Journal of the Korea Convergence Society
Vol. 8. No. 11, pp. 21-27, 2017 https://doi.org/10.15207/JKCS.2017.8.11.021

임시파일 데이터 조작을 통한 아두이노 보드 공격 기법에 관한 연구

이우호[1], 정현미[2], 정기문[2*]
[1]전남대학교 정보보안협동과정, [2]한국과학기술정보연구원 슈퍼컴퓨터개발센터

Research about Security Attack Methods to Arduino Boards Using Temporary Files Data Manipulation

Woo Ho Lee[1], Hyun Mi Jung[2], Kimoon Jeong[2*]
[1]Interdisciplinary Program of Information Security, Chonnam National University
[2]Center for Supercomputer Development, Korea Institute of Science and Technology Information

요 약 초연결사회를 지향하기 위해 발전하고 있는 사물인터넷(Internet of Things)은 아두이노 등의 OSHW (Open Source Hardware)를 기반으로 두고 있으며 다양한 소형 제품 등이 등장하고 있다. 이러한 사물인터넷은 저성능, 저메모리라는 한계로 인하여 강력한 보안 기술을 적용하기 어렵다는 심각한 정보보안 문제를 야기하고 있다. 본 논문에서는 사물인터넷 기기로 주로 사용되는 아두이노의 응용프로그램이 호스트컴퓨터에서 컴파일과 로딩이 수행됨에 따라 발생할 수 있는 취약성을 분석하여 아두이노 보드의 센서로부터 입력되는 값을 공격자가 임의로 변경할 수 있는 새로운 공격 방법을 제안한다. 이러한 방법을 통해 아두이노 보드가 환경정보를 오인식하여 정상적인 동작이 불가능하게 할 수 있다. 이러한 공격 기법의 이해를 통해 안전한 개발환경 구축방안을 고려할 수 있으며 이러한 공격으로부터 대응할 수 있다.

• **주제어** : 사물인터넷 보안, 아두이노, 오픈소스하드웨어, 공격기법, 해킹

Abstract Internet of Things(IoT), which is developing for the hyper connection society, is based on OSHW (Open Source Hardware) such as Arduino and various small products are emerging. Because of the limitation of low performance and low memory, the IoT is causing serious information security problem that it is difficult to apply strong security technology. In this paper, we analyze the vulnerability that can occur as a result of compiling and loading the application program of Arduino on the host computer. And we propose a new attack method that allows an attacker to arbitrarily change the value input from the sensor of the arduino board. Such as a proposed attack method may cause the arduino board to misinterpret environmental information and render it inoperable. By understanding these attack techniques, it is possible to consider how to build a secure development environment and cope with these attacks.

• **Key Words** : IoT security, Arduino, OSHW, Security attack, Hacking

*Corresponding Author : 정기문(kmjeong@kisti.re.kr)
Received September 26, 2017 Revised November 2, 2017
Accepted November 20, 2017 Published November 28, 2017

1. 서론

현재 사물인터넷은 다양한 분야에서 활용되고 있으며, 국내외 산업분야에서 응용분야가 빠르게 확대되고 있다. 특히 기존 사용되고 있는 홈 케어, 농업, 스마트 카 등의 분야는 시장의 규모가 크게 확장되고 있다[1].

스마트 홈의 경우 원격으로 가정 내에서의 센서를 이용한 방범과 검침 그리고 전력을 제어할 수 있다. 세계 주요국의 사물인터넷 시장은 빠르게 증가하고 있으며 가전제품, 자동차 그리고 헬스케어 기기들은 50%이상 사용량이 증가할 것이라고 전망되고 있다[2]. 이렇게 빠르게 증가하고 있는 사물인터넷은 아두이노(Arduino)[3], 라즈베리파이(Raspberry Pi)[4], 비글보드(Beagelboard)[5] 등과 같은 OSHW (Open Source HardWare)에 기반을 두고 있으며 다양한 소형 제품 등이 등장하고 있다. OSHW는 사용이 용이하기도 하지만 저메모리, 저성능으로 인하여 강력한 암호 및 인증 등이 어렵다는 같은 심각한 보안 문제를 야기하고 있다[6]. 최근에는 사물인터넷 기기를 봇넷으로 이용한 심각한 DDoS 공격이 급격히 증가하고 있는 실정이다[7,8].

사물인터넷 플랫폼으로 많이 사용되고 있는 아두이노 보드 또한 물리적인 훼손 등의 공격뿐만 아니라 아두이노 보드 메모리의 취약성을 이용한 공격[9]등이 발생하고 있으며 이에 대한 대응 방법 또한 제안되고 있다. 본 논문에서는 아두이노 실행파일을 생성하는 단계에서 발생하는 취약성 등을 이용한 새로운 공격기법에 대해서 소개하고자 한다.

아두이노는 다른 OSHW와 마찬가지로 실행파일 개발 시 IDE도구 등을 이용하여야 한다. 개발을 위하여 호스트 컴퓨터에서 소스코드가 작성되고 실행파일 생성을 위한 컴파일, 빌드 및 업로딩이 실행된다. 이러한 과정 중에 임시파일이 생성되게 되는데 이를 이용하여 필수적으로 사용되는 함수값 등을 조작하는 인젝션 공격 등이 가능하다. 이를 통해 센서에서 입력되는 값이 저장되는 레지스터 값 등을 변조하여 아두이노가 비정상적인 실행을 하도록 할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 아두이노 보드에 대한 간략한 내용과 이에 대한 공격 및 방어 기법의 동향에 대해 살펴본다. 3장에서는 아두이노 보드의 개발 환경 및 함수 처리 등에서 발생할 수 있는 취약성에 대해 설명한다. 4장에서는 위와 같은 취약성을 이용한 공격기법에 대하여 설명하며 5장에서 정리하고 결론을 맺는다.

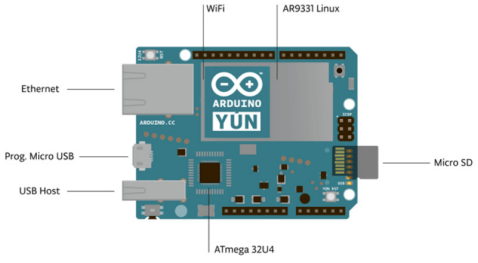
2. 관련연구

2.1 아두이노 보드

아두이노는 사용하기 쉬운 하드웨어 및 소프트웨어를 기반으로 하는 오픈소스 전자 플랫폼[10]으로서 저사양의 MCU(Micro Controller Unit)를 탑재하였고 센서와 액츄에이터를 이용할 수 있는 여러 개의 디지털 핀과 아날로그 핀이 있다. 디지털 핀은 자이로 센서 등의 장비에 응용되며 아날로그 회로를 이용한 센서는 조도, 온도, 습도 등을 측정할 수 있다[3].

아두이노 장치에서 사용하는 MCU는 아트멜(Atmel)에서 제작한 AVR계열로서 크게 AVR Tiny와 MEGA AVR, 그리고 AT90 AVR로 나눌 수 있다[11]. 이와 같은 다양한 MCU 중에서 주로 MEGA AVR을 이용하여 아두이노 보드를 제작한다[Fig 1].

아두이노 보드의 메모리는 EEPROM, SRAM, Flash Memory로 구성되며 MCU에 따라 아두이노 보드의 메모리 크기가 다르다. SRAM은 휘발성 메모리이며, 전원 공급이 끊기게 되면 데이터가 지워지게 된다. SRAM은 Flash Memory와 데이터를 공유하며 아두이노 IDE를 통하여 소스 코드의 각종 필요한 변수와 버퍼 등이 생성되는 공간이다. 하드웨어에 대한 직접적인 전원공급의 중단 및 디바이스 초기화 버튼(Reset)을 이용하여 메모리를 초기화할 수 있다.



[Fig. 1] Arduino board sample

2.2 아두이노 보안 공격 및 방어기법 동향

사물인터넷 디바이스의 급격한 증가에 따라 실제로 사물인터넷 기기가 악성코드에 감염되는 사례가 발견됐으며 하드웨어 공격에 대한 방어 기법들이 연구되어 지고 있다.

메모리가 한정적인 AVR 기반의 하드웨어를 대상으로 크기가 큰 공격 익스플로잇 페이로드를 사용할 수 없기

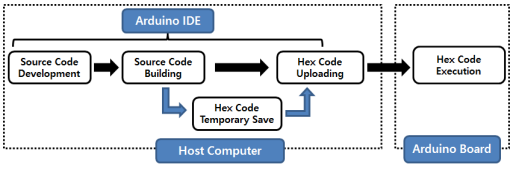
때문에 ROP(Return-oriented programming) 라이브러리 함수를 통해 추가적인 데이터를 RAM으로 주입하는 방법이 이용될 수 있다. ROP공격 기법은 리턴(ret) 명령어를 가지고 프로그래밍을 해서 공격을 하는 기법이며 DEP(Data Execution Prevention)와 ASLR(Address Space Layout Randomization), ASCII ARMOR 와 같은 보호 방법이 적용되어 있더라도 공격이 가능하다[12].

AVR 기반 하드웨어는 함수 인자를 레지스터를 통해 저장하고 인자가 많을 경우 스택에 저장하게 되는데 공격자는 SRAM의 데이터를 변경하여 공격할 수 있다. [9]에서 Habibi 등은 빌드 후 플래쉬 메모리에서 SRAM으로 데이터가 넘어가는 것을 이용한 코드 재사용 공격에 성공하였으며 레지스터를 수정하여 무인자동차의 자이로스코프 센서에 대한 컨트롤이 가능함을 보여주었다.

기존의 코드 재사용 공격의 대안으로 ASLR(Address Space Layout Randomization)[13]기법이나, 프로그램 제어 무결성 보장 기법[14] 등이 있지만 일반적으로 AVR 디바이스와 같이 자원이 한정적인 시스템은 큰 오버헤드를 초래하기 때문에 적합하지 않아 실행코드가 실행되지 못하게 막는 DEP 방어가 기본적으로 제공되고 있다. 최근에는 별도의 하드웨어를 추가하거나, 기존의 하드웨어를 수정함으로서 코드 재사용 공격을 못하게 하는 방법이 연구되고 있지만 추가적인 비용이 든다는 단점이 있다[15].

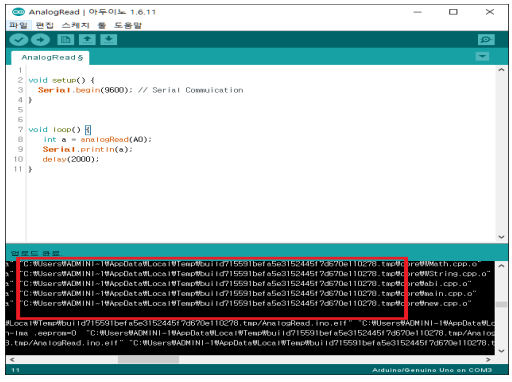
3. 아두이노 보드 취약성 분석

본 장에서는 아두이노를 이용하기 위한 프로그램 개발 환경 및 아두이노 보드에 있는 센서를 통해 입력 받은 정보가 처리되는 방식에 대해서 분석하여 아두이노의 취약성에 대해 알아본다.



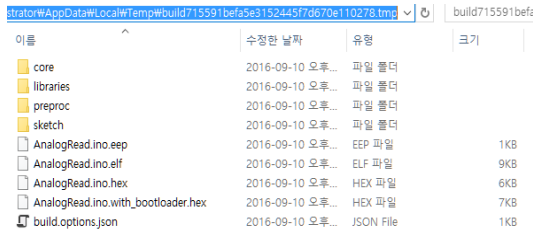
[Fig. 2] Process of arduino program execution

아두이노 프로그램은 [Fig. 2]와 같은 과정을 통해서 아두이노 보드에서 실행될 수 있다. 먼저 아두이노 프로그램을 개발하기 위한 IDE를 설치하여 소스코드 작성 및 컴파일과 링크 등 빌드과정을 수행한다. 그리고 빌드된 실행코드는 업로드 과정을 통해 아두이노 보드에 로딩되어 실행된다. 소스코드가 호스트 컴퓨터에서 빌드되면서 실행 가능한 Hex 코드가 임시로 저장되며 이를 이용하여 업로딩이 이루어진다. Windows 환경에서 실행된 아두이노 IDE의 경우 임시파일이 저장되는 경로는 [Fig. 3]과 같다.



[Fig. 3] Path of arduino's temporary files

Windows 운영체제의 사용자 임시 디렉터리(예: C:\Users\사용자명\AppData\Local\Temp)에 "프로젝트 명칭의 알파벳명칭+임시번호.tmp" 라는 이름의 디렉터리가 임시파일이 저장되는 위치다. 아두이노의 임시파일은 프로젝트 파일 생성 위치에 상관없이 고정된다. 컴파일 시마다 임시파일은 새롭게 재생성, 수정되며 임시파일 디렉터리에 저장된 임시 Hex코드는 ICSP(In Circuit Serial Programming) 업로드 장치를 통해 직접 업로드해도 실행이 가능하다.[16]. 생성된 임시 디렉터리에는 [Fig. 4]와 같이 core, sketch, preproc 등의 파일이 저장된다.



[Fig. 4] Generation of temporary file path

이와 같이 저장되는 Hex 코드에 대한 조작을 통하여 공격자가 원하는 내용의 실행코드를 아두이노 보드에 업로드 시킬 수 있는 취약성이 있다.

다음으로 아두이노 함수의 조작 취약성을 확인하기 위해 센서입력 처리를 하는 아두이노 함수의 동작과 관련된 레지스트리 및 메모리에 대한 분석을 수행한다.

analogRead()는 아두이노 보드의 아날로그 핀으로부터 입력받는 값을 처리하는 함수이다. 센서로부터 입력된 습도, 조도 등의 아날로그 값은 0V~5V 사이의 전압값으로 인식되며 전압에 따라 1024등분되어 0에서 1023까지 값으로 반환한다. 이와 같이 아날로그 값을 디지털 값으로 변환해주는 모듈을 ADC(Analog to Digital Converter)라고 하며 아두이노가 자체적으로 가지고 있다. analogRead()함수를 사용하기 위해서는 레지스트리를 이용하는데 이에 대한 처리 값은 아두이노 임시파일 디렉토리의 analogRead.hex 파일에 저장된다.

analogRead()함수는 5가지 종류의 레지스터를 이용하여 데이터를 계측하는데 다음과 같다. ADCL(ADC Data Register - Low), ADCH(ADC Data Register - High), ADCSRA(ADC Control and Status Register A), ADCSRB(ADC Control and Status Register B), ADMUX(ADC Multiplexer Selection Register). 이 중에서 ADCH, ADCL 레지스터는 센서를 통해 입력받는 값의 최대값과 최소값으로 각각 2진수로 변환되어 16비트로 결합하여 analogRead()함수의 반환 값으로 이용된다. ADMUX 레지스터는 ADC의 입력 핀을 선택하는 기능과 기준 전압원을 선택하거나 변환 결과를 레지스터에 저장하는 형식을 지정하는 기능을 수행한다. ADCSRA 레지스터는 ADC의 전반적인 동작을 제어하는 기능을 수행한다.

본 논문에서는 ADCH, ADCL 레지스터의 값을 임의로 변조하여 오작동을 일으키는 것을 목표로 한다.

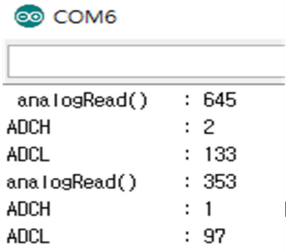
[Fig. 5]는 analogRead()에 대한 함수를 이용한 간단한 출력 소스코드이다. 아두이노 보드의 아날로그 핀인 “A0”로 부터 입력되는 센서값의 최대값과 최소값을 확인하기 위하여 ADCH와 ADCL 레지스터 값을 출력한다. 그리고 두 값을 결합한 analogRead() 함수값을 출력한다.

```
void analogRead_registy() {

    uint8_t high, low; //ADC레지스터의 값을 담을 hight,low 변수 선언
    int Aread_sensor = analogRead(A0);
    high = ADCH;
    low = ADCL;
    Serial.print("analogRead()  : ");
    Serial.println(Aread_sensor);
    Serial.print("ADCH      : ");
    Serial.println(high);
    Serial.print("ADCL      : ");
    Serial.println(low);
    return;
}
```

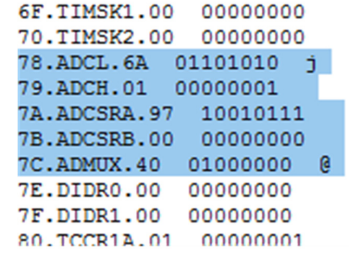
[Fig. 5] analogRead () function’ s source code

위 소스코드를 실행한 예제는 [Fig. 6]에서 확인할 수 있다. analogRead() 함수값은 645(0000 0010 1000 0101)로서 ADCH 값 2(0000 0010)과 ADCL값 133(1000 0101) 값을 8비트씩 결합하여 구성한 16비트 값임을 알 수 있다. 레지스터는 일정한 주기마다 데이터를 읽어 오고 있으며 ADCH와 ADCL 레지스터 값을 순서대로 출력하는 것을 알 수 있다.



[Fig. 6] Example of sample code execution

아두이노 보드의 메모리를 분석하여 함수 값 조작을 할 수 있는 취약성을 알아보기 위하여 아두이노 IDE의 시리얼모니터링을 통해 확인한 내용은 [Fig. 7]과 같다. 0x78 메모리부터 순차적으로 ADCL, ADCH, ADCSRA, ADCSRB, ADMUX 레지스터 값이 저장됨을 알 수 있다.



[Fig. 7] analogRead() function’s register data

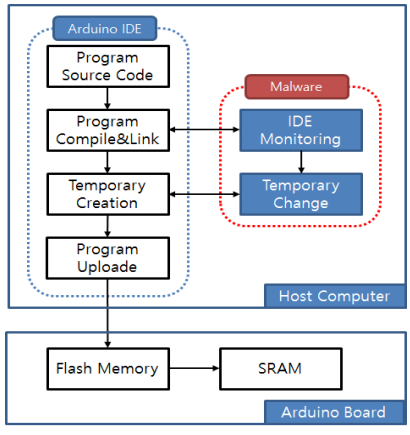
이와 같은 레지스트리값은 Hex코드 업로딩 과정에서 아두이노 보드의 플래쉬메모리와 SRAM에 입력되어 아두이노 프로그램이 실행된다.

따라서 아날로그 센서로부터 입력되는 ADCL, ADCH 두 개의 레지스터 값을 조작하게 되면 아두이노 보드는 센서를 통해 입력된 습도, 조도, 고도 등의 환경 정보를 오인식하여 잘못된 처리를 수행하게 된다. 예를 들어 조작된 습도 값이 입력된 스마트 팜은 농작물 관리에 실패할 수 있고 고도 값이 조작된 드론의 경우 오작동으로 인하여 추락하게 될 수 있다.

4. 함주 조작에 의한 공격 기법

호스트 컴퓨터를 기반으로 아두이노 IDE를 사용하는 개발환경에서 임시파일이 생성된다는 취약성이 있음을 확인하였다. 또한 센서 입력값을 처리하는 함수에 대한 분석을 수행하여 센서 입력값이 메모리에 어떻게 저장되는지에 대해서도 살펴보았다.

이러한 점을 이용하여 본 논문에서는 아두이노 보드에서 실행되는 프로그램을 변조하여 공격할 수 있는 새로운 방법을 제안한다. 제안하는 공격과정은 [Fig. 8]과 같다.



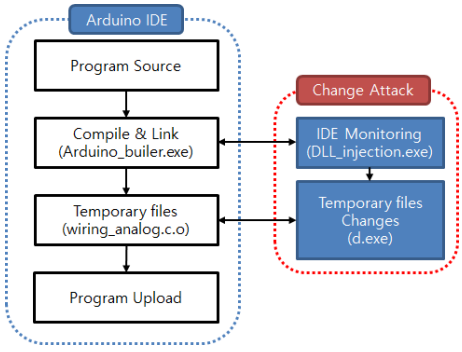
[Fig. 8] Process of arduino attack method

호스트컴퓨터에 설치되어 있는 악성코드는 [IDE 모니터링] 기능을 통하여 아두이노 IDE관련 과정을 실시간으로 모니터링한다. 모니터링 과정 중에 프로그램을 컴파일하는 프로세스가 인식되면 [임시파일 변조] 기능을 통해 컴파일되어 임시로 저장되어 있는 파일을 찾아 공격자가 원하는 동작을 수행하도록 데이터를 변조할 수 있다.

본 논문에서 제시하는 방법은 아두이노 보드가 아닌 컴파일 및 빌드 과정 중에 컴파일되는 파일을 재사용하여 변조하기 때문에 조기에 발견하기가 어렵다.

이와 같은 기능을 수행하는 악성코드가 먼저 호스트 컴퓨터에 설치되어야 하는 제약이 있지만 이는 컴퓨터에 악성코드를 감염시키는 다양한 방법으로 충분히 가능할 수 있다.

Windows 환경에서 아두이노 IDE를 설치하여 제안된 방법을 구현한 사례는 [Fig. 9]에서 볼 수 있다.



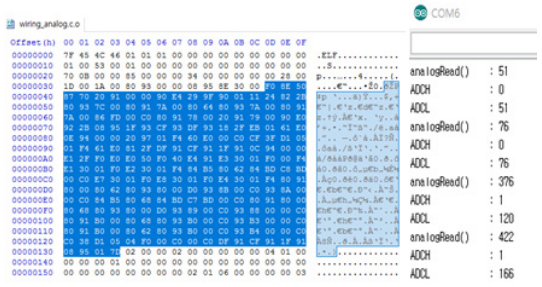
[Fig. 9] Example of arduino attack method

[IDE 모니터링] 기능을 수행하는 Dll_injection.exe 파일은 아두이노 IDE에서 실행되고 있는 Arduino_builder.exe에 의해서 실행되는 프로세스를 실시간으로 모니터링한다. 이를 통해 아두이노 프로그램이 컴파일이 실행되는 순간을 알아낼 수 있다.

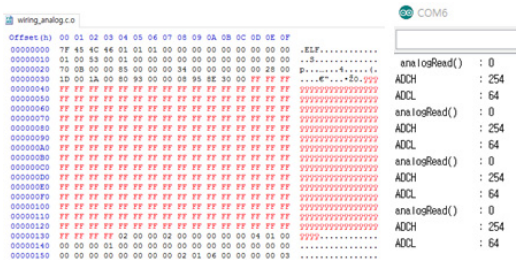
[임시파일 변조] 기능을 수행하는 d.exe 파일은 아두이노 프로그램이 컴파일되어 호스트컴퓨터에 임시로 생성된 wiring_analog.c.o 파일을 변조하는 기능을 수행한다. analogRead()함수의 ADCH와 ADCL 레지스터의 값을 “0xFF”로 초기화하는 기능을 수행한다. 이를 통해 임시파일의 레지시터 값 조작이 가능함을 보여준다.

임시파일을 변조 시도하는 프로그램이 적용되기 이전의 wiring_analog.c.o 파일의 내용은 [Fig. 10]과 같다. ADCH와 ADCL 레지스터의 값이 설정되어 있음을 알 수 있다.

반면 변조시도 예제파일을 실행한 경우의 wiring_analog.c.o 파일의 내용은 [Fig. 11]과 같이 모두 “0xFF” 값으로 바뀌어 있음을 알 수 있다.



[Fig. 10] Status of wiring_analog.c.o(before)



[Fig. 11] Status of wiring_analog.c.o(after)

아두이노 보드의 펌웨어는 컴파일에 의해 새롭게 생긴 빌드파일을 새로운 빌드 파일로 인식되기 때문에 아두이노 보드는 악성코드 감염을 인식하지 못한다. 감염을 파악하고 Reset 버튼을 이용해 아두이노 보드 하드웨어를 초기화하더라도 부트로더가 감염되었을 경우 코드가 계속 EEPROM에 남아있을 수 있다. 이에 따라 다양하고 연속적인 공격(사용자 정보탈취, 센서 디바이스 공격)이 가능하며 하드웨어가 인터넷과 연결될 경우 정보 하드웨어에 대한 사용자 정보와 빌드 정보를 가로 챌 수도 있다.

이와 같은 공격으로 인해 아두이노 보드는 변조된 센서 입력정보에 따라 동작하게 되면 전체 시스템에 심각한 장애를 초래할 수 있다. 앞에서는 아날로그 센서 데이터만을 처리하는 analogRead()함수 만을 다루었지만 자이로 센서와 같은 디지털 센서 입력정보도 동일한 방법으로 변조할 수 있다. 드론의 경우 자이로센서의 입력값을 처리하기 위해 digitalRead() 함수를 사용하는데 여기에 사용되는 레지스터 값을 조작하면 드론은 드론의 위치정보를 오인하여 정상적인 운행을 못하게 된다.

5. 결론

본 논문은 AVR 보드 중 많이 사용되고 있는 아두이노 개발 과정에서 발생할 수 있는 취약점에 대해 분석하였고 빌드과정 중 임시파일에서 하드웨어의 레지스트리 정보를 획득할 수 있었다. 또한 정보를 바탕으로 사물인터넷 디바이스의 하드웨어 정보 및 센서 디바이스에 대한 레지스터 조작 및 악성코드를 이용한 오작동을 확인할 수 있었다. 본 공격은 감염된 빌드파일의 경우 감염된 하드웨어를 발견하기 어려우며, 좀비 PC에 이은 좀비 하드웨어가 생성될 수 있다는 점에 위험성이 높다. 또한 현재 많이 사용되는 아날로그 함수(analogRead())함수 뿐만 아니라 디지털 함수(DigitalRead())까지도 공격이 가능하였다.

아두이노를 기반으로 한 스마트 팜 그리고 스마트 홈, 드론에 대해 개발자 PC를 감염시키는 실험을 하였으며, 그 결과 제어디바이스에 대한 오작동과 데이터 전송 시 잘못된 정보를 전송하는 것을 확인할 수 있었다. 따라서 개발도구를 사용할 때 임시파일에 대한 접근권한을 차단하고 개발자는 임시파일에 대한 코드확인과 레지스터 변조여부에 대한 확인이 필요하다.

향후 연구에서는 악성코드를 이용한 좀비 하드웨어에 대한 연구와 본 논문에서 제시하는 취약점에 대한 암호화에 대한 연구를 진행할 것이며, 개발환경에서 이미 감염된 기기들을 분석할 수 있는 백신프로그램에 대한 연구를 할 것이다. 마지막으로 임시파일이 생성되고 파일의 재사용을 막기 위한 방어기법과 암호화 기법에 대해서 연구할 것이다.

ACKNOWLEDGMENTS

본 연구는 (2017년도) 한국과학기술정보연구원(KISTI) 주요사업 과제로 수행한 것이다

REFERENCES

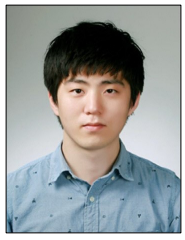
[1] H. S. Ryu, "A Study on the Security Architecture for Secure Smart Home System in IoT", Department of Computer Engineering, Ajou University, 12. 2015.

[2] "The Internet of Things: The Next Growth Engine for the Semiconductor Industrt." PWC, 2015, 3. pp. 23,26.

[3] https://www.arduino.cc/en/Guide/Introduction..
[4] https://www.raspberrypi.org/.
[5] https://beagleboard.org/.
[6] Matthew Ahlmeyer, Alina M. Chircu, ,“SECURING THE INTERNET OF THINGS: A REVIEW”, Issues in Information Systems, Volume 17, Issue IV, pp. 21-28, 2016
[7] https://security.radware.com/ddos-threats-attacks/threat-advisories-attack-reports/mirai-botnet/.
[8] Alexander Khalimonenko, Oleg Kupreev, "DDOS attacks in Q1 2017", Securelist, 05. 2017
[9] Javid Habibi, Aditi Gupa, Stephen Carlsony, Ajay Panicker, “MAVR : Code Reuse Stealthy Attacks and Mitigation on Unmanned Aerial Vehicles,” 2015 IEEE 35th International Conference on Distributed Computing Systems, 2015.
[10] Massimo Banzi, “Arduino, Open Source Hardware Summit Speech”, OSHW Summit, 09.2011.
[11] http://www.atmel.com/products/microcontrollers/avr/default.aspx
[12] Lucas Davi, Ahmad-Reza, “ROP defender: A detection tool to defend against return-oriented programming attacks”, System Security Lab, Ruhr University Bochum, Germany, 03, 2010.
[13] Ralf Hund, Carsten Willems, “Practical Timing Side Channel Attacks against Kernel Space ASLR,” 2013 IEEE Symposium on Security and Privacy, pp. 191-205, 2013.
[14] Martin Abadi, Mihai Budiu, “Control-Flow Integrity Principles, Implementations, and Applications,” ACM Transactions on Information and System Security, Vol. 13, No. 1, Article 4, pp. 1-40, 2009.
[15] Sergio Pastrana, “AVRAND: A Software-Based Defense Against Code Reuse Attacks for AVR Embedded Devices”, DIMVA, 07.2016.
[16] W. H. Lee, S. M. Kang, C. S. Lim, B. N. Noh, “Research on Memory Initialization through Using Ardunio Temporary Files,” KIPS 2016, Vol 23, No 2, 2016.

저자소개

이 우 호(Woo Ho Lee) **[정회원]**



- 2016년 2월 : 순천대학교 정보통신공학전공 (학사)
- 2016년 3월~현재 : 전남대학교 정보보안협동과정(석사과정)

<관심분야> : 네트워크 보안, IoT 보안

정 현 미(Hyun Mi Jung) **[정회원]**

- 2014년 2월 : 한남대학교 컴퓨터공학전공(공학박사)
- 2012년 10월~현재 : 한국과학기술정보연구원 슈퍼컴퓨터시스템개발실 선임연구원

<관심분야> : HPC, HPC 보안, 클라우드 컴퓨팅

정 기 문(Kimoon Jeong) **[정회원]**

- 2009년 8월 : 전남대학교 정보보호협동과정 (이학박사)
- 2001년 7월~2004년 12월 : 한국정보보호진흥원 연구원
- 2004년 12월~2005년 5월 : 국가사이버안전센터 연구원
- 2005년 6월~현재 : 한국과학기술정보연구원 선임연구원

<관심분야> : 네트워크 보안, IoT 보안, 클라우드 보안, 서버HW 보안