Secure Multicast Communication

- **Examples:** Live broadcast of a match, stock quotes, video conferencing.
- **Security** has become a major issue.

**Challenges:**
1. Secrecy of messages.
2. Authenticity:
   a) Group Authenticity
   b) Source Authenticity
3. Anonymity
4. Access Control
Key Exchange

- Main Step: Key Exchange is the main step in multicast communication.
- Members communicate to set up a common key that is then used to encrypt messages.
- Several key exchange protocols exist today.
- Examples:
  1. 2-party: IKE, JFK.
  2. Multi-party: GDH.1, GDH.2, GDH.3.
Security Issues

- Depends on kind of adversary:
  1. Passive Adversary: Can read messages but not inject/delete/modify messages.
  2. Active Adversary: Can read/modify/delete messages.
Passive Adversary

- Secrecy: The key exchanged must be a secret.
- Key Agreement: All participants in the protocol agree on the same key.
- Resistance to Known-Key attacks: A key compromised in one session cannot help in compromising keys in other sessions.
- Key Independence: For dynamic memberships, old keys cannot be known to new members and new keys cannot be known to old members.
Active Adversary

- **Authentication**: Each participant has the assurance that only legitimate users belong to the group.
- **Perfect Forward Secrecy (PFS)**: Compromise of long-term keys cannot result in the compromise of past session keys.
- **Resistance to Known-Key attacks**: Session keys known in one session cannot help an active adversary to impersonate one of the protocol parties in another session.
Five Group Key Exchange (GKE) protocols are proposed.

First three assume static group membership.

Last two deal with member addition and deletion.

We will focus on the first three.

Proved secure against passive attacker.

Ateniese, Steiner et al proposed an authenticated GKE protocol that “tolerates” active adversary.
Let 'g' be the generator of a group.

For 4 participants the protocol works as follows:

- Each participant $P_1$, $P_2$, $P_3$ and $P_4$ generates a nonce $n_1$, $n_2$, $n_3$ and $n_4$ respectively.
- $P_1$ sends $\{g^{n_1}\}$ to $P_2$.
- $P_2$ sends $\{g^{n_1}, g^{n_1n_2}\}$ to $P_3$.
- $P_3$ sends $\{g^{n_1}, g^{n_1n_2}, g^{n_1n_2n_3}\}$ to $P_4$.
- $P_4$ sets group key to $g^{n_1n_2n_3n_4}$.
- $P_4$ sends $\{g^{n_4}, g^{n_1n_4}, g^{n_1n_2n_4}\}$ to $P_3$.
- $P_3$ sends $\{g^{n_4n_3}, g^{n_1n_4n_3}\}$ to $P_2$.
- $P_2$ sends $\{g^{n_4n_3n_2}\}$ to $P_1$. 

GDH.1
• $P_1$ sends $\{g^{n_1}\}$ to $P_2$.
• $P_2$ sends $\{g^{n_1}, g^{n_2}, g^{n_1n_2}\}$ to $P_3$.
• $P_3$ sends $\{g^{n_1n_2}, g^{n_1n_3}, g^{n_2n_3}, g^{n_1n_2n_3}\}$ to $P_4$.
• $P_4$ sets group key to $g^{n_1n_2n_3n_4}$.
• $P_4$ broadcasts $\{g^{n_1n_2n_4}, g^{n_1n_3n_4}, g^{n_2n_3n_4}\}$ to everyone.
GDH.3

- $P_1$ sends $\{g^{n_1}\}$ to $P_2$.
- $P_2$ sends $\{g^{n_1n_2}\}$ to $P_3$.
- $P_3$ sends $\{g^{n_1n_2n_3}\}$ to $P_4$.
- $P_4$ sets group key to $g^{n_1n_2n_3n_4}$.
- $P_4$ broadcasts $\{g^{n_1n_2n_3}\}$ to everyone.
- $P_3$ computes inverse and sends $\{g^{n_1n_2}\}$ to $P_4$.
- $P_2$ computes inverse and sends $\{g^{n_1n_3}\}$ to $P_4$.
- $P_1$ computes inverse and sends $\{g^{n_2n_3}\}$ to $P_4$.
- $P_4$ broadcasts $\{g^{n_1n_2n_4}, g^{n_1n_3n_4}, g^{n_2n_3n_4}\}$ to everyone.
Comparison of GDH protocols

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Rounds</th>
<th>Messages</th>
<th>Exponentiations per $P_i$</th>
<th>Total Exponentiations</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDH.1</td>
<td>$2(n-1)$</td>
<td>$2(n-1)$</td>
<td>$(i+1)$ for $i &lt; n$, n for $P_n$</td>
<td>$\frac{(n+3)n}{2} - 1$</td>
</tr>
<tr>
<td>GDH.2</td>
<td>$n$</td>
<td>$n$</td>
<td>$(i+1)$ for $i &lt; n$, n for $P_n$</td>
<td>$\frac{(n+3)n}{2} - 1$</td>
</tr>
<tr>
<td>GDH.3</td>
<td>$n+1$</td>
<td>$2n-1$</td>
<td>$4$ for $i &lt; n-1$, $2$ for $n-1$, n for $P_n$</td>
<td>$5n-6$</td>
</tr>
</tbody>
</table>
Authenticated GDH.2

- Above protocols tolerate only passive adversary.
- For static membership, an easy fix to GDH.2 “tolerates” active adversary.
- An attack was later found against AGDH.2 in which an adversary behaving as a legitimate participant in one session can learn the key in another session of which it is not a member.
AGDH.2

- $P_4$ shares long term shared keys $K_{14}$, $K_{24}$, $K_{34}$ with $P_1$, $P_2$ and $P_3$.
- $P_1$ sends $\{g^{n_1}\}$ to $P_2$.
- $P_2$ sends $\{g^{n_1}, g^{n_2}, g^{n_1n_2}\}$ to $P_3$.
- $P_3$ sends $\{g^{n_1n_2}, g^{n_1n_3}, g^{n_2n_3}, g^{n_1n_2n_3}\}$ to $P_4$.
- $P_4$ sets group key to $g^{n_1n_2n_3n_4}$.
- $P_4$ broadcasts $\{g^{n_1n_2n_4k_{34}}, g^{n_1n_3n_4k_{24}}, g^{n_2n_3n_4k_{14}}\}$ to everyone.
ProVerif
Bruno Blanchet

- Protocols can be modeled as applied pi-calculus processes.
- Explicit modeling of attacker not required.
- Possible to state if an attacker is passive or active.
- Reasonable arithmetic properties of encryption/decryption can be specified as mathematical equations in ProVerif.
- Security proofs are done by querying ProVerif if an attacker knows a key or content of an encrypted message.
GDH.2 in ProVerif

- free c01, c30, c12, c31, c23, c32, c, sc.
- private free m, sameKey, p04, p14, p24, p34.
- (* Check if attacker can recover m and that all participants generate the same key *)

- query attacker:m;
- attacker:sameKey.
- (* Shared key cryptography *)

- fun enc/2.
- fun dec/2.
- equation dec(enc(x,y),y) = x.
GDH.2 Contd.

- (* Diffie-Hellman functions *)
- data g/0.
- fun exp/2.
- equation \( \exp(\exp(g,x),y) = \exp(\exp(g,y),x) \).
- equation \( \exp(\exp(\exp(g,y),z),x) = \exp(\exp(\exp(g,y),x),z) \).
- equation \( \exp(\exp(\exp(g,y),z),x) = \exp(\exp(\exp(g,x),z),y) \).
- equation \( \exp(\exp(\exp(\exp(g,x),y),z),t) = \exp(\exp(\exp(\exp(g,x),y),t),z) \).
- equation \( \exp(\exp(\exp(\exp(g,x),y),z),t) = \exp(\exp(\exp(\exp(g,x),z),t),y) \).
- equation \( \exp(\exp(\exp(\exp(g,x),y),z),t) = \exp(\exp(\exp(\exp(g,y),z),t),x) \).
- reduc inv(\( \exp(\exp(\exp(\exp(g,x),y),z),t),t) = \exp(\exp(\exp(\exp(g,x),y),z) \).
- inv(\( \exp(\exp(\exp(\exp(g,x),y),z),t),z) = \exp(\exp(\exp(g,x),y),t) \).
- inv(\( \exp(\exp(\exp(\exp(g,x),y),z),t),y) = \exp(\exp(\exp(g,x),z),t) \).
- inv(\( \exp(\exp(\exp(\exp(g,x),y),z),t),x) = \exp(\exp(\exp(g,y),z),t) \).
- inv(\( \exp(\exp(\exp(g,y),z),t),y) = \exp(\exp(g,t),z) \).
- inv(\( \exp(\exp(\exp(g,y),z),t),z) = \exp(\exp(g,y),t) \).
- inv(\( \exp(\exp(\exp(g,y),z),t),t) = \exp(\exp(g,y),z) \).
- inv(\( \exp(g,y),z),z) = \exp(g,y) \).
- inv(\( \exp(g,y),z),y) = \exp(g,z) \).
GDH.2 Contd.

- param attacker = passive.

- let p0 = new n0;
  - out(c01,exp(g,n0)); (* g^n0 *)
  - in(c30,u);
  - let comk0 = exp(u,n0) in
    - out(c, enc(m,comk0));
    - out(p04,comk0).

- let p1 = new n1;
  - in(c01,v);
  - out(c12,(v,exp(g,n1),exp(v,n1))); (* (g^n0, g^n1, g^n0n1) *)
  - in(c31,w);
  - let comk1 = exp(w,n1) in
    - out(p14,comk1).
let p3 = new n3;
    in(c23,(u,v,w,x)); (* g^n0n1, g^n0n2, g^n1n2, g^n0n1n2 *)
    out(c30,exp(w,n3)); (* g^n1n2n3*)
    out(c31,exp(v,n3)); (* g^n0n2n3*)
    out(c32,exp(u,n3)); (* g^n0n1n3*)
    let comk3 = exp(x,n3) in
    out(p34,comk3).

let p4 =
    in(p04, k0);
    in(p14, k1);
    in(p24, k2);
    in(p34, k3);
    if k0 = k1 then
        if k1 = k2 then
            if k2 <> k3 then
                out(sc,sameKey)
            else
                0
        else
            out(sc, sameKey)
    else
    out(sc, sameKey).

process ( p0 | p1 | p2 | p3 )
Conclusion

- Modeled GDH.1, GDH.2, and GDH.3 protocols in ProVerif.
  - Proved they preserve secrecy and key agreement against a passive attacker.
- Modeled AGDH.2 to allow active adversary.
  - ProVerif was not able to prove/disprove its security properties.