Key Management Techniques for Group Communications in MANETs

Maria Striki, John S. Baras

Problem Specification

▪**Objective:** Construct **Secure**, **Efficient, Robust** & **Scalable** Key Management Schemes in MANETs, for Secure Group Comm/tions

▪**Secure:** enables/protects data exchange (it reaches all intended recipients, and only they must "read" it)

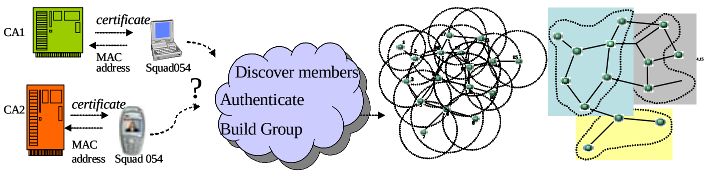
▪**Efficient:** overhead caused by security (**Comm/tion**, **Computation**, **Storage, Delay** Costs) low as possible

▪**Robust:** handles the dynamics of the network

▪**Scalable:** handles successfully a large number of nodes

Secure, efficient, scalable Group Communications for MANETs			Application: Secure Routing for MANETs		Application: Secure the WEP of MAC for MANETs
Key Management					
Bootstrapping	Key Generation	Entity Authentication	Key Distribution	Steady State Operations	Leader Election

Problem Specification: Roadmap for Group & Key Establishment & for Group Security Maintenance in a MANET



▪**Group Formation Operations:** Bootstrapping, Key Generation-Distribution, Entity Authentication

▪**Steady State Operations:** Re-Keying, Re-authentication, Privileges Update/ Revocation

Focus on the Design of Suitable for MANETs Techniques for Group Formation and Steady State Operations. Emphasis on Incorporation of Entity Authentication to Key Generation/ Distribution

Motivation

Example I (on-line Trusted Entities vs. off-line Trusted Entities)

Wire-line

▪ On-line and known Trusted Entities because:

- ▪Stable infrastructure
- ▪Infinite power

▪Effects:

- ▪Two unknown nodes can trust each other anytime through Trusted Entities

MANETs

▪Trusted Entities are not available at all times because:

- ▪Not stable infrastructure
- ▪Dynamic changes in the network

▪Effects

- ▪Unknown nodes do not have access on demand to Trusted Entities
- ▪Establishment of trust between them is a difficult task

Key Generation-Distribution Approach: Octopus Based Protocols

Original 2^d-Octopus (O): Breaks a Large Group into 2^d Smaller Subgroups, Requires Three Steps:

1. Subgroup Key Establishment by means of a Centralized Scheme
2. Group Key Establishment from Partial Subgroup Keys via Hypercube
3. Each Subgroup Leader Communicates Group Key to All its Members

▪ **GDH.2 based 2^d-Octopus (MO):** Maintains 2^{nd} Step Intact, Substitutes the Centralized Scheme of the 1^{st} & 3^d Step with GDH.2. The M_n Member of GDH.2 becomes Subgroup Leader for MO

Subgroup Key Becomes: $a^{ab\cdots z} = a^N$

▪ **TGDH based 2^d-Octopus (MOT):** Maintains 2^{nd} Step Intact, Substitutes the Centralized Scheme of the 1^{st} & 3^d Step with TGDH. The sponsor of TGDH becomes Subgroup Leader for MOT

Subgroup Key Becomes: $a^{xy} = a^N$

Advantages

▪**Flexibility:** Select Key Generation/Distribution Subgroup Protocols Freely

▪**Independence of Subgroups:** Different Key Generation Protocols May Be Applied to Each Subgroup

▪**Localization:** Subgroups Deployed in a Relatively Restricted Network Area

▪**Adaptability:** No Restrictions on Subgroup Size subject to Network Topology

▪**Efficiency:** Less Comm/tion OH & Bandwidth Consumption per Subgroup

▪**Robustness:** Faulty Subgroup Leaders Tolerated or Replaced Dynamically

Cost	2^d-Octopus (O)	Mod. 2^d-Octopus (GDH.2)- (MO)	Mod.2^d-Octopus (TGDH)-(MOT)
GSC Storage	$K(\lceil \frac{n-2^d}{2^d} \rceil + d)$ / $K(2\lceil \frac{n-2^d}{2^d} \rceil + d)$	$K(\lceil \frac{n-2^d}{2^d} \rceil + d)$	$(\lceil \frac{n}{2^d} \rceil + \log\lceil \frac{n}{2^d} \rceil + d)K$
Member Storage	$(2+d)K$	$(d+1)K$	$(\lceil \frac{n}{2^d} \rceil + \log\lceil \frac{n}{2^d} \rceil + d)K$
Initial GSC Computation	$(3\lceil \frac{n-2^d}{2^d} \rceil + 2d)C_E + (\lceil \frac{n-2^d}{2^d} \rceil)^{\#3} + 1.25(\lceil \frac{n-2^d}{2^d} \rceil)K^2 + \lceil \frac{n}{2^d} \rceil C_w$	$(\lceil \frac{n}{2^d} \rceil + 2d)C_E + \lceil \frac{n}{2^d} \rceil C_w$	$(2\log\lceil \frac{n}{2^d} \rceil + 2d)C_E + \lceil \frac{n}{2^d} \rceil C_w$ at max.
Initial Members Computation	$(d+2)C_E$	$((1/2)\lceil \frac{n}{2^d} \rceil + d)C_E$	$(2\log\lceil \frac{n}{2^d} \rceil + d)C_E$ at max
Initial Comm/tion	$(2n+(d-1)2^{d+1})K$	$(2^{d+1}(\lceil \frac{n}{2^d} \rceil^2 + 3\lceil \frac{n}{2^d} \rceil - 2) + 2^{d+1}d)K$	$(2^d 2\lceil \frac{n}{2^d} \rceil + 2^{d+1}d)K$
Add GSC Computation	$(3\lceil \frac{n-2^d}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil + 4)C_E + 2C_w + 2(\lceil \frac{n-2^d}{2^d} \rceil - 1)K^2$, one $(2\lceil \frac{n-2^d}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil)C_E$ rest	$C_E(\lceil \frac{n+1}{2^d} \rceil + 1 + 2\lceil \frac{d+1}{2} \rceil) + C_w$, one $C_E(2\lceil \frac{d+1}{2} \rceil)$, rest	$C_E(2\log\lceil \frac{n+1}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil) + 2C_w$, one GSC $C_E(2\lceil \frac{d+1}{2} \rceil)$, rest
Add Members Computation	$4C_E$, two $(2+d)C_E$ max. $2C_E$, the rest dC_E max.	$3C_E$, one subgroup $(1+d)C_E$ max. $2C_E$, rest dC_E max.	$4C_E$, one member $(h+d)C_E$ max $2C_E$, rest dC_E max
Add Comm/tion	$(4+2(2^d-1)+(2^d-2)+2\lceil \frac{n-2^d}{2^d} \rceil)K$	$(2\lceil \frac{n+1}{2^d} \rceil + 2(2^d-1)+(2^d-1))K$	$(\log\lceil \frac{n+1}{2^d} \rceil + \lceil \frac{n+1}{2^d} \rceil + 2(2^d-1)+(2^d-1))K$.
Delete GSC Computation	$(3\lceil \frac{n-2^d}{2^d} \rceil + 2 + 2\lceil \frac{d+1}{2} \rceil)C_E + C_w + (2\lceil \frac{n-2^d}{2^d} \rceil - 5)K^2$, one $(2\lceil \frac{n-2^d}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil)$ rest	$C_E(\lceil \frac{n-1}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil) + C_w$, one $C_E(2\lceil \frac{d+1}{2} \rceil)$, rest	$C_E(2\log\lceil \frac{n-1}{2^d} \rceil + 2\lceil \frac{d+1}{2} \rceil) + C_w$, one $C_E(2\lceil \frac{d+1}{2} \rceil)$, rest
Del. Members Computation	$3C_E$, two $(1+d)C_E$ max. $2C_E$, the rest dC_E max.	$3C_E$, one subgroup or $(1+d)C_E$ max. $2C_E$, rest dC_E max.	$4C_E$, one member $(h+d)C_E$ max $2C_E$, rest dC_E max
Delete Comm/tion	$(2+2(2^d-1)+(2^d-2)+2\lceil \frac{n-2^d}{2^d} \rceil)K$	$((\lceil \frac{n-1}{2^d} \rceil - 1)+2(2^d-1)+(2^d+1))K$	$(\log\lceil \frac{n-1}{2^d} \rceil + 2(2^d-1)+(2^d-1))K$

Table 1: Evaluation and Comparison of Octopus based protocols: (O), (MO) and (MOT)

Entity Authentication Approaches for MANETs

▪**Modification of Lamport Hash for Entity Authentication/Re-Authentication**

Original Lamport: Simple Password Protocol for Wire-Line Network, Prevents Eavesdropping, Impersonation, Replay Attacks. No Mutual Authentication

Our Modification: Suitable in MANETs, Eliminate Man-In-the-Middle Attack:

STEP 1: Authenticated DH key Exchange at Bootstrapping, Obtain Secret Key, use it to Exchange Initial Quantities <*n,xn*>.

STEP K (>1): Apply Original *Lamport*

▪**Modified Merkle Trees (MMT) for (Re)Authentication, Privileges Update/ Revocation**

Assumption: Hierarchical Structure. Group Divided to Subgroups. MMT Applied per Subgroup and a per Subgroup CA Generated

How it works: Member *i* Creates Secret Share m_i from 2-Party DH Exchange with Leader. Leader Hashes these Values, and Sends Back to Node *logN* Values

Results

Step I: Comparative Performance Evaluation of Key Generation for protocols OFT, MO, MOT & original Octopus

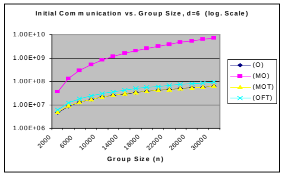


Figure 1: MO is by far the worst. MOT along with (O) are the winners, followed closely by OFT

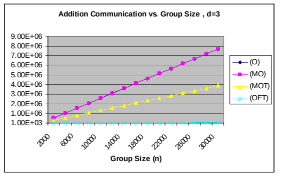


Figure 2: MO & (O) are by far the worst. MOT reduces overhead to almost half in regard to OFT

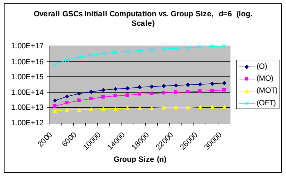


Figure 3: OFT is by far the worst. MOT has the best performance as far as Initial Comm/tion, then MO

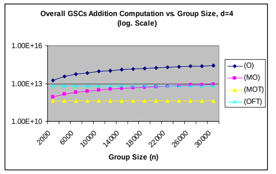
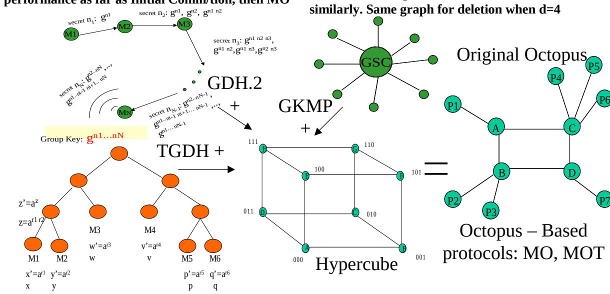


Figure 4: (O) is by far the worst. MOT is the winner for addition computation, MO and OFT behave similarly. Same graph for deletion when d=4



Step II: Comparative Performance Evaluation of Key Generation with Entity Authentication incorporation for protocols OFT, MO, MOT & original Octopus

Figure1: Initial Communication vs. Group size, d=6 for authenticated O and MOT, compared to OFT. O and MOT achieve almost the same lowest cost

Figure2: GSC comp. cost of authenticated (O), MOT, O. Despite the two added services, MOT still performs best

CONCLUSIONS – FUTURE WORK

▪Continue Design & Develop Efficient, Robust & Scalable Algorithms for Key Generation-Distribution, Entity Authentication, Steady State Operations of Key Management (KM)

▪Develop Analytical Mobility Models from Realistic Mobility Patterns, Incorporate them in KM, Evaluate Effect of Mobility on KM Schemes

▪Implementation of KM in Linux Platforms, to be Tested Over AODV