Arduino를 이용한 CAN의 소규모 실험 환경 구축 및 보안 취약성 검증

안수현* 김광조*

*KAIST 정보보호대학원

Building Small-Scale Testbed for CAN using Arduino and Checking its Vulnerabilities

Soohyun Ahn* Kwangjo Kim*

*Graduate School of Information Security, KAIST.

요 약

Information and Communications Technologies (ICT) 기술의 발전은 여러 분야에 영향을 주었고 특히 자동차의 전자화라는 부분에서 큰 영향을 주었다. 자동차의 전자화는 Controller Area Network (CAN), Local Interconnect Network (LIN), FlexRay로 대변되는 자동차 네트워크 프로토콜의 영향으로 인해 많은 발전을 이루었다. 그 중에서 CAN이 현재 가장 널리 사용되고 있지만 CAN의 대중화에 비해 이에 대한 보안은 미흡하다. 이를 보완하기 위해 제시된 CAN에서의 보안 기술들은 자동차라는 특성 때문에 제대로 검증되지 못하고 있는 상황이다. 따라서 본 논문에서는 Arduino를 이용하여 소규모 CAN 실험 환경을 구축하여 CAN에서 큰 문제가 되고 있는 스푸핑과 Denial of Service (DoS)공격을 실험한 결과를 제시하였다.

I. 서론

ICT 기술의 발전은 많은 분야에 영향을 주었다. 그 중 자동차 분야 역시 ICT 기술과 접목 되면서 많은 영향을 받았다. 이로 인해 기계만으로 작동된다고 생각되던 자동차가 이제는 점점 발전하여 거의 모든 것이 전자식으로 바뀌게 되었다. 이렇게 자동차가 전자식으로 변화할 수 있게 된 것은 CAN, LIN, FlexRay로 대변되는 자동차 네트워크 프로토콜 때문이라고 할 수 있다. 그 중 CAN은 대표적인 자동차용 네트워크 프로토콜로 현재 대부분의 자동차에서 표준으로 사용하고 있다. CAN을 사용함으로써 자동차 엔진, 브레이크 등을 제어 할 수 있게 해주는 Electronic Control Unit (ECU)들이 서로 통신을 할 수 있게 되었다. 이렇게 CAN이 도입됨으로써 이제는 70여개의 ECU에 의해 자동차가 모든 부분에서 제어가 가능하게 되었다.

CAN의 도입으로 인해 자동차의 제어가 더 편리해지고 효율적으로 바뀌었지만 이러한 발전의 비례하여 CAN의 보안에 대한 대비는 연구가 거의 되지 않았다. 기본적으로 CAN은 브로드캐스팅 방식이기에 모든 ECU들이 메시지들을 받을 수 있지만 이는 곧 보안상에 취약점이 있음을 보여준다. 예를 들어 공격자가 악의적인 의도를 가지고 ECU를 자동차 내부에 설치한다면 이 ECU를 통해 공격자는 그 자동차 내부에 정보를 쉽게 파악할 수 있고 심지어는 DoS 공격을 시도함으로써 자동차 제어를 어렵게 만들 수도 있다. 이러한 시도는 최근에 여러 연구 [1][2]를 통해 실현 가능성이 입증 되었으며 CAN에 대한 보안 대책이 큰 문제로 대두되었다. 이러한 현상을 반영하듯 CAN의 보안 문제를 보완하기 위한 여러 연구들이 수행되었다 [3][4][5]. 하지만 많은 논문들이 자동차가 갖고 있는 비용과 접근성 때문에 실험환경을 구축하기 어려워 실제 실험을 수행하지 못하고 제안만 하고 있다. 따라서 검증을 위한 실험 모델 개발이 필요한 상황이다. 이러한 상황에서 본 논문은 국내에서 아직 시도되지 않

은 방법인 Arduino를 이용하여 소규모 CAN 실험 환경을 만들고 이러한 실험 환경에서 스푸핑과 DoS 공격을 시도한다.

본 논문의 구성은 다음과 같다. 논문의 2장에서는 CAN과 Arduino, CAN 실드에 대해서 알아보며 3장에서는 Arduino를 이용하여 구축된 실험 환경을 설명하고 4장에서는 구축된 실험 환경하에서 스푸핑 공격과 DoS 공격 실험결과를 제시한다. 마지막으로 5장에서는 결론을 맺는다.

II. 배경 지식

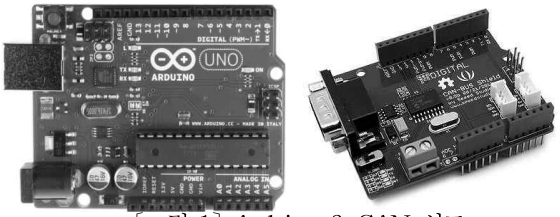
2.1 CAN

현재 사실상의 자동차용 네트워크 프로토콜 표준이라고 할 수 있는 CAN은 1986년 2월에 미국의 자동차 관련 업계 단체인 SAE (Society of Automotive Engineers : 자동차 기술자 협회)의 회의에서 독일의 Robert Bosch사가 제안하였다. Bosch사는 1991년에 현재의 사양인 CAN 프로토콜 사양 2.0을 발표하고 ISO (International Organization for Standardization : 국제표준화기구)에 제출하였다. 1993년 11월에는 정식 ISO 표준으로 'ISO 11989'이 공개되었고 물리 계층의 전송속도는 최대 1.0Mbps로 정의 되었다. 또한 1995년에는 개정안이 제출되어 ISO 11898로 확장되고 29bit의 ID를 가진 CAN이 등장했다. 이것은 현재 'CAN 2.0B'로 불리고 있으며 [6]. CAN의 주요 특징은 아래와 같다.

1) 브로드캐스트 방식
2) 통신 속도가 빠르고 최대 1km 까지 전송이 가능
3) 하드웨어적인 오류보정 기능 제공
4) CAN만의 자체적인 중재 기능 제공
5) 두 가지 레벨이 존재 (도미넌트, 리세시브)

2.2 Arduino 와 CAN 실드

Arduino는 오픈소스를 기반으로 한 단일 보드 마이크로컨트롤러이다. 컴퓨터 메인보드의 단순 버전으로 Arduino에 다양한 센서나 부품 등의 장치를 연결할 수 있다. Arduino에는 추가적인 기능을 제공하는 실드라는 것이 존재하는데 이를 이용하면 기본 Arduino 보드가 제공하지 못하는 기능들인 이더넷, 와이파이, 블루투스 등을 사용할 수 있다. 또한 Arduino에서는 CAN 통신을 지원해주는 실드를 제공해주고 있는데 이 점의 착안하여 본 논문에서는 이 CAN 실드와 CAN 실드 제작회사에서 제공해준 라이브러리를 사용하여 실험을 진행하였다. [그림 1] 은 Arduino와 CAN 실드를 보여주고 있다.

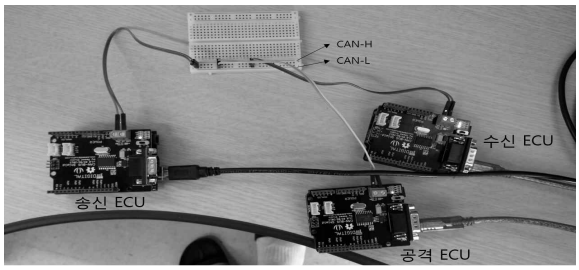


[그림 1] Arduino & CAN 실드

III. 실험 환경 구축

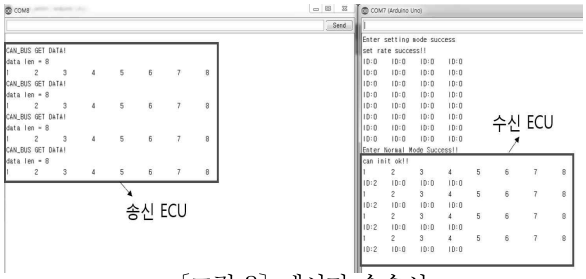
본 논문에서는 간단한 CAN 실험 모델을 만들기 위해서 CAN 실드와 함께 제공되는 라이브러리를 사용하였다. 하지만 기본적으로 제공해주는 라이브러리에는 CAN의 중요한 특징인 중재부분이 구현이 되어 있지 않다. DoS 공격에서 중재 기능이 중요하게 사용되기 때문에 이 기능을 라이브러리를 수정하여 추가하였다. 또한 중재기능을 위해서 버스에 상태를 확인해주는 기능이 필요하기 때문에 이 부분 또한 추가하였다.

기본적으로 CAN에서는 CAN-H와 CAN-L 2개의 채널을 통해 통신을 하는데 CAN 실드에서도 이를 제공하고 있다. 이를 이용하여 Arduino끼리 연결하거나 브레드 보드에 연결하여 사용하면 메시지를 전송하거나 수신할 수 있다. 이러한 특징을 이용하여 Arduino, CAN 실드를 하나의 ECU라고 가정하고 실험 환경을 구축해 보았다. 실험 환경에서는 3개의 Arduino, CAN 실드를 사용하였고 각각 송신 ECU, 수신 ECU, 공격 ECU라고 가정하고 실험 환경을 구축하였으며. [그림 2] 는 본 논문에서 사용된 실험 환경이다.



[그림 2] 실험 환경

[그림 2]를 보면 알 수 있듯이 브레드 보드에 각 CAN 실드의 CAN-H, CAN-L을 브레드 보드에 +, - 선에 대응시켜서 연결 하였다. [그림 3]은 이를 이용하여 데이터를 송수신한 결과를 나타낸다.



[그림 3] 메시지 송수신

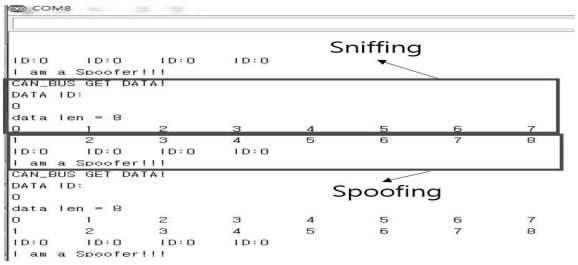
[그림 3]에서 왼쪽이 송신 ECU, 오른쪽이 수신 ECU를 나타내며 메시지 송수신이 잘 이루어지고 있음을 볼 수 있다.

IV. 결과

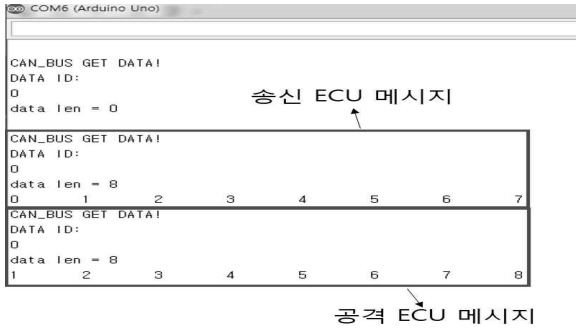
3장에서 구축된 실험 환경을 이용하여 CAN에서 가장 문제시 되고 있는 두 가지 공격인 스푸핑 공격과 DoS 공격을 실험해보았다.

3.1 스푸핑 공격

기본적으로 CAN은 브로드캐스트 환경이다. 이는 CAN을 사용하는 환경에서는 연결된 모든 ECU가 메시지를 아무런 제약 없이 받을 수 있다는 것이다. 이는 곧 스니핑과 스푸핑 공격이 쉽게 가능하다는 것을 뜻한다. 이러한 특징을 이용하여 구축된 실험 환경에서 스푸핑 공격을 시도해 보았다. 송신 ECU가 메시지를 보내면 공격 ECU는 이 메시지를 스니핑 후 곧 바로 송신 ECU의 메시지 ID를 이용하여 새로운 메시지를 만든 후 이를 수신 ECU에게 송신한다. 이 때, 송신자의 메시지 ID는 0으로 설정하고 메시지를 송신하였다. [그림 4], [그림 5]는 각각 공격 ECU와 수신 ECU의 메시지 송수신 상황을 보여 주고 있다.



[그림 4] 공격 ECU



[그림 5] 수신 ECU

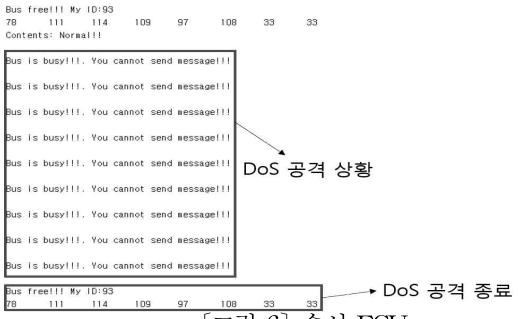
[그림 5]에서 메시지의 ID는 같지만 메시지의 내용이 다른 것을 통해 수신 ECU가 송신 ECU의 메시지뿐만 아니라 공격 ECU의 메시지도 수신을 하고 있음을 볼 수 있다. 즉, 스푸핑 공격이 되고 있음을 나타내고 있다.

3.2 DoS 공격

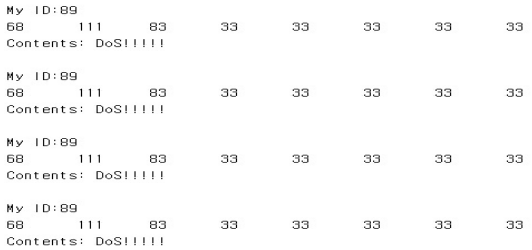
CAN에서의 DoS 공격은 일반적인 DoS 공격과는 다른 특징을 보이는데 이는 CAN에서 사용되는 중재 기능 때문이다. CAN에서는 Carrier Sense Multiple Access wih Bitwise Arbitration (CSMA/BA)기술을 이용하여 동시에 메시지를 전송할 시 메시지의 우선순위를 정한다. 이 때, 앞에서 설명한 두 가지 레벨인 도미넌트와 리세시브를 이용한다. CAN에서는 비트 0이 도미넌트, 비트 1이 리세시브를 나타내고 도미넌트가 리세시브 보다 우선순위를 가진다. 이 특징을 이용하여 여러 개의 ECU가 메시지를 전송할 시 메시지의 각 ID필드를 비트단위로 비교하고 만약 한 노드가 1인 비트 (리세시브)를 갖는다면 이 ECU의 메시지는 우선순위를 잃게 된다. 결국 한 개의 ECU의 메시지만이 남게 되고 이 메시지만이 수신된다. 또한 이러한 상황에서 작은 ID가 큰 우선순위를 갖게 된다. 이러한 특징을 이용하여 CAN에서는 공격 ECU가 스니핑를 통해 높은 우선순위를 갖는 작은 ID를 가진 메시지를 얻게 되고 이를 이용하여 메시지를 지속적으로 전송하게 되면 다른 ECU는 메시지 우선순위에서 밀리게 되어 전송이 되지 못한다. 즉, DoS 공격이 되는 것이다. 이러한 특징을 이용하여 구축된 실험 환경에서 DoS 공격을 실험해 보았다. 이 때, 사용된 ID는 아래와 같다.

- 송신 ECU: 0x5D/93
- 공격 ECU: 0x59/89

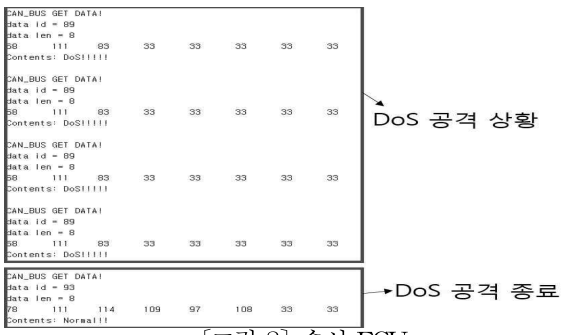
공격 ECU의 ID가 더 작기 때문에 송신 ECU의 메시지는 우선순위에서 밀리게 된다. 이러한 조건에서 실험을 실시하였고 [그림 6], [그림 7], [그림 8] 은 실험 결과를 보여주고 있다.



[그림 6] 송신 ECU



[그림 7] 공격 ECU



[그림 8] 수신 ECU

송신 ECU에서는 메시지를 보내기 전 버스 상태를 체크하고 송신되는 메시지가 있으면 중재 과정을 거쳐 메시지를 보내게 된다. 이 때, 우선순위에서 밀리게 되면 “Bus is busy!!!” 라는 메시지를 띄우면서 메시지를 전송하지 못하는데 DoS 공격 중에는 이 메시지가 계속 나타나게 된다. [그림 6] 에서 그러한 상황을 보여 주고 있다. 이 후 DoS 공격이 끝나면 메시지가 정상적으로 송신되는 것을 볼 수 있다. [그림 7] 에서는 공격 ECU가 지속적으로 메시지를 보내고 있는 상황을 보여주고 있으며 [그림 8] 에서는 수신 ECU가 DoS 공격 상황에서 공격 ECU의 메시지만을 수신 받는 것을 보여주고 있으며 이로 인해 송신 ECU에서 전송된 메시지를 받지 못하는 것을 확인 할 수 있다. 이 후 DoS 공격 종료 후에는 송신 ECU의 메시지가 전달되는 것을 보여주고 있다.

V.결론

본 논문에서는 Arduino를 이용하여 소규모 CAN 실험 환경을 구축해보았다. 이를 이용하여 CAN에서의 스푸핑, DoS 공격을 실험하였으며 공격이 성공하였음을 보였다. 구축된 소규모 실험 환경을 이용하여 CAN 실험을 한다면 비용과 편의성면에서 이점을 줄 수 있을 것으로 기대된다.

향후 연구에서는 이 논문에서 사용된 실험 환경을 이용하여 실제 자동차 ECU를 연결하여 실험을 진행해보고 이와 유사한 기존의 연구[7]와 비교를 진행하여 효율성과 비용면에서 얼마나 큰 이점을 가지는지 비교하는 연구와 구축된 실험 환경을 이용하여 스푸핑과 DoS 공격을 탐지하고 방어하는 추가적인 연구가 필요하다.

[참고문헌]

[1] Karl Koscher, *et al.* "Experimental security analysis of a modern automobile", Security and Privacy (SP), 2010 IEEE Symposium on. IEEE, 2010.

[2] Stephen Checkoway, *et al.* "Comprehensive Experimental Analyses of Automotive Attack Surfaces", USENIX Security Symposium. 2011.

[3] 김광조, 이동수, "escar 회의 등을 통한 각국의 자동차 보안 기술 동향 연구", 한국정보보호학회지 제 24권 제 2호 pp.7 - 20, 2014. 4

[4] Jun Yajima, *et al.* "White-List CAN Hub with Disable Functionality againt Attack Message", 2015 Symposium on Cryptography and Information Security (SCIS 2015), Jan. 20-23, 2015, Fukuoka, Japan.

[5] Yoshihiro Ujiie, *et al.* ”Proposal of CAN Filtering Technology for In-Vehicle Network”, Symposium on Cryptography and Information Security, 2015.

[6] Sato Michio, “자동차 네트워크 시스템”, 성안당

[7] 이혜련 외 5인. “자동차용 ECU의 CAN 메시지를 통한 자동차 공격 방법 연구”, 한국컴퓨터정보학회 제 18권 제 11호 pp.39 - 50, 2013. 11