메쉬 네트워크 상에서 노드 자율적 소프트웨어 업데이트 방법 제안

김민수○ 이승형 이성원
경희대학교 컴퓨터공학과
betas@khu.ac.kr, shlee7@khu.ac.kr, drsungwon@khu.ac.kr

Autonomically Software Updating Method on Mesh Network

Min-Su Kim○ Seung-Hyung Lee Sung-Won Lee
Department of Computer Engineering, Kyung Hee University

요 약

본 논문에서는 원격지에서 무선 메쉬 네트워크로 여러 노드들의 특정 바이너리를 손쉽게 업데이트 할 수 있도록 하는 프로그램을 제안한다. 구체적으로 1) 데이터링크 계층에서의 통신 방법과 2) 바이너리가 외부로 부터 공격받을 가능성을 대비하여 바이너리의 일부를 암호화 하여 전송하는 방법을 특징으로 한다. 이러한 방법을 통해 메쉬 네트워크 관리자는 보안성이 확보된 자율적인 소프트웨어 업데이트를 수행할 수 있어, 장비를 업데이트 하거나 관리하는데 드는 시간과 비용을 줄일 수 있을 것이다.

1. 서 론

최근 IoT시대로 접어듬에 따라 무수히 장비가 무선 네트워크로 연결되고 있다.

하지만 이러한 많은 장비가 특정 한 기지국에 연결되어 기능을 수행하게 되면, 기지국에는 많은 부하가 가므로 노드와 노드가 서로 독립적으로 자기 자신과 네트워크를 제어할 수 있도록 하여 완전 분산 네트워크를 만드는 것이 중요해지고 있다.

특히 이러한 완전 분산 네트워킹을 수행하려면 IoT장비에 탑재되는 펌웨어, SDN정책, 구동에 중요한 핵심 소프트웨어 바이너리 등을 노드끼리 항상 최신 버전으로 유지해주어야 할 필요성이 있는데, 수십 대의 장비에 물리적으로 업데이트를 수행한다면 많은 시간과 비용이 소모될 것이다.

따라서 본 논문에서는 메쉬네트워크 상에서 각각의 노드가 자발적으로 바이너리를 최신버전으로 업데이트할 수 있는 방법을 제안하고, 이러한 방법에 따라 발생하는 취약점인 Injection을 통한 펌웨어 해킹에 대비할 수 있는 방법을 제안하고자 한다.

2. 관련연구

2.1 메쉬 네트워크

일반적으로 무선 네트워크 환경에서 각각의 단말을 노드로 지칭하고, 이를 서로 그물망처럼 연결시켜서 메쉬 네트워크를 구성한다.

이에 따라 각각의 노드들은 1개 이상의 무선 네트워크 인터페이스를 바탕으로 서로 AdHoc 네트워크를 구성하게 되는데, 이를 통해서 각각의 노드는 Multi Hop으로 서로 연결될 수 있게 된다**[1]**.

이러한 네트워크 구성방식의 장점은 셀룰러 네트워크와 같이 네트워크의 중앙이 되는 기지국에서 전파 전송을 통제할 필요 없이, 각각의 노드가 자율적으로 네트워크를 구성하여 노드간 통신이 가능토록 하고, 이렇게 구성된 노드 중 어떠한 노드가 외부 망에 연결된다면 자연스럽게 해당 노드가 게이트웨이 역할을 수행하게 네트워크 접속이 가능해진다. 따라서 이러한 형태의 네트워크 기술은 IoT가 발달함에 따라 필수적 요소로 자리잡고 있다 **[2]**.

2.2. RSA 비대칭 암호화

RSA암호화 기법은 공개키와 비밀키의 2개의키를 바탕으로 암호화 및 복호화를 수행하는 알고리즘으로 공개키만으로는 비밀키를 쉽게 예측할 수 없다**[3]**.

이러한 RSA기법을 바탕으로 암호화된 내용은 1) 비밀키를 이용하여 암호화 된 경우 공개키로만 해독이 가능하고, 2) 공개키를 이용하여 암호화 된 경우 비밀키로만 해독이 가능하다는 특징이 존재한다.

따라서 브로드캐스팅과 같은 일방적인 전송이 필요할 때에는 전송주체가 비밀키를 이용해서 암호화를 수행하고, 수신측에서 사전에 공개된 공개키를 이용하여 복호화를 수행한다면 데이터의 위, 변조를 판단할 수 있게 된다.

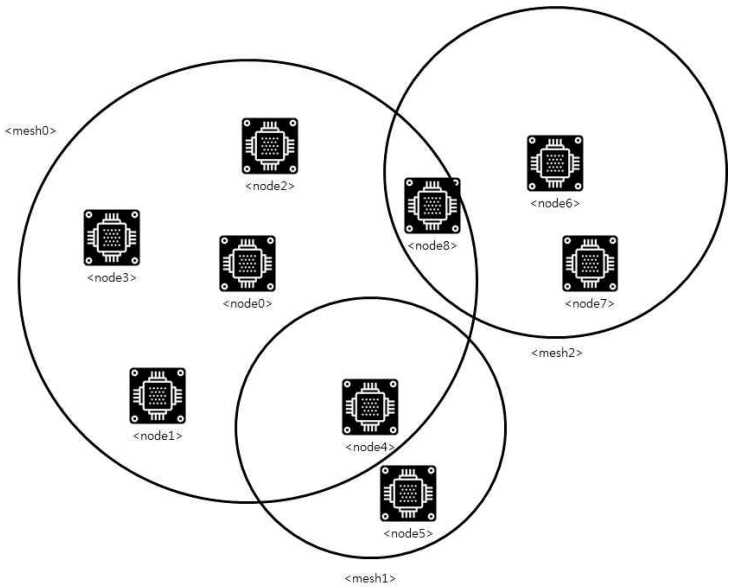
3. 구현 방안

3.1. 메쉬 네트워크 구성

검증을 위한 환경에서의 메쉬 네트워크의 노드는 아래와 같은 환경으로 구성되며, 1개 이상의 네트워크 인터페이스를 바탕으로 AdHoc모드로 동작한다.

#	NIC	Connected SSID
1	wlan0	mesh0
2	wlan0	mesh0
3	wlan0	mesh0
4	wlan0	mesh1
	wlan1	mesh0
5	wlan0	mesh1
6	wlan0	mesh2
7	wlan0	mesh2
8	wlan0	mesh2
	wlan1	mesh0

<표1. 메쉬네트워크 구성 테이블>



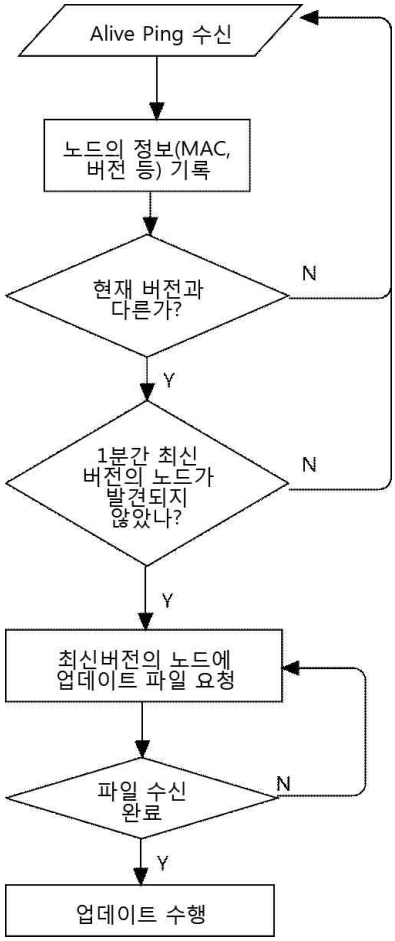
<그림2. 메쉬 네트워크 구성 도식>

<표1>을 바탕으로 구성된 <그림2>와 같은 네트워크의 각 노드 들에는 자기 자신을 업데이트하고 관리할 수 있는 매니저 데몬이 설치된다.

3.2. 매니저 데몬

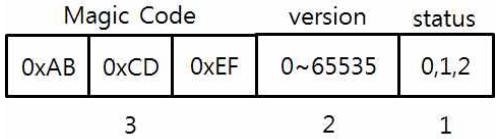
매니저 데몬은 OS의 System Service에 내장되어 부팅 시 자동으로 실행되며, 안정성을 위하여 리눅스 cron서비스를 이용하여 매니저 데몬 프로세스가 내려가지 않았는지 주기적으로 검사한다**[4]**.

구현된 매니저 데몬은 <그림3>과 같은 기능을 수행한다. 1) 연결된 노드끼리 DataLink 계층을 통해 주기적으로 Alive Ping을 브로드캐스팅 한다. 2) 수신된 Alive Ping을 바탕으로 인접 노드와 최신버전에 대한 정보를 기록 한다. 3) 가장 최신 버전을 가진 노드에게 업데이트 바이너리를 요청하여 다운로드 하여 업데이트를 적용한다.



<그림3. 업데이트 프로세스>

Alive Ping의 프레임은 아래와 같이 구성되어 진다.



<그림4. Alive Ping DataLink Frame>

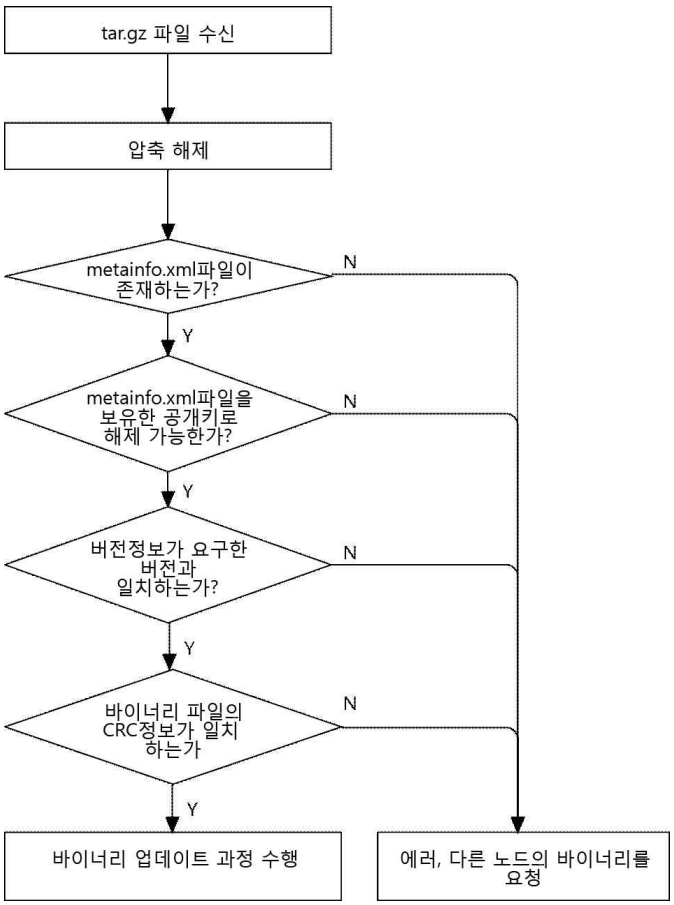
코드	설명
0	에러
1	정상
2	업데이트중

<표5. Alive Ping Status 목록>

<그림4>의 프레임은 각각 본 논문에서 제안한 메니저 노드가 발생시킨 데이터프레임 임을 알 수 있도록 3bytes의 MagicCode가 들어가며, 이후는 서비스의 버전, 노드의 Status값이 <표5>와 같이 들어가게 된다.

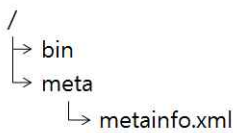
3.3. 바이너리 배포 프로토콜

3.2절에서 설명한 것 과 같이 바이너리를 배포 할 때의 문제점은 자율적으로 직접 배포하므로 어떤 한 악의를 품은 사용자 또는 프로그램이 버전 업데이트를 가장한 크래킹 바이너리(Backdoor 등)를 전송하게 된다면 모든 프로그램이 해커의 손에 장악 될 수 있다는 문제점이 있다**[5]**.



<그림6. 바이너리 보안 인증 과정>

따라서 <그림6>과 같은 바이너리 전송 프로토콜에 따라 업데이트가 이루어지게 된다.



<그림7. 바이너리 전송을 위한 폴더 구조>

바이너리를 배포 할 때는 <그림7>과 같은 폴더구조로 구성되며, 이를 gzip을 이용하여 tar.gz파일구조로 압축, 전송 한다.

이중 <그림7>의 meta폴더 안에 위치한 metainfo.xml은 1) 어플리케이션의 버전정보, 2) 바이너리의 손상 또는 위, 변조를 검사하기 위한 CRC Checksum정보를 가지고 있다**[6]**.

또한, <그림6>에 따르면 metainfo.xml파일을 RSA공개키를 이용해 복호화 하게 되는데, 이를 위하여 사전에 네트워크 관리자는 RSA암호화 방식을 이용하여 공개키와 개인키를 생성 하여야 한다. 이중 공개키는 언급된 것과 같이 개별 노드에 저장되며 metainfo.xml을 복호화 하는데 에 사용하게 되고, 개인키는 어플리케이션 바이너리를 배포할 때 사용하게 된다. 결과적으로 이렇게 전송된 바이너리는 최종적으로 무결성을 보장받을 수 있게 된다.

4. 결론

앞서 살펴본바와 메쉬 노드의 관리는 앞으로 IoT분야의 규모가 커질수록 장비의 개수가 늘어나므로 어려워 질 것이라는 사실은 매우 자명하다.

따라서 한 노드에게만 업데이트를 수행하면, 나머지 인접노드들이 자율적으로 바이너리를 주고받아 자동으로 업데이트 되어 바이너리를 항상 최신으로 유지하기 때문에 어느 정도 시간이 지나면 모든 노드들이 최신버전으로 구동될 수 있게 되며, 이러한 방법에 따라서 발생할 수 있는 취약점인 Backdoor의 배포 문제 등 해킹에 대한 가능성도 바이너리 전송 프로토콜과 그의 암호화를 통해 막을 수 있는 방법을 제안하였다.

본 연구를 통해 기대할 수 있는 점으로는 위에서 제안한 방식의 프로토콜을 구현한 단일 시스템 서비스 하나로 기존 프로그램에 통합할 필요 없이 구축이 가능하다. 또한, 여러대의 물리노드를 최신으로 유지하기 위한 노력을 획기적으로 단축시킬 수 있을 것 이라 생각한다.

따라서 이 연구를 바탕으로 오픈소스 소프트웨어를 구현하여 향후 실제 무선 메쉬 네트워크에 적용하여 실효성을 검증해볼 계획이다.

* "본 연구는 미래창조과학부 및 정보통신기술진흥센터(IITP)에서 지원하는 서울어코드활성화지원사업의 연구결과로 수행되었음" (R0613-16-1203)

5. 참고문헌

[1] https://www.lifewire.com/ad-hoc-mode-in-wireless-networking-816560

[2] https://ko.wikipedia.org/wiki/%EB%AC%B4%EC%84%A0_%EB%A9%94%EC%8B%9C_%EB%84%A4%ED%8A%B8%EC%9B%8C%ED%81%AC

[3] https://ko.wikipedia.org/wiki/RSA_%EC%95%94%ED%98%B8

[4] https://en.wikipedia.org/wiki/Cron

[5] https://en.wikipedia.org/wiki/Backdoor_(computing)

[6] https://en.wikipedia.org/wiki/Checksum