**제목 : CoAP, 6LoWPAN, DTLS를 사용한 홈 네트워크 시스템 구축 방법**

이 문서는 작년 2017년에 진행되었던 졸업작품을 이어받은 새로운 프로젝트가 진행됨에 따라 발생한 많은 시행착오와 기본지식들을 이해하기 쉽게 설명하여 이후 프로젝트를 진행하는 친구들에게 조금이나마 도움이 되기를 바라며 작성하였습니다. - 2018년 3월 31일 김연수 -

1. 기본 개념

**CoAP**는 응용 계층 프로토콜이다. HTTP와 매우 비슷하나 보통 UDP 프로토콜로 구현하며 HTTP보다 패킷 크기가 매우 작다. HTTP는 헤더의 내용이 직관적인 문자열 형태로 구성되어 있다. 컴퓨터 네트워크 시간에 패킷 헤더 분석 프로젝트를 했을 것이다. 프로그램을 짤 때 HTTP 패킷을 골라내기 위해서 어떻게 했는지 기억해보라. 일단 if문에서 UDP와 TCP를 구분하고, TCP 부분에서 어플리케이션 계층 헤더에 “http”라는 문자열이 있으면 HTTP 프로토콜이었다(보통 strstr 함수를 사용한다). 또한 GET 요청을 보내면 실제로 패킷에 “get” 이라는 문자열이 포함된다. Coap도 http와 똑같이 GET, PUT, POST, DELETE의 동일한 옵션이 있다. 하지만 패킷의 크기를 줄이기 위해서 GET는 1로 약속하고 PUT는 3으로, 등등 HTTP에서 사용하는 기능을 숫자(비트)형태로 치환(압축)하여 헤더를 구성하였고, 이러한 압축 결과가 RFC 7252에 나와있는 다음의 CoAP 헤더 양식이다.

|  |
| --- |
| **0 1 2 3**  **0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1**  **+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+**  **|Ver| T | TKL | Code | Message ID |**  **+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+**  **| Token (if any, TKL bytes) ...**  **+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+**  **| Options (if any) ...**  **+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+**  **|1 1 1 1 1 1 1 1| Payload (if any) ...**  **+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+** |

RFC7252는 coap의 표준이 나와있는 문서이다. 하지만 영어여서 별로 도움은 안된다. 이 프로젝트를 진행할 때 coap에 대해 자세히 알 필요는 없다. 표준 문서에는 프로토콜의 용도부터 시작해서 신뢰적인 데이터 전송을 위한 핸드쉐이킹 방법과 매우 다양한 옵션 필드의 사용법들이 나와있으나, 실제로 사용할 것은 Code 부분의 GET과 PUT 메소드가 다다.

**6LoWPAN**은 조금 생소한 내용이긴 하지만, 알고보면 쉽다. 우리가 평소 사용하는 이더넷은 IEEE 802.3, 와이파이는 IEEE 802.11 출신이다. 이더넷과 와이파이는 링크/물리 계층이 서로 다른 프로토콜이지만, 우리는 어떤 방법으로 네트워크를 연결하여도 구글크롬에 접속해서 웹툰을 보거나 스타크래프트를 할 수 있다. 그 이유는 바로 이더넷과 와이파이 둘 다 상위 계층에 IP 계층이 있고 그 위에 TCP나 UDP 프로토콜을 구성하여 응용 프로그램을 만들기 때문이다.

링크/물리 계층 프로토콜을 하나 더 소개하자면 바로 802.15 이다. 802.11이 와이파이라고 불리는것 처럼 WPAN이라고 불리는 이 표준은 Wireless Personal Area Network의 약자로, 무선개인통신망으로 번역된다. 이 WPAN은 802.15.1 부터 802.15.7까지 일곱 종류가 있는데, 우리가 관심 있는 것은 바로 802.15.1인 ‘블루투스’ 이다. 평소에 사용하는 블루투스 기기들을 생각 해 보자. 우리는 스마트폰을 와이파이로 연결해서 유튜브 동영상을 보고, 블루투스 스피커를 스마트폰과 연결해서 소리를 들을 수 있다. 이처럼 블루투스는 보통 개인적으로 사용되는 물건과 연결된다. 사실 스피커를 와이파이로 연결 할 수도 있다. 다만 블루투스는 소비 전력이 적어 사용 시간이 오래가고 성능 제약이 적고 사용 반경이 넓다는 장점이 있어 작은 기기에 많이 쓰이게 되었고, ‘와이파이 스피커’ 라는 말은 생소하게 느껴진다. (그런데 블루투스는 한번에 보낼 수 있는 데이터가 적어서 디지털 음원을 한번 더 압축해서 전송하기 때문에 음질이 안좋다고 하여 요즘에는 와이파이 스피커가 보이고 있다.) 이와 같이 블루투스는 저전력 기기를 위한 링크/물리 계층 프로토콜 이기 때문에 IP계층 까지 갈 것도 없이 MAC주소 만으로 연결된다. 스마트폰에 블루투스 기기를 연결하면 연결된 기기의 정보를 볼 수 있는데, IP주소는 없고 MAC주소를 확인 할 수 있을 것이다. 이는 블루투스가 안좋아서가 아니라, 저전력이라는 블루투스의 특징이다. 그런데 이 블루투스를 와이파이나 이더넷 처럼 그 위에 IP 계층을 올려서 네트워크 통신을 할 수 있는데, 그것이 바로 지그비에서도 사용하는 6LoWPAN 기술을 사용하는 것이다 6lowpan 기술에 대한 자세한 이해는 아직 부족하므로 스킵하도록 하겠다.

1. 무엇을 만드는가?

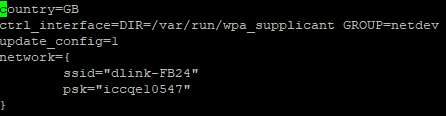
보안 기능이 있는 홈 네트워크 시스템을 만든다.

* 1. 사용자를 위한 CoAP클라이언트 어플리케이션: 집의 불을 껐다 켜거나 온도를 확인할 수 있다. Dtls 보안 기능도 있음.
  2. 게이트웨이: 사용자와 서버에게 ipv6 주소를 할당해주고, 패킷을 받으면 이를 6lowpan 망으로 라우팅한다. 이 부분의 개념이 조금 어려운데, 4학년 1학기 과목인 네트워크 매니지먼트를 듣기 바람.
  3. CoAP 서버: 사용자가 요구하는 데이터를 잘 반환한다. Dtls 보안 기능도 있음

1. 준비물

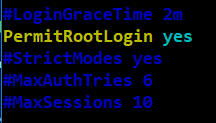
최소 준비물

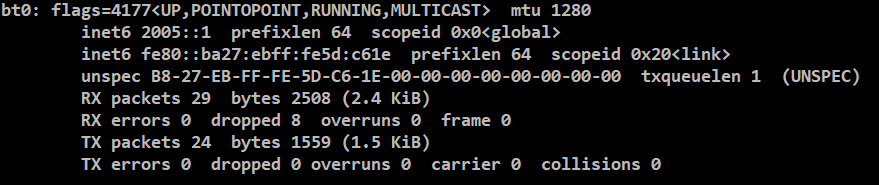
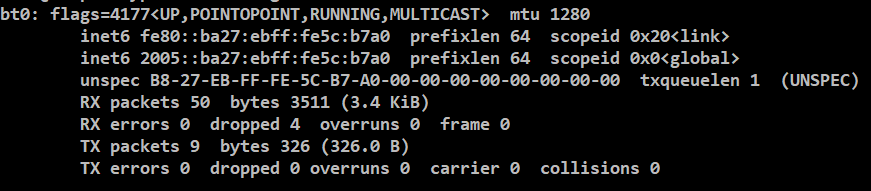
* ipv6를 지원하는 공유기 한 대(iptime꺼는 거의 안되니 중국회사인 Dlink 제품을 추천함, DIR-850)
* 라즈베리파이 2개 이상(제일최신거 사면 됨, 하나는 게이트웨이용, 나머지는 서버용, 방 하나당 서버 1개)
* 라즈베리파이에 꽂을 마이크로 SD카드 16기가이상 라즈베리파이 개수 만큼
* 마이크로 SD카드 리더기 1개
* 라즈베리파이 충전할 충전기 라즈베리파이 개수만큼 준비
* 아두이노 우노 키트 아무거나(당연히 비쌀수록 좋고, 원하는 센서가 들어있는 것으로 구매)

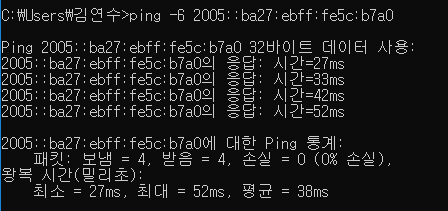
1. 만드는 법
2. 라즈베리파이 에 라즈비안 **lite** 를 깔고 준비된 공유기에 와이파이를 연결한다.
   1. vi /etc/wpa\_supplicant/wpa\_supplicant.conf
   2. 

위와 같이 ssid에는 와이파이 이름을, psk는 비밀번호를 적으면 된다.

1. raspi-config 입력, Interfacing Options에서 ssh접속을 허용한다.
2. 라즈베리파이의 root 비밀번호를 설정하고, ssh를 통한 root로그인을 허용한다.
   1. sudo passwd root #root 비밀번호 변경
   2. vi /etc/ssh/sshd\_config #ssh에서 root로그인 허용



1. 같은 공유기에 연결된 노트북이나 데탑의 Putty를 통해 라즈베리파이에 접속 할 수 있는지 확인한다. 이제부터는 모니터 필요 없이 putty로 작업한다.
2. Apt-get update, apt-get upgrade 를 진행한다. 여기까지는 라즈베리파이를 포함한 모든 리눅스 환경의 기본 설정이므로 쉽게 할 수 있을 것이다.(임베디드 시스템 수업을 들었으면 더 쉬움)
3. <https://wikidocs.net/3243> 이 홈페이지를 참고해서 각각의 라즈베리파이에 예전 커널인 4.9.80-v7+를 **크로스컴파일**을 통해 설치한다. 해당 커널의 소스는 <https://github.com/raspberrypi/linux/tree/rpi-4.9.y-stable> 에 있다. **라즈베리파이 공식 홈페이지에 있는 라즈비안 이미지는 최신 커널로 2018년 기준 4.14.y 버전인데, 6lowpan의 다중 연결이 되지 않는 오류가 있다. 이 문제로 시간을 많이 날렸었으니 참고바람.**
   1. 팁1: 일단 sd카드에 최신 라즈비안 이미지를 설치 한 뒤에 진행하는 작업임.
   2. 팁2: vm웨어 창 위에 마우스를 올린 상태로 sd카드를 꽂아야 가상머신 상의 우분투가 인식할 수 있음.
4. 게이트웨이로 사용할 라즈베리파이에 apt-get radvd 를 입력하여 설치한다. 그리고 **재부팅 한 뒤에** Script-sh의 1\_ble\_router.sh를 실행한다.
5. 서버로 사용할 라즈베리파이에서는 Script-sh의 ble\_host.sh를 실행한다. 영어로 된 설명이 나올텐데, 요약하자면 hcitool lescan를 입력하여 블루투스 장치를 검색하고, echo “connect 연결할MAC주소 1” > /sys/kernel/debug/Bluetooth/6lowpan\_control 을 입력하여 연결하라는 의미이다. 설명을 잘 읽고 따라한다.
   1. 게이트웨이: sh 1\_ble\_router.sh
   2. 서버: sh ble\_host.sh
   3. 게이트웨이: hciconfig hci0 leadv
   4. 서버: hcitool lescan
   5. 서버: (스캔을 강제종료 시킨 뒤) echo “connect B8:27:XX:XX:XX:XX 1” > /sys/kernel/debug/Bluetooth/6lowpan\_control
6. 서버와 게이트웨이 각각 ifconfig를 입력하여 bt0 이라는 인터페이스가 생성되었는지 확인한다. Bt0이 있다면, 게이트웨이에서 ifconfig bt0 add 2005::1/64 를 입력한다.
   1. 게이트웨이 
   2. 서버 
7. 이제 다음과 같은 형태가 된 것이다. 노트북이나 데스크탑은 2005:: 네트워크 망에 포함되지 않지만, cmd창을 열어 서버 중 한 곳의 ip에 ping을 보내면 정상적으로 신호가 간다. 게이트웨이의 radvd 라는 프로그램이 라즈베리를 라우터와 같이 동작하게 한 것이다.



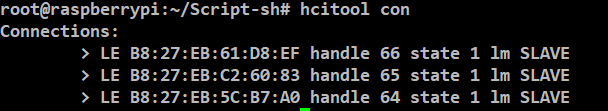
1. 추가로 서버를 더 연결하고 싶다면, 게이트웨이에서 hciconfig hci0 leadv를 입력하여 블루투스 연결 허용을 설정하고, 서버 라즈베리파이에서 sh ble\_host.sh 를 하고 echo “connect ~~”를 하면 된다. 다음은 3개의 서버를 연결한 상태이다.

|  |
| --- |
| 서버1 |
| 2005::ba27:ebff:fe5c:b7a0 |

|  |
| --- |
| 서버2 |
| 2005::ba27:ebff:fec2:6083 |

|  |
| --- |
| 서버3 |
| 2005::ba27:ebff:fe61:d8ef |

|  |  |
| --- | --- |
| 게이트웨이 | |
| fe80::ba27:ebff:fe5d:c61e | 2005::1/64 |



서버는 일단 제일 먼저 jdk부터 설치한다. 명령어는 Apt-get install default-jdk 이고 서버에 JDK 설치 시 아마 오류가 생길 것이다.

cp -r /usr/lib/jvm/java-8-openjdk-armhf/jre/lib/arm/client/ /usr/lib/jvm/java-8-openjdk-armhf/jre/lib/arm/server

을 입력 해 준 뒤 다시 apt-get install default-jdk 를 입력해 준다.

그리고 git clone <https://github.com/eclipse/californium> 명령어로 깃허브 저장소의 프로젝트를 다운받으면 된다. 프로젝트를 컴파일 하려면 maven 이라는 프로그램이 필요하니 apt-get install maven 을 쳐서 설치한다.(jdk가 설치되어 있어야 설치가능) Californium 프로젝트를 다운받아 폴더에 들어가보면 여러가지 폴더가 있는데, demo-apps 폴더에 예제 프로그램들이 많이 있다. 이 중 cf-helloworld-server 랑 client가 Coap 기본 예제이므로 일단 노트북에도 jdk랑 이클립스를 깔고 클라이언트를 실행시켜서 라즈베리파이에서 실행한 서버를 테스트 해 본다. 컴파일 하는 방법은 cf-helloworld-server 폴더에 들어가서 mvn clean install을 입력하면 된다.

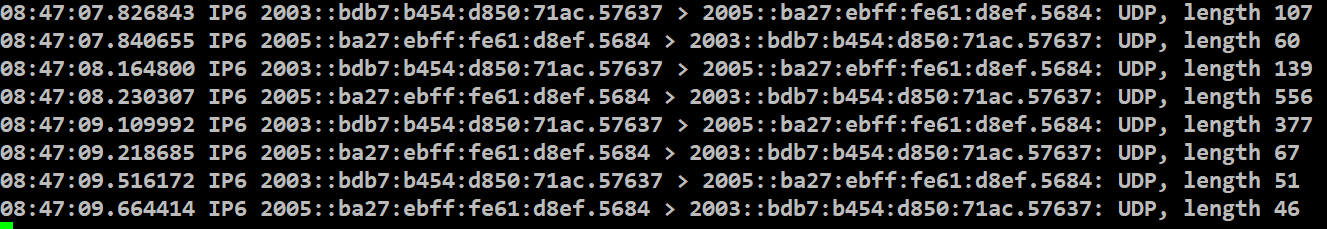
사실, 아까 설치했던 Radvd라는 프로그램은 이 프로젝트에서 매우 큰 역할을 한다. 우리가 사용하는 노트북, 스마트폰에는 ipv6 주소가 없다. 그래서 클라이언트는 원래 ipv6를 사용하는 서버와 통신을 할 수 없다. 그런데 게이트웨이에 설치한 radvd가 바로 ipv6 공유기 역할을 해준다. 클라이언트에게는 2003으로 시작하는 와이파이 주소를, 서버에게는 2005로 시작하는 블루투스 주소를 할당해주는 것이다. (덧붙여 말하자면 자동으로 라우팅 경로 추가까지 해준다. 원래는 주소를 할당만 하면 통신 안되고, ip -6 route ~~ 또는 route –inet6 add ~~ 명령어로 라우팅 설정도 해야됨)

그런데 여기서 문제가 뭐냐면 게이트웨이는 와이파이로 연결된 모든 기기에 2003번대의 주소를 할당해주는데, 이것이 서버도 예외는 아니라는 것이다. 현재 Californium 라이브러리를 사용한 dtls 프로그램을 만들 때 특정 인터페이스로 수신하는 것 까지는 해결하여서, bt0 인터페이스로 수신은 잘 되지만, 송신 할 때는 자동적으로 인터페이스를 찾아서 보내기 때문에, 자꾸 6lowpan 통신을 해야 하는 서버가 자꾸 와이파이로 패킷을 보내버린다.(서버에서 tcpdump –I bt0을 하고 dtls요청신호를 보내보면 패킷이 들어오기만 함. Tcpdump –I wlan0 을 해보면 bt0의 주소인 2005번에서 나가는 것으로 표시는 되어있으나, 실제로는 와이파이로 통신하는 것 같음.)

그래서 아무튼 이 문제를 해결하기 위해 서버의 wlan0의 ipv6 주소를 아예 다운시켜 버려야 한다.

1. vi /etc/sysctl.conf
2. net.ipv6.conf.wlan0.disable\_ipv6=1 을 추가한다.

이렇게 하면 서버는 완전히 블루투스를 통해서만 통신을 할 것이다.



위와 같이 tcpdump –i bt0 을 했을 때, 들어오는 것과 나가는 것 각각 네 번씩 총 8개의 dtls 핸드쉐이크 + Coap 메시지 교환 과정이 다 찝혔으면 성공이다.