Automatic Node Configuration Protocol Using Modified CGA in Hierarchical MANETs

Hyewon K. Lee

School of Computing, Soongsil University, Seoul, Korea
kerenlee@nate.com

Abstract. The CGA is designed to prevent address spoofing and stealing and to provide digital signature to users without any help from security infrastructures, but fake key generation and address collision appear in flat-tiered network. To solve these problems, CGA defines security parameter (SEC), which is set to high value when high security is required. Although CGA with high SEC makes attackers be difficult to find fake key, it brings an alarming increase in processing time to generate CGA. On the contrary, the probability to find a fake key is high if low SEC is applied to CGA. In this paper, MCGA applicable to as well public networks as ad-hoc networks is proposed. Address collision problems are settled by employing hierarchy. Using MCGA, no previous setup is required before communication, and automatic node configuration is feasible.

1 Introduction

Mobile ad-hoc network (MANET) is a multi-hop wireless network without any prepared base station. It is capable of building a mobile network automatically without any help from DHCP servers or routers. Routing protocols to find shortest or optimistic route have been proposed, but these assume that nodes have been pre-configured. MANETconf [1], node configuration protocol [2] and prophet address allocation [3] have been proposed; however, these do not consider how to generate address.

CGA is designed to solve address spoofing and stealing attacks in IPv6. CGA offers digital signature without any help from CA, which is proper to ad-hoc nodes that have low processing power and memory capacity. However, fake key generation and address collision appear in flat-tiered network due to 64-bit-taken operation. To solve it, CGA defines SEC and allows a node to generate address only when the specific condition is satisfied. When SEC is set to high value, it becomes difficult for attackers to find fake key, but processing time to generate CGA increases incredibly.

2 Proposed Modified CGA (MCGA)

MCGA is composed of 64-bit subnet prefix learned from network and 64-bit interface identifier generated by individual node. For interface identifier, a random number or NIC address may be used. For subnet prefix, local-scoped prefix, FE80::/64 is used. When a node moves and gets different kinds of IEEE 802.11 beacon message, then it may accept new subnet prefix. Address format is similar to CGA except the SEC field.

MCGA generation process is detailed as follows: build key pair using RSA algorithm. Generate random number for MODIFIER, and set collision count to 0. Concatenate MODIFIER, collision count and user's public key. Put the concatenation into MD5. Take the first 64 bits from 128 bits key value generated by MD5, and set interface identifier to them. Set the u and g bits of interface identifier to 0. Concatenate subnet prefix and interface identifier, and put the concatenation into MCGA. If DAD is done successfully, allocate the MCGA to interface. If the check goes wrong and collision count is equal to 3, build new MODIFIER. Else, add 1 to collision count and build new concatenation.

Once a node enters into network, it generates MCGA and will request duplication check to the nearest agent, which lookups its resource table and gives appropriate answer to the requester. If requested address is not registered, the agent will give positive answer to the requester, and vice versa [2]. If no duplication is found, the agent will give final answer to the requester. When there are n nodes in a network, the probability that at least 1 fail occurs in n generations can be expressed as (1). In hierarchical network, a node in logically higher position holds information about all address resources in network, so duplication check between two nodes in the different logical positions seems to be enough. In this paper, opti-DAD [5] is employed for DAD. A node is able to initiate communication with others using on-pending address, which reduces delay due to long DAD process. Even though duplication ratio for address generation is very low, unallocated address may go to on-pending state concurrently by different nodes, and priority from arbitrary contention algorithm can be used.

$$1 - \frac{2^{64} P_n}{(2^{64})^n} \quad (1)$$

If there are a hash function ($h(\cdot)$) and two different inputs (m_1, m_2), $h(m_1) \neq h(m_2)$ is true. Picking specific part from hash's output, though, provokes collision. When we think of CGA, 2^{96} cases are mapped to one 64-bit identifier, mathematically. The probability for collision is proved by 'birthday problem,' and it becomes 0.63. Attackers are easily able to build fake key pair corresponding to origin key pairs by brute-force. Once fake key pair is found, an origin node encounters with address spoofing or stealing attack. For example, User_A builds its key pair and generates its CGA_A. If User_B finds a fake key pair which yields CGA_A and begins to send a message to User_C. Once User_C receives the message, it will identify User_B as proper owner for CGA_A. Unless User_A's key is disclose, User_B cannot mimic User_A's signature nor can it decrypt any message from User_A, but nodes can be induced into wrong communication.

When MCGA is applied to hierarchical ad-hoc network, an agent holds information about all address resources in network. If a stranger sends any message with different parameters for registered address, intermediate agent will notice it and drop the message. Let's go back to above example. If User_A and User_C locate at the same MUnit, User_C notices that the message from User_B is strange and drop it. If User_A and User_C locate at different regions, any message from User_B will be dropped by any intermediate agent between User_B and User_C.

3 Performance Evaluation

Processing time for CGA is the sum of requisition time of proper MODIFIER, generation time of interface identifier and delay due to duplication check, as (2). Processing time for MCGA is the sum of generation time of random number for MODIFIER, generation time of interface identifier and delay due to duplication check, as (3).

$$L_{CGA} = \left\lfloor \frac{m}{2} + 1 \right\rfloor l_{MOD} + (m+1)(l_{SHA} + l_{DAD}) \quad (2)$$

$$L_{MCGA} = \left\lfloor \frac{m}{2} + 1 \right\rfloor l_{RV} + (m+1)(l_{MD5} + l_{DAD}) \quad (3)$$

$$2l_d(d + \frac{8s}{b}) \leq l_{DAD} \leq 2l_d(d + r + \frac{8s}{b}) \quad (4)$$

Fig. 1. Processing time for CGA and MCGA, respectively

Fig. 2. The number of exchanged message between nodes

From (1), the number of duplication during address generation is assumed as 0. Generation time of MODIFIER has higher value than random number ($l_{MOD} \gg l_{RV}$). It is known that to process SHA takes more time than MD5 ($l_{SHA} \gg l_{MD5}$). From convergence time from [6], DAD time (l_{DAD}) can be expressed as (4).

System for simulations has following resources; CPU Pentium 4.3GHz and Memory 1GB. For operating system, Linux is employed, especially Kernel 2.4. Fig. 1 shows variation on generation time of 3000 CGAs and 3000 MCGAs. SEC is set to 0, 1 and 2, respectively. The average execution time for MCGA is 4.77 μ s while the average execution time for CGA when SEC is set to 0 is 15.57 μ s. The unit, μ s, is very small, but address generation by ad-hoc nodes will need more time. For example, 400 MHz ad-hoc node will perform the process 10 times slower than the above

system. No address duplication is occurred in both CGA and MCGA generations. Propagation delay is not considered in this simulation. Fig. 1 clearly proves that processing time for CGA increases dramatically when SEC increases, and processing time is strongly affected by SEC. When SEC is set to 3, it requires more than 200 hours. CGA with larger than 3 seems to be inappropriate as well public network as ad-hoc network.

The next simulation starts with initial network configuration and finishes when the number of nodes becomes 100. Ad hoc nodes distributed in two dimensional region of size 200×200 , 400×400 and $600 \times 600 \text{m}^2$, respectively. The network is randomly generated with the constraint that the graph be fully connected. Each node randomly moves with 1~2 m/s to random directions, and it is equipped with a radio transceiver, which is capable of transmitting a signal from 80 m. Processing delay for transmitting a message is randomly chosen between 5 ms and 10 ms. Propagation delay is 500 ms. MCGA with link-local prefix is used for address. Each node is supplied by a battery with enough power to at least make it able to carry out a complete operation. As shown in Fig. 2, the number of exchanged packets for node configuration increases as the number of nodes increases. MANETConf requires more packets. The result from this simulation obviously shows that node configuration protocol offers scalability to large network. Network topology and size give an effect on the number of packets.

4 Conclusions

In this paper, MCGA which is proper to hierarchical ad-hoc network is proposed. The MCGA has shorter processing time than CGA and offers digital signature with no additional overheads. To solve fake key and collision problems, we adopt hierarchical network structure. The MCGA is applicable to as well public networks as ad-hoc network. Simulations show that processing time for MCGA is reduced down 3.3 times and 68,000 times, compared to CGA with SEC 0 and SEC 1, respectively. Further, CGA with SEC 3 is inappropriate for both ad-hoc and public networks.

References

1. Nesargi, S. and Prakash, R., "MANETconf: Configuration of Hosts in a Mobile ad Hoc Network," INFOCOM, IEEE, 2002
2. Lee, H. and Mun, Y., "Node Configuration Protocol based on Hierarchical Network Architecture for Mobile Ad-Hoc networks," LNCS 3090, 2004
3. Zhou, H., Ni, L. and Mutka, M., "Prophet Address Allocation for Large Scale MANET," INFOCOM, IEEE, 2003
4. Aura, T., "Cryptographically Generated Address," RFC 3972, IETF, 2005
5. Moore, N., "Optimistic DAD for IPv6," work in progress, IETF, 2004
6. Kulik, J., Heinzelman, W. and Balakrishnan, H., "Negotiation-Based Protocols for Dissemination of Information in Wireless Sensor Networks," 2002