# *Securing Group Management in IPv6 with CGA (Cryptographically Generated Addresses)*

*Claude Castelluccia (INRIA, France)*

*Gabriel Montenegro (SunLabs Europe, France)*

*February, 2002*

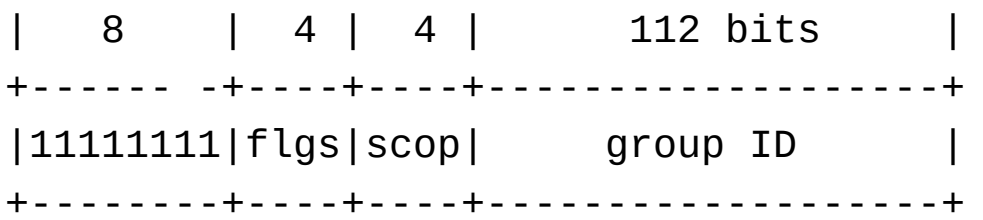
*(Slides by Claude Castelluccia)*

# Group Management in IP

- IP Multicast
  - packets sent to an IP Multicast group reach all its members
  - packets are duplicated in the net.:reduce traffic in the network
  - useful for group communications
    - games, video/audio conferences
    - files, content distributions
- IP Anycast
  - packets sent to an IP Anycast group reach only one member
    - the “closest” one
    - not always the same
  - useful for
    - service discovery (Home Agent, DNS,…)
    - load balancing

# IP Multicast (1)

- A Multicast group is defined by an IP multicast address

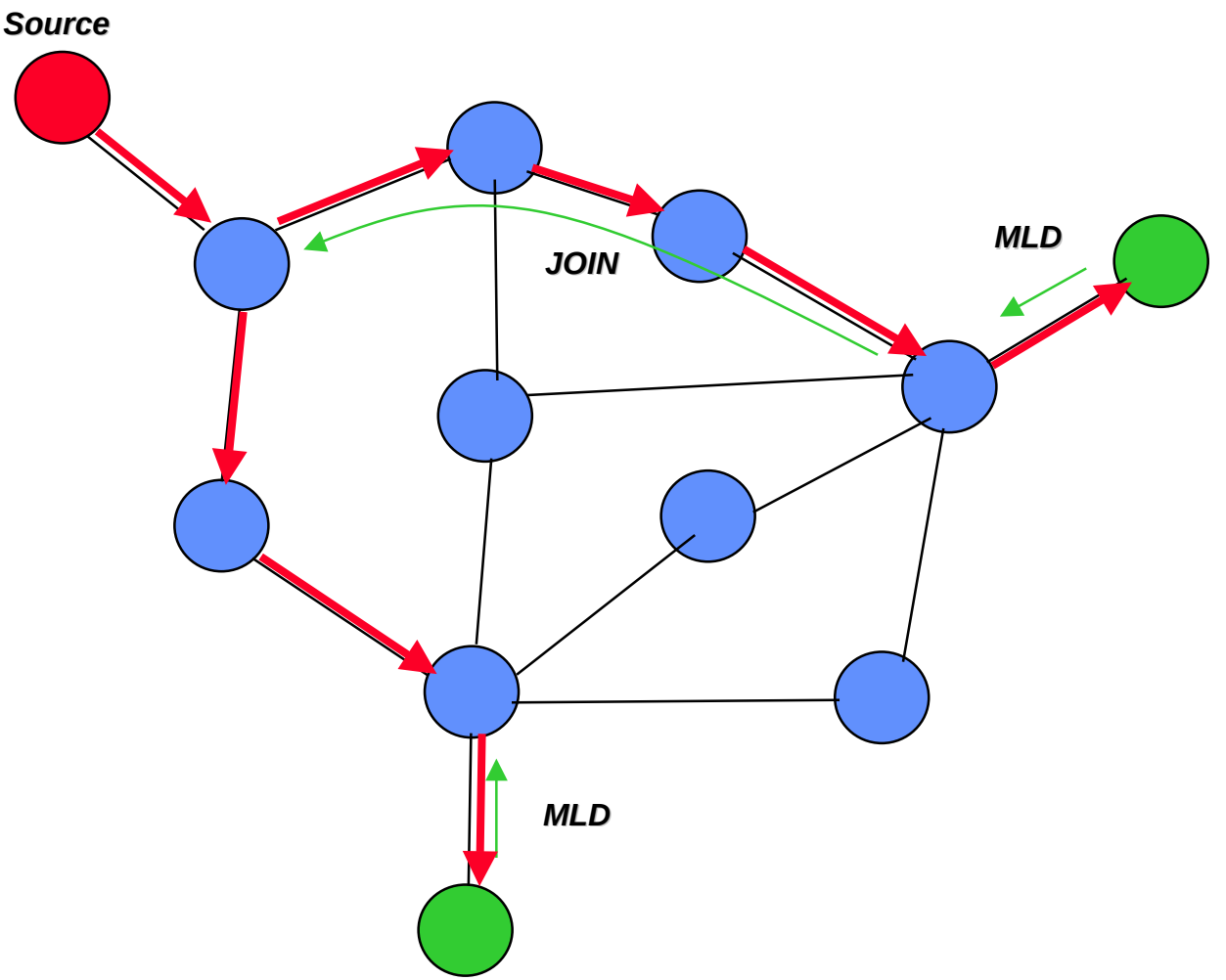



- A host that wants to send packets to a group
  - use the group's multicast address as destination address
  - the packets are duplicated in the network
- A host that wants to receive packets of a given group must explicitly *join* this group

# IP Multicast (2)

- 2 (independent) components:
  - the multicast routing: delivery tree construction from source to members (DVMRP, PIM, ….)
  - the group management: manages hosts' membership request (IGMP, MLD)
- The model:
  - a host that wants to join a group sends a *MLD message* to its local router…
  - the local router then informs the routing fabric of the new member

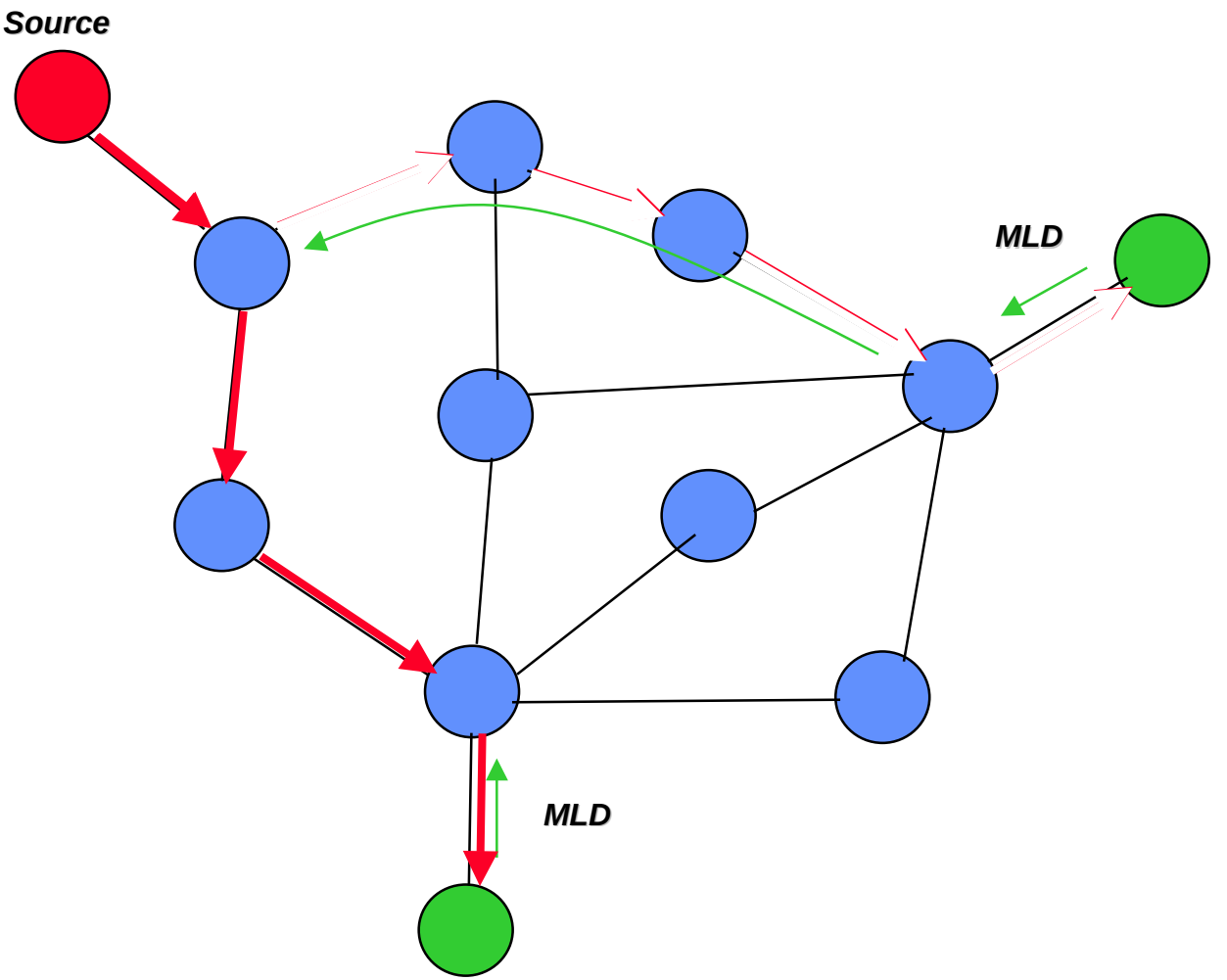
# IP Multicast (3)



# IP Anycast (1)

- An Anycast group is defined by an IP anycast address
  - IP anycast addresses are  allocated from the unicast address space
- A host that wants to send packets to an anycast group:
  - use the group's anycast address as destination address
  - the packets are routed by the network to the "*closest*" member
- The model:
  - a host that wants to join a group sends a MLD message to its local router…
  - the local router then informs the routing fabric of the new member

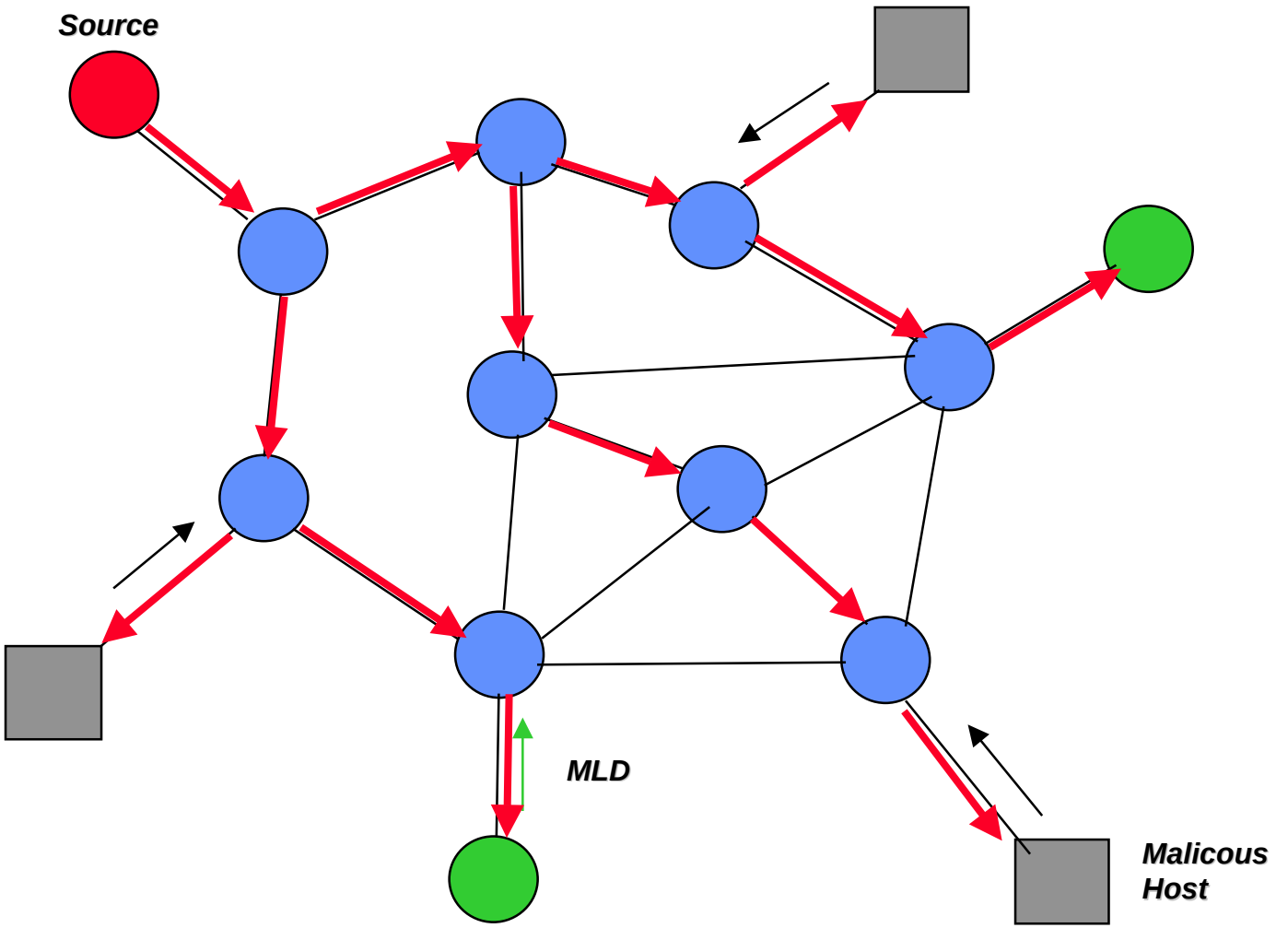
# IP Anycast (2)



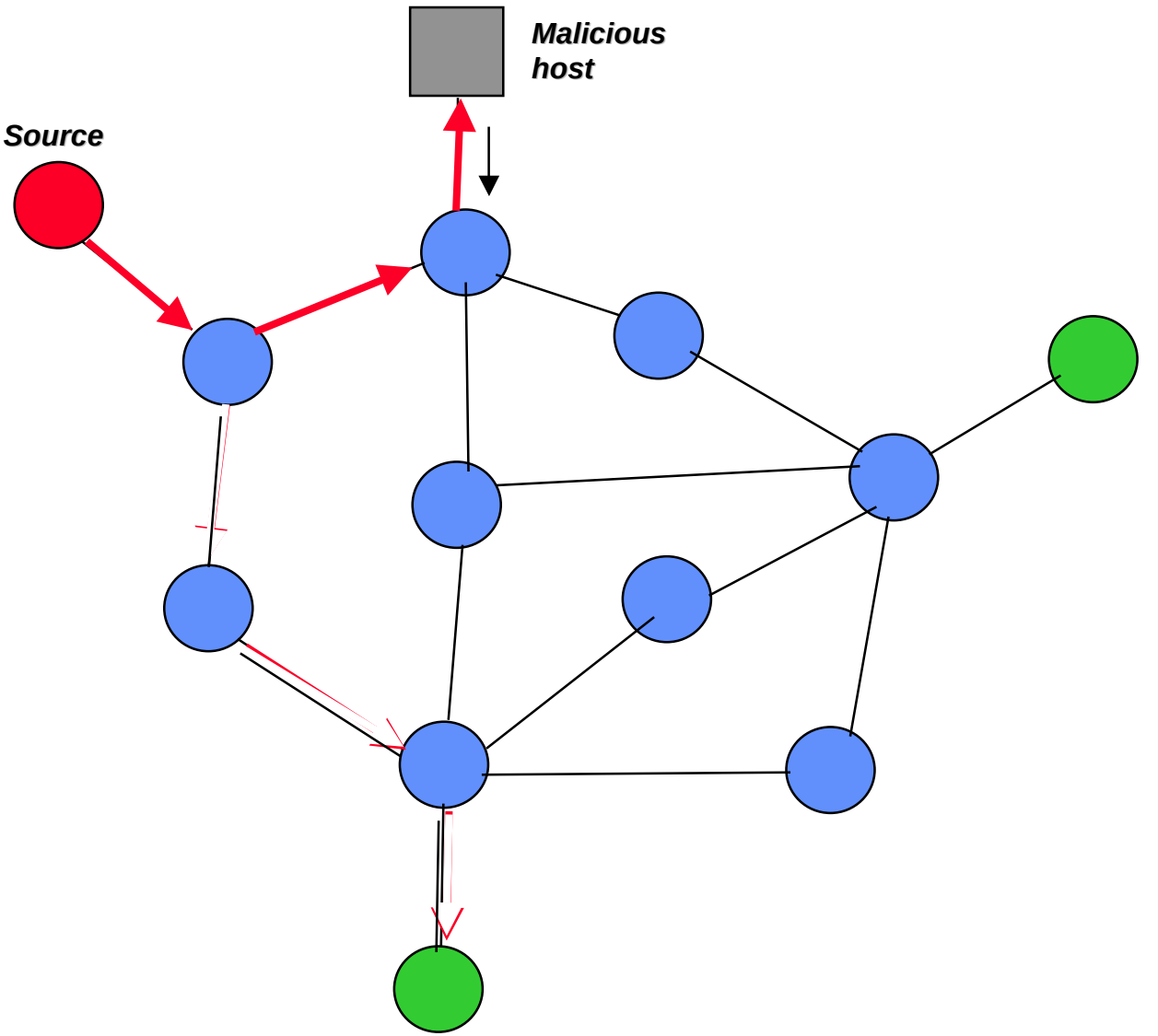
# Proof-of-Membership Problem

- Both IP Multicast and Anycast suffers from potential *DoS* attacks
- Anyone can issue a MLD-join message for a multicast or anycast address
  - *Multicast:* a malicious host can overflow the network by adding extra branches in the delivery tree!
  - *Anycast:* a malicious host can redirect the traffic and prevents the legitimate hosts from seeing it!
- The problem is that the (access) routers do not know who is authorized to join each group !
- This is the *proof-of-membership* problem!

# IP Multicast Attack



# IP Anycast Attack



# Related Work

- Multicast
  - encrypt Multicast traffic
    - an unauthorized host won't be able to understand the data …
    - …but can still join the group just for the purpose of flooding the net!
    - Does not really solve the authorization problem...
  - Use tokens
    - the group managers issue token for authorized members and send them to the (access) routers and members
    - does not scale very much!
    - Does not support mobile members!
  - IGMP security is becoming a hot topic at the IETF/MSEC and IRTF/GSEC!
- Anycast
  - not much so far….but part of the GSEC charter.

# Solution Requirements

- The solution should be “light”, i.e. not too computationally expensive.
- 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
  - should provide fast group registration to support fast handoff
  - should not assume any pre-established contexts in the routers
- The solution should be end-to-end
  - avoid reliance on global infrastructure such as PKI (Public Key Infrastructure) and TTP (Trusted Third Party).

# *CGA* Addresses

- Our proposal relies on Cryptographically Generated Addresses
- A CGA Address is generated from a Public Key
- IPv6 (unicast) address format:
  - prefix (64 bits) + HostID (64 bits) = 128 bits
- IPv6 CGA address, the HostID is generated from a public key (PK)
  - HostID = hmac-64(sha1(imprint), sha1(PK))
- CGA addresses are statistically unique
- A host can prove that it owns a CGA address by proving that it knows the corresponding private key, by signing its packets for example.
- NO infrastructure required: crypto relation between address and signature
- a malicious host could only steal a CGA address if:
  - 1- it can find the private key or
  - 2- it can find a public/private key pair that hashes to the target HostID
  - both are very difficult !

# *CGA* and Mobile IPv6

- CGA are very useful to solve the *proof-of-ownership* problem and secure redirection protocols such as Mobile IPv6
- In Mobile IPv6, a mobile host (MH):
  - has a Home Address (HoA) in its Home network
  - gets a Care-of Address (CoA) in the visited network
  - informs its Correspond Nodes (CN) of its current CoA by sending a Binding Update (HoA->CoA)
- If the CN and the MH do not have a pre-established secret that binds the HoA with the MH
  - any malicious host could send a Binding Update to redirect someone's else HoA…
  - PKI might be useful here …but we don't want to rely on them…
- If the HoA is a CGA:
  - a MH can prove that it owns its HoA by including its public key in the BU and by signing the resulting message…
  - no PKI is needed!

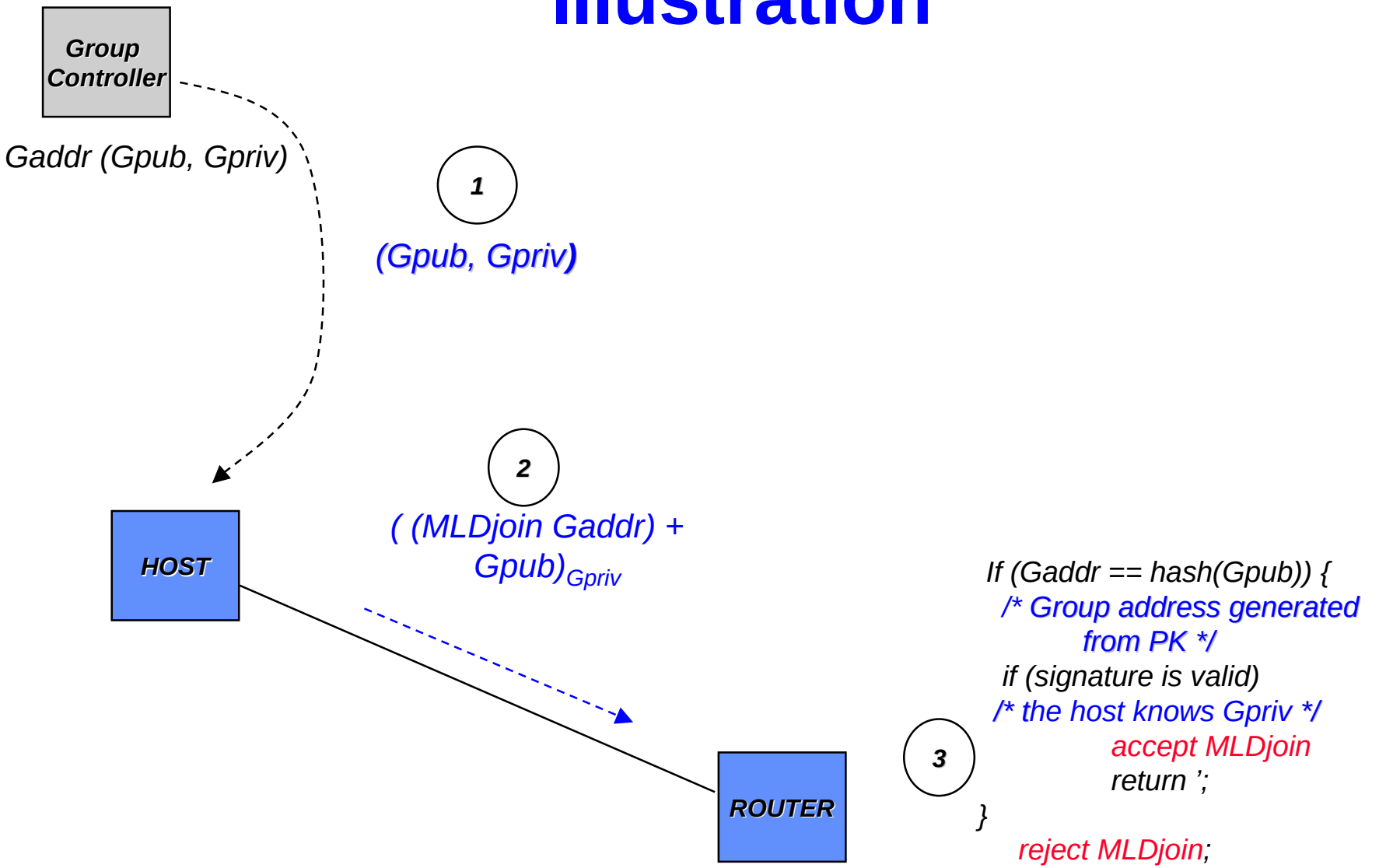
# Group CGA Addresses

- We define 2 new types of CGA addresses
  - Multicast CGA addresses (M-CGA)
    - GroupID = sha1-112(PK)
    - new S bit in the flag field distinguishes these addresses
  - Anycast CGA addresses (A-CGA)
    - same format as unicast CGA address i.e.
    - HostID = hmac-64(sha1(imprint), sha1(PK))

# Simplistic (Basic) Scheme

- a group is defined by a group CGA address (M-CGA or A-CGA)
- Any of the group controllers (could be any host, for example the *source*):
    - generates the group CGA address from a public/private key pair
    - authorizes members to join.
- *authorization scheme (off-line)*
  - the group manager authorizes the member by disclosing the group pub/priv. Key pair via a secure channel…
- *registration scheme (on-line)*
  - A host that wants to join the group MUST include the *group public key* with the MLD join message and sign it with the *group private key*...
  - The router can verify that the host is authorized to join the group if:
    - the Group CGA was generated from the group PK included in the MLD join msg
    - the signature is valid and therefore the host knows the group private key
    - Router performs a hash and a signature verification

# Illustration



# Simplistic Protocol's Properties

- Pros:
  - registration is *end-to-end*: all authorization info. in MLD message. Routers do not have to contact a TTP.
    - Low registration latency
    - scalable
    - mobile supported
  - no pre-established states are needed in the routers

- Cons:
  - membership revocation not provided
  - key disclosure problem
  - requires a secure channel between group manager and group members!

- We propose another scheme that uses authorization certificates such as SPKI, Keynote...

# SPKI Review

- The IETF SPKI Working Group has developed a standard form of digital certificates whose purpose is *authorization* (not authentication).
- a SPKI certificate has 5 fields:
  - *issuer*: the entity that provides the authorization
  - *subject*: the entity that acquires the authorization
  - *delegation* : a field that is set if the subject can delegate the permission
  - *authorization*: a field that specifies the permission being communicated
  - *validity*: defined how long the certificate is valid
- SPKI is *key-oriented* i.e. the issuer and subject are defined by keys or hashes of keys. No (name, key) binding, and therefore CA is necessary!
- Any key may issue a certificate: there is *no central* or trusted authority
  - anyone can issue certificates..
- SPKI provides delegation and chain of certificates...

# More Refined Protocol Overview

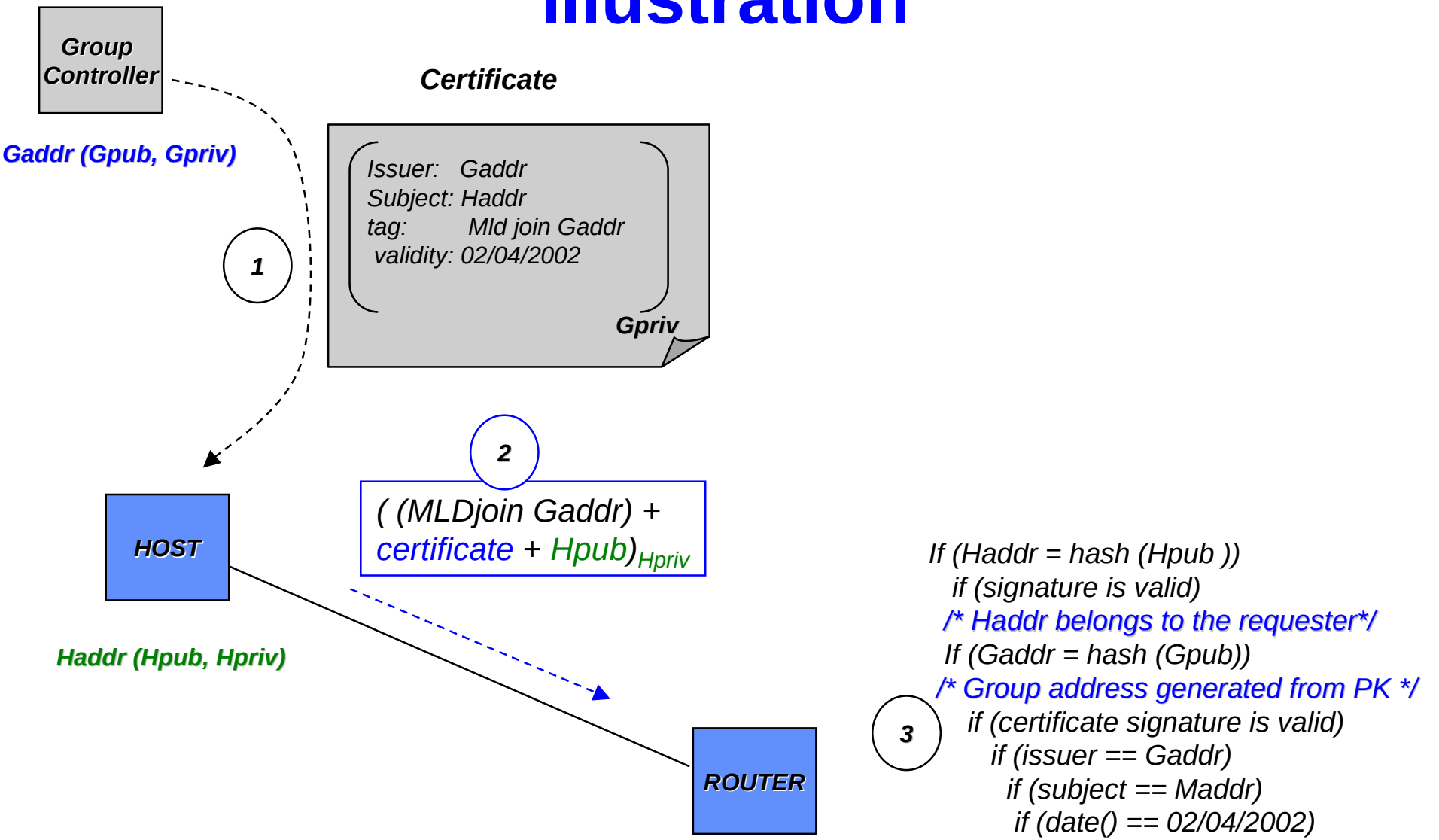
- Assumptions:
  - a group is defined by a group CGA address (M-CGA or A-CGA)
  - each member also has a (unicast) CGA address
  - group controller (possibly several)
    - generates the group CGA address from a public/private key pair
    - authorizes members to join by issuing certificates
- *Authorization management (off-line)*
  - a host that is authorized to join a group gets a SPKI certificate from the group controller
  - the certificates contains the *group public key, the host CGA address, a validity period* and is *signed with the group private key*.

# More Refined Protocol Overview (2)

*Registration management (On-line)*

- A host that wants to join the group must include its SPKI certificate (signed with the *Group public key* by a group controller) and *its CGA public key* in the MLD message and sign it with *its CGA private key* (the one used to generate its unicast address).
- A router that receives a MLD-join messages will only accept it if:
  - (1) *The host owns its CGA address*:
    - the CGA address contained in the certificate was generated from the host's *CGA public key*
    - the MLD message's signature is correct (the request knows the CGA private key and therefore owns the corresponding CGA address).
  - (2) *the MLD-SPKI certificate is valid (CGA has been authorized)*:
    - the Host's CGA address in the certificate is the same than the requester's one.
    - the group address was generated from the *Group public key*, the validity period is correct, the certificate signature is correct...

# Illustration



# MLD/SPKI Certificate

- Certificate format

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

- Example

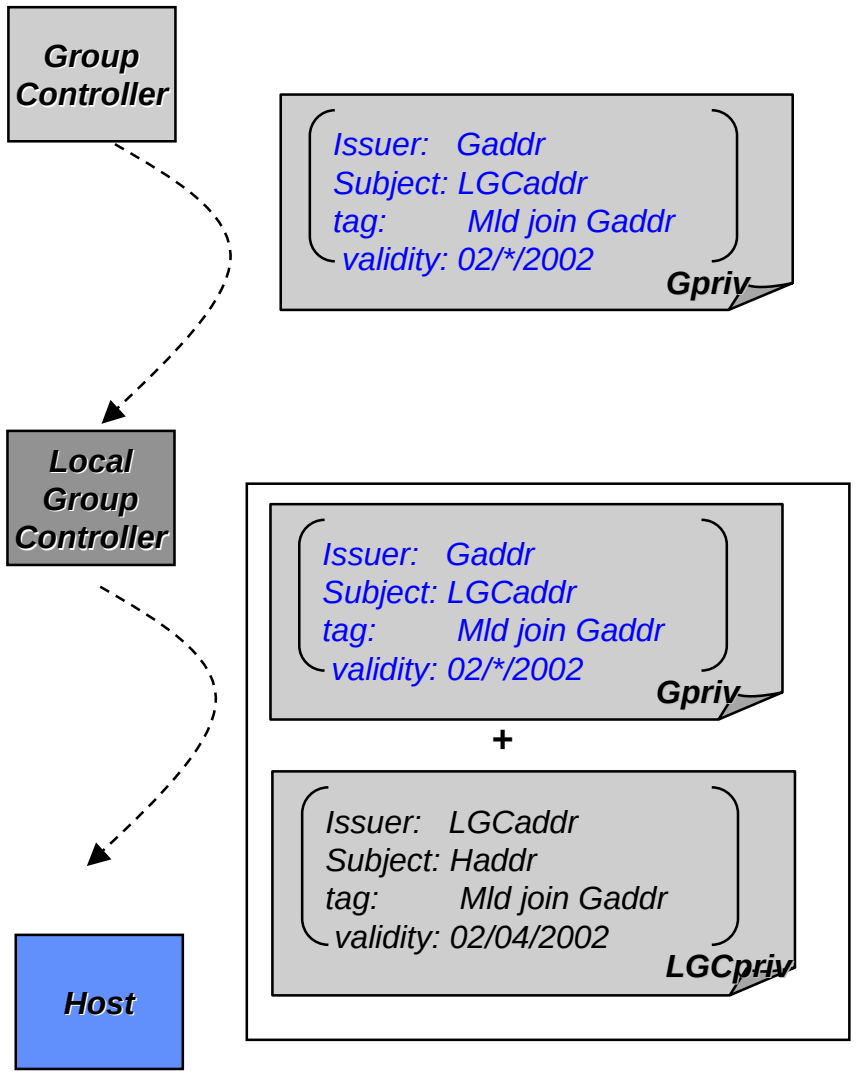
```
(cert
  (issuer  (addr FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678)
  (subject (addr 2001:660:1002:2000:7E45:F543:9C74:BA76)
  (tag (mldjoin 2B FF5E:45CF:4AB2:12CF:6745:23B1:FE82:9678 *)
  (propagate)
  (not-before ``2002-02-02_00:00:00'')
  (not-after  ``2002-03-02_00:00:00'')
)
```

# More Refined Proposal's Properties

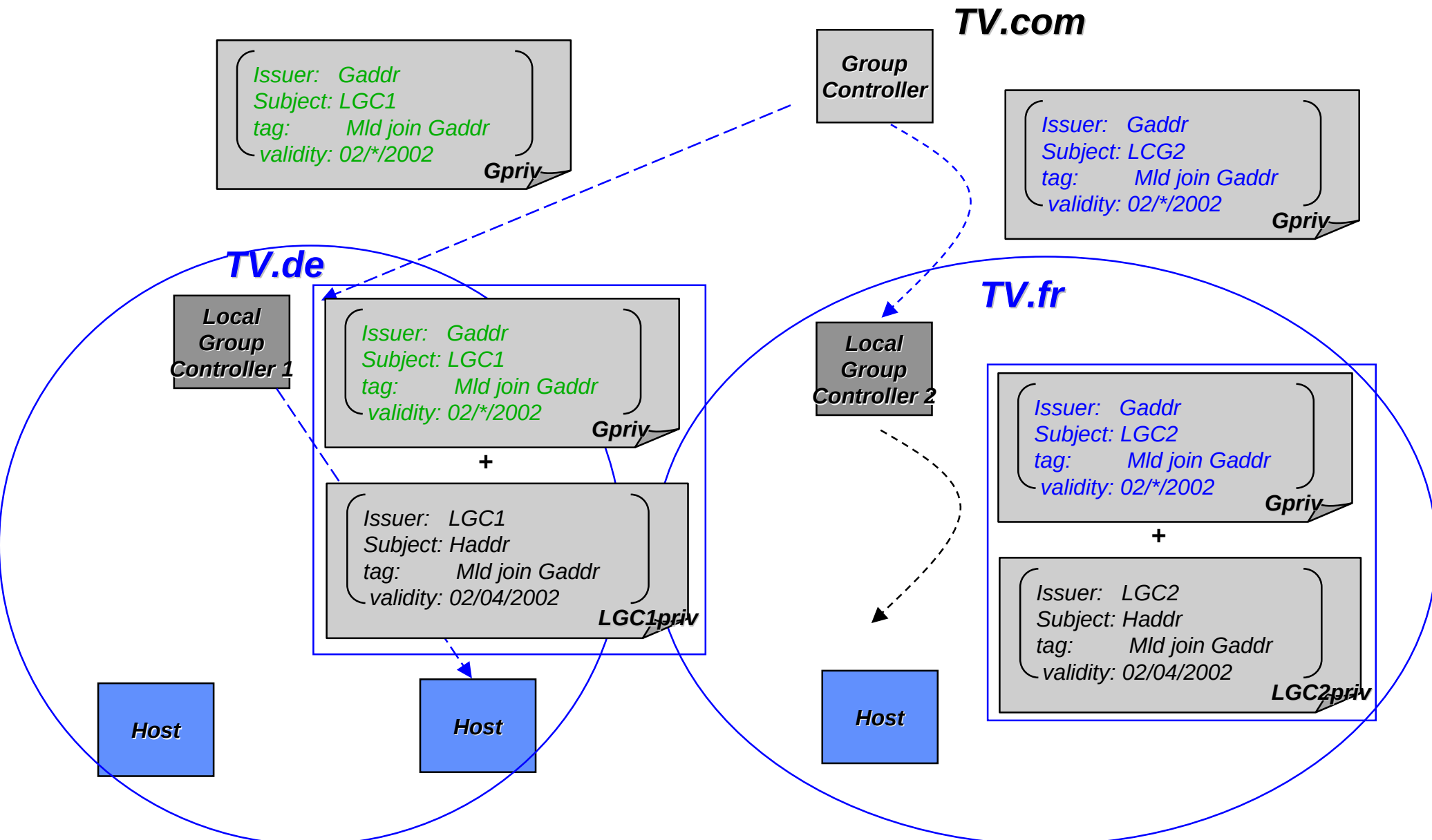
- *Membership expiration supported*
  - each certificate has a validity period field
- *No secure channel required*
  - the certificates are sent from the group controller to the member in clear
- *No private key disclosure risk*
  - the group controller never discloses the group private key
- *Scalable*
  - distributed
  - no TTP required: the routers can verify the authorization directly from the MLD msg
  - No pre-established context required in the routers : support mobile hosts

# Chain of Certificates

- The Group controller may delegate rights to join a group to a local group controller (LGC)
- The LGC can then authorize local hosts… the delegation certificates form a chain.
- When a host requests the authorization to the LGM, it gets the entire chain ...
- When a host sends a MLD-join message it must include the entire chain
- This is useful to design a hierarchical authorization scheme...

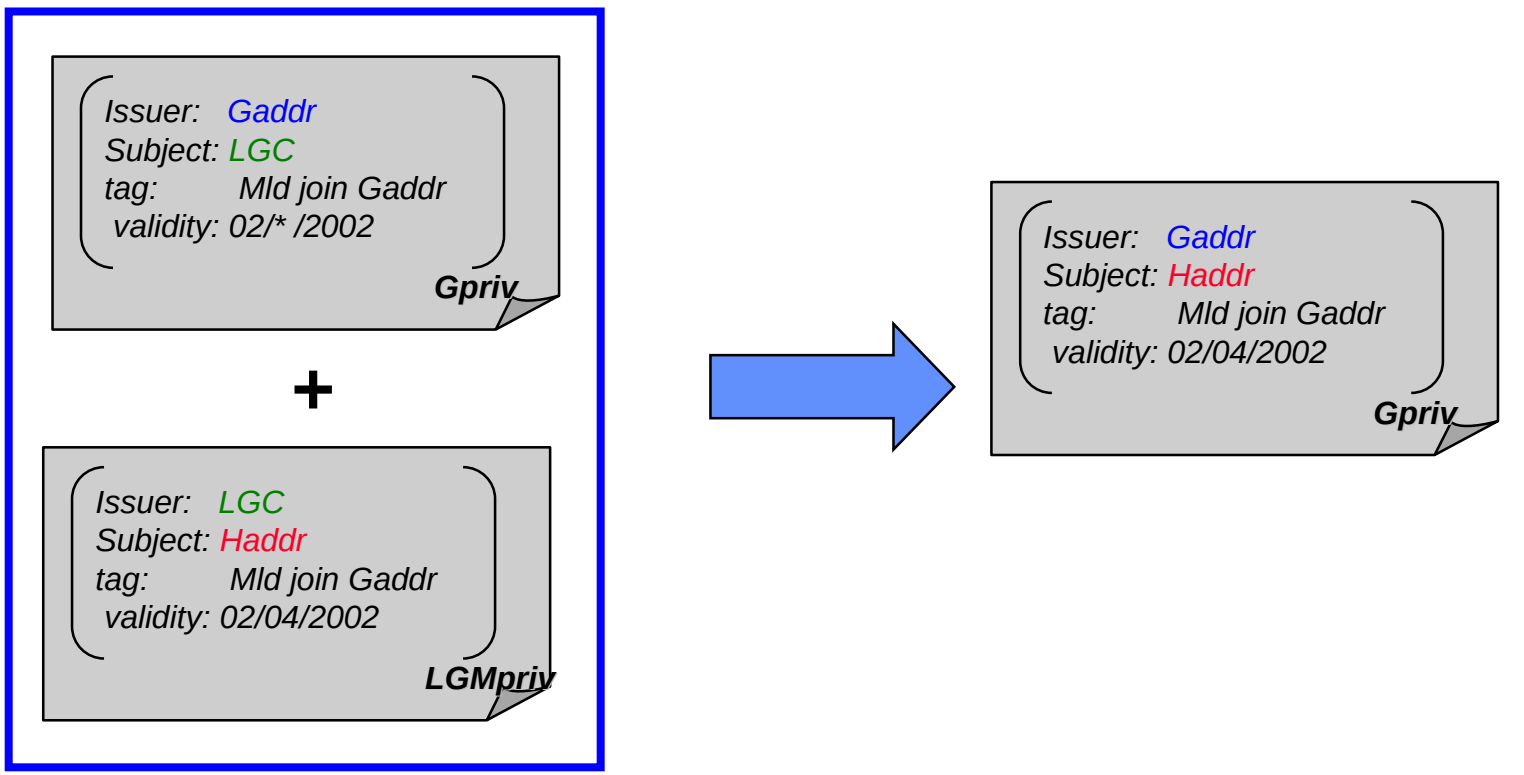


# Hierarchical Authorization Scheme (off-line)



# Chain Reduction

- Chain of certificates can be reduced through certificate reduction
  - $(K2 \text{ has } R1, t1)_{K1}$ and $(K3 \text{ has } R2, t2)_{K2}$ => $(K3 \text{ has } (R1 \Lambda R2), (t1 \Lambda t2))_{K1}$
- useful to reduce chain of certificates size and processing …



# Security Analysis/ CGA Collision

- a malicious host could join a group if it can find a public/private key that hashes to the group CGA
- it would then need to try $2^{(n-1)}$ keys on average (n is the size of the hash)
- M-CGA:
  - n=112. A malicious host must then try $2^{111}$ keys…
  - if the malicious host can performs 1 million hashes per second, it would need $8.23 \times 10^{19}$ years!
  - Since furthermore multicast groups have usually a limited life time, this attack is almost impossible!
- A-CGA:
  - n=63. A malicious host must then try $2^{63}$ keys…
  - if the malicious host can performs 1 million hashes per second, it would need 142.235 years!
  - Anycast addresses have a limited scope i.e. a member should physically be on the network it wants to attack. This requirement combined with the previous one makes this attack very unpractical.

# Future Work/ Open issues

- Our proposal makes extensive use of signatures which are costly operations
  - a member needs to sign its MLD messages …
  - a router must (1) verify the SPKI chain of certificates and (2) verify the MLD signature
    - (1) is only necessary when the chain of certificates is renewed and (2) is required for each MLD messages
- Potentials Solutions/future work
  - The MLD signatures generation and verification could be avoided by using SUCVp that would establish a secret key between the router and the member.
  - Certificate reduction helps in reducing the verification cost.
  - Elliptic Curve Crypto (ECC) could be very helpful here..
  - Smart cards could also help…..

# *Other Possible Extensions*

- DoS against the routers (false signature verifications)
  - Puzzle challenges
  - sucvP to derive shared keys and subsequent hmac verifications
  - Hash chain started at induction (when the certificate is communicated)
- DoS against the infrastructure
  - Unauthorized G-CGA addresses
  - Routing fabric could itself be a group authorizing groups to exist
- Traffic encryption by communicating key(s) at induction
  - Group controller generates keys K1 [t1;t2], K2 [t2;t3], …Kj[tj;tj+1]
  - At induction a host receives a certificate and relevant keys (perhaps encrypted, for example using RSA encryption instead of DSA)
  - Host encrypts with Kj and includes j in plain text

# Conclusions

- CGA addresses are very useful
  - …not only for the *proof-of ownership* problems of Mobile IPv6
  - ..but also for the *proof-of membership* problems in group communication.

- They are useful whenever you don’t want (or you can not rely) on the infrastructure to verify authorization
  - *IPv6 DaD*
  - *might be useful to secure adhoc networks*
  - *peer-to-peer protocols...*

# THANKS!