Chapter 6

Review of Autoconfiguration for MANETs

Hongbo Zhou and Matt W. Mutka

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/52687

1. Introduction

A MANET is a temporary multi-hop wireless network composed of mobile nodes without an underlying infrastructure. The mobile nodes in a MANET are not connected to an access point to access the Internet (although in some special cases a MANET and the Internet may co-exist). Instead, their wireless network interface cards operate in ad-hoc mode. The nodes that are within the transmission range of each other can communicate directly. For the nodes that are out of the range, they have to resort to the nodes in between to relay the messages.

Due to the popularity of mobile devices and independence from the infrastructure, a MANET can find wide applications in temporary wireless networks in meeting rooms, airports, and stadiums. It is fast, convenient, and economical to set up a MANET in a battlefield and for search and rescue. A Vehicular Ad-hoc Network, an variation of MANET, connects the running cars and fixed traffic lights and other sensors, is vital to implementation of smart transportation.

Before the application of such an IP-based network, IP address assignment is one of the most important network configuration parameters for the mobile nodes. Without a valid unique IP address, a mobile node cannot participate in unicast communications. It can only receive and send broadcast messages, which consumes valuable bandwidth and power, and thus it is desirable to limit the duration and scope of broadcast communications in a MANET.

For a small-scale MANET or a closed MANET, it may be simple to assign IP addresses to mobiles node by hand. It is also possible to burn an IP address in the ROM of a mobile node to re-use it repeatedly. However, the procedure will become inefficient and even impractical for a large-scale system or an open system where different kinds of nodes (such as laptops, smartphones, tablets, PDAs, and specialized computers) are free to join and leave.

Automatic IP address allocation is far more difficult to implement in a MANET than a hard-wired network such as a local area network, due to instability of mobile nodes, multi-hop transmission of messages, openness of the system, and lack of infrastructure. Therefore, al-

though DHCP [1] or SAA [2] is popular for hardwired networks, they cannot be directly ported to a MANET. A distributed algorithm that adapts to node mobility and topology change is more desirable.

Autoconfiguration for MANETs brings other issues that need to addressed as well.

The first issue is the change of IP address during communication. This issue is rare for a fixed node in a hardwired network running DHCP service. Even after the lease expires, the client tends to receive the same IP address from the server. In a MANET with autoconfiguration, there may be address conflicts when two separate MANETs join together. As a result, some of the nodes need to relinquish the duplicate addresses, which will interrupt on-going communications. An IP address handoff mechanism is necessary to make the transition smooth.

Security of the autoconfiguration is another issue that needs to be considered. If malicious nodes may be present, they may launch attacks on the autoconfiguration scheme to make it fail. Thus, a secure autoconfiguration is necessary for applications that have strict requirement for security level.

Autoconfiguration impacts the design of the security framework in a MANET. Most security frameworks assume that the IP address is the ID of the node, which is associated with the node's security parameters such as its public key. In an autoconfiguration scheme, the IP address is generated dynamically, and it may need to change during the session, which exposes the vulnerabilities in some pre-existing security frameworks.

This chapter gives a comprehensive review of autoconfiguration and related issues. It is structured as follows. Section 2 gives a review of autoconfiguration schemes, which can be divided into three groups. Among others, our Prophet Address Allocation outperforms in terms of bandwidth and latency. Section 3 introduces the IP address handoff scheme, which maintains the routing fabrics and on-going communications if there is an address change. In Section 4, we present different attack patterns and improve Prophet Address Allocation to survive these attacks. Thus, unique IP address allocation can still be achieved in the presence of malicious nodes. In Section 5, the Sybil attack is demonstrated to defeat some security frameworks within the scenario of autoconfiguration, and thus a different security framework is desirable. To further solve this problem, a combination of secure autoconfiguration and security framework is described in Section 6. Section 7 concludes the chapter.

2. Autoconfiguration for MANETs

Because unique IP address allocation is the first step towards a functioning MANET, this section studies the issue of autoconfiguration itself.

2.1. Introduction

In an open MANET where mobile nodes are free to join and leave, there are three scenarios that are of concern:

a. In the simplest scenario in Fig. 1, the node joins the network and then participates in the communication with other nodes. Once it is done, it leaves the network forever;

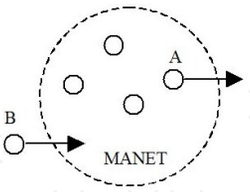

Figure 1. A node joins and leaves the MANET once

b. It is possible that some nodes move out of transmission range of other nodes. Thus, the network is divided into two or more partitions, as in Fig. 2 (a). If more nodes join different partitions, they may get duplicate addresses. When partitions merge later, address conflicts need to be resolved, as in Fig. 2 (b);

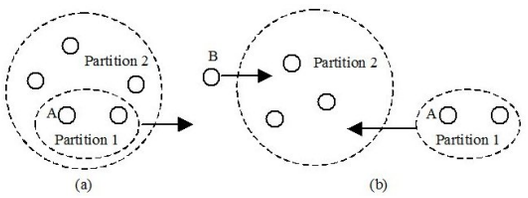

Figure 2. Network partitions and merges

c. In the third case, two or more independent MANETs merge. Because IP addresses are allocated independently, there may be duplicate addresses, which is similar to the second case.

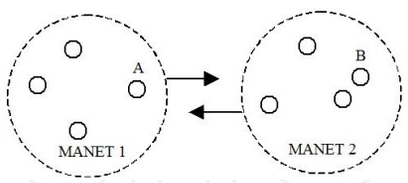

Figure 3. Merger of two independent MANETs

The autoconfiguration scheme needs to address all the three scenarios.

2.2. Related work

Several autoconfiguration schemes have been proposed. Depending on the IP address allocation state they have, they can be divided into the following three groups:

a. Stateless allocation

In the stateless allocation algorithm, no IP address allocation state is recorded. Instead, it utilizes Duplicate Address Detection (DAD) to determine a free IP address for a newcomer, as in [3]. The scheme reserved a pool of IP addresses for DAD communication only. The new node chooses one IP address from the pool to broadcast DAD messages, which contains another random IP address for actual data communications. The broadcast is utilized to see if the address is still available. If not, the new node will receive a veto message from some member and then choose another random address again. Otherwise, it repeats the DAD procedure for a few more times. Once it is certain that the address is free, it can proceed with the address for subsequent data communications.

The scheme in [3] did not address scenarios 2 and 3. A modified version, which is called weak DAD, was proposed in [4] to detect duplicate addresses on network merger. It favors proactive routing protocols and requires some changes to routing protocols.

b. Locally stateful allocation

The locally stateful allocation, such as one in [5] is based on a buddy system. Each allocator maintains a disjoint address pool. When a new node joins the network, the allocator divides the address pool into halves between itself and the new node. Thus, the address received from the allocator is guaranteed to be unique.

Although this algorithm can solve the scenarios 1 and 2 easily, it lacks the mechanism for scenario 3. Another problem is the reclamation of lost address pools. To solve this problem, allocators may need to exchange beacon messages frequently to track all the address pools and thus result in high communication overhead.

A similar idea was proposed in [6] that tries to deal with the network's partition and merger.

c. Globally stateful allocation

Distributed Dynamic Host Configuration Protocol (DDHCP) proposed in [7] maintains the global allocation state of IP addresses in all the members. When a new node joins, the allocator chooses a free IP address according to the allocation state. But the new node still needs to perform DAD to avoid the allocation of the same free IP address to two new nodes arriving simultaneously. It is also used to update the global allocation state in all the members.

DDHCP works well with proactive routing protocols. It also introduced network ID to detect merger of partitions or independent networks. It is generated from the node with the lowest IP address and is piggybacked in the periodic HELLO messages. Once network merger is detected, conflict detection and resolution will be initiated.

2.3. Prophet Address Allocation

We proposed Prophet Address Allocation (PAA) in [8] and [9], in which each node maintains a local allocation state. Unlike the disjoint address pools used in [5], the allocation state in PAA is an integer sequence. The integer sequence is generated by a stateful function with a seed. We deliberately designed the stateful function and update of seeds to satisfy the following two properties:

i. The interval between two occurrences of the same integer in a given sequence is extremely long;

ii. The probability of the same number in different sequences with different seeds in a given allocation session is extremely low.

The general procedures work as below:

iii. The first node chooses an initial state. Based on the initial state, it generates an integer as its own IP address. It updates the state;

iv. When another node joins the MANET, it receives the new state from the allocator, which is used to calculate its own IP address. The states in the allocator and the new node are updated simultaneously.

The design of the stateful function is based on the fundamental theory in arithmetic that every positive integer may be expressed uniquely as a product of prime numbers, apart from the arrangement of terms, as in the following formula:

$n=\prod_{i=1}^{k} pi^{ei}$, where the primes pi satisfy p1<p2< … <pk.

If each node has different tuples, they will have unique integers for their IP addresses. Thus, the most important issue is to generate different tuples during allocation. In our design, we chose the state to be (a, (p1, p2, …,pk), index), and the stateful function is:

$IPAddr = (a+\prod_{i=1}^{k} pi^{ei}) \% range + 1$

Parameter a in the state is used by the first node in each MANET to generate unique seed. The tuple is used to generate the product of prime numbers. Parameter index is the location of exponential value in the tuple that is to be increased by the allocator, while index itself is increased at the new node.

Based on the state and stateful function as above, the procedures of PAA can be illustrated with a 4-tuple:

a. The first node chooses a random number for a, an initial tuple of (0, 0, 0, 0) for p1 to p4, and 0 for index. It generates its own IP address, which is a + 1;

b. When the second node approaches the first node, the allocator updates its state to (a, (1,0, 0, 0), 0), and passes it to the new node. The new node increases index by 1. Thus, the new node's state is (a, (1,0, 0, 0), 1), and its address is a + 3;

c. If the third node approaches the first node, the first node updates its state to (a, (2,0, 0, 0), 0) and passes it to the new node. The third node gets the address of a + 5, and updates its own state to (a, (2,0, 0, 0), 1).

d. Similarly, when the fourth node approaches the second node, the allocator updates its state to (a, (1,1, …, 0), 1) and passes it to the third node. The new node increases index by 1. Thus, the third node's state is (a, (1,1, 0, 0), 2), and its address is a + 7.

In summary, all the four nodes will have different states and different IP addresses. They will generate different integers in a sequence in the subsequent allocations.

2.4. Protocol

The protocol of the PAA includes the following steps:

i. When a new node switches to ad-hoc mode, it starts to broadcast State Request Message periodically. Because each node in the MANET can act as an allocator, so only one-hop broadcast is necessary;

ii. If the new node does not receive any State Reply Message, it concludes that it is the first node in the MANET. Thus, it generates a random number for a and configures itself;

iii. If it receives a State Reply Message that contains the state, it applies the stateful function to configure itself, and update the state;

iv. During its session, if it receives a State Request Message from some other new node, it updates its state and includes it in the State Reply Message.

The above-mentioned procedures can solve Scenario 2 easily because even the network is partitioned, the addresses allocated in different parts will still be different. To handle Scenario 3, we borrowed the idea of Network ID (NID) from DDHCP. The first node also chooses a random NID and propagate it throughout the network during the allocation. If seed a and NID are also contained in the periodic HELLO messages, the nodes between two separate MANETs will detect the merger of two networks and initiate conflict resolution procedures. With the seeds, the node on the border can calculate two integer sequences and locate potentially duplicate addresses. The duplicate address list is then broadcast throughout the two MANETs. If a node happens to have that duplicate address, it changes its address accordingly.

2.5. Performance evaluation

Because every node can be an allocator in Prophet Address Allocation, only the communications between one-hop neighbors are necessary. Thus, PAA outperforms other schemes in the term of bandwidth and latency and is more suitable for a large-scale MANET.

We also ran simulations to demonstrate its superiority of PAA over stateless address allocation with ns-2. The simulations are run on ns-2.34 [10] with 50 nodes to 250 nodes. The random waypoint mobility model was used in the simulation [10]. After a node pauses for several seconds, a random destination point is chosen. The maximum speed is set to 5 m/s, which is repeated until the end of simulation. The pause time is 10 s for 50 nodes and 100 nodes, 20 s for 150, 200, and 250 nodes. These nodes join the MANET every 30 s (for 50, 100, and 150 nodes), or every 10 s (for 200 and 250 nodes). Different area sizes are also introduced to show the effect of node density on the algorithm. For example, scenario files of 800×800, 1000×1000, and 1200×1200 are utilized for 100 and 150 nodes, while scenario files of

1000×1000, 1200×1200, and 1400×1400 are tested for 200 and 250 nodes. The final results are the average results obtained with all the area sizes.

Although we chose AODV as the ad hoc routing protocol during the simulation, both address allocation schemes use one-hop and multi-hop broadcast, respectively for control message exchanges.

Figure 4 shows the ratio of total messages collected in stateless address allocation and Prophet Address Allocation, in contrast with a linear line. It shows that the ratio increases linearly with the number of nodes. Because in stateful allocation, each node is going to receive one copy of the message from all its neighbors, while in PAA only the neighbors of the new node receive a copy, so the communication overhead is almost constant regardless of the number of nodes in the MANET.

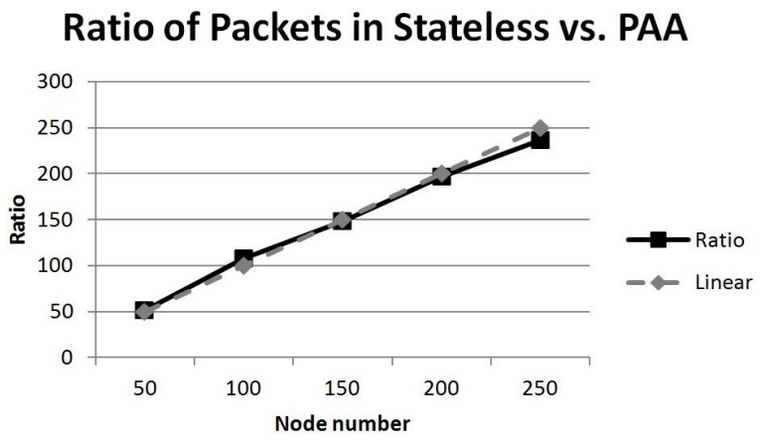

Figure 4. Ratio of communication overhead of Stateless to PA

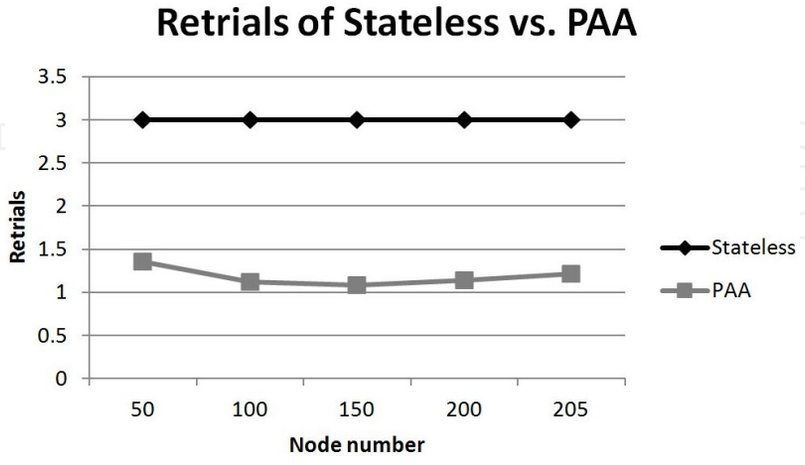

Figure 5. Latency for different node numbers

Figure 5 shows the average number of retrials for all the nodes in the MANET. For stateless allocation, each node tries for a constant three times. For APP, except for the first node that

tries for three times, all the nodes try for infinite time. However, the simulation results show that most of them get reply messages within 2 rounds.

3. IP address handoff in MANETs

With autoconfiguration implemented in MANETs, the IP address of a mobile node is not fixed any more. If a node changes its IP address, on-going communications will be interrupted, and routing fabrics will be broken. Thus, we need a handoff scheme to address these issues.

3.1. Introduction

Due to the node mobility and topology change, the following scenarios may lead to the necessity of IP address change in some mobile nodes:

a. A MANET is divided into two or more partitions. Some new members join different partitions and duplicate addresses may be allocated. Once the partitions merge, these nodes need to change their IP addresses;

b. Two separate MANETs merge together. Because address allocation is independent in each network, there could be duplicate addresses;

c. In case where hierarchical addressing scheme is applied [11], a node needs to change its address when roaming from one subnet to another, as in Fig 6;

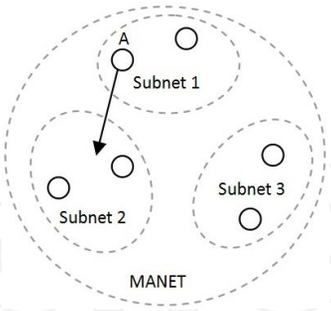

Figure 6. A MANET with hierarchical addressing

d. A MANET can get connected to the Internet if a hardwired node that also has a wireless network interface card working in ad-hoc mode, as in Fig. 7. That node would behave like a bridge between the hardwired network and the MANET. If the MANET and the hardwired network use the same private address range, there could be duplicate addresses.

When a node (say node A) changes its IP address, its current data communication will be interrupted. Even the other party (say node B) may initiate to rebuild the connection, it will

not find node A because that IP address does not exist anymore, unless the DNS scheme proposed in [12] is combined with the reactive routing protocol.

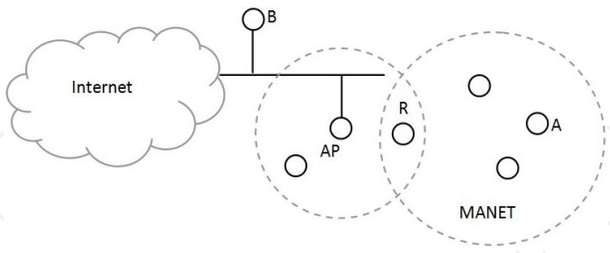

Figure 7. A MANET is connected to a hardwired network

Another issue with IP address change is the routing fabrics will be broken. Although the effect is not immediate since the forwarding is based on next hop's MAC address. The ARP entry will gradually time out and its upstream nodes cannot reach it anymore. Thus, local routing repair mechanism or route rediscovery procedures needs to be initiated eventually.

IP address change will also pose the threat to privacy. Suppose node A is talking with node B. On merger of two MANETs, node A needs to change its address because it has the same address of node C. Without notification to node B, the messages from node B will be routed to node C.

3.2. Related work

Mobile IP supports host mobility among LANs [13]. The host is assigned with a permanent home address. Once it connects a foreign network and gets a temporary care-of address, it registers the care-of address at its home agent. A tunnel is then built to forward messages between its home agent and foreign agent. Thus, any messages that destined to its home address will be forwarded to its current care-of address. Because a MANET is different from a LAN, so it is not working for address handoff.

The tunneling scheme was proposed in [14] that aimed to maintain on-going communications after address change in MANET. Once a node changes its address, it sends an Address Error (AERR) to the other party and an IP-in-IP tunnel is created between them. The outer IP header contains the new address while the inner IP header uses the old address. This solution can preserver communication states. However, it ignores the overhead cause by routing repair or route rediscovery. Besides, it introduced Denial of Service issue, as illustrated in Fig. 8. Suppose node A changes its address from xto y because the another node (node C) has the same address of x. A tunnel is created between nodes A and B that forwards all the packets destined to IP address x to node A. Thus, node B cannot communicate with node C anymore.

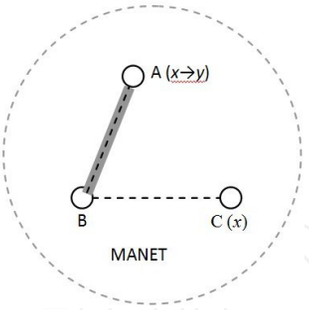

Figure 8. Denial of Service issue in tunneling scheme

3.3. IP address handoff scheme

We proposed IP address handoff scheme in [15] to maintain routing fabrics and keep ongoing communications.

Firstly, because the node is aware of the IP address change, we can require that node broadcast a Route Shift message that contains its old and new IP addresses to its one-hop neighbors. To prevent IP spoofing attacks, we can also require the message to be signed with its private key. A lightweight solution is that the node chooses a random number and attaches the hash value of the random number in Route Request message, Route Reply message, and periodic HELLO messages. In the Route Shift message, the random value is included to verify the identity of the origin.

Secondly, a NAT-based solution is utilized to maintain communication states, in which the old address is changed to new address for outgoing packets while the new address is changed back for incoming packets. NAT needs to be performed at both ends, as below:

a. On address change, node A as in Fig 7 creates a NAT table (such as Table 1) that maps the old address to the new address according to the protocol and source port number of outgoing TCP/UDP packets if the packet still contains the old address.

Old Address	New Address	Protocol	Port Number
x	y	TCP	5472
x	y	…	…

Table 1. NAT table at node A

According to the NAT table, the outgoing packet's old source address x is modified to y, and the checksums need to be re-calculated. For incoming packets, if its protocol and destination port number match an entry, the destination address is changed back to x together with the checksums updated.

b. Node A sends an Address Change message to node B so node B can create its NAT table. The control message contains the old address, new address, node A's port number,

node B's port number, and sequence number of message from A to B. Based on these data, node B can insert an entry to its NAT table, such as table 2:

Old Address	New Address	Protocol	Remote Port Number	Local Port Number	Seq Number
x	y	TCP	5472	80	626473
x	y	…	…	…	…

Table 2. NAT table at node B

If an incoming packet matches an entry in the NAT table, its new address y is changed to x. At the same time, the Seq Number field is increased by the payload length. For an outgoing packet with the destination address x, if it matches an entry, its destination address will be set to y. The NAT table at node B includes both port numbers and sequence number from node A, so the packets destined to node A can be distinguished from those to node C. To prevent IP spoofing attacks, the Address Change message is required to include node A's digital signature.

3.4. Prototype implementation

A prototype of IP address handoff scheme was implemented on Fedora Linux 12 with netfilter [16] to test the preservation of communication states in a LAN, as illustrated in Fig. 9. After a TCP connection is created between a laptop client and a server, the client changes its address from 192.168.1.155 to 192.168.1.140.

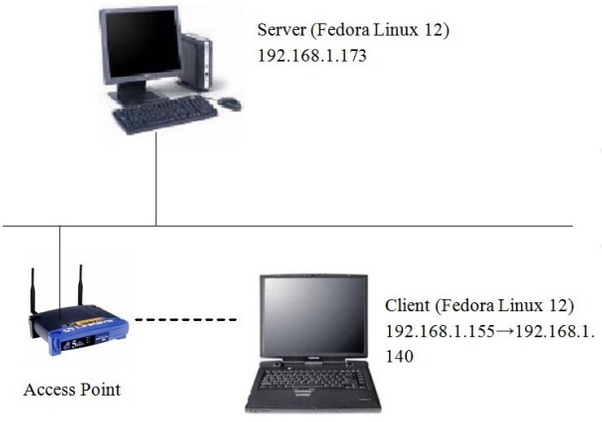

Figure 9. Thetestbed of the prototype

The outgoing packets are handled in NF_IP_LOCAL_OUT hook, while the incoming packets are modified in NF_IP_PRE_ROUTING hook. The code to modify outgoing packets is the client is illustrated in Table 3. The processing of incoming packets is similar. In the pro-

totype, the addresses are hardcoded. But in real application, NAT tables mentioned in the previous section should be utilized.

```
#define OLD_ADDRESS 0xC0A8019B // 192.168.1.155
#define NEW_ADDRESS 0xC0A8018C // 192.168.1.140
static unsigned int handoff_NAT_out(unsigned int hook, struct sk_buff **pskb, const struct net_device *indev,
const struct net_device *outdev, int (*okfn)(struct sk_buff *)
{
struct tcphdr* th;
// Whether we should perform NAT or not
if ((*pskb)-"/>nh.iph-"/>saddr == htonl(OLD_ADDRESS) && (*pskb)-"/>nh.iph-"/>protocol == 6)
{
th = (struct tcphdr*)((char*)(*pskb)-"/>nh.iph + (*pskb)-"/>nh.iph-"/>ihl*4);
if (th-"/>dest == htons(10000))
{
// Change (source) address
(*pskb)-"/>nh.iph-"/>saddr = (DWORD)htonl(NEW_ADDRESS);
// Recompute IP checksum
(*pskb)-"/>nh.iph-"/>check = 0;
(*pskb)-"/>nh.iph-"/>check = in_checksum((WORD*)((*pskb)-"/>nh.iph), (*pskb)-"/>nh.iph-"/>ihl*4);
}
}
return NF_ACCEPT;
}
```

Table 3. NAT processing of outgoing packets at the client

We ran Wireshark at both the server and client to capture all the TCP packets and verified that the session continues without being aware of the address change.

4. Secure address allocation for MANETs

All the autoconfiguration schemes introduced in Section 2 assume that every node in the MANET is trustworthy and have no security mechanism. Thus, if there is a malicious node in the network, the autoconfiguration scheme may fail: either no new node will be allowed to join the network, or there will be duplicate addresses. This section focuses on a secure autoconfiguration scheme.

4.1. Attacks on autoconfiguration

There are several common kinds of attacks that target at autoconfiguration schemes:

a. IP spoofing attack

IP spoofing attack means the malicious node impersonates as another node. In stateless autoconfiguration scheme, the malicious node masquerades as a node with the same IP address as the new node chooses. For each request message from the new node, it replies with a veto message to deter the new node from joining the network. In Prophet Address Allocation, the malicious node uses another member's IP address and state in allocation, and thus duplicate addresses will be assigned.

IP spoofing attack is extremely difficult to detect and prevent, even with the deployment of Certificate Authority, because the IP address of CA itself needs to be obtained with autoconfiguration.

b. State pollution attack

In stateful allocation schemes, the malicious node passes incorrect or forged state in the reply message to the new node, thus duplicate addresses will be allocated.

c. Sybil attack

In Sybil attack [17], a malicious node impersonate several non-existent nodes at the same time. Thus, these nodes will seem like a group and can undermine the network service that requires cooperation among all or most nodes. In stateful allocation scheme, a malicious node can initiate a Sybil attack and convince the new node of incorrect parameters.

4.2. Secure prophet address allocation

In the original Prophet Address Allocation, the new node does not verify the parameters in the reply message, which leaves the exposure to IP spoofing attacks and state pollution attacks. In Secure Prophet Address Allocation, we made the following improvements to guarantee a unique state for each address allocation:

a. More parameters are included in the reply message

Now the reply messages contains the following parameters: (1) The seed value of the MANET (a); (2) The index of increasing exponential (c); (3) The source address of the allocator (x); (4) The initial exponential array (i[1,2, …, n]); (5) Priority (p), which indicates the freshness of the parameters.

b. The new node verifies the parameters and chooses a seemingly correct one randomly

The relationship among these parameters is:

$$x = f\left(a, i\left[1,2,\ldots,n\right]\right) \tag{1}$$

which means the source address of the allocator should be calculated with the seed value and the initial exponential array.

The new node chooses the reply message randomly that confirms to the relationship and has the highest p value, chooses a random value (r), and generates its own address with the following equation:

$$y = f\left(a, e\left[1,2,\ldots,n\right]\right) \tag{2}$$

where

$$e\left[\ j\ \right] = \begin{cases} i[\ j\ ], & j < c \\ r + p, & j = c \\ i[\ j\ ] = 0, & j > c \end{cases}$$

c. A broadcast is included to force all the members to update its state

An ACK message is broadcast throughout the MANET to include p + r value, which will be the new priority value in each member's state.

Although all the parameters in a reply message could be forged, since the new node always chooses the highest priority value and its own random value, it is guaranteed that the state it receives would be unique at present. The subsequent ACK message forces all the members to update their states. Thus, the state that the new node just used will become obsolete and will not be used in the subsequent allocations.

4.3. Simulation

Simulations of Prophet Address Allocation and Secure Prophet Address Allocation in the are run with ns-2 (version 2.34). Malicious nodes are randomly chosen during the simulation. Statistics data about the number of duplicate addresses are collected to show the invulnerability of the latter. Both allocation schemes are tested with 50 nodes in the area of 800 × 800 with random waypoint mobility model. The size of exponential array is chosen to be 50. However, at most 6 exponentials are used. Table 4 shows that no duplicate addresses are generated in Secure Prophet Address Allocation.

Percentage of malicious nodes	IP spoofing attacks		State pollution attacks and Sybil attacks	
	PA	SPA	PA	SPA
10%	1	0	1	0
20%	2	0	2	0
25%	2	0	2	0
33%	5	0	5	0
50%	3	0	3	0

Table 4. The number of duplicate address in simulations

http://dx.doi.org/10.5772/52687

5. Autoconfiguration's impact on distributed certificate authority

Most prevalent Distributed Certificate Authority (DCA) schemes for MANET are based on threshold cryptography. However, in the context of autoconfiguration, these DCA schemes will fail in the presence of Sybil attacks. This section introduces a different DCA scheme.

5.1. Threshold cryptography-based DCA

Threshold cryptography-based DCA scheme [20] [21] was originally proposed for hard-wired networks, in which the administrators of servers can verify others' identities and trust each other. In (k, n)-threshold cryptography, the public key of the DCA is known to all the users, while the secret key is divided into many secrete shares among n servers. When a user wants its message signed, the message is sent to all the DCA servers. Each server signs it with its secret share to generate a partial signature. With at least k partial signatures, a valid signature can be constructed that can be verified with the public key. Because these secret shares have no explicit relationship, and can be refreshed periodically without changing the public key, a malicious node has to gather at least k secret shares within some time interval to compromise the system, which makes the system very secure. Besides, the system can tolerate the loss of n - k nodes, it seems ideal for a MANET where nodes may leave or power down unexpectedly [22][23].

5.2. Vulnerability of threshold cryptography-based DCA

In Certificate Authority, the IP address of the node is regarded as the identity of the node. However, in the presence of an autoconfiguration scheme, the identity is generated dynamically. With Sybil attacks from a malicious node, threshold cryptography-based DCA will be compromised, as illustrated below:

Suppose node M is a malicious node in the MANET. Because the IP addresses are assigned with autoconfiguration, node M may request or generate multiple identities in advance. During the formation of a DCA server group based on (k, n)-threshold cryptography, it uses k identities to join the DCA server group with other good nodes, such as nodes A, B, and C. Thus, it has enough secret shares to generate a valid signature for any kind of its own messages.

5.3. Multiple-key cryptography-based DCA

To co-exist with an autoconfiguration scheme, we proposed multiple-key cryptography-based DCA scheme (MC-DCA) in [24]. The idea of multiple-key cryptography was first presented in [25], which is a variation of public-key cryptography. In traditional public-key cryptography, there are only two keys, one of which is the public key and the other is the private key. The message encrypted or signed with one key can be decrypted/verified with the other. In multiple-key cryptography, there are multiple keys. We can choose any subset of the keys to be public keys and the complementary subset will be private keys. The mes-

sage encrypted/signed with one subset of keys needs to be decrypted/verified with all the keys in the other subset.

MC-DCA scheme is based on a distributed algorithm to generate secret shares and the public key [26]. Suppose there are n servers in the DCA server group. Firstly, all the servers agree on three parameters: two large prime numbers p and q such that q divides p-1, and g that is a generator of Gq (Gq is the unique subgroup of Z*p of order q). These three parameters are a part of the public key and should be known to all the other nodes. Each server (say server i) chooses its secret share xi, and computes the public part of hi = gxi. The sum of xi is the private key, while the product of hi is the public key. The following steps in [26] are used for threshold cryptography and unrelated to MC-DCA scheme.

The protocol of MC-DCA works as follows:

i. When a client needs DCA service, it broadcasts an INVITE message throughout the MANET to initiate the invitation procedure;

ii. On receipt of INVITE message, each node decides if it wants to participate in DCA service. If it wants to join, it broadcasts a PARTCP message with its public key;

iii. All the server nodes agree on the parameters p, q, and g;

iv. Each server node chooses its secret share independently and calculates the corresponding public part. One server node collects all the public part and announces the public key;

v. The client sends its IP address, public key, and other related information encrypted with server node own public key in a REQUEST message;

vi. The server calculates its partial signature for the message, and signs the partial signature with its private key;

vii. The client verifies the signature with the public key after combining all the partial signatures.

If a server node leaves the MANET abruptly, the client can get all the partial signatures. In this case, new nodes would be invited to join the DCA server group. They will choose different secret shares and result in a different public key. Thus, we need to associate a version number with the public key. The client needs to store the all the public keys. We can also require that the certificate be renewed once the version number is increased by a threshold value to remove old public keys.

6. Secure autoconfiguration and public-key distribution

Section 4 and 5 addressed secure autoconfiguration and distributed certificate authority for MANETs separately. This section intends to combine both secure autoconfiguration and public-key distribution when a new node joins the network, which may provide a bootstrapping procedures to build a distributed certificate authority.

6.1. Related work

A self-authentication scheme was proposed in [27] that is an application of Cryptographically Generated Address [28]. A new node generates its public/private key pair randomly and simply uses the hash value of its public key as its IP address. To avoid address conflict, a DAD procedure as described in Section 2 is used. This method is simple and elegant. To verify the ownership of the public key, the other node just needs to perform a hash function its public key. However, there are some problems with this scheme: (1) Only one pair of private/public keys are supported; (2) With the autoconfiguration, the IP address may need to change, which leads to the change of public/private key pair; (3) In case when a MANET is connected to the Internet, the private address of the node may need to be changed with NAT. Thus, there is no relationship with the public key and the public IP address.

The challenge-response scheme [29] is based on the buddy system in [6]. The difference is in the security mechanism in the former: the new node broadcasts its MAC address and public key to its one-hop neighbors for authentication. Because only one-hop neighbors get the public key, other nodes are still susceptible to the "man-in-the-middle" attack. Another issue is that the allocator itself could be a malicious node and allocate a non-disjoint address pool to the new node.

The trust model in [30] is based on MANETconf [7]. It assumes that the number of malicious nodes is small in the network. Each node maintains a trust value for its neighbors. Only the node whose trust value is greater than or equal to a threshold value is considered to be trustworthy. During autoconfiguration, the new node chooses a trustworthy node as an allocator. The allocator also ignores the veto messages from non-trustworthy nodes. However, this scheme is vulnerable to Sybil attacks.

Another trust model proposed in [31] is based on the buddy system [6] and threshold cryptography-based DCA [22]. It assumes that there is already a DCA in the MANET, so the messages can be authenticated. But as we already pointed out in [24], threshold cryptography-based DCA is vulnerable to Sybil attacks.

6.2. Secure autoconfiguration and public-key distribution

Because the identity of the mobile node is generated with the autoconfiguration, it is desirable that the new node's public key is distributed throughout the network at the same time to avoid the "man-in-the-middle" attack. Thus, we combined both in SA-PKD scheme proposed in [32].

The procedures work as follows:

i. The new node, node N, generates its own public/ private key pair (PbN/PrN) and a random number RN;

ii. The new node applies a hash function on the random number RN and gets its address AddrN = Hash(RN);

iii. Node N performs DAD for a few times with a temporary address, as described in [3]. In the DAD message, it puts the hash value of its IP address Hash(AddrN) and the address signed with its private key SignN(AddrN);

iv. On receipt of the DAD message, each node performs the same hash function on its own address. If the result is the same as the hash value in the DAD message, there may be a potential address conflict, so it sends back a NACK message to veto it;

v. If node N does not receive any NACK message, it broadcasts a CMT message to commit the autoconfiguration, in which it puts its public key and uses the address it chooses. On receipt of the CMT message, each node can verify the hash value of the address and the association between the IP address and the public key.

During the broadcast of DAD message, each node gets a copy of the hash value of the address and signature, which can only be verified with the parameters in the subsequent CMT message. If a malicious node uses a random string to replace the parameters in the DAD message, the receiver cannot recover the address. If a malicious node chooses another pair of public/private keys and a random address to replace both parameters in DAD message and CMT message, the receiver will get two associations, including the original one from the new node and a new association of another public key to a different address. In either case, the receiver will get the correct association of PbN to AddrN.

6.3. Simulation

We ran simulation of SA-PKD on ns-2 (version 2.34) with 50 nodes. The random waypoint mobility model is used, in which the nodes are constantly moving within the simulation area. The maximum speed is 20 m/s, and the minimum speed is 5 m/s. The pause time is 0 second. Once the simulation starts, each node joins the MANET every 10 seconds. It broadcasts DAD message and CMT message every 3.0 seconds for 3 times.

We used the MD5 algorithm in [33] for the hash function and a simplified RSA algorithm for signing/verification. We implemented application-level broadcasting, thus there is no preference for the routing protocol.

Except for the first three nodes, we chose malicious nodes randomly. The percentage of malicious nodes is 2%, 4%, 8%, and 10% for different simulations. We let each node print some debug information such as its IP address and public key, and the associations of IP addresses to public keys it received from other nodes. According to the simulation results, all the members can get the associations of new nodes correctly.

7. Conclusion

In this chapter, we gave a comprehensive review of the innovative solutions proposed by the network research community to solve the problems associated with the autoconfiguration. IP address assignment is so important for a node to participate in unicast communications, it is

worth our research effort. Unlike a hardwired network, there is no fixed infrastructure in the MANET. Due to node mobility and instability, the network topology keeps changing. In an open system, all kinds of mobile nodes ranging from powerful laptops to energy-efficient sensor nodes may join as long as they confirm to the wireless communication standards. Subsequently, the design of the protocols and algorithms is more complicated.

Autoconfiguration brings a lot of related issues that are trivial to solve or even unseen in a hardwired network. For example, a mobile node may need to change its IP address during the communication, which will break routing protocols and interrupt on-going communications. Security on autoconfiguration and related issues is another important factor to the successful application of the MANET. In Sybil attacks, a malicious node can forge many non-existent fake identities to appear as a group, which can knock out the seemingly robust threshold cryptography-based DCA. All these issues are challenging, and we tried to provide a satisfactory solution that is supported by theoretical analysis and simulation results. They are summarized below:

i. An efficient autoconfiguration solution was proposed for a large-scale MANET;

ii. An IP address handoff scheme aims to reduce the communication overhead caused by address change;

iii. A secure autoconfiguration can withstand several common forms of attacks;

iv. A multiple-key cryptography-based DCA may replace threshold cryptography-based DCA;

v. SA-MKD scheme combines secure autoconfiguration and public-key distribution when a new node joins the MANET.

There is still much work awaiting us. We will continue to investigate more attack patterns on autoconfiguration scheme, study the application of MC-DCA to a large scale network, and explore the possibility of combination of MC-DCA and SA-MKD.

Author details

Hongbo Zhou[1] and Matt W. Mutka[2]

1 Dept. of Computer Science, Slippery Rock University, USA

2 Dept. of Computer Science & Engineering, Michigan State University, USA

References

[1] R. Droms, “Dynamic Host Configuration Protocol,” Network Working Group RFC 2131, March 1997

[2] S. Thomson and T. Narten, “IPv6 Stateless Address Autoconfiguration,” Network Working Group RFC 2462, December 1998

[3] C. Perkins, J. Malinen, R. Wakikawa, E. Belding-Royer, and Y. Sun, “IP Address Autoconfiguration for Ad Hoc Networks,” draft-ietf-manet-autoconf-01.txt, November 2001

[4] N. Vaidya, “Weak Duplicate Address Detection in Mobile Ad Hoc Networks,” In Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc’02), Lausanne, Switzerland, June 2002

[5] A.Misra, S. Das, A.McAuley, and S. K. Das, “Autoconfiguration, Registration, and Mobility Management for Pervasive Computing,” IEEE Personal Communication, August 2001, pp 24-31

[6] M. Mohsin and R. Prakash, “IP Address Assignment in a Mobile Ad Hoc Network,” In Proceedings of MILCOM 2002, Anaheim, CA, October 2002

[7] S. Nesargi and R. Prakash, “MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network,” In Proceedings of the 21st Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM 2002), New York, NY, June 2002

[8] H. Zhou, L. M. Ni, and M. W. Mutka, "Prophet Address Allocation for Large Scale MANETs," In Proceedings of the 22nd Annual Joint Conference of IEEE Computer and Communication Societies (INFOCOM 2003), San Francisco, CA, April 2003

[9] H. Zhou, L. M. Ni, and M. W. Mutka, "Prophet Address Allocation for Large Scale MANETs," Elsevier Ad Hoc Networks Journal, Vol. 1, Issue 4, pp 423-434, November 2003

[10] ns-2 wiki, http://nsnam.isi.edu/nsnam/index.php/Main_Page

[11] G. Pei and M. Gerla, “Mobility management for hierarchical wireless networks,” Mobile Networks and Application (MONET), Vol. 6, No. 4, pp 331-337, August 2001

[12] P. Engelstad and G. Egeland, “Name resolution in on-demand MANETS and external IP networks,” draft-engelstad-manet-name-resoltuion-00.txt, February 2003

[13] C. Perkins (editor), “IP mobility support,” Network Working Group RFC 2002, October 1996

[14] J.-H. Jeong, H.-W. Cha, J.-S. Park, and H.-J. Kim, “Ad hoc IP address autoconfiguration,” draft-jeong-adhoc-ip-addr-autoconf-00.txt, May 2003

[15] H. Zhou, M. W. Mutka, and L. M. Ni, "IP Address Handoff in the MANET," In Proceedings of the 23rd Conference of IEEE Communication Society (INFOCOM 2004), Hong Kong, China, March 2004

[16] The netfilter.org project, http://www.netfilter.org/

[17] J. Couceru, "The sybil attack," In Proceedings of the 1st Workshop on Peer-to-Peer Systems (IPTPS'02), Cambridge, MA, March 2002

[18] H. Zhou, "Secure Prophet Address Allocation for Mobile Ad-hoc Networks," In Proceedings of IFIP International Workshop on Network and System Security (NSS 2008), Shanghai, China, October 2008

[19] H. Zhou, M. W. Mutka, and L. M. Ni, "Secure Prophet Address Allocation for MANETs," Wiley International Journal of Security and Communication Networks, Vol. 3, Issue 1, pp 31-34, January/February, 2010

[20] A. Shamir, "How to share a secret," Communications of ACM, Vol. 22, pp. 612-613, November 1979

[21] Y. Desmedt and Y. Frankel, "Threshold Cryptosystems," Proceedings of Advances in Cryptography (Crypto 89), Lecture Notes in Computer Science, Vol. 435, Springer-Verlag, pp. 307 - 315, 1989

[22] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," IEEE Network, Vol. 13, No. 6, pp. 24 - 30, November/December 1999

[23] M. Bechler. H.-J. Hof, D. Kraft, F. Pahlke, and L. Wolf, "A Cluster-based Security Architecture for Ad Hoc Neteworks," In Proceedings of the 23rd Conference IEEE Communication Society (INFOCOM 2004), Hong Kong, China, March 2004

[24] H. Zhou, M. W. Mutka, and L. M. Ni, "Multiple-key Cryptography-based Distributed Certificate Authority in Mobile Ad-hoc Networks," In Proceedings of IEEE Global Telecommunications Conference (GLOBECOM 2005), St. Louis, MO, November 2005

[25] C. Boyd, "Some Applications of Multiple Key Ciphers," In Proceedings of Advances in Cryptography (Eurocrypt'88), Lecture Notes in Computer Science, Springer-Verlag, pp. 455 - 467, 1988

[26] T. P. Pedersen, "A Threshold Cryptosystem without a Trusted Party," In Proceedings of Advances in Cryptography (Eurocrypt'91), Lecture Notes in Computer Science, Vol. 547, Springer-Verlag, pp. 522 - 526, 1991

[27] P. Wang,D. S. Reeves, and P. Ning, "Secure Address Autoconfiguration for Mobile Ad Hoc Networks," In Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2005), pp. 519 - 521, San Diego, CA July 2005

[28] T. Aura, "Cryptographically Generated Address (CGA)," Networking Group RFC 3972, March 2005

[29] A. Cavalli and J.-M. Orset, "Secure Hosts Authentication in Mobile Ad Hoc Networks," In Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW 2004), Tokyo, Japan, March 2004

[30] S. Hu and C. J. Mitchell, "Improving IP Address Autoconfiguration Security in MANETs Using Trust Modeling," In Proceedings of 1st International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2005), Wuhan, China, December 2005

[31] F. Buiati, R. Puttini, and R. D. Sousa, "A Secure Autoconfiguration Protocol for MANET Nodes," In Proceedings of the 3rd International Conference on Ad-hoc Networks and Wireless (ADHOC-NOW 2004), Vancouver, Canada, July 2004

[32] H. Zhou, M. W. Mutka, and L. M. Ni, "Secure Autoconfiguration and Public-key Distribution for MANETs," In Proceedings of 6th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2009), Macau SAR, China, October, 2009

[33] R. Rivest, "The MD5 Message-Digest Algorithm," Network Working Group RFC 1321, April 1992