Key Management in IP Multicast

17.11.2006

Petri Jokela
petri.jokela@nomadiclab.com
IP multicast

- Receiver joins a group
- Data transmission tree is built
- Source Specific
  - S = source, G = group

Join (G, S)
IP multicast

Data is multiplied at routers

If data is encrypted, how to distribute keys?
Multicast Security

- IETF Multicast Security (MSEC) WG
- Initial target:
  - Security for one source, large number of receivers
- Currently:
  - Finish current work before next summer
  - Rechartering?
Multicast Security issues

• Integrity of data
  - Receiver: data is not modified

• Secrecy
  - Data cannot be seen by non-group members

• Source authentication
  - The data is coming from the correct source
  - With shared traffic encryption keys, requires other functions in IP multicast
  - Not considered in Key Management
Keying

• Keys
  – Shared: Traffic Enc. Key (TEK), Key Enc. Keys (KEK)
  – Point-to-Point: Registration association

• Problem: How to distribute shared keys?
  – Currently we have centralized server

• Use point-to-point link to deliver the KEK
  – Use a KEK to encrypt TEK; deliver e.g. using multicast data path

• Re-key: member joins or leaves a group
Security Architecture

Multicast Security Policies

Policy Server

Group Key Management

Group Controller / Key Server

Multicast Data Handling

Sender

Receiver

Policy Server

Group Controller / Key Server

Receiver
Group Security Association

- **Registration association**
  - point-to-point

- **Re-key association**
  - Shared keys
    - (Key Encryption Keys)
  - Re-key message e.g. using multicast

- **Data association**
  - Shared key
  - Data transmission
Logical Key Hierarchy
• Change in group (originally \( n \) users)
  
  – Only TEK + Ku\(_n\)
    
    • join: \( n+1 \) encryptions (TEK with Ku)
    • leave: \( n-1 \) encryptions (TEK with Ku)
  
  – TEK + 1 KEK + Ku\(_n\)
    
    • join: 1 encr. (TEK with KEK), \( n \) encr. (TEK with Ku)
    • leave: \( n-1 \) encr. (KEK with Ku), 1 encr (TEK with KEK)
Logical Key Hierarchy

- RFC2627
- Key Encryption Keys hierarchically
  - Less encryption operations
  - Less transmitted messages
- GSAKMP and GDOI define this as optional
- Defined but is it used?
  - E.g. not in 3GPP
Logical Key Hierarchy
Keys and hierarchy

Server

K₁

K₁₁
u₁

K₁
K₁₁
K₁₁
K₁u₂
K₁D

K₁

K₁₂
u₂

K₁
K₁₂
K₁₂
K₁u₃
K₁D

K₁

K2

K₂₁
u₃

K₂
K₂₁
K₂₁
K₂u₅
K₂D

K₂

K₂₂
u₄

K₂
K₂₂
K₂₂
K₂u₆
K₂D

Key Encryption Key Array
Host's key (registration association)
Data Encryption key
Logical Key Hierarchy
Node leaving, keys that have to be renewed

Server

$K_1$
- $K_{11}$
  - $K_1$
    - $K_{11}$
      - $u_1$
        - $K_{u1}$
          - $K_D$
  - $K_{12}$
    - $K_1$
      - $K_{12}$
        - $K_{u2}$
          - $K_D$

$K_2$
- $K_{21}$
  - $K_1$
    - $K_{11}$
      - $u_3$
        - $K_{u3}$
          - $K_D$
  - $K_{12}$
    - $K_1$
      - $K_{12}$
        - $K_{u4}$
          - $K_D$

- $K_{22}$
  - $K_2$
    - $K_{21}$
      - $u_5$
        - $K_{u5}$
          - $K_D$
  - $K_{22}$
    - $K_2$
      - $K_{22}$
        - $K_{u6}$
          - $K_D$
Logical Key Hierarchy

New keys: $K'_1$, $K'_{12}$, $K'_D$
Logical Key Hierarchy

Keys and hierarchy

E(K_{11})\{K'_{11}, K'_D\}

E(K_{u4})\{K'_{11}, K'_{12}, K'_D\}

E(K_{2})\{K'_D\}

K'_{1}

K'_{11}

K'_{12}

K'_{u4}

K'_{21}

K'_{22}

K'_{u5}

K'_{u6}

u_1

u_2

u_4

u_5

u_6
# Logical Key Hierarchy

## Table of required storage and re-key transmissions

<table>
<thead>
<tr>
<th>Users</th>
<th>Degree</th>
<th>Storage per User</th>
<th>Re-key transmissions (single key)</th>
<th>(multi key)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>2048</td>
<td>2</td>
<td>12</td>
<td>21</td>
<td>11</td>
</tr>
<tr>
<td>2187</td>
<td>3</td>
<td>8</td>
<td>20</td>
<td>14</td>
</tr>
<tr>
<td>131072</td>
<td>2</td>
<td>18</td>
<td>33</td>
<td>17</td>
</tr>
<tr>
<td>177147</td>
<td>3</td>
<td>12</td>
<td>32</td>
<td>22</td>
</tr>
</tbody>
</table>
Key Exchange Protocols

- Protocols defined in the IETF
  - Group Security Association Key Management Protocol (GSAKMP)
  - Multimedia Internet Keying (MIKEY)
  - Group Domain of Interpretation (GDOI)
- All define only multicast keying
  - Re-key SA, Data SA
- Registration not defined
  - E.g. IKE used for creating registration SA
Group Security Association Key Management Protocol (GSAKMP)
GSAKMP Trust Model

**Group Owner (GO)**
- Known and trusted party
- Creates the Policy Token (PT)
- PT contains
  - Identification of the group
  - Access Control rules
  - Authorization rules
  - SA, Security Policy information
  - PT verification information
GSAKMP Trust Model

- **GCKS**
  - Responsible for
    - key generation
    - traffic and key encryption keys
    - key distribution
    - re-keying
    - group membership management

- **S-GCKS**

- **GM** (receiver)

- **(receiver)**

- **(source)**

- **Group Owner**
Subordinate GCKS
- Support for distributed GCKS functions
- Same responsibilities as GCKS
  - BUT: TEK only from GCKS
- Register to the GCKS
- Verify GCKS's authority
**Group Member (GM)**

GM has to:
- Verify that all security related actions are authorized
- Use group keys properly

GSAKMP cannot control who sends data to the group
-> Multicast protocol and application issue
Senders authorized in PT

Sender configurations: 1, subset of GMs, or all
GSAKMP Assumptions

- GCKS or GO never compromised
- PKI is trustworthy (for cert validation)
- Compromised GM reported to GO
- No precise time dependency (in security related actions)
- Compromised GM cannot decrypt further traffic
- Confidentiality, integrity, multicast source authentication, and anti-replay protection for GSAKMP messages
Create "Registration SA"

Request to Join

Key Download

Key Download, Ack

Shared Keyed Group Session

Use e.g. IKEv1

Only when the GM is a source node

Download the required key information from the GCKS

Acknowledge the key download
Request to Join: \{Key Creation; Nonce_1;\} Signature [; Certificate]

Key Download: \{Member ID; Key Generation; PT; Key Download\} Signature [; certificate] 

Key Download, Ack: \{Notification Ack\} Signature
• Diffie-Hellman used for key generation
  – protecting further downloads from the GCKS
• GM leaves the group
  – LKH MAY be used for re-keying
  – “Many times it is best to rebuild the group”
  • Problem: This doesn't work with large groups
Multimedia Internet Keying
• Originally designed for real-time applications
  – Secure RTP

• Issues
  – Lower latency
  – heterogeneous networks
  – better performance for small, interactive groups
• Source handles GCKS functions (usually)
• No actual re-keying
  - Changes in groups handled by setting up a new connection
  - Cannot efficiently support big and unstable groups
  - MBMS (3GPP) defines re-keying
MIKEY - scenarios

peer-to-peer / one-to-many

Host A
  - Host B
  - Host C
  - Host D

many-to-many (centralized)

1-2-n:
- sender responsible for setting up security parameters

p-2-p: e.g. SIP call
-(mutual security agreement or each party for own outgoing)
MIKEY - scenarios

Small size group

1) Initiator acts as a GC (MIKEY)

2) Authorization information is delegated to other participants (not defined in MIKEY)
Larger groups

- GCKS responsible for setting up security parameters
- Not main focus in MIKEY

many-to-many (distributed)

many-to-many (centralized)
MIKEY – Generating a TGK

• TGK = TEK Generation Key

• Three methods
  – Pre-shared key
    • TGK transferred using the pre-shared key
    • Efficient but not scalable
  – Public-key based method
    • PKI needed for distributing public keys
  – Diffie-Hellman key exchange
    • For peer-to-peer case
MIKEY: pre-shared key

Host A (initiator)

Host B (responder)

Pre-shared key exchange

E(encr_key, IDi || \{TGK\} || MAC)

pre-shared key is used to generate encr_key and auth_key

HDR, T, RAND, [Idi], [Idr], {SP}, KEMAC

Optional response: HDR, T, [Idr], V
Host A (initiator)

E(encr_key, IDi || {TGK} || MAC)

E(pub.key R, env_key)

env_key is used to generate encr_key and auth_key

HDR, T, RAND, [Idi|CERTi], [Idr], {SP}, KEMAC, [CHASH], PKE, SIGNi

Public-key exchange / Retrieval

Host B (responder)

Optional response: HDR, T, [Idr], V
MIKEY: Diffie-Hellman

Host A (initiator)

Host B (responder)

Public-key exchange / Retrieval

HDR, T, RAND, [Idi|CERTi], [Idr], {SP}, DHi, SIGNi

HDR, T, [Idr|CERTr], Idi, Dhr, Dhi, SIGNr

TGK is the shared secret calculated from DH values
Group Domain Of Interpretation
• Registration association with ISAKMP phase 1
• GDOI defines
  – Re-key association setup
  – Data association setup
• TEK & KEK key transfer
  – GROUPKEY_PULL: initiated by the member
  – GROUPKEY_PUSH: initiated by the GCKS
GDOI: GROUPKEY_PULL

Member

GCKS

Authentication

Use e.g. IKEv1

HDR*, HASH(1), Ni, ID

HDR*, HASH(2), Nr, SA

HDR*, HASH(3), [KE_I], [CERT], [POP_I]

HDR*, HASH(4), [KE_R], [SEQ], KD, [CERT], [POP_R]

Liveliness check:
If Nr in HASH(3), calculate DH, install SA
GDOI: GROUPKEY_PULL

Member

GCKS

Authentication

Use e.g. IKEv1

Liveliness check:
If Nr in HASH(3),
calculate DH, create SA

HDR*, HASH(1), Ni, ID

HDR: ISAKMP header
HASH: prf(SKEYID_a, M-ID | Ni | ID )
Ni: Initiator Nonce
ID: Group ID to join
Use e.g. IKEv1

Liveliness check:
- If Nr in HASH(3), calculate DH, create SA

Authentication

GDOI: GROUPKEY_PULL

Member

GCKS

HDR*, HASH(1), Ni, ID

HDR*, HASH(2), Nr, SA

 HDR: ISAKMP header
 HASH: prf(SKEYID_a, M-ID | Ni_b | Nr | SA )
 Nr: Responder Nonce
 SA: Security Association Policy Payload
e.g. SPI, traffic/re-key, POP algo, ...

HASH:
prf(SKEYID_a, M-ID | Ni_b | Nr | SA )
GDOI: GROUPKEY_PULL

HDR: ISAKMP header
HASH: \text{prf}(SKEYID_a, M-ID | Ni_b | Nr_b [ | KE_I ] [ | CERT ] [ | POP_I ])
KE_I: Diffie-Hellman value for key generation
CERT: Certificate, if some other identity is used (than in Phase 1)
POP_I: Proof of Possession (signature)

Liveliness check:
If Nr in HASH(3), calculate DH, create SA
GDOI: GROUPKEY_PULL

HDR: ISAKMP header
HASH: prf(SKEYID_a, M-ID | Ni_b | Nr_b [ | KE_R ] [ | SEQ ] | KD [ | CERT ] [ | POP_R ] )
KE_R: Diffie-Hellman value for key generation
SEQ: Sequence number (PULL key exchange)
KD: TEK, KEK (depending on definitions in SA)
CERT: Certificate, authorization for providing keys
POP_R: Proof of Possession (signature)

Liveliness check:
If Nr in HASH(3), calculate DH, create SA
**GDOI: GROUPKEY_PUSH**

**Member**

**GCKS**

**Authentication**

**Use e.g. IKEv1**

- **HDR**: ISAKMP header
- **SEQ**: Sequence number (PUSH key exchange)
- **SA**: Security Association Policy Payload
- **KD**: TEK, KEK (depending on definitions in SA)
- **CERT**: Certificate, authorization for providing keys
- **SIG**: Proof of Possession (signature)
Host Identity Protocol
Host Identity Protocol

• IP address roles currently
  – Locator: describes the host's topological location in the network
  – Identifier: identifies the host

• Problems
  – How to know who is at the other end – IP address is not enough
  – Mobility difficult
**HIP: Host Identities**

- Host Identity (HI): public key of a key pair
  - Hosts can authenticate each other
- Secure binding between HI and IP address
- Locator is used only for data routing
  - IP address not needed once the packet arrives
  - ESP mandatory (currently)
    - SPI used to find a correct ESP SA
    - HITs are mapped to the SA
    - Checksums using HITs
A new layer

- New layer
- IP <-> HI mapping
- Sockets bound to HIs, not IPs
- Transparent to applications
HIP: negotiation

- 4-way message exchange
  - Base Exchange (BEX)
  - Host authentication: public and private keys
  - Diffie-Hellman: common keying material
  - Creates HIP association
- Data traffic protection
  - ESP currently mandatory
  - ESP SA setup during BEX
  - Other protocols may be defined later
Other HIP features

- HI long => HIT (IPv6), LSI (IPv4)
- IPv4/v6 interoperability
  - mobility between v4 and v6 networks
  - v4 and v6 applications can communicate
    - Some limitations due to applications
- Easy mobility
  - Dynamic IP – HIT mapping
  - invisible to applications
- Multihoming support (based on mobility)
  - Independent of access technology
HIP Base Exchange

Initiator

ESP SA established

I1 <HIT_I, (HIT_R or NULL)>

R1 <HIT_I, HIT_R, ESP_TR, Challenge>

I2 <HIT_I, HIT_R, ESP_TR, SPI, Response, Authentication>

R2 <HIT_I, HIT_R, SPI, Authentication>

Respender

ESP SA established

Security Context Established

Data over ESP SA
HIP: v4 and v6 interoperability

HI
HIT / LSI
IP

TCPv4
TCPv6
IPv4
IPv6

Host identity

Link layer

TCPv4
TCPv6
IPv4
IPv6

Link layer
HIP and current solutions

- IPsec: considered ~hard to configure
- Mobile IP large and complex
- Mobile IPv4 and IPv6 do not work together
- No simple solution for multihoming
- LOC: >100,000 vs. ~20,000
**HIP Registration Protocol**

**Initiator**
- Select service, register

**Responder**
- Inform about services

1. **I1 <...>**
2. **R1 <... REG_INFO>**
3. **I2 <... REG_REQ>**
4. **R2 <... REG_RESP>**

Inform about services
Merging HIP and GDOI
GDOI and HIP

• GDOI: two phases
  • 1) replace phase 1 with HIP
    - Registration association
    - New “service” needed (GCKS)
  • 2) Group Key Exchange
    - For now, use the GDOI phase 2
      • SKEYID_a (for hashes) from the negotiated keying material
    - In the future; HIP has UPDATE mechanism, define multicast key transfer in UPDATE
HIP “Phase 1: registration”

Member

I1 <...

R1 <... REG_INFO (GCKS)>

I2 <... CERT, REG_REQ (GCKS)>

R2 <... REG_RESP>

GCKS

Inform about GCKS, challenge, D-H parameters, GCKS authentication

Member authentication, Authorization (cert), D-H params, challenge solution, SPI

GCKS Authorization (cert), SPI (registration SA),
UPDATE messages are not encrypted, key information has to be inside ENCRYPTED parameter.

**HIP “Phase 2: group keys (PULL)”**

**Member**

UPDATE <SEQ, GK_REQ, HMAC, SIGN>

UPDATE <SEQ, ACK, ENCRYPTED (Kmember, {SA, KD})>

**GCKS**

UPDATE <ACK>
UPDATE messages are not encrypted, key information has to be inside ENCRYPTED parameter.

Member

UPDATE <SEQ, ACK, ENCRYPTED (KEK, {SA, KD}, HMAC, SIGN)>

GCKS

UPDATE <ACK> (?)
Advantages / disadvantages

• For HIP hosts
  - Small updates to existing HIP implementations
  - No need for other types of security negotiations

• Mobility management
  - Mobile Member updates location to the GCKS
  - Does not solve the IP multicast (“data connection”) mobility

• Future work
  - Further optimization: Group UPDATE