es la negociación del servicio. En segundo lugar, en el lado del proveedor, un agente de tipo *Access Negotiator* (AN) se encarga de negociar, con los UN que llegan, los términos de acceso al servicio. Por último, dada la vulnerabilidad que supone que un agente acarree información privada, surge la necesidad de aparición en el esquema de los agentes de tipo *User Overseer* (UO). Los agentes de tipo UO son los encargados de enviar agentes UN para el descubrimiento y la negociación de servicios. Estos agentes, además, son los encargados de gestionar la información que el UN contiene acerca del usuario durante cada fase de la negociación.

Como vemos en la figura 2, esta arquitectura no solo permite gestionar la información que es presentada al proveedor en cada fase de la negociación, sino también la cantidad de información que el agente negociador contiene sobre el usuario y, por ello, susceptible de ser comprometida. Se hace esencial un mecanismo entre UO y el UN que proporcione las funcionalidades necesarias para controlar el intercambio de información sensible. Dicho mecanismo se fundamenta de la siguiente manera:

- Definimos formalmente el riesgo en el contexto de negociación como:
$$\sigma : [0, 1] \tag{1}$$
donde $\sigma = 0$ significa que no hay ningún riesgo asociado a la negociación y $\sigma = 1$ significa el mayor nivel de criticidad.
- Denotamos como $Attr_i$ al conjunto de atributos que representan al usuario. El usuario está interesado en obtener la mejor oferta de varios proveedores. El intercambio de datos $Attr_i$ durante la negociación entre UN y AN se condiciona por ciertas restricciones (por ejemplo, al resultado de alguna estrategia previa de negociación). De esta manera el UN debería intercambiar gradualmente $Attr_i$ preservando las preferencias del usuario.
- El revelado de información durante la negociación debe de ser gradual y consecuente al contexto de privacidad. Para ello, la criticidad de cada atributo se define como:
$$\mu(Attr_i) : [0, 1] \tag{2}$$
donde $\mu(Attr_i) = 0$ implica el menor grado de criticidad asociado al atributo $Attr_i$ y $\mu(Attr_i) = 1$ significa que el atributo posee el mayor nivel de criticidad. A su vez, el conjunto de atributos cuya criticidad es cero, esto es $\{Attr_i | \mu(Attr_i) = 0\}$, representa el conjunto de datos públicos que el usuario está dispuesto a intercambiar por defecto. Un ejemplo sobre la relación atributo-criticidad puede verse en el cuadro I.
- El usuario negocia con todos los proveedores de la misma manera. Por ello, inicialmente todos los proveedores se consideran no confiables.
- Dado un contexto de privacidad definido, el usuario revela cada uno de los atributos si y solo si:
$$\mu(Attr_i) \leq \sigma \tag{3}$$
Es decir, un atributo solo es revelado si su nivel de criticidad asociado es inferior o igual al umbral marcado por el riesgo asociado al contexto de negociación.

En este caso, el UN es el sujeto que realizará la petición de la política que mantiene el UO. Asumimos que los agentes de tipo UN incluyen la funcionalidad necesaria para realizar el proceso de negociación [1] ya que pueden evaluar todas las peticiones respuestas y gestionar los contextos. Si aplicamos un refinamiento de políticas, se delega a los UNs la capacidad

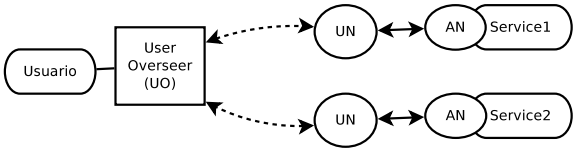

Figura 2. Arquitectura basada en [1].

Cuadro I
EJEMPLO DE LA RELACIÓN ATRIBUTO-CRITICIDAD.

$Attr_i$	$\mu(Attr_i)$
$attr_1$	0,8
$attr_2$	0,7
$attr_3$	0,3
$attr_4$	0

de decisión para controlar el intercambio privado de datos con los ANs. Una vez que los UNs son enviados a las plataformas de los proveedores, el usuario no puede —ni debe— interferir en la negociación.

III-B. Notas de implementación

El módulo UO, encargado de la generación dinámica de la política dependiendo del contexto de privacidad, presenta las siguientes funcionalidades:

- Representación de los atributos del usuario junto a su nivel de criticidad. El usuario debe, de forma sencilla, poder especificar los atributos que lo caracterizan así como su nivel de criticidad asociando a cada atributo un valor dentro del rango $[0, 1]$.
- Contener el patrón de la política de negociación. El UO debe contener el patrón de la política de negociación que le permitirá, una vez conocido el riesgo asociado al contexto de negociación, generar la política adecuada.
- Presentar la lógica necesaria para la generación de la política de negociación de forma transparente al usuario. Esta lógica actúa sobre el patrón de la política, para generar de forma dinámica la política de negociación.

Para la generación automática de la política en XACML, existen varias herramientas que permiten la inclusión de código dentro de documentos de texto patrón. La ejecución de éste código redunda en un documento de texto alterado en el que el código ha sido substituido por el resultado de su ejecución. En nuestro mecanismo, usamos la herramienta de *templating* ERB [14]. De esta manera, se incluye, dentro de un fichero patrón que contiene la política de negociación expresada en XACML, la lógica necesaria, expresada en el lenguaje Ruby [15], para la generación de las líneas que relacionan los atributos con el usuario dependiendo del nivel de criticidad. La generación automática de código XACML es una idea análoga a la creación de servicios web dinámicos.

Previo al proceso de negociación, el usuario debe facilitar al UO tanto sus atributos como el nivel de criticidad asociado a cada uno de ellos. De la misma forma, el usuario debe facilitar el patrón de la política de negociación. Este patrón se compone de un documento XACML con cláusulas que contienen código en ruby. El usuario únicamente se debe preocupar por incluir una llamada a un método llamado *expand_attributes*. Método cuyo resultado de ejecución redunda en la impresión de aquellos atributos de usuario cuya criticidad sea menor o igual a la cota fijada por el contexto de privacidad. Cuando el UN establezca el contexto de privacidad, pedirá al UO que genere una política de negociación ad-hoc al contexto. Conociendo el contexto de privacidad, el patrón de la política de negociación y los atributos referentes al usuario, el UO es capaz de ejecutar la lógica incluida en el patrón para obtener la política de negociación que será enviada al UN. Una vez recibida la política, el UN continuará con el proceso de negociación.

La figura 3 muestra un ejemplo XACML simplificado del módulo que regula la revelación de atributos del UO, que generará las políticas de negociación del UN condicionadas por el contexto de privacidad. En dicha figura se observa que

```
...
<Subject>
  <%= expand_attributes %>
</Subject>
...
```

Figura 3. Patrón de la política de negociación.

```
...
<Subject>
        <Attribute  AttributeId =" attribute "
         DataType="http://www.w3.org/2001/XMLSchema#string">
                <AttributeValue>attr3</AttributeValue>
        </ Attribute >
        <Attribute  AttributeId =" attribute "
         DataType="http://www.w3.org/2001/XMLSchema#string">
                <AttributeValue>attr4</AttributeValue>
        </ Attribute >
</Subject>
...
```

Figura 4. Política generada a través del patrón especificado en la figura 3.

la inclusión de los atributos del usuario dentro de la política de negociación depende del contexto y se realiza a través de la llamada al método *expand_attributes*. La figura 4 muestra el resultado de la ejecución, dónde los atributos reflejados en el cuadro I se han incluido en la política de negociación bajo un contexto de privacidad $\sigma = 0{,}5$.

IV. Conclusiones

En este trabajo se ha presentado una solución de protección de la privacidad en procesos de negociación basada en un modelo de refinamiento de políticas condicionadas. El uso de un conjunto de transformaciones dinámicas de las políticas delegadas a un conjunto de agentes móviles se ha propuesto como mecanismo de implementación. Consideramos que nuestra propuesta podría ser también valida para otros escenarios, tales como procesos de integración de políticas de control de acceso, despliegue de configuraciones para sistemas de seguridad, o intercambio de alertas de detección en entornos de detección cooperativa. Una presentación más elaborada de nuestra propuesta así como su adaptación al resto de escenarios será tratada en un futuro informe.

Agradecimientos

Este trabajo está respaldado por el Departament d'Innovació, Universitats i Empresa (2009SGR1224), por la Universitat Autònoma de Barcelona (PIF 472-01-1/07) y por el Ministerio de Ciencia y Educación (proyectos E-AEGIS TSI2007-65406-C03-03, TSI2007-65406-C03-02, y CONSOLIDER-INGENIO CSD2007-00004 ARES). G. Navarro-Arribas disfruta de una beca Juan de la Cierva (JCI-2008-3162) del MICINN.

Referencias

[1] Benmammar, B., Jrad, Z., and Krief. F. QoS management in mobile IP networks using a terminal assistant. In *International Journal of Network Management*, 19(1):1–24, 2009.

[2] Bonatti, P., De Coi, J., Olmedilla, D., Sauro, L. Policy-Driven Negotiations and Explanations: Exploiting Logic-Programming for Trust Management, Privacy & Security. In *Logic Programming*, vol. 5366, LNCS, p. 779-784, 2008.

[3] Grupo de trabajo sobre protección de Datos del artículo 29. `http://ec.europa.eu/justice_home/fsj/privacy/docs/wpdocs/2007/wp136_en.pdf`

[4] Karrass, C. L. Give and Take: The Complete Guide to Negotiating Strategies and Tactics. HarperCollins Publishers, New York, NY (1993).

[5] Krief, F. Self-aware management of IP networks with QoS guarantees. In *International Journal of Network Management*, 14(5), pp. 351-364, 2004.

[6] Moses, T. (Ed.). eXtensible Access Control Markup Language (XACML) Version 2.0. OASIS Standard, 1 Feb 2005.

[7] Narayanan, A., and Shmatikov, V. Robust De-anonymization of Large Sparse Datasets. In *IEEE Symposium on Security and Privacy 2008*, pp. 111-125, Oakland, California, USA (2008).

[8] Nejdl, W., Olmedilla, D., Winslett, M. PeerTrust: Automated Trust Negotiation for Peers on the Semantic Web. In *Workshop on Secure Data Management in a Connected World*, Canada, 2004.

[9] Rebstock, M., Thun, P., Tafreschi, O. A. Supporting Interactive Multi-Attribute Electronic Negotiations with ebXML. *Group Decision and Negotiation Journal*. vol. 12, p. 269–286, 2003.

[10] Stuhlmacher, A. F., Stevenson M. K. Using Policy Modeling to Describe the Negotiation Exchange. *Group Decision and Negotiation Journal*. vol. 6, p. 317–337, 1997.

[11] Vetschera, R. Preference Structures of Negotiators and Negotiation Outcomes. *Group Decision and Negotiation Journal*. vol. 15, p. 111–125, 2006.

[12] Yee, G. and Korba, L. Feature Interactions in Policy-Driven Privacy Management. *Seventh International Workshop on Feature Interactions in Telecommunications and Software Systems (FIW'03)*. 2003.

[13] Yu, T., Winslett, M., and Seamons, K. E. Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. *ACM Transactions on Information and System Security (TISSEC)* vol. 6, no. 1, (Feb. 2003), 1-42.

[14] http://www.ruby-doc.org/stdlib/libdoc/erb/rdoc/

[15] http://www.ruby-lang.org/es/