논문 2007-44CI-1-14

Hop-depth 알고리즘을 이용한 무선 센서 네트워크상에서의 내부공격자 및 공모노드 검출

(Detecting Inner Attackers and Colluded nodes in Wireless Sensor Networks Using Hop-depth algorithm)

이 강 현*

(Kang Hyeon RHEE)

요 약

다수의 노드로 구성된 무선센서네트워크는 Ad-hoc 방식으로 노드간 통신이 이루어지며, 센싱데이터는 베이스노드로 취합되어진 후 Host PC에 의해 처리되어지게 된다. 하지만 Ad-hoc 방식의 네트워크는 잘못된 라우팅 정보를 전달하거나 데이터를 중간에 소실 및 변경시키는 싱크홀 공격에 대해 취약할 수밖에 없다. 이러한 싱크홀 공격은 네트워크의 오버헤드를 가중시키고 전체 네트워크의 배터리 소모를 가속화 시켜 전체 네트워크의 수명을 단축시키고 또한 다른 공격을 적용하기가 용이하므로 싱크홀 공격에 대한 대응방법은 신중하게 고려되어져야 한다. 본 논문은 이러한 싱크홀 공격이 일어났을 때 공격노드와 주변의 공모노드를 검출해 낼 수 있는 Hop-depth 알고리즘을 제안하였다. 제안된 알고리즘은 홉 깊이가 변경되는 공격의심 노드들을 검색한 후 주변의 공모노드의 경로 값을 재계산한 후 실제 공격이 이루어지는 노드를 검출해 낸다. 제안된 알고리즘의 성능평가는 NS2를 사용하여 이루어 졌으며 원 공격노드 검출성공률, 양성오류율 및 음성오류율, 에너지 소비효율 값을 비교분석하였다.

Abstract

Commonly, in the Sensor Network that composed with multiple nodes uses Ad-hoc protocol to communicate each other. Each sensed data packets are collected by base node and processed by Host PC. But the Ad-hoc protocol is too vulnerable to Sinkhole attack, where the intruder attracts surrounding nodes with unfaithful routing information, and then performs selective forwarding or changes the data passing through it. The Sinkhole attack increases overhead over the network and boosts energy consumption speed to decrease network's life time. Since the other attacks can be easily adopted through sinkhole attack, the countermeasure must be considered carefully. In this paper, we proposed the Hop-depth algorithm that detects intruder in Sinkhole attack and colluded nodes. First, the proposed algorithm makes list of suspected nodes and identifies the real intruder in the suspected node list through the Hop-depth count value. And recalculates colluder's path information to find the real intruder. We evaluated the performance of the proposed algorithm using NS2. We compared and analyzed the success ratio of finding real intruder, false positive ratio, false negative ratio, and energy consumption.

Keywords: Hop-depth, Sinkhole, malicious node, misdirecting

I. 서 론

무선통신기술과 전자기술의 발전으로 말미암아 저가격, 저전력, 다기능 센서 노드로 구성된 무선 센서 네트워크에 대한 관심이 급격히 고조되고 있다. 기존의 무선 네트워크와는 달리 센서 네트워크는 배터리, 메모리 등과 같은 제한된 하드웨어를 가지고 있기 때문에 네트워크상의 프로토콜 및 보안정책은 신중하게 고려되어야 한다.[1]

WSN(Wireless Sensor Network)의 응용분야로는 환경 모니터링, 군사용 센싱 및 추적, 스마트 환경 등으로

* 평생회원, 조선대학교 전자정보공과대학 전자공학과
(Dept. of Electronic Eng., College of Elec-Info Eng., Chosun University, Korea)
접수일자: 2006년12월11일, 수정완료일: 2007년1월17일

특히 센서노드가 위험지역에 설치되었을 경우 보안은 매우 중요한 요소가 된다. 네트워크는 내부 또는 외부로부터 공격을 받을 수 있으며, 이와 관련하여 많은 보안정책이 연구되어 왔다. 그 중 공개키를 이용한 방식은 에너지와 노드의 계산능력의 한계로 인하여 센서 네트워크상에 적용하는 것이 바람직하지 않으며, 사전 키 분배방식은 센서노드가 배치되기 전에 키를 미리 저장하는 방법으로서 기존연구에서는 노드별로 키를 보유해야 하는 단점이 있었다. 일반적으로 무선망을 이용한 센서 네트워크상에서의 통신방법은 브로드캐스팅 방식이다. 이것은 최소의 자원소모를 가지면서 적절한 보안수준을 구현하기 위한 것이다. 하지만 브로드캐스팅 방식의 경우 공격자는 무선 네트워크상의 어느 위치에서든지 도청이 가능하며 이로 인한 공격자의 메시지 위·변조에 의한 공격이 전체 네트워크에 큰 위험이 될 수 있다. 센서 네트워크의 특성상 각 노드에 대한 보안노출이 다른 노드에도 영향을 주기 때문에 특정노드의 보안노출이 다른 노드에 영향을 끼치지 못하도록 노드간의 신뢰성보장 및 공격자 식별은 무선망에서의 중요한 보안 요소가 된다. 특히 Ad-hoc 방식에 의해 통신이 이루어지는 센서네트워크는 싱크홀 공격에 대하여 매우 취약한 구조를 갖는다. 싱크홀 공격에서 공격노드는 주변의 노드에 잘못된 라우팅 정보를 제공하여 공격반경 안에 있는 노드들의 데이터를 모두 전송받은 후 잘못된 경로로 재전송하거나 중간에 데이터를 소실시키는 등 베이스 노드가 정확한 데이터를 받지 못하게 하며 전체 네트워크의 오버헤드를 가중시켜 네트워크의 수명을 줄이게 한다[2,3].

본 논문은 이러한 싱크홀 공격에 대하여 공격노드를 정확하게 식별해 낼 수 있는 알고리즘을 제안하였다. 공격노드 식별을 위해 베이스 노드는 홉의 실제 베이스 노드로부터의 깊이 값과 각 노드에 저장되어 있는 홉 카운트 값을 비교하여 공격노드로 의심되는 노드의 리스트를 만든다. 그리고 전송경로의 데이터 전송흐름을 분석하여 리스트로부터 원 공격노드를 검출한다. 이때 원 공격노드의 주변에 있는 악의를 가진 공모노드는 원래 공격노드가 아닌 주변의 노드로 데이터를 임의로 전송하여 베이스 노드가 원 공격노드를 검출해낼 수 없도록 한다.

본 논문의 구성은 Ⅱ장에서 싱크홀 공격과 In-network 프로토콜에 대해 설명한 후, 제안된 알고리즘에 대해 설명하고 Ⅲ장에서 제안된 알고리즘의 성능을 비교 및 분석한 후, Ⅳ장에서 결론을 맺는다.

Ⅱ. 본 론

2.1 싱크홀(Sinkhole) 공격

다대일 통신을 하기 위해 Ad-hoc 프로토콜을 사용하는 베이스 노드와 다수의 노드들로 구성된 센서네트워크가 있다고 가정해 보자. 각 노드는 고유의 $Node_{ID}$ 값을 가지며, 노드의 초기배치 후 베이스 노드를 뿌리로 가지는 데이터 전송트리를 형성한다. 싱크홀 공격노드는 자신의 RF반경안에 위치한 주변노드들에게 advertising 메시지를 전송함으로써 자신이 베이스노드와 가장 가까운 노드라는 거짓 정보를 알리고, 주변노드의 센싱데이터가 자신에게로 유입되도록 유도한다.[4,5] 그림 1은 네트워크 상에서의 싱크홀 공격을 나타낸다.

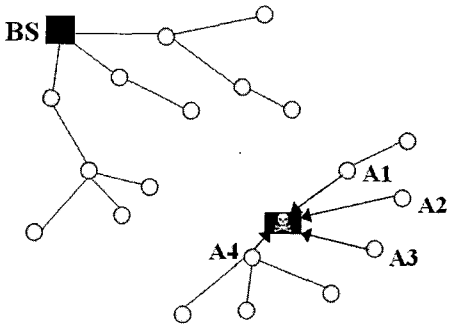

그림 1. 무선 센서 네트워크상에서의 싱크홀 공격

Fig. 1. Sinkhole attack in wireless sensor network.

2.2 In-Network 프로토콜

In-Network 보안구조[6]는 내부공격자가 존재하더라도 전송경로 상에서 발생할 수 있는 데이터의 변경이나 헤더의 손실을 최소화하기 위한 보안방법이다. 베이스 스테이션은 네트워크상의 전체 노드수에 비해 극히 적은 비율의 key set을 구성하고 이 중에서 임의로 하나의 key값을 선택한 후 각 노드의 데이터에 대한 변경 및 검사 권한을 할당한다.

변경권한을 할당받은 검사노드들은 전송경로 상에서 데이터패킷을 검사하여 내부공격이 감지되면 중간에 데이터를 차단시킴으로써 내부공격에 대응할 수 있게 된다. 하지만 실제로 초기에 센서노드가 배치될 때 데이터 변경권한이 부여된 검사노드가 전체 네트워크상에 항상 균일하게 퍼져 있을 수는 없다. 만약 검사노드가 전체 네트워크상의 특정부분에 몰려서 배치되었다고 할 때 검사노드를 거치지 않은 네트워크의 나머지영역은

내부공격자의 공격에 대해 취약해질 수밖에 없으며, 공격노드에 의해 시간이 지날수록 노드들의 에너지 자원은 고갈된다. 또한 검사노드에는 다른 노드들에 비해 상대적으로 많은 오버헤드가 걸리기 때문에 검사권한을 일정주기마다 다른 노드에게 위임하는 안전한 보안정책도 고려되어져야 한다. 만약 검사노드자체가 악의를 지닌 공격자에 의해 공격을 당하여 취득 당했을 경우 베이스노드는 네트워크상의 공격노드를 식별해 낼 수 없게 되고 만약 이러한 공격노드를 통해 수정된 데이터가 베이스노드로 유입되면 전체 네트워크의 보안에 치명적일 수밖에 없다. 그림 2에 노드 배치시 일어날 수 있는 In-network 프로토콜의 취약점을 나타내었다. 그림 2와 같이 검사노드가 네트워크상에 균일하게 배치되지 않을 경우 검사노드가 존재하지 않는 영역이 내부공격자에 의해 오염되면 공격노드는 주변노드에 계속해서 advertising message를 보내게 되고 싱크홀 공격이 발생하게 된다.[2]

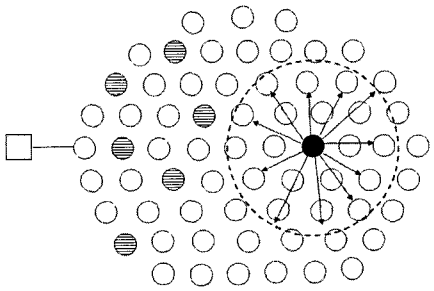

그림 2. 노드배치시 발생하는 보안의 취약점

Fig. 2. Fragility of sensor depolyment.

2.3 공격영역의 범위산출

대부분의 싱크홀 공격은 misdirecting에 의한 공격을 사용한다. 만약 어떤 특정영역으로부터 센싱데이터가 지속적으로 전달되지 않는다면 베이스 노드는 그 영역으로부터 misdirecting에 의한 공격이 일어났다고 가정할 수 있으며, 식 (1)을 이용하여 공격여부를 판단할 수 있다. 식 (1)에서 $X_1, \cdots, X_n$을 수집된 센싱데이터를 의미하며 $\overline{X}$는 평균값을 의미한다.

$$f(X_j) = \sqrt{\frac{(X_j - \overline{X})^2}{\overline{X}}} \quad (1)$$

각 노드에 대한 $f(X_j)$가 임계값보다 클 경우, 이것은 같은 지역으로부터 전달되는 데이터가 일정하지 않다는 것을 의미하므로 공격노드로 의심할 수 있다.[7,8] 이때 공격의심노드들의 집합이 원모양을 이루면 베이스노드는 싱크홀 공격이 이루어지는 영역을 가늠할 수 있게 된다. 일단 공격노드로 의심되는 노드들을 찾아내면 베이스 노드는 싱크홀이 어디에 존재하는지를 추측할 수 있다. 그림 3에 공격의심노드를 이용한 공격영역의 산출을 나타내었다.

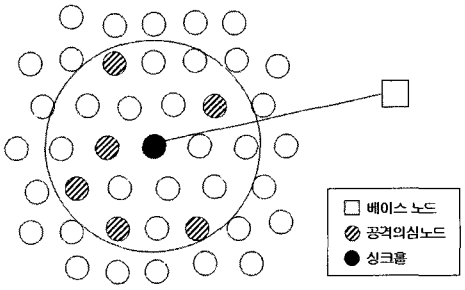

그림 3. 공격의심노드를 이용한 공격영역의 추정

Fig. 3. Estimating attacked area using suspicious nodes.

2.4 제안된 프로토콜

베이스 노드는 각 노드의 ID가 포함된 요구메세지를 네트워크에 전송한다. 공모노드의 요구메세지에 대한 패킷 재전송이나 데이터 패킷의 조작을 막기 위해 베이스 노드는 타임스탬프 TS를 메시지패킷에 설정하고 개인키 K_{BS}로 서명한다. 따라서 베이스 노드가 보내는 메시지는 $[TS, ID_1, \cdots, ID_n]K_{BS}$가 된다. 요구메시지를 전달받은 노드 v는 자신의 노드 ID와 다음으로 패킷이 전달될 노드의 ID번호, 노드가중치를 구한 후 각각의 노드에 저장한 후 반송메세지를 베이스노드에 재전송한다. 따라서 각 노드에서 베이스노드로 반송되는 패킷은 $< ID_v, ID_{nexthop}, cost >$와 같다. 이렇게 하여 각 노드는 베이스노드와 유니캐스트 경로를 설정할 수 있으며, 베이스노드는 네트워크의 각 노드에 대한 데이터 전달트리를 생성하게 된다. 센싱데이터는 처음에 베이스 스테이션으로부터 전달받은 경로의 역방향으로 베이스 스테이션에 전달된다.

센서노드가 센싱데이터를 베이스 스테이션에 전송할 때 노드에 저장된 다음 홉 노드 ID로 데이터를 보내게 되며, 매 홉 별로 데이터가 전달될 때마다 cost는 감소하게 된다.

2.4.1 공유키생성

공모노드에 의해 전송도중 데이터가 변경되는 것을 막기 위해 각 노드는 베이스노드와 공유하는 각각의 개인키를 가지고 있게 하였다. 개인키는 배치되기 전에 각각의 노드에 입력되어지며, 베이스노드의 저장공간을 줄이기 위해 베이스노드의 개인키 K_{BS}에 의해 계산되어진다. 각 노드의 공유키는 각 노드의 ID번호를 사용하여 식 (2)의 의사난수 함수에 의하여 계산되어진다. 센싱된 데이터는 각 노드의 계산되어진 공유키로 암호화되어 베이스노드로 전달된다.

$$K_v = F_{KV}(ID_v) \tag{2}$$

2.4.2 예비경로 설정

센싱데이터 패킷은 중간에 공모노드에 의해 misdirecting되거나 소실되어지기 때문에 본 논문에서는 각 데이터 패킷을 주변노드로 플러딩시켜 베이스 노드로 재전송되어지도록 하였다. k는 데이터를 전달할 주변노드의 개수고 d는 공모노드가 네트워크상에 존재할 확률이라고 할 때 베이스노드로 재전송되어지는 데이터패킷의 개수 n은 식 (3)과 같이 쓸 수 있다. 식 (3)에서 L_{max}는 베이스노드로부터 가장 멀리 떨어진 노드의 홉 거리를 나타낸다.

$$n = \sum_{l}^{L_{max}} (1-p)(1-p^k)^{l-1} t^l \tag{3}$$

2.4.3 싱크홀 공격노드 검출

싱크홀 공격범위에 있는 다수의 악의를 가진 공모노드들은 노드의 홉 카운트 값을 수정하여 전송경로를 변경하여 베이스노드가 공격노드를 정확하게 찾아낼 수 없도록 한다. 공모노드들은 실제 공격노드인 싱크홀 노드 대신에 주변의 임의의 노드를 공격노드인 것처럼 보이게 하도록 홉 카운트 값을 수정한다. 실제 공격노드를 감추기 위한 공모노드들의 거짓 공격노드 생성을 그림 4에 나타내었다.

그림 4에서 보면 싱크홀 공격노드주변의 공모노드들은 패킷을 원래의 싱크홀 노드에 전달하지 않고 거짓 공격노드인 Fake노드에 전달함으로써 베이스노드가 원 싱크홀 노드를 식별하지 못하도록 한다. 또한 공모노드에 의해 실제 cost값이 감소되지 않고 임의로 수정되어 Fask노드로 전송되는 것을 볼 수 있다. 하지만 Fake노드로 들어오는 cost값이 일정하지 않기 때문에 베이스

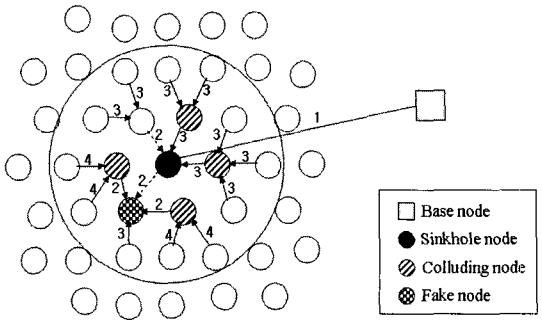

그림 4. 공모노드들에 의한 거짓 공격노드 생성
Fig. 4. Creating fake intruder node by colluded nodes.

```
Procedure Find real_intruder()
  For each root r
    initialize Hopcnt
    CheckHopCount(r,count,1)
  End

  checkHopCount(Node r, Array Hopcnt, int depth)
    depth=depth+1
    for each precedent Node c of r
        increase Hopcnt[hopcount(c)-depth] by 1
        checkHopCount(c,count,depth)
    end for
  end checkHopCount
```

그림 5. 깊이값을 이용한 실제공격노드의 추정 알고리즘
Fig. 5. Real intruder estimating algorithm using depth value.

노드는 현재 싱크홀 공격노드로 의심되는 노드가 실제 공격노드가 아니라는 것을 알 수 있으며 공격노드주변에 공모노드가 있다는 것을 추측할 수 있다. 이러한 공모노드를 식별하기 위하여 베이스 노드는 각 노드별로 실제 데이터 전송트리에서의 이전 깊이값과 노드의 cost값의 차를 Hopcnt배열에 저장한 후 배열성분이 가장 큰 노드값을 실제 공격노드로 의심되는 노드라고 추정한다. 그림 5에 깊이값과 홉 카운트값의 차이를 이용한 실제 공격노드의 추정 알고리즘을 나타내었다.

그림 6은 표 1의 알고리즘에 의한 공격의심 노드의 산출을 나타내었다.

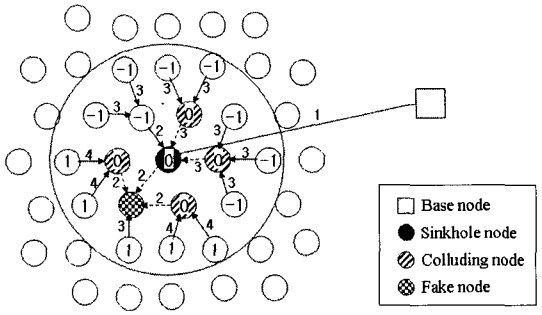

그림 6. 공격노드 추정 알고리즘을 이용한 Hopcnt값 계산
Fig. 6. Calculating Hopcnt value using the intruder node estimating algorithm.

표 1. 공격노드 추정알고리즘에 의해 계산된 Hopcnt 값
Table 1. Calculated Hopcnt value using intruder node estimating algorithm.

v	-2	-1	0	1	2
Hopcnt[v]	0	8	5	5	0

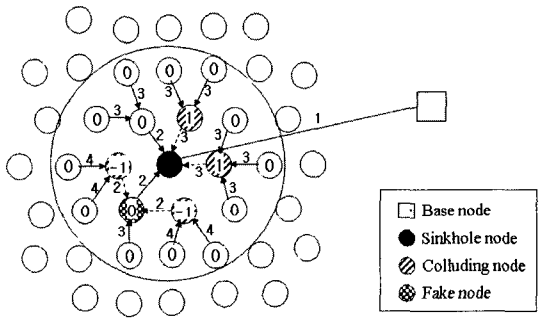

그림 7. 실제공격노드검출 후 카운트 계산된 카운트값
Fig. 7. Recalculated Hopcnt using Hop-depth algorithm.

표 2. 정정된 공격노드에서 계산된 Hopcnt
Table 2. Recalculated Hopcnt at corrected intruder node.

v	-2	-1	0	1	2
Hopcnt[v]	0	2	14	2	0

그림 6의 각 노드위에 쓰여 있는 값은 그림 5의 알고리즘에 의한 Hopcnt배열의 계산값이다. 실제 싱크홀 공격노드와 공모노드의 홉 카운트 값만 0으로 계산되었고 나머지 노드값은 1이나 -1로 계산되었다. 계산된 Hopcnt배열값을 표 1에 나타내었다.

표 1을 보면 계산된 Hopcnt배열값이 각각 -1에서 1 값 사이에 분포하고 있다. 이것으로 베이스 노드는 실제 공격노드가 현재 공격노드로 의심되는 노드의 ±1홉 거리내에 위치한다는 것을 알 수 있다. 표 1에서 계산된 Hopcnt값이 가장 높게 분포된 곳은 -1이다. 이것은 실제 공격노드가 현재 의심노드에 비해 1홉 더 가까운

```
Procedure Find real_intruder()
BEGIN
 For each root r
   initialize Hopcnt
   initialize correct Path
   CheckHopCount(r,count,1)
   S={x>0|forall y>0, count[x]+count[-x] >
      count[y]+count[-y]}
   x=minimum(S)
   corrrectRoot(r,r,x,0,corrrectPath on Network G
   applycorrectPath on Network G
 end fo
END

 CorrectRoot(Node r,Path p, int sumLevel,
   int currentLevel, Path correctPath, int
bestCount)
   if(currentLevel >= sumLevel)
      return
   end if
   currentLevel = currentLevel+1
   for each ahead node c of r
      initialize Hopcnt
      reverse edge(c,r)
      checkHopCount(c,count,1)
         if(count[0] > bestCount)
            correctPath = p->c
            bestCount = count[0]
         end if
        correctRoot(c,p->c,          sumLevel,
currentLevel
                    , correctPath, bestCount)
        reverse edge(c,r)
   end for
 end correctRoot
```

그림 8. Hop-depth 알고리즘을 이용한 실제 공격노드 식별
Fig. 8. Real intruder detecting using Hop-depth algorithm.

곳에 위치한다는 것과 그 위치에서 공모노드들이 홉 가중치를 조작하여 전송하였음을 의미한다. 이 결과값을 가지고 실제 공격노드를 감지하는 Hop-Depth 정정알고리즘을 그림 8에 나타내었다. 홉 카운트 값을 재수정

한 결과 그림 7과 같이 패킷전송방향이 수정되며, 수정된 후의 Hopcnt배열값을 표 2에 나타내었다. 정정된 후의 배열값은 0의 위치가 14의 값을 가짐으로써 현재 패킷전송이 집중되고 있는 노드가 싱크홀 공격노드임을 나타내며 베이스 노드는 공격노드를 네트워크상에서 제거시킬 수 있게 된다.

2.4.4 공모노드 검출

제안된 Hop-depth 알고리즘을 사용하면 실제 싱크홀 공격노드의 위치를 찾아낼 수 있으며, 네트워크상에서 제거가 가능하다. 하지만 공모노드들에 의해서 여전히 misdirecting과 데이터패킷의 소실이 일어날 수 있기 때문에 초기 설정된 데이터 전송트리의 유니캐스트한 경로로 데이터가 전송되지 않고 다른 전송루트를 통해 지연되어 전달되기 때문에 결과적으로 전체네트워크의 에너지 소비량이 증가하게 된다. 따라서 주변의 공모노드를 검색해낼 수 있는 알고리즘도 고려되어져야 한다. 이에 대하여 본 논문에서는 각 노드에서 베이스노드로 전송되어지는 역승인신호인 Ack에 기인하여 중간 공모노드를 찾아내는 알고리즘을 사용하였다.

소스노드는 공모노드를 찾기 위하여 report패킷, Ack패킷, alarm패킷을 사용한다. 여기서 alarm패킷은 공모노드로 의심되는 노드가 감지되었을 때 소스노드로 보내어진다. 어떤 노드에서 전송하고자 하는 센싱데이터가 있을 때 노드는 베이스 노드방향의 다음 홉 노드에 report패킷을 전달한다. 이때 report패킷의 Ack_count는 미리 정의된 카운트값인 Ack_Start로 초기화된다. 전송경로 중간에 있는 각 노드들은 report패킷을 받을 때마다 report패킷을 각 노드에 저장하며, Ack_count필드값을 1씩 감소시킨다. 만약 전송되는 도중 Ack_count필드가 0이 되면 다시 Ack_Start값으로 초기화시킨 후 Ack_count값이 0이 되었음을 알리기 위해 베이스노드로 보낼 Ack패킷을 생성한다. 이때 Ack패킷의 TTL필드를 미리 정의해둔 Ack_TTL값으로 초기화 시킨 후 TTL값이 0이 될 때까지 원래 repor패킷이 전달되어진 경로의 역방향으로 Ack패킷을 전송한다. 처음 Report패킷을 보낸 노드는 Ack패킷이 오기를 기다리며, Ack패킷이 T_{ack} 시간내에 전송되어 오지 않으면 전송경로의 바로 다음에 위치하는 이웃노드를 공모노드로 의심하고 alarm패킷을 생성한다. alarm패킷 중 DstID는 원래 report패킷이 전달되어진 소스노드의 ID를 의미하며 Lost_PacketID_begin과 Lost_PacketID_end는 중간에 잃어버린 패킷의 ID를 의미한다. 이때 공모노드의

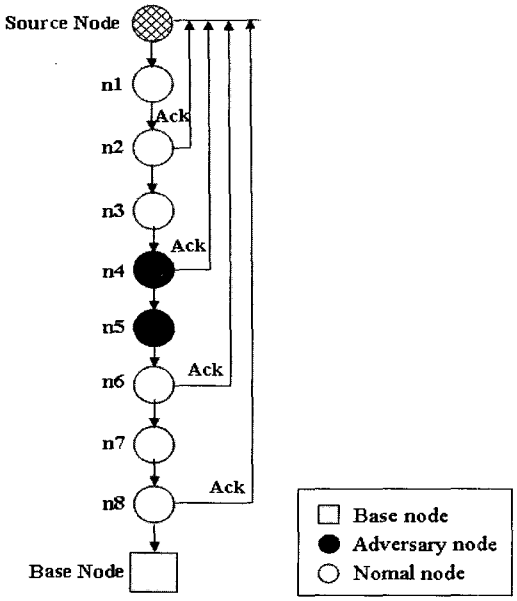

그림 9. Ack신호에 의한 공모노드의 검출
Fig. 9. Detecting malicious node using Ack signal.

replay공격에 대비하여 타임스탬프필드가 현재의 시스템시간으로 설정되며, 의심되는 노드ID가 Suspicious_NodeID필드에 설정된다. 이렇게 생성된 alarm패킷은 주변노드로 플러딩되어 원래의 소스노드로 전달되어진다.

결국 일정 홉 간격으로 Ack패킷이 소스노드로 전달되며, 소스노드는 이 Ack패킷의 전달유무로 데이터가 어디까지 제대로 전달되었는지를 판단할 수 있고 Ack패킷이 전달된 노드를 이용하면 유효하지 않는 공격의심노드를 제거시킬 수 있다. 그림 9에 Ack와 Alarm패킷을 이용한 공모노드검출을 나타내었다. 그림 9에서 소스노드는 report패킷을 보낸 후 Ack 신호가 돌아오기를 기다린다. report패킷이 전송되면서 Ack_count 필드값이 감소하여 주기적으로 Ack패킷이 돌아오게 되는데 이때 돌아오는 Ack패킷의 유효성을 판단하면 데이터 패킷의 소실이 이루어지는 노드의 위치를 추측해낼 수 있다.

report패킷의 Ack_count필드에 설정되어지는 Start값이 작거나 Ack_TTL필드값이 클수록 공모노드의 공격에 대한 강인성이 증가하나 전체 네트워크에 걸리는 오버헤드가 증가하기 때문에 전체 네트워크의 에너지소비와 이들 필드는 상보적 관계(trade-off)를 가지게 된다.

III. 실험 및 결과

본 논문에서는 제안된 공격검출 알고리즘의 성능을 측정하기 위하여 공격노드 식별률, 통신 오버헤드, 에너

표 3. 실험 변수
Table 3. Experiment parameter.

네트워크의 크기	200m × 200m
총 노드수	400
노드당 전송반경	10m
베이스 노드의 위치	100,100
싱크홀공격노드의 위치	50,50
공모노드의 분포확률	0~50%
데이터 패킷 소실율(p)	0~80%
패킷 크기	100bytes

지 소비량을 분석하였다. 성능평가는 NS2를 이용하여 이루어졌으며 실험변수는 표 3과 같이 설정하였다. 총 400개의 노드중 50개의 노드가 싱크홀 공격노드에 의하여 공격받고 있으며, 메시지 패킷은 주변의 공모노드들에 의해 베이스 노드가 공격노드를 식별할 수 없도록 공격노드주변의 다른 노드로 Misdirecting 되어지거나, 임의의 확률 p의 비율로 중간에 소실되어진다.

3.1 원 공격노드 식별율

그림 10에 싱크홀 공격노드의 식별률을 나타내었다. 실험결과 중간에 데이터가 소실되어지는 확률값 p가 0이면 공모노드의 개수에 상관없이 원래의 공격노드를 정확하게 찾아냄을 알 수 있었다. 하지만 데이터 소실확률 p가 증가하고 공모노드가 많아짐에 따라 원 공격노드 식별률이 떨어지는 것을 확인할 수 있었다.

공격노드 검출시 식별된 노드의 오판율을 분석하기 위하여 양성오류율과 음성오류율을 구하였다. 양성오류율은 실제 공격노드가 아니지만 공격노드라고 오판하는 비율이며, 음성오류율은 실제 공격노드인데도 불구하고 공격노드가 아닌 것으로 오판하는 비율이다. 그림 11과

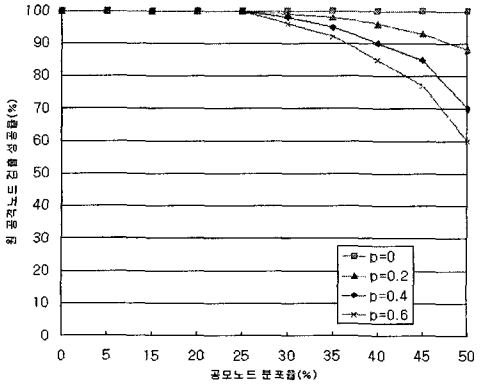

그림 10. 노드수에 따른 공격자 식별률
Fig. 10. Detecting rate in identification process.

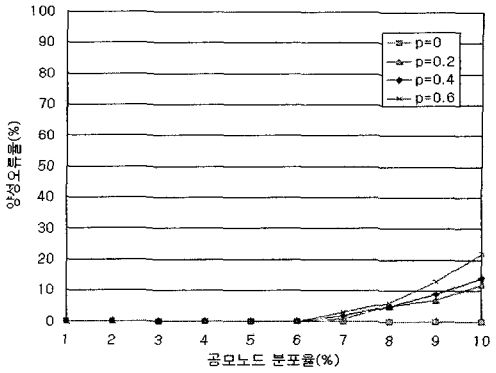

그림 11. 데이터 소실확률에 따른 양성오류율
Fig. 11. False-positive rate in identification process.

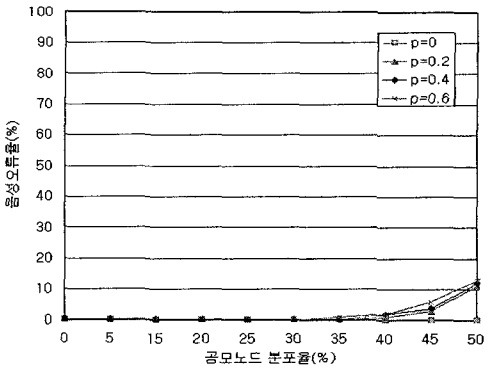

그림 12. 데이터 소실확률에 따른 음성오류율
Fig. 12. False-Negative rate in identification process.

12에 각각 양성오류율과 음성오류율을 나타내었다. 실험결과 데이터소실확률 p가 0일 때 노드오판율은 0으로 정확하게 공격노드를 식별해냄을 알 수 있었다. 하지만 공모노드가 증가하고 데이터소실확률 p가 증가함에 따라 베이스 노드로 잘못된 정보가 전달되어지기 때문에 오판율이 증가함을 알 수 있었다.

3.2 에너지 소비량

센서네트워크에서 전체 에너지 소비량은 데이터 패킷의 트래픽에 의한 네트워크의 오버헤드와 데이터의 전송 및 인증 시 걸리는 오버헤드로 나눌 수 있다. 표 4에 실험값으로 사용한 네트워크 에너지량을 나타내었다.[9]

본 논문에서는 각 노드에 대한 에너지 소비량을 계

표 4. 에너지 소비 파라미터
Fig. 4. Energy consumption parameter.

회로의 에너지 소비량	5×10^{-8} j/bit
안테나 파워	1×10^{-10} j/bit/m^2
MAC 암호화시 에너지 소모	3×10^{-9} j/bit

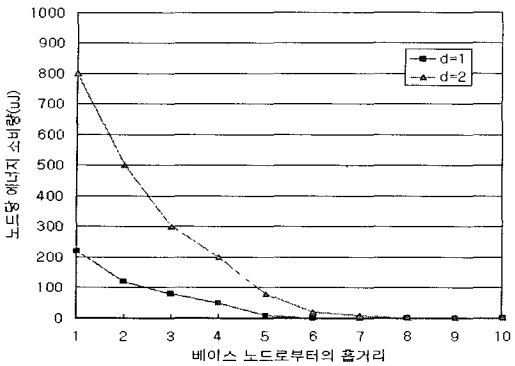

그림 13. 베이스노드로부터 홉거리에 따른 에너지 소비량

Fig. 13. Energy consumption when the hop-distance increase.

산하기 위하여 표 4의 에너지 변수값을 사용하였다. 그림 13은 베이스 노드로부터 떨어진 거리에 따른 에너지 소비량을 나타내었다. 그림 13에서 보면 베이스노드로부터 가까운 홉 거리에 위치할수록 베이스노드와 주변 네트워크사이는 좁은 통로와 같이 되어 데이터 전송률이 높아지기 때문에 에너지 소비량이 증가함을 알 수 있었다.

Ⅳ. 결 론

본 논문에서는 센서 네트워크상에서 외부공격자에 의해 싱크홀 공격이 일어났을 때 실제 싱크홀 공격노드와 주변의 공모노드들에 의한 가짜 공격노드를 가려내고 주변의 공모노드를 식별해 낼 수 있는 알고리즘을 제안하였다. 실제 공격노드를 가려내기 위해서 베이스노드로부터의 실제 홉 거리와 거짓으로 전송된 홉 카운트값의 차이값을 배열에 저장함으로써 거짓노드를 판별한 후 실제 공격노드의 위치를 식별해 내었으며, 싱크홀 주변의 악의를 가진 공모노드는 일정홉 간격으로 전송 성공유무를 알리는 Ack패킷을 전송하도록 함으로써 어떤 노드에서 데이터의 소실이 일어나는지를 식별해 센서 네트워크상에서 제거해낼 수 있도록 하였다. 제안된 프로토콜의 성능을 테스트하기 위해 NS2를 사용하였으며, 싱크홀 공격노드의 식별시간을 측정하였다. 하지만 본 논문에서 제안된 알고리즘은 내부공격에 대한 강인성은 증가한 반면 주기적으로 Ack패킷을 데이터에 추가하여 매 주기마다 소스노드로 재전송해야 하기 때문에 상대적으로 에너지 효율이 감소한다. 그러므로 앞으로 에너지효율을 증가시킬 수 있는 알고리즘의 개발이 필요하다.

참 고 문 헌

[1] Hongmel Deng, Wel LI, and Dharma P. Agrawal, "Routing Security in Wireless Ad Hoc Networks," In IEEE Communication magazine., pp. 70-75, Oct. 2002.

[2] Dorothy E. Denning, "An Intrusion Detection Model," in Proceeding of the IEEE Symposium on Security and Privacy., pp.118-131, 1986.

[3] Y.Zhang and W. Lee, "Intrusion Detection in Wireless Ad-hoc Networks," in proceeding of the 6th ACM MobiCom., pp. 275-283, 2000.

[4] C. Karlof, and D. Wagner, "Secure Routing in Wireless Sensor Networks : Attack and Countermeasures," In Elsevier's AdHoc Networks Journal, Special Issue on Sensor Network Application and Procotols., pp. 293-315, Sep. 2003.

[5] L. Lazos, R. Poovendran, C. Meadows, P. Syverson, and L. W. Chang , "Preventing Wormhole Attacks in Wireless Ad Hoc Networks," In IEEE Wireless and Communications and Networking Conference (WCNC)., pp. 1193-1199, Mar. 2005.

[6] J. Deng, R. Han, S. Mishra, "Security Support In-Network Processing in Wireless Networks," In Proceeding of 1st ACM Workshop on Security of Adhoc and Sensor Networks., 2003.

[7] D. Wagner, "Resilent Aggregation in Sensor Networks," in ACM Workshop on Security of Ad Hoc and Sensor Networks(SASN), pp. 78-97, Oct. 2004.

[8] N. Ye and Q. Chen, "An Anomaly Detection Technique Based on a Chi-square Statistic for Detecting Intrusions into Information System," Quality and Reliability Engineering International, vol. 17, no. 2, pp. 105-112, 2001.

[9] W. B. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for wireless Microsensor Networks," IEEE Transactions on Wireless Communications, vol. 1, no. 4, pp. 660-670, 2002.

저 자 소 개

이 강 현(평생회원)
1979년, 1981년 조선대학교 전자공학과 공학사 및 석사
1991년 아주대학교 대학원 공학박사
1977년~현재 조선대학교 교수
1991년, 1994년 미 스탠포드대 CRC 협동연구원.
1996년 호주 시드니대 SEDAL 객원교수
2000년~현재 한국 멀티미디어기술사협회 이사
2002년 영국 런던대 객원 교수
2002년 대한전자공학회 멀티미디어연구회 전문위원장
2003년 한국 인터넷 방송/TV 학회 부회장
2003년~현재 대한전자공학회 상임이사
2005년~현재 조선대학교 RIS 사업단장
<주관심분야 : 멀티미디어 시스템설계, Ubiquitous convergence>