H. Schulzrinne
Columbia U.
J. Morris
CDT
H. Tschofenig
J. Cuellar
Siemens
J. Polk
Cisco
J. Rosenberg
Cisco Systems

February 11, 2006

A Document Format for Expressing Privacy Preferences
draft-ietf-geopriv-common-policy-07.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 15, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document defines a framework for authorization policies controling access to application specific data. This framework combines common location- and presence-specific authorization aspects. An XML schema specifies the language in which common policy rules are represented. The common policy framework can be extended to other application domains.

Table of Contents

1. Introduction

This document defines a framework for creating authorization policies for access to application specific data. This framework is the result of combining the common aspects of single authorization systems that more specifically control access to presence and location information and that previously had been developed separately. The benefit of combining these two authorization systems is two-fold. First, it allows to build a system which enhances the value of presence with location information in a natural way and reuses the same underlying authorization mechanism. Second, it encourages a more generic authorization framework with mechanisms for extensibility. The applicability of the framework specified in this document is not limited to policies controling access to presence and location information data, but can be extended to other application domains.

The general framework defined in this document is intended to be accompanied and enhanced by application-specific policies specified elsewhere. The common policy framework described here is enhanced by domain-speific policy documents, including presence [6] and location[7]. This relationship is shown inFigure 1.

```
                           +-----------------+
                           |                 |
                           |     Common      |
                           |     Policy      |
                           |                 |
                           +---+---------+---+
                              /|\       /|\
                               |         |
      +-------------------+    |         |    +-------------------+
      |                   |    | enhance |    |                   |
      | Location-specific |    |         |    | Presence-specific |
      |      Policy       |----+         +----|      Policy       |
      |                   |                   |                   |
      +-------------------+                   +-------------------+
```

Figure 1: Common Policy Enhancements

This document starts with an introduction to the terminology in Section 2, an illustration of basic modes of operation in Section 3, a description of goals (see Section 4) and non-goals (see Section 5) of the authorization policy framework, followed by the data model in Section 6. The structure of a rule, namely conditions, actions and transformations, are described in Section 7, in Section 8 and in Section 9. The procedure for combining permissions is explained in Section 10 and used when more than one rule fires. A short description of meta policies is given in Section 11. An example is provided in Section 12. The XML schema will be discussed in Section 13. IANA considerations in Section 15 follow security considerations Section 14.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT","RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [1].

This document introduces the following terms:

PT - Presentity / Target: The PT is the entity about whom information has been requested.

RM - Rule Maker: RM is an entity which creates the authorization rules which restrict access to data items.

PS - (Authorization) Policy Server: This entity has access to both the authorization policies and to the data items. In location-specific applications, the entity PS is labeled as location server (LS).

WR - Watcher / Recipient: This entity requests access to data items of the PT. An access operation might be either be a read, write or any other operation. In case of access to location information it might be a read operation.

An 'authorization policy' is given by a 'rule set'. A 'rule set' contains an unordered list of 'rules'. A 'rule' has a 'conditions', an 'actions' and a 'transformations' part.

The term 'permission' indicates the action and transformation components of a 'rule'.

The terms 'authorization policy', 'policy' and 'rule set' are used interchangeably.

The terms 'authorization policy rule', 'policy rule' and 'rule' are used interchangeable.

The term 'using protocol' is defined in [8]. It refers to the protocol which is used to request access to and to return privacy sensitive data items.

3. Modes of Operation

The abstract sequence of operations can roughly be described as follows. The PS receives a query for data items for a particular PT, via the using protocol. The using protocol provides the identity of the requestor (or more precisely the authentication protocol), either at the time of the query or at the subscription time. The authenticated identity of the WR, together with other information provided by the using protocol or generally available to the server, is then used for searching through the rule set. All matching rules are combined according to a permission combining algorithm described in Section 10. The combined rules are applied to the application data, resulting in the application of privacy based on the transformation policies. The resulting application data is returned to the WR.

Three different modes of operation can be distinguished:

3.1. Passive Request-Response - PS as Server (Responder)

In a passive request-response mode, the WR queries the PS for data items about the PT. Examples of protocols following this mode of operation include HTTP, FTP, LDAP, finger or various RPC protocols, including Sun RPC, DCE, DCOM, Corba and SOAP. The PS uses the ruleset to determine whether the WR is authorized to access the PTs information, refusing the request if necessary. Furthermore, the PS might filter information by removing elements or by reducing the resolution of elements.

3.2. Active Request-Response - PS as Client (Initiator)

Alternatively, the PS may contact the WR and convey data items. Examples include HTTP, SIP session setup (INVITE request), H.323 session setup or SMTP.

3.3. Event Notification

Event notification adds a subscription phase to the "PS as client" mode of operation. A watcher or subscriber asks to be added to the notification list for a particular presentity or event. When the presentity changes state or the event occurs, the PS sends a message to the WR containing the updated state. (Presence is a special case of event notification; thus, we often use the term interchangeably.)

In addition, the subscriber may itself add a filter to the subscription, limiting the rate or content of the notifications. If an event, after filtering by the rulemaker-provided rules and by the subscriber-provided rules, only produces the same notification content that was sent previously, no event notification is sent.

A single PS may authorize access to data items in more than one mode. Rather than having different rule sets for different modes all three modes are supported with a one rule set schema. Specific instances of the rule set can omit elements that are only applicable to the subscription model.

4. Goals and Assumptions

Below, we summarize our design goals and constraints.

Table representation:

Each rule must be representable as a row in a relational database. This design goal should allow efficient policy rule implementation by utilizing standard database optimization techniques.

Permit only:

Rules only provide permissions rather than denying them. Allowing both 'permit' and 'deny' actions would require some rule ordering which had implications on the update operations executed on these rules. Additionally it would make distributed rule sets more complicated. Hence, only 'permit' actions are allowed which result in more efficient rule processing. This also implies that rule ordering is not important. Consequently, to make a policy decision requires processing all policy rules.

Additive permissions:

A query for access to data items is matched against the rules in the rule database. If several rules match, then the overall permissions granted to the WR are the union of those permissions. A more detailed discussion is provided inSection 10.

Upgradeable:

It should be possible to add additional rules later, without breaking PSs that have not been upgraded. Any such upgrades must not degrade privacy constraints, but PSs not yet upgraded may reveal less information than the rulemaker would have chosen.

Versioning support:

In addition to the previous goal, a RM should be able to determine which types of rules are supported by the PS. The mechanism used to determine the capability of a PS is outside the scope of this specification.

Protocol-independent:

The rule set supports constraints on both notifications or queries as well as subscriptions for event-based systems such as presence systems.

No false assurance:

It appears more dangerous to give the user the impression that the system will prevent disclosure automatically, but fail to do so with a significant probability of operator error or misunderstanding, than to force the user to explicitly invoke simpler rules. For example, rules based on weekday and time-of-day ranges seem particularly subject to misinterpretation and false assumptions on part of the RM. (For example, a non-technical RM would probably assume that the rules are based on the timezone of his current location, which may not be known to other components of the system.)

5. Non-Goals

We explicitly decided that a number of possibly worthwhile capabilities are beyond the scope of this first version. Future versions may include these capabilities, using the extension mechanism described in this document. Non-goals include:

No external references:

Attributes within specific rules cannot refer to external rule sets, databases, directories or other network elements. Any such external reference would make simple database implementation difficult and hence they are not supported in this version.

No regular expression or wildcard matching:

Conditions are matched on equality or 'greater-than'-style comparisons, not regular expressions, partial matches such as the SQL LIKE operator (e.g., LIKE "%foo%") or glob-style matches ("*@example.com"). Most of these are better expressed as explicit elements.

No all-except conditions:

It is not possible to express exclusion conditions based on identities such as "everybody except Alice". However, this restriction does not prevent all forms of blacklisting. It is still possible to express an authorization rule like 'I allow access to my location information for everyone of domain example.com except for John'. See the example in Section 7.1 describing how exceptions can be made to work. The reason for this choice is the ease with which identities can be manufactured, and the implication that all-except types of rules are easily subverted.

No repeat times:

Repeat times are difficult to make work correctly, due to the different time zones that PT, WR, PS and RM may occupy. It appears that suggestions for including time intervals are often based on supporting work/non-work distinctions, which unfortunately are difficult to capture by time alone.

6. Basic Data Model and Processing

A rule set (or synonymously, a policy) consists of zero or more rules. The ordering of these rules is irrelevant. The rule set can be stored at the PS and conveyed from RM to PS as a single document, in subsets or as individual rules. A rule consists of three parts - conditions (see Section 7), actions (see Section 8), and transformations (see Section 9).

The conditions part is a set of expressions, each of which evaluates to either TRUE or FALSE, i.e. each of which is equipped with a value of either TRUE or FALSE by the PS. When a WR asks for information about a PT, the PS goes through each rule in the rule set. For each rule, it evaluates the expressions in the conditions part. If all of the expressions evaluate to TRUE, then the rule is applicable to this request. Generally, each expression specifies a condition based on some variable that is associated with the context of the request. These variables can include the identity of the WR, the domain of the WR, the time of day, or even external variables, such as the temperature or the mood of the PT.

Assuming that the rule is applicable to the request, the actions and transformations (commonly referred to as permissions) in the rule specify how the PS is supposed to handle this request. If the request is to view the location of the PT, or to view its presence, the typical action is "permit", which allows the request to proceed.

Assuming the action allows the request to proceed, the transformations part of the rule specifies how the information about the PT - their location information, their presence, etc. - is modified before being presented to the WR. These transformations are in the form of positive permissions. That is, they always specify a piece of information which is allowed to be seen by the WR. When a PS processes a request, it takes the transformations specified across all rules that match, and creates the union of them. For computing this union the data type, such as Integer, Boolean, Set, or the Undef data type, plays a role. The details of the algorithm for combining permissions is described in Section 10. The resulting union effectively represents a "mask" - it defines what information is exposed to the WR. This mask is applied to the actual location or presence data for the PT, and the data which is permitted by the mask is shown to the WR. If the WR request a subset of information only (such as city-level civil location data only, instead of the full civil location information), the information delivered to the WR SHOULD be the intersection of the permissions granted to the WR and the data requested by the WR.

In accordance to this document, rules are encoded in XML. To this end, Section 13 contains an XML schema defining the Common Policy Markup Language. This, however, is purely an exchange format between RM and PS. The format does not imply that the RM or the PS use this format internally, e.g., in matching a query with the policy rules. The rules are designed so that a PS can translate the rules into a relational database table, with each rule represented by one row in the database. The database representation is by no means mandatory; we will use it as a convenient and widely-understood example of an internal representation. The database model has the advantage that operations on rows have tightly defined meanings. In addition, it appears plausible that larger-scale implementations will employ a backend database to store and query rules, as they can then benefit from existing optimized indexing, access control, scaling and integrity constraint mechanisms. Smaller-scale implementations may well choose different implementations, e.g., a simple traversal of the set of rules.

6.1. Identification of Rules

Each rule is equipped with a parameter that identifies the rule. This rule identifier is an opaque token chosen by the RM. A RM MUST NOT use the same identifier for two rules that are available to the PS at the same time for a given PT.

6.2. Extensions

The authorization policy framework defined in this document is meant to be extensible towards specific application domains. Such an extension is accomplished by defining conditions, actions and transformations that are specific to the desired application domain. Each extension MUST define its own namespace.

Extensions cannot change the schema defined in this document, and this schema is not expected to change excepting a revision to this specification, and that no versioning procedures for this schema or namespace are therfore provided.

7. Conditions

The access to data items needs to be matched with the rule set stored at the PS. Each instance of a request has different attributes (e.g., the identity of the requestor) which are used for authorization. A rule in a rule set might have a number of conditions which need to be met before executing the remaining parts of a rule (i.e., actions and transformations). Details about rule matching are described in Section 10. This document specifies only a few conditions (namely identity, sphere, and validity). Other conditions are left for extensions of this document.

7.1. Identity Condition

7.1.1. Overview

The identity condition restricts matching of a rule either to a single entity or a group of entitites. Only authenticated entities can be matched; acceptable means of authentication are defined in protocol-specific documents. As usual, if the <identity> element is (or all its child elements are) omitted, identities are not considered and, thus, the condition matches any user, authenticated or not.

The <identity> condition is considered TRUE if any of its child elements (i.e., the <one/> and the <many/> elements as defined in this document) evaluate to TRUE, i.e., the results of the individual child element are combined using a logical OR.

7.1.2. Matching One Entity

The <one> element matches the authenticated identity (as contained in the 'id' attribute) of exactly one entity or user. For considerations regarding the 'id' attribute refer to Section 7.1.3.

An example is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

    <rule id="f3g44r1">
        <conditions>
            <identity>
                <one id="sip:alice@example.com"/>
                <one id="tel:+1-212-555-1234" />
                <one id="mailto:bob@example.net" />
            </identity>
        </conditions>
        <actions/>
        <transformations/>
    </rule>

</ruleset>
```

This example matches if the authenticated identity of the WR is either sip:alice@example.com, tel:+1-212-555-1234 or mailto:bob@example.net.

7.1.3. Single Entity

The 'id' attribute used in the <one> and in the <except> element refers to a single entity. In the subsequent text we use the term 'single-user' element as a placeholder for the <one> and the <except> element.

The <single-user> element matches the authenticated identity (as contained in the 'id' attribute) of exactly one entity or user. The <single-user> element MUST NOT contain a 'domain' attribute.

The 'id' attribute contains an identity that MUST first be expressed as a URI. Applications using this framework must describe how the identities they are using can be expressed as a URIs.

7.1.4. Matching Multiple Entities

The <many> element is a mechanism to perform authorization decisions based on the domain part of the authenticated identity. As such, it allows to match a large and possibly unknown number of users within a domain.

Furthermore, it is possible to include one or multiple <except> elements to exclude either individual users or users belonging to a specific domain. Excluding individual entities is implemented using a <except id="..."/> statement. The semantic of the 'id' attribute of the <except> element has the same meaning as the 'id' attribute of the <one> element (see Section 7.1.3). Excluding users belonging to a specific domain is implemented using the <except domain="..."/> element that excludes any user from the indicated domain.

If multiple <except> elements are listed as child elements of the <many> element then the result of each <except> element is combined using a logical OR.

Common policy MUST either use UTF-8 or UTF-16 to store domain names in the 'domain' attribute. For non-IDNs, lower-case ASCII SHOULD be used. For the comparison operation between the value stored in the 'domain' attribute and the domain value provided via the using protocol (referred as "protocol domain identifier") the following rules are applicable:

1. If the values of the 'domain' attribute and the value of the protocol domain identifier does not begin with xn--, attempt a string comparison. If the string comparison indicates equality, the comparison succeeds and the remaining steps are skipped.

2. Translate percent-encoding for either string and repeat (1).

3. Convert both domain strings using the toASCII operation described in RFC 3490 [2]. (Naturally, if one of the strings already begins with the ACE prefix xn--, the conversion operation has already been performed.)

4. Compare the two domain strings for ASCII equality, for each label.

If the conversion fails in step (3), the domains are not equal.

7.1.4.1. Matching Any Authenticated Entity

The <many/> element without any child elements or attributes matches any authenticated user.

The following example shows such a rule that matches any authenticated user:

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

    <rule id="f3g44r5">
        <conditions>
            <identity>
              <many/>
            </identity>
        </conditions>
        <actions/>
        <transformations/>
    </rule>

</ruleset>
```

The following rule, in comparison, would match any user, authenticated and unauthenticated:

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

    <rule id="f3g44r5">
        <conditions>
          <identity/>
        </conditions>
        <actions/>
        <transformations/>
    </rule>

</ruleset>
```

7.1.4.2. Matching Any Identity Excepting Enumerated Domains and Identities

The <many> element enclosing one or more <except domain="..."/> elements matches any user from any domain except those enumerated. The <except id="..."/> element excludes particular users. The semantic of the 'id' attribute of the <except> element is described in Section 7.1.3. The results of the child elements of the <many> element are combined using a logical OR.

An example is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

    <rule id="f3g44r1">
        <conditions>
            <sphere value="work"/>
            <identity>
                <many>
                    <except domain="example.com"/>
                    <except domain="example.org"/>
                    <except id="sip:alice@bad.example.net"/>
                    <except id="sip:bob@good.example.net"/>
                    <except id="tel:+1-212-555-1234" />
                    <except id="sip:alice@example.com"/>
                </many>
            </identity>
            <validity>
                <from>2003-12-24T17:00:00+01:00</from>
                <until>2003-12-24T19:00:00+01:00</until>
            </validity>
        </conditions>
        <actions/>
        <transformations/>
    </rule>

</ruleset>
```

This example matches all users except any user in example.com, or any user in example.org or the particular users alice@bad.example.net, bob@good.example.net and the user with the telephone number 'tel:+1-212-555-1234'. The last 'except' element is redundant since alice@example.com is already excluded through the first line.

7.1.4.3. Matching Any Identity Within a Domain Excepting Enumerated Identities

The <many> element with a 'domain' attribute and zero or more <except id="..."/> elements matches any authenticated user from the indicated domain except those explicitly enumerated. The semantic of the 'id' attribute of the <except> element is described in Section 7.1.3.

An example is shown below:

```
   <?xml version="1.0" encoding="UTF-8"?>
   <ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

       <rule id="f3g44r1">
           <conditions>
               <identity>
                   <many domain="example.com">
                       <except id="sip:alice@example.com"/>
                       <except id="sip:bob@example.com"/>
                   </many>
               </identity>
           </conditions>
           <actions/>
           <transformations/>
       </rule>

   </ruleset>
```

This example matches any user within example.com (such as carol@example.com) except alice@example and bob@example.com.

7.2. Sphere

The <sphere> element belongs to the group of condition elements. It can be used to indicate a state (e.g., 'work', 'home', 'meeting', 'travel') the PT is currently in. A sphere condition matches only if the PT is currently in the state indicated. The state may be conveyed by manual configuration or by some protocol. For example, RPID [9] provides the ability to inform the PS of its current sphere. The application domain needs to describe in more detail how the sphere state is determined. Switching from one sphere to another causes to switch between different modes of visibility. As a result different subsets of rules might be applicable.

The content of the 'value' attribute of the <sphere> element MAY contain more than one token. The individual tokens MUST be separated by a blank character. A logical OR is used for the matching the tokens against the sphere settings of the PT. As an example, if the the content of the 'value' attribute in the sphere attribute contains two tokens 'work' and 'home' then this part of the rule matches if the sphere for a particular PT is either 'work' OR 'home'. To compare the content of the 'value' attribute in the <sphere> element with the stored state information about the PT's sphere setting a string comparison MUST be used for each individual token. There is no registry for these values nor a language specific indication of the sphere content. As such, the tokens are treated as opaque strings.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <ruleset xmlns="urn:ietf:params:xml:ns:common-policy">


     <rule id="f3g44r2">
       <conditions>
         <sphere value="work"/>
         <identity>
           <one id="sip:andrew@example.com"/>
         </identity>
       </conditions>
       <actions/>
       <transformations/>
     </rule>

     <rule id="y6y55r2">
       <conditions>
         <sphere value="home"/>
         <identity>
           <one id="sip:allison@example.com"/>
         </identity>
       </conditions>
       <actions/>
       <transformations/>
     </rule>

     <rule id="z6y55r2">
       <conditions>
         <identity>
              <one id="sip:john@doe.com"/>
         </identity>
         <sphere value="home work"/>
       </conditions>
       <actions/>
       <transformations/>
     </rule>
   </ruleset>
```

The rule example above illustrates that the rule with the entity andrew@example.com matches if the sphere is been set to 'work'. In the second rule with the entity allison@example.com matches if the sphere is set to 'home'.

7.3. Validity

The <validity> element is the third condition element specified in this document. It expresses the rule validity period by two attributes, a starting and a ending time. The validity condition is

TRUE if the current time is greater than or equal to at least one <from> child, but less than the <until> child after it. This represents a logical OR operation across each <from> and <until> pair. Times are expressed in XML dateTime format. A rule maker might not have always access to the PS to invalidate some rules which grant permissions. Hence this mechanisms allows to invalidate granted permissions automatically without further interaction between the rule maker and the PS. The PS does not remove the rules instead the rule maker has to clean them up.

An example of a rule fragment is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

  <rule id="f3g44r3">
    <conditions>
        <validity>
            <from>2003-08-15T10:20:00.000-05:00</from>
            <until>2003-09-15T10:20:00.000-05:00</until>
        </validity>
    </conditions>
    <actions/>
    <transformations/>
  </rule>
</ruleset>
```

The <validity> element MUST have the <from> and <until> subelements in pairs. Multiple <from> and <until> elements might appear in pairs (i.e., without nesting of <from> and <until> elements). Using multiple <validity> elements as subelements of the <conditions> element is not useful since all subelements of the <conditions> element are combined as a logical AND.

8. Actions

While conditions are the 'if'-part of rules, actions and transformations build the 'then'-part of them. The actions and transformations parts of a rule determine which operations the PS MUST execute after having received from a WR a data access request that matches all conditions of this rule. Actions and transformations only permit certain operations; there is no 'deny' functionality. Transformations exclusively specify PS-side operations that lead to a modification of the data items requested by the WR. Regarding location data items, for instance, a transformation could force the PS to lower the precision of the location information which is returned to the WR.

Actions, on the other hand, specify all remaining types of operations the PS is obliged to execute, i.e., all operations that are not of transformation type. Actions are defined by application specific usages of this framework. The reader is referred to the corresponding extensions to see examples of such elements.

9. Transformations

Two sub-parts follow the conditions part of a rule: transformations and actions. As defined in Section 8, transformations specify operations that the PS MUST execute and that modify the result which is returned to the WR. This functionality is particularly helpful in reducing the granularity of information provided to the WR, as for example required for location privacy. Transformations are defined by application specific usages of this framework.

A simple transformation example is provided in Section 10.

10. Procedure for Combining Permissions

10.1. Introduction

This section describes the mechanism to evaluate the final result of a rule evaluation. The result is reflected in the action and transformation part of a rule. This procedure is sometimes referred as conflict resolution.

We use the following terminology (which in parts has already been introduced in previous sections): The term 'permission' stands for an action or a transformation. The notion 'attribute' terms a condition, an action, or a transformation. An attribute has a name, has a certain data type. A value may be assigned to an attribute or it may be undefined, in case it does not have a value associated with the attribute. For example, the name of the <sphere> attribute discussed in Section 7 is 'sphere', its data type is 'string', and its value may be set to 'home'. To evaluate a condition means to associate either TRUE or FALSE to the condition. Please note that the <identity> element is a condition whereas the <id> element is a parameter of that condition. A rule matches if all conditions contained in the conditions part of a rule evaluate to TRUE.

When the PS receives a request for access to privacy-sensitive data then it needs to be matched against a rule set. The conditions part of each individual rule is evaluated and as a result one or more rules might match. If only a single rule matches then the result is determined by executing the actions and the transformations part following the conditions part of a rule. However, it can also be the case that two or more matching rules contain a permission of the same name (e.g., two rules contain a permission named 'precision of geospatial location information'), but do not specify the same value for that permission (e.g., the two rule might specify values of '10 km' and '200 km', respectively, for the permission named 'precision of geospatial location information'). This section describes the procedure for combining permissions in such cases.

10.2. Algorithm

The combining rules are simple and depend on the data types of the values of permissions: Let P be a policy. Let M be the subset of P consisting of rules r in P that match with respect to a given request. Let n be a name of a permission contained in a rule r in M, and let M(n) be the subset of M consisting of rules r in M that have a permission of name n. For each rule r in M(n), let v(r,n) and d(r,n) be the value and the data type, respectively, of the attribute of r with name n. Finally, let V(n) be the combined value of all the permissions values v(r,n), r in M(n). The combining rules that lead to the resulting value V(n) are the following:

CR 1: If d(r,n)=Boolean for all r in M(n), then V(n) is given as follows: If there is a r in M(n) with v(r,n)=TRUE, then V(n)=TRUE. Otherwise, V(n)=FALSE.

CR 2: If d(r,n)=Integer for all r in M(n), then V(n) is given as follows: If v(r,n)=undefined for all r in M(n), then V(n) is not specified by this specification. Otherwise, V(n)=max{v(r,n) | r in M(n)}.

CR 3: If d(r,n)=Set for all r in M(n), then V(n) is given as follows: V(n)=union of all v(r,n), the union to be computed over all r in M(n) with v(r,n)!=undefined.

The combining operation will result in the largest value for an Integral type, the Or operation for boolean, and union for set.

As a result, applications should define values such that, for integers, the lowest value corresponds to the most privacy, for booleans, false corresponds to the most privacy, and for sets, the empty set corresponds to the most privacy. More

10.3. Example

In the following example we illustrate the process of combining permissions. We will consider three conditions for our purpose, namely those of name identity, sphere, and validity. For editorial reasons the rule set in this example is represented in a table. Furthermore, the domain part of the identity of the WR is omitted. For actions we use two permissions with names X and Y. The values of X and Y are of data types Boolean and Integer, respectively. Permission X might, for example, represent the <sub-handling> action. For transformations we use the attribute with the name Z whose value can be set either to '+'(or 1), 'o' (or 2) or '-' (or 3). Permission Z allows us to show the granularity reduction whereby a value of '+' shows the corresponding information unrestricted and '-' shows nothing. This permission might be related to location information or other presence attributes like mood. Internally we use the data type Integer for computing the permission of this attribute.

Conditions					Actions/Transformations		
Id	WR-ID	sphere	from	until	X	Y	Z
1	bob	home	A1	A2	TRUE	10	o
2	alice	work	A1	A2	FALSE	5	+
3	bob	work	A1	A2	TRUE	3	-
4	tom	work	A1	A2	TRUE	5	+
5	bob	work	A1	A3	undef	12	o
6	bob	work	B1	B2	FALSE	10	-

Again for editorial reasons, we use the following abbreviations for the two <validity> attributes 'from' and 'until':

```
A1=2003-12-24T17:00:00+01:00
A2=2003-12-24T21:00:00+01:00
A3=2003-12-24T23:30:00+01:00
B1=2003-12-22T17:00:00+01:00
B2=2003-12-23T17:00:00+01:00
```

Note that B1 < B2 < A1 < A2 < A3.

The entity 'bob' acts as a WR and requests data items. The policy P consists of the six rules shown in the table and identified by the values 1 to 6 in the 'Id' column. The PS receives the query at 2003-12-24T17:15:00+01:00 which falls between A1 and A2. The value of the attribute with name 'sphere' indicating the state the PT is currently in is set to 'work'.

Rule 1 does not match since the sphere condition does not match. Rule 2 does not match as the identity of the WR (here 'alice') does not equal 'bob'. Rule 3 matches since all conditions evaluate to TRUE. Rule 4 does not match as the identity of the WR (here 'tom') does not equal 'bob'. Rule 5 matches. Rule 6 does not match since the rule is not valid anymore. Therefore, the set M of matching rules consists of the rules 3 and 5. These two rules are used to compute the combined permission V(X), V(Y), and V(Z) for each of the permissions X, Y, and Z:

	Actions/Transformations		
Id	X	Y	Z
3	TRUE	3	-
5	undef	12	o

The results of the permission combining algorithm is shown below. The combined value V(X) regarding the permission with name X equals TRUE according to the first combining rule listed above. The maximum of 3 and 12 is 12, so that V(Y)=12. For the attribute Z in this example the maximum between 'o' and '-' (i.e., between 2 and 3) is '-'.

Actions/Transformations

Id	X	Y	Z
5	TRUE	12	-

11. Meta Policies

Meta policies authorize a rulemaker to insert, update or delete a particular rule or an entire rule set. Some authorization policies are required to prevent unauthorized modification of rule sets. Meta policies are outside the scope of this document.

A simple implementation could restrict access to the rule set only to the PT but more sophisticated mechanisms could be useful. As an example of such policies one could think of parents configuring the policies for their children.

12. Example

This section gives an example of an XML document valid with respect to the XML schema defined in Section 13. Semantically richer examples can be found in documents which extend this schema with application domain specific data (e.g., location or presence information).

Below a rule is shown with a condition that matches for a given authenticated identity (bob@example.com) and within a given time period. Additionally, the rule matches only if the target has set its sphere to 'work'.

```
<?xml version="1.0" encoding="UTF-8"?>
<ruleset xmlns="urn:ietf:params:xml:ns:common-policy">

    <rule id="f3g44r1">
        <conditions>
            <identity>
                <one id="sip:bob@example.com"/>
            </identity>
            <sphere value="work"/>
            <validity>
                <from>2003-12-24T17:00:00+01:00</from>
                <until>2003-12-24T19:00:00+01:00</until>
            </validity>
        </conditions>
        <actions/>
        <transformations/>
    </rule>
</ruleset>
```

13. XML Schema Definition

This section provides the XML schema definition for the common policy markup language described in this document.

```
   <?xml version="1.0" encoding="UTF-8"?>
   <xs:schema targetNamespace="urn:ietf:params:xml:ns:common-policy"
       xmlns:cp="urn:ietf:params:xml:ns:common-policy"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       elementFormDefault="qualified" attributeFormDefault="unqualified">
       <!-- /ruleset -->
       <xs:element name="ruleset">
           <xs:complexType>
               <xs:complexContent>
                   <xs:restriction base="xs:anyType">
                       <xs:sequence>
                           <xs:element name="rule" type="cp:ruleType"
                           minOccurs="0" maxOccurs="unbounded"/>
                       </xs:sequence>
                   </xs:restriction>
               </xs:complexContent>
           </xs:complexType>
       </xs:element>
       <!-- /ruleset/rule -->
       <xs:complexType name="ruleType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:sequence>
                       <xs:element name="conditions"
                       type="cp:conditionsType" minOccurs="0"/>
                       <xs:element name="actions"
                       type="cp:extensibleType" minOccurs="0"/>
                       <xs:element name="transformations"
                       type="cp:extensibleType" minOccurs="0"/>
                   </xs:sequence>
                   <xs:attribute name="id" type="xs:ID" use="required"/>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //rule/conditions -->
       <xs:complexType name="conditionsType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:choice maxOccurs="unbounded">
                       <xs:element name="identity"
                       type="cp:identityType" minOccurs="0"/>
                       <xs:element name="sphere"
                       type="cp:sphereType" minOccurs="0"/>
                       <xs:element name="validity"
                       type="cp:validityType" minOccurs="0"/>
                       <xs:any namespace="##other" processContents="lax"
                       minOccurs="0" maxOccurs="unbounded"/>
                   </xs:choice>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //conditions/identity -->
       <xs:complexType name="identityType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:choice  minOccurs="0" maxOccurs="unbounded">
                       <xs:element name="one" type="cp:oneType"/>
                       <xs:element name="many" type="cp:manyType"/>
                   </xs:choice>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //identity/one -->
       <xs:complexType name="oneType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:sequence>
                       <xs:any namespace="##other"
                       minOccurs="0" processContents="lax"/>
                   </xs:sequence>
                   <xs:attribute name="id"
                   type="xs:anyURI" use="required"/>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //identity/many -->
       <xs:complexType name="manyType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:choice minOccurs="0" maxOccurs="unbounded">
                       <xs:element name="except" type="cp:exceptType"/>
                       <xs:any namespace="##other"
                       minOccurs="0" processContents="lax"/>
                   </xs:choice>
                   <xs:attribute name="domain"
                   use="optional" type="xs:string"/>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //many/except -->
       <xs:complexType name="exceptType">
           <xs:attribute name="domain" type="xs:string" use="optional"/>
           <xs:attribute name="id" type="xs:anyURI" use="optional"/>
       </xs:complexType>
       <!-- //conditions/sphere -->
       <xs:complexType name="sphereType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:attribute name="value"
                   type="xs:string" use="required"/>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //conditions/validity -->
       <xs:complexType name="validityType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:sequence minOccurs="0" maxOccurs="unbounded">
                       <xs:element name="from" type="xs:dateTime"/>
                       <xs:element name="until" type="xs:dateTime"/>
                   </xs:sequence>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
       <!-- //rule/actions or //rule/transformations -->
       <xs:complexType name="extensibleType">
           <xs:complexContent>
               <xs:restriction base="xs:anyType">
                   <xs:sequence>
                       <xs:any namespace="##other"
                       processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
                   </xs:sequence>
               </xs:restriction>
           </xs:complexContent>
       </xs:complexType>
   </xs:schema>
```

14. Security Considerations

This document describes a framework for authorization policy rules. This framework is intended to be enhanced elsewhere towards application domain specific data. Security considerations are to a great extent application data dependent, and therefore need to be covered by documents that extend the framework defined in this specification. However, new action and transformation permissions along with their allowed values must be defined in a way so that the usage of the permissions combining rules of Section 10 does not lower the level of privacy protection. See Section 10 for more details on this privacy issue.

15. IANA Considerations

This section registers a new XML namespace, a new XML schema and a new MIME-type. This section registers a new XML namespace per the procedures in [3].

15.1. Common Policy Namespace Registration

URI: urn:ietf:params:xml:ns:common-policy

Registrant Contact: IETF Geopriv Working Group, Henning Schulzrinne (hgs+geopriv@cs.columbia.edu).

XML:

```
BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
  "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type"
        content="text/html;charset=iso-8859-1"/>
  <title>Common Policy Namespace</title>
</head>
<body>
  <h1>Namespace for Common Authorization Policies</h1>
  <h2>urn:ietf:params:xml:ns:common-policy</h2>
<p>See <a href="[URL of published RFC]">RFCXXXX
    [NOTE TO IANA/RFC-EDITOR:
     Please replace XXXX with the RFC number of this
    specification.]</a>.</p>
</body>
</html>
END
```

15.2. Content-type registration for 'application/auth-policy+xml'

This specification requests the registration of a new MIME type according to the procedures of RFC 2048 [4] and guidelines in RFC 3023 [5].

MIME media type name: application

MIME subtype name: auth-policy+xml

Mandatory parameters: none

Optional parameters: charset

Indicates the character encoding of enclosed XML. Default is UTF-8.

Encoding considerations:

Uses XML, which can employ 8-bit characters, depending on the character encoding used. See RFC 3023 [5], Section 3.2.

Security considerations:

This content type is designed to carry authorization policies. Appropriate precautions should be adopted to limit disclosure of this information. Please refer to RFCXXXX [NOTE TO IANA/ RFC-EDITOR: Please replace XXXX with the RFC number of this specification.] security considerations section for more information.

Interoperability considerations: none

Published specification: RFCXXXX [NOTE TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this specification.] this document

Applications which use this media type:

Presence- and location-based systems

Additional information:

Magic Number: None

File Extension: .xml

Macintosh file type code: 'TEXT'

Personal and email address for further information: Hannes Tschofenig, Hannes.Tschofenig@siemens.com

Intended usage: LIMITED USE

Author/Change controller:

This specification is a work item of the IETF GEOPRIV working group, with mailing list address <geopriv@ietf.org>.

15.3. Common Policy Schema Registration

URI: urn:ietf:params:xml:schema:common-policy

Registrant Contact: IETF Geopriv Working Group, Henning Schulzrinne (hgs+geopriv@cs.columbia.edu).

XML: The XML schema to be registered is contained in Section 13. Its first line is

```
<?xml version="1.0" encoding="UTF-8"?>
```

and its last line is

```
</xs:schema>
```

16. References

16.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

[2] Faltstrom, P., Hoffman, P., and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[3] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[4] Freed, N., Klensin, J., and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", BCP 13, RFC 2048, November 1996.

[5] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", RFC 3023, January 2001.

16.2. Informative References

[6] Rosenberg, J., "Presence Authorization Rules", draft-ietf-simple-presence-rules-04 (work in progress), October 2005.

[7] Schulzrinne, H., "A Document Format for Expressing Privacy Preferences for Location Information", draft-ietf-geopriv-policy-07 (work in progress), October 2005.

[8] Cuellar, J., Morris, J., Mulligan, D., Peterson, J., and J. Polk, "Geopriv Requirements", RFC 3693, February 2004.

[9] Schulzrinne, H., "RPID: Rich Presence Extensions to the Presence Information Data Format (PIDF)", draft-ietf-simple-rpid-10 (work in progress), December 2005.

[10] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

Appendix A. Contributors

We would like to thank Christian Guenther for his help with initial versions of this document.

Appendix B. Acknowledgments

This document is partially based on the discussions within the IETF GEOPRIV working group. Discussions at the Geopriv Interim Meeting 2003 in Washington, D.C., helped the working group to make progress on the authorization policies based on the discussions among the participants.

We particularly want to thank Allison Mankin <mankin@psg.com>, Randall Gellens <rg+ietf@qualcomm.com>, Andrew Newton <anewton@ecotroph.net>, Ted Hardie <hardie@qualcomm.com>, Jon Peterson <jon.peterson@neustar.biz> for discussing a number of details with us. They helped us to improve the quality of this document. Allison, Ted and Andrew helped us to make good progress with the internationalization support of the identifier/domain attributes.

Furthermore, we would like to thank the IETF SIMPLE working group for their discussions of J. Rosenberg's draft on presence authorization policies. We would also like to thank Stefan Berg, Murugaraj Shanmugam, Christian Schmidt, Martin Thomson, Markus Isomaki, Aki Niemi and Eva Maria Leppanen for their comments. Martin Thomson helped us with the XML schema.

Authors' Addresses

Henning Schulzrinne
Columbia University
Department of Computer Science
450 Computer Science Building
New York, NY 10027
USA

Phone: +1 212 939 7042
Email: schulzrinne@cs.columbia.edu
URI: http://www.cs.columbia.edu/~hgs

John B. Morris, Jr.
Center for Democracy and Technology
1634 I Street NW, Suite 1100
Washington, DC 20006
USA

Email: jmorris@cdt.org
URI: http://www.cdt.org

Hannes Tschofenig
Siemens
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: Hannes.Tschofenig@siemens.com
URI: http://www.tschofenig.com

Jorge R. Cuellar
Siemens
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: Jorge.Cuellar@siemens.com

James Polk
Cisco
2200 East President George Bush Turnpike
Richardson, Texas 75082
USA

Email: jmpolk@cisco.com

Jonathan Rosenberg
Cisco Systems
600 Lanidex Plaza
Parsippany, New York 07054
USA

Email: jdrosen@cisco.com
URI: http://www.jdrosen.net

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.