

Internet Engineering Task Force
Internet Draft

Geopriv
H. Tschofenig
J. Cuellar
Siemens

Document:

[draft-tschofenig-geopriv-authz-policies-00.txt](#)

Expires: December 2003

June 2003

Geopriv Authorization Policies
<[draft-tschofenig-geopriv-authz-policies-00.txt](#)>

Status of this Memo

This document is an Internet-Draft and is subject to all provisions of [Section 10 of RFC2026](#).

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/lid-abstracts.html>

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

Abstract

This document describes authorization policies for usage with Geopriv. It suggests using the eXtensible Access Control Markup Language (XACML). XACML provides functionality required to express policies for access to location information.

Geopriv Authorization Policies

June 2003

Table of Contents

1.	Introduction.....	2
2.	Terminology.....	3
3.	The Process of Access Control.....	3
4.	Features provided by XACML.....	6
4.1	Basic functionality of XACML.....	6
4.2	Combining Algorithms.....	7
4.3	Operators.....	8
5.	Examples.....	9
5.1	Request Context Example 1.....	9
5.2	Policy Example 1.....	10
5.3	Reponse Context Example 1.....	12
6.	Security Considerations.....	12
7.	Open Issues.....	13
8.	Acknowledgments.....	13
9.	References.....	13
	Author's Addresses.....	15

[1.](#) Introduction

Geopriv provides Location Information in a secure and private way.

A critical role is played by user-controlled Privacy Rules, which describe the restrictions imposed or permissions given by the Rule Maker. The Privacy Rules specify the necessary conditions that allow a Location Server to forward Location Information to a Location Recipient, and the conditions under which and purposes for which the Location Information can be used.

One type of Privacy Rules specify in particular how location information should be filtered, depending on who the recipient is. Filtering is the process of reducing the precision or resolution of the data. A typical rule may be of the form: "my location can only be disclosed to the owner of such credentials in such precision or resolution" (e.g., "my co-workers can be told the city I am currently in").

The Location Object should be able to carry a limited but core set of Privacy Rules.

The access to location information (as XML objects) can be

controlled by XACML policies. The same is true for writing and deleting Geopriv rules themselves. The Geopriv working group can benefit from reusing existing work on access control.

This document therefore aims to provide the following description:

- a) It introduces the reader to XACML and describes some of the relevant features.
- b) It explores how access control is accomplished in the Geopriv environment.
- c) It gives examples of access control policies.

2. Terminology

This draft uses terminology described in [CM+03] and in [[XACML](#)].

3. The Process of Access Control

Figure 1 gives an abstract overview of the interaction between the participating entities. Different actions are performed by different entities requesting access to different information items stored at the location server. The location server needs to store information about users, group of users (including pseudonyms and roles). These entities are subject to authorization (among other attributes).

The request from an entity to the Policy Decision Point, which is in this case the location server, is carried over a Geopriv 'using protocol'. A number of protocols might serve as a using protocol such as the Presence protocol as described in [[Pet03](#)]. Recently J. Rosenberg published the XML Configuration Access Protocol (XCAP) [[XCAP](#)] which describes how to read, write and delete application configuration data stored in XML format. XCAP can also be used to manipulate XML based location information (see [[CG03](#)]) and it is therefore another building block on the way to complete the work on the Geopriv framework. Instantiation of XCAP for usage with SIP in the area of presence (e.g. modification of buddy lists) can be found in [[Ros03a](#)] and in [[Ros03b](#)]. XCAP, however, does not describe how to control access to perform certain operations on these documents in a sophisticated fashion. This draft adds this missing piece.

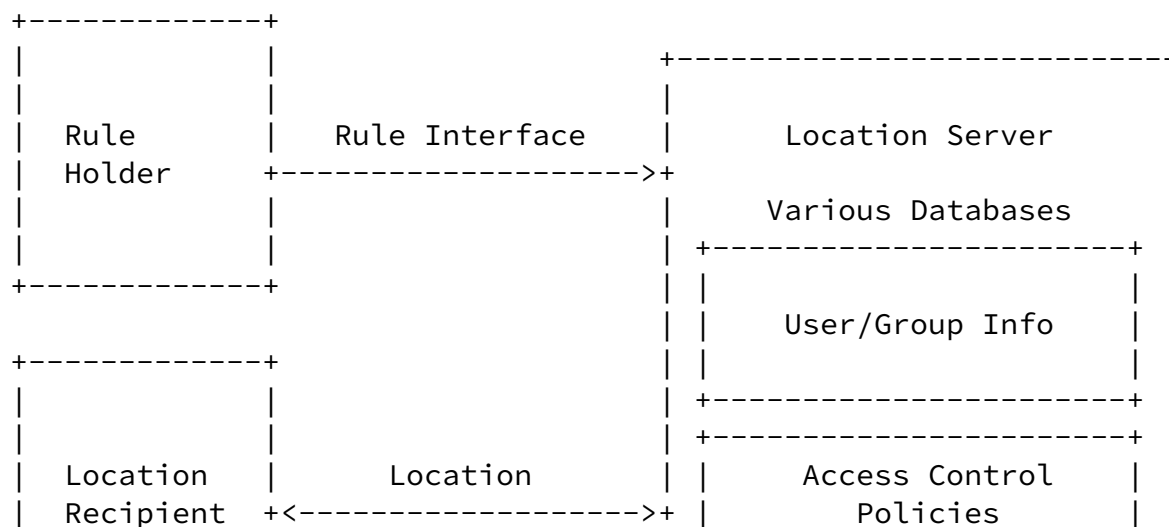
After the request reaches the location server it is necessary to compare the request with a number of policies deciding whether the operation is permitted or denied. More information on this step is given with the description of Figure 2 This verification requires matching a request with a number of rules and some sort of conflict resolution. For combining algorithms provided by XACML, the details of conflict resolution are given in [Section 4.2](#).

Access control policies for location information demand a flexible language to express the different needs such as:

- My boss should be granted access to my location (civil location, building and room number only) only during working hours. My family is always allowed to access my exact location.
- My students are allowed to access my location during the week if they are in possession of an authorization token.

Furthermore it is necessary to specify policies even for access to the policies itself. (But this MAY be out of scope of the Geopriv WG).

Finally, it must be possible to cope with information stored in XML format.



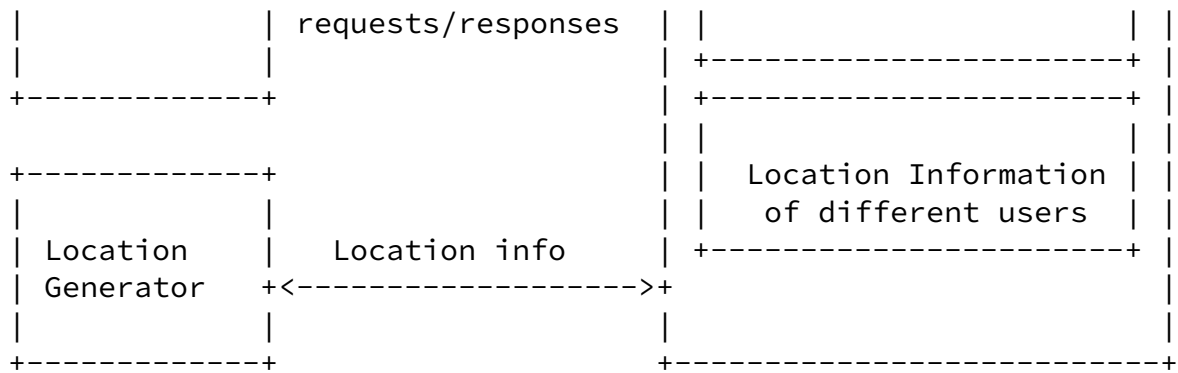


Figure 1: Interaction between the participating Entities

Access control policies require a number of input parameters to compute the authorization decision. Some of these input parameters can be obtained from the request itself (i.e. from domain-specific input). To be accessible to the policy engine (as described in the policy rules) it is convenient to have the context request defined in an XACML schema.

Based on Figure 2 of [[XACML](#)] the relationship to Geopriv is drawn.

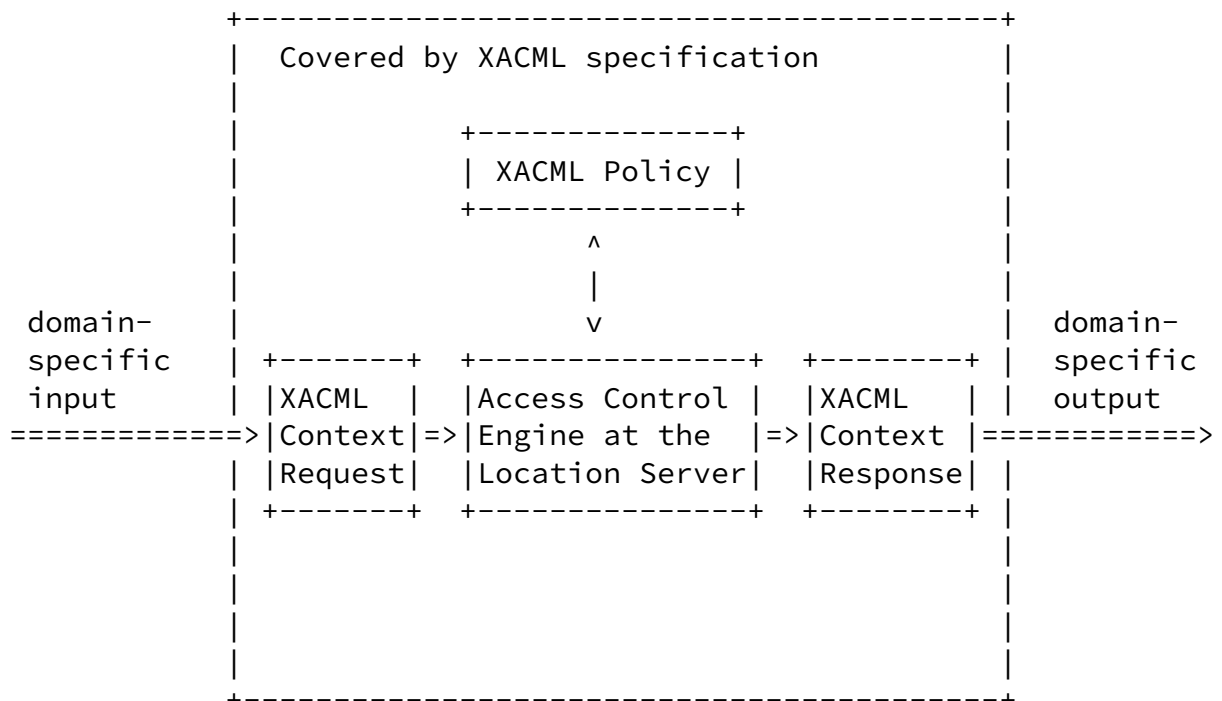


Figure 2: XACML Context

The access control engine evaluates incoming requests from specific subjects with specific actions (read, write, delete) for certain objects. For Geopriv some of these objects are location information "documents", as for example defined, in [\[CG03\]](#). Location information is stored in XML documents.

As shown in Figure 2 a native request is mapped into an XML request (following a XACML schema) which allows the access control engine and the policies to refer to certain items of the request. The XACML Context Request identifies, for example, which elements the subject wants to read. It provides information about the requesting entity such as role, subject identity, information about the security protocols used to authenticate, which credentials have been used, etc. The Geopriv group might want to evaluate which information items are typically associated with a request to re-evaluate the XACL Context schema definition.

Note that attributes might appear in different envelopes such as X.509 attribute certificates or as Security Assertion Markup Language (SAML) assertions [\[SAML\]](#). SAML allows distributing security information. Thereby the security information is expressed as assertions about subjects. These assertions carry information about authentication acts, attributes and authorization decisions. Conversion between the native format and the XACML context might be specified with the help of the Extensible Stylesheet Language Transformation [\[XSLT\]](#).

[4.](#) Features provided by XACML

This section describes some of the features which make it an attractive access control markup language for Geopriv policies. Implementing a new access control markup language solely for Geopriv without reusing existing systems requires recreating the same functions and concepts. It seems that the solution space for standardized languages is actually fairly small when designing a flexible access control mechanism for XML based documents.

[4.1](#) Basic functionality of XACML

Three top-level elements are defined in XACML [\[XACML\]](#): <Rule>,

<Policy> and <PolicySet>

The policy language model is defined in [[XACML](#)], Section 3.3 (see also Figure 3 there).

A <Rule> contains a Boolean expression and consists of the following components:

- Target
- Effect and
- Condition

The rule target defines a set of resources, subjects and actions to which the rule applies. The rule itself forms the basic building block for a Policy.

The rule target consists of

- Resources
- Subjects and
- Actions

Subjects and resources might be internally structured. This allows referencing them via an XPath [[XPath](#)] or XPointer [[XPointer](#)]

reference. Referring to resources could imply to

- refer to the content of the identified node itself
- refer to the content of the identified node and its immediate child nodes or
- refer to the content of the identified node and all its descendant nodes.

It is therefore also possible to base the authorization decision on data contained in the resource to which access requested.

The effect describes the intended consequence of a "true" evaluation. Two values are defined "permit" and "deny".

The condition furthermore allows refining the applicability of a rule. This element is optional.

Rules themselves are not exchanged between entities. Therefore <Rules> are always embedded into a <Policy>. A <Policy> consists of

- a target
- a rule-combining algorithm id
- a set of rules and
- obligations.

<Policy> elements might additionally attach to an information resource itself. One application would be to attach the policy itself to Geopriv location object.

Obligations allow to enforce certain actions that must be performed either instead or in addition to the actions that may be performed. This functionality simulates the idea of a provisional action as described in [[Kudo00](#)]. Provisions are attached to the access decision.

An example for such an access decision might be a logging action or the encryption of the objects returned in response.

A <PolicySet> again allows combining set of Policies together with the following elements:

- a target
- a policy-combining algorithm id
- obligations.

[4.2](#) Combining Algorithms

The rule-combining algorithm identifier is used to implement conflict resolution (i.e. a mechanism for achieving a final authorization decision after evaluating individual rules) with the following standard algorithms:

- * Deny-overrides
- * Permit-overrides
- * First applicable and
- * Only-one-applicable

The policy-combining algorithm identifier enforces the same concept as the rule-combining algorithm identifier but at a policy and not a rule level.

An example: The permit-overrides algorithm causes a "permit" evaluation if a single rule (or policy) in the applicable policy is found with a "permit" evaluation, regardless of other rules or policies in the applicable policy.

New rule-combining algorithms (or policy-combining algorithms) can be specified, if necessary.

[4.3](#) Operators

In order to compute an authorization decision it is necessary to perform operations on attributes of subjects and resources. XACML defines a number of operators:

- Operators on numerical data types

Examples: integer-add, double-add, integer-subtract, integer-multiply, double-mod, integer-divide, round, floor, integer-abs, double-to-integer, etc.

Additionally arithmetic comparison functions are defined such as integer-less-than, double-greater-than-or-equal, etc.

Various non-numeric comparison functions also exist such as string-greater-than, string-less-than, time-less-than, date-greater-than, etc.

- Operators on date, time and duration data types

Examples: dateTime-subtract-yearMonthDuration, dateTime-add-dayTimeDuration, date-subtract-yearMonthDuration

- Operators on Boolean data types

Examples: and, or, not, n-of

These operators allow specifying logical combination of predicates of a rule.

- Operators on sets

Examples: type-bag-size, type-is-in, type-intersection, type-union, type-subset, type-set-equals, any-of, all-of, any-of-any, map, etc.

- Relationship operators

Examples: Equality and comparison on [RFC 822](#) and X.500 name forms, strings, URIs, etc. (string-equal, Boolean-equal, integer-equal, date-equal, time-equal, X500Name-equal, rfc822Name-equal, anyURI-equal, etc.)

Additionally it is possible to add non-standard functions.

5. Examples

This section shows some examples for the XACML policies. For this version of the draft some very basic examples are provided. Currently the examples are very much aligned to the examples in Section 4 of [[XACML](#)]. Comments are inserted into the XML code within "`<!--`" and "`-->`".

5.1 Request Context Example 1

The request in XACML for the first example might be stated as follows:

"Jorge Cuellar wants to know (i.e. read) the location of Hannes Tschofenig. Jorge Cuellar is identified with an [RFC 822](#) type of email address (for example obtained during the SIP authentication procedure)".

This request therefore translates to the following XACML request context:

```
<?xml version="1.0" encoding="UTF-8"?>
<Request xmlns="urn:oasis:names:tc:xacml:1.0:context"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context
http://www.oasis-open.org/tc/xacml/1.0/cs-xacml-schema-context-01.xsd">

<!-- These lines show the header of the request context. -->

<Subject>
  <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
  DataType="urn:oasis:names:tc:xacml:1.0:data-type:rfc822Name">
    <AttributeValue>Jorge.Cuellar@siemens.com</AttributeValue>
  </Attribute>
</Subject>
```

<!-- The <Subject> element describes information about the requesting entity. Multiple subjects and multiple attributes per

subject are possible. -->

```
<Resource>
  <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:resource:ufs-path"
  DataType="http://www.w3.org/2001/XMLSchema#anyURI">
```

```
<AttributeValue>/geopriv/HannesTschofenig@siemens.com</AttributeValue>
  </Attribute>
</Resource>
```

<!-- The <Resource> element indicates to which resource access is requested. For this example it is assumed that the location information is stored in a directory oriented (pathname /geopriv/HannesTschofenig@siemens.com) fashion. To support compatibility it is required to agree on a common structure for referencing the required elements. -->

```
<Action>
  <Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
  DataType="http://www.w3.org/2001/XMLSchema#string">
    <AttributeValue>read</AttributeValue>
  </Attribute>
</Action>
</Request>
```

<!-- The <Action> element describes the desired action. In this case a "read" action is requested to retrieve location information. -->

[5.2](#) Policy Example 1

The request described in [Section 5.1](#) is then matched against a policy at the location server. This policy might, for example, be provided by the owner of the location itself.

```
<?xml version=1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:1.0:policy"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:policy
http://www.oasis-open.org/tc/xacml/1.0/cs-xacml-schema-policy-
```

[01.xsd](#)"

PolicyId="identifier:example:GeoprivPolicy1"

RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">

<!-- This part of the policy represents header information containing information about the encoding, the URI to the XACML policy schema, a unique policy identifier and finally the rule combining algorithm identifier. The deny-overrides policy forces a "deny" to be returned when any of the policy rules evaluates to "deny". -->

<Description>

Simple Geopriv policy.

</Description>

<!-- An optional description of the policy. -->

<Target>

<Subjects>

<AnySubject/>

</Subjects>

<Resources>

<AnyResource/>

</Resources>

<Actions>

<AnyAction/>

</Actions>

</Target>

<!-- The <Target> statement indicates to which policies this request applies. It allows to create an index to policies. -->

<Rule

RuleId= "urn:oasis:names:tc:xacml:1.0:example:GeoprivRule1"

Effect="Permit">

<Description>

Every subject with an identifier in the siemens.com domain is allowed to read the location.

</Description>

<!-- This XML text introduces a single rule which shows the effect of a "true" evaluation of the rule. The effect of this rule is "Permit". Again, the description is optional. -->

```
<Target>
  <Subjects>
    <Subject>
      <SubjectMatch MatchId="
urn:oasis:names:tc:xacml:1.0:function:rfc822Name-match">
        <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
          DataType="urn:oasis:names:tc:xacml:1.0:data-
type:rfc822Name"/>
        <AttributeValue
          DataType="urn:oasis:names:tc:xacml:1.0:data-
type:rfc822Name">siemens.com
        </AttributeValue>
      </SubjectMatch>
    </Subject>
  </Subjects>
```

```
<Resources>
  <AnyResource/>
</Resources>
<Actions>
<Attribute
AttributeId="urn:oasis:names:tc:xacml:1.0:action:action-id"
  DataType="http://www.w3.org/2001/XMLSchema#string">
  <AttributeValue>read</AttributeValue>
</Attribute>
</Actions>
</Target>
```

<!-- Similarly as above the <Target> statement describes to whether the remainder of the rule have to be evaluated after matching the subject, resource and action part of the <Target>. This rule matches the subject-id of the requestor with "siemens.com". The <SubjectMatch MatchId=...> specifies how matching should be performed and what data types are expected. Furthermore it returns "Permit" only if the requested action is "read". No restrictions are made regarding the resources (i.e. any location information might be accessed). -->

```
</Rule>
</xacml:Policy>
```

[5.3](#) Reponse Context Example 1

The following XML text might represent a possible reponse context to the request described in this example.

```
<?xml version="1.0" encoding="UTF-8"?>
<Response xmlns="urn:oasis:names:tc:xacml:1.0:context"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:oasis:names:tc:xacml:1.0:context
http://www.oasis-open.org/tc/xacml/1.0/cs-xacml-schema-context-01.xsd">

<!-- These lines show the header of the response context. -->

<Result>
  <Decision>NotApplicable</Decision>
</Result>

<!-- These lines show the actual response after evaluation of the
request against the policy rules. -->
```

[6.](#) Security Considerations

<TBD>

Refer to the threats doc here [DM+03]. There may be also some additional threats to the xacml solution.

[7.](#) Open Issues

This draft is a first discussion starter. As such it creates a number of open issues, such as:

- What attributes are useful for location requests? What information should typically serve as input to the authorization engine?
- As soon as the work on the location object makes progress it is necessary to describe how to access the attributes of the location object.

- More sophisticated Geopriv example policies have to be added to test the adequacy of the capabilities of XACML.
- Some data might not be available in form of a XML document - the data might be computed on the fly (e.g. location information at a different granularity). Another approach is to have all information already available and to access as any other data item. Some discussions might be required to determine the best approach possible.
- How to structure access to location object information of different entities? As mentioned in [XCAP] it is necessary to define a schema for data used by a number of entities. One possible approach is to access individual data items with the help of URIs. In any case a naming convention for access to items of the location objects and other information items has to be created.

8. Acknowledgments

We would like to thank Christian Guenther for his input to this version of the draft.

This entire draft heavily borrows from the XACML [XACML] specification. We would like to thank the authors of the [XACML] specification for their excellent work.

9. References

[Kudo00] M. Kudo and S. Hada: "XML document security based on provisional authorization", in: "Proceedings of the Seventh ACM

Conference on Computer and Communications Security", Nov. 2000, Athens, Greece, pp. 87-96.

[XPath] XML Path Language (XPath), Version 1.0, W3C Recommendation, 16 November 1999, Available at: <http://www.w3.org/TR/xpath>.

[XPointer] XML Pointer Language (XPointer), Version 1.0, W3C Candidate Recommendation, 11 September 2001, Available at:

<http://www.w3.org/TR/2001/CR-xptr-20010911>.

[XSLT] XSL Transformations (XLT) Version 1.0, W3C Recommendation, 16 November 1999, Available at: <http://www.w3.org/TR/xslt>.

[XACML] The OASIS eXtensible Access Control Markup Language (XACML), OASIS Standard, Version 1.0, 18 February, 2003, Available at: <http://www.oasis-open.org/committees/xacml>.

[SAML] Security Assertion Markup Language, Available at: <http://www.oasis-open.org/committees/security/#documents>.

[Pet03] J. Peterson: "A Presence Architecture for the Distribution of Geopriv Location Objects", Internet Draft, Internet Engineering Task Force, (work in progress), February 2003.

[CG03] J. Cuellar and C. Guenther: "Geopriv Location Object Markup Language", Internet Draft, Internet Engineering Task Force, (work in progress), June 2003.

[CM+03] J. Cuellar, J. Morris, D. Mulligan, J. Peterson and J. Polk: "Geopriv requirements", Internet Draft, Internet Engineering Task Force, (work in progress), March 2003.

[DM+03] M. Danley, D. Mulligan, J. Morris and J. Peterson: "Threat Analysis of the Geopriv Protocol", Internet Draft, Internet Engineering Task Force, (work in progress), February 2003.

[XCAP] J. Rosenberg: "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", Internet Draft, Internet Engineering Task Force, (work in progress), May 2003.

[Ros03a] J. Rosenberg: "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usage for Presence Lists", Internet Draft, Internet Engineering Task Force, (work in progress), May 2003.

[Ros03b] J. Rosenberg: "A Session Initiation Protocol (SIP) Event Package for Modification Events for the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Managed Documents",

May 2003.

Author's Addresses

Hannes Tschofenig
Siemens AG
Corporate Technology
CT IC 3
Otto-Hahn-Ring 6
81739 Munich
Germany
EMail: Hannes.Tschofenig@siemens.com

Jorge R Cuellar
Siemens AG
Corporate Technology
CT IC 3
81730 Munich
Germany
EMail: Jorge.Cuellar@siemens.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in [BCP-11](#). Copies of claims of rights made available for publication 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 implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS 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.

Acknowledgement

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

