



THE SKEMASNET PROJECT

SESSION KEY MANAGEMENT IN A SPONTANEOUS NETWORK

HGP Team

Alauzet Pierre, Park Hyunho , Ricciardi Gianni M.

TABLE OF CONTENTS

1. Proposal review
2. Design specifications
3. Implementation

BACKGROUND

- ❑ In the context of spontaneous networks, we focused on session key management during the merger of two networks and when a user leaves the network
- ❑ We surveyed some paper about GKA to compare it to our idea
- ❑ We started to simulate our idea using the ns2 simulator

PROBLEM DEFINITION

❑ In the private networks they are using common session keys for secure communication,

- ❑ When merging network(s) it needs to manage session keys.
 - Creating a new session key or choosing one of them for the merged network.
 - Share the new session key to all members.

❑ Related works

- ❑ GKA(Group Key Agreement)-key paper
 - A mechanism to create a common session key for a group of users.
 - Each member provide a public contribution for creating a common session key.
 - It can share a common session key **without** the use of a *secure* channel.

❑ Problem

- ❑ Require creation of a new session key at every times when the network members are changed(join, leave, merge, separating)
- ❑ Requires $2n$ messages exchanges for creation and distribution of a new session key.
- ❑ Each message for exchange a session key is in size of encrypting $\text{SizeOfSessionKey} * 2 * n$
 - ❑ Ex) if the key size is 256bit, and size of node is 100=> $256 * 100 * 2 = 51200$ bit = 6.4kbytes.

DESIGN CASE: MERGING 2 NETWORKS

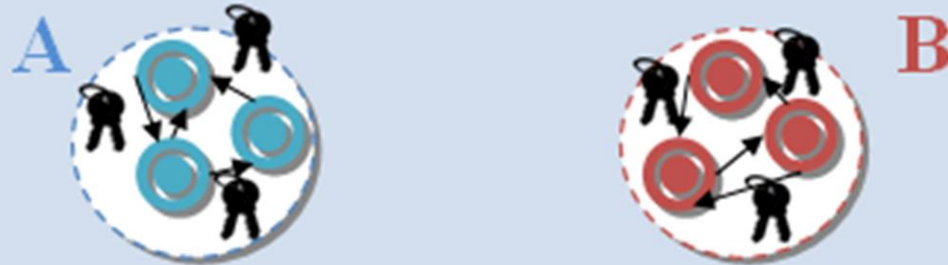
□ Preface

*Decision making comes from human interaction.
Two groups meet and decide to merge their networks and then they choose two leaders (one per network).*

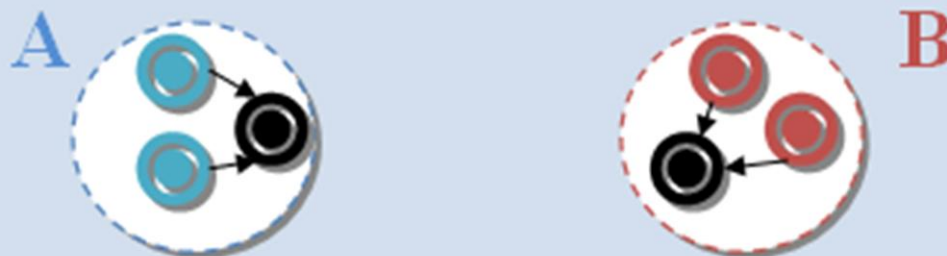
DESIGN CASE: MERGING 2 NETWORKS (CONT.)

□ Initial phase

Thanks to the joining procedure, each user has the public key of all other users in the same network



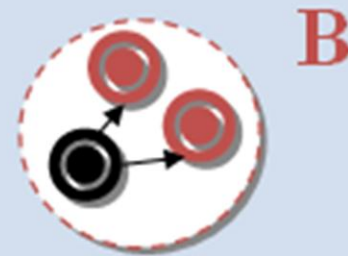
All users of each network, after a *social* agreement, select the leader on the users list.



DESIGN CASE: MERGING 2 NETWORKS (CONT.)

□ Initial phase (cont.)

When the leader receives a signed *election message* from each user, he assumes the role of leader and sends a signed *confirmation message* to all users



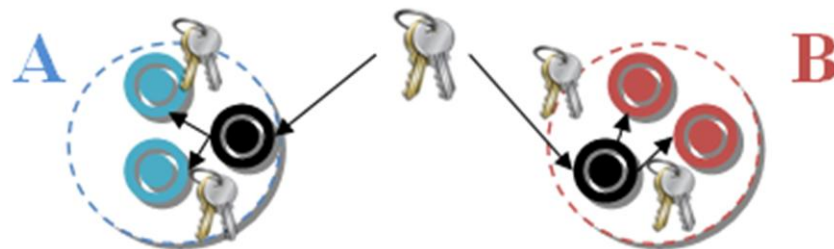
The two leaders meet **face to face** and share a new session key using a *secure side channel*



DESIGN CASE: MERGING 2 NETWORKS (CONT.)

□ Propagation phase

Each leader propagates the new key  to his/her members of pre-existing network through the network itself using a signed message.



□ Communication phase


Users from both original networks can communicate one to each other




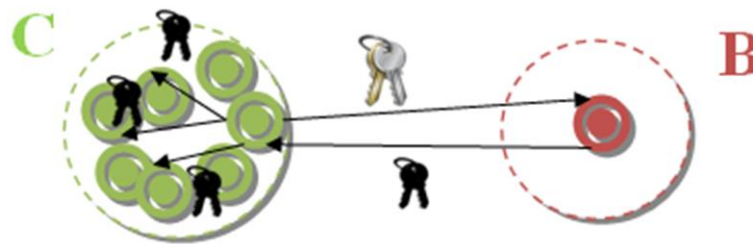
DESIGN CASE: MERGING 2 NETWORKS (CONT.)

□ Joining phase

If a new node requests to join the new network, it is performed as a common joining process. Each host owning the new key is able to share it again.

Public key  of the new user is sent to the connected user who broadcasts it to

all other users; the session key  is delivered to joining user.



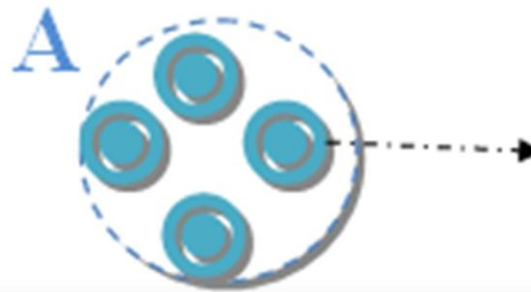
The new node joined the network and communicates with the others.



DESIGN CASE: LEAVING A NETWORK

□ Preface

*Decision making comes from human interaction.
A user decides to leave the network.*



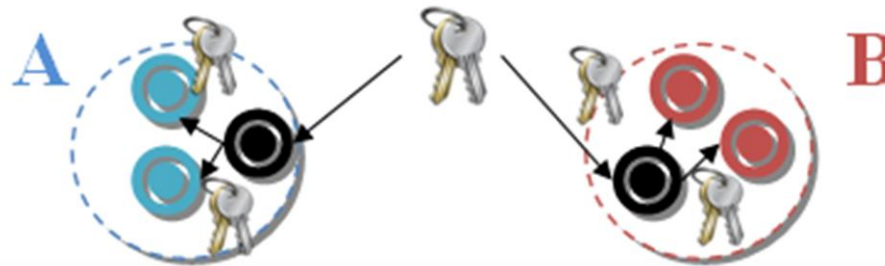
□ Initial phase

Idem that for the first design case: merging 2 networks

DESIGN CASE: LEAVING A NETWORK (CONT.)

□ Propagation phase

Each leader propagates the new key  to his/her pre-existing network nodes through the network itself using a signed message.



□ Communication phase

Nodes can now communicate once again, in a new secure network that the previous node cannot see anymore



COMPARISON TO GKA

□ Number of exchanged messages

	GKA	Skemasnet
Merging Networks	$2*(2N-1)$	$2(N-1) + (N-2)$
Leaving User	$2*(2N-1)$	$2(N-1) + (N-1)$

COMPARISON TO GKA

- Size of exchanged messages to deliver a new session key

GKA	$\text{SizeOfSessionKey} * 2 * N$
Skemasnet	$\text{SizeOfSessionKey} + \max(N1, N2) * \text{SizeOfPubKey}$

IMPLEMENTATION STARTING REVIEW

- ❑ We use *ns-2* (**N***etwork* **S***imulation version* **2**) in order to
 - ❑ **Implement** our *Skemasnet* algorithm
 - ❑ **Emulate** a bunch of **spontaneous networks**
 - ❑ **Simulate** a **merging** between 2 networks, the **joining** and the **leaving** of several nodes
 - ❑ **Compare** our implementation with **GKA** in term of **number & size of exchanging messages**
- ❑ We implemented *joining*, *leaving* and *merging* phase for the GKA protocol (without election phase)
- ❑ We are not implementing encryption algorithm: our concern is number and size of messages even if current implementation does not reflect this

TO DO LIST

1. Implement the election phase in GKA
2. Implement all the *Skemasnet* algorithm phases
3. Emulate a bunch of spontaneous networks
4. Simulate a merging between 2 networks, the joining and the leaving of several nodes
5. Compare our implementation with GKA in term of number & size of exchanging messages

Thank you for your attention !

Any question ?



THE SKEMASNET PROJECT

SESSION KEY MANAGEMENT IN A SPONTANEOUS NETWORK

HGP Team

Hyunho Park , Gianni M. Ricciardi, Pierre Alauzet

REFERENCES

- [1] Johann Van Der Merwe, Dawoud Dawoud, and Stephen McDonald
A Survey on Peer-to-Peer Key Management for Mobile Ad Hoc Networks
- [2] Dirk Balfanz, D. K. Smetters, Paul Stewart and H. Chi Wong
Talking To Strangers: Authentication in Ad-Hoc Wireless Networks
- [3] Michael Steiner, Gene Tsudik, Michael Waidner
Diffie-Hellman Key Distribution Extended to Group Communication
- [4] Daniel Augot, Raghav Bhaskar, Valerie Issarny and Daniele Sacchetti
An Efficient Group Key Agreement Protocol for Ad hoc Networks
- [5] <http://www.isi.edu/nsnam/ns/>
Official website of ns-2 simulator, accessed on November 29th 2009