

(Не)Безопасность 101

Григорий Джанелидзе



Danil Kravchenko
Dec 6, 2016 · 8 min read

How To Make Your Android Application Secured



Ali Muzaffar



Yonatan V. Levin
CO-Founder & CTO of KolGene, Founder of Android Academy, Google Developer Expert
Jan 3 · 11 min read

Bang! Bang! You have been hacked.



pretty hate machine @fuckingsun · Jan 10
It's called Medium because the articles are neither rare nor well-done.

2

211

536



Ah the sweet, enticing world of Android. My first true love, always beckoning me, whispering seductively in my ear:

Про что поговорим:

- Как делать точно не надо
- Как делать скорее всего не надо
- Какие инструменты существуют для разных задач
- Как с этим всем жить

101 is a topic for beginners in any area.

It has all the basic principles and concepts that is expected in a particular field.

– *Wikipedia*

Вся информация представлена
исключительно в ознакомительных и
образовательных целях.



Храним ключи

Demo

Храним ключи

Or you can use Java Native Interface (JNI). It is harder to decompile C/C++ compiled code. Decompilers like JaDx, dex2jar won't help because they are Java-oriented decompilers.

All the methods above can add extra time to the “health” of your app. Use something is better than use nothing.



Храним ключи

Demo

Храним ключи





Храним ключи

<https://github.com/KeepSafe/ReLinker>

Храним ключи

Итоги

**Хранить в коде
и шифровать**

Хранить в коде

Хранить в манифесте/ресурсах

Хранить в нативной библиотеке и доставать через JNI



Храним ключи

Инструменты

- apktool
- JD-GUI, ByteCode Viewer, ...
- IDA Pro, Hopper, ...
 - если нужны только строки: `man 1 strings`



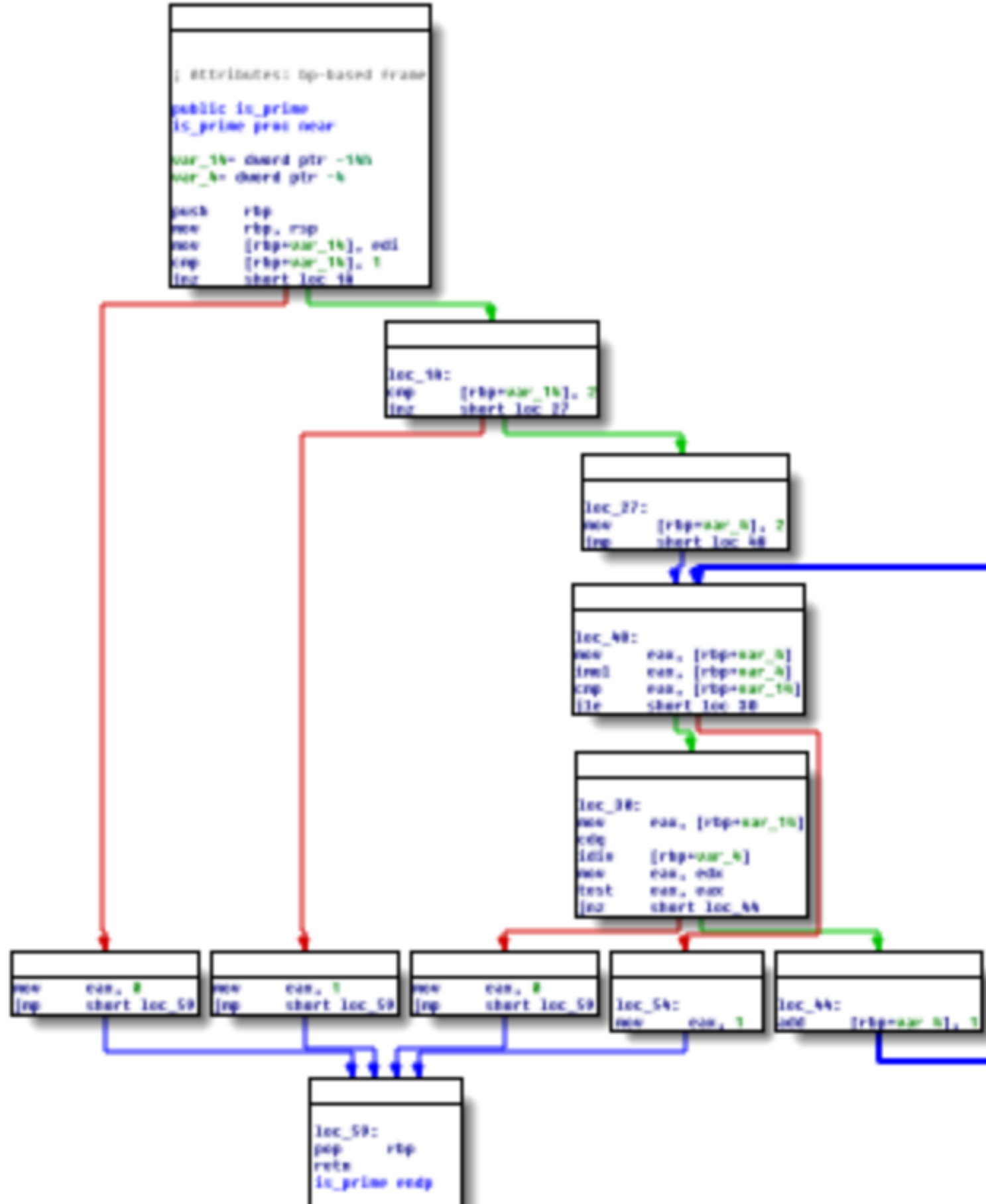
Обфусцируем

Demo

Обфусцируем Инструменты

- smali / backsmali
- <https://github.com/CalebFenton/simplify>
- frida, xposed, ...
- <https://github.com/xoreaxeaxe/movfuscator>

Control flow graphs:

GCC	M/o/Vfuscator
 <pre> graph TD Entry["; Attributes: bp-based frame public is_prime is_prime proc near var_14= dword ptr -14h var_4= dword ptr -4 push rbp mov rbp, esp mov [rbp+var_14], edi xor [rbp+var_14], 1 jmp short loc_14"] loc_14["loc_14: xor [rbp+var_14], 1 jmp short loc_27"] loc_27["loc_27: mov [rbp+var_4], 2 jmp short loc_40"] loc_40["loc_40: mov eax, [rbp+var_4] imul eax, [rbp+var_4] xor eax, [rbp+var_14] jle short loc_38"] loc_38["loc_38: mov eax, [rbp+var_14] xor eax, eax leah [rbp+var_4], eax mov eax, eax test eax, eax jnz short loc_44"] loc_59["loc_59: mov eax, 0 jmp short loc_59"] loc_5a["loc_5a: mov eax, 1 jmp short loc_59"] loc_5b["loc_5b: mov eax, 0 jmp short loc_59"] loc_54["loc_54: mov eax, 1"] loc_44["loc_44: xor [rbp+var_4], 1"] loc_59_exit["loc_59: pop rbp ret 4 is_prime endp"] Entry -- red --> loc_59 Entry -- green --> loc_14 loc_14 -- red --> loc_59 loc_14 -- green --> loc_27 loc_27 -- blue --> loc_40 loc_27 -- green --> loc_44 loc_40 -- red --> loc_59 loc_40 -- green --> loc_38 loc_38 -- red --> loc_59 loc_38 -- green --> loc_44 loc_44 -- blue --> loc_54 loc_44 -- green --> loc_44 loc_54 -- blue --> loc_59 loc_59 -- blue --> loc_59_exit </pre>	 <pre> graph TD Entry["; Attributes: bp-based frame public is_prime is_prime proc near var_14= dword ptr -14h var_4= dword ptr -4 push rbp mov rbp, esp mov [rbp+var_14], edi xor [rbp+var_14], 1 jmp short loc_14"] loc_14["loc_14: xor [rbp+var_14], 1 jmp short loc_27"] loc_27["loc_27: mov [rbp+var_4], 2 jmp short loc_40"] loc_40["loc_40: mov eax, [rbp+var_4] imul eax, [rbp+var_4] xor eax, [rbp+var_14] jle short loc_38"] loc_38["loc_38: mov eax, [rbp+var_14] xor eax, eax leah [rbp+var_4], eax mov eax, eax test eax, eax jnz short loc_44"] loc_59["loc_59: mov eax, 0 jmp short loc_59"] loc_5a["loc_5a: mov eax, 1 jmp short loc_59"] loc_5b["loc_5b: mov eax, 0 jmp short loc_59"] loc_54["loc_54: mov eax, 1"] loc_44["loc_44: xor [rbp+var_4], 1"] loc_59_exit["loc_59: pop rbp ret 4 is_prime endp"] Entry -- red --> loc_59 Entry -- green --> loc_14 loc_14 -- red --> loc_59 loc_14 -- green --> loc_27 loc_27 -- blue --> loc_40 loc_27 -- green --> loc_44 loc_40 -- red --> loc_59 loc_40 -- green --> loc_38 loc_38 -- red --> loc_59 loc_38 -- green --> loc_44 loc_44 -- blue --> loc_54 loc_44 -- green --> loc_44 loc_54 -- blue --> loc_59 loc_59 -- blue --> loc_59_exit </pre>

Обфусцируем

Итоги

- ProGuard полезный и его в любом случае надо включить
 - Но он не спасет
- Ревью любых изменений в конфиге ProGuard'a ≠ обычное кодревью
 - декомпилируйте и попробуйте реверснуть

Обфусцируем

Итоги

- Я уже говорил, что не надо использовать JNI для «безопасности»?
- ProGuard не трогает все методы с модификатором `native`
- Выпиливайте логи, имена переменных, ...



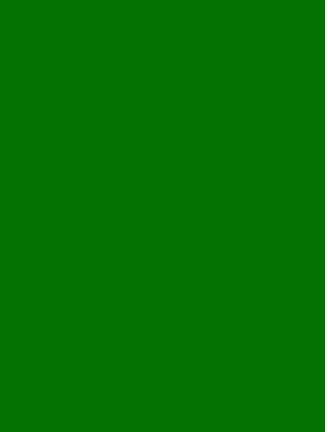
Обфусцируем Зло

- Consumer ProGuard Files

Обфусцируем Зло

(ツ)/

(спасибо Google, стало удобнее)



Сопровождаем

Demo

Сопровождаемся

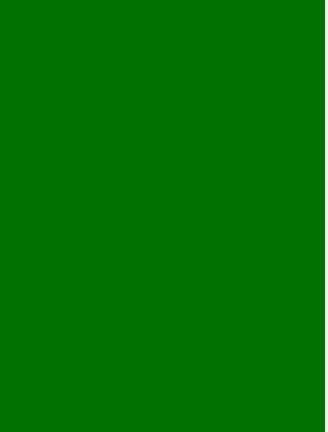
Итоги

- `Debug.isDebuggerConnected()`
 - не надо
- лайфхаки с `ptrace`
 - могут быть полезными
- миграция с json'а на `protobuf`, `flatbuf`, ...
 - поможет, но не сильно

Сопровождаемся

Итоги

- SSL pinning нужен и спасёт
 - ...но не от реверсинжиниринга



Сопровождаемся Инструментами

- man 1 ps
- gdb, lldb, ...
- frida, xposed, ...

Как быть?

- Пользоваться проприетарными решениями
 - DexGuard, SecNeo, DexProtector, ...
- Если есть ресурсы и время, то делать свое

Заключение

- Всегда нуж
- Проверяйте
-



на практике

ИТЬ

И если что:

На iOS всё не сильно лучше
(но это тема отдельного выступления)

Спасибо за внимание

t.me/androidguards