PGSharp: Analysis of a Cheat Engine on Android

Romain Thomas

November, 2021

Mobile Hacking Space – Ekoparty 2021

Introduction

About

- Security engineer at Underwriters Laboratories
- Formerly at Quarkslab
- Working on banking app certifications (EMVCo, VISA, ...)
- Author of LIEF
- Enjoy Android, reverse-engineering and, obfuscation.

PGSharp

PGSHARP

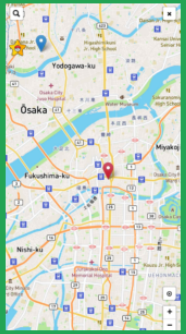
Enjoy POGO anywhere & anytime

Easy to Install and Use, NO ROOT Required

Download

Latest Version: 1.32.0 (Android Only)
Update: Sep 14, 2021
Download Link#2

*Incompatible with Google account.



PGSharp

Feature	Free	Standard
Teleport	✓	✓
Quick Catch	✓	✓
Nearby Radar	✓	✓
Autowalk	✗	✓
Skip evolve animation	✗	✓
...	...	...
Custom GPX	✗	✓

Agenda

1. Code Protection
2. Cheat Engine
3. How to detect it

Code Protection

Obfuscation

Niantic – PokemonGO

- native code heavily obfuscated with a commercial solution
- no Java obfuscation
- control-flow & data-flow obfuscation
- native strings encoding
- Java, C++, Unity

PGSharp

- native code obfuscation based on O-LLVM
- string *encryption* in Java
- control-flow obfuscation
- native strings encoding
- Java, C++, *custom* Lua VM

Native Code: Lua

- PGSharp is written in Lua with native bindings for low-level functionalities (e.g. hooking, JNI, ...)

[1] Try to make believe a VM based on Lua 5.1 while it is based on Lua 5.3
[2] `OP_RUN`, `OP_GETDOWNVAL`, `OP_OLDTABLE` and, `OP_XXOR`

Native Code: Lua

- PGSharp is written in Lua with native bindings for low-level functionalities (e.g. hooking, JNI, ...)
- the Lua logic and the Lua VM are self-contained in `lib/arm64-v8a/libmain.so`

[1] Try to make believe a VM based on Lua 5.1 while it is based on Lua 5.3
[2] `OP_RUN`, `OP_GETDOWNVAL`, `OP_OLDTABLE` and, `OP_XXOR`

Native Code: Lua

- PGSharp is written in Lua with native bindings for low-level functionalities (e.g. hooking, JNI, ...)
- the Lua logic and the Lua VM are self-contained in `lib/arm64-v8a/libmain.so`
- the Lua VM has been modified to fake the version[1] and to include custom opcodes[2].

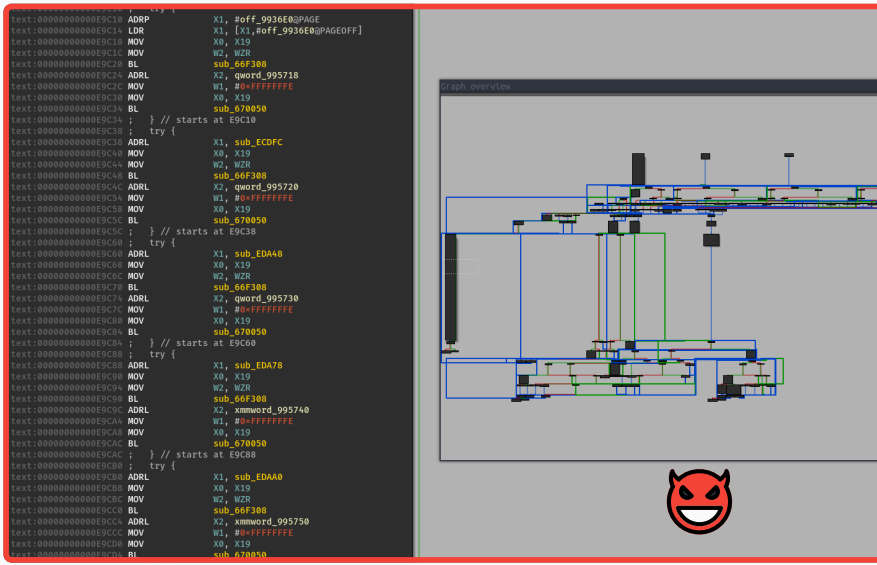
[1] Try to make believe a VM based on Lua 5.1 while it is based on Lua 5.3
[2] `OP_RUN`, `OP_GETDOWNVAL`, `OP_OLDTABLE` and, `OP_XXOR`

Native Code: Lua

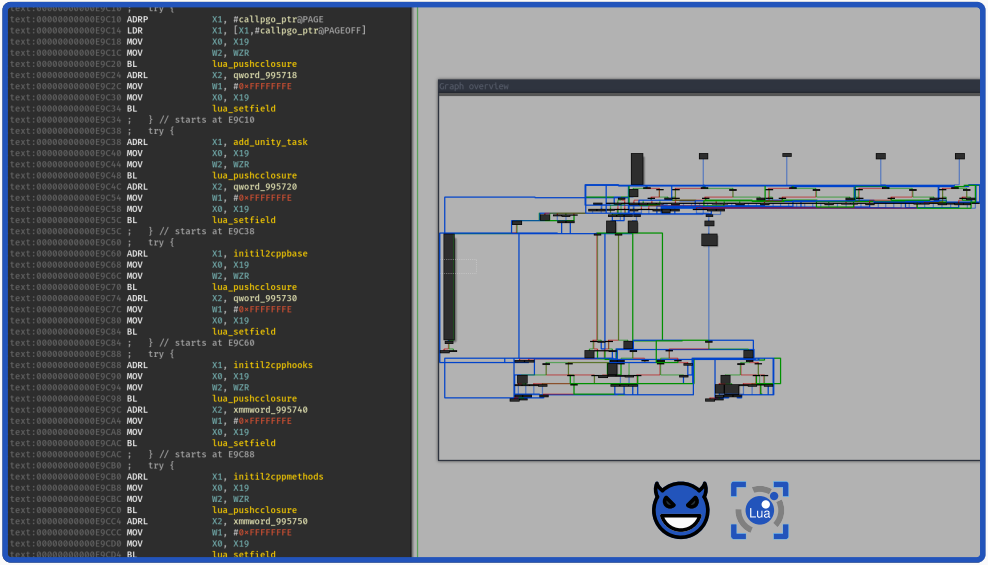
- PGSharp is written in Lua with native bindings for low-level functionalities (e.g. hooking, JNI, ...)
- the Lua logic and the Lua VM are self-contained in `lib/arm64-v8a/libmain.so`
- the Lua VM has been modified to fake the version[1] and to include custom opcodes[2].
- the whole being obfuscated with O-LLVM 😈

[1] Try to make believe a VM based on Lua 5.1 while it is based on Lua 5.3
[2] `OP_RUN`, `OP_GETDOWNVAL`, `OP_OLDTABLE` and, `OP_XXOR`

Native Code: Lua



Native Code: Lua



Java Strings Encryption

PGSharp dynamically loads another APK:

- class and method names are stripped with Proguard
- strings are encoded with a custom function

Java Strings Encryption

```
/* renamed from: q */
public void m106q() {
  String a = C2314GL.m215a(C1576r3.m1597a("FAQgLiQlLw=="), (String) null);
  if (a ≠ null) {
    JSONObject jSONObject = new JSONObject();
    try {
      Context context = C2314GL.f6528c;
      jSONObject.put(C1576r3.m1597a("Agg3FA=="), C1576r3.m1597a("Axg="));
      jSONObject.put(C1576r3.m1597a("Axgj"), C2338UI.m7g(context));
      jSONObject.put(C1576r3.m1597a("BQUmBTQ="), this.f6574s);
      jSONObject.put(C1576r3.m1597a("BQEoHjc+LTE="), ((Boolean) ... );
      jSONObject.put(C1576r3.m1597a("BAUr"), C2338UI.m9f());
      jSONObject.put(C1576r3.m1597a("Gh8g"), Locale.getDefault().getDisplayLanguage());
      jSONObject.put(C1576r3.m1597a("FxMu"), C2338UI.m25a());
      jSONObject.put(C1576r3.m1597a("FBA1"), LayoutInflater$Factory2C0167o.C0178i.m3763e(context));
      jSONObject.put(C1576r3.m1597a("Gx4j"), Build.MODEL);
      String str = Build.VERSION.RELEASE;
    }
}
```

Java Strings Encryption

All the strings are *encrypted* with a hardcoded key. The operation is merely a xor with `vqGqQWCVnDRrNXTR`[3].

[3] The key does not change across the versions

Jadx Plugin

PGSharpPlugin.java

```java
public class PGSharp extends AbstractVisitor {

  private static final Logger LOG = LoggerFactory.getLogger(PGSharp.class);

  public static byte[] decode(byte[] bArr) {
    String KEY_s = new String("vqGqQWCVnDRrNXTR");
    char[] KEY = KEY_s.toCharArray();
    byte[] bArr2 = new byte[bArr.length];
    for (int i = 0; i < bArr.length; i++) {
      byte c = (byte)(KEY[i % KEY_s.length()] & 0xFF);
      bArr2[i] = (byte) (bArr[i] ^ c);
    }
    return bArr2;
  }

  @Override
  public void visit(MethodNode mth) {
    LOG.info("Process: {}", mth.toString());
    if (mth.isNoCode()) {
      return;
    }
    for (BlockNode block : mth.getBasicBlocks()) {
```

NORMAL main </PGSharpPlugin.java PGSharp « java utf-8 18% 33/177≡ :1

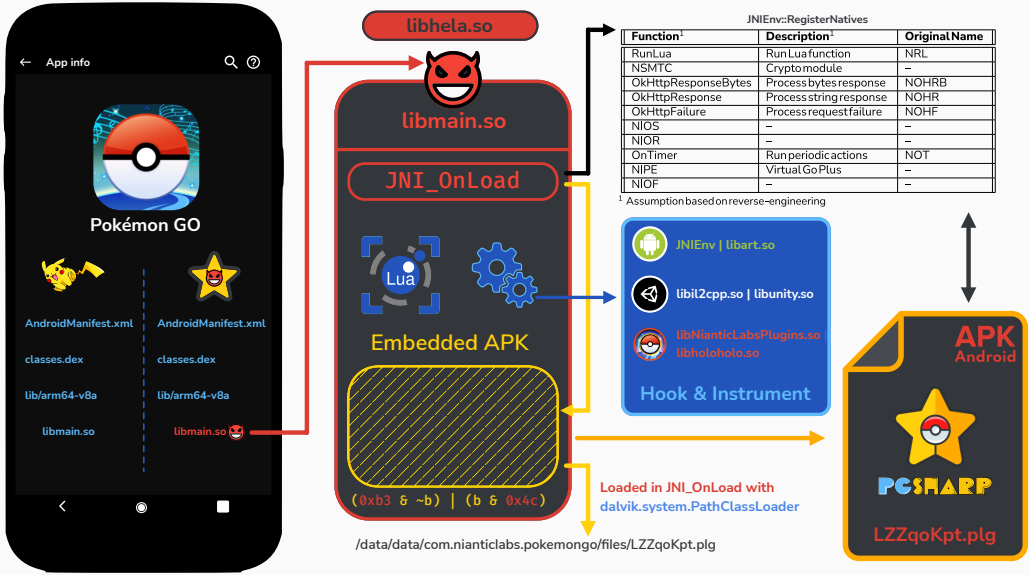
Jadx Plugin

```
/* renamed from: q */
public void m106q() {
  String a = C2314GL.m215a("bug_url", (String) null);
  if (a ≠ null) {
    JSONObject jSONObject = new JSONObject();
    try {
      Context context = C2314GL.f6528c;
      jSONObject.put("type", "ui");
      jSONObject.put("uid", C2338UI.m7g(context));
      jSONObject.put("state", this.f6574s);
      jSONObject.put("spoofing", ((Boolean) C2324PL.m83a("hlspoofing")).booleanValue());
      jSONObject.put("rtl", C2338UI.m9f());
      jSONObject.put("lng", Locale.getDefault().getDisplayLanguage());
      jSONObject.put("abi", C2338UI.m25a());
      jSONObject.put("bar", LayoutInflater$Factory2C0167o.C0178i.m3763e(context));
      jSONObject.put("mod", Build.MODEL);
      String str = Build.VERSION.RELEASE;
    }
}
```

Cheat Engine Overview

Cheat Engine Overview

Cheat Engine Overview



Features Overview

Among the interesting tricks of PGSharp, we find:

- signature bypass
- JNIEnv *proxify*
- hook PokemonGO Unity functions
- bypass SafetyNet attestation with a custom attestation server[4]

[4] `https://tens.pgsharp.com/v1/scc-2-XXX/`

Signature Bypass

Since PGSarp **repackages** the original application, **the signature is not the same** as the genuine PokemonGO app.

⇒ Niantic likely implements integrity checks based on the APK signature.

Signature Bypass

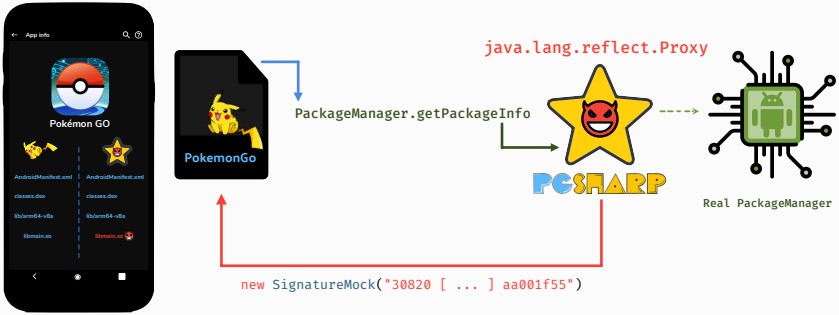


Figure 1: Signature bypass based on a proxy of the Android PackageManager

GPS Spoofing

GPS Spoofing

PokemonGO manages the user's location with three native functions:

- `NianticLocationManager.nativeAddLocationProviders`
- `NianticLocationManager.nativeGpsStatusUpdate`
- `NianticLocationManager.nativeLocationUpdate`

GPS Spoofing

These three functions are implemented in `libNianticLabsPlugin.so` which is obfuscated by the commercial solution.

GPS Spoofing

PGSharp *hook*s these functions to change the location given in the parameters.

- 😈 `NianticLocationManager.nativeAddLocationProviders`
- `NianticLocationManager.nativeGpsStatusUpdate`
- 😈 `NianticLocationManager.nativeLocationUpdate`

GPS Spoofing

These three functions are implemented in `libNianticLabsPlugin.so` which is obfuscated by the commercial solution.

That prevents hooking (?)

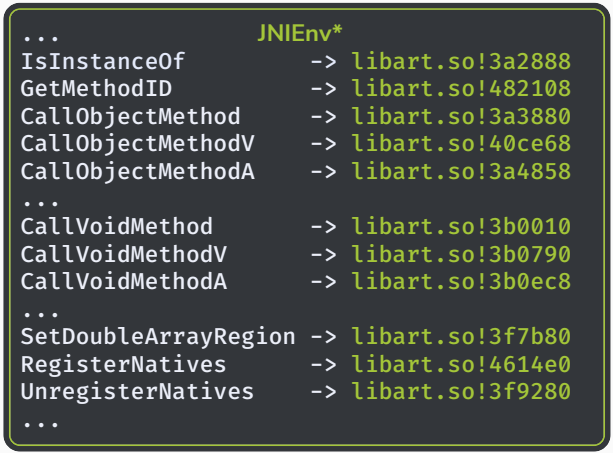
JNIEnv Proxifier

JNIEnv Proxifier

JNIEnv is a pointer to a an **array of function pointers**

e.g. `env->FindClass(...)`

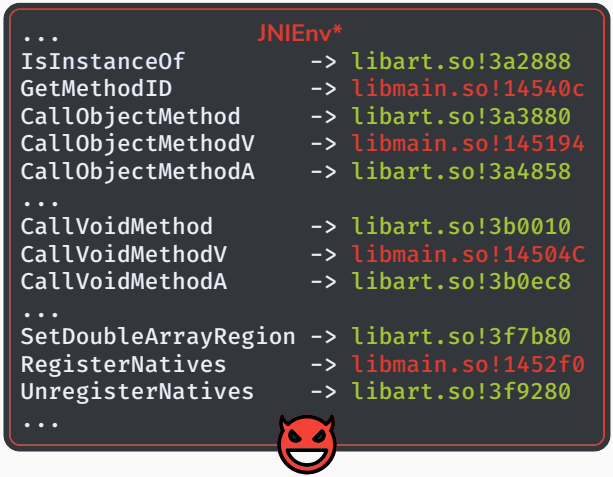
JNIEnv


JNIEnv*
... -> libart.so!3a2888
IsInstanceOf -> libart.so!482108
GetMethodID -> libart.so!3a3880
CallObjectMethod -> libart.so!40ce68
CallObjectMethodV -> libart.so!3a4858
CallObjectMethodA -> libart.so!3b0010
...
CallVoidMethod -> libart.so!3b0790
CallVoidMethodV -> libart.so!3b0ec8
CallVoidMethodA -> libart.so!3f7b80
...
SetDoubleArrayRegion -> libart.so!4614e0
RegisterNatives -> libart.so!3f9280
UnregisterNatives
...

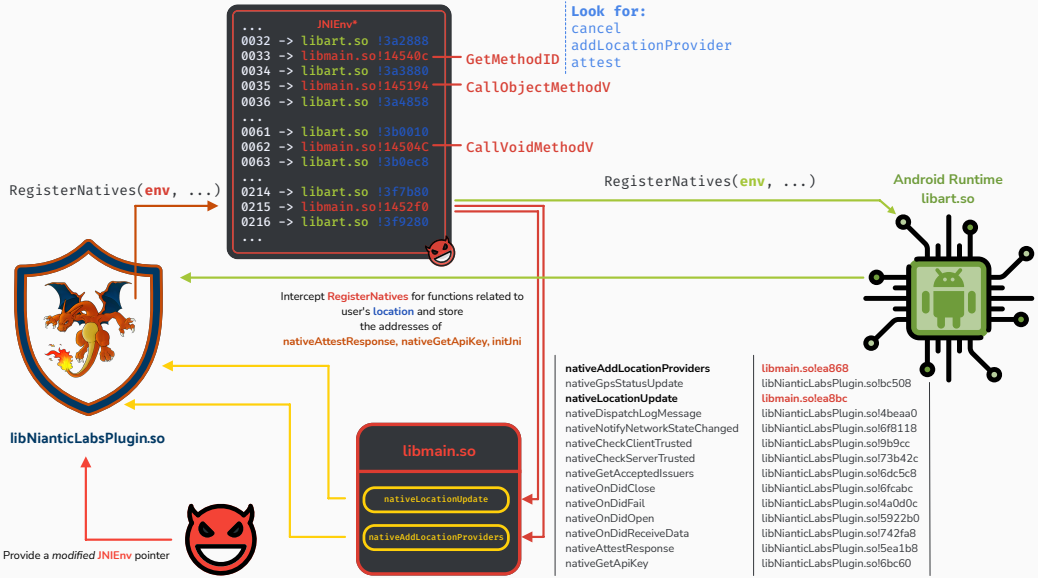
JNIEnv

```
hook("libNianticLabsPlugin.so", "Java_com_nianticlabs_nia_unity_UnityUtil_nativeInit",
    [] (NativeListener*, GumInvocationContext* ic, bool is_enter) {
     if (is_enter) {
      auto* ptr = *reinterpret_cast<uintptr_t**>(ic->cpu_context->x[0]);
      const size_t nb_elements = sizeof(JNINativeInterface) / sizeof(uintptr_t);
      for (size_t i = 0; i < nb_elements; ++i) {
       target_info_t info = pointer_info(ptr[i]);
       spdlog::info("{:04d} → {}!{:x}", i, info.libname, info.offset);
      }
     }
    });
```

JNIEnv


JNIEnv*
... -> libart.so!3a2888
IsInstanceOf -> libart.so!3a2888
GetMethodID -> libmain.so!14540c
CallObjectMethod -> libart.so!3a3880
CallObjectMethodV -> libmain.so!145194
CallObjectMethodA -> libart.so!3a4858
...
CallVoidMethod -> libart.so!3b0010
CallVoidMethodV -> libmain.so!14504C
CallVoidMethodA -> libart.so!3b0ec8
...
SetDoubleArrayRegion -> libart.so!3f7b80
RegisterNatives -> libmain.so!1452f0
UnregisterNatives -> libart.so!3f9280
...

JNIEnv


JNIEnv*
... -> libart.so !3a2888
0032 -> libart.so !3a2888
0033 -> libmain.so!14540c
0034 -> libart.so !3a3880
0035 -> libmain.so!145194
0036 -> libart.so !3a4858
...
0061 -> libart.so !3b0010
0062 -> libmain.so!14504C
0063 -> libart.so !3b0ec8
...
0214 -> libart.so !3f7b80
0215 -> libmain.so!1452f0
0216 -> libart.so !3f9280
...
GetMethodID
CallObjectMethodV
CallVoidMethodV
Look for:
cancel
addLocationProvider
attest
RegisterNatives(env, ...)
RegisterNatives(env, ...)
Android Runtime
libart.so
Intercept RegisterNatives for functions related to user's location and store the addresses of nativeAttestResponse, nativeGetApiKey, initJni
libNianticLabsPlugin.so
libmain.so
nativeLocationUpdate
nativeAddLocationProviders
Provide a modified JNIEnv pointer
nativeAddLocationProviders
nativeGpsStatusUpdate
nativeLocationUpdate
nativeDispatchLogMessage
nativeNotifyNetworkStateChanged
nativeCheckClientTrusted
nativeCheckServerTrusted
nativeGetAcceptedIssuers
nativeOnDidClose
nativeOnDidFail
nativeOnDidOpen
nativeOnDidReceiveData
nativeAttestResponse
nativeGetApiKey
...
libmain.so!ea868
libNianticLabsPlugin.so!bc508
libmain.so!ea8bc
libNianticLabsPlugin.so!4beaa0
libNianticLabsPlugin.so!6f8118
libNianticLabsPlugin.so!9b9cc
libNianticLabsPlugin.so!73b42c
libNianticLabsPlugin.so!6dc5c8
libNianticLabsPlugin.so!6fcabc
libNianticLabsPlugin.so!4a0d0c
libNianticLabsPlugin.so!5922b0
libNianticLabsPlugin.so!742fa8
libNianticLabsPlugin.so!5ea1b8
libNianticLabsPlugin.so!6bc60
...

How to prevent such a cheat?

How to prevent such a cheat?

It’s a cat & mouse game and none of the following elements are cheaterproof.

How to prevent such a cheat?

Signature Bypass

For the specific case of PGSharp, one could check if a given class is *proxyfied* with `Proxy.isProxyClass`.

How to prevent such a cheat?

Signature Bypass

More generally, integrity checks should also be performed on native libraries[5] and DEX files.

[5] Compared to iOS, ELF binaries are not signed

How to prevent such a cheat?

JNIEnv proxifier

The JNI functions in **libNianticLabsPlugin.so** could check the integrity of the JNIEnv structure before executing the function.

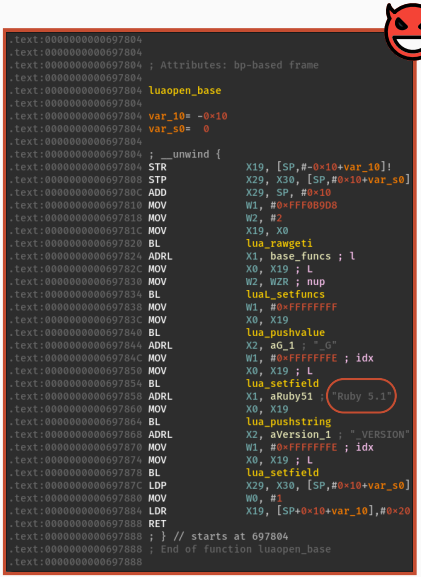
In particular, **all the JNIEnv functions should point to `libart.so`**.

Conclusion

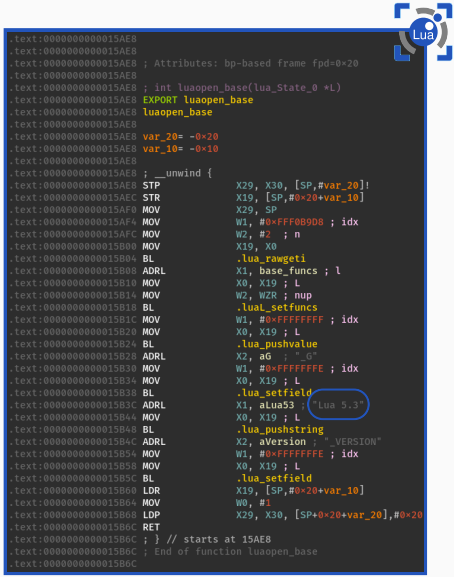
Conclusion

- A very nice and challenging cheat-engine! ~~Well done!~~
- In this case, Lua VM + O-LLVM are not enough to prevent reverse-engineering.
- PGSharp authors seem to have a strong background in reverse-engineering and Android.

Ruby?



```
.text:0000000000697804
.text:0000000000697804
.text:0000000000697804 ; Attributes: bp-based frame
.text:0000000000697804
.text:0000000000697804 luaopen_base
.text:0000000000697804
.text:0000000000697804 var_10= -0x10
.text:0000000000697804 var_s0=  0
.text:0000000000697804
.text:0000000000697804 ; __unwind {
.text:0000000000697804 STR             X19, [SP,#-0x10+var_10]!
.text:0000000000697808 STP             X29, X30, [SP,#0x10+var_s0]
.text:000000000069780C ADD             X29, SP, #0x10
.text:0000000000697810 MOV             W1, #0xFFF0B9D8
.text:0000000000697818 MOV             W2, #2
.text:000000000069781C MOV             X19, X0
.text:0000000000697820 BL              lua_rawgeti
.text:0000000000697824 ADRL            X1, base_funcs ; l
.text:000000000069782C MOV             X0, X19 ; L
.text:0000000000697830 MOV             W2, WZR ; nup
.text:0000000000697834 BL              luaL_setfuncs
.text:0000000000697838 MOV             W1, #0xFFFFFFFF
.text:000000000069783C MOV             X0, X19
.text:0000000000697840 BL              lua_pushvalue
.text:0000000000697844 ADRL            X2, aG_1 ; "_G"
.text:000000000069784C MOV             W1, #0xFFFFFFFE ; idx
.text:0000000000697850 MOV             X0, X19 ; L
.text:0000000000697854 BL              lua_setfield
.text:0000000000697858 ADRL            X1, aRuby51 ; "Ruby 5.1"
.text:0000000000697860 MOV             X0, X19
.text:0000000000697864 BL              lua_pushstring
.text:0000000000697868 ADRL            X2, aVersion_1 ; "_VERSION"
.text:0000000000697870 MOV             W1, #0xFFFFFFFE ; idx
.text:0000000000697874 MOV             X0, X19 ; L
.text:0000000000697878 BL              lua_setfield
.text:000000000069787C LDP             X29, X30, [SP,#0x10+var_s0]
.text:0000000000697880 MOV             W0, #1
.text:0000000000697884 LDR             X19, [SP+0x10+var_10],#0x20
.text:0000000000697888 RET
.text:0000000000697888 ; } // starts at 697804
.text:0000000000697888 ; End of function luaopen_base
.text:0000000000697888
```



```
.text:000000000015AE8
.text:000000000015AE8
.text:000000000015AE8 ; Attributes: bp-based frame fpd=0x20
.text:000000000015AE8
.text:000000000015AE8 ; int luaopen_base(lua_State_0 *L)
.text:000000000015AE8 EXPORT luaopen_base
.text:000000000015AE8 luaopen_base
.text:000000000015AE8
.text:000000000015AE8 var_20= -0x20
.text:000000000015AE8 var_10= -0x10
.text:000000000015AE8
.text:000000000015AE8 ; __unwind {
.text:000000000015AE8 STP             X29, X30, [SP,#var_20]!
.text:000000000015AEC STR             X19, [SP,#0x20+var_10]
.text:000000000015AF0 MOV             X29, SP
.text:000000000015AF4 MOV             W1, #0xFFF0B9D8 ; idx
.text:000000000015AFC MOV             W2, #2  ; n
.text:000000000015B00 MOV             X19, X0
.text:000000000015B04 BL              .lua_rawgeti
.text:000000000015B08 ADRL            X1, base_funcs ; l
.text:000000000015B10 MOV             X0, X19 ; L
.text:000000000015B14 MOV             W2, WZR ; nup
.text:000000000015B18 BL              .luaL_setfuncs
.text:000000000015B1C MOV             W1, #0xFFFFFFFF ; idx
.text:000000000015B20 MOV             X0, X19 ; L
.text:000000000015B24 BL              .lua_pushvalue
.text:000000000015B28 ADRL            X2, aG  ; "_G"
.text:000000000015B30 MOV             W1, #0xFFFFFFFE ; idx
.text:000000000015B34 MOV             X0, X19 ; L
.text:000000000015B38 BL              .lua_setfield
.text:000000000015B3C ADRL            X1, aLua53 ; "Lua 5.3"
.text:000000000015B44 MOV             X0, X19 ; L
.text:000000000015B48 BL              .lua_pushstring
.text:000000000015B4C ADRL            X2, aVersion ; "_VERSION"
.text:000000000015B54 MOV             W1, #0xFFFFFFFE ; idx
.text:000000000015B58 MOV             X0, X19 ; L
.text:000000000015B5C BL              .lua_setfield
.text:000000000015B60 LDR             X19, [SP,#0x20+var_10]
.text:000000000015B64 MOV             W0, #1
.text:000000000015B68 LDP             X29, X30, [SP+0x20+var_20],#0x20
.text:000000000015B6C RET
.text:000000000015B6C ; } // starts at 15AE8
.text:000000000015B6C ; End of function luaopen_base
.text:000000000015B6C
```

Ruby?

```
.text:00000000000CA574
.text:00000000000CA574 loc_CA574
.text:00000000000CA574 ;   try {
.text:00000000000CA574 ADRL            X1, aRubytkCallAtte ; "rubytk::call() attempt to call global `" ...
.text:00000000000CA57C MOV             X0, X19
.text:00000000000CA580 MOV             X2, X22
.text:00000000000CA584 BL              lua_tinker_print_error
```

```
emplate<typename RVal>
RVal call(lua_State* L, const char* name)
{
  lua_pushcclosure(L, on_error, 0);
  int errfunc = lua_gettop(L);

  lua_pushstring(L, name);
  lua_gettable(L, LUA_GLOBALSINDEX);

  if(lua_isfunction(L,-1))
  {
    lua_pcall(L, 0, 1, errfunc);
  }
  else
  {
    print_error(L, "lua_tinker::call() attempt to call global `%s' (not a function)", name);
  }

  lua_remove(L, errfunc);
  return pop<RVal>(L);
}
```

Thank You

 https://github.com/romainthomas/pgsharp

 @rh0main

Thank You

Thank You!

Bonus: il2cpp hook

Unity hook on libil2cpp.so

```
sub_6C97CC

var_10= -0x10
var_s0=  0

; __unwind { // __gxx_personality_v0
STP             X20, X19, [SP,#-0x10+var_10]!
STP             X29, X30, [SP,#0x10+var_s0]
ADD             X29, SP, #0x10
ADRP            X8, #off_993F80@PAGE
LDR             X8, [X8,#off_993F80@PAGEOFF]
ADD             X19, X8, #4
MOV             X0, X19 ; this
BL              std::recursive_mutex::lock(void)
ADRP            X8, #off_993288@PAGE
LDR             X8, [X8,#off_993288@PAGEOFF]
MOV             W9, #0x4BFEEA0
LDR             X8, [X8]
ADD             X0, X8, X9
;  try {
ADRP            X1, #sub_6C983C@PAGE
ADRP            X2, #qword_14D3678@PAGE
ADD             X1, X1, #sub_6C983C@PAGEOFF
ADD             X2, X2, #qword_14D3678@PAGEOFF
BL              sub_98410
;  } // starts at 6C9804
LDP             X29, X30, [SP,#0x10+var_s0]
MOV             X0, X19 ; this
LDP             X20, X19, [SP+0x10+var_10],#0x20
B               std::recursive_mutex::unlock(void)
```

UnityEngine.Application$$OpenURL

DobbyHook()

"Dobby a lightweight, multi-platform, multi-architecture exploit hook framework."
https://github.com/jmpews/Dobby

on_OpenURL

```
sub_6C983C

var_10= -0x10
var_s0=  0

; __unwind {
STP             X20, X19, [SP,#-0x10+var_10]!
STP             X29, X30, [SP,#0x10+var_s0]
ADD             X29, SP, #0x10
LDR             W8, [X0,#0x10]
MOV             X19, X0
CMP             W8, #0x47 ; 'G'
B.NE            loc_6C9870
```

```
ADRP            X0, #aHttpsPlayGoogl@PAGE ; "https://play.google.com/store/apps/deta"...
ADD             X1, X19, #0x14 ; s2
ADD             X0, X0, #aHttpsPlayGoogl@PAGEOFF ; "https://play.google.com/store/apps/deta"...
MOV             W2, #0x8E ; n
BL              _memcmp
CBZ             W0, loc_6C9880
```

```
loc_6C9870
ADRP            X8, #qword_14D3678@PAGE
LDR             X1, [X8,#qword_14D3678@PAGEOFF]
MOV             X0, X19
B               loc_6C98BC
```

```
loc_6C9880
ADRP            X8, #il2cpp_System_String_CreateString_ptr@PAGE
LDR             X8, [X8,#il2cpp_System_String_CreateString_ptr@PAGEOFF]
ADRL            X19, aHttpsWwwPgshar ; "https://www.pgsharp.com/"
MOV             W1, #0x19 ; size_t
LDR             X20, [X8]
MOV             X0, X19 ; char *
BL              ___strlen_chk
MOV             X3, X0
MOV             X0, XZR
MOV             X1, X19
MOV             W2, WZR
BLR             X20
ADRP            X8, #qword_14D3678@PAGE
LDR             X1, [X8,#qword_14D3678@PAGEOFF]
```

```
loc_6C98BC
LDP             X29, X30, [SP,#0x10+var_s0]
LDP             X20, X19, [SP+0x10+var_10],#0x20
BR              X1
; } // starts at 6C983C
; End of function sub_6C983C
```

PGSharp hooks **OpenURL** from Unity to **redirect** the Play Store URL of PokemonGO to **www.pgsharp.com**