Securing Group Management in IPv6 with Cryptographically Generated Addresses

Claude Castelluccia, Gabriel Montenegro

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Securing Group Management in IPv6 with Cryptographically Based Addresses

Claude Castelluccia — Gabriel Montenegro

N° 4523

July 2002

THÈME 1

ISSN 0249-6399 ISRN INRIA/RR--4523--FR+ENG

Securing Group Management in IPv6 with Cryptographically Based Addresses

Claude Castelluccia*, Gabriel Montenegro †

Thème 1 — Réseaux et systèmes
Projets Planete

Rapport de recherche n° 4523 — July 2002 — 26 pages

Abstract: Currently, group membership management in IP Multicast and Anycast can be abused in order to launch denial-of-service (DoS) attacks. The root of the problem is that routers cannot determine if a given host is authorized to join a group (sometimes referred to as the *Proof-of-Membership Problem* [1]). We propose a solution for IPv6 based on Group Cryptographically Based Addresses (G-CBA). These addresses have characteristics of statistical uniqueness and cryptographic verifiability that lend themselves to severely limiting certain classes of DoS attacks. Our scheme is fully distributed and does not require any trusted third party or pre-established security association between the routers and the hosts. This is not only a huge gain in terms of scalability, reliability and overhead, but also in terms of privacy.

Key-words: Security, MLD, IGMP, Multicast, Anycast, Group Management, Proof-of-Membership Problem

* INRIA Rhône-Alpes, PLANETE group
† SUN Labs-Europe, 29, chemin du Vieux Chêne, 38240 Meylan, France

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 Montbonnot-St-Martin (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Securing Group Management in IPv6 with Cryptographically Based Addresses

Résumé : Ce rapport propose une solution de sécurisation du protocole d'adhésion MLD pour IPv6.

Mots-clés : Security, MLD, IGMP, Multicast, Anycast, Group Management, Proof-of-Membership Problem

1 Introduction

Group membership management in IP Multicast and Anycast suffers from potential Denial of Service (DoS) attacks. With Multicast, a malicious host that joins a group can overflow the network by adding branches to the group delivery tree. In contrast, a malicious host that joins an Anycast group will not be able to flood a network but can redirect the traffic and prevent other legitimate hosts from seeing it. The source of the problem is that currently routers cannot verify whether a particular host is authorized to join a particular group. The problem is sometimes referred as the *Proof-of-Membership Problem* [1]. We propose a solution to this problem for IPv6, based on Group Cryptographically Based Addresses (G-CBA), which are an extension of CBA (Cryptographically Based Addresses) for Group addresses (Multicast and Anycast groups). In particular, we define two types of G-CBA addresses: M-CBA for multicast and A-CBA for anycast. We use these addresses to severely limit certain classes of DoS attacks. Our proposal is fully *distributed*. It does not require any trusted third party or pre-established security association between the routers and the hosts. This is not only a huge gain in terms of scalability, reliability and overhead, but also in terms of privacy.

This paper is structured as follows: Section 2 is an overview of Multicast and Anycast in IPv6, as well as of MLD, the protocol used for group management. Section 3 deals with our motivations: it describes the security liabilities in MLD, and includes our problem statement. Section 4 talks about related work. Section 5 details our proposal by reviewing the concept of CBA addresses, and explaining both a *basic scheme* and a *certificate-based scheme* to handle multicast and anycast addresses. Section 6 presents a security analysis of our proposal. Section 7 discusses how this proposal benefits privacy considerations. Section 8 concludes and appendix A explains the certificate format we propose for use with MLD in our *certificate-based scheme*.

2 Multicast and Anycast Groups

The Internet Protocol (IP) defines two types of groups: *Multicast* and *Anycast* groups.

2.1 IP Multicast

Internet Protocol (IP) Multicast [2] defines network support that allows IP traffic to be sent from one or multiple sources and delivered to multiple destinations without sending individual packets to each destination. A single packet is sent by a source to a Multicast group, which is identified by a single IP address. The network is then responsible for duplicating the packet and delivering it to each member of the group. IP Multicast distinguishes between *multicast routers* and *multicast listeners*. Nodes register dynamically with a group by sending either IGMP (Internet Group Membership Protocol) messages [3, 4] (if using IPv4) or MLDv2 (Multicast Listener Discovery) [5, 6] messages (if using IPv6). Since our solution is specific to IPv6, in the subsequent discussion we will assume this process is carried out

via MLDv2. Unless otherwise noted, in this paper MLD refers to MLDv2. Nevertheless, IGMPv3 and MLDv2 are essentially identical except for the size of the addresses carried and the fact that the latter uses ICMPv6 message types instead of IGMP. Since the differences are largely immaterial, this document refers to previous work on securing IGMPv3, given the dearth of equivalent material for MLDv2. Routers use MLD messages to discover which groups on their directly attached links have active multicast listeners. Conversely, nodes use MLD to express interest in certain multicast groups and thus become multicast listeners. Notice that multicast routers may also be multicast listeners in which case the corresponding protocol behavior as multicast listeners applies as well. Routers coordinate multicast routing for the different groups by using one or more protocols such as DVMRP [7], PIM [8] or SSM [9].

2.2 IP Anycast

An IPv6 Anycast address is an address that is assigned to more than one interface. Thus an IPv6 Anycast address defines a group but as opposed to multicast group a packet sent to an Anycast address is not routed to all members of the group but only to the source's "nearest" one [10]. All interfaces belonging to an Anycast address usually resides within a topological region defined by an address prefix. Within this region, each member must be advertised as a separate "host route" entry in the routing system. A *router* that is member of an Anycast group will advertise its membership using the routing protocol (RIP, OSPF, BGP, etc). A *host* that wants to join an Anycast group will have to use a group membership protocol, such as MLD [11], to register with the local router(s) that will then propagate this registration to the region using the routing protocol.

2.3 Group Membership Management via MLD

This section gives some details of MLD, in particular, version 2 [6]. This description is just enough to understand our contribution. The MLD protocol is asymmetric, that is, it specifies different behavior for multicast routers and for multicast listeners. Furthermore, MLD does not apply to the link-local all-nodes multicast group (FF02::1) as all multicast-capable nodes are members automatically.

MLD for Multicast Listeners

Multicast (or Anycast) listeners use MLD to report interest in Multicast groups on each of their interfaces. These reports may be triggered by (1) applications expressing interest in certain groups, or (2) by *Query* messages sent by the Multicast routers. Whereas in the former case, listeners send out the reports immediately, in the latter case reports are delayed according to certain timers to avoid an implosion of a large number of reports. Multicast routers send any of three types of Query messages on their attached links:

1. *General:* To learn which Multicast groups have listeners.

2. *Multicast Address Specific:* To learn if specific Multicast addresses have listeners.
3. *Multicast Address and Source Specific:* To learn if any of several possible sources have listeners for the given Multicast addresses.

A Multicast listener sends *Listener Reports* to inform neighboring routers about which Multicast addresses it is interested in, or if there was any change in that list. These reports include Multicast Address Records of different types:

1. *Current State:* Sent in response to a Query to report listening state.
2. *Filter Mode Change:* To notify of changes in the filtering (INCLUDE or EXCLUDE) of existing Multicast addresses.
3. *Source List Change:* To notify of changes in the list of addresses being accepted (INCLUDE) or filtered out (EXCLUDE).

Reports that include either items 2 or 3 above are typically referred to as State Change Report messages.

MLD for Multicast Routers

At any given time only one Multicast router in a given link is the Querier. This is the router with the lowest IPv6 address. In fact, a router issuing a Query inhibits all other routers with higher IPv6 addresses on the same link from sending a Query of their own. All higher addressed routers remain in this Non-Querier state for a time known as the Other Querier Present Timeout. Once this timer expires they start issuing Query messages again.

3 Motivations

3.1 MLD-Specific Attacks

As described in [5, 6] and [1], *MLD* is prone to the following attacks.

- *Query messages*

  A forged Query message from a machine with a lower IP address than the current Querier will cause Querier duties to be assigned to the forger. If the forger then ignores Listener Report Messages, traffic might flow to groups with no members.

  A forged Query message sent to a group with members will cause the hosts which are members of the group to report their memberships. This causes a small amount of extra traffic on the LAN, but causes no protocol problems.

- *Report messages*

  A forged Report message to join a group may cause routers to assume there are members of a group on a link where, in fact, none exist. The fake Report messages are

only harmful if there are no other hosts in the LAN interested in the Multicast group. The effects of such fake reports are:

- The local router will create state for the group reported in the fake report message. This might be the source of a DoS attack: an attacker could send numerous report messages for different groups just for the sake of creating state at the local queriers.
- If the group address is fake (i.e. there is no source), a fake report may generate signaling traffic in the network.
- If the group address is valid (i.e. there is at least one active source), the local router will send routing messages into the network infrastructure. This will have different effects according to whether the group is a Multicast or an Anycast group. If the group is a Multicast one, the routing messages will create states in the Multicast router infrastructure, and add branches to the Multicast tree resulting in additional traffic. A malicious host could use this attack to overflow a network. If the group is an Anycast group, the routing message will also create additional states in the network infrastructure, that will possibly redirect all the Anycast traffic to the malicious host, leading to a Denial-of Service attack (DoS).

In addition to the above, a forged State Change Report message to leave a group will cause the Querier to send out Group-Specific Query messages for the group in question. This causes extra processing on each router and on each member of the group, but cannot cause loss of desired traffic. [5, 6] present some defences against such externally forged messages. Local forged State Change Report messages are more difficult to prevent.

Routers that receive MLD messages must verify that the source is link-local. This requirement defends routers from forged MLD messages originated off-link. The attacks described previously are therefore only possible by local hosts.

3.2 Problem Statement

The goals of our work is to propose a solution to the security problems related to the Report messages presented previously. These problems are exacerbated in mobile environments, as in constantly varying environments, routers do not necessarily know the Multicast listeners. It is therefore difficult to authenticate them. This problem is referred in [1] as the *Proof-of-Membership* problem, i.e.: a router will process a Report message from a host for a specific group, only if this host can prove that it is a legitimate member of the group. The proposed solution must satisfy the following objectives:

- The solution should be "light", i.e. should not be too expensive in terms of computation, memory or bandwidth.
- The solution should be scalable, i.e. able to support a very large number of members per group and a large numbers of groups.

- The solution should support mobile hosts efficiently. It should therefore provide fast group registration and should not assume any pre-established contexts in the routers.
- Avoid reliance on a global infrastructure such as a Public Key Infrastructure (PKI) as much as possible.
- Avoid reliance on a Trusted Third Party (TTP) as much as possible.

As far as the two security issues with forged Query messages are concerned, we believe that the problem related to Querier selection is solvable. For example, all routers of a link could share a secret key. It would then be enough for routers to verify the authenticity of the Query messages. The problem related to fake routers causing extraneous traffic by sending fake Query messages is much harder to solve. This problem, known as the *fake bank teller* problem, is not specific to group communication. This issue of validating routers or, in general, nodes that offer services remains an unsolved authorization problem in the Internet at large.

4 Related Work

The IETF MSec working group [12] is developing protocols for securing groups over the Internet. They propose to solve the *Proof-of-Membership* problem described previously by encrypting the Multicast flows and by revealing the encrypting keys only to the authorized members. As a result only the authorized members of a group will be able to decrypt and use the Multicast traffic. This solution has the following drawbacks:

- An unauthorized host is still able to join a Multicast group and receive packets from this group. He will not be able to decrypt the packets but will be able to generate extra traffic on the network. A malicious host could use this protocol's feature to launch a DoS attack.
- This solution requires that the Multicast flows be encrypted. We believe that this is a valid requirement to (1) protect the Multicast content, but is not as applicable if the objective is to (2) protect the routing infrastructure. Whereas the MSec efforts are aimed primarily at (1), our work places emphasis on (2).

MLDv2 [6] suggests that IPSec in Authentication Header mode [13] may be used to protect against remote attacks by ensuring that MLDv2 messages come from a system on the LAN (or, more specifically, a system with the proper key). When using IPSec, the messages sent to the all-systems Multicast group and the all-router Multicast group should be authenticated using AH. When keying, there are two possibilities:

1. Use a message authentication code (e.g., HMAC) with a single key that is shared by the routers and the hosts of the LAN (or a key for each group). This allows validation that a packet was sent by a system with the key. This has the limitation that any host with the key can forge a message even if its moves out of the current LAN;

2. When appropriate key management standards have been developed, use a digital signature algorithm. All hosts need to know the public key of all routers, and all routers need to know the public key of all hosts in the LAN. This requires a large amount of key management but has the advantage that senders can be authenticated individually so, for instance, a host cannot forge a message that only routers should be allowed to send.

These proposals solve the remote attacks problem but do not consider the proof-of-membership. These are two different problems. In fact, with the previous solutions, a router is able to authenticate a host or a set of hosts but it is unable to verify whether this host or set of hosts are authorized to receive packets from the Multicast group they subscribe to.

[1] proposes a solution to the IGMP proof-of-membership problem that uses a Key server. The proof consists of a symmetric-key, *IGMP-key* that is used by the receiver and the Multicast router to protect the IGMP message. The IGMP-key is enclosed within an access-token that is signed by the issuing-authority and stored in the Key server. Key server then sends the token to the Multicast routers and authorized hosts. When a router receives an IGMP message from a host it verifies if it has a valid token (and therefore a secret key) for this host before processing it. This solution seems to work for Multicast group within a domain only. It is not clear how a group that has members in different domains could use this proposal. Similarly, this proposal does not support mobile hosts that visit a foreign domain (since they do not have tokens in this domain). Furthermore, the scalability property of this solution is questionable. A domain key server must have tokens for each potential member of each Multicast group of the domain. Additionally each router must have tokens for each potential member of each Multicast group of its links.

There is not much work that concentrates on Anycast security. Nevertheless, [14] identifies several issues although it does not propose any solutions. Today the group membership verification problem in IP Anycast is basically handled by 1) restricting which boxes are trusted to participate in the routing protocol and 2) the ability to filter routes at boundaries. In fact, as mentioned in [11], it is expected that routers will employ some filtering mechanism when receiving a Report message containing an Anycast address. For example, one policy might be to deny all Anycast joins except those that match a configured list of (source address, Anycast address) pairs. This list-based filtering is definitely not scalable and not adapted to dynamic and mobile environments. It requires updating the access lists each time a new host moves or leaves a network.

5 Proposal Overview

5.1 Review of CBA IPv6 address

[15, 16] present the concept Cryptographically Based IPv6 addresses. CBA addresses were proposed to solve the IPv6 *address ownership* problem [17], and are generally useful to secure redirect operations in numerous protocols. For example, in Mobile IPv6 [18], a node starts

using its home address, and, each time it moves to a different link, it issues a Binding Update that specifies its current address. The issue is in handling these Binding Updates securely. Why should the correspondent node believe the mobile node when it claims that it does, in fact, own the home address contained in the binding update? The risk is that this mobile node could be issuing a redirect for another node's home address in order to redirect its packets. Ignoring this address ownership problem can lead to *denial-of-service* (DoS) and unauthorized redirect attacks.

[15, 16] associate to each host a public-private key pair and derive CBA addresses from it. A CBA IPv6 is a valid IPv6 address. The top 64 bits are the host's routing prefix as in [19]. The bottom 64 bits, the *Cryptographically Based Host Identifier* (*CGHID*) [15], are derived from the host's public key as follows:

$$CGHID = hmac - 64(sha1(imprint), sha1(PK)) \ ^{1} \quad (1)$$

Where *imprint* is a 64-bit field and *PK* is the public key associated with the host.

The generated CGHID must be compared against a list of well-known Anycast addresses [20] and the value 0. If there is a conflict, the host can retry using a different imprint, or, alternatively, start from the beginning by generating a new public-private key pair.

Note that, usually, IPv6 refers to the bottom 64 bits of an address as the "interface identifier". In contrast, the CGHID is a "host identifier". If the host has more than one address (because it is multihomed or has several interfaces), it could use the same CGHID for its addresses or generate different addresses by using different imprint values. In any case, the same public-private key pair is used. In general, the CGHID identifies a host, an IP stack and not just any particular interface.

When a host wants to prove to its correspondent node that it owns its CBA address, it reveals its public key, the imprint and signs its message(s) with its private key. The correspondent node then verifies that (1) the interface identifier was derived from the public key and the imprint and, (2) that the signature is correct, and therefore that the host knows the private key component. As a result, this address belongs to the host because no other hosts could have used an imprint and public-private key pair combination that hashes to the same CGHID.

As detailed in Section 6.1, CBA address collisions are very unlikely, so they are statistically unique. For further details on CBA, please refer to [15, 16].

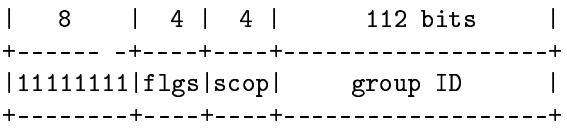
5.2 Group CBA Addresses (G-CBA)

In this paper, we propose to extend the CBA concept to group addresses in order to solve the *proof-of-membership* problem in group management. We defined two new types of addresses, namely the Anycast CBA Address (A-CBA) and the Multicast CBA Address (M-CBA).

[1] Note that according to [10], bit 6 of the CGHID (the universal/local) has to be set to zero to indicate that the CGHID is not guaranteed to be globally unique.

5.2.1 Multicast CBA Addresses (M-CBA)

An IPv6 Multicast address is an identifier for a group of interfaces (typically on different nodes). An interface may belong to any number of Multicast groups. Multicast addresses have the following format [10]:



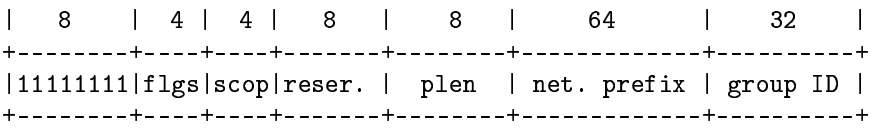
where:

- binary 11111111 at the start of the address identifies the address as being a Multicast address.
- *flgs* is a set of 4 flags. The high-order 3 flags are reserved, and must be initialized to 0. The last bit, T, is set to to indicate a permanently-assigned ("well-known") Multicast address, assigned by the Internet Assigned Number Authority (IANA). T = 1 indicates a non-permanently-assigned ("transient") Multicast address.
- *scop* is a 4-bit Multicast scope value used to limit the scope of the Multicast group.
- *group ID* identifies the Multicast group, either permanent or transient, within the given scope.

We propose to associate to each group a public-private group key pair. The Multicast-CBA (M-CBA) address of this group is then an IPv6 Multicast address whose group ID is an 112-bits long CBI (Cryptographically Based Identifier) i.e.

$$GroupID = hash - 112(PK) \tag{2}$$

Recent work [21] introduces Multicast addresses derived from unicast prefixes. The format of a *unicast-prefix based Multicast address* is defined as follows:



Where *network prefix* identifies the network prefix of the unicast subnet owning the Multicast address. Such addresses solve the Multicast address collision problem and simplifies dynamic Multicast address allocation by obviating the need for the Multicast Address Allocation Protocol [22] and the Multicast Address-Set Claim (MASC) protocol [23]. M-CBA addresses also achieve this simplification and solve the Multicast address collision problem by virtue of their statistical uniqueness. In fact, as shown in Section 6.1, M-CBA provides

very low probability of address collision and it is therefore almost impossible that two groups will generate the same CBA address.

Note that *unicast-prefix based multicast* address proposal also tries to avoid collisions for L2 addresses when the L2 Multicast address is formed from the low-order 32 bit of the IPv6 Multicast address. A M-CBA, being a hash of a public key, has good randomness properties in any sets of bits in the hash. Thus it solves this problem as well.

Nevertheless, we must distinguish between M-CBA addresses and the others, so we define a new bit in the *flag* field, the *S* bit. If the *S* bit is set, then the address is a M-CBA address. Note that [21] defines the *P* bit in the *flgs* field to distinguish *unicast-prefix based multicast* addresses from those defined in [10].

5.2.2 Anycast CBA Addresse (A-CBA)

Similarly to the Multicast case, we define IPv6 A-CBA (Anycast Cryptographically Based) addresses. Anycast addresses are allocated from the unicast address space. Thus, Anycast addresses are syntactically indistinguishable from unicast addresses [10]. An Anycast CBA address is then generated as described in Section 5.1.

5.3 Protocol Overview

Our proposal relies on Group CBA address to solve the proof-of-membership problem. We actually propose two schemes: a *basic scheme* and *certificate-based scheme*. This section describes these two schemes and their respective benefits.

5.3.1 Basic scheme

The main steps of this scheme are:

1. The group controller generates the group's Public-private key pair and derives from it the G-CBA as described in Section 5.2.

2. The private key, the public key and the G-CBA are distributed to the (authorized) group members. Note that private key requires confidentiality because it only has to be known to the authorized group members. The public key and the G-CBA can be sent in clear but require integrity.

3. A host that wants to join or leave the group must include the group public key when it joins or leaves the group, and signs the resulting messages with the group private key.

4. A router that receives a Report message for a Group CBA address must verify the host's proof-of-membership to the group by verifying that: (1) G-CBA was generated from the public key and (2) the signature is valid. Only a host that was authorized to join the group would know the private key associated to the group and therefore

would be able to sign the messages. If the proof-of-membership is correct, then the router processes the report message normally, otherwise it rejects it.

This proposal does not rely on a global infrastructure such as a PKI or TTP. The secret used to prove the group membership is directly sent to the host members. This information is sent once for the whole duration of the group session. The routers do not have to contact a Trusted Third Party (TTP) as in [1] to verify the proof-of-membership of a host to a group each time it receives an Report message from a new host. All necessary information is included in the Report messages and can be processed immediately. As a result, joining a group is *faster* and *less bandwidth consuming* (no extra messages are sent). Furthermore the routers do not have to be statistically pre-configured with symmetric keys for each of potential hosts that could send Report messages. These two properties make our solution very suitable to mobile environments where fast handoff is an issue [24].

However this *basic scheme* has the following limitations/problems:

1. It does not provide *Membership expiration*. This scheme makes the assumption that once a member is accepted in a group it will remain so for the duration of the group. This assumption might not hold for all applications.

2. It requires a *secure channel* between the group controller and the group members for the private key distribution.

3. It suffers from the *private key disclosure* problem. The basic scheme makes the assumption that the members of a group are trusted hosts and that they won't disclose (intentionally or not) the group private key to anybody. This assumption is clearly not valid for open groups.

As a result of the following limitations, the basic scheme is mainly suited to closed groups (for example within a company) whose members are trustworthy.

For the open groups, we propose another scheme, the *certificate-based* scheme, that is also end-to-end but solves all the problems described previously. This scheme supports permanent and transient group, support membership expiration and does not suffer from the *private key disclosure* problem. This scheme does not require that the members share a secure channel with the group controller. However it requires that each member have a (unicast) CBA address, defined as described in Section 5.1.

5.3.2 Certificate-based scheme

This scheme uses authorization certificates (for example SPKI [25] or Keynote [26] certificates). It assumes that (1) the group address is a G-CBA address (as in the basic scheme) and that additionally (2) each member of the group has a (unicast) CBA address (this was not required in the basic scheme).

The main steps of this scheme are (see Figure 1):

1. The group controller generates the group's Public-private key pair, G_PK and G_SK, and derives from it the Group CBA, G_CBA, as described in Section 5.2.

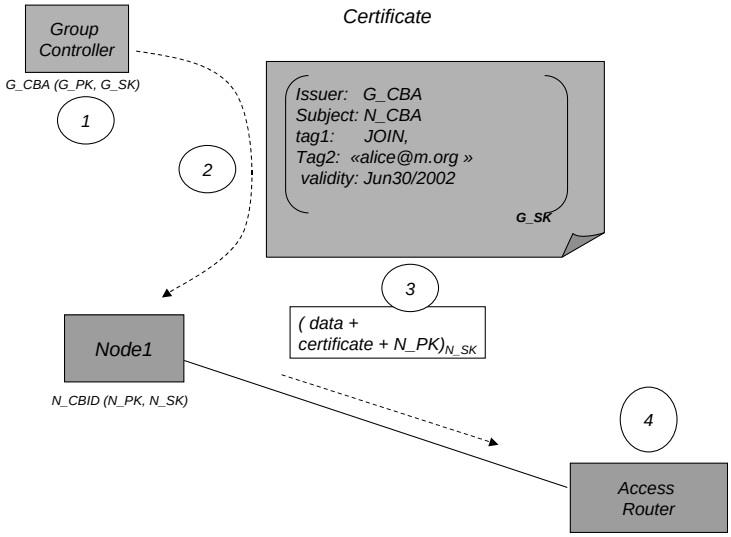


Figure 1: Certificate-Based Scheme

2. The group controller generates for each authorized member a certificate (see the Appendix for details about the SPKI certificate we use) that states that the host, defined by its CBA address, is authorized to join the group defined by the G-CBA address. This cerficate is signed with the private key whose public key was used to generate the G-CBA address. It contains the group public key that was used to generate the G-CBA address, a validity period, the group controller address and the member CBA address.

3. A host that wants to join or leave the group must include the certificate, its public key (the one used to generate its CBA address i.e. N_PK), and signs the resulting messages with the private key whose public key was used to generate its unicast CBA address, i.e. N_SK.

4. A router that receives a Report message for a Group CBA address must verify the host's proof-of-membership to the group by verifying that: (1) the certificate is valid, i.e. that its signature is correct, that the validity period is still valid and that the group address was generated from the group public key. (2) the MLD message is valid, i.e. that the member's CBA address was generated from the public included in

the message, that the member's CBA address is the same than the one contained in the certificate and that the signature is valid (i.e. the host owns the CBA address).

The pseudo-code executed by a router upon reception of a Report message is the following:

```
if (N_CBA == hash(N_PK)) {
 /* N_CBA seems valid */
 if (signature is valid) {
  /* N_CBA belongs to the requester */
   if (G_CBA == hash(G_PK)) {
    /* Group address generated from PK */
     if ( (certificate signature is valid)
   && (issuer  == G_CBA)
   && (subject == N_CBA)
   && (validDate(June30/2002)) ) {
       /* Cert. is valid */
           process MLD request
          return;
 }}}}
 else
  reject MLD request;
```

After these two steps the router has the assurance that the member (1) the authorization certificate is valid and (2) it is used by the legitimate member (it has not be stolen).

If the proof-of-membership is correct, then the router processes the report message normally, otherwise it rejects it.

Note that the authorization certificate contains all the necessary information and that the routers that receive such as certificate do not have to contact any server nor to have some pre-establised states (keys,etc). As a result, this scheme is very scalable and well adapted to mobile environment. Additionally, it solves all the problems of the *basic scheme*. In fact, the *certificate-based* scheme:

1. provides *membership expiration*. Each certificate has a validity field. When the validity of a certificate expires, the member needs to get a new certificate. The group controller can then refuse to re-issue the certificate, in effect, excluding the member if necessary. This use of short-term certificates avoids the need for a revocation system.
2. does not require a *secure channel* between the group controller and the group members for the private key distribution.
3. does not have the *private key disclosure* problem because the private key is not revealed by the group controller.

The format of the SPKI certificate that is used by this scheme is described in Appendix A.

6 Security Analysis

6.1 Hash ID Size Considerations

In Multicast CBA addresses (M-CBA), the GroupID is 112 bits-long (63 bits-long for Anycast CBA), and is generated from the hash of the group public key. In this section we evaluate the probability of intentional or unintentional collisions (i.e. the probability that 2 public keys generate the same GroupID).

Suppose the hash function produces an n-bit long output. If we try to find some input which produces some target output value y, then since each output is equally likely we expect to have to try $2^{(n-1)}$ possible input values on average. On the other hand, if we are worried about naturally occurring CBA address duplications, then by the birthday paradox we would expect that after trying $1.2 \times 2^{n/2}$ possible input values we would have a 50% probability of collision [27]. So if $n = 112$ ($n = 63$ for anycast), you need a population of 1.2×2^{56} i.e. 8.64×10^{16} Multicast addresses ($1.2 \times 2^{31.5}$ i.e. 3.64×10^{9} for anycast) on average before any two produce duplicate addresses. So unintentional collisions are very unlikely. If an attacker wishes to impersonate a given CBA Multicast address, it must attempt 2^{111} (2^{62} for anycast), i.e. approximately 2.59×10^{33} (4.8×10^{18} for anycast), tries to find a public key that hashes to this CBA address. If the attacker can perform 1 million hashes per second it will need 8.23×10^{19} (142,235 for anycast) years.

Note that the previous analysis only considers the cost of computing the hash of the public key. Additionally, an attacker must also generate a valid public-private key pair. This is a significantly more expensive operation.

6.2 Denial-of-Service Attacks

Denial-of-service (DoS) attacks that exhaust host resources (memory and processing resources) is a major security threat on the Internet. With our approach, the routers have to verify the signatures on the *Report* messages before processing them. Signature verification is an expensive operation and an attacker could attempt a DoS attack by sending numerous fake MLD messages to a router. The router will then spend most of its time verifying (wrong) signatures. However note that in order to attempt this type of attacks, the attacker must be attached to the same link as the router. This limits the risks considerably. Some potential solutions to this threat are:

- The routers may use some puzzle mechanisms as the one described in [28]. If a router believes that it is under attack it will reply to a MLD message with a puzzle request and will accept to process it only if it receives a valid puzzle reply.
- For hosts that are not mobile or that are well known, a router might actually derive a shared key for each of them using *sucvP* [15] (which itself includes the puzzle mechanism mentioned above). The MLD messages from these nodes will then be authenticated using a message authentication code such as HMAC instead of a signature

algorithm. If a router believes that it is under attack it might then decide to process HMAC authenticated MLD messages before signed ones.

- As an alternative to the previous scheme, a security association derived between certain hosts and the routers could be used to start a hash chain. The "proof-of-membership" phase could at the same time communicate a signed initial value for the hash chain. Each subsequent MLD message (up to a certain limit) would use corresponding values taken from the hash chain. At the cost of keeping some state (the previous value for the hash) for each host, routers would carry out only an initial signature verification. Subsequent MLD messages would be verified by calculating a hash. This would allow routers to give preferential treatment to MLD messages authenticated via hash-chains.

6.3 Replay Attacks

A malicious host might replay a valid Report message. Since the signature is valid, the router will accept the message. Replaying a valid Report message to join a group can be quite harmful because the router will re-join the group and packets will start flowing again. The effect of replaying a valid Report message to leave a group is less obvious. When a router receives such a message from a host, it sends a query message to the relevant group to check whether there are other hosts interested in it. Interested hosts will then reply with a report message. As a result, this attack will only generate extra load on the network but will not prevent legitimate members from receiving the packets.

A solution to this replay problem is to use a nonce which the querier must include in each of its Query messages. A host that wants to join or leave a group must include the last nonce in its messages. A router will accept an MLD message only if it has a valid nonce. This solution is not optimal when fast registration is needed, as is the case with a mobile Multicast listener, because it would need to wait for the next Query message in order to learn the nonce. An optimization is for the mobile listener to actively solicit the nonce from the Querier, in preparation for or in response to moving into a new link. On the down side, this scheme does imply some level of nonce synchronization.

In the certificate-based scheme, it is possible to limit the window of exposure to replay attacks by issuing short-lived certificates. Of course, in order to avoid expired certificates from being used in replay attacks, it is necessary to have some time synchronization between the issuers of the certificates and the routers. Coarse time synchronization is probably enough, though.

The best protection against replay attacks would be to use sucvP (which itself has replay protection) as outlined in the previous section to derive a security association between a given host and a router. At the cost of the routers' maintaining some state per host, subsequent use of either IPsec ESP or hash chains would provide replay protection for the MLD messages.

As in the previous section, the risk is limited because the attack can only be launched by malicious nodes on the same link as the targetted router.

6.4 Unauthorized Group CBA Address

With our proposal, nothing prevents a malicious host from generating an unauthorized Group CBA address (from a valid public key) and then sending messages to join this group. Routers will accept those messages since the address is verifiable and the signature is valid. This attack is not very harmful to the network itself because no active source is associated to this address and therefore no packet will be sent over the network. However this can be the source of a DoS attack on the infrastructure routers because joining groups creates entries for the corresponding addresses in the routers. Some solutions to this problem are presented in Section 6.2. This attack can become much more harmful if several malicious hosts register this address with different routers distributed over the network and if some of them actually send packets to this address. This DDoS attack can generate a lot of traffic on the network especially if there is a large number hosts that are malicious or merely have been broken into.

This problem is difficult to solve. We believe that the solution space includes an access control mechanism that will forward packets to a group address only if the source is authorized to do so. This problem which can be referred as a source authorization problem is not an MLD problem, and is therefore outside the scope of this note. Here, we only consider the receiver authorization problem.

Nevertheless, it may be possible to leverage the certificate-based solution presented here to limit this problem as well. Suppose the routing infrastructure is itself a group, and that it is defined by a Group CBA address (G-CBA). The routing infrastructure would then be able to explicitly authorize certain groups to exist. In addition to the requirements spelled out in section 5.3.2 on the certificate-based scheme, a router verifying a host's proof-of-membership in a group would need the certificate in which the the routing infrastructure authorizes the relevant group to exist. This certificate would be issued by the routing infrastructure, so it would need to be signed with the corresponding private key.

7 Privacy Considerations

7.1 Group Structure Privacy

By solving the "group membership problem" via group CBA addresses (either in M-CBA or A-CBA format), it is possible (and quite natural) to not disclose to the routers any information beyond what is absolutely necessary for them. In particular, other proposals [1] require at least that the group controllers or key issuers reveal their identity to the routers as part of distributing the tickets. Group CBA addresses obviate the need for such contact between the routers and any third parties or group controllers. This is not only a huge gain in terms of reliability and overhead, but also in terms of privacy. In fact, a MLD report message only reveals the CBA address of the group and the current address of the subscribing member. This group CBA address is basically an identifier and does not provide any information about the group, the group controller or the members identities. For further privacy guarantees, the subscribing member's current address should be configured using the

IPv6 address privacy extensions defined in [19] in order to hide the subscribing member's identity. As a result, a router that receives an MLD message will know only that a group that uses a group CBA address exists and that there is at least one member on its links.

7.2 Traffic Privacy

For certain groups, traffic confidentiality is a requirement. The certificate-based proposal could be leveraged to provide this as well. For example, a group controller(s) could create a shared secret to be used for traffic encryption. But how would it communicate this shared secret to each and every one of the group members?

As part of joining a group, a host obtains an authorization certificate either from the group controller or from a node authorized by it 5.3.2. In this "induction" step each host is explicitly addressed by the group. At this very useful moment of induction, the group can (in addition to issuing the authorization certificate) communicate to the receiving host the shared secret. The shared secret would be encrypted using the public key of the target host, so only the intended recipient would be able to decrypt and use it.

Notice that this does not add any new communication requirements. The existing induction step is leveraged to provide one more service to the inductee: enabling it to participate in encrypted group communications.

This is a very useful service, but there are some limitations in this simple scheme. For example, it is not a good idea to use the same shared encryption key during the entire lifetime of the group. But how to change the keys? Assuming short-lived certificates, the often-repeating induction step could be used to communicate new keys. It may be necessary to use windows of synchronization between group memberships durations (as specified in the per-host authorization certificates issued by the group controller) and the shared keys used. In other words, it may be simpler to follow the approach taken with the authorization certificates, in which their duration is used to impose membership expiration, instead of providing for explicit membership revocation. In similar manner, it may be easier to provide for shared encryption key expiration by specifying a limited lifetime for the keys (in terms of a time after which the key is valid, and a time beyond which it must not be used). This constitutes the activation period for the keys.

If a hosts waits for the key to expire, it may be unable to decrypt traffic for some duration of time. This may be avoided if the induction step is treated as a registration. The host would need to re-register with the group periodically in order to obtain the required membership authorization certificate as well as the corresponding encryption key(s) (which may not become active until some time in the future). Assuming some numbering on the generation of the key used, it may be possible for all member hosts to switch to new keys without very fine grained time synchronization.

This could be done along the following lines:

1. The group controller creates some keys in advance, for example, K_1 $[t_1;t_2]$, K_2 $[t_2;t_3]$, K_3 $[t_3;t_4]$, K_4 $[t_4;t_5]$...K_j $[t_j;t_{j+1}]$, where K_j is the key, its activation period is defined by start and end times t_j and t_{j+1}, and j is the key index. Notice that it is possible

to create keys during the lifetime of the group. Of course, the group controller must endeavor to create the keys and begin distributing them in advance of their activation period (that is, before they are used to encrypt traffic).

2. When a host registers, it obtains an authorization certificate and a list of keys that covers the duration of the certificate's validity. For example, if a member's certificate is valid between $[T0,T_1]$, and:
   $t_j + e \leq T_0 < t_{j+1}$
   $t_k < T_1 + e \leq t_{k+1}$

   (where, $k > j$ and e is a time skew to account for imperfect synchronization), the induction step must include keys K_j, K_{j+1},...,K_k.

3. At this point, the host is enabled to participate in a confidential group. It has the required keys to decrypt during the duration of its authorized membership period. When performing encryption with key K_j, it must include the index j in plain text, so receivers know which key they must use for decryption. This is very important if encryption or decryption happens close to key activation or expiration times, or if keys have overlapping activation periods.

4. When the certificate expires (at T_1), the host needs to request a new certificate. An even better strategy is for the host to request a certificate renewal before its current certificate expires. If at this point the group controller decides to renew the certificate it also discloses the required keys as per step 1 above.

If it is on a subnet where there are other authorized members, an expired member can potentially listen to the traffic between $[t_j; T_0]$ and $[T_1; t_{k+1}]$. Depending on the duration of the key activation time slots and the nature of the group's traffic, this may be acceptable.

8 Conclusion

Handling group membership safely is a thorny problem which if ignored can lead to denial of service attacks. At the root of the problem is a node's being able to prove that it has the authorization to be a member of any given group. We propose a certain type of group addresses for both Multicast and Anycast groups. These addresses allow a fully distributed solution. When a routers receives an MLDv2 report from a multicast listener, it can verify that this listener is an authorized member of the group or groups. In order to do this the router does not need to contact any trusted third party, nor does it need any pre-established security association with the listeners. This is not only a huge gain in terms of scalability, reliability and overhead, but also in terms of privacy.

Acknowledgements

We would like to thank Erik Nordmark for many helpful comments and discussions and Pekka Nikander who suggested the "certificate-based" approach described in Section 5.3.2. Their suggestions have improved our proposal significantly.

A SPKI MLD Certificate

This section describes the SPKI certificate that is used by the certificate-based scheme. Note that any authorization schemes such as Keynote [26] could also be used.

A.1 SPKI Review

The IETF SPKI Working Group has developed a standard form for digital certificates whose main purpose is *authorization* rather than authentication. SPKI main principles can be summarized as follows:

- a certificate has 5 fields: (1) issuer (who is vouching), (2) subject (who is acquiring the permission, (3) delegation (set if the subject can delegate the permission), (4) authorization (specifies the permission being communicated) and (5) validity.
- SPKI is *key-oriented.* No (name, key) binding , and therefore no CA, is necessary. The entities possesing, delegating and receiving access rights are cryptographic key pairs. A certificate can in short be written as: Sk(K' has the right R., t) (K gives the right R to K' and the validity period is t), where K and K' are 2 public key.
- A certificate has a validity period.
- Delegation certificates differ from traditional access control schemes in that any key may issue certificates. There is no central or trusted authority.
- A key may delegate rights to services it controls, it may also redelegate rights it received by delegation from other keys. When a key delegates to another key and this key in turn redelegates to a third one, and so on, the delegation certificates form a *chain.* The rights and the validity obtained by a chain of certificates are the intersections of the rights and validity of each certificate. When a key requests a service and it has obtained the access rights through a chain of certificates, it attaches the entire chain to its request. Certificate chains can be reduced through certificate reduction.

  Sk1(K2 has R1, t1) and Sk2 (K3 has R2, t2) => Sk1(K2 has intersection(R1,R2), intersection(t1,t2))

  Certificate reduction is the main technique of certificate management in SPKI.

A.2 MLD SPKI certificate

A full authorization certificate as used in our proposal is composed of a sequence of three objects [29]: the *public-key object* that contains the issuer public key, the *certificate object* that defines the authorization and *a signature object* that contains the signature.

```
(sequence
 (public-key object)
 (cert object)
 (signature object)
)
```

The public-key and signature objects are generic and defined in [25]. The *cert.* object is specific to each application. The MLD certificate object that is used to authorize hosts to join a group in our proposal has the following format:

```
(cert
  (issuer  (addr <gcga_addr>)
  (subject (addr <cga_addr>)
  (tag (mldjoin <gcga_addr> [<sourceaddr1>
           <sourceaddr2> ...<sourceaddrN>])
  (propagate)
  (not-before <date1>)
  (not-after  <date2>)
)
```

This certificate is issued by the group controller. It authorizes the host that has the CBA address defined in the *subject* field to join the group (potentially a Source Specific Multicast group) group defined in the *tag* field. The subject can propagate this authorization (this is optional). This certificate is only valid after the *date1* and before *date2*.

```
(cert
  (issuer  (addr
     FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678)
  (subject (addr
     2001:660:1002:2000:7E45:F543:9C74:BA76)
  (tag (mldjoin
     FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678 *)
  (propagate)
  (not-before ‘‘2002-02-02_00:00:00’’)
  (not-after  ‘‘2002-03-02_00:00:00’’)
)
```

For example, the above certificate authorizes the host, *H1*, that has the CBA address, 2001:660:1002:2000:7E45:F543:9C74:BA76, to join the Multicast group defined by the M-CBA address, FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678. The $<$ *sourceaddr* $>$ field it

set to "*". This means that the host can receive packets from any source of that group. This certificate is only valid between the 2nd of february 2002 and the 2nd of march 2002. This certicate has to be signed by the group private key. The host H1 is allowed to propagate this authorization. For example it could issue the following certificate to authorize the host H2 defined by the CBA address,
2001:660:1002:2000:45BE:CF12:3AF2:7843:ABDE, to join the same group by issuing the following certificate:

```
(cert
  (issuer  (addr
     2001:660:1002:2000:7E45:F543:9C74:BA76)
  (subject (addr
     2001:660:1002:2000:45BE:CF12:3AF2:7843:ABDE)
  (tag (mldjoin
     FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678 *)
  (not-before ``2002-02-05_00:00:00'')
  (not-after  ``2002-02-20_00:00:00'')
)
```

This certificate has to be signed by H1's private key (the one whose public key was used to generate its CBA address). When H2 joins the group it has to include the two certificates in its message. Note that the chain of certificates can be reduced by the group controller. In fact, as described in [25],

```
Two SPKI certificates:
   <I1,S1,D1,A1,V1> + <I2,S2,D2,A2,V2>
yield
   <I1,S2,D2,AIntersect(A1,A2),VIntersect(V1,V2)>
provided the two intersections succeed,
   S1 = I2
and
   D1 = TRUE
```

Therefore the two previous certificates can be reduced as follows:

```
(cert
  (issuer  (addr
     FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678)
  (subject (addr
     2001:660:1002:2000:45BE:CF12:3AF2:7843:ABDE)
  (tag (mldjoin
     FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678 *)
  (not-before ``2002-02-05_00:00:00'')
  (not-after  ``2002-02-20_00:00:00'')
)
```

A.3 Chain of Certificates

Chains of certificates can be very useful to improve the scalability and manageability of our scheme. For example, instead of managing the membership authorization of each group member (i.e. verifying that the member is allowed to join and generating the appropriate SPKI certificate), the group controller can, instead, generate a SPKI certificate to a server in each domain, referred in this paper as a local group controller, that is in charged of delegating this authorization to the members or to a sub-domain server (and so on). This delegation of membership authorization make our approach much more manageable for large-scale networks.

Let's take the example of a content distribution application that uses IP Multicast to deliver its data to its members worldwide. If no delegation is used, the group controller must manage the membership authorization request of each member of the group. This approach does not perform if the number of members is large and if the members are far from the group controller. It is obviously more practical and efficient if the group controller (that is for example in the US) delegates the right to join the group to several local group controllers (for example one per country) that are themselves in charge of authorizing local members. The (global) group controller will typically issue a certificate to the local group controllers whose validity is much larger than the validity of the certificates issued by the local group controllers to the final members (for example one year versus one month). A host that wants to join a group will then ask to its local group controller a certificate. The local group controller will issue it upon reception of the host's credential (for example its fees). The host will then join the group by including the chain of certificates in its MLD join message.

This approach is much more scalable because the members' certificates are issued locally. It is also much more manageable because (1) the authorization credentials can be defined locally (for example, the registration price can then differ from one provider to another), and (2) the local group controller can exclude a member (by not reissuing its certificate) transparently to the global group controller.

Certificate reduction is very useful to reduce the processing load at the routers. However, it deviates from the hierarchical model, because the reduction has to be performed by the group controller (or any node that knows the group private key). In spite of this, the hierarchy is maintained for authorization management: verification that a host is allowed to join a group is still performed locally. Typically, the local controller will (1) verify that the member is allowed to join the group; (2) generate a certificate stating that the member's credentials have been verified, and that the member is authorized to join the group; (3) send this certificate to the group controller (which will reduce it and send it back to the local controller); and, finally, (4) send the reduced certificate to the member. Even though the group controller still has to reduce the certificate, credential verification (step 1) is performed locally.

References

[1] T. Hardjono and B. Cain, "Key establishment for igmp authentication in ip multicast," in *ECUMN*, Colmar, France, October 2000.

[2] S. Deering, *Host Extensions for IP Multicasting*, IETF, RFC 1112, August 1989.

[3] W. Fenner, *Internet Group Management Protocol, Version 2*, IETF, RFC2236, November 1997.

[4] B. Cain and al., *Internet Group Management Protocol, Version 3*, IETF, draft-ietf-idmr-igmp-v3-07.txt, March 2001.

[5] S. Deering, W. Fenner, and B. Haberman, *Multicast Listener Discovery (MLD) for IPv6*, IETF, RFC2710, November 1999.

[6] R. Vida and al, *Multicast Listener Discovery Version 2 (MLDv2) for IPv6*, IETF, draft-vida-mld-v2-01.txt, July 2001.

[7] Pusateri P., *Distance Vector Multicast Routing Protocol*, IETF, draft-ietf-idmr-dvmrp-v3-10.txt, August 2000.

[8] D. Estrin and al., *Protocol Independent Multicast-Sparse Mode (PIM-SM)*, IETF, RFC2362, June 1998.

[9] H. Holbrook and B. Cain, *Source-Specific Multicast for IP*, IETF, draft-ietf-holbrook-ssm-arch-00.txt.

[10] B. Hinden and S. Deering, *IP Version6 Addressing Architecture*, IETF, RFC2373, July 1998.

[11] B. Haberman and D. Thaler, *Host-based Anycast using MLD*, IETF, draft-haberman-ipngwg-host-anycast-00.txt, February 2001.

[12] *IETF Multicast Security (msec) Working Group*, http://www.ietf.org/html.charters/msec-charter.html.

[13] S. Kent and R. Athkinson, *IP Authentication Header*, IETF, RFC2402, November 1998.

[14] L. Dondeti, T. Hardjono, and B. Haberman, *Security Requirements of IPv6 Anycast*, IETF, draft-dondeti-ipv6-anycast-security-00.txt, June 2001.

[15] G. Montenegro and C. Castelluccia, "Statistically Unique and Cryptographically Verifiable (SUCV) identifiers and addresses," in *NDSS'02*, February 2002.

[16] Greg O'Shea and Michael Roe, ""Child-proof Authentication for MIPv6 (CAM)," *ACM Computer Communications Review*, April 2001.

[17] Pekka Nikander, *An Address Ownership Problem in IPv6*, IETF, draft-nikander-ipng-address-ownership-00.txt, February 2001.

[18] D. Johnson and C. Perkins, *Mobile IP for IPv6*, IETF, draft-ietf-mobileip-ipv6-14.txt, work in progress, July 2001.

[19] T. Narten and R. Draves, *Privacy Extensions for Stateless Address Autoconfiguration in IPv6*, IETF, RFC3041, January 2001.

[20] D. Johnson and S. Deering, *Reserved IPv6 Subnet Anycast Addresses*, IETF, RFC2526, March 1999.

[21] B. Haberman and D. Thaler, *Unicast-Prefix-based IPv6 Multicast Addresses*, IETF, draft-ietf-ipngwg-uni-based-mcast-03.txt, October 2001.

[22] M. Handley and S. Hanna, *Multicast Address Allocation Protocol (AAP)*, IETF, draft-ietf-malloc-aap-04.txt, December 2000.

[23] P. Radoslavov and al, *The Multicast Address-Set Claim (MASC) Protocol*, IETF, RFC2909, September 2000.

[24] *IETF Seamoby Working Group*, http://www.ietf.org/html.charters/seamoby-charter.html.

[25] C. Ellison and al., *SPKI Certificate Theory*, IETF, RFC2693, September 1999.

[26] M. Blaze, J. Feigenbaum, J. Ioannidis, and A. Keromytis, *The KeyNote Trust-Man agement System Version 2*, IETF, RFC2704, September 1999.

[27] A.J. Menezes, P.C. Van Oorschot, and S.A. Vanstone, *Handbook of applied cryptography*, CRC Press, 1997.

[28] T. Aura, P. Nikander, and J. Leiwo, "DOS-resistant authentication with client puzzles," in *8th International Workshop on Security Protocols*, April 2000.

[29] C. Ellison and al., *SPKI Examples*, IETF Internet Draft, Internet Draft, Available at http://world.std.com/ cme/examples.txt, March 1998.

Contents

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399