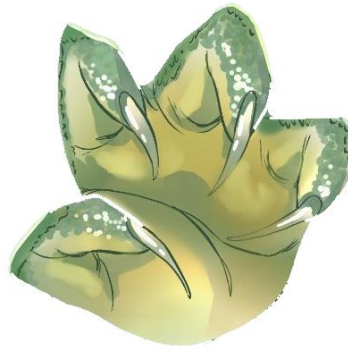


Unity based Game Vuln. Auto-Scanning Tool

사용자 매뉴얼



DragonFist

ver 1.0

내용

1. DragonFist 소개	2
1.1. 주요 기능	2
1.2. 프로그램 구성	2
2. 프로그램 설치	3
2.1. 사전 설치 사항	3
2.2. 프로그램 설치	4
3. 프로그램 사용	8
3.1. 분석할 게임 선택	8
3.2. Metadata 기능	11
3.3. Method Check 기능	13
3.4. Time/Random 기능	19
3.5. Data Search 기능	23
3.6. OCR 기능	26
3.7. Report 기능	29
4. 문제 해결	31

1. DRAGONFIST 소개

DragonFist 는 안드로이드 환경에서 사용할 수 있는 유니티 엔진 기반 모바일 게임 취약점 점검 및 대응방안 제시 자동화 도구입니다. 본 프로그램은 안드로이드 환경에서 동작하며, 에뮬레이터 또한 지원합니다. 개발자의 관점이 아니라 공격자의 관점에서 취약점을 진단하고 그에 대한 상세 내역과 해결방안을 보고서로 제공합니다. 뿐만 아니라 사용자 친화적 환경을 제공하기 위하여 GUI(Graphical User Interface)로 프로그램을 제작하였습니다.

1.1. 주요 기능

- 1) 메타 데이터 기능
- 2) 메서드 체크 기능
- 3) 시간 변조 공격 및 무작위성 조작 공격 검증
- 4) 데이터 서치 기능
- 5) 메모리 변조 가능성 검증 기능
- 6) 보고서 자동 생성 기능

1.2. 프로그램 구성

DragonFist.exe.는 설치 프로그램과 사용자 매뉴얼 및 FAQ 로 구성됩니다.

- 1) DragonFist_Setup.exe 는 설치 프로그램입니다.
- 2) DragonFist_UserManual.pdf 는 사용자 매뉴얼입니다.
- 3) DragonFist_FAQ.pdf 는 에러에 대한 상세한 설명을 제공합니다.

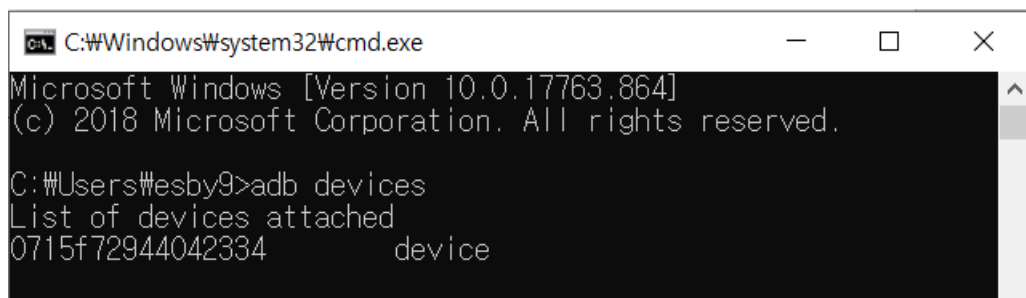
2. 프로그램 설치

2.1. 사전 설치 사항

1) adb

프로그램 사용을 위해서 adb 의 설치가 필요합니다. Android SDK Platform Tools 페이지(<https://developer.android.com/studio/releases/platform-tools>)에서 Windows 용 Android SDK Platform 을 설치한 뒤, 환경 변수 등록 절차를 진행합니다.

이후 명령 프롬프트(cmd)에서 'adb devices' 명령어를 사용하여 정상 작동 여부를 확인합니다. 아래 그림과 같이 기기의 이름이 출력되면 설치가 완료된 것입니다.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\wesby9>adb devices
List of devices attached
0715f72944042334    device
```

2) Python3

Python3 을 설치하고 사용할 수 있는 상태여야 합니다. Python 공식 홈페이지(<https://www.python.org/downloads/windows>)에서 버전 3 이상을 설치한 뒤, 환경 변수 등록 절차를 진행합니다.

3) Java JDK 설치

Java 를 설치하여 사용할 수 있는 상태여야 합니다. Oracle 공식 홈페이지(<https://www.oracle.com/technetwork/java/javase/downloads/index.html>)에서 JDK(Java Development Kit)을 설치한 뒤, 환경 변수 등록 절차를 진행합니다.

4) Frida 설치

윈도우즈 환경과 안드로이드 환경 내 모두 Frida 를 설치하도록 합니다.

- Windows : Frida 공식 홈페이지 (<https://frida.re/docs/installation>) 참고
- Android : Frida GitHub 공식 페이지 (<https://github.com/frida/frida/releases>) 참고

Android 환경의 경우, 위 페이지에서 가장 최신 버전의 Frida-server 를 설치하는 것을 권장하며, 설치 이후 명령 프롬프트 에서 'adb push' 명령어를 이용하여 안드로이드 환경 내 /data/local/tmp 경로로 복사합니다.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Wesby9\Desktop\frida>adb push frida-server-12.7.12-android-arm64 /data/local/tmp
[100%] /data/local/tmp/frida-server-12.7.12-android-arm64
```

5) Frida-server 실행

동적 분석을 진행하기 위해서는 frida-server 가 반드시 실행되고 있어야 합니다.

따라서, 분석을 시작하기에 앞서 아래 절차를 따라 frida-server 를 실행시키고 정상적으로 실행 중인지 확인하도록 합니다.

- ① 'chmod' 명령을 통해 Frida-server 파일 실행 권한 부여
- ② './[frida-server 파일명] &' 명령을 통해 해당 파일 백그라운드 실행
- ③ 'ps | grep [frida-server 파일명]' 명령을 통해 해당 파일 실행 여부 확인

```
C:\Windows\system32\cmd.exe - adb shell
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Wesby9>adb shell
zeroltekt:/ $ su
zeroltekt:/ # cd /data/local/tmp
zeroltekt:/data/local/tmp # chmod 755 frida-server-12.7.12-android-arm64
zeroltekt:/data/local/tmp # ./frida-server-12.7.12-android-arm64 &
[1] 19921
zeroltekt:/data/local/tmp # ps | grep frida
root      19921 19840 127280 74360 poll_sched 7ad1a91714 S ./frida-server-12.7.12-android-arm64
root      19971 19970 9260   1996  poll_sched 00f5198b00 S /data/local/tmp/re.frida.server/frida-helper-32
zeroltekt:/data/local/tmp #
```

6) 분석하고자 하는 게임 설치 및 실행

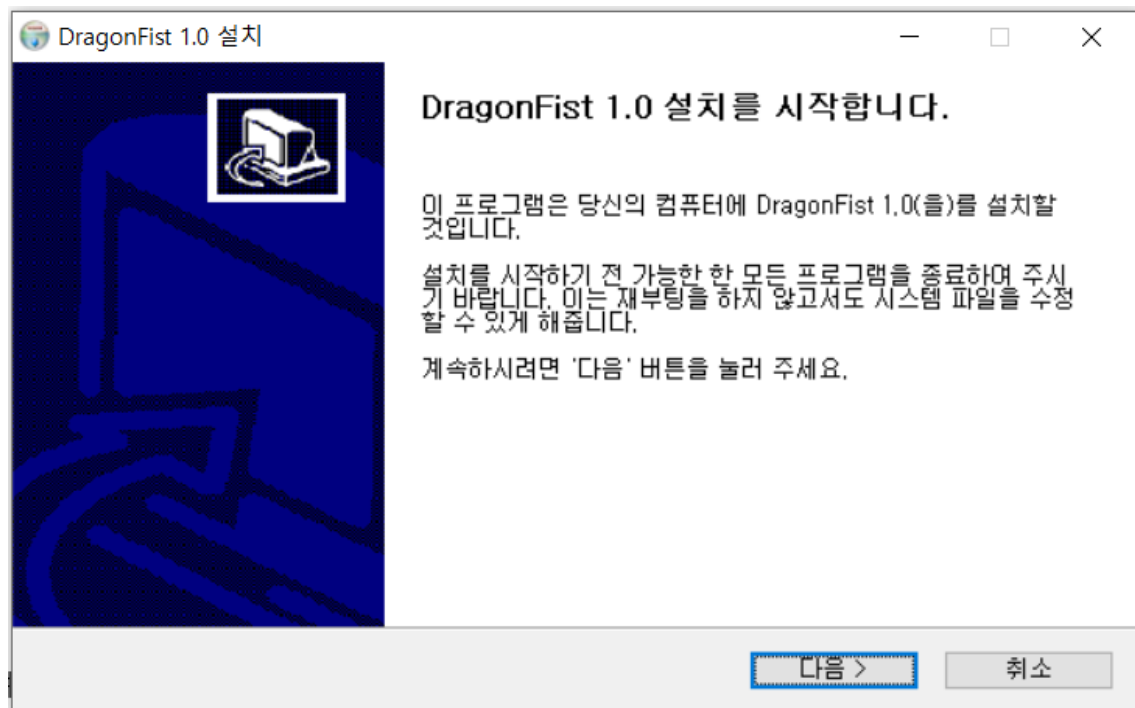
분석하고자 하는 게임이 안드로이드 환경에 설치되어 있어야 하며, 정확한 분석을 위해 최초 실행 이후에 분석하는 것을 권장합니다. 또한 동적 분석 진행시에는 게임이 진행중인 상태에서 분석을 진행하여야 합니다.

2.2. 프로그램 설치

다운로드하신 DragonFist 프로그램은 다음과 같은 과정을 거쳐서 설치하셔야 합니다.

- 1) DragonFist_Setup.exe 파일을 실행합니다.

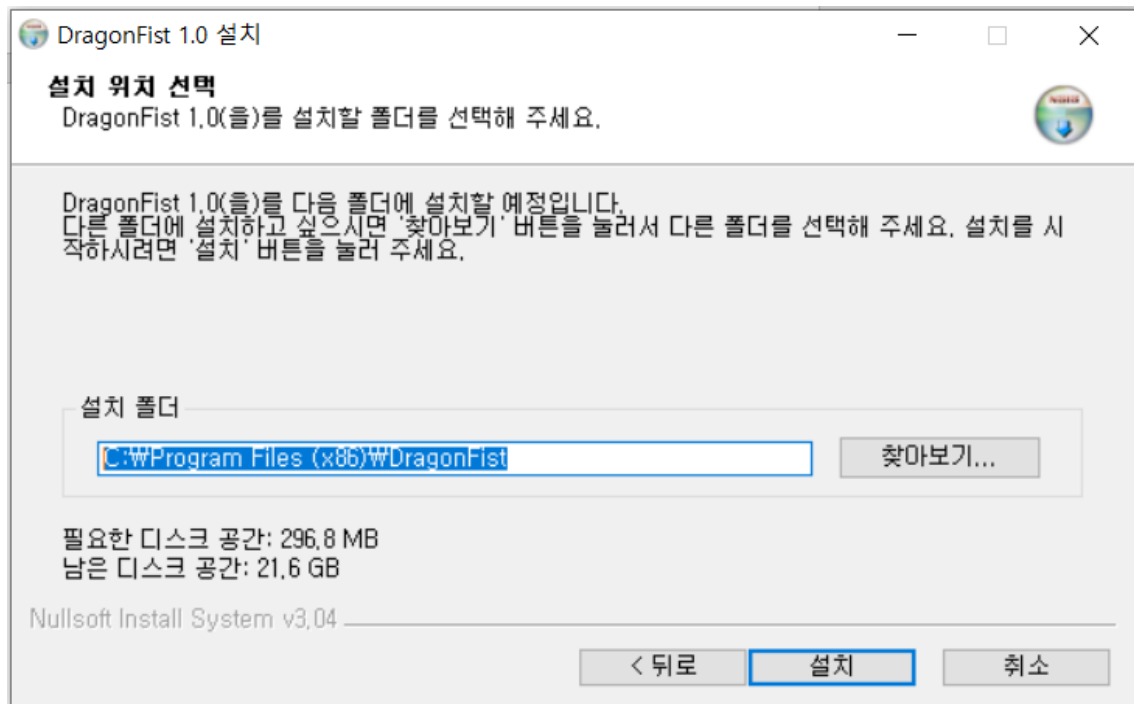
2) '다음' 버튼을 클릭합니다.



3) 사용권 계약에 '동의함' 버튼을 클릭합니다. (약관 수정하고 사진 수정 필요)



4) 설치 경로를 선택하고 '설치' 버튼을 클릭합니다.

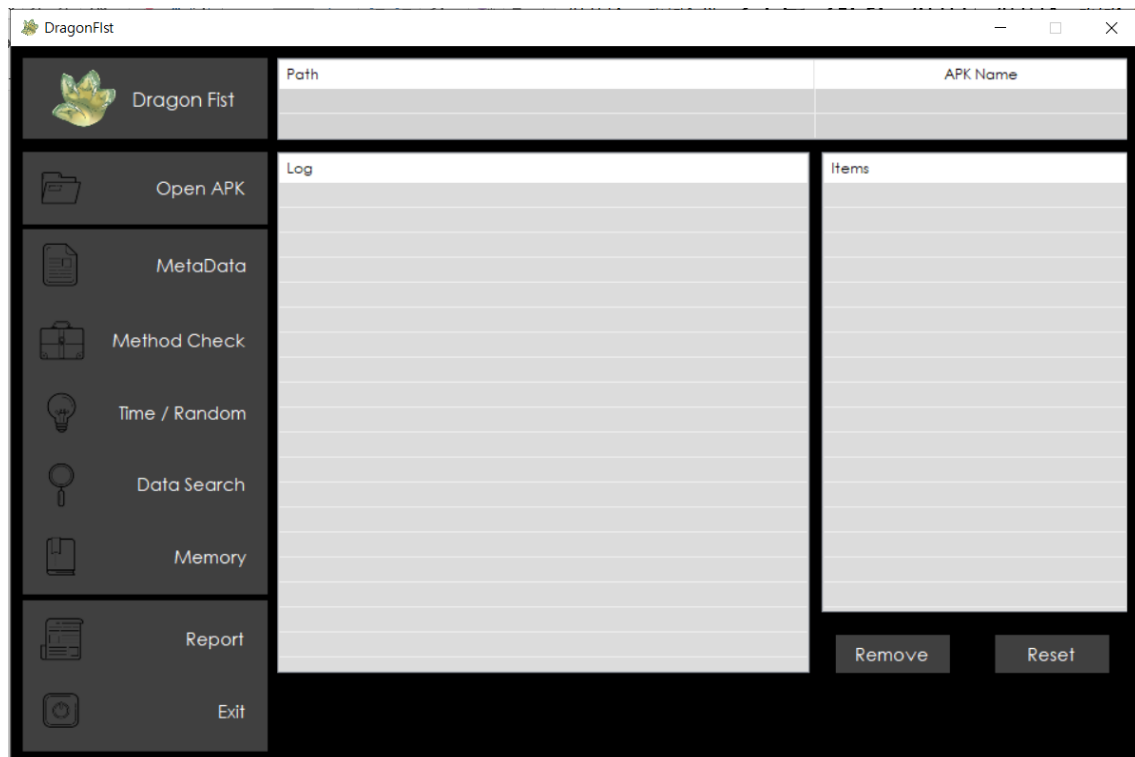


5) 설치가 완료되면, '마침' 버튼을 클릭합니다.



6) Read 파일을 따라 사전 설치 사항들의 설치를 진행합니다.

7) 아래와 같은 화면이 표시되면, 설치가 완료된 것입니다.

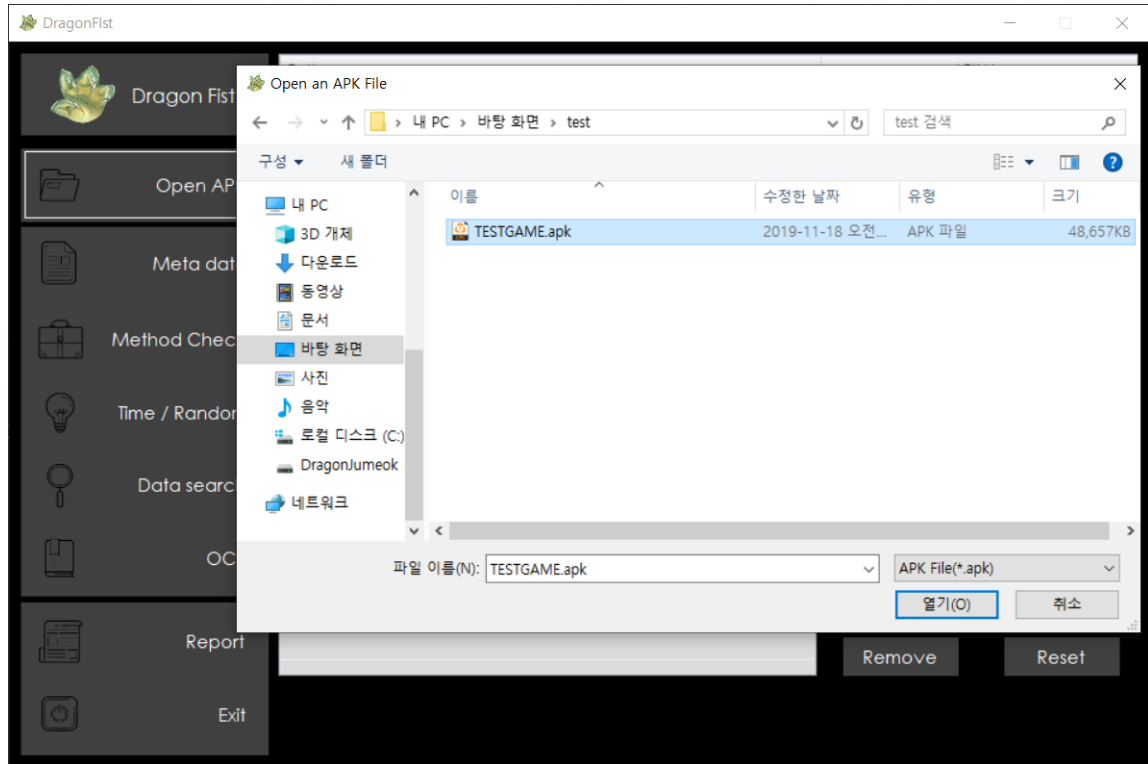


3. 프로그램 사용

3.1. 분석할 게임 선택

3.1.1. 사용 방법

- 1) Open APK 버튼을 눌러 분석할 게임을 선택하고 열기 버튼을 누릅니다.



- 2) 분석을 수행할 플랫폼을 선택합니다.

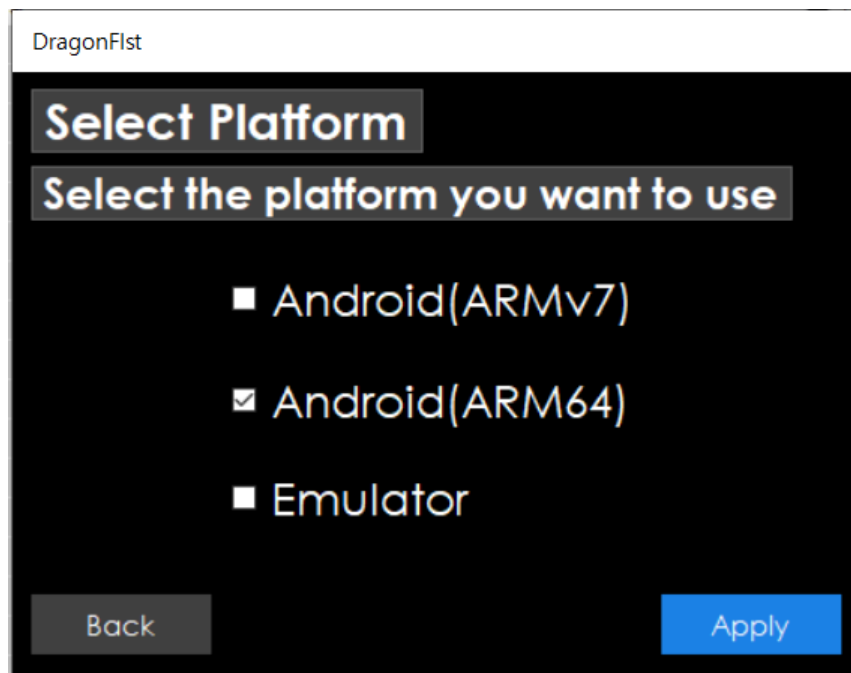
분석을 수행할 플랫폼은 일반적으로 보유하고 있는 안드로이드 기기의 환경에 따라 정해지며, 스마트폰을 이용할 경우에는 ARMv7 혹은 ARM64 환경이며 에뮬레이터일 경우에는 일반적으로 x86 환경입니다. adb 명령어를 통해 환경을 확인할 수 있습니다.

- ① 명령 프롬프트(cmd)를 실행합니다.
- ② 'adb shell getprop ro.product.cpu.abi' 명령어를 통해 다음과 같이 스마트폰 환경에 대한 정보를 알 수 있습니다.

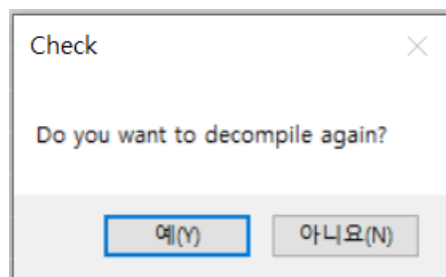
```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\mesby9>adb shell getprop ro.product.cpu.abi
arm64-v8a
```

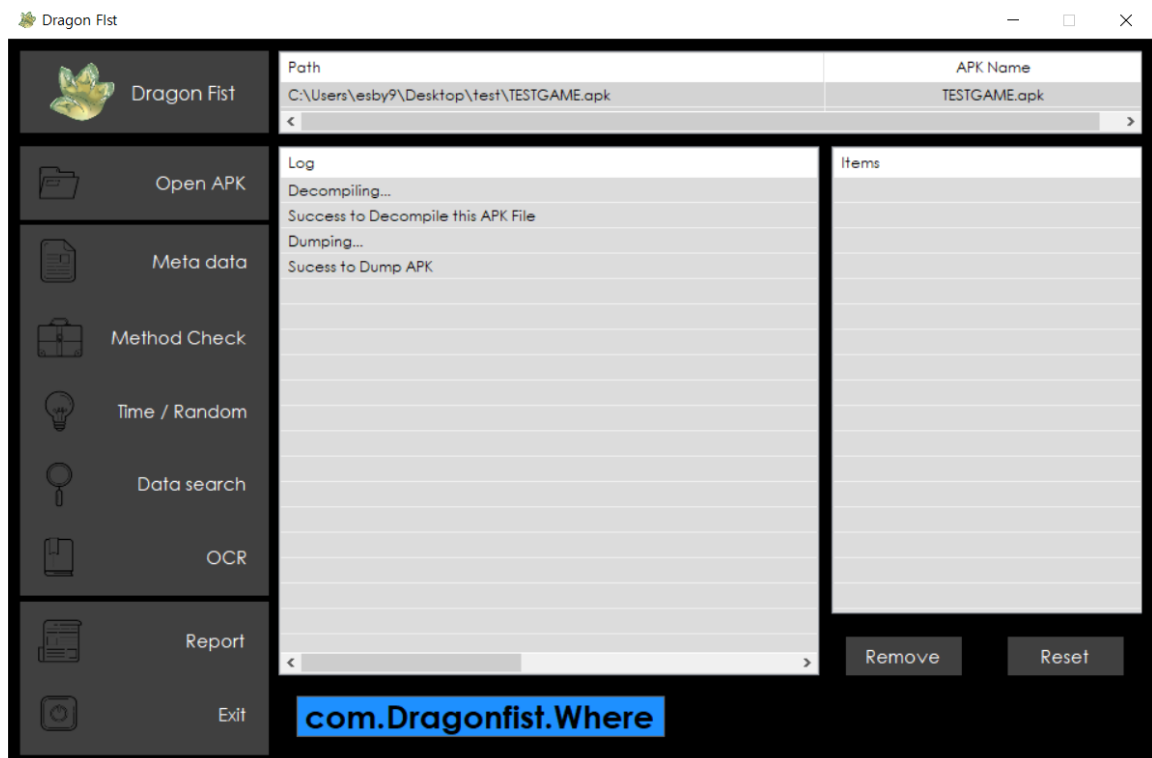
③ 플랫폼을 선택한 후에 Apply 버튼을 누릅니다.



3) 자동으로 APK 디컴파일 및 디버그 심볼 복구가 진행됩니다. 이 작업은 최대 5 분이 소요되며, 작업 중에는 프로그램에서 다른 동작을 하는 것이 불가합니다. 이미 APK 디컴파일 및 디버그 심볼 복구가 진행되어 폴더가 생성된 적이 있다면, 아래와 같은 알림창이 생성됩니다. 예를 누르면 기존 폴더의 내용을 삭제하고 다시 작업을 진행하고, 아니오를 누르면 기존 폴더를 유지합니다.



4) 작업이 완료되면 Success to Dump APK 라는 로그가 출력되며 하단에 게임의 패키지명이 출력됩니다.



5) 게임 파일이 존재하던 폴더에 2 개의 폴더가 생성된 것을 확인할 수 있습니다.

게임 파일과 동일한 이름의 폴더는 게임 파일의 디컴파일 결과를 담고 있으며, 다른 폴더는 게임 파일에 대한 디버그 심볼을 복구한 결과를 담고 있습니다.

test		
이름	수정된 날짜	유형
TESTGAME	2019-12-07 오후 1...	파일 폴더
TESTGAME_dump_arm64	2019-12-07 오후 1...	파일 폴더
TESTGAME.apk	2019-11-18 오전 9...	APK 파일

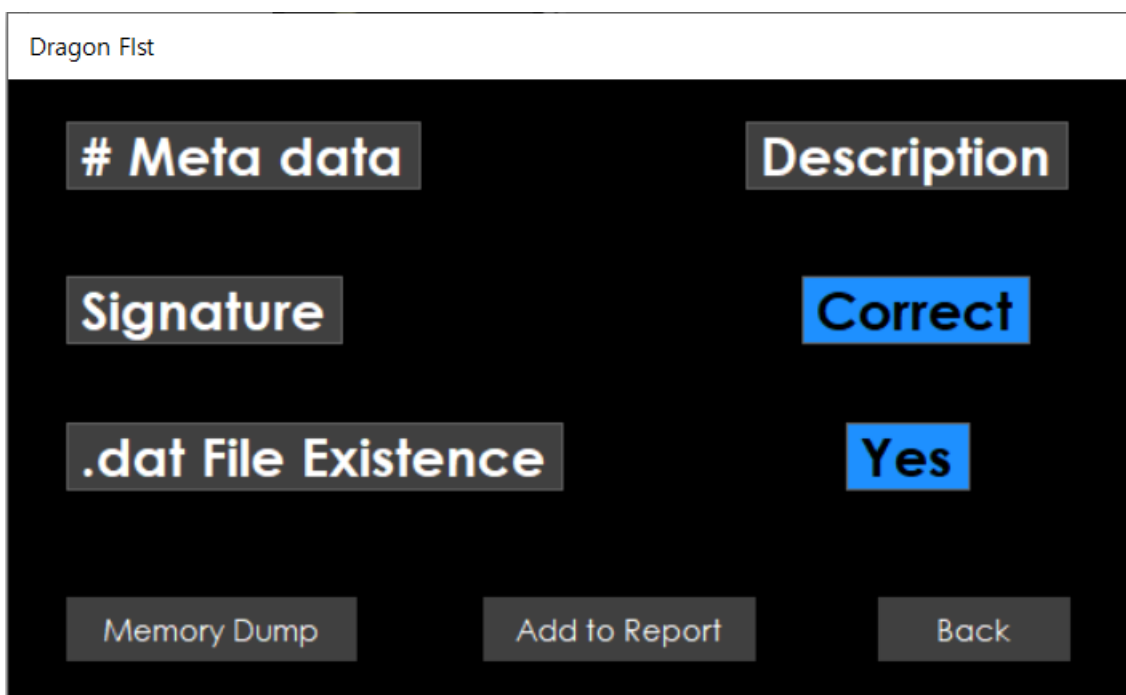
3.2. METADATA 기능

3.2.1. 기능 소개

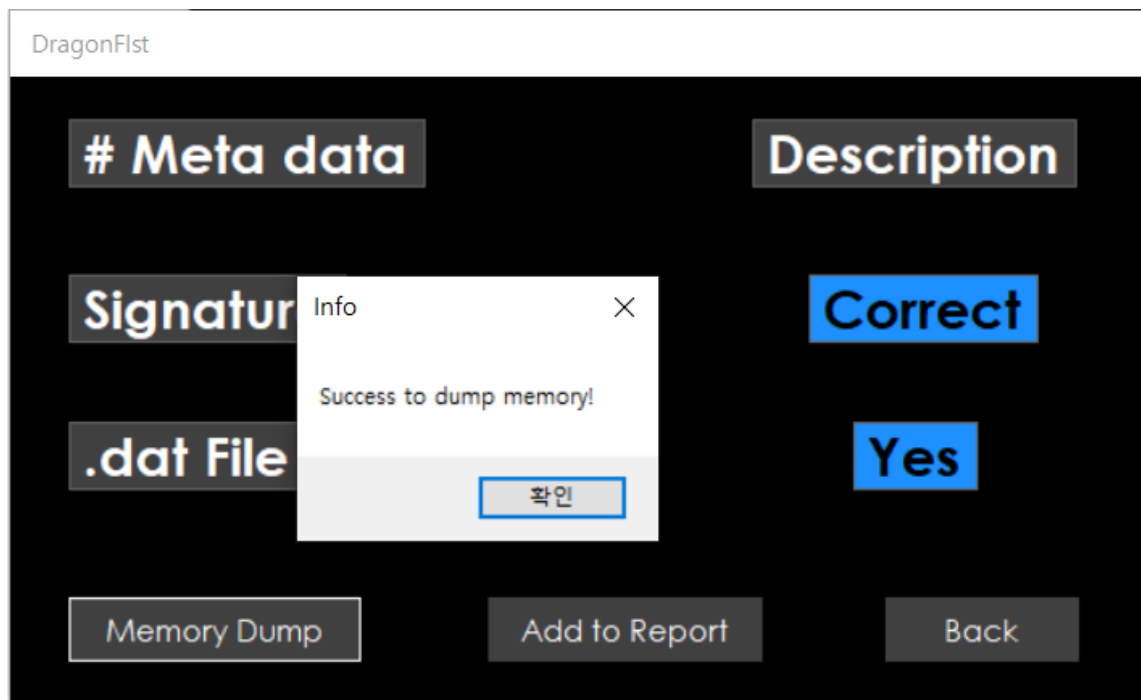
유니티 엔진으로 개발된 게임 중에서 il2cpp 방식으로 컴파일된 게임에서는 global-metadata.dat 라는 파일에 게임 내부 클래스, 함수, 변수에 대한 이름 및 위치 정보를 담고 있습니다. Metadata 기능은 해당 파일의 존재 여부를 확인하고, 존재하지 않는 경우에 복구하는 기능입니다.

3.2.2. 사용 방법

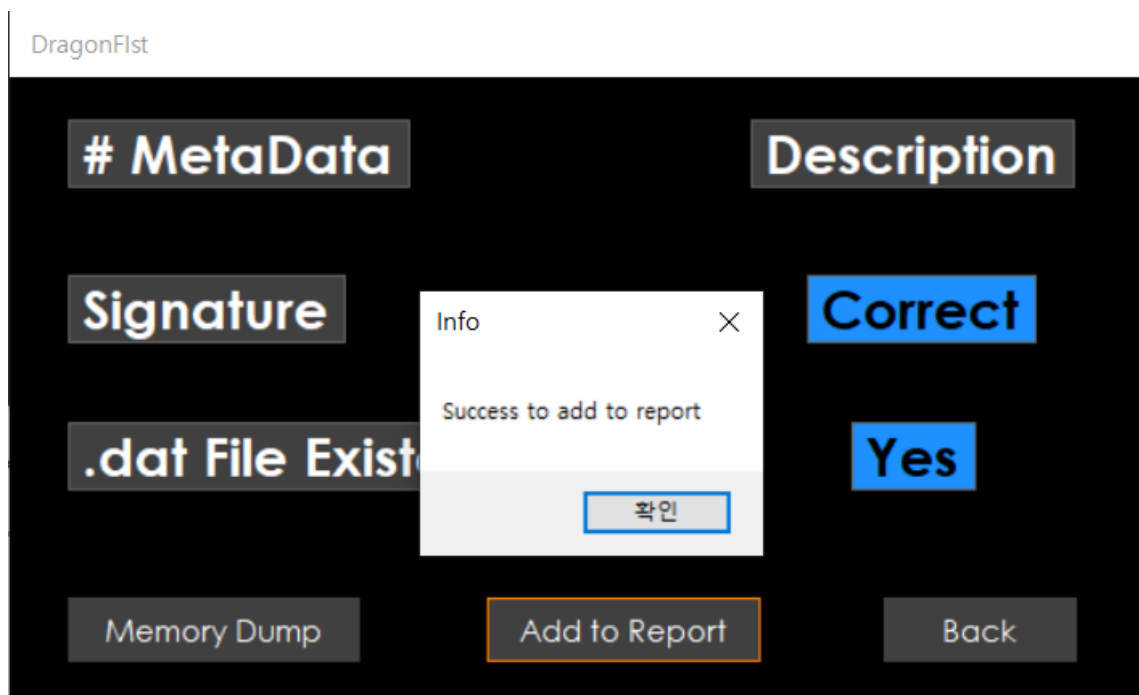
메인 화면에서 Meta data 버튼을 누르면 아래와 같은 창을 볼 수 있습니다.



- 1) **Signature** 기능에서는 global-metadata.dat 파일 시그니처 검증을 통하여 암호화 여부를 Description 에 표기합니다.
- 2) **.dat File Existence** 기능에서는 global-metadata.dat 파일 존재 여부를 확인하여 Description 에 표기합니다.
- 3) global-metadata.dat 파일이 존재하지 않거나 암호화되어 있는 경우에는 **Memory Dump** 버튼을 누르면 게임 프로세스 메모리 속에서 global-metadata.dat 파일을 복구하여 가져옵니다. 복구에 성공하면 아래와 같은 메시지를 출력합니다.



4) 결과 보고서에 Metadata 관련 내용을 추가하려면 **Add to Report** 버튼을 누릅니다.



3.3. METHOD CHECK 기능

3.3.1. 기능 소개

함수 후킹(Function Hooking)이란, 게임 동작 중 호출되는 함수를 가로채는 행위를 말합니다. 함수 후킹 공격을 통해 인자값과 반환값을 공격자가 임의로 변경할 수 있습니다. 또한 함수의 실행을 무효화시키거나, 본래의 함수가 아닌 다른 함수를 실행시키는 것도 가능합니다. Method Check 기능은 함수 후킹 공격에 함수가 얼마나 쉽게 노출되어 있는지를 확인합니다. 후킹을 위해 Frida 스크립트를 자동으로 생성하여 함수의 클래스 정보, 인자값, 반환값, 스택 정보, 레지스터, 백트레이스 정보를 출력해줍니다.

3.3.2. 사용 방법

메인 화면에서 Method Check 버튼을 누르면 다음과 같은 창을 볼 수 있습니다.

DragonFist

Method Check

Run Filter Add Save

Class	Method	RVA	Offset

Hooking List

Run Remove Reset

Class	Method

Dictionary List

Enter

Name

Gold

Attack

Random

< >

Remove Reset

Filter List

Enter

Name

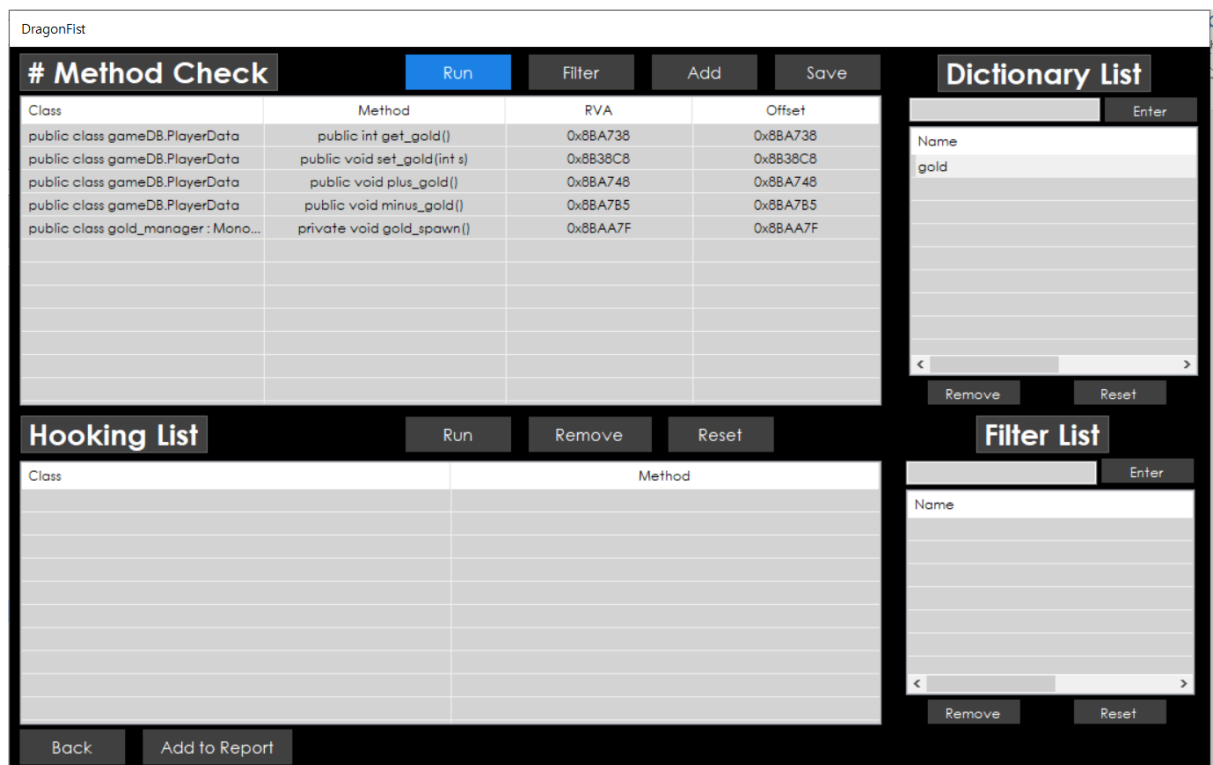
Remove Reset

Back Add to Report

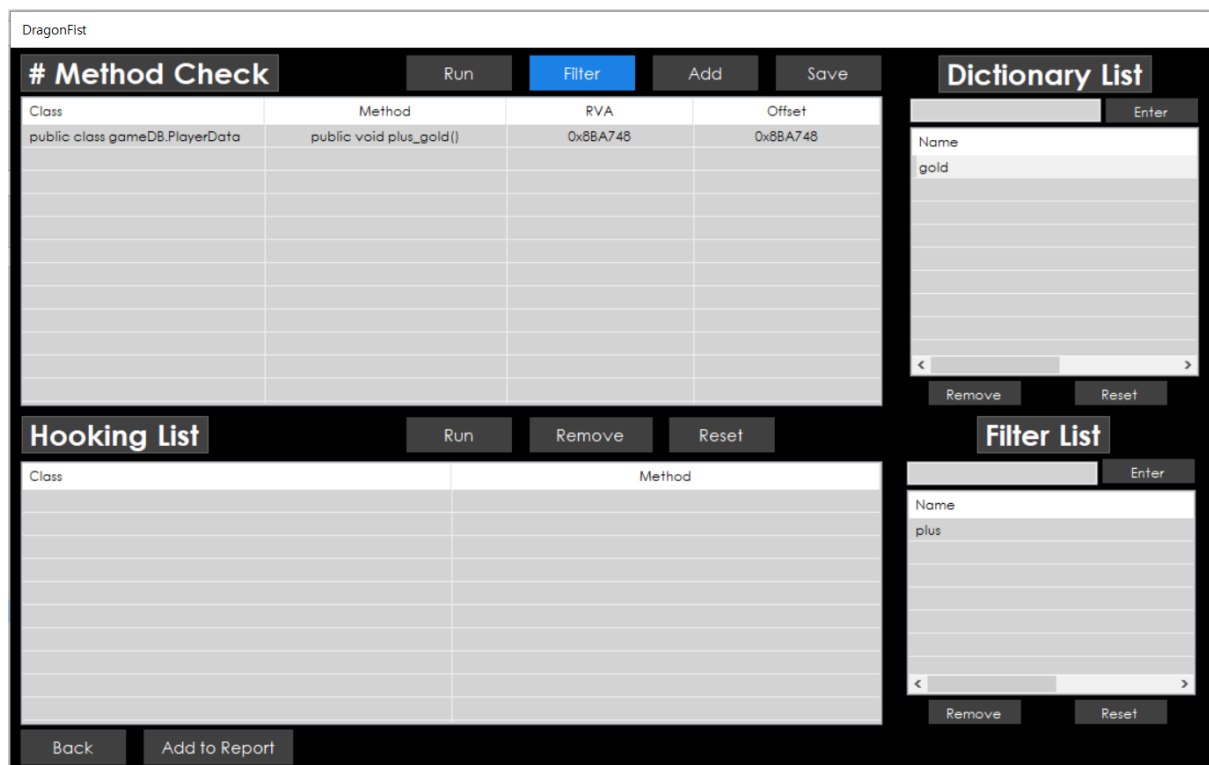
- 1) 함수 검색을 위해 Dictionary List 를 입력합니다. 딕셔너리 리스트는 사용자가 원하는 대로 입력(Enter)할 수 있으며, 기본으로 제공되는 리스트를 제거하거나(Remove), 초기화(Reset)할 수 있습니다.



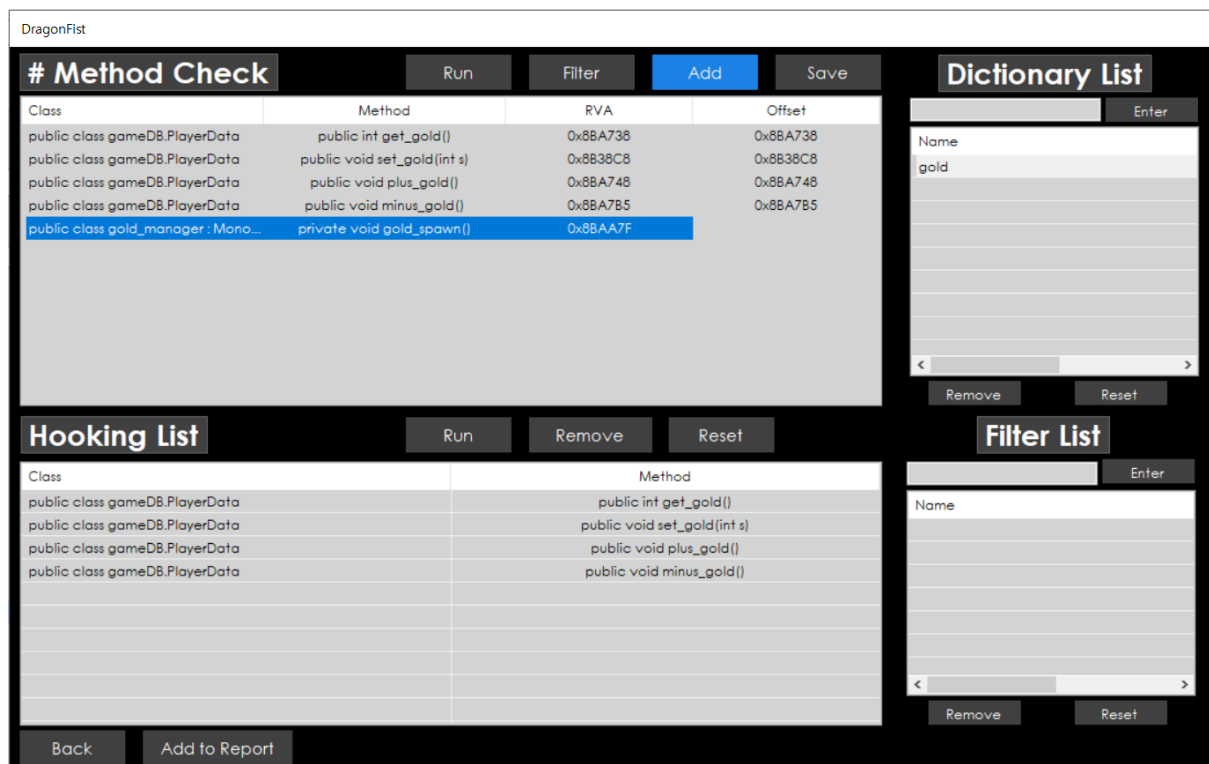
- 2) 함수 검색을 실행(Run)하면, 좌측 상단에 함수의 클래스 정보, 이름, RVA, Offset 이 표시됩니다.



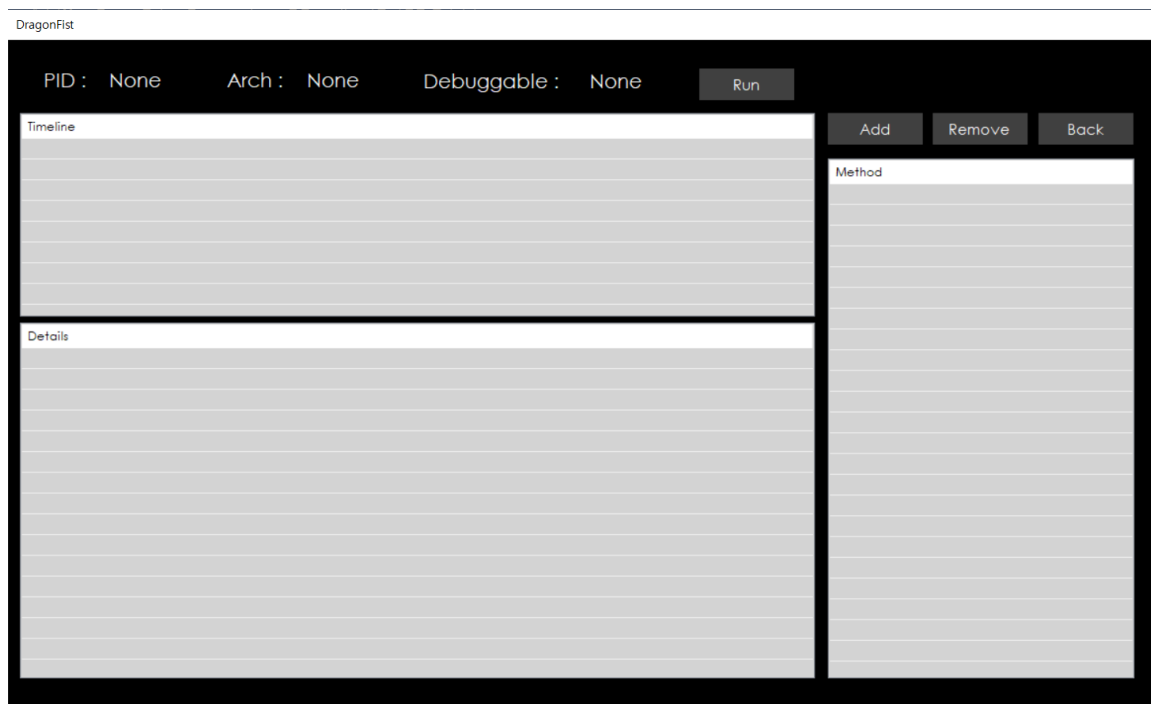
3) 우측 하단의 Filter List 를 작성하면 결과를 필터링하여 확인할 수 있습니다.



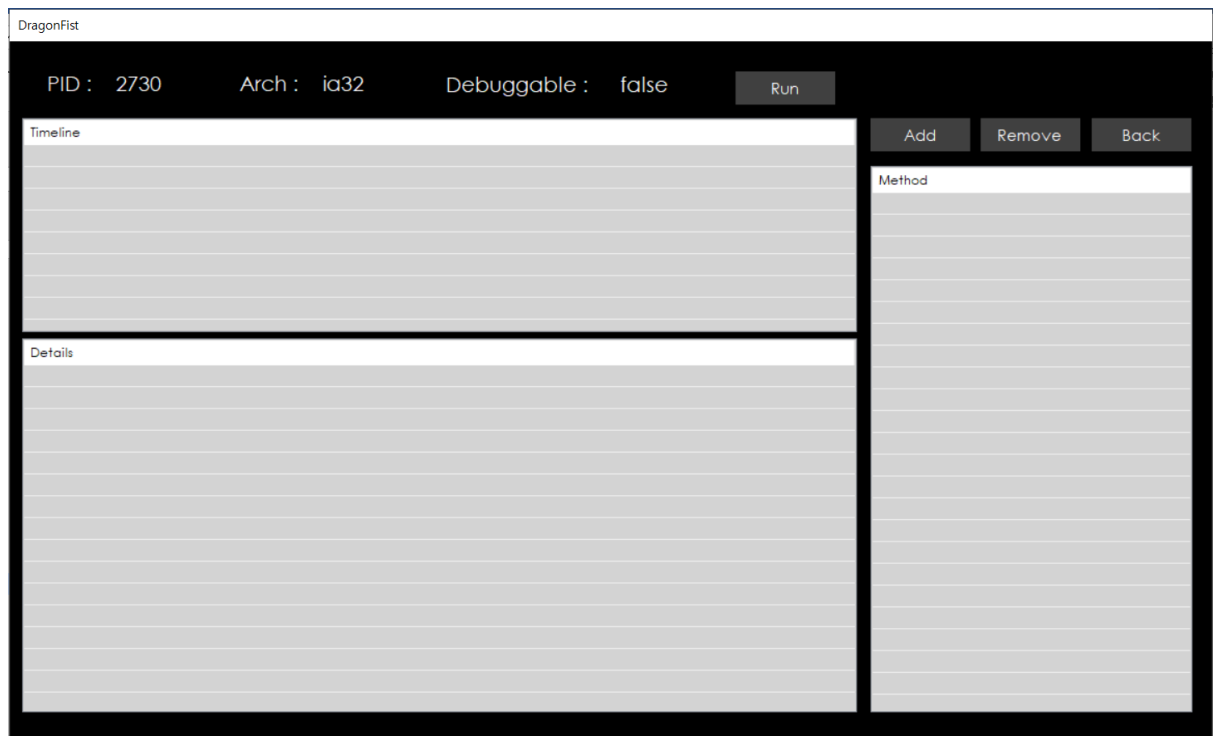
4) 함수를 Hooking List 에 추가하려면 함수를 선택한 후 Add 버튼을 누릅니다.



5) Run 버튼을 누르면 동적 분석을 시작하며 아래와 같이 새로운 창이 등장합니다.

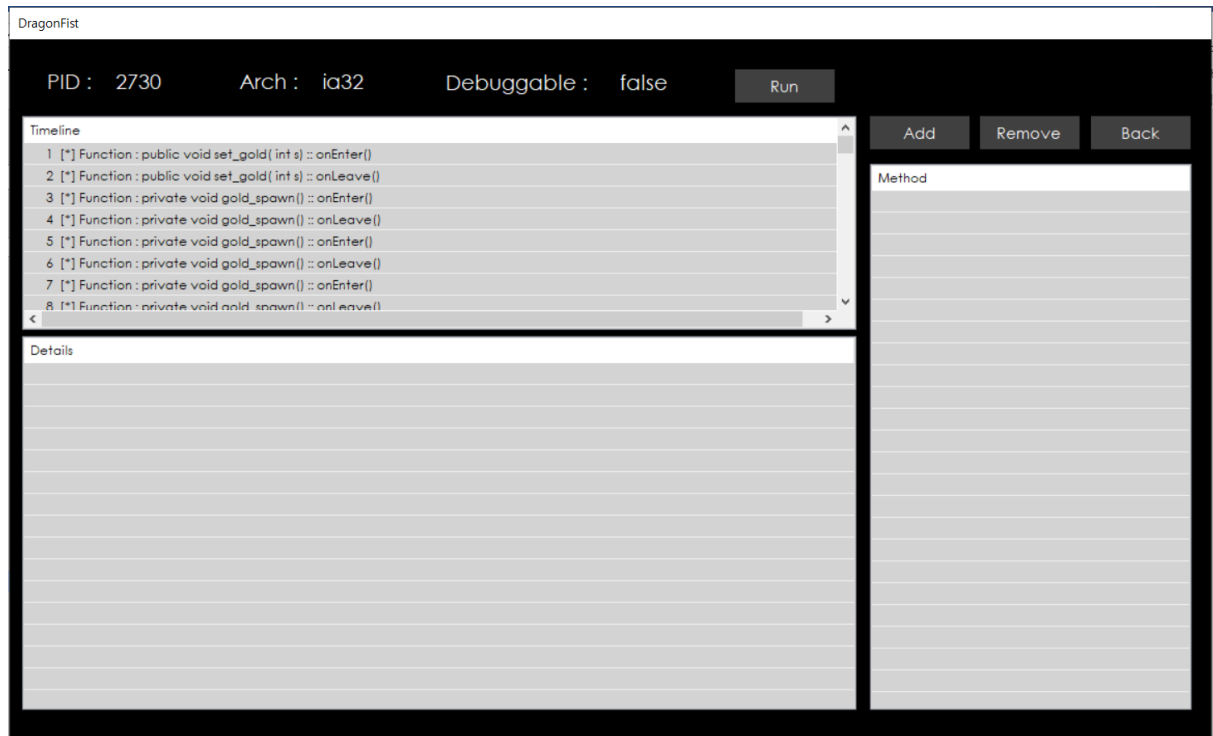


6) Run 버튼을 누르면 동적 분석을 시작하며, PID 가 None 에서 숫자로 바뀌면 정상적으로 스크립트가 로드된 것입니다.

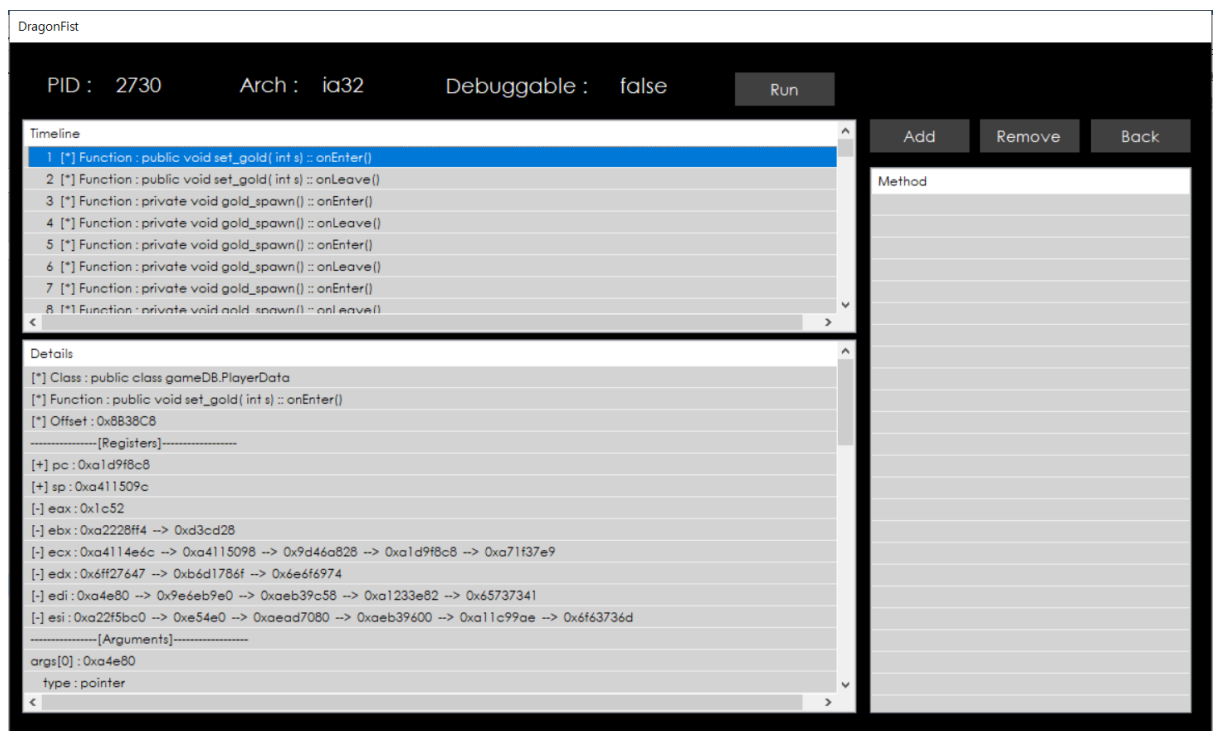


7) Hooking List 에 추가한 함수가 동작하도록 게임에서 다양한 동작을 하여야 합니다.
예를 들어 get_Attack 함수를 Hooking List 에 추가하였다면, get_Attack 함수가
동작하도록 게임 내에서 공격을 하여야 합니다.

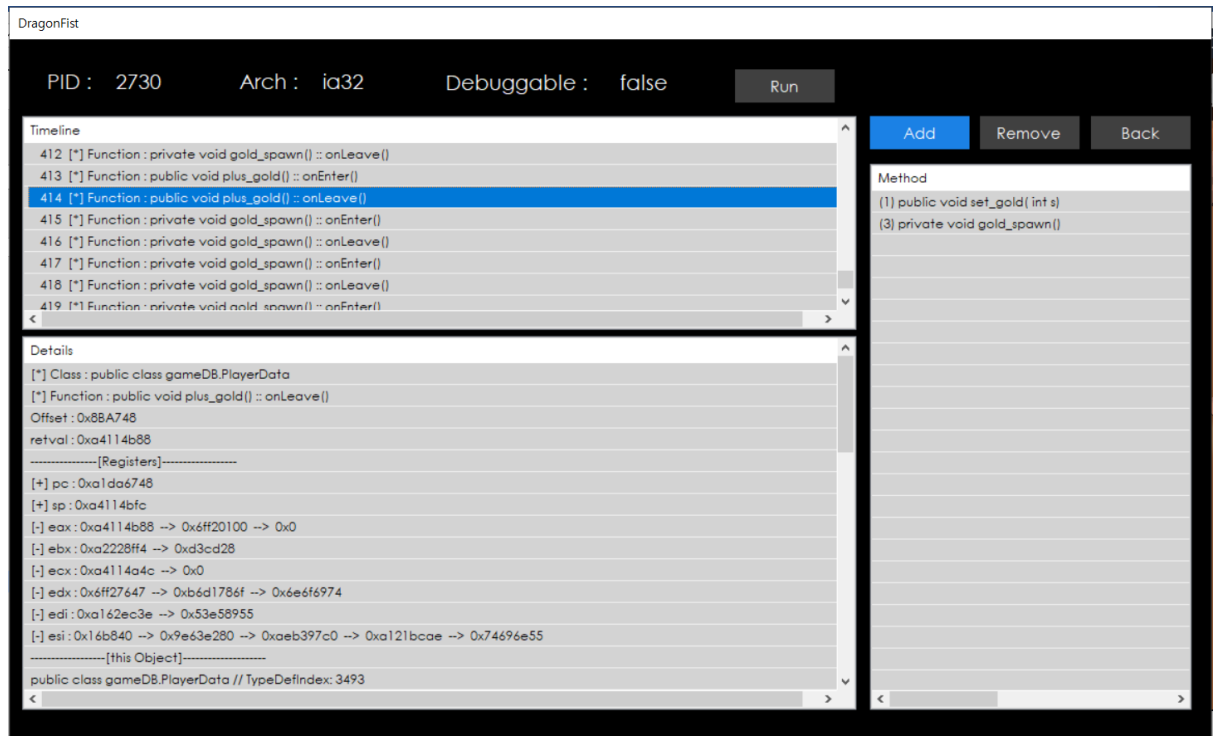
- 8) 함수가 실행되면 좌측 상단 Timeline 에 시간 순서대로 함수의 이름이 표시되며, onEnter 은 함수의 시작 부분, onLeave 는 함수의 끝 부분에서 각각 함수의 상태를 확인한 것입니다.



- 9) 자세한 정보는 Timeline 목록에서 함수를 선택하면 좌측 하단 Details 에 표시됩니다. 함수의 클래스 정보, 인자값, 반환값, 스택 정보, 레지스터, 백트레이스 정보를 출력해줍니다.



- 10) 보고서에 출력하고 싶은 내용은 Timeline 에서 함수를 클릭한 후에 Add 버튼을 누르면 우측 Method 리스트에 추가됩니다. 또한 Back 버튼을 눌러 창을 종료한 후에 좌측 하단에 Add to Report 버튼을 눌러서 최종적으로 보고서에 추가합니다.



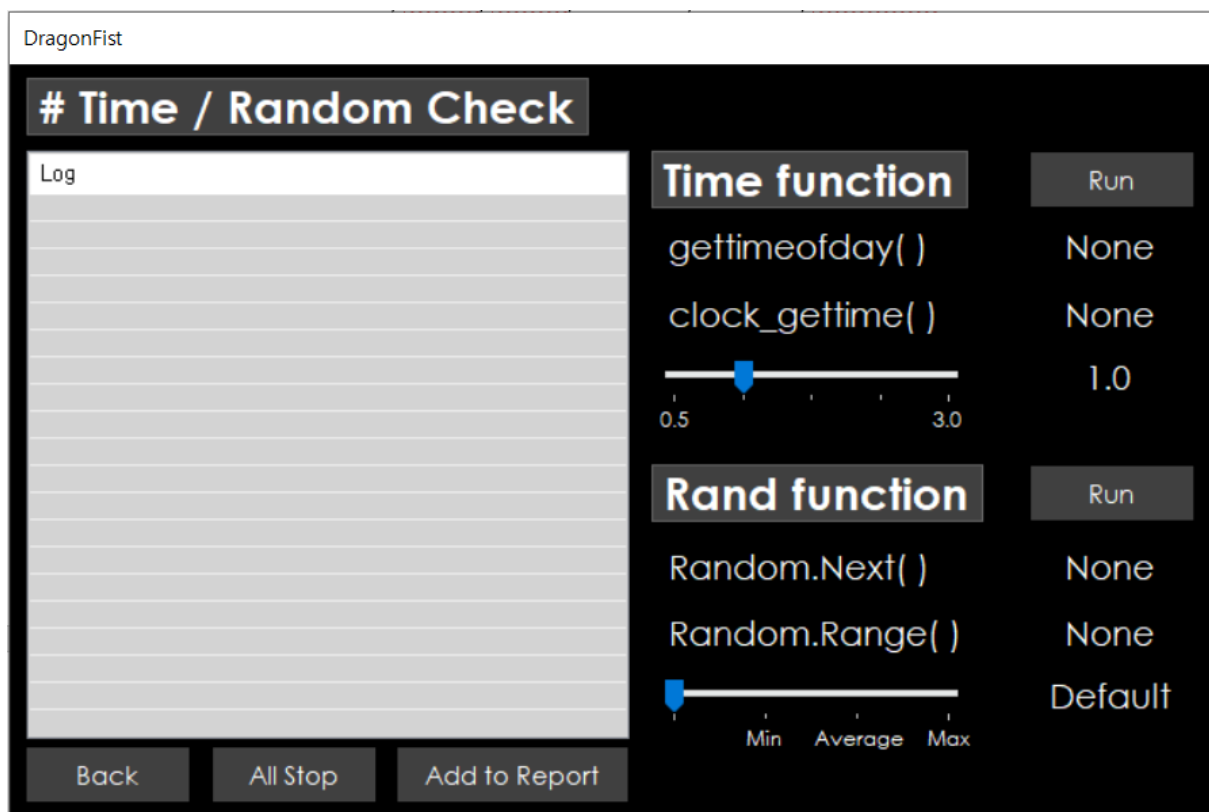
3.4. TIME/RANDOM 기능

3.4.1. 기능 소개

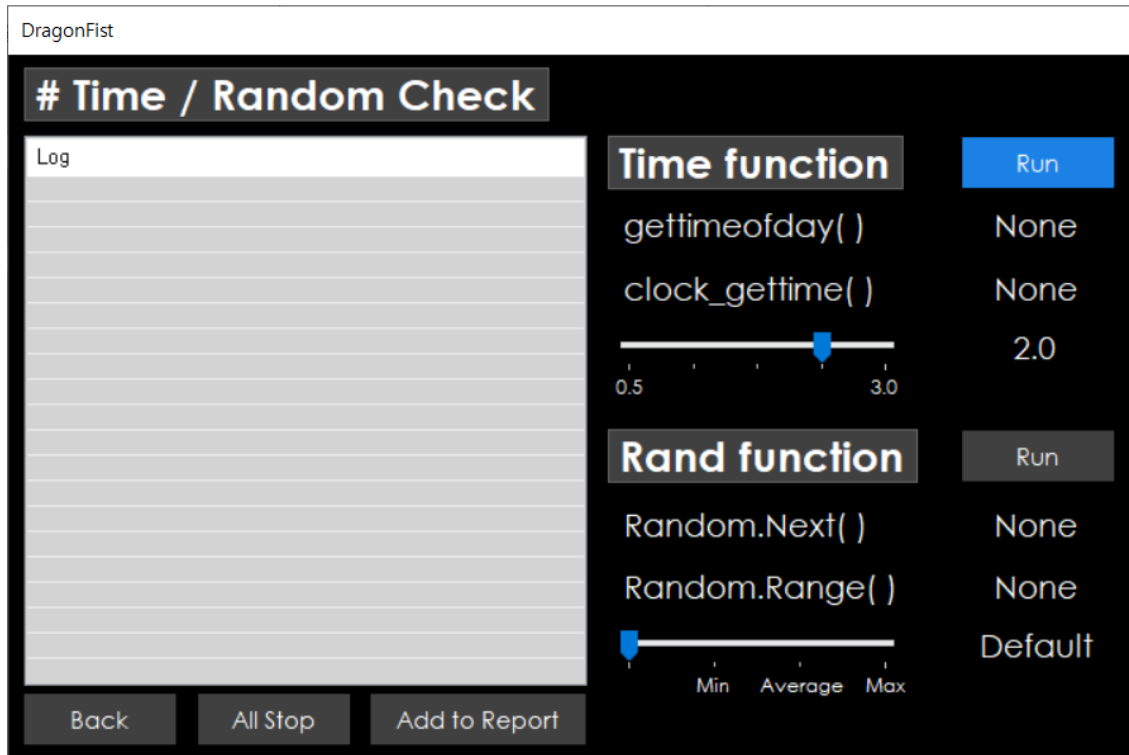
시간 정보 조작이란, 시간 정보나 시간의 속도를 변경하여 시간에 관련된 게임의 흐름을 비정상적으로 바꾸는 것을 말합니다. 난수 생성 함수 조작이란, 안드로이드 시스템 혹은 Unity 에서 기본적으로 제공하는 랜덤 함수를 이용하는 경우에 관련 함수를 후킹하여 확률 관련 정보를 조작하는 것입니다. Time/Random 기능에서는 시간 정보 조작 공격과 난수 생성 함수 조작 공격에 대해 게임이 안전한지를 진단합니다.

3.4.2. 사용 방법

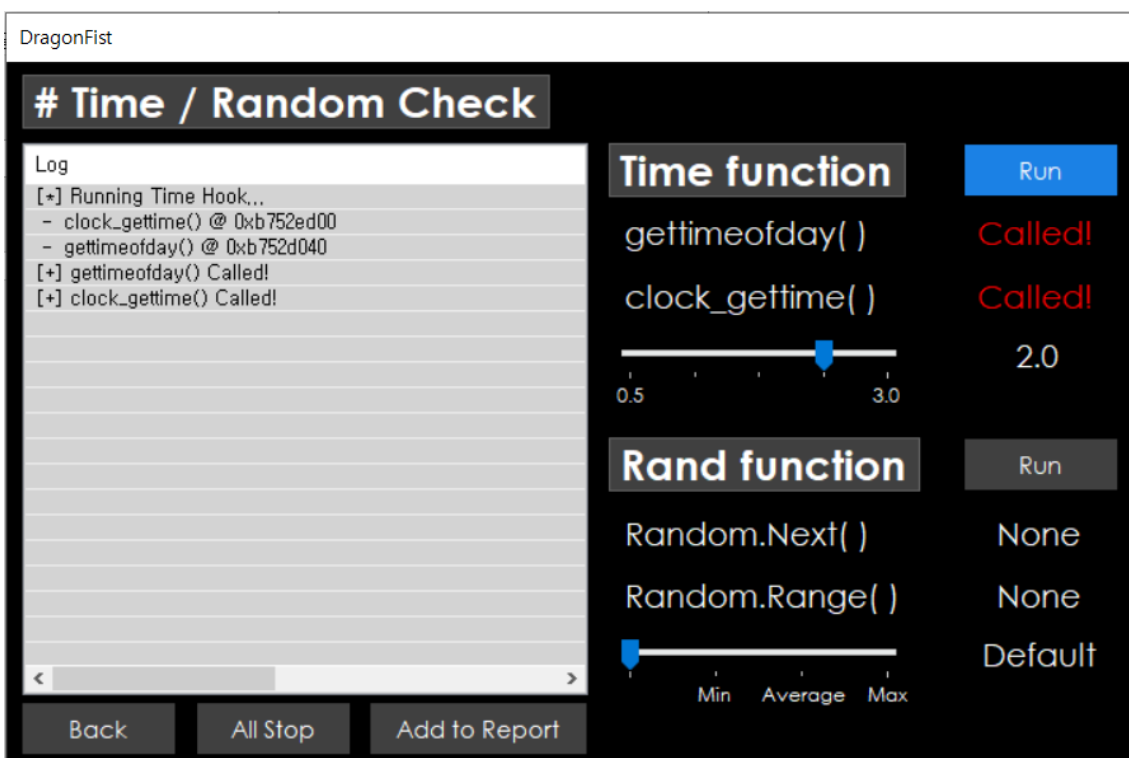
메인 화면에서 Time/Random 버튼을 누르면 다음과 같은 창을 볼 수 있습니다.



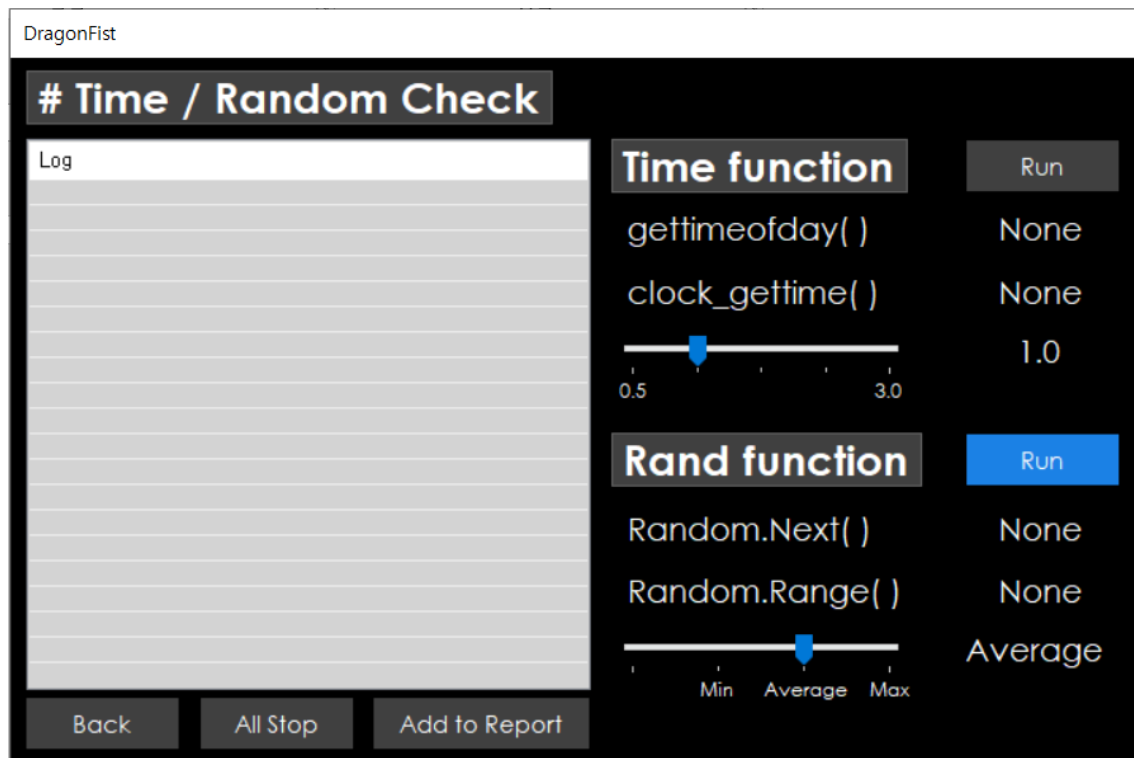
- 1) 시간의 속도를 조절하기 위해서는 우측 중앙의 트랙바를 움직여야 합니다. 1.0 배속이 기본적으로 제공되며, 0.5 배속, 1.5 배속, 2.0 배속, 3.0 배속을 선택할 수 있습니다.



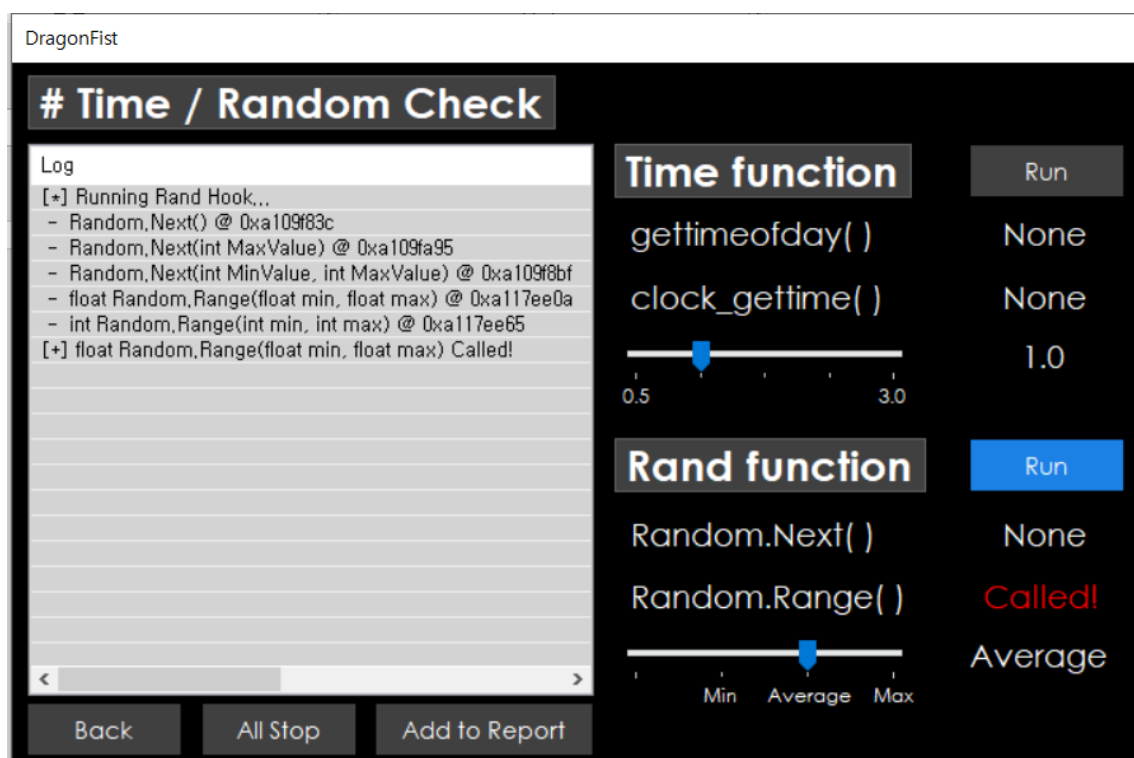
- 2) Run 버튼을 누르면 시간의 속도가 바뀝니다. 후킹이 성공하면 함수에 대한 Log 가 출력되며 함수의 상태가 None 에서 Called 로 바뀌게 됩니다. 게임의 속도가 실제로 바뀌었는지는 게임에서 확인할 수 있습니다.



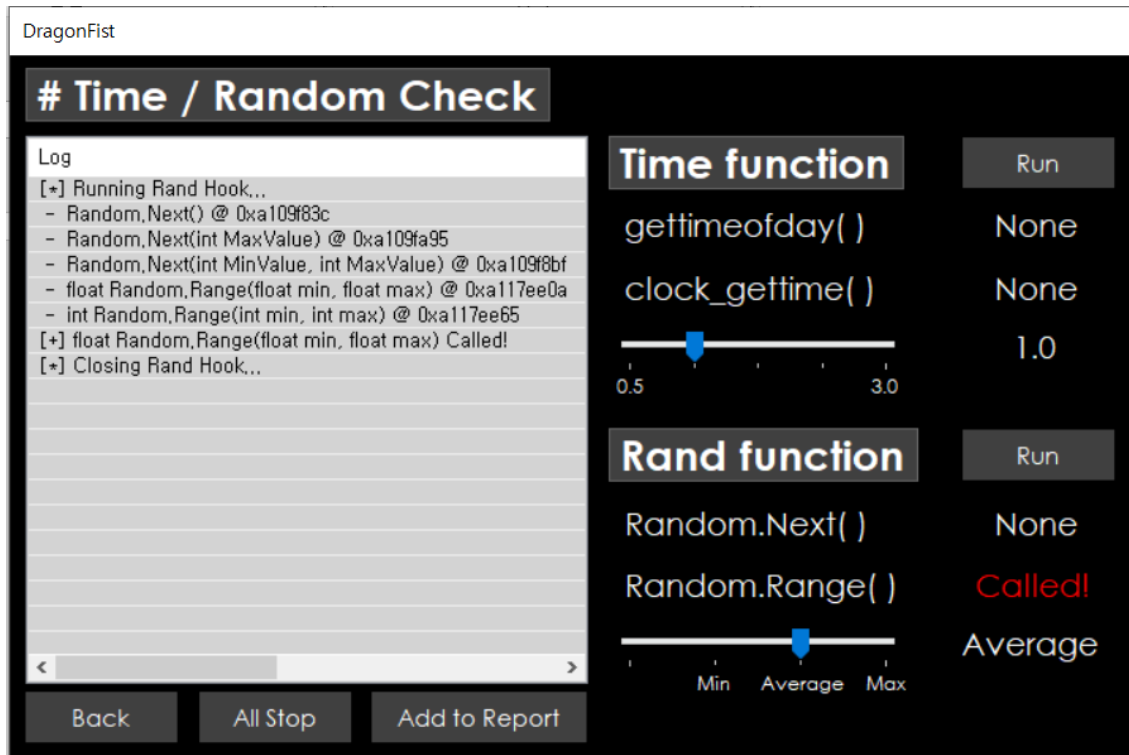
- 3) 난수를 조작하기 위해서는 우측 하단의 트랙바를 움직여야 합니다. 지정된 범위에서 Min 은 최솟값으로, Max 는 최댓값으로, Average 는 중간값으로 난수를 조작합니다.



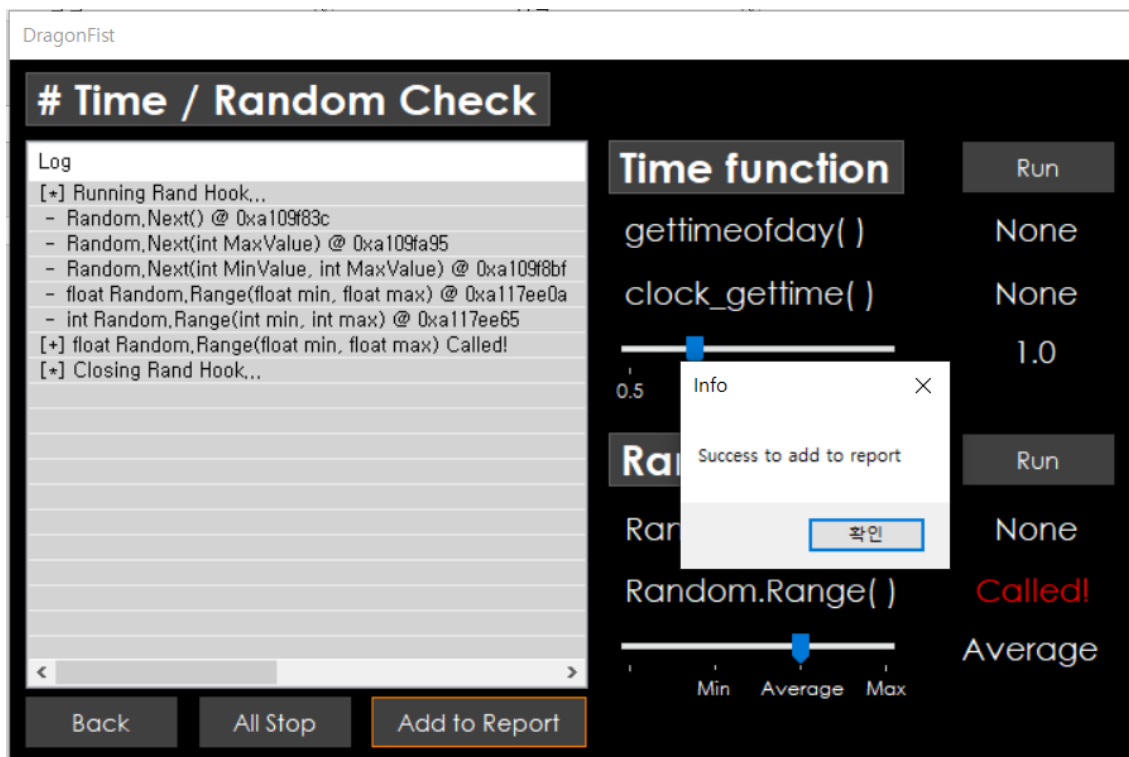
- 4) Run 버튼을 누르면 난수가 조작됩니다. 후킹이 성공하면 함수에 대한 Log 가 출력되며 함수의 상태가 None 에서 Called 로 바뀌게 됩니다. 난수 조작이 실제로 이루어졌는지는 게임에서 확인할 수 있습니다.



5) 후킹을 종료하기 위해서는 All Stop 버튼을 누릅니다. 종료 로그가 출력됩니다.



6) 시간 정보 조작 공격과 난수 생성 함수 조작 공격에 대한 내용을 보고서에 추가하기 위해서는 Add to Report 버튼을 누릅니다.



3.5. DATA SEARCH 기능

3.5.1. 기능 소개

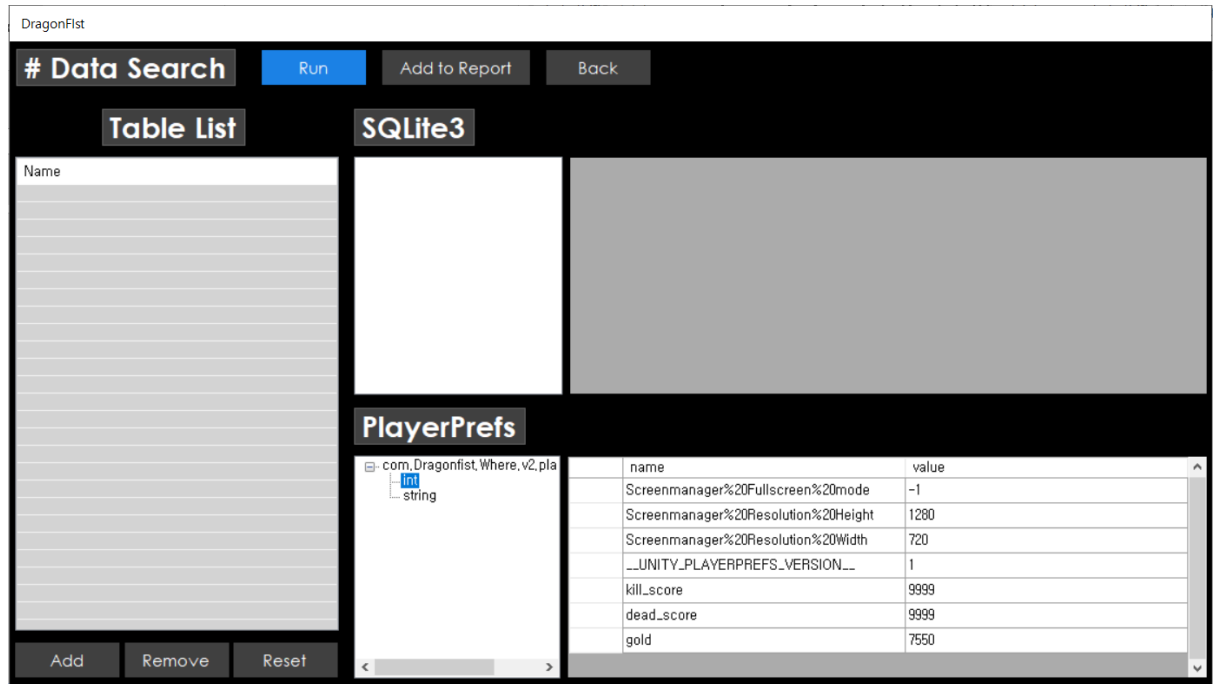
저장 데이터 변조 공격이란, 게임에서 사용하는 저장 파일(PlayerPrefs, JSON 등)이나 DB 파일(MySQL, SQLite 등)을 변조하는 것입니다. 저장 파일에는 게임의 진행 상황 이외에도 유저의 로그인 토큰이나 서버에서의 ID 값이 저장되는 경우도 있으므로 이를 보호하는 것은 중요합니다. Data Search 기능에서는 저장 파일 및 DB 파일을 자동으로 탐색하여 사용자에게 보여줍니다.

3.5.2. 사용 방법

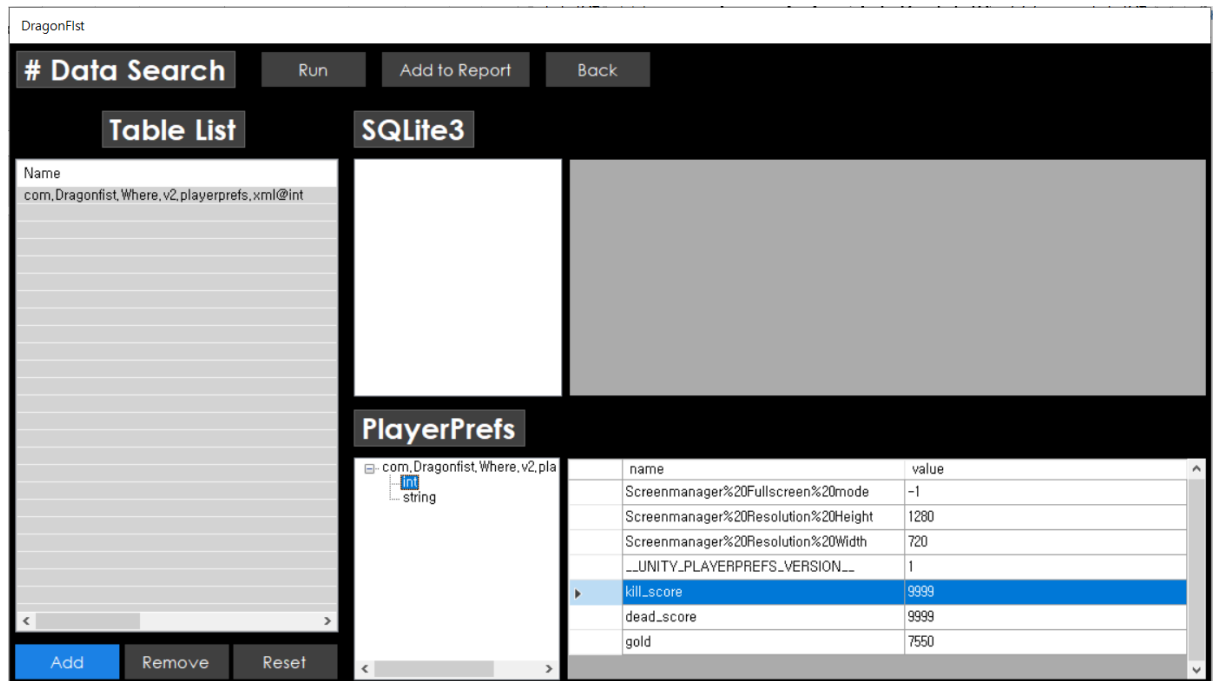
메인 화면에서 Data Search 버튼을 누르면 다음과 같은 창을 볼 수 있습니다.



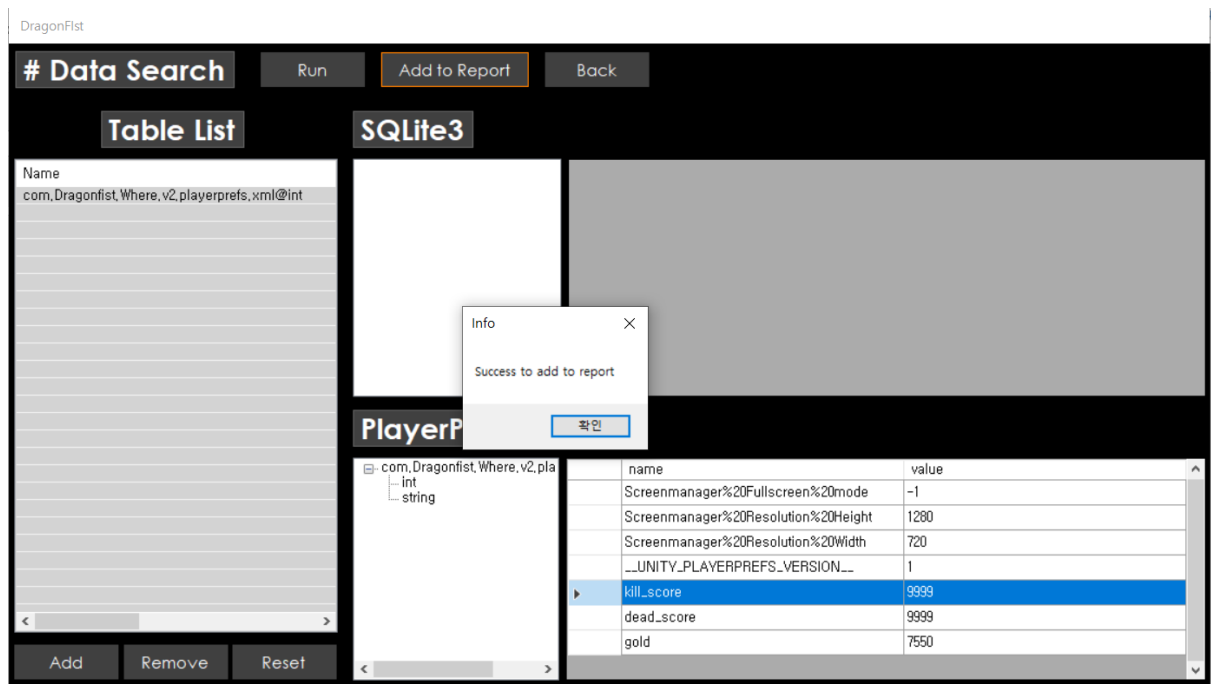
- 1) Run 버튼을 누르면 저장파일을 자동으로 탐색합니다. 게임에서 PlayerPrefs 저장 파일을 사용한다면 PlayerPrefs 칸에 파일이 표시됩니다.



- 2) 저장 파일의 내용 중에서 보고서에 추가하고자 하는 내용을 선택한 후 Add 버튼을 누르면 좌측 Table List 에 추가됩니다.



- 3) Table List 를 완성한 이후에는 상단의 Add to Report 버튼을 눌러 보고서에 최종적으로 추가합니다.



3.6. OCR 기능

3.6.1. 기능 소개

메모리 변조 공격이란, 게임 프로세스가 사용하는 메모리에 직접 접근하여 그 값을 수정하는 것을 말합니다. 메모리 변조 공격을 통해 게임 내의 중요 데이터를 변조할 수 있으며, 게임의 정상적인 동작을 방해할 수도 있습니다. 특히 메모리 변조 공격은 가장 일반적이고 자주 사용되는 공격 방법입니다. “Cheat Engine”이나 “GameGuardian”을 이용하면 전문 지식 없이도 쉽게 메모리를 스캔하고 변조할 수 있습니다. OCR 기능에서는 메모리 변조 공격에 게임이 얼마나 취약한지를 진단해줍니다.

Value Search 는 프로세스 메모리 내에서 특정 값을 가진 메모리 영역을 찾아주는 것입니다. 게임 화면에 대해 OCR(광학 문자 인식)을 하고, 그 결과값을 메모리에서 검색하는 기능입니다. Class Search 는 ...(이후 추가 예정)

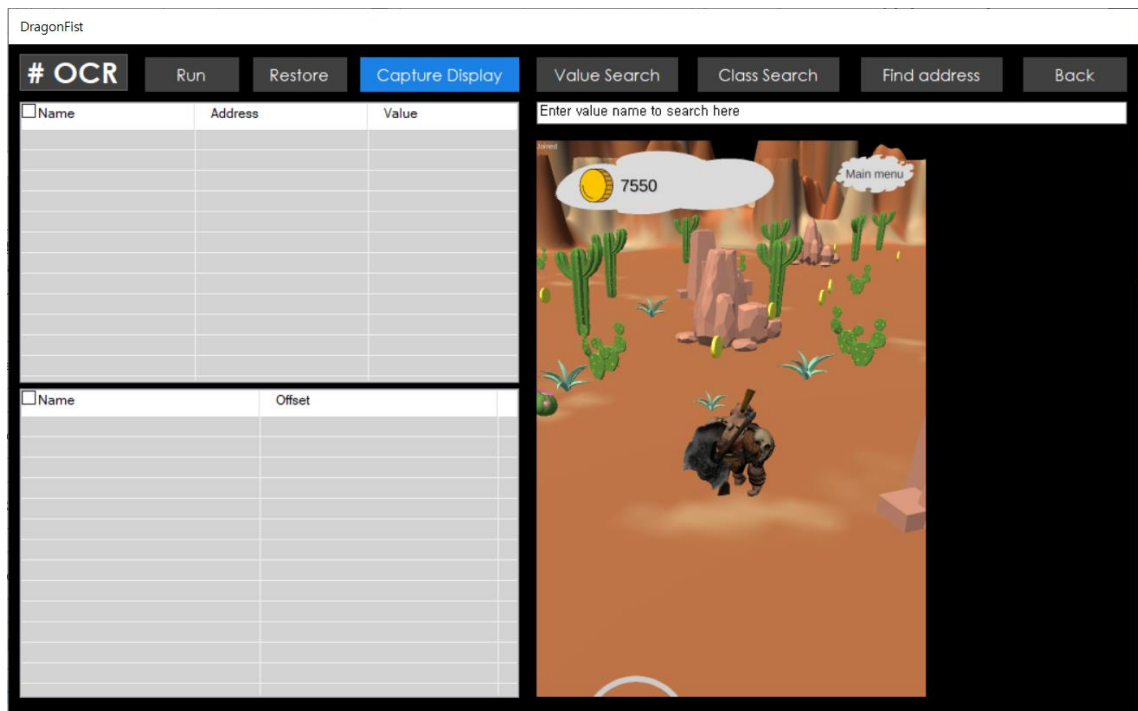
3.6.2. 사용 방법

메인 화면에서 OCR 버튼을 누르면 다음과 같은 창을 볼 수 있습니다.

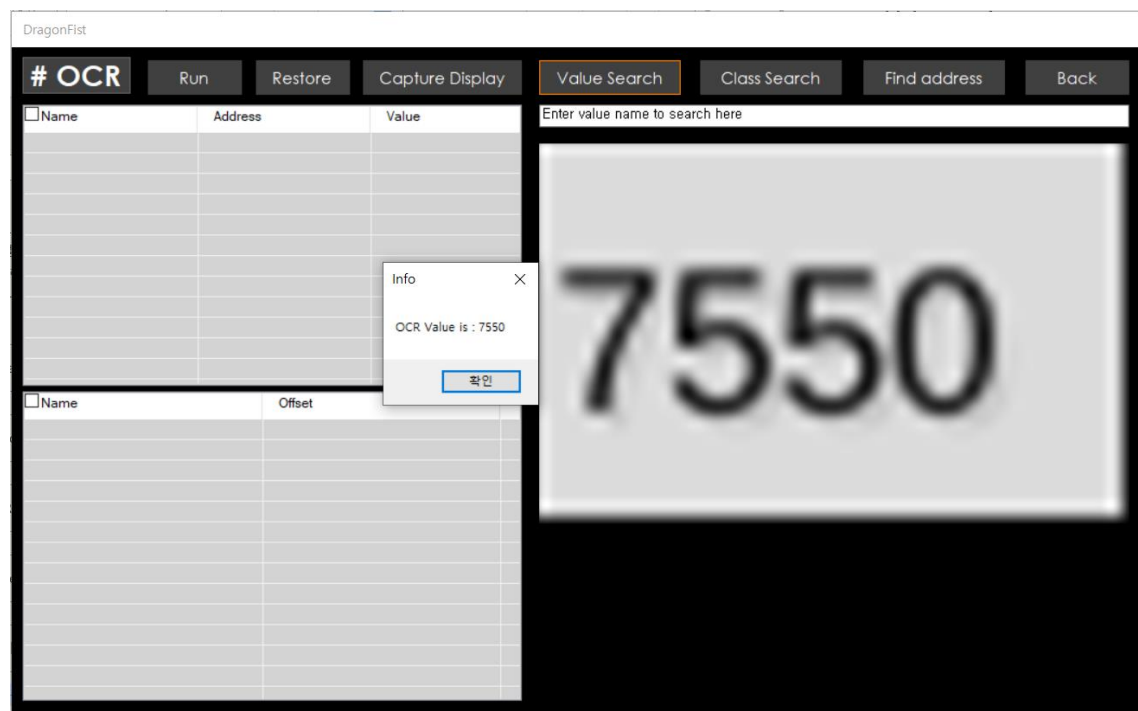
#	OCR	Run	Restore	Capture Display	Value Search	Class Search	Find address	Back
<input type="checkbox"/>	Name	Address	Value	Enter value name to search here				

<input type="checkbox"/>	Name	Offset

- 1) Value Search 를 하기 위해서는 Capture Display 가 선행되어야 합니다. Capture Display 버튼을 누르면 우측에 게임 화면이 표시됩니다.



- 2) OCR 하고 싶은 부분을 커서로 끌어서 선택한 후에 Value Search 버튼을 누르면 OCR 결과가 출력됩니다.



- 3) Run 버튼을 누르면 좌측 상단에 Value Search 에 대한 결과가 출력됩니다.
- 4) Class Search 를 하기 위해서는 (이후 추가 예정)

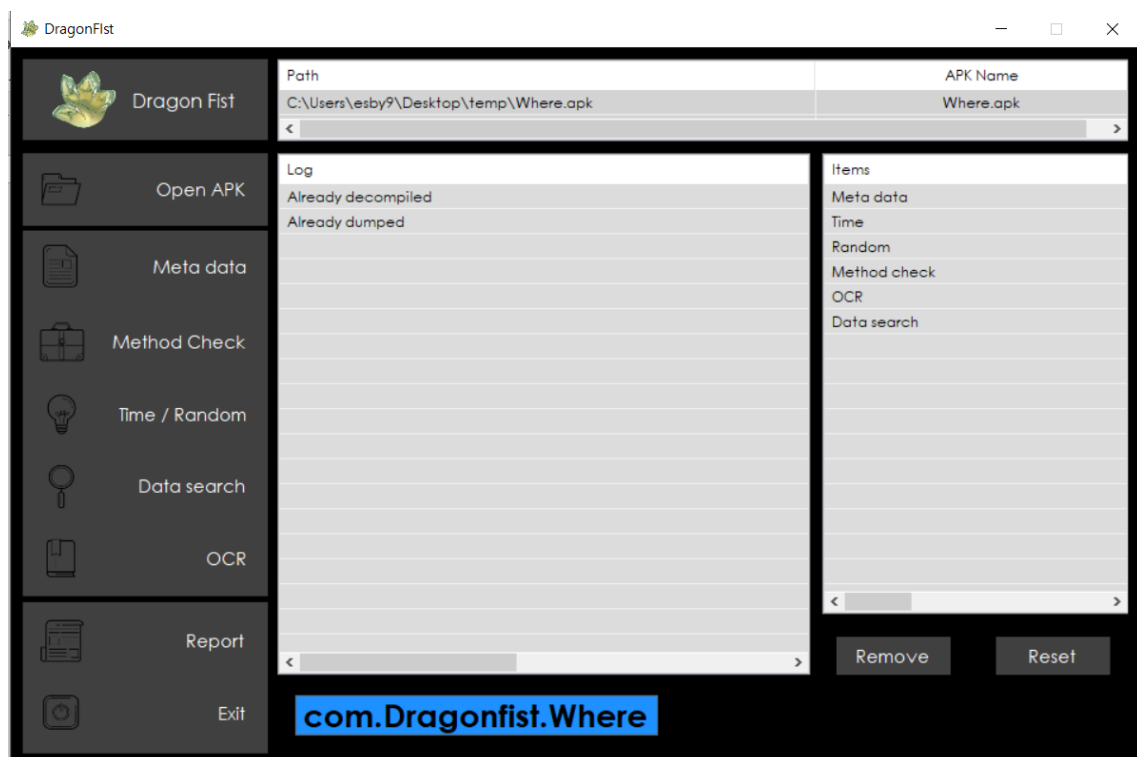
3.7. REPORT 기능

3.7.1. 기능 소개

진단한 내용을 보고서로 만들어주는 기능입니다. 발견한 취약점에 대한 상세 내역과 대응 방안까지 보고서로 자동 생성됩니다.

3.7.2. 사용 방법

- 1) 메인 화면에서 Items 에는 각 기능에서 Add to Report 버튼을 통해 보고서에 추가한 내역이 표시됩니다.



2) 메인 화면에서 Report 버튼을 누르면 다음과 같은 창을 볼 수 있습니다.

3) 보고서에 추가할 Items 를 다시 확인합니다. 체크한 내용은 보고서에 추가됩니다.

4) Make 버튼을 눌러 보고서를 생성합니다. 이 작업은 최대 5 분이 소요됩니다.

- 5) Show 버튼을 누르면 보고서를 볼 수 있습니다. 생성된 보고서의 예시는 다음과 같습니다.

Game Vuln. Auto-Scanning Report

Dragon Fist

Scanning report summary

개요	체크리스트 기반 모바일 게임 취약점 진단
목적	보안성 검토를 통해 발견된 취약점을 바탕으로 가이드라인 제공

[표 0-1] 개요 및 목적

진단 목록 및 결과

순번	점검 항목	위험도	점검 결과
1	Metadata 노출 여부	H	취약
2	함수 후킹 공격 가능성	H	취약
3	메모리 변조 가능성	H	취약
4	저장 데이터 변조 가능성	M	취약
5	시간 정보 조작 가능성	M	취약
6	난수 생성 함수 조작 가능성	M	취약

[표 0-2] 진단 결과 요약

- 6) 생성된 보고서는 DragonFist.exe 파일이 있는 경로와 동일한 경로에 생성됩니다. 보고서는 word 파일(.docx)로 제공되므로 사용자가 직접 수정할 수 있습니다.

4. 문제 해결

도구 사용법에 대해서는 사용자 매뉴얼을 참고하시고, 도구 사용 중 에러가 발생할 때에는 FAQ 를 참고하여 해결하시기를 바랍니다.

그 밖의 문제가 발생할 경우 dragonfist@gmail.com 으로 문의하여 주시기 바랍니다.