情報保護學會論文誌
第16卷 第1號, 2006. 2

모바일 Ad hoc 네트워크를 위한 신용 평가 보상 프로토콜

축 뢰,† 강 전 일, 양 대 헌‡
인하대학교 정보통신대학원

A Reputation Compensation Protocol For Mobile Ad Hoc Networks

ZHU LEI,† Jeonil Kang, DaeHun Nyang‡
INHA University Graduate School of IT&T

요 약

최근 몇 년간, ad hoc 네트워크 영역에 대한 연구자들의 관심이 높아지고 있다. 또한, ad hoc 네트워크에서 동작하는 많은 라우팅 프로토콜이 제안되었다. 이기적 노드란 자신의 전력을 보존하기 위해, 일반 노드가 해야만 하는 특정한 동작의 절차를 따르지 않는 노드를 의미한다. 이 논문에서는 이러한 이기적 노드가 유발하는 유해한 행동을 경감시키는 기법을 제시한다. 또한, 노드의 행동에 영향을 미칠지도 모르는 새로운 환경을 가정한다. 노드들이 네트워크에서 정상적으로 통신하도록 하기 위하여, 이러한 문제를 해결하는 방법에 대해서 제시한다. 또한 이에 따른 모의실험 결과를 논문에 포함하였으며, 이러한 결과를 가지고 우리는 약간의 트래픽을 희생하여 이러한 문제를 해결할 수 있었음을 보였다.

ABSTRACT

The area of ad hoc networking has been receiving increasing attention among researchers in recent years and a variety of routing protocols targeted specifically at the ad hoc networking environment have been proposed. Selfish nodes are those which do not perform certain operations that the protocol specifies that they should, through a wish to conserve power. We propose a scheme as a mean to mitigate the detrimental effect of selfish nodes. We also propose a new area that might affect nodes' behavior - the environment's influence. In order to let nodes fairly be able to communicate in the network, we proposed solution to this problem. And our scheme can be applied to other reputation methods. We also contain the simulation results in our paper, and, through the result, we can conclude that we can solve the problem by adding a little overhead.

Keywords : *Ad hoc, Reputation*

Ⅰ. 서 론

신용 평가 시스템(reputation system)은 P2P (peer-to-peer) 네트워크에서 효과적 피어의 선출, 온라인 경매 상의 거래 상대 선택, 모바일 ad hoc 네트워크상의 무례한(misbehaving) 노드의 발견과 같은 여러 응용 분야에 쓰인다. 이 시스템은 가용 정보의 사용에 대한 효율성과 오류 등급(false rating)에 대한 견고성 사이의 상호 교환(trade-

접수일 : 2005년 9월 8일 ; 채택일 : 2006년 1월 18일
† 주저자: koudai@seclab.inha.ac.kr
‡ 교신저자: nyang@inha.ac.kr

off) 관계를 가지고 있다. 만약 다른 노드에 의해서 정해지는 오류 등급(false ratings)을 고려한다면, 신용 평가 시스템은 잘못되거나 의도된 오류 보고에 취약할 수 있다. 하지만 오직 자기 자신만의 경험만을 중시한다면, 다른 노드들의 경험을 바탕으로 하는 잠재적인 학습 가능성은 무시 될 것이다. 긍정적이기만 한 정보나, 부정적이기만 한 정보를 사용하는 것은 각각 오류 칭찬(false praise)과 오류 고발(false accusation)의 취약점을 감소시킨다.

Ad hoc 라우팅을 악의적인 노드에 대해 보호하기 위하여 제시되었던 두 가지 신용 평가 방법은 CONFIDANT(Cooperation of Nodes: Fairness in Dynamic Ad-Hoc NeTworks)[14]와 CORE(Collaborative Reputation Mechanism)[9] 프로토콜이 있다. CORE 프로토콜은 과거 행동에 더 많은 가중치를 줌으로써, 공격자가 악의적으로 동작하기 전에 높은 신용을 쌓을 수 있다면 이러한 공격자에게 취약할 수 있다. 어떠한 노드가 이기적으로 동작하기 전에 '신용 쌓기'를 이용한 공격을 한다면, 좋은 행동에 대해서 보상하지 않는 CONFIDANT에서 더 적은 영향을 미친다. 그렇기 때문에 CONFIDANT 프로토콜에서는 모든 노드는 언제나 적대적 행위를 할 것이라는 의심을 받는다. 하지만, 이러한 프로토콜 동작은 CONFIDANT가 전원 고갈과 같은 부정적 행동을 수행하는 노드들에 대해서도 관대하지 않게 만든다.

이 논문에서 소개하는 기법의 목표는 이기적 노드에 대하여 견고하고, 부정적인 움직임을 찾아내는데 효율적인 지역 감시 시스템(neighborhood watching systems)을 만드는 것이다. 긍정적이든 부정적이든, 자신의 경험과 다른 노드들의 경험으로부터 나오는 모든 가용 정보를 이용하는 메커니즘을 제공하며 이러한 신용 평가 시스템을 견고하게 만들기 위하여 오류 등급을 다루는 방법을 포함하고 있다.

II. 신용 평가 방법

이 장에서는 논문에서 제안하고자하는 기법을 위한 몇 가지 가정과 정의를 설명한다.

1. 가정

이 논문에서는 다음과 같은 것을 가정한다.

- 각각의 노드는 고유한 아이디를 가지고 있다.
- 링크는 양방향성이다.
- 노드는 이전에 "신뢰" 관계를 맺지 않는다.
- 모든 노드는 다른 노드에게 올바른 신용 값을 넘겨준다.
- 무례한 노드는 데이터 패킷을 전달하지 않는다. 하지만 다른 모든 것은 올바르게 동작한다.
- 네트워크를 붕괴시키길 원하는 악의적인 노드는 네트워크 내에 존재하지 않는다.

2. 방향 신뢰(direct trust, DT)

어떠한 노드 B를 신뢰할 수 있는지 알고 싶을 때, 우리는 B에게 어떠한 패킷을 전송할 수 있고, 임의의 노드를 스니핑하여 B가 그러한 패킷을 올바르게 보내는지 확인할 수 있을 것이다. 그러면 패킷 총량에 대해 올바르게 전달 된 패킷들의 비율은 얼마나 B를 신용할 수 있는지에 대한 단서를 제공해준다.

어떠한 노드는 자신의 행동을 때때로 바꿀 수도 있기 때문에, 우리는 단지 현재의 행동만을 관심의 대상으로 한다. 이 때문에 B를 통하여 전송한 마지막 x개의 패킷을 위해, B의 행동을 저장할 수 있는 크기 x의 배열을 만들었다. 성공은 "1", 실패는 "0"이다. 만약 배열의 모든 값을 더하고 배열의 개수 x로 나누며 B에 의해서 올바르게 전달 된 패킷의 비율을 얻을 수 있다. 노드 A는 이 값을 B를 위한 방향 신뢰 값으로 사용할 수 있다. $\#forwarded$가 B를 통해 올바르게 전달 된 패킷의 수, $\#sent$를 A로부터 B를 향해 보내지는 패킷의 수라고 하면, 방향 신뢰 값은 다음과 같다.

$$DT(A,B) = \frac{\#forwarded}{\#sent} \quad (1)$$

3. 우회 신뢰(Indirect Trust, IDT)

새로운 노드가 다른 노드의 통신 가능 영역에 출현하였을 때 무슨 일이 벌어질까? 우리는 그 노드를 위해서 새로운 방향 신뢰 값을 만들거나, 다른 노드들에게 어떻게 해야 할지 물어볼 수 있을 것이다. 이럴 경우, 어떻게 그 의사를 물어보고, 무슨 답변을 얻을 수 있으며, 그것들을 어떻게 조합할 것이냐 하는 것이 문제로 남는다.

만약 A가 B를 위한 의사를 물어보았을 경우, A는 신용 참조 요청(reputation request) 메시지

를 만들고, 그 자신을 소스로 두고 B를 타깃으로 하여 그 주위 노드들에게 (TTL=1 로 하여) 브로드캐스트 한다. 이 요청 메시지를 받은 모든 노드들은 자신이 B를 위한 방향 신뢰 값을 가지고 있는지 확인하고, 있다면 신용 참조 응답(reputation reply)를 만들어 A에게 이를 되돌려준다. 얼마 후, A는 이렇게 얻은 값들을 조합하여 B의 신용 평가 값을 만들 수 있을 것이다. 이 우회 신뢰 값은 언제 계산되고, 누군가로부터 얼마나 많은 응답을 받을 수 있는지에 따라 완벽히 달라진다. 신용 참조 응답들에 들어 있는 방향 신뢰 값으로 하나의 우회 신뢰 값을 만들기 위한 한 가지 방법은 응답을 돌려준 노드들의 방향 신뢰 값에 가중치를 두어 결정하는 것이다. N_i를 노드 A의 i번째 주변 노드라고 한다면, 우회 신뢰 값은 다음과 같다.

$$IDT(A,B)=\frac{\sum_{i=1}^{n} DT(A,N_i)\times DT(N_i,B)}{n} \quad (2)$$

다른 가능한 방법은 응답들을 보고 서로를 비교해 보는 것일 것이다. 만약, 올바른 것으로 간주할 수 있는 매우 유사한 값들이 아주 많다면 다른 몇몇 값들은 올바르지 않은 것으로 간주할 수 있다. 그런 다음에 올바른 값들의 평균을 구하여 신용 값으로 사용할 수 있다. 또한 올바른 응답을 준 노드가 누구이고 그렇지 않은 노드가 누구인지 기억한 뒤에 이러한 정보를 앞으로의 응답에 가중치를 주는데 이용할 수 있을 것이다.

하지만 다른 모든 노드들이 0.9라고 대답한다고 해서 0.1이라고 대답한 노드가 정말로 올바르지 않은 값이라고 말할 수 있겠는가? 단지 0.9는 전달 작업이 완벽했음을 알려주고, 0.1은 지연이 매우 늦다고 말하고 있을지도 모른다. 또는 하나의 노드에서 만 패킷이 탈락되고 있음을 알려줄지도 모른다. 이 또한 매우 다른 방향 신뢰 값들의 결과일 수 있다.

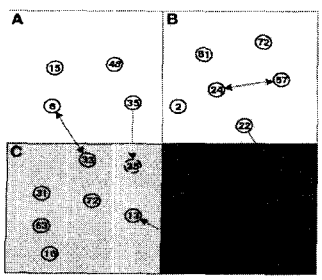


그림 1. 네트워크 모델

만약 어떤 한 노드가 모든 것이 올바르게 동작하거나 모든 것이 올바르게 동작하지 않는다면, 노드들의 응답은 그들이 가지고 있는 방향 신뢰 값과 함께 가중치를 줄 수도 있다. 그래서 첫 번째 방법이 보다 정당해 보인다.

4. 신용 평가

이제, 위와 같은 방법을 통해서 어떠한 방향 신뢰 값과 우회 신뢰 값을 조합한 방법을 생각해보자. ω를 방향 신뢰 값을 위한 가중치라고 했을 때 신용 값은 다음과 같다.

$$REP(A,B)=\omega\times DT(A,B)+(1-\omega)\times IDT(A,B) \quad (0<\omega<1) \quad (3)$$

Ⅲ. 신용 보상 방법

모바일 네트워크를 위해 존재하는 수많은 신용 평가 방법들 중에서 노드들의 행동에 영향을 미치는 환경에 대해 관심을 기울이는 것은 아직 없었다. 예를 들어, 여러 지형에 걸쳐 이루어지는 네트워크를 생각해보라. 각각의 영역은 다른 환경을 가지고 있을 것이다. (언덕 지형과 비교해볼 때 평지가 노드가 통신하기에 더 수월하다.) 만약 이러한 부분을 고려하지 않은 채 모든 노드들에게 동일한 규칙을 부여한다면, 이것은 공정하지 못하다. 부적정한 행동을 보이는 노드뿐만이 아니라 열악한 환경에 의해 유발되는 낮은 성능에 의하여 낮은 신용 평가를 받을 수 있기 때문이다.

이 논문에서는 이렇게 열악한 지역에 있는 노드들에게 신용을 보상해줄 수 있는 방법을 제시한다. 이 방법은 CORE[2]나 CONFIDANT[1]와 같은 다른 프로토콜에도 적용할 수 있을 것이다. 환경이 노드의 행동에 어떠한 영향을 미치는지에 대한 예로써, 신용 평가 방법을 제시한다.

이 방법에서 전체 네트워크는 처한 환경에 따라 여러 영역으로 나눠질 수 있다. 그림 1은 이러한 예를 보여주고 있다. 전체 네트워크는 A, B, C, D로 나누어져 있으며, 각각의 영역 안에서의 노드의 전파 반경과 같은 기타 환경은 모두 같다. 차례로 A,

B, C, D 순으로 노드가 통신하기 쉬운 환경이라고 가정하자. D는 노드가 통신하기 가장 어려운 영역이다. 8번과 33번 노드를 예를 들면, 33번 노드가 더 통신하기 어려운 영역에 속해 있다. 그러면 그들의 의도가 아닌 환경의 영향에 의하여 이러한 8번과 33번 노드의 통신에서 높은 패킷 탈락이 발생하게 된다. 이때, 8번 노드가 33번 노드의 방향 신뢰 값을 계산하면 이는 낮은 값이 나올 것이다. 33번 노드의 신용 값은 한계치보다 낮을 수도 있고 무례한 노드로 인식될 수 있을 것이다. 그렇기 때문에 이러한 신용 보상 방법이 필요하다고 할 것이다.

1. 보상 방향 신뢰(Compensated Direct Trust, CDT)

보상 방향 신뢰 값을 구하는 식은 다음과 같다.

$$CDT(A,B) = \alpha_{A,B} \times \frac{\#\ forwarded}{\#sent} \tag{4}$$

여기서 $\alpha_{A,B}$ 는 노드 A와 B 사이의 방향 신뢰 값을 위한 보상 요소이다.

상황 1. 같은 영역에 존재하는 노드들

그림 1에서 24번과 57번 노드는 같은 영역에 속해 있기 때문에 같은 환경을 공유한다. 이러한 경우 이들의 신용은 보상해줄 필요가 없으며, α값은 1이다.

상황 2. 다른 영역에 존재하는 노드들

이번에는 8번과 33번 노드에 대해서 생각해보자. 이들 노드는 서로 다른 영역에 존재하며 33번 노드가 더 좋지 않은 영역에 존재한다. 이러한 경우 신용에 대한 보상을 생각할 수 있을 것이다. 또한 35번과 22번 노드가 다른 영역으로 이동했을 때도 α 값은 다른 값으로 바뀌게 된다. 노드 A와 B 사이의 $\alpha_{A,B}$ 값은 다음과 같다.

$$\alpha_{A,B} = \frac{avg(A)}{avg(B)} \tag{5}$$

여기서 $avg()$는 어떠한 노드가 속해 있는 영역에 속한 노드들의 평균 신용 값을 나타낸다.

2. 보상 우회 신뢰(Compensated Indirect Trust, CIDT)와 보상 신용(Compensated Reputation, CREP)

α값은 이미 $DT(A,N_i)$와 $DT(N_i,B)$에 더해져 있기 때문에, 우회 신뢰 값과 신용 값에 α를 이용한 보상은 필요하지 않다. 그렇기 때문에 보상 우회 신뢰 값은 와 같고,

$$CIDT(A,B) = \frac{\sum_{i=1}^{n} CDT(A,N_i) \times CDT(N_i,B)}{n} \tag{6}$$

보상 신용 값은 다음과 같다.

$$CREP(A,B) = \omega \times CDT(A,B) + (1-\omega) \times CIDT(A,B) \quad (0 < \omega < 1) \tag{7}$$

Ⅳ. 전체 시나리오

전체 네트워크는 환경에 따라 여러 영역으로 나누어진다. 하나의 노드는 모든 주위 노드들의 목록과 어떠한 기능을 수행하기 위한 그들의 방향 신뢰 값으로 이루어진 방향 신뢰 테이블을 관리한다. 노드들은 임시적으로 자신이 가지고 있는 다른 노드들의 방향 신뢰 값과 자기 자신의 α값을 포함한 DT_{update} 메시지를 이웃 노드들에게 보낸다. 이 메시지를 받으면, 다른 노드들은 보낸 노드의 아이디와 이 노드가 부정한 행동을 하는지 여부를 확인한다. 만약 보낸 노드가 신뢰할 만하다면, 노드들은 이 메시지를 받아들이고 다른 노드들의 우회 신뢰 값을 업데이트하며 다른 노드들의 신용 값을 계산한다. 신뢰 값 rep는 ★ $tdtv$ (start direct trust value)로 초기화 된다.

한 노드가 주위로부터 서비스를 요구할 때, 이웃에게 x번의 응답 기회를 주며, 여기서 x는 ★ $tdtv$와 동일한 값으로 초기화 된다. 만약 응답이 긍정이면 x는 cv(change value)만큼 증가한다. 응답이 긍정적이라면 x의 값은 타임아웃 뒤 초기 시작 값 ★ $tdtv$으로 되돌려져야만 한다. 특정한 횟수의 어떠한 부정적인 행동도 일어나지 않는 연속적인 타임아웃 뒤에는 rep는 cv만큼 증가해야 한다.

응답이 없거나 응답이 부정적이라면, x는 $2cv$만큼 감소한다. 노드는 x가 영(0)이 되기까지 이러한 일을 계속해야만 하며, 만약 x가 0이 되면 해당하는 방향 신뢰 값이 $2cv$만큼 감소한다. 이 과정에서 노드는 다른 노드로부터 서비스를 요구할 수도 있을 것이다. 그 후에 노드가 만약 동일한 이웃 노드들로부터 서비스를 받기를 원한다면 위와 같은 일을 반

복한다. 이 경우 서비스를 요청받은 노드들은 줄어든 x만큼 줄어든 응답 기회가 있을 뿐이다. 노드는 어떠한 노드들에게 다른 (전원이 차단되는 등의) 일시적인 문제들을 해결할 수 있도록 하기 위해서 지수적으로 응답 기회의 증가를 줘야 한다. 따라서 만약 어떠한 한 노드가 다른 선택이 없고 이기적인 노드를 이용해야만 하는 경우, 그 노드는 초기 x를 1로 두고 서비스를 요청하는 수밖에 없을 것이다. 방향 신뢰 값이 감소하는 경우 같이, 이러한 경우는 이기적이거나 실패한 이웃 노드들에게 더 적은 자원을 사용하는 결과를 가져온다. 또한 원하지 않는 행동을 방지하기 위해서, 다른 노드들이 한계 값 이하의 신용 값을 가지고 서비스를 요청한다면 이는 무시되어야 한다.

1. 어떤 노드가 무례한 일을 하는가?

우선, 다른 종류의 무례한 일을 구별할 수 없다는 것을 확실히 해야 한다. 어떠한 한 노드가 악의적이거나, 그냥 이기적이거나, 남아있는 배터리가 없어서, 또는 다른 이유로 무례한 일을 하는지 알 수 없다는 것이다. 단지 다음과 같이 어떠한 노드가 무례한 일을 하는지 결정할 수 있을 뿐이다.

다음 절에서는 신뢰 값(trust value)을 계산할 것이다. 하지만 이를 어떻게 쓰고, 다른 노드들이 패킷을 라우팅 하는 동안 언제 노드를 신뢰할 수 있는지는 다른 문제이다.

기본적인 아이디어는 무례한 노드를 네트워크에서 배척시키는 것이다. 어떠한 노드도 자신의 패킷이 목적지에 올바르게 도착하는지 알 수 없기 때문에 그 무례한 노드를 통해서 보내려고 하지 않을 것이다. 하지만 무례한 노드를 통해서 아무도 패킷을 보내지 않는다면, 그 무례한 노드들은 다른 노드들이 보내는 패킷을 전달하는데 추가적으로 발생하는 부담(burden)으로부터 자유로울 것이고, 그렇기 때문에 그 무례한 노드들은 그들의 무례한 행동에 대해서 보상받게 된다. 이미 앞서 제안된 많은 프로토콜은 이렇게 작동하지만 이 논문에서는 이러한 무례한 행동을 독려하지 않으며 협력(cooperation)을 강제한다. 이것은 다른 노드들에 의해서 무례한 노드의 패킷을 폐기함으로써 가능하다. 이렇게 하면 무례한 노드는 네트워크에서 완벽히 배척된다. 하지만 무례한 노드에게 자신의 무례한 태도를 바꿀 기회를 주었으면 하기에, 이 논문에서는 그들을 통해서 패킷의 일부를 보내려고 하지만(때문에 그들의 무례한 행동을 관찰할 수 있어야 한다.) 그들을 위한 패킷은 전달 해주지 않으려고 한다.

그렇다면 어떻게 노드가 무례한 일을 하는지 알 수 있을까. 만약 한 노드가 패킷을 버림으로써 패킷이 목적지에 도달하지 않는다거나 올바른 포워딩을 관찰할 수 없다면 신뢰 값은 작아질 것이다. 패킷의 전달에는 노드가 작은 신뢰 값을 가지고 있는 이유는 신경 쓰지 않는다. 그렇기 때문에 목적지에 전달될 가능성이 가장 높은 노드를 고른다. 다른 경우 무례한 노드의 패킷만을 버려야 한다.

이 모든 과정에서 100%에 도달 할 수 없을 것이기에, 오류만 최소화할 수 있어야 한다. 그렇기 때문에 한계치가 필요하다. 그러나 좋지 않은 환경을 보상하기 위하여 α값을 사용하기 때문에 전체 네트워크는 동일한 한계치를 사용할 수 있다. 그러한 한계치를 τ라고 했을 때 $CREP < \tau$ 인 모든 노드들은 무례한 노드로 인식될 것이다.

2. 병목(bottleneck) 문제

비교적 안정적으로 목적지로 향하는 길을 찾기 위하여 이 논문에서는 신용 평가 시스템을 사용한다. 하지만 만약 어떠한 한 노드가 높은 신용 값을 가지고 있고 모든 노드들이 그 것을 통해 패킷을 보내고자 한다면 혼잡(congestion)이 발생할 것이고, 그 노드는 네트워크의 병목(bottleneck)이 될 것이다. 그 길은 안전하지만 더 이상 효율적이지 않을 수 있다.

이 논문에서는 노드를 고르는데 다음과 같은 규칙을 사용한다.

- PDR_x : x번 노드의 패킷 폐기 비율
- $avg(PDR_x)$: x번 노드가 속한 영역의 평균 패킷 폐기 비율
- REP_x : x번 노드의 신용 값
- $avg(REP_x)$: x번 노드가 속한 영역의 평균 신용 값이라고 할 때, 만약 $\frac{PDR_x}{avg(PDR_x)} > \frac{REP_x}{avg(REP_x)}$ 라면, 이 노드에게 많은 대역폭을 할당하지 않으며, 그렇지 않고 $\frac{PDR_x}{avg(PDR_x)} \leq \frac{REP_x}{avg(REP_x)}$ 라면, 이 노드에게 더 많은 대역폭을 할당한다.

표 1. 모의실험 파라미터

파라미터	레벨
영역	1000m × 1000m
노드 분포	균일
응용 계층 프로토콜	CBR
노드 수	100
노드 최고 속도	50m/s
패킷 사이즈	128Byte
정지 시간	0
이기적 노드의 비율	20%
가중치(ω)	0.5
한계치(τ)	0.4

V. 시뮬레이션 및 성능 분석

이 논문에서 제안하는 프로토콜을 평가하기 위하여, 모의실험을 수행하였다. NS2[16]는 DSR 프로토콜이 이미 구현된 네트워크 시뮬레이터이기 때문에 모의실험은 이 DSR을 사용하여 수행하였다.

1. 정의

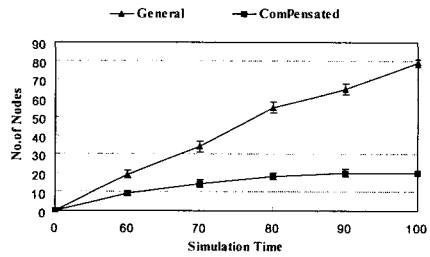
- Original DSR : 신용 평가 시스템을 사용하지 않는 원래의 DSR 프로토콜
- General : 신용 평가 방법을 사용한 DSR 프로토콜
- Compensated : 신용 보상 방법을 사용한 DSR 프로토콜
- Goodput : 보낸 패킷 대 받은 패킷 비율
- Overhead : 라우팅 메시지 대 신용 평가 메시지의 비율

2. 파라미터

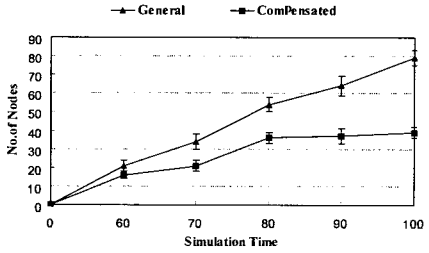
모의실험의 결과는 별도의 내용이 없다면 표 1과 같은 파라미터를 갖고 수행되었다.

3. 모의실험 결과

그림 2는 모의실험 시간의 증가에 따라 무례한 노드로 분류되는 노드들의 수를 보여주고 있다. 그림 2(a)에서 20개를 그림 2(b)에서는 40개를 이기적인 노드로 두었다. 그림에서 보다시피 신용 보상 방법은 신용 평가 방법에 비해서 우수하다는 것을 알 수 있다. 또한 신용 보상 방법은 다른 좋은 노드를 불합리하게 이기적인 노드로 처리하지 않고 모든 이기적인 노드만을 잡아낼 수 있었다. 하지만 신용 평가 방법은 거의 80%의 노드를 이기적인 노드로 분류하였는데, 왜냐하면 신용 평가 방법에서는 좋지 않은 영역에 존재하는 노드들에게 보상을 하지 않기

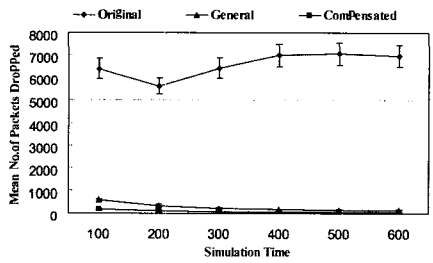


(a) 이기적인 노드 수 : 20

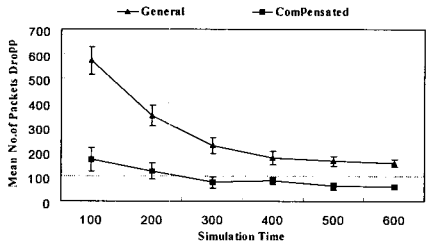


(b) 이기적인 노드 수 : 40

그림 2. 시간의 증가에 따른 이기적인 노드로 분류되는 노드 수의 비교



(a) 세 프로토콜의 비교



(b) (a)에서의 아래 두 프로토콜의 비교

그림 3. 시간의 증가에 따른 버려지는 패킷의 평균량

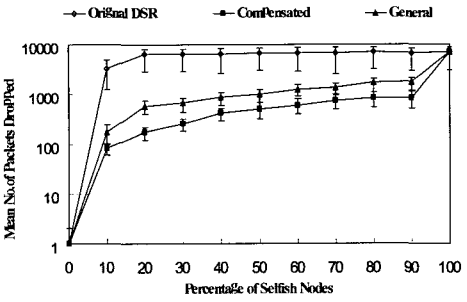


그림 4. 이기적인 노드 수의 증가(%)에 따른 버려지는 평균 패킷 수

때문이다. 그래서 다른 노드와 통신을 할 때, 이기적인 노드로 여겨지는 것이다.

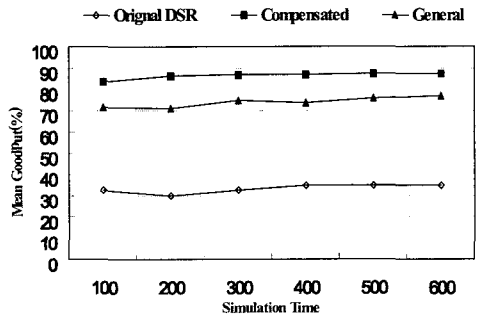
그림 3은 시간의 증가에 다른 폐기 패킷의 평균을 구한 것이다. 원래의 DSR 프로토콜에서 약 7000 패킷이 이기적인 노드 때문에 버려지는 것을 볼 수 있다. 일반적이거나 보상을 해주는 신용 평가 기법에서는 이 원래의 DSR 프로토콜보다 훨씬 좋은 결과를 얻어냈다. 왜냐하면 이러한 신용 평가 기법에서는 이기적인 노드를 효율적으로 찾아낼 수 있었기 때문이다. 그림 3(b)는 그림 3(a)에서 신용 평가 방법과 신용 보상 방법을 비교해놓은 것이다. 그림에서 보다시피 신용 보상 방법이 신용 평가 방법보다 좋은 성능을 보임을 알 수 있는데, 왜냐하면 신용 보상 방법에서 더 적은 수의 노드만을 이기적인 노드로 처리하기 때문이다.

그림 4는 이기적인 노드 수의 증가에 따른 평균 폐기 패킷 수를 나타낸다. 그림에서 보다시피 원래의 DSR에서는 적은 수의 이기적인 노드의 수라고 할지라도 많은 수의 패킷이 버려짐을 알 수 있다. 또한 이기적인 노드가 증가한다고 하더라도 버리지는 패킷의 수의 변화는 크지 않음을 알 수 있다. 이것은 경로 상에서 패킷을 잃어버리는 곳이 어디인지 상관없다는 사실로 설명할 수 있다. 신용 보상 방법에서는 절반 이상이 이기적으로 행동하는 (그러나 이기적인 노드 주변으로 안전한 다른 경로를 위해서 충분한 수의 노드가 있는) 매우 적대적인 환경에서조차 전달과정에서 버려지는 패킷의 수가 낮은 상태를 유지한다는 것을 알 수 있다.

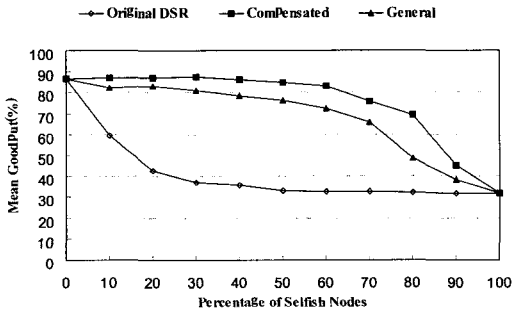
그림 5(a)는 모의실험의 변화에 따른 평균 양호 동작 비율을 보여준다. 원래의 DSR 기법은 매우 안 좋은 성능을 보이며, 양호한 동작 비율 또한 30~40%정도에 머물고 있다. 신용 평가 방법은 조금 더 좋은 성능을 보이며, 양호한 동작 비율이 70~80%에 이르고 있다. 그리고 신용 보상 방법은 이중에서 가장 좋은 성능을 보이고 모의실험 끝에서는 90% 가까운 비율을 보이고 있다.

그림 5(b)는 이기적인 노드의 증가에 따른 평균 양호 동작 비율을 보여준다. 보다시피 신용 보상 방법은 매우 좋은 성능을 보이고 있지만 원래의 DSR 기법은 이기적인 노드가 증가하자 급격하게 양호한 동작 비율이 떨어지며 30~40% 사이에서 안정화하고 있음 알 수 있다. 하지만 신용 보상 방법은 절반 정도의 이기적인 노드 비율까지는 안정적인 모습을 보여주고 있다.

그림 6은 모의실험 시간의 변화에 따른 평균 오버헤드를 보여준다. 새로운 프로토콜을 추가할 때 이에 따른 오버헤드는 너무 크지 않아야 한다. 신용 보상 방법은 15% 미만의 오버헤드가 필요하지만 평균 양호한 동작 비율에서 50%이상의 이득을 발생하기 때문에 신용 보상 방법을 추가할 가치가 있다 하겠다.

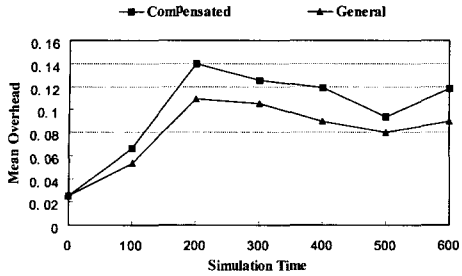


(a) 시간의 증가 (0-600초)

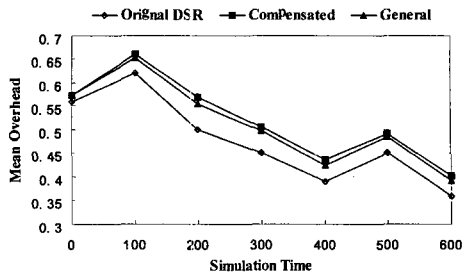


(b) 이기적인 노드의 증가

그림 5. 시간의 증가와 이기적인 노드의 증가에 따른 평균적인 양호 동작 비율(Goodput) 변화



(a) 전체 패킷 중 신용 패킷의 비율



(b) 라우팅 오버헤드 + 신용 보상 기법 오버헤드

그림 6. 시간의 증가에 따른 평균 오버헤드 비교

Ⅵ. 결 론

모바일 ad hoc 네트워크는 많은 중요한 보안 문제, 특히 노드 간 라우팅에 관련된 문제가 있다. 또한 네트워크 내부의 노드들을 대상으로 한 수많은 공격방법이 존재한다. 신용 평가 시스템은 신뢰를 구축하고 신용할만한 행동을 하도록 하는데 사용된다.

이 논문에서 이기적인 노드 때문에 발생하는 유해한 영향을 감소하기 위한 일반적 신용 평가 방법을 제시하였고, 더 나아가 이 신용 평가 방법에 노드가 처한 환경에 따른 영향을 최소할 수 있는 방법을 더했다. 또한 신용 기반의 신뢰 관리 기법을 사용한 DSR 프로토콜을 사용하여 적은 오버헤드만을 요구하지만 많은 성능이 향상됨을 보였다.

우리의 프로토콜은 여전히 몇몇 문제점 - 예를 들면, 공격자에 의하여 신용 값들이 변화 될 수 있다 - 을 가지고 있지만, 앞으로 이러한 문제를 해결한 다른 프로토콜을 기대할 수 있을 것이다. 또한 여기에 더해 위조 공격에 대비하여 패킷들을 암호화하거나 서명하는 방법과 노드의 방향 신뢰 값 정의를 위하여 다른 규약이 연구되어야 할 것이다.

신용 값을 정량화하는 것은 매우 힘든 일이며, 매우 정교한 방법으로 정의되어야만 할 것이다. 신뢰할 수 있고 안전한 방법을 통하여 신용 정보를 분배하는 것은 ad hoc 네트워크에서 특히나 어렵다. 그래서 우리의 앞으로의 연구는 어떻게 신용 값의 계산과 갱신을 더 정교하게 수행할지, 그리고 그것을 어떻게 안전하게 분배할지에 초점을 맞출 것이다.

참 고 문 헌

[1] 권정옥, 황정연, 김현정, 이동훈, 임종인, "일방향 함수와 XOR을 이용한 효율적인 그룹키 관리 프로토콜 : ELKH", 한국정보보호학회, 정보보호학회논문지 제12권 6호, 2002

[2] 박영호, 이경현, "이동네트워크 환경에서 그룹키 관리구조", 한국정보보호학회, 정보보호학회논문지 제12권 2호 pp. 77-89, 2002

[3] 박영희, 정병천, 이윤호, 김희열, 이재원, 윤현수, "Diffie-Hallman 키 교환을 이용한 확장성을 가진 계층적 그룹키 설정 프로토콜", 한국정보보호학회, 정보보호학회논문지 제13권 5호, 2003

[4] 이상원, 천정희, 김용대, "Pairing을 이용한 트리 기반 그룹키 합의 프로토콜", 한국정보보호학회, 정보보호학회논문지 제13권 3호, 2003

[5] 조태남, 이상호, "(2,4)-트리를 이용한 그룹키 관리", 한국정보보호학회, 정보보호학회논문지 제11권 4호, pp. 64-77, 2001

[6] IETF MANET Working Group Internet Drafts. http://www.ietf.org/ids.by.wg/manet.html.

[7] J. Broch, David B. Johnson and David A. Maltz, "The dynamic source routing protocol for mobile ad hoc networks", Internet-Draft Version 03, IETF, October 1999.

[8] L Zhou and Z.J Haas, "Securing ad hoc networks", IEEE Network Magazine, vol.13, no.6, November December 1999.

[9] Pietro Michiardi and Refik Molva, "CORE: A Collaborative Reputation mechanism to enforce node cooperation in Mobile Ad Hoc Networks",

Proceedings of the IFIP TC6/TC11 Sixth Joint Working Conference on Communications and Multimedia Security: Advanced Communications and Multimedia Security, p.107-121, September 26-27, 2002.

[10] Pirzada, A. A. and McDonald, C, "Establishing trust in pure ad-hoc networks", ACM International Conference Proceeding Series, Vol. 56. Proceedings of the 27th conference on Australasian computer science - Volume 26.

[11] Prashant Dewan, Partha Dasgupta and Amiya Bhattacharya, "On Using Reputations in Ad hoc Networks to Counter Malicious Nodes", QoS and Dynamic Systems in conjunction with IEEE ICPADS Newport Beach, USA, 2004.

[12] S. Buchegger and J.-Y. L. Boudec, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks", Proceedings of the Tenth Euromicro Workshop on Parallel, Distributed and Network-based Processing, Canary Islands, Spain: IEEE Computer Society, January 2002, pp. 403--410.

[13] Sergio Marti, T.J. Giuli, Kevin Lai and Mary Baker, "Mitigating Routing Misbehaviour in Mobile Ad Hoc Networks", Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom 2000)

[14] Sonja Buchegger and Jean-Yves Le Boudec. "Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes: Fairness In Dynamic Ad-hoc NeTworks)", In Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Lausanne, CH, June 2002.

[15] The CMU Monarch Project, "The CMU Monarch Projects Wireless and Mobility Extensions", Http://www.monarch.cs.rice.edu/cmu-ns.html. Oct. 12, 1999

[16] The Network Simulator - ns-2. http://www.isi.edu/nsnam/ns/, 2002.

〈著者紹介〉

축 뢰 (ZHU LEI) 학생회원
2003년 7월 : Automatization from Zhejiang Univ. of Technology 졸업
2004년 3월~현재 : 인하대학교 정보통신대학원 석사과정
〈관심분야〉 Network 보안

강 전 일 (Jeonil Kang) 학생회원
2003년 2월 : 인하대학교 컴퓨터 공학과 졸업
2004년 3월~현재 : 인하대학교 정보통신대학원 석사과정
〈관심분야〉 RFID 보안

양 대 헌 (DaeHun Nyang) 정회원
1994년 2월 : 한국과학기술원 과학기술 대학 전기 및 전자 공학과 졸업
1996년 2월 : 연세대학교 컴퓨터 과학과 석사
2000년 8월 : 연세대학교 컴퓨터 과학과 박사
2000년 9월~2003년 2월 : 한국전자통신연구원 정보보호연구본부 선임연구원
2003년 2월~현재 : 인하대학교 정보통신대학원 조교수
〈관심분야〉 암호이론, 암호프로토콜, 인증프로토콜, 무선 인터넷 보안