논문번호 12-37A-09-07
한국통신학회논문지 '12-09 Vol.37A No.09
http://dx.doi.org/10.7840/kics.2012.37A.9.772

애드혹 네트워크 상에 트리구조 깊이를 이용한 다중홉 클러스터링 기반 TNA(Traceback against Network Attacks) 설계

김 주 용◆, 이 병 관°

A Design of TNA(Traceback against Network Attacks) Based on Multihop Clustering using the depth of Tree structure on Ad-hoc Networks

Ju-Yung Kim◆, Byung-Kwan Lee°

요 약

현재 MANET에서는 DOS나 DDOS 공격이 증가하고 있지만, MANET은 제한된 대역폭, 계산 자원 및 배터리 전원을 가진 노드들로 구성된 네트워크이기 때문에 기존의 역추적 메커니즘을 적용할 수가 없다. 따라서 MANET에서 역추적 기법을 적용 시킬 시에는 각 노드가 가지는 자원을 효율적으로 사용해야한다. 그러나 기존의 애드혹 네트워크에 적용한 역추적 기법은 클러스터링 영역에서 각 노드를 대표하는 Cluster head가 역추적 정보를 관리하기 때문에 Cluster head의 과부하로 노드의 수명을 단축시키는 문제점을 가지게 된다. 게다가, 다중홉 클러스터링일 경우, 하나의 Cluster head가 더 많은 노드를 관리하기 때문에 문제는 더욱 심각해진다. 본 논문에서는 이러한 Cluster head의 오버헤드를 줄이기 위한, 역추적 정보를 관리하기 위하여 트리구조의 깊이를 이용한 TNA(Traceback against Network Attacks)를 제안한다.

Key Words : TNA, Multihop Clustering, MANET, Cluster head, Marking_ID, Marking table

ABSTRACT

In the current MANET, DOS or DDOS attacks are increasing, but as MANET has limited bandwidth, computational resources and battery power, the existing traceback mechanisms can not be applied to it. Therefore, in case of traceback techniques being applied to MANET, the resource of each node must be used efficiently. However, in the traceback techniques applied to an existing ad hoc network, as a cluster head which represents all nodes in the cluster area manages the traceback, the overhead of the cluster head shortens each node's life. In addition, in case of multi-hop clustering, as one Cluster head manages more node than one, its problem is getting even worse. This paper proposes TNA(Traceback against Network Attacks) based on multihop clustering using the depth of tree structure in order to reduce the overhead of distributed information management

◆ 주저자 : 관동대학교 전자계산공학과, jjeleun@naver.com, 정회원
° 교신저자 : 관동대학교 컴퓨터학과, bklee@kd.ac.kr, 정회원
논문번호 : KICS2012-07-317, 접수일자 : 2012년 7월 13일, 최종논문접수일자 : 2012년 9월 14일

Ⅰ. 서 론

최근 인터넷의 급속한 성장을 배경으로 모바일 애드혹 네트워크(MANET : mobile ad-hoc Network)는 초기의 원거리 망에서 근거리 지역망, 일반 대중의 가정 내 개인용 디바이스간 통신망 등으로 광범위하게 관련 연구 분야를 넓혀가고 있다[1,2]. 애드혹 네트워크와 IP네트워크는 동일한 보안 위협을 가지고 있으며 DOS나 DDOS 공격이 증가하고 있지만 제한적인 자원 등의 문제로 기존의 역추적 메커니즘을 적용할 수가 없다[3]. 현재 MANET을 위한 역추적 메커니즘에 대한 연구는 클러스터링 기법을 적용한 역추적 메커니즘은 다른 구조들에 비해 최대 29%의 CPU 사용률을 줄일 수 있기 때문에 클러스터링 기법을 이용한 역추적 메커니즘 방향으로 연구가 진행되고 있다[4-7]. 하지만 다중 홉 기반의 애드혹 네트워크에서 각 노드를 대표하는 Cluster_Head에서 많은 노드정보를 관리하기 때문에 Cluster_Head의 오버헤드를 유발하고, Cluster_Head의 수명을 감소시킨다.

본 논문에서는 다중 홉 기반의 애드혹 네트워크에서 Cluster_Head의 오버헤드를 줄이기 위하여 역추적 정보를 분산 관리하기 위한 트리구조를 이용한 TNA(Traceback against Network Attacks)을 제안한다. TNA는 다중홉 애드혹에서 각각의 노드에 개인 IDS에 있다는 가정 하에 IDS내에 Marking_Table을 생성한다. Marking_Table내에는 각 노드의 자식노드의 역추적 정보를 관리함으로써, 차후에 역추적 정보를 요구할 경우 Cluster_Head에서는 근원지 주소에서 보내는 요청에 응답하여, 해당 클러스터 내에 근원지 주소에서 보내는 역추적 정보를 가지는 노드의 부모노드에게 공격자 노드를 단절 시킨다. 본 논문의 구성을 살펴보면 2장은 관련연구, 3장은 TNA설계, 역추적 정보 재구성 및 근원지 추적 4장은 비교분석, 5장은 결론으로 되어 있다.

Ⅱ. 관련 연구

현재 다양한 클러스터링 기법들이 연구되었는데 그 중 단일홉 클러스터링의 대표적인 방법은 Lowest ID Clustering[8]과 Highest Connectivity Clustering[9]이고, 다중홉으로 토폴로지를 구성하는 알고리즘으로는 Adaptive Multihop Clustering[10]이 대표적이며, 본 논문에서는 AMC클러스터링 알고리즘을 기반으로 역추적 기능을 탑재 하였으며 또한 현재의 애드혹 기반 역추적 메커니즘을 살펴보도록 한다[3].

AMC(Adaptive Multihop Clustering)알고리즘은 애드혹 네트워크의 클러스터 형성 시 멀티 홉을 지원하는 대표적인 방법이다. 각각의 노드를 5 개의 상태별로 분류하여 이 상태정보를 Hello 패킷에 담아 브로드 캐스팅하는 방법으로 토폴로지를 유지한다.

Node ID	CH ID	Hop count	State	Weight

그림 1. hello Message Format
Fig. 1. hello Message Format

2.1. 시간-태그 블룸 필터

시간-태그 블룸 필터 역추적 메커니즘[4]는 데이터 수집을 위해 네트워크를 클러스터로 나누고 각 클러스터 헤더 노드는 지나가는 패킷을 모니터링하기 위해 N-IDS(Network Intrusion Detection System)이 있다고 가정하여 로그를 축적하고, 그 로그 기반으로 역추적을 실행한다. 여기서 클러스터 헤더 내에 블룸필터는 패킷 정보를 입력하며 시간-태그에 해당 패킷의 수명 시간을 저장하여 제한된 저장 공간에서 새로운 로그 정보를 유지할 수 있다.

2.2. Hierarchical traceback 기법

Hierarchical traceback기법[5]는 애드혹 네트워크에 대한 해시 기반 역추적을 사용한 핫스팟 기반 역추적의 확장이며, 클러스터 기법을 사용하여 네트워크를 분할하고, 역추적 정보의 효율적인 수집을 위해서 XOR 연산을 사용하여 필요한 메모리의 양을 감소 시켰다. Hierarchical traceback기법은 각 클러스터에서 역추적 정보를 관리하고 효율적으로 추적 경로를 복원하며, 각 Cluster_Head 사이에 역추적 정보를 교환하는 방식으로, 각 노드의 이동성에 의한 추적 경로 복원의 성공률 저하를 피할 수 있고 역추적이 클러스터 헤더에 의해 처리되기 때문에 역추적에 대한 트래픽과 시간을 줄일 수 있다.

2.3. 기존 연구의 문제점

기존의 역추적기법에서는 중앙 집중식으로 클러스터헤드가 클러스터링 영역의 노드들을 관리하게 되는데, 멀티 홉을 기반으로 하는 클러스터링의 경우 Cluster_Head와 멤버 노드간의 홉 수는 매우 다양하다. 역추적 정보를 관리하기 위하여

Cluster_Head에게 자식 노드의 패킷의 역추적 정보를 요구 하는 건 Cluster_Head의 오버헤드뿐만 아니라 클러스터 노드의 자원의 소비로 인한, 빈번한 Cluster_Head의 재선출 과정을 하게 되며, 네트워크 상에 큰 영향을 끼치게 된다. 또한 클러스터 헤더가 동적으로 변하기 때문에 볼룸 필터의 유지가 매우 힘들다. Cluster_Head가 수명 및 노드의 이동 등으로 인하여, Cluster_Head가 변경될 시엔 볼룸 필터의 정보를 새로 선출된 Cluster_Head에게 전달해야 하는데, 이때 1홉 단위의 노드들이 Cluster_Head의 정보를 따로 백업을 해야 하는 번거로움을 가지기 때문에, 네트워크의 성능을 저하시킨다.

제안하고자 하는 시스템은 클러스터에 집중된 정보를 다른 부 클러스터 헤더들이 분할 관리하여 클러스터 헤더의 오버헤드를 줄일 수 있도록 역추적 기법을 설계하였다.

Ⅲ. TNA 설계

설계 부분에서는 먼저 기존 다중 홉 클러스터링 기법인 AMC 기반으로 클러스터링 영역 설정과 각 노드의 병합/분할, 추가/삭제가 진행된다는 가정 하에 본 논문에서 제안하는 다중홉 클러스터링 기반에서 역추적 정보를 분할하여 각각의 노드에서 관리를 위한 TNA(Traceback against Network Attacks)은 역추적 정보를 Cluster_Head에 집중하여 저장 하는 것이 아니라 각 노드의 Marking_table을 생성하여 노드의 marking_ID기반으로 역추적 정보를 관리하여, 각 Cluster_Zone에서 대표하는 Cluster_Head의 오버헤드를 줄이고, 해당 노드의 수명을 향상 시키고자 한다.

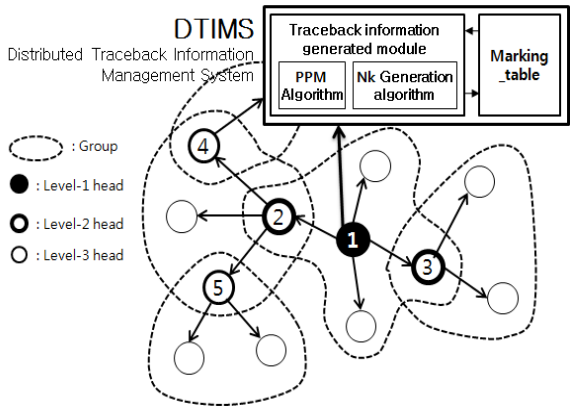


그림 2. TNA 전체적인 구성과 구성요소
Fig. 2. TNA overall structure and components

논문에서 제안하는 TNA는 그림2와 같이 역추적 정보 생성을 위한 Marking_table, Marking_table의 데이터를 이용하여 역추적 정보를 생성하기 위한 역추적 정보 생성 모듈 그리고 역추적 정보를 생성/수집 후에 역추적 정보를 관리하기 위한 역추적 관리 모듈로 구성된다. 그림2는 본 논문에서 설계한 TNA의 전체적인 구성과 하나의 클러스터링 영역 내에서 계층적으로 그룹을 나누어 부모노드가 자식 노드를 관리하는 구조로 형성한다.

3.1. 각각의 Cluster_Zone의 트리 구조 생성

AMC 알고리즘은 초기 각각의 Node의 ID를 생성한 후, 여러 개의 클러스터링 영역으로 구분되어. 각 영역마다 클러스터를 대표하는 Cluster_Head에 CH_ID를 부여 된다. 각각의 클러스터링이 나뉘어진후 본 논문에서는 각각의 클러스터링 내에 Cluster_Head를 기준으로 트리구조를 형성한다. 트리구조 형성 과정은 각각의 클러스터 Zone마다 독립적으로 수행하며, Cluster_Head는 관리자 역할을 수행하며, AMC 알고리즘의 Hello 메시지의 정보를 바탕으로 그림3과 같이 트리구조를 형성한다. 그림3은 AMC알고리즘에서 최대 임계치값이 20일경우의 Cluster_Zone의 트리구조로 나타낸 그림이다. AMC알고리즘의 Hello메시지에 있는 hop Count는 각각의 노드에 대한 Cluster_head의 Hop정보를 가지고 있으므로, 이를 기반으로 트리를 유지 할 수 있지만, AMC알고리즘의 Hop count는 Cluster_head의 Hop정보만을 가지고 있기 때문에 실제로 직계 부모가 누구인지를 알 수 없으므로, 초기 트리구조를 형성할 때는 flooding에 의존할 수밖에 없다. 먼저 Cluster_head인 A노드는 1-hop단위로 flooding을 하여 주변 노드탐색을 한다. 그리고 이러한 주변 노드의 정보를 Marking_table에 작성을 한 후, A노드에서는 자식노드들의 Marking_ID값을 부여해준다. Marking_ID값은 초기 노드_ID값을 이용하여, Marking_table에 순차적으로 Marking_ID값을 부여한다. A노드에서 등록이 완료된 노드에서는 차후 다른 노드에서 보내는 메시지에 대해서는 무시를 한다. Level-1에서 각각의 자식노드인 5개의 노드중에서 가입되어 있지 않는 주변 노드와의 연결도가 가장 높은 노드가 자식노드를 가질 수 있는 우선권이 주어지며, B노드에서도 마찬가지로 flooding하여, 주변 노드를 탐색한 후, 가입 메시지에 대한, 응답 메시지가 돌아오는 노드에 한하여, Marking_table에 노드의 정보를 저장하고, 해당 노드의 자식노드의 수를 부모노드인 A노드에게 전송

하게 된다. 이렇게 순차적으로 진행하면 그림3-(B)의 같은 트리구조가 생성 된다. 여기서 중요한 것은 노드의 배치에 따라, 예를 들어, 1-hop내에 많은 노드들이 배치되었다면, 하나 노드에서 많은 자식 노드를 관리해야 하므로, 이럴 경우 자식 노드의 숫자를 제한하여, 하나의 노드에서 관리할 수 있는 자식 노드의 수를 줄인다.

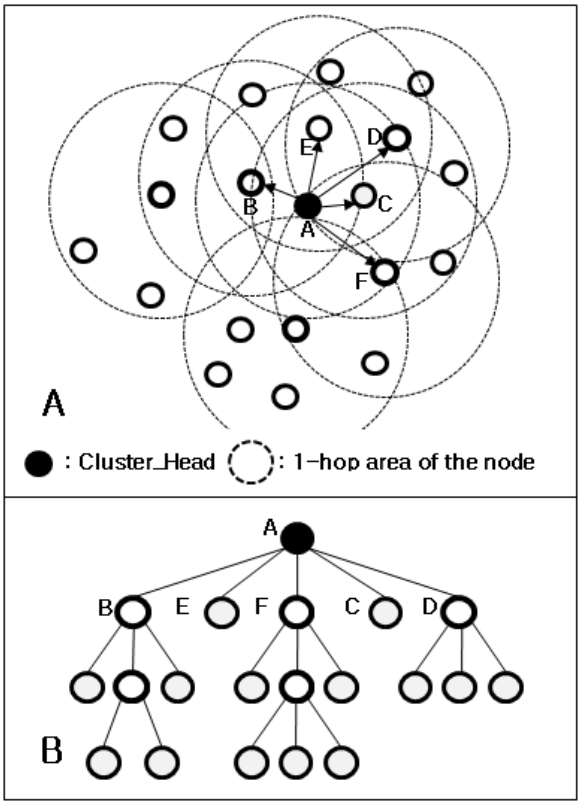


그림 3. Cluster_Zone의 트리구조 형성
Fig. 3. The tree structure of the Cluster_zone formation

3.2. Marking_table 설계

각각의 Cluster_Zone의 트리구조가 형성이 된다면, 먼저 Cluster_Head에서는 각각의 자식노드의 정보를 관리해줄 Marking_table의 정보를 작성한다. 그림 4는 제안하는 클러스터 기반의 트리 구조의 자식노드의 Marking_ID생성 및 자식노드의 정보를 관리해주는 Marking_table이다. 자식노드의 Marking_ID값은 초기 Node_ID값에서 가장 높은 값의 ID나 낮은 값의 ID을 기준하여 순차적으로 Marking_table에 Marking_ID값을 부여한다. marking_table은 클러스터 삽입과 삭제, 병합, 분할 그리고 역추적 정보를 관리해주는 테이블로써 각 노드에 Marking table을 삽입하여 트리구조의 유지, 관리한다.

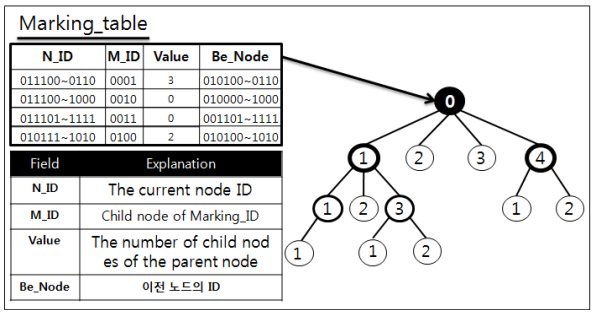


그림 4. 자식노드의 Marking_ID 생성과 Marking_table
Fig. 4. Child nodes of Marking_ID generated and Marking_table

그림 4에서의 Marking_table은 각각의 노드에 저장되는 meta정보다. Marking_table에서는 현재 노드의 ID인 N_ID, 자식 노드의 Marking_ID인 M_ID, 해당 노드의 자식 노드의 수를 나타내는 Value값 그리고 이전에 해당위치에 존재했던 Be_Node 필드로 구성되어 있다. Be_Node는 MANET의 특성상 노드의 위치가 고정적이지 못하기 때문에 노드의 위치가 변동될 시 Be_Node의 정보를 이용하여, 근원지 추적을 하기 위한 정보로 만약 그림 4의 Level-1의 4노드위치에서 기존의 노드가 이동하여 새로운 노드가 4노드로 위치하게 된다면, 패킷 로그 바탕으로 시간대를 확인하여 공격 시 발생한 시간대와 기존의 노드의 이동시간, 현재 노드가 새로 위치선정한 시간대를 살펴 역추적을 실행한다.

3.2.1. Marking_table의 트리구조화

실제로 역추적을 하기위한 정보는 Marking_table의 정보를 바탕으로 역추적정보를 생성하고 이를 기반으로 역추적을 실행한다. Marking_table을 작성 시 부모 노드는 자식노드의 정보를 관리하며 직계 자식의 정보만을 관리/유지한다. 단말 노드는 따로 관리할 정보가 없기 때문에 Marking_table의 정보를 유지하지 않는다. Marking_table은 직계자식 노드가 있는 부모 노드에 관해서만 관리/유지하며, 이러한 정보는 그림 5와 같이 트리 구조를 작성할 수 있다.

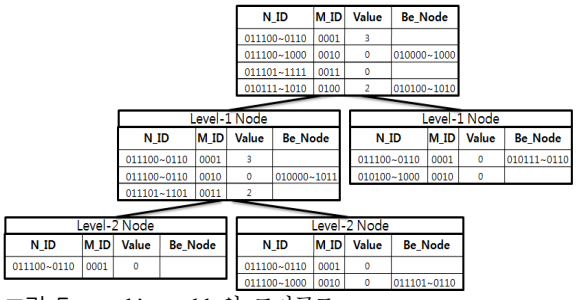
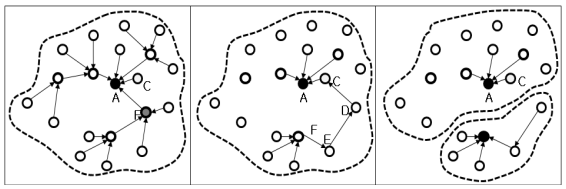


그림 5. marking_table의 트리구조
Fig. 5. Marking_table tree structure

그림 5의 Making_table의 트리구조는 그림 4의 트리구조를 표현한 것으로 실제적으로 관리하는 정보만을 다중홉 클러스터링의 Level-2 트리로 표현한 것을 확인할 수 있다.

3.2.2. 노드 삽입과 삭제

노드가 이동하거나 노드수명으로 인하여 다중홉 클러스터링을 유지하기 위한 Hello 메시지의 통신 두절이 일어나거나 새로운 노드가 합류하게 되면, AMC 알고리즘은 노드의 최대 임계치와 최소 임계치값에 맞춰 삽입이나 삭제 및 클러스터링의 병합 및 분할 과정을 거치게 된다. 제안하는 시스템에서는 트리구조를 유지해야 하는 번거로움을 가지고 있지만, 특정한 노드를 제외한 예를 들어, 단말노드의 삽입/삭제 연산은 해당 부모노드의 Marking_table의 메타데이터만을 수정하기 때문에 트리 구조에 큰 영향을 끼치지 못한다. 또한 노드 삽입에 관해서는 주변 부모노드의 자식노드로 트리에 포함하면 쉽게 문제가 해결 된다. 그러나 자식이 딸린 부모노드의 삭제는 트리 구조에 큰 영향을 끼치며 이러한 단점을 보완하기 위해 이 절에서는 노드의 효율적인 삭제를 위한 방법을 제시한다.



(a) (b) (c)

그림 6. 노드삭제와 클러스터링의 분할
Fig. 6. Delete a node and the splitting of the clustering

그림 6-(a)에서 부모 노드 C가 삭제나 이동을 하였을 경우 먼저 자식 노드들은 주변의 노드를 탐색하게 되며, Cluster_haed와의 홉 수를 측정하여, 최단위치의 노드를 선정하여 자식노드로 가입하게 된다. 즉 우선순위는 노드의 트리의 깊이이며, 해당 조건에 만족하면, 해당노드의 자식 노드로 그림 6-(b)과 같이 연결 된다. 그러나 트리의 깊이가 높으면 높을수록 트리의 성능이 떨어지기 때문에 트리의 깊이에 제한하여 해당 깊이가 넘으면, 다른 클러스터링에 포함되거나 혹은 그림 6-(b)의 트리의 깊이는 level-5 이며 트리깊이의 제한을 Level-4로 하여 그림6-(c)과 같이 두 개의 클러스터링이 형성하게 된다. Marking_table의 변화를 보면 그림 6-(b)의 경우 단말 노드였던 node C의 Marking_table에 새로 추가된 노드만을 따로 메타데이터를 작성하고 이정보는 바로 직계 부모인 노드A에게 전달하게 된다. node D,E또한 node C의 방법과 마찬가지로 Marking_table을 새로 작성을 하게 되며, node F는 기존과 동일한 것을 확인할 수 있다. 그림 6-(c)의 경우 node B에 있던 Marking_table을 삭제함으로써 Marking_table의 트리구조가 유지 되는 것을 확인할 수 있다.

3.3. 역추적 정보 생성 모듈

클러스터 노드가 통신을 하기 위해선 자신이 포함된 Cluster_Zone의 Cluster_Head를 통해 경로 설정을 확인한 후 노드는 해당 경로를 통해 통신을 하게 될 것이다. 대상 노드가 Cluster_Head와 통신을 할 때 부모 노드의 결정에 의해 확률 기반 패킷 마킹 기반으로 식 (1)을 통해 그림 7과 같이 역추적 정보를 생성한다.

$$(C_{(f)} + P_{(n)} \cdot P_{(f)}) - C_n = N_k \tag{1}$$

부모노드의 자식 Marking_ID과 멤버의 수는 Marking table의 정보를 통해 확인할 수 있으며, 이러한 정보를 기반으로 역추적 정보를 생성하게 된다. 식 (1)에서 C_f는 부모노드에 딸린 자식노드의 멤버의 수를 의미하며, C_n은 노드의 Marking_ID를 의미한다. P_f는 부모노드의 멤버의 수 P_n은 부모노드의 Marking_ID를 의미하며, N_k는 역추적 정보를 의미한다. 이와 같은 공식을 통하여, 역추적 정보를 생성하게 된다. N_k는 전체 트리에서는 같은 값을 가질 수 있지만, 트리의 같은 깊이 내에서는 똑같은 값을 가질 수 없다. 그림 7은 각각의 노드들이 식(1)을 통하여 역추적에 사용할 N_k값 가진다.

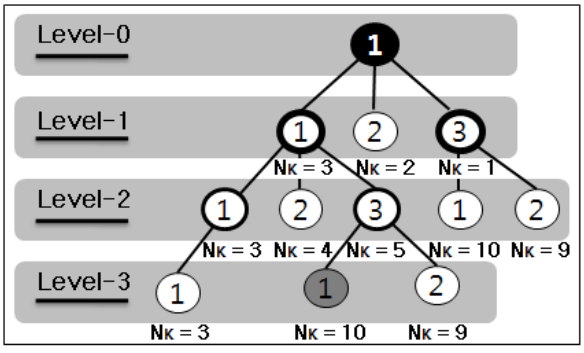


그림 7. 클러스터링 노드들의 역추적 정보 생성
Fig. 7. Clustering of nodes traceback information generated

예를 들어 Level-2에서 Marking_ID값이 3인 자식노드는 Marking_ID 1인 노드가 통신을 하게 된다면, 부모노드에서는 패킷 마킹을 p확률로 마킹을 하게 된다. 식(1)에서 C_f값은 2가 되며, C_n은 자료를 전송하려는 노드의 Marking_ID값인 1이 되며, P_f는 부모노드의 멤버수인 값 3이 되며. P_n은 부모 노드의 Marking_ID값 3을 가진다.

이러한 값을 기반으로 식(1)에 대입하면, N_k값은 10이라는 값을 가지게 된다. 여기서 역추적에 필요한 정보로는 첫째로, N_k이며, 둘째로는 Cluster_Head의 Marking_ID 세 번째로는 트리의 깊이이다. 여기서 트리의 깊이는 AMC알고리즘의 hello Message Format에서 hop count가 트리의 깊이를 나타내기 때문에 이정보의 값을 트리의 깊이로 사용한다. 이와 같은 3가지 정보를 IP패킷에서 사용되지 않는 부분인 identification공간에 그림 8과 같이 마킹한다.

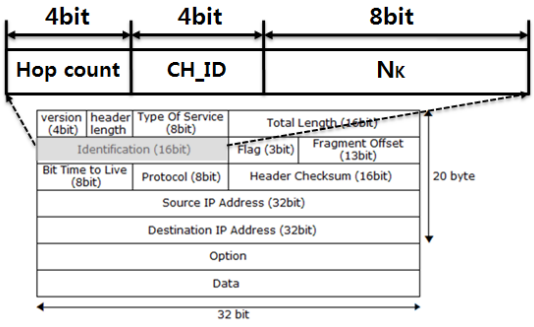


그림 8. IP패킷 헤더의 identification 필드 내에 역추적 정보 삽입방법
Fig. 8. Traceback in the identification field of the IP packet header information inserted

그림9는 확률 기반 패킷 마킹 알고리즘에서 노드가 역추적을 위해 받는 패킷을 처리하는 방법을 나타낸다.

```
For each Packet, P
 if(P.Marking flag==on){
  store the previous Marking information and sends is
to the Parent node;
 P.marking field = IP address of current node;
 P.marking flag = off;
}
 else {
choose a random number x in [0,1];
if (x<q) {
 store the previous Marking information and sends is
to the Parent node;
  P.Marking field = ip address of current node;
  }
  else forward P;
 }
```

그림 9. PPM 알고리즘
Fig. 9. PPM algorithm

패킷을 자식노드에게 받는 경우, 부모노드에서는 자식 노드에게 전송받은 패킷에 대하여 사용자가 지정하는 확률 q로 패킷을 마킹을 할 것 인지를 결정하며, 패킷에 마킹을 하게 된다면, identification 공간에 Hop count, Cluster_Head의 Marking_ID, N_k 값을 삽입한다.

3.4. 역추적 정보 재구성 및 근원지 추적

네트워크를 구성하는 모든 노드는 호스트 기반의 IDS를 탑재하고, 노드 자신으로의 공격을 탐지할 수 있다고 가정한다. 네트워크 내부의 모든 노드들은 공격이 탐지되면, 자신이 속한 클러스터 헤더 노드에게 공격패킷 내에 identification필드에 있는 마킹된 역추적 정보를 전송한다. 그림10은 공격 패킷의 흐름과 역추적 과정을 설명한 그림으로써, 클러스터헤드는 자신과 인접한 클러스터의 헤드들에게 해당 Marking_ID를 가진 Cluster_Head확인 요청 메시지를 전송한다.

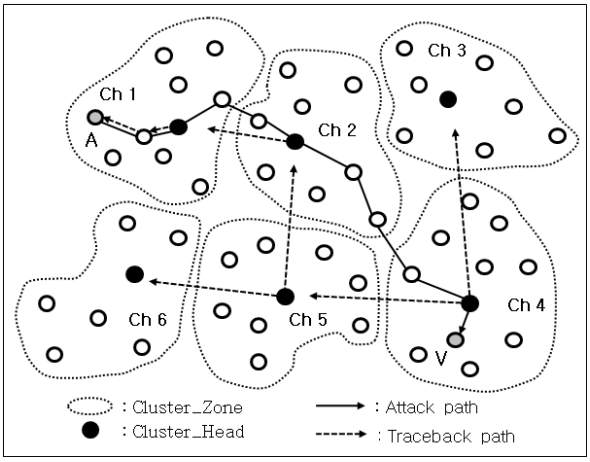


그림 10. 공격 패킷의 흐름과 역추적 과정
Fig. 10. Tracking process and reverse the flow of the attack packets

만약 자신이 해당 Marking_ID를 가진 클러스터 헤드라면 요청에 대한 응답을 보내고, 만약 가지고 있지 안다면, 인접한 Cluster_Head들에게 Marking_ID 확인 요청 메시지를 보낸다. 이와 같은 과정을 통해 해당 정보를 가진 Cluster_Head를 찾게 된다면, 해당 클러스터는 역추적 정보의 홉 카운트를 확인하여 해당 노드의 위치를 파악한다. 홉 카운트는 곧 트리의 깊이를 의미하므로 Marking table의 정보를 통하여, 해당 위치에 있는 노드들의 부모들은 식 (1)을 통하여, N_k 값을 가지는 노드를 확인하면, 해당 클러스터에서 공격노드로부터 발생되는 모든 트래픽을 필터링하거나 클러스터링 영역에

서 탈퇴 및 삭제 등의 조치를 취할 수 있다.

Ⅳ. 비교 분석

표 1은 앞서 제안한 다중홉 클러스터 기반의 애드혹 네트워크 환경을 위한 계층적 구조의 역추적 기법과 기존의 연구를 토대로 비교 분석 하였다.

표 1. 트리기반 역추적 기법과 기존의 역추적 기법의 비교 분석
Table 1. The inverse of the tree-based tracking technology and comparative analysis of existing traceback techniques

	Hierarchical traceback	Time-tagged Bloom Filter	The proposed scheme
Compatibility	X	O	O
Overhead Minimize	X	X	O
Scalability	O	△	△
Traceability	O	△	O
Packet Demand	O	△	△
Module changes of Node	X	O	X

(O : Good, △ : Medium, X : Bed)

호환성 비교분석에서는 Hierarchical traceback기법은 별도의 SPIE 시스템을 사용하기 때문에 애드혹 네트워크의 특징상 외부 시스템의 도움 없기 독립적인 망을 형성해야 하며, 시간 블룸필터 기법과 제안하는 기법은 외부의 시스템 도움 없이 자체적으로 역추적이 가능하다.

오버헤드 최소화 분석에서는 기본적으로 매우 큰 클러스터가 생성되면, 클러스터 헤드가 관찰해야 할 노드 수가 증가하고 이는 오버헤드를 유발하는 원인이 된다. 또한 크기가 작은 클러스터가 많으면 클러스터 헤드들 사이에서 발생하는 데이터의 교환이 증가하여 네트워크의 트래픽을 증가시킨다.

Hierarchical traceback과 시간블룸필터 역추적 기법은 중앙 집중식으로 클러스터헤드가 클러스터링 영역의 모든 노드들의 역추적 정보를 관리하게 되며, 이는 다중홉 애드혹 네트워크에서 Cluster_Head가 관리해야 할 노드의 수는 더욱 증가하기 때문에 Cluster_Head의 오버헤드를 유발한다. 제안하는 기법에서는 이러한 오버헤드를 막기 위해 각각의 노드는 직계 자식만을 관리 하게끔 하여, 오버헤드를 줄였다.

확장성 비교분석은 애드혹 네트워크는 주기적인 노드의 변화에 따른 대처능력을 말하며, 시간블룸필터는 헤드의 주기적인 변화에 맞춰서 Cluster_Head의 블룸필터의 데이터를 새로 선정된 헤드에게 데이터를 전송해야하며, 또한 헤드의 통신 두절을 위해 1홉 단위의 노드들에게 블룸 필터의 내용을 따로 저장을 해야 하는 번거로움을 가지고 있으며, 제안하는 기법은 클러스터 토폴로지가 변경 되었을 때, 다시 트리 구조를 재설정해야 하는 번거로움을 가지고 있다.

추적성 비교분석은 DDOS의 공격에 대한 정확한 노드의 역추적 성능을 말하며, 시간블룸필터는 Cluster_Zone을 블룸필터를 이용하여, 클러스터헤드를 확인을 할 수 있지만, 공격자가 해당노드의 주소값을 변경하면, 정확한 노드를 찾을 수 있는 방법이 없다. 패킷 요구량 비교분석에서는 역추적을 할 때, 필요로 하는 패킷의 요구량을 말하며, 시간볼룸태그와 제안하는 기법은 해당 클러스터헤드를 찾기 위해 flooding을 하여 주변 노드를 탐색해야하기 때문에 네트워크 트래픽을 유발 시킬 수 있다는 단점을 가지고 있다.

노드의 모듈 변화 비교분석에서 Hierarchical traceback기법과 제안하는 기법은 역추적을 하기 위해서는 각각의 노드에 부가 적인 모듈이 필요로 하기 때문에 많은 변화가 요구 된다.
위에서 비교 결과 제안하는 역추적기법은 다양한 기준의 측면에서 애드혹 네트워크에서 기존의 메커니즘보다 더 나은 성능을 보여주고 있다. 또한, 제안하는 역추적 기법은 다중홉 클러스터링 기반의 특징상 클러스터헤드의 자원소모를 줄이기 위하여, 역추적 정보를 분산관리 할 수 있도록 제안하였다.

Ⅴ. 결 론

MANET은 제한된 대역폭, 계산 자원 및 배터리 전원을 가진 노드들로 구성된 네트워크이기 때문에, 각 노드가 가지는 자원을 효율적으로 관리해야하며, 다중 홉 클러스터링일 경우 Cluster_Head에서 관리해야할 노드의 수는 더욱 많아진다. 각 노드를 대표하는 Cluster_Head에서 많은 노드정보를 관리하기 때문에 Cluster_Head의 오버헤드를 유발하고,

Cluster_Head의 수명을 감소시킨다. 본 논문에서는 다중 홉 기반의 애드혹 네트워크 환경에서, 기존에 존재하던 애드혹 역추적기법들에 비해 Cluster_Head의 중앙집중식 역추적정보를 관리하는 방법에서 벗어나 Cluster_Head의 오버헤드 및 헤드의 수명단축을 줄이기 위해 역추적 정보를 분산관리 하기 위한 트리기반인 분산 관리 시스템을 제안 하였다. 각 클러스터의 토폴로지에 존재하는 Cluster_Head와 각 노드들은 트리 형태의 구조를 가져 직계자손만을 관리함으로써, 특정 노드에 집중될 수 있는 패킷을 분할, 관리 하였다. 각각의 노드의 Marking_table에 있는 Marking_ID 정보 토대로 확률적 패킷 마킹 기반의 패킷에 마킹하는 방식이며, 차후 역추적을 실행 시에는 패킷에 저장되어 있는 정보를 바탕으로 역추적을 실행 할 수 있도록 하였다. 또한 Hierarchical traceback기법과 시간블룸필터 역추적 기법을 토대로 비교분석하였으며, 두 기법에 비해 Cluster_Head에 유입되는 패킷의 양이 현저히 줄어든 것을 확인할 수 있었다.

참 고 문 헌

[1] Geun-Bin Hong, Ji-hun Yun, Kwan-Woong Kim, ″Study of the Wireless Ad-hoc Networks with Robust Route Maintenance Scheme.″ KIIECT vol.3, no.2, pp.46-49, Jun. 2010.

[2] In-ho Yeo, Hyo-sik Yang, Jong-Myung Rhee ″Impacts of Radio Propagation Model on Mobile Ad-hoc Network (MANET) Performance in Group Mobility Environments″ KIIECT vol.3, no.3, pp.62-72, Sep. 2010.

[3] Y Kim, A Helmy, ″Attacker Tracevack With Cross-layer Monitoring in Wireless Multi-hop Networks.″ SASN, Oct. 2006

[4] I1 yong Kim, Ki Chang Kim, ″A Resource-efficient IP Traceback Technique for Mobile Ad-hoc Networks Based on Time-tagged Bloom Filter.″ ICCIT. Nov. 2008

[5] yatagai, Iwao sasase, m_ishii, ″Hierarchical traceback method using clustering for mobile ad hoc networks″, PIMRC IEEE. PP. 2623-2627, 2009

[6] Yongjin Kim, Ahmed Helmy, ″SWAT : Small World-based Attacker Traceback in ad-hoc Networks.″ *MOBIQUITOUS*, Jul. 2005

[7] yi-sn husng, wenke Lee. ″hotspot-Based for Mobile Ad-hoc Networks,″ *ACM Pro. of the 4th ACM workshop on wireleess security*. pp. 43-54. Sept. 2005

[8] Y. Huang and W. Leem ″A Cooperative Intrusion Detection System for Ad Hoc Networks.″ *Pro. of the ACM Workshop on Security in Ad hoc and Sensor Networks. Fairfax*, VA. USA. 2003.

[9] MAINAK CHATTETJEE, SAJAL K. DAS and DAMLA TURGUT, ″WCA: A Weighted Clustering Algorithm for Mobile AD Hoc Networks″, *Cluster Computing 5*, pp.193-204, 2002

[10] Sung-Gook Lim, Hun-je Yeon, JaiYong Lee ″Adaptive Clusterhead Positioning for Muti-hop Ad-hoc Networks″ *JCCI 2006*

김 주 용 (Ju-Young Kim)

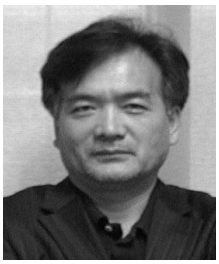
2007년 2월 관동대학교 컴퓨터공학과 졸업

2009년 2월 관동대학교 전자계산 공학과 석사

2009년 3월~현재 관동대학교 전자계산공학과 박사과정

<관심분야> 네트워크보안, 센서 네트워크

이 병 관 (Byung-Kwan Lee)



1979년 2월 부산대학교 기계설계학과 공학사

1986년 2월 중앙대학교 전자계산공학과 공학석사

1990년 2월 중앙대학교 전자계산학과 공학박사

1988년 3월~현재 관동대학교 컴퓨터학과 교수

<관심분야> 네트워크 보안, 컴퓨터 네트워크, 전자상거래