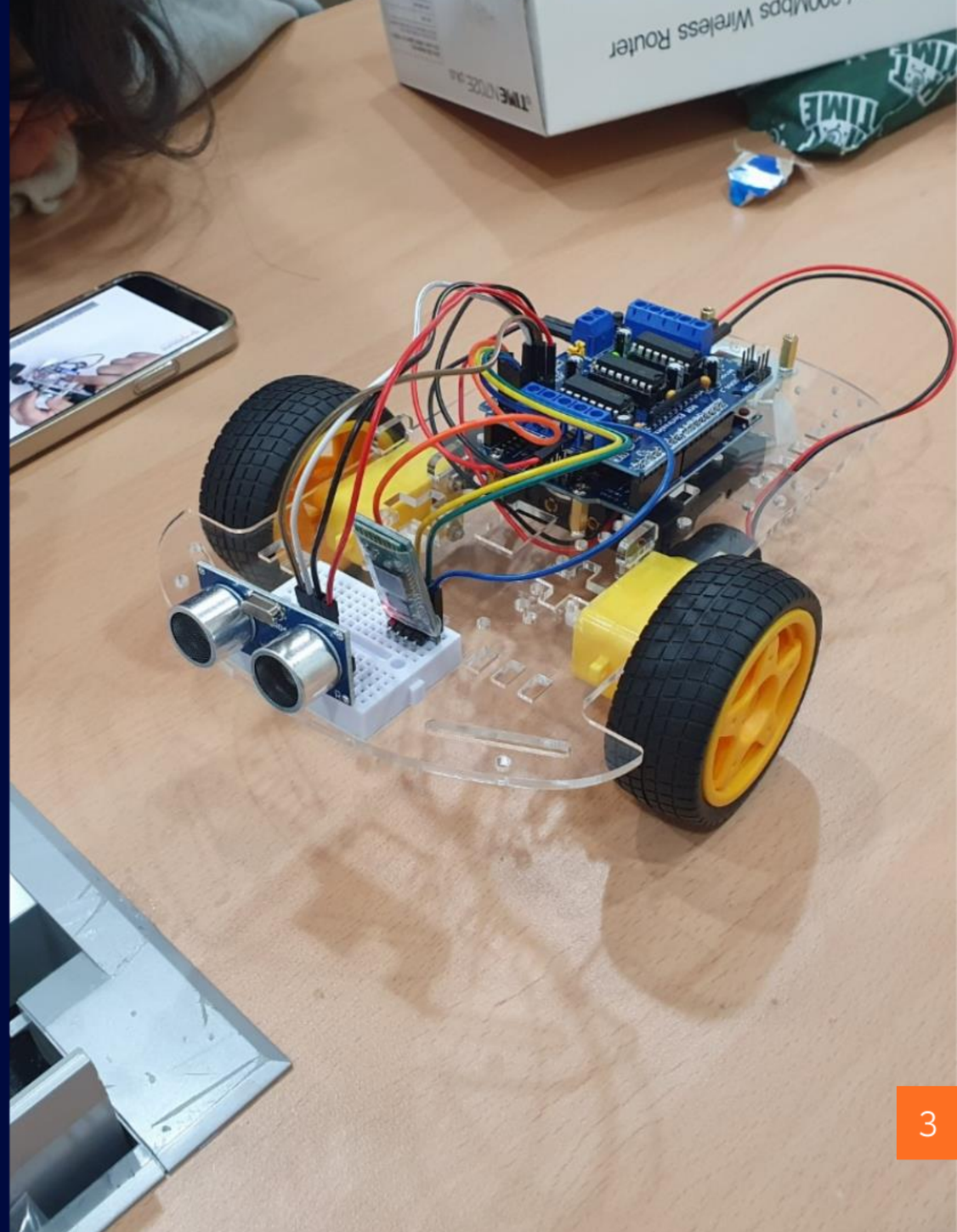
아두이노 취약점을 이용한 자율주행자동차 해킹 실습

2023.02.24
INTERLUDE

김유진, 박채령, 이가연, 한아림, 한재영

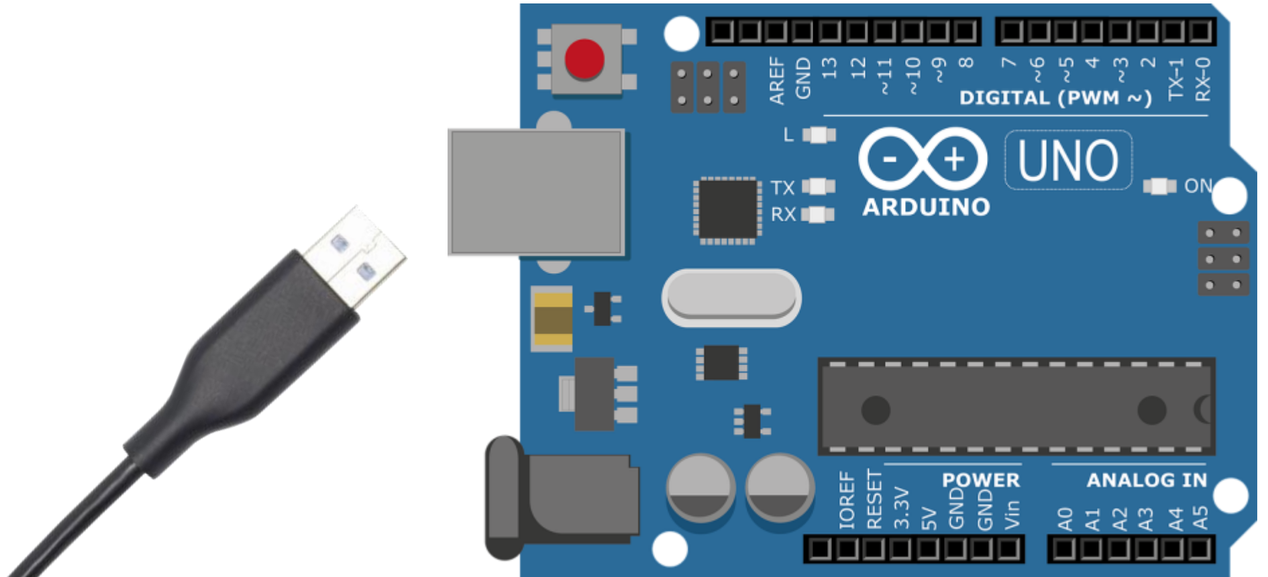
Table of Contents

1. 아두이노 기반 자율주행 자동차 제작



아두이노

- 아두이노(Arduino)는 오픈소스를 기반으로 한 단일 보드 마이크로컨트롤러
- 저렴한 가격과 간단한 개발환경으로 접근성이 좋음
- 위와 같은 장점으로 IoT, 로봇 등 다양한 프로젝트에 아두이노를 사용



사용한 센서

- 초음파 센서(HC-SR04)
  - 자동차 앞의 장애물과의 거리 측정에 사용

- 블루투스 모듈(HC-06)
  - 스마트폰을 이용한 원격 조종 기능 구현에 사용

- 모터드라이버 쉴드(L293D)
  - DC 모터 제어에 사용

장애물 회피 주행 코드

Distance_Mesurement() // 앞의 장애물과의 거리를 감지하는 함수

```
////////거리감지//////////
void Distance_Measurement() {
  digitalWrite(TrigPin, LOW);
  delay(2);
  digitalWrite(TrigPin, HIGH);  // trigPin에서 초음파 발생(echoPin도 HIGH)
  delayMicroseconds(10);
  digitalWrite(TrigPin, LOW);
  duration = pulseIn(EchoPin, HIGH);    // echoPin 이 HIGH를 유지한 시간을 저장
  distance = ((float)(340 * duration) / 1000) / 2;
  delay(50);
}
```

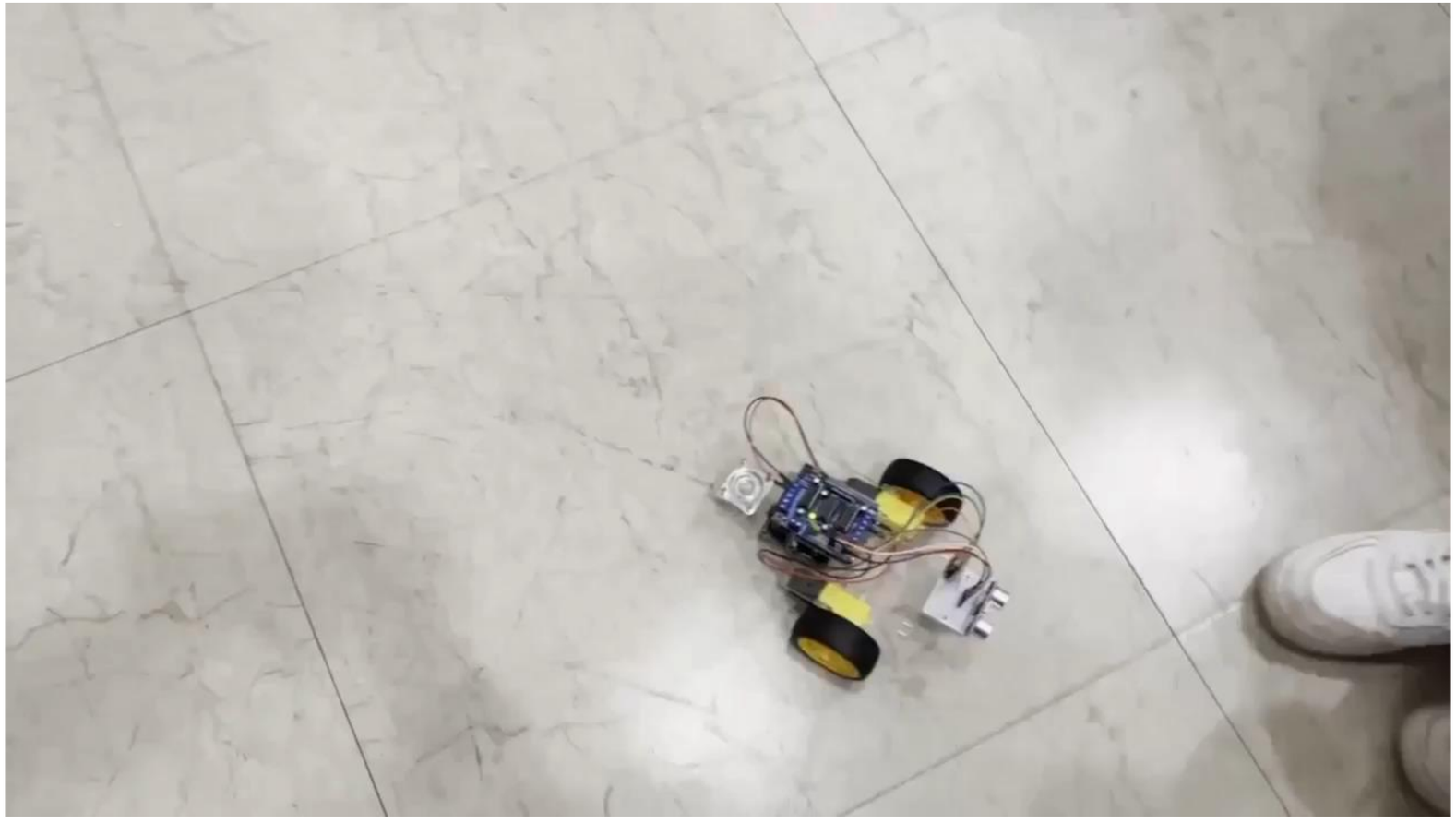
장애물 회피 주행 코드

Obstacle_Check() : 장애물 확인 및 회피 방향 결정

```
////////////장애물 확인 및 회피 방향 결정//////////
void Obstacle_Check() {
  int val = random(2);
  Distance_Measurement();
  delay(50);

  Serial.println(distance);

  while (distance < 300) {
    if (distance < 150) {
      Backward();
      delay(800);
      Stop();
      delay(50);
      Distance_Measurement();
      delay(100);
    }
```

```
      else {
        if (val == 0) {
          Right();
          delay(400);
        }
        else if (val == 1) {
          Left();
          delay(400);
        }
        Distance_Measurement();
        delay(100);
      }
    }
  }
```

주행 영상

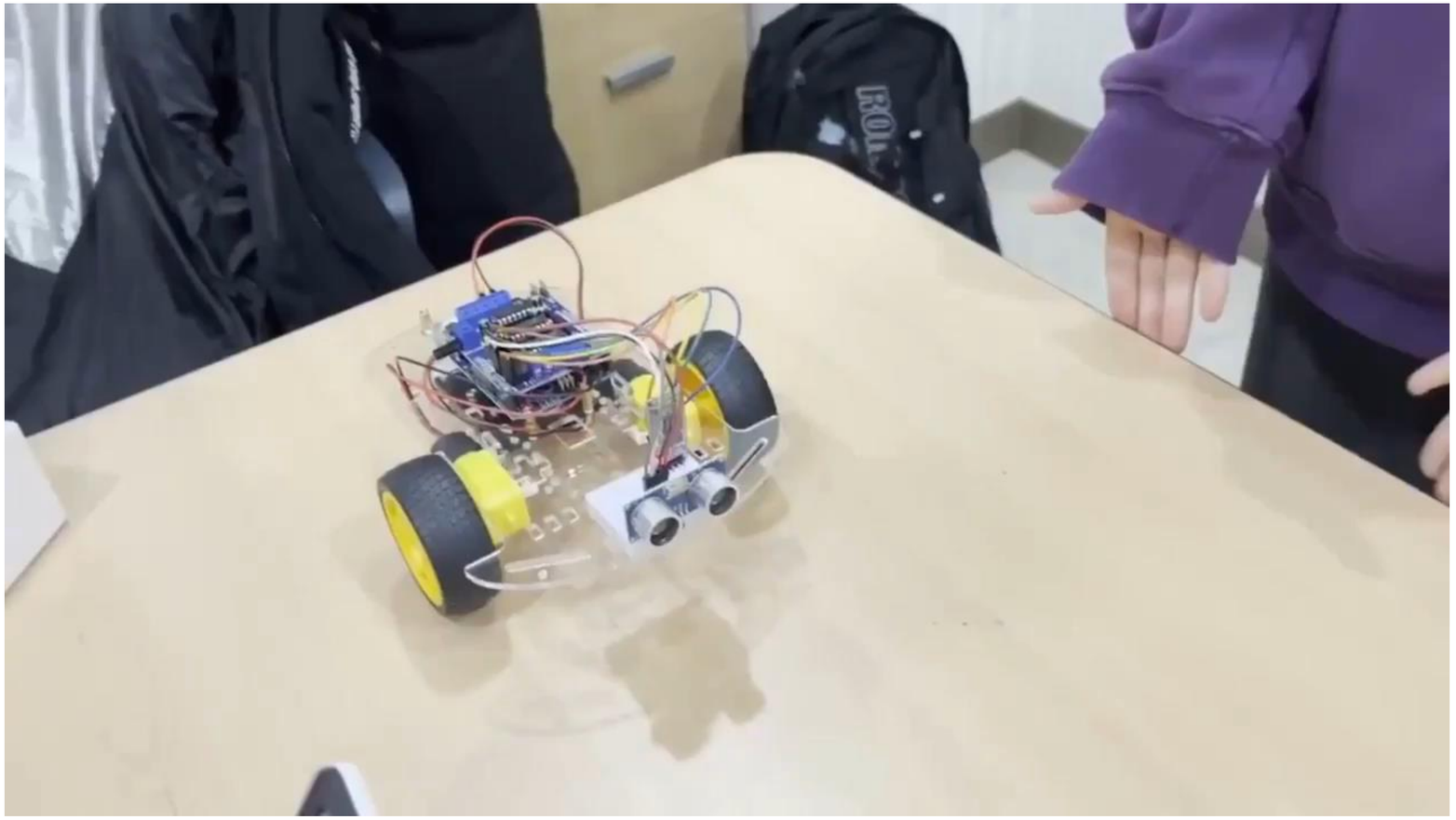


스마트폰 원격 조종 코드

```
void loop(){

  if(bluetooth.available()){          // 블루투스 명령 수신
     rec_data = bluetooth.read();
     Serial.write(rec_data);
     rec_chk = true;
  }

  if(rec_data == 'g'){  // 전진, go
     motor_L.run(FORWARD);  motor_R.run(FORWARD);
  }
  else if(rec_data == 'b'){ // 후진, back
     motor_L.run(BACKWARD);  motor_R.run(BACKWARD);
  }
  else if(rec_data == 'l'){ // 좌회전, Go Left
   motor_L.run(RELEASE);  motor_R.run(FORWARD);
  }
  else if(rec_data == 'r'){ // 우회전, Go Right
    motor_L.run(FORWARD);  motor_R.run(RELEASE);
  }
```

```
else if(rec_data == 'q'){ // 제자리 회전, Left Rotation
   motor_L.run(BACKWARD);   motor_R.run(FORWARD);
}
else if(rec_data == 'w'){ // 제자리 회전, Right Rotation
    motor_L.run(FORWARD);   motor_R.run(BACKWARD);
}
else if(rec_data == 's'){ // Stop
  motor_L.run(RELEASE);       motor_R.run(RELEASE);
  }


if(rec_data == 'f' ){       // 정지
  if(rec_chk == true){
     for (i=250; i>=0; i=i-20) {
        motor_L.setSpeed(i);  motor_R.setSpeed(i);
        delay(10);
     }

  }
}
```

주행 영상



논문 소개

임시파일 데이터 조작을 통한
아두이노 보드 공격 기법에 관한 연구

- 사물인터넷 기기로 주로 사용되는 아두이노의 응용프로그램이 호스트컴퓨터에서 컴파일과 로딩이 수행됨에 따라 발생할 수 있는 취약성을 분석

- 아두이노 보드의 센서로부터 입력되는 값을 공격자가 임의로 변경할 수 있는 새로운 공격 방법을 제안

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 비정상적인 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

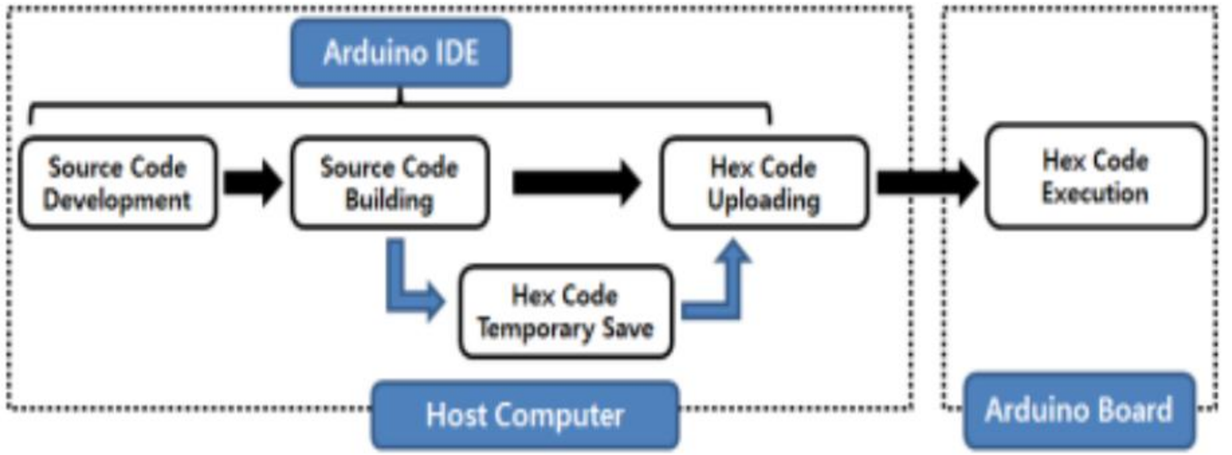
Abstract Internet of Things(IoT), which is developing for the hyper connection society, is based on OSHW (Open Source Hardware) such as Arduino and various small products are emerging. Because of the limitation of low performance and low memory, the IoT is causing serious information security problem 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

이우호, 정현미, 정기문
전남대학교 정보보안협동과정,
한국과학기술정보연구원 슈퍼컴퓨터개발센터

아두이노 보드 취약성 분석

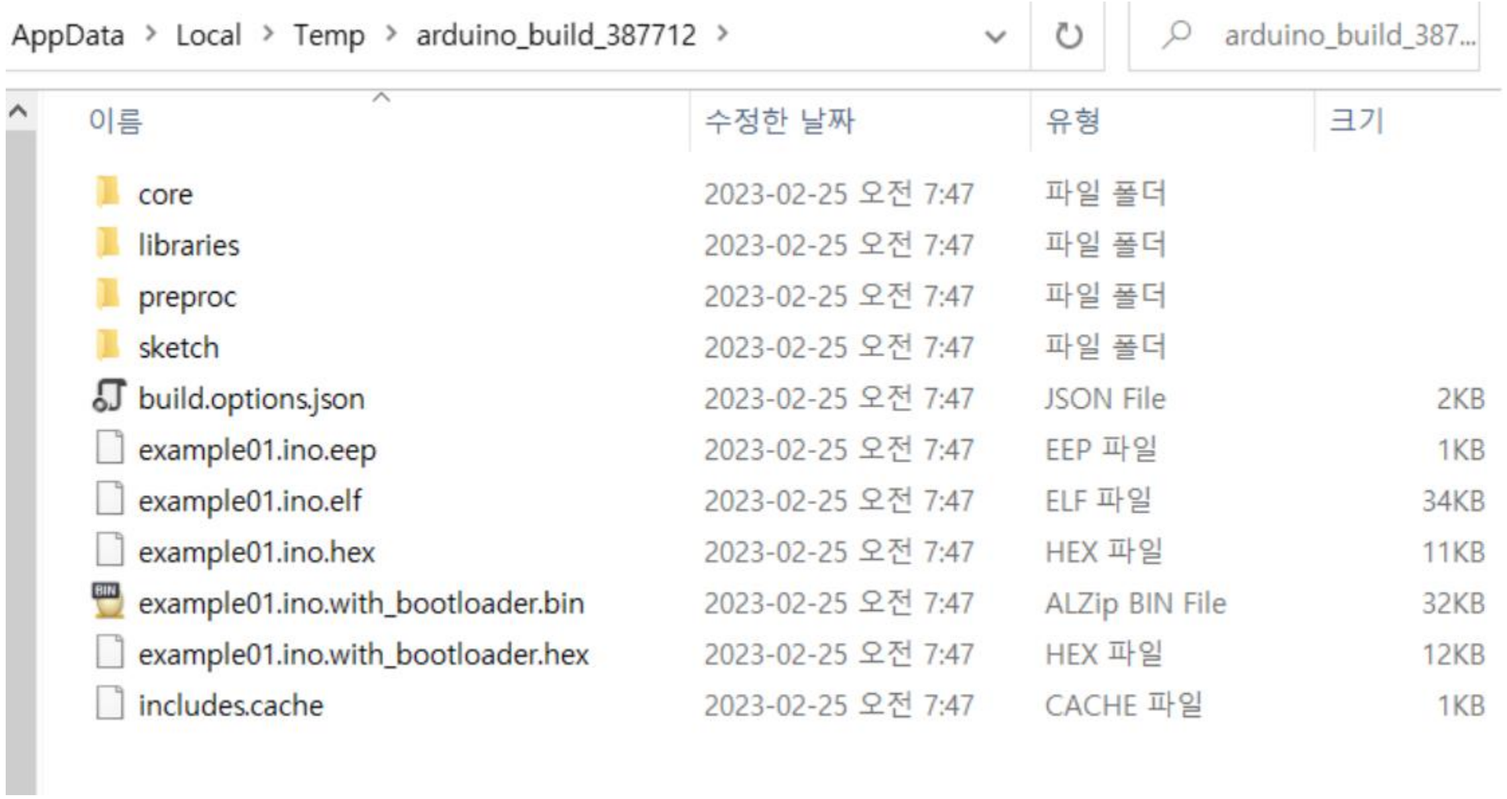


[Fig. 2] Process of arduino program execution

- 프로그램 개발을 위한 IDE 설치 및 소스코드 작성 및 빌드 과정을 수행
- 빌드된 실행코드는 업로드 과정을 통해 아두이노 보드에 로딩
- 소스코드가 빌드 되면서 실행 가능한 Hex 코드가 임시로 저장되며 이를 이용하여 업로딩
- Windows 운영체제의 사용자 임시 디렉터리에 프로젝트의 알파벳명칭+임시번호.tmp” 라는 이름의 디렉터리가 임시파일이 저장되는 위치

아두이노 보드 취약성 분석

- 저장되는 Hex 코드에 대한 조작을 통하여 공격자가 원하는 내용의 실행코드를 아두이노 보드에 업로드 시킬 수 있는 취약성이 존재



아두이노 보드 취약성 분석

- analogRead() : 보드의 아날로그 핀으로부터 입력받는 값을 처리하는 함수

- analogRead()함수는 5가지 종류의 레지스터를 이용하여 데이터를 계측
  - 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)

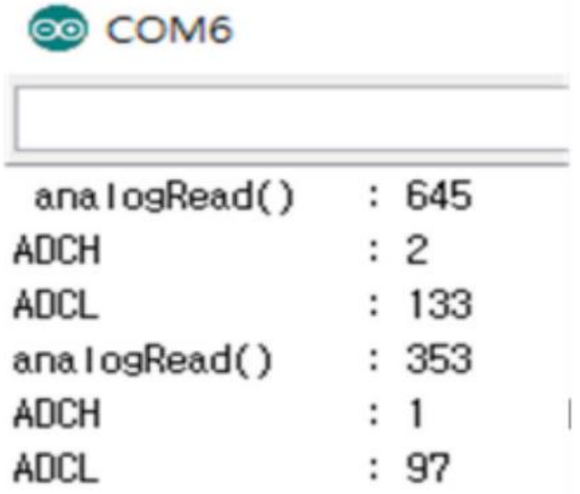
이 중 ADCH, ADCL 레지스터는 센서를 통해 입력받는 값의 최대값과 최소값으로 각각 2진수로 변환되어 16비트로 결합하여 함수의 반환 값으로 이용

아두이노 보드 취약성 분석

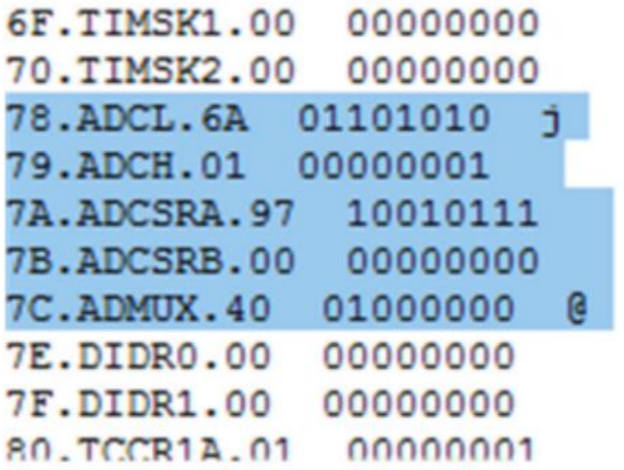
- ADCH 값 2(0000 0010)
- ADCL값 133(1000 0101)

- analogRead() 함수값 : 645
  (0000 0010 1000 0101)
- 0x78 메모리부터 순차적으로 ADCL, ADCH,
  ADCSRA,ADCSRB, ADMUX 레지스터 값이 저장

ADCL, ADCH 두 개의 레지스터 값을 조작하게 되면 아두이노 보드는
센서를 통해 입력된 환경 정보를 오인식하여 잘못된 처리를 수행



[Fig. 6] Example of sample code execution

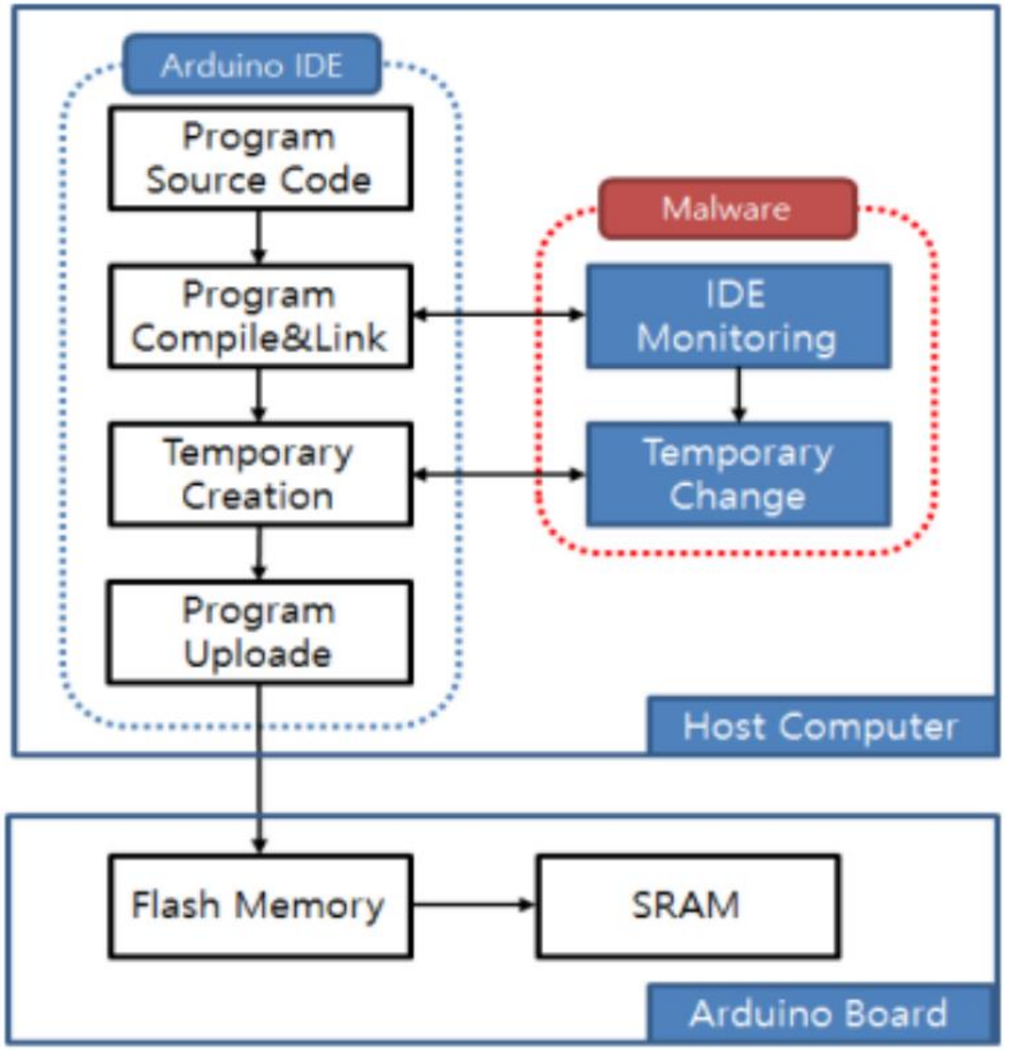


[Fig. 7] analogRead() function's register data

함수 조작에 의한 공격 기법

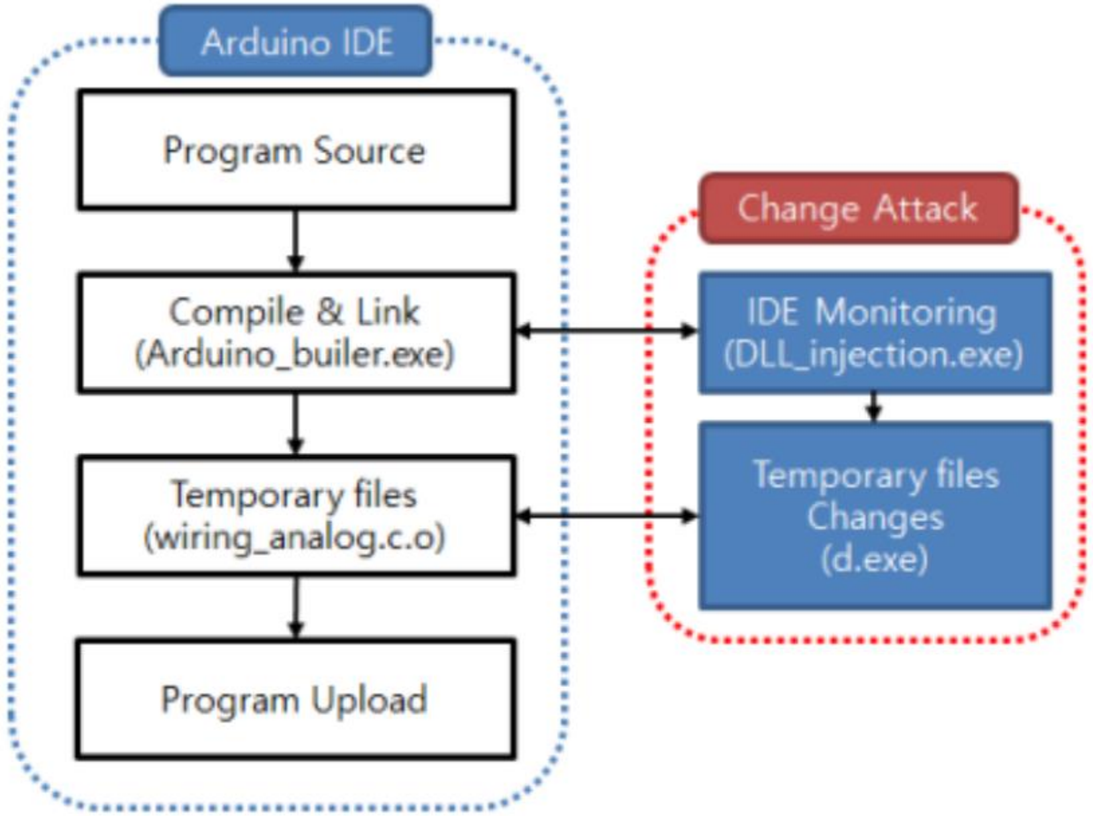
(1). 호스트컴퓨터에 설치된 악성코드는
[IDE 모니터링] 기능을 통하여 아두이노 IDE를 실시간으로 모니터링

(2). 모니터링 중 컴파일 프로세스가 인식되면
[임시파일 변조] 기능으로 임시 파일을 찾아 데이터변조



[Fig. 8] Process of arduino attack method

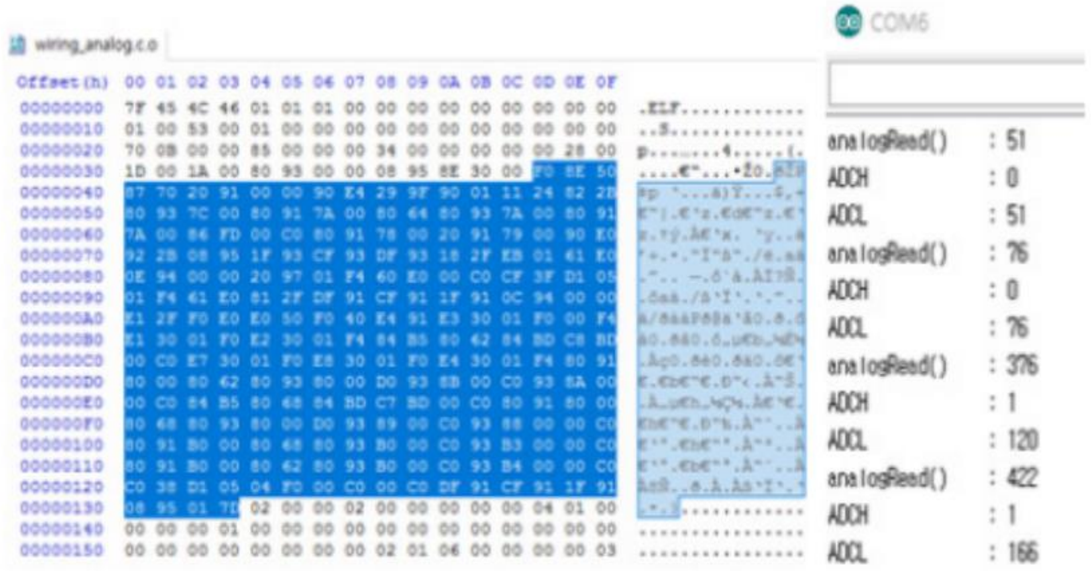
함수 조작에 의한 공격 기법



[Fig. 9] Example of arduino attack method

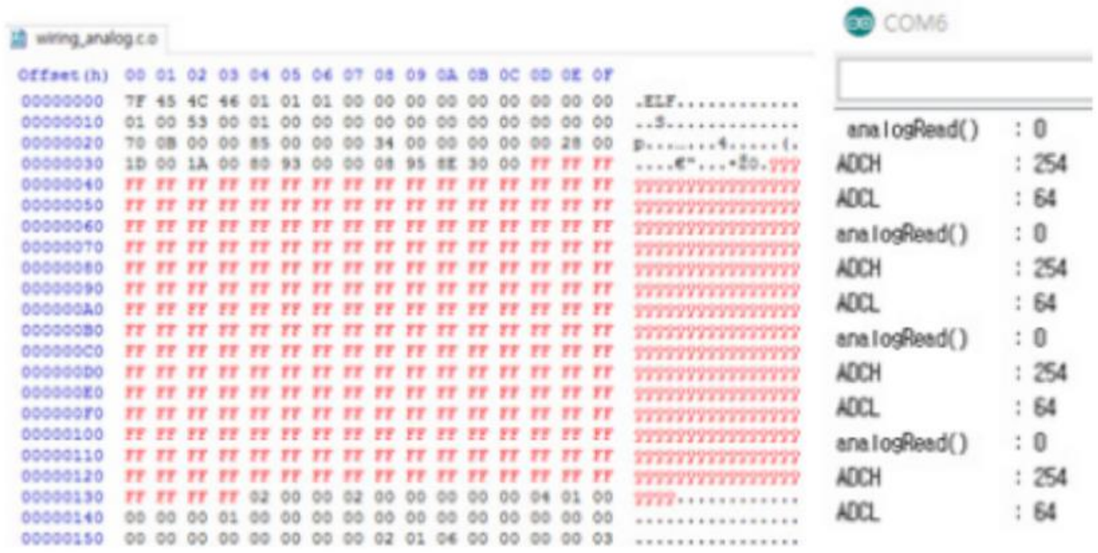
- Dll_Injection.exe : [IDE 모니터링] 기능 수행
  - Arduino_builder.exe에 의해 실행되는 프로세스를 실시간으로 모니터링 하여 컴파일 실행 순간을 알아냄
- d.exe : [임시파일 변조] 기능을 수행
  - 프로그램이 컴파일되어 호스트컴퓨터에 임시로 생성된 wiring_analog.c.o 파일을 변조하는 기능 수행
  - analogRead()함수의 ADCH와 ADCL 레지스터의 값을 0xFF로 초기화함

함수 조작에 의한 공격 기법



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

임시파일을 변조 시도 이전



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

임시파일을 변조 시도 후

파일의 내용이 모두 0xFF값으로 바뀌어 있음

함수 조작에 의한 공격 기법

- 아날로그 센서 데이터를 처리하는 analogRead()함수를 다루었지만 자이로 센서와 같은 디지털 센서 입력정보도 동일한 방법으로 변조 가능
- 드론의 경우 자이로센서의 입력값을 처리하기 위해 digitalRead() 함수를 사용하는데 이에 사용되는 레지스터 값을 조작하면 드론은 위치정보를 오인하여 정상적인 운행이 불가

이와 같은 공격으로 인해 아두이노 보드는 변조된 센서 입력정보에 따라 동작하게 될 수 있고 전체 시스템에 심각한 장애를 초래할 수 있음

3. 자율주행차 모의해킹 실습 과정

실습(1) : 자율주행자동차 코드 조작을 통한 운행 실습

기존코드의 문제점

1. 무작위 방향 변경: 장애물을 만날 때 로봇의 방향을 무작위로 변경
→ 예측할 수 없는 동작을 일으켜 로봇이 다른 장애물과 충돌할 가능성이 존재

2. 장애물 메모리 없음: 이전에 만난 장애물의 위치를 기억하지 않음
→ 로봇이 같은 방향으로 계속 회전하고, 주변 환경을 효과적으로 탐색하지 못함

실습(1) : 자율주행자동차 코드 조작을 통한 운행 실습

기존코드의 문제점

3. 보정 없음: Distance_Measurement()에서 센서 읽기의 보정이나 유효성 검사를 수행하지 않음
→ 거리가 부정확하게 측정되거나 장애물 회피 작업이 원활하게 이루어지지 않을 가능성 존재

4. 오류 처리 없음: 본 코드는 오류 처리나 장애 허용 메커니즘을 포함하지 않고 있음

5. 제한된 센서 범위: 위의 코드는 제한된 범위(300cm) 내의 장애물만 검사
→ 검사 범위 불충분, 더 멀리 위치한 장애물과 충돌할 위험성 존재

실습(1) : 코드 수정(1)

```
while (distance <= 0) {    //요 부분 수정
   Backward();
   delay(800);
   Stop();
   delay(50);
   Distance_Measurement();
   delay(100);
}
```

거리가 0일때 후진하도록 조작한 소스 코드

⇒ 장애물 인지 기능을 무력화해 장애물과 충돌하도록 조작

실습(1) : 코드1 주행 영상

실습(1) : 코드 수정(2)

```
////////거리감지//////////
void Distance_Measurement() {
  //digitalWrite(TrigPin, LOW);
  //delay(2);
  //digitalWrite(TrigPin, HIGH);  // trigPin에서 초음파
  //delayMicroseconds(10);
  //digitalWrite(TrigPin, LOW);
  //duration = pulseIn(EchoPin, HIGH);    // echoPin C
  //distance = ((float)(340 * duration) / 1000) / 2;
  //delay(50);
  distance = 400; //요 부분 수정
}
```

Distance_Measurement() 함수의 반환값을 항상 400이 되도록 조작

⇒ 장애물과의 실제 거리를 반영하지 못하게 함으로써 장애물과 충돌하도록 조작

실습(1) : 코드2 주행 영상

실습(1) : 코드 보완

```
// 변경할 방향 검사
int left_distance, right_distance;
Left();
delay(300);
Distance_Measurement();
left_distance = distance;
Stop();
delay(50);

Right();
delay(300);
Distance_Measurement();
right_distance = distance;
Stop();
delay(50);
```

```
// 이전에 만난 장애물의 위치를 저장
prev_distance = distance;
```

장애물을 만날 때 로봇의 방향 변경 전
앞에 장애물이 있는지 검사하는 과정과
장애물의 위치를 기억하게 하는 코드 추가

실습(1) : 코드 보완 주행 영상

실습(2) : 임시파일 데이터 조작 대상 파일 분석

0224test.ino.cpp.o

변조 대상 임시파일

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	7F	45	4C	46	01	01	01	00	00	00	00	00	00	00	00	00	.ELF....
00000010	01	00	53	00	01	00	00	00	00	00	00	00	00	00	00	00	..S.....
00000020	20	21	00	00	85	00	00	00	34	00	00	00	00	00	28	00	!...4.....(.
00000030	16	00	13	00	78	9C	63	63	60	60	60	05	62	46	86	B9	....xœcc```.bF†¹
00000040	8C	BA	00	02	93	00	D8	78	9C	63	63	60	60	90	62	40	Œº..".Øxœcc``.b@
00000050	00	16	86	B5	97	DB	FA	58	19	37	FE	FE	B0	8D	93	F9	..†µ—ÛúX.7þþ°."ù
00000060	CC	AB	D6	E3	9C	2C	1B	DF	9F	5E	C4	CE	00	00	A5	C8	Ì«Öãœ,.ßŸ^ÄÎ..¥È
00000070	0C	40	78	9C	9D	94	BB	4B	1C	51	14	87	CF	6F	EE	EE	.@xœ.”»K.Q.‡Ïoîî
00000080	CC	3E	74	B4	51	2C	2C	76	8B	40	52	84	6D	52	A5	13	Ì>t´Q,,v‹@R„mR¥.
00000090	53	A4	B0	CA	7F	B0	60	08	82	68	6D	B5	BB	45	48	11	S¤°Ê.°`.‚hmµ»EH.
000000A0	21	16	2A	8A	03	8A	85	18	0C	29	B2	F1	B1	EE	26	3E	!.*Š.Š…..)²ñ±î&>
000000B0	63	34	46	77	63	A1	82	DA	24	C4	CA	17	76	41	96	F5	c4Fwc¡‚Ú$ÄÊ.vA–õ
000000C0	DC	7B	47	58	C5	31	EA	C0	85	3B	67	CE	E3	BB	1F	33	Ü{GXÅ1êÀ…;gÎã».3

오브젝트 파일이므로 elf 형태임

0224test.ino.hex

레지스트리 값은 아두이노 임시파일 디렉토리의 hex 파일에 저장

00001630	44	30	31	43	45	30	30	41	43	30	30	31	44	0D	0A	3A	D01CE00AC001D..:
00001640	31	30	30	37	45	36	30	30	43	30	30	30	30	44	30	41	1007E600C0000D0A
00001650	30	30	36	31	36	45	36	31	36	43	36	46	36	37	35	32	00616E616C6F6752
00001660	36	35	36	31	36	34	32	38	31	36	0D	0A	3A	31	30	30	656164 2816..:100
00001670	37	46	36	30	30	32	39	32	30	33	41	32	30	30	30	34	7F60029203A20004

= analogRead

향후 계획

1. 코드 고도화
    - 코드 고도화 및 알고리즘을 견고화를 통해 보다 정교한 자율주행차로 발전 시킬 계획

2. 임시파일 조작을 통한 아두이노 보드 공격 모의 실습 진행 예정
    - 자율주행차 센서에 사용되는 레지스터 분석 및 임시파일 패치 실습

THANK YOU

2023.02.24
INTERLUDE

김유진, 박채령, 이가연, 한아림, 한재영