<https://link.springer.com/content/pdf/10.1007/s11276-015-1166-y.pdf>

**Trust based Intelligent Routing Algorithm for Delay Tolerant Network using Artificial Neural Network, 8 Jan 2016**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[1. 개요]**  DTN에서 각 node의 Trust (신뢰성) 값에 기반하여, 딥 러닝을 이용하여 라우팅을 하는 알고리즘에 대한 논문이다.  **[2. 모델 요약]**  Trust function은 다음과 같다.  신경망을 학습시키는 binary activation function은 다음과 같다.   |  | | --- | | where |   **[3. 관련 알고리즘]**   |  | | --- | | Notations: | | **CDR:** Call Data Record, **RC:** Route cache  **bias:** initialized as -999  ***:***node  If fault condition encountered, set Trust Value = -999 |  |  | | --- | | Algorithm 1: Initial Trust based Intelligent Routing Algorithm | | 다음을 모든 intermediate node에 대하여 반복한다.   1. CDR을 이용하여 을 fetch한다. 2. Y를 계산한다. 3. RC를 이용하여 를 fetch한다. 4. do  |  |  |  | | --- | --- | --- | | Y=0 | |  | | Y=1 |  |  | |  |  | | |

|  |  |  |
| --- | --- | --- |
| **[3. 관련 알고리즘]**   |  | | --- | | Algorithm 2: Trust based Intelligent Routing Algorithm for Intermediate Node | | 다음을 모든 intermediate node에 대하여 반복한다.   1. RC를 이용하여 가 intermediate node인지 확인한다. 2. 모든 과 에 대하여 **를 계산**한다. 3. **를 Trust Value가 가장 큰 intermediate node**로 선택한다. **그렇지 않으면**    1. 의 RC를 확인한다.    2. **2개 이상의 node가 같은 highest Trust Value를 가지면**       1. 의 CDR을 이용한다.       2. **모든 node의 Trust Value가 서로 같으면**          1. 의 값을 fetch한다.          2. 을 의 값이 이것보다 작은 next intermediate node로 설정한다.       3. **그렇지 않고 2개 이상의 node가 에 대해 서로 같은 값을 가지면**          1. Next intermediate node를 랜덤하게 지정한다. | |

<https://link.springer.com/content/pdf/10.1007/s11276-016-1320-1.pdf>

**Design of efficient lightweight strategies to combat DoS attack in delay tolerant network routing, 29 Jun 2016**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **[1. Spray & Focus routing protocol]**  메시지는 **Spray와 Focus라는 2개의 phase**를 이용하여 포워딩된다.   |  |  | | --- | --- | | Spray phase | node는 n>1개의 forwarding token과 함께 message copy를 갖는다.   * 주어진 메시지에 대한 **copy가 없는 node를 만나면 해당 copy를 해당 node에 forwarding**한다. * 이때 개의 forwarding token을 전송하고 개를 갖는다. | | Focus phase | **1개의 forwarding token이 있는 copy**를 가진 node에 대한 phase이다.   * 이때 copy는 single copy utility-based scheme with transitivity에 의해 **이 message copy가 없는 node로 라우팅**된다. |   **[2. Utility Value Settings]**  2개의 node i, j에 대해 **해당 2개의 node가 마지막으로 만난 시점**을 기준으로 시간을 측정하는 timer가 maintain된다. (1 tick마다 1씩 증가)   * **Utility Value between node i and j:**  |  | | --- | | 1. Node i, j가 서로 만나면 2. Transitivity에 의해서도 업데이트된다.  * 는 **node i에서 mobility model m을 통해 node k에 이르는** 데에 걸리는 시간 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[3. Thread model]**  **Black Hole Denial-of-Service (DoS)** 공격에서 malicious node는 **수신하는 모든 packet을 drop**할 수 있다. 이런 malicious node들이 메시지의 라우팅에 참여할 수 있는데, **이때 packet은 malicious node 이후 더 이상 전송되지 않는다**.   |  |  | | --- | --- | | Spray Phase | 이웃 node로부터 **packet을 수신한 다음 즉시 drop**하는 것으로 나타난다. (Spray Malicious / Spray Malicious Node) | | Focus Phase | Forwarding decision은 utility 값에 의해서 나타난다.   * maliciousness가 **destination에 대해 fake utility value (less being better)를 정의**하는 node로 보여진다. | | Greyhole DoS attack | 어떤 node가 **특정 확률(0<p<1)로** packet을 drop하고 라우팅 테이블을 수정한다. |   **[4. Detection mechanism using trusted node] ( 1/ 2)**  Peer node checking으로 했을 때 **overhead가 커지는 문제점을 방지하기 위해, specialized trusted node를 이용**하여 DoS attack을 탐지한다.  **<Regular Node와 Trusted Node의 역할>**   |  |  |  | | --- | --- | --- | | 역할 | Regular | Trusted | | **각 encounter를 entry로 하는 테이블** (Network Encounter Table, NET) 을 만들고 유지한다.   * **필드:** **(a)(b)** Sender and Receiver node’s ID, **(c)** time of encounter, **(d)** Message ID exchanged, **(e)(f)** Copies remaining with sender and receiver after message exchange | **O** | **O** | | **Malice Identification List (MIL)**을 저장한다.   * **전체 네트워크에서 Malicious node로 탐지된 node의 ID**를 저장한 list | **O** | **O** | | **<regular node 간의 통신>**   1. 각자의 **MIL에서 상대방 node**의 ID를 찾는다. 2. Spray & Focus 프로토콜에 의해 **Packet을 교환**한다. 3. Packet 교환에 대한 **record 생성, 이것을 node의 NET에 저장** 4. **MIL을 상대방 node와 서로 교환**하여, 각자의 MIL에 없지만 상대방의 MIL에 있는 node의 ID를 각자의 MIL에 추가한다. | **O** |  |   **Sink node:** false detection을 피하기 위한 개념으로, **TTL 또는 buffer constraint에 의해 메시지를 제거**해야 하는 node에 대해, **그것의 receiver**를 sink node로 지정한다. |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **[4. Detection mechanism using trusted node] ( 2/ 2)**  **<Regular Node와 Trusted Node의 역할>**   |  |  |  | | --- | --- | --- | | 역할 | Regular | Trusted | | **Spray Maliciousness Detection을 수행한다.**  **Spray phase detection:**   1. Trusted Node (TN)가 regular node를 만나면 이 node로부터 대응되는 NET을 받는다. 2. TN은 NET에서 **interacting node가 수신했지만 account되지 않은 message copy**가 있는지 찾는다. 3. **그런 message copy가 있으면 해당 node를 malicious node로 지정**한다.   **Focus maliciousness detection:**   1. TN은 **Log Table (LT)**에 encounter record를 저장한다. 2. Encounter record가 양측(interacting node 등)에 의해 확인되면, 이것은 **Latest encounter record (LER) Table**에 추가된다. 3. LER table에 추가된 encounter record는 detection computation에 사용된다.    1. 이 **confirmed record**에 기반하여, TN은 **Change by Transitivity (CT) table**에 있는, **node 간 transitivity에 의한 utility value의 변화**를 기록하여 저장한다.    2. 여기서 TN은 또한 **encounter information에 기반한 utility value**를 계산하며, 이 값은 **Declared Utility Vector (DUV)**에 저장된다. 4. **Phase in focus maliciousness detection**  |  |  | | --- | --- | | Information gathering mode | TN은 **참여하는 node들과 랜덤하게 상호작용**하고, 이때 얻은 정보를 LT, LER, CT에 추가한다. | | Detection mode | Information gathering mode에서 얻은 정보를 이용하여 **utility value의 불일치 여부를 계산**한다.   * 이때 TN은 node와의 상호 작용을 중지하고 계산에만 집중한다. | |  | **O** |   **<Figures>**   |  |  |  | | --- | --- | --- | | Encounter between regular nodes | Information gathering phase of TN | Detection phase of TN | |  |  |  | |

<https://etd.ohiolink.edu/!etd.send_file?accession=case1545036060190319&disposition=inline>

**APPLICATION OF MACHINE LEARNING TECHNIQUES TO DELAY TOLERANT NETWORK ROUTING, December 5, 2018**

|  |  |
| --- | --- |
| **[1. Approach]**  **Q-routing:** **packet routing을 위해 개발된 Q-learning**을 적용한 방법으로, Q-table에 기반하여, estimate된 **end-to-end packet delivery time**을 계산한다.   |  | | --- | | While true:   1. queue에서 **패킷을 선택**한다. 2. **가장 작은 Q(y, d)**에 해당하는 이웃 node로부터 를 선택한다. 3. 로부터의 **응답을 대기**한다. 4. 를 이용하여 를 업데이트한다. |       **Classification Based Routing:** **routing problem을 분류를 위한 머신러닝 문제로 간주**하고 해결한다.      **Clustering:** **거대한 dataset에서 패턴을 인식**할 수 있다.   * 여기서는 **K-means clustering** 알고리즘을 사용한다. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **[2. Machine Learning Algorithms]**  **<Classification Algorithm Overview for Central Node>**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | node로부터 **network statistics**를 수신한다.  학습을 위하여 data format을 한다.  데이터를 저장한다.  Training data와 test data를 선택한다.  알고리즘을 선택한다.   |  |  | | --- | --- | | Base Classifier | Naïve Bayes, K-Nearest Neighbors, DT | | Multi-label Classifier | Classifier Chain, Ensemble of Classifier Chains, LP |   모델을 학습시킨다.  모델을 테스트하고 점수를 산출한다.  모든 node에 대해 모델을 적용한다. |   **<Classification Algorithm Overview for Worker Nodes>**   |  | | --- | | **Network statistics**를 기록한다.  Central node로부터 학습된 모델을 수신한다.  **Bundle search event가 발생하기 전까지 다음을 반복한다.**   1. **알려진 valid model**이 있으면    1. Bundle이 **epidemic routing을 위한 기준**을 만족시키면       1. Time index를 결정한다.       2. 분류한다. **(local id, neighbor id, destination id, time index)**       3. Bundle이 전송되어야 하면 bundle list에 추가하고 전송한다. 2. 그렇지 않으면 **epidemic criteria에 따라 라우팅**을 한다.   Period time t가 지난 후에, 저장된 network statistics를 전송한다. | |