Security in IoT Ecosystem

Module 7

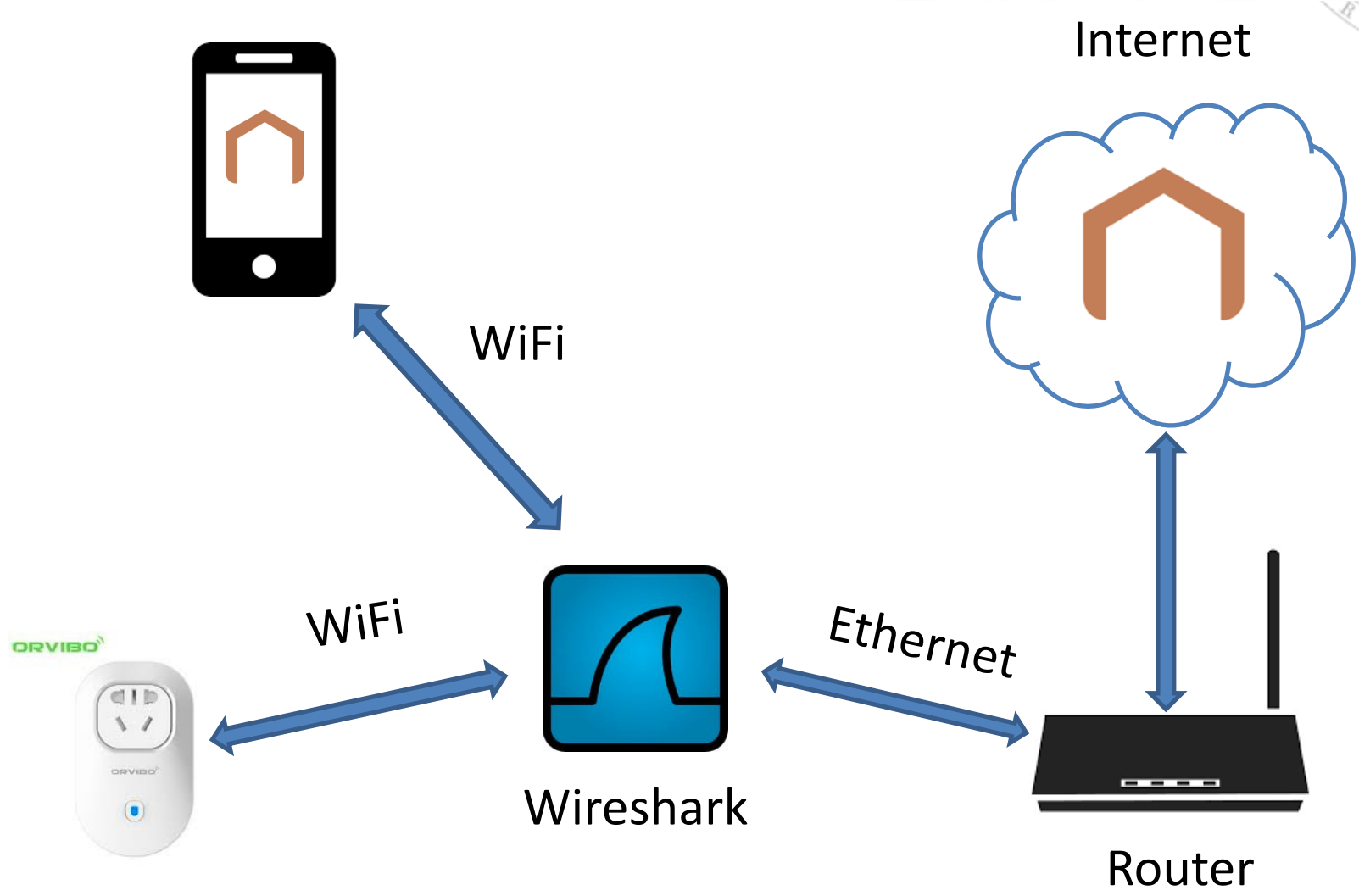
Smart Socket Pentest
Part III

Prof.: Joaquín Recas

Smart Socket Initial Setup

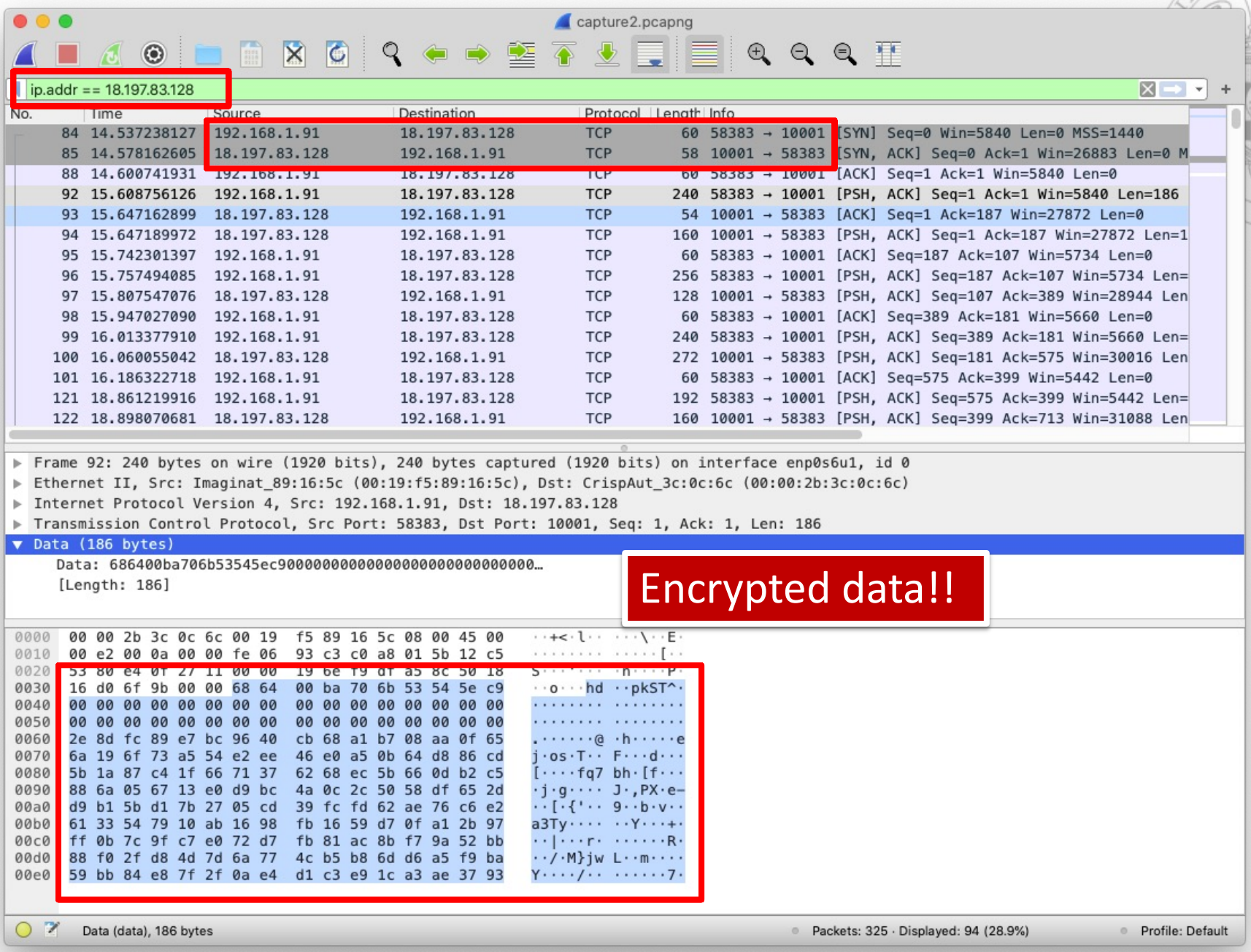
1. Initial Setup
2. Capture Traffic
3. Traffic analysis
4. Apk Analysis

Study communications:



Information we know

- IP numbers:
  - Orvibo server, Raspberry Pi, Smartphone & Smartsocket
- There are messages with payload between the Smartplug and the Orvibo server
  - Encrypted content!


capture2.pcapng
ip.addr == 18.197.83.128
No. Time Source Destination Protocol Length Info
84 14.537238127 192.168.1.91 18.197.83.128 TCP 60 58383 → 10001 [SYN] Seq=0 Win=5840 Len=0 MSS=1440
85 14.578162605 18.197.83.128 192.168.1.91 TCP 58 10001 → 58383 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 M
88 14.600741931 [ACK] Seq=1 Ack=1 Win=5840 Len=0
92 15.608756126 192.168.1.91 18.197.83.128 TCP 240 58383 → 10001 [PSH, ACK] Seq=1 Ack=1 Win=5840 Len=186
93 15.647162899 18.197.83.128 192.168.1.91 TCP 54 10001 → 58383 [ACK] Seq=1 Ack=187 Win=27872 Len=0
94 15.647189972 18.197.83.128 192.168.1.91 TCP 160 10001 → 58383 [PSH, ACK] Seq=1 Ack=187 Win=27872 Len=1
95 15.742301397 192.168.1.91 18.197.83.128 TCP 60 58383 → 10001 [ACK] Seq=187 Ack=107 Win=5734 Len=0
96 15.757494085 192.168.1.91 18.197.83.128 TCP 256 58383 → 10001 [PSH, ACK] Seq=187 Ack=107 Win=5734 Len=
97 15.807547076 18.197.83.128 192.168.1.91 TCP 128 10001 → 58383 [PSH, ACK] Seq=107 Ack=389 Win=28944 Len
98 15.947027090 192.168.1.91 18.197.83.128 TCP 60 58383 → 10001 [ACK] Seq=389 Ack=181 Win=5660 Len=0
99 16.013377910 192.168.1.91 18.197.83.128 TCP 240 58383 → 10001 [PSH, ACK] Seq=389 Ack=181 Win=5660 Len=
100 16.060055042 18.197.83.128 192.168.1.91 TCP 272 10001 → 58383 [PSH, ACK] Seq=181 Ack=575 Win=30016 Len
101 16.186322718 192.168.1.91 18.197.83.128 TCP 60 58383 → 10001 [ACK] Seq=575 Ack=399 Win=5442 Len=0
121 18.861219916 192.168.1.91 18.197.83.128 TCP 192 58383 → 10001 [PSH, ACK] Seq=575 Ack=399 Win=5442 Len=
122 18.898070681 18.197.83.128 192.168.1.91 TCP 160 10001 → 58383 [PSH, ACK] Seq=399 Ack=713 Win=31088 Len
Frame 92: 240 bytes on wire (1920 bits), 240 bytes captured (1920 bits) on interface enp0s6u1, id 0
Ethernet II, Src: Imaginat_89:16:5c (00:19:f5:89:16:5c), Dst: CrispAut_3c:0c:6c (00:00:2b:3c:0c:6c)
Internet Protocol Version 4, Src: 192.168.1.91, Dst: 18.197.83.128
Transmission Control Protocol, Src Port: 58383, Dst Port: 10001, Seq: 1, Ack: 1, Len: 186
Data (186 bytes)
Data: 686400ba706b53545ec900000000000000000000000000…
[Length: 186]
Encrypted data!!
0000 00 00 2b 3c 0c 6c 00 19 f5 89 16 5c 08 00 45 00
0010 00 e2 00 0a 00 00 fe 06 93 c3 c0 a8 01 5b 12 c5
0020 53 80 e4 0f 27 11 00 00 19 6e f9 df a5 8c 50 18
0030 16 d0 6f 9b 00 00 68 64 00 ba 70 6b 53 54 5e c9
0040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060 2e 8d fc 89 e7 bc 96 40 cb 68 a1 b7 08 aa 0f 65
0070 6a 19 6f 73 a5 54 e2 ee 46 e0 a5 0b 64 d8 86 cd
0080 5b 1a 87 c4 1f 66 71 37 62 68 ec 5b 66 0d b2 c5
0090 88 6a 05 67 13 e0 d9 bc 4a 0c 2c 50 58 df 65 2d
00a0 d9 b1 5b d1 7b 27 05 cd 39 fc fd 62 ae 76 c6 e2
00b0 61 33 54 79 10 ab 16 98 fb 16 59 d7 0f a1 2b 97
00c0 ff 0b 7c 9f c7 e0 72 d7 fb 81 ac 8b f7 9a 52 bb
00d0 88 f0 2f d8 4d 7d 6a 77 4c b5 b8 6d d6 a5 f9 ba
00e0 59 bb 84 e8 7f 2f 0a e4 d1 c3 e9 1c a3 ae 37 93
Data (data), 186 bytes
Packets: 325 · Displayed: 94 (28.9%)
Profile: Default

Next steps

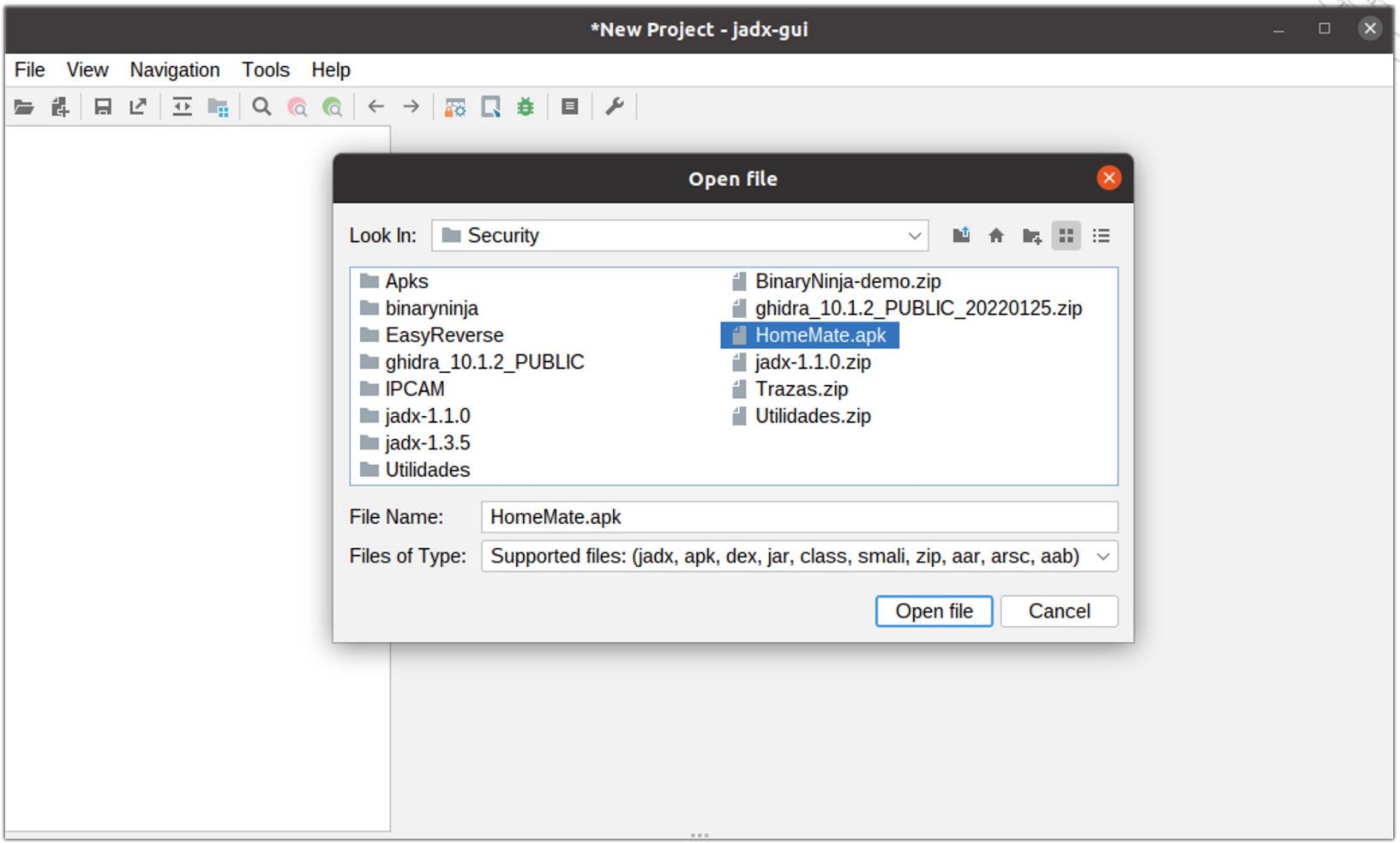
- Can we decrypt the messages?
  - We need method and encryption key
- Where can we find the key?
  - Inside the device
  - On the Orvibo server
  - **In the HomeMate App**

JADX

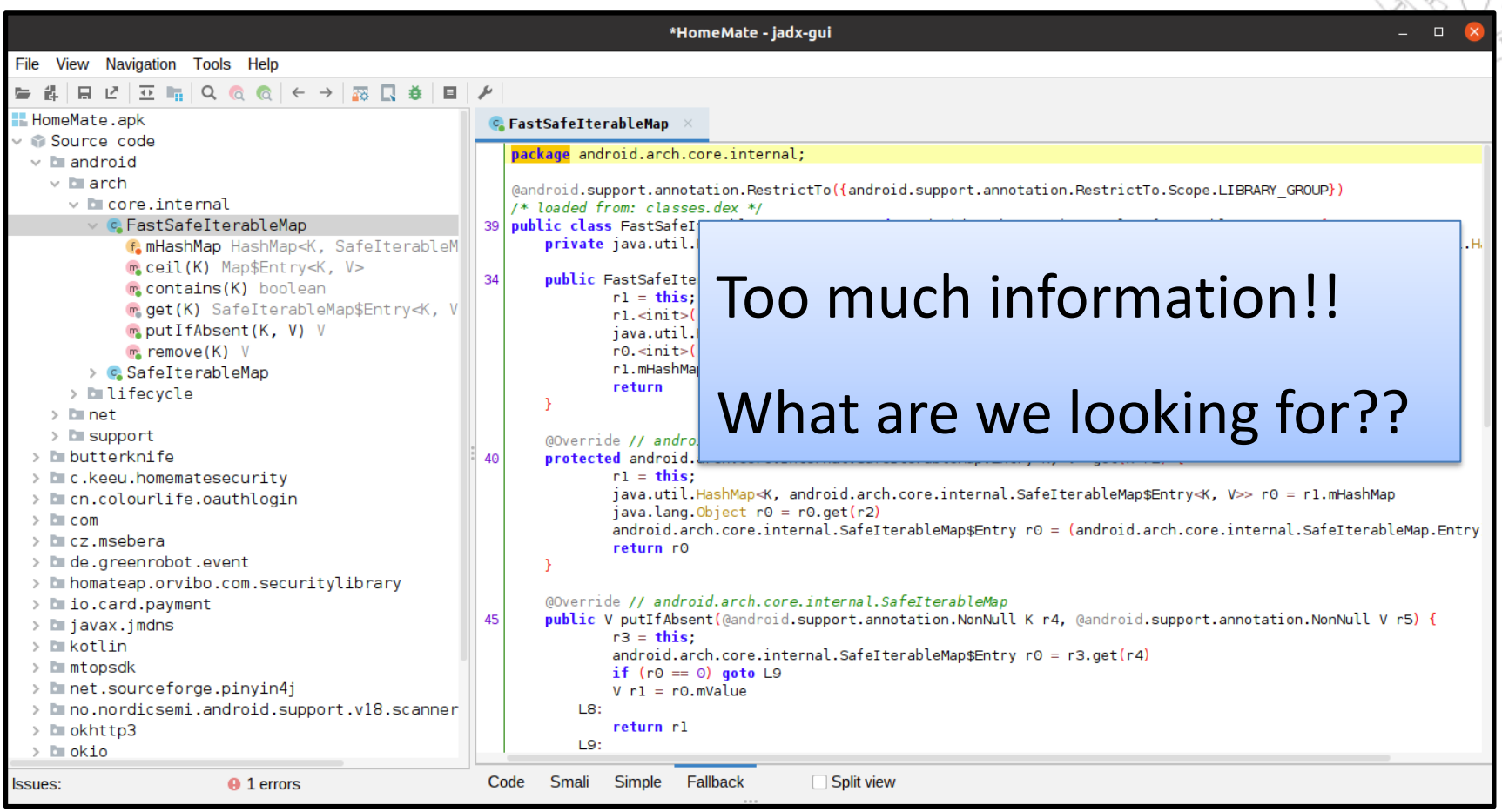
- Dex (Dalvik Executable) to Java decompiler
  - Command line and GUI tools for produce Java source code from Android **Dex** and **Apk** files
- Main features:
  - Decompile Dalvik bytecode to java classes from **APK**, **dex**, **aar** and **zip** files
  - Decode *AndroidManifest.xml* and other resources from *resources.arsc*
  - Deobfuscator included
- `jadx-gui` features:
  - View **decompiled code** with highlighted syntax
  - Jump to declaration
  - Find usage
  - Full text search

https://github.com/skylot/jadx

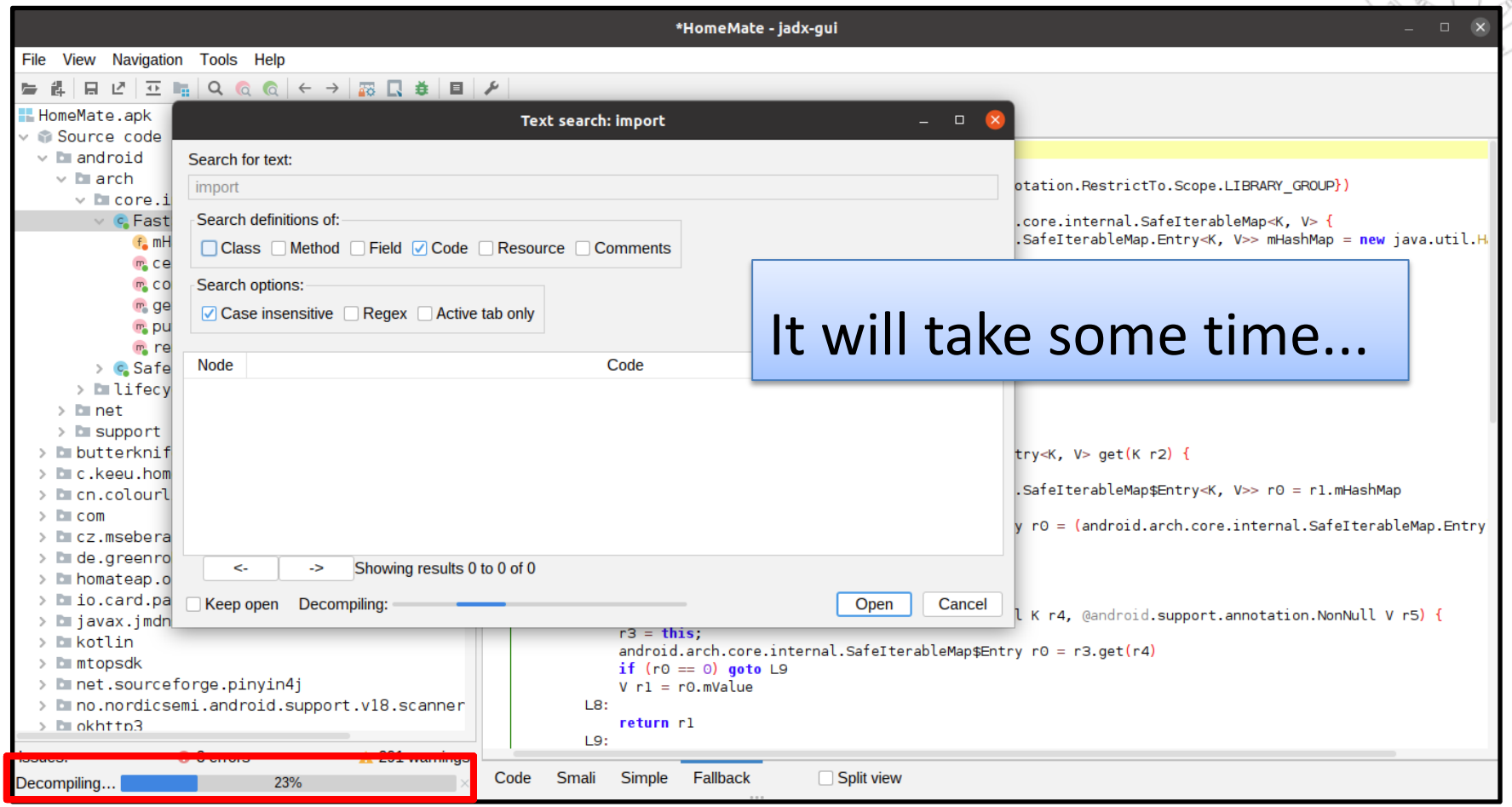
jadx-gui


*New Project - jadx-gui
File View Navigation Tools Help
Open file
Look In: Security
Apks
binaryninja
EasyReverse
ghidra_10.1.2_PUBLIC
IPCAM
jadx-1.1.0
jadx-1.3.5
Utilidades
BinaryNinja-demo.zip
ghidra_10.1.2_PUBLIC_20220125.zip
HomeMate.apk
jadx-1.1.0.zip
Trazas.zip
Utilidades.zip
File Name: HomeMate.apk
Files of Type: Supported files: (jadx, apk, dex, jar, class, smali, zip, aar, arsc, aab)
Open file
Cancel

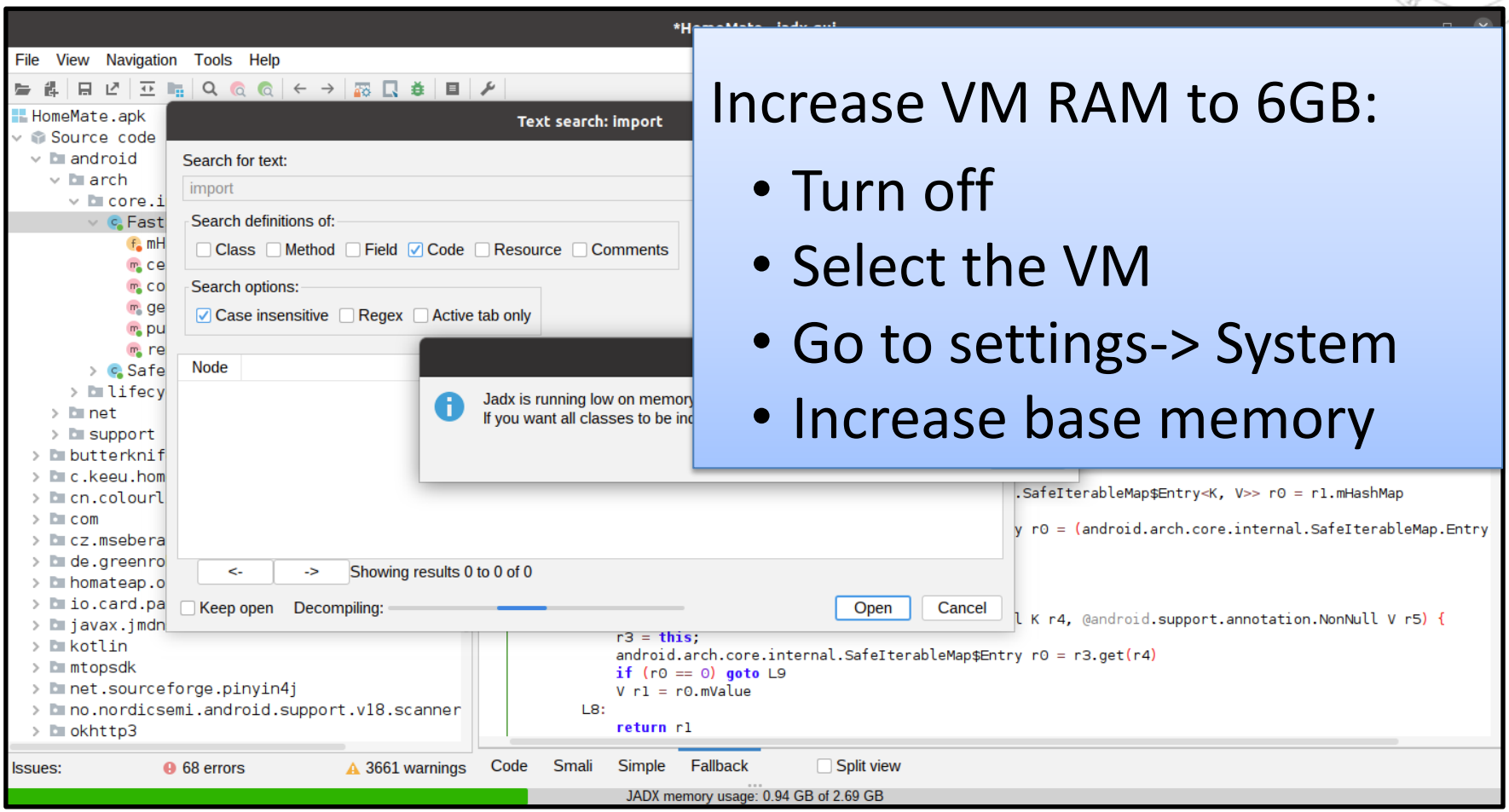
jadx-gui: HomeMate



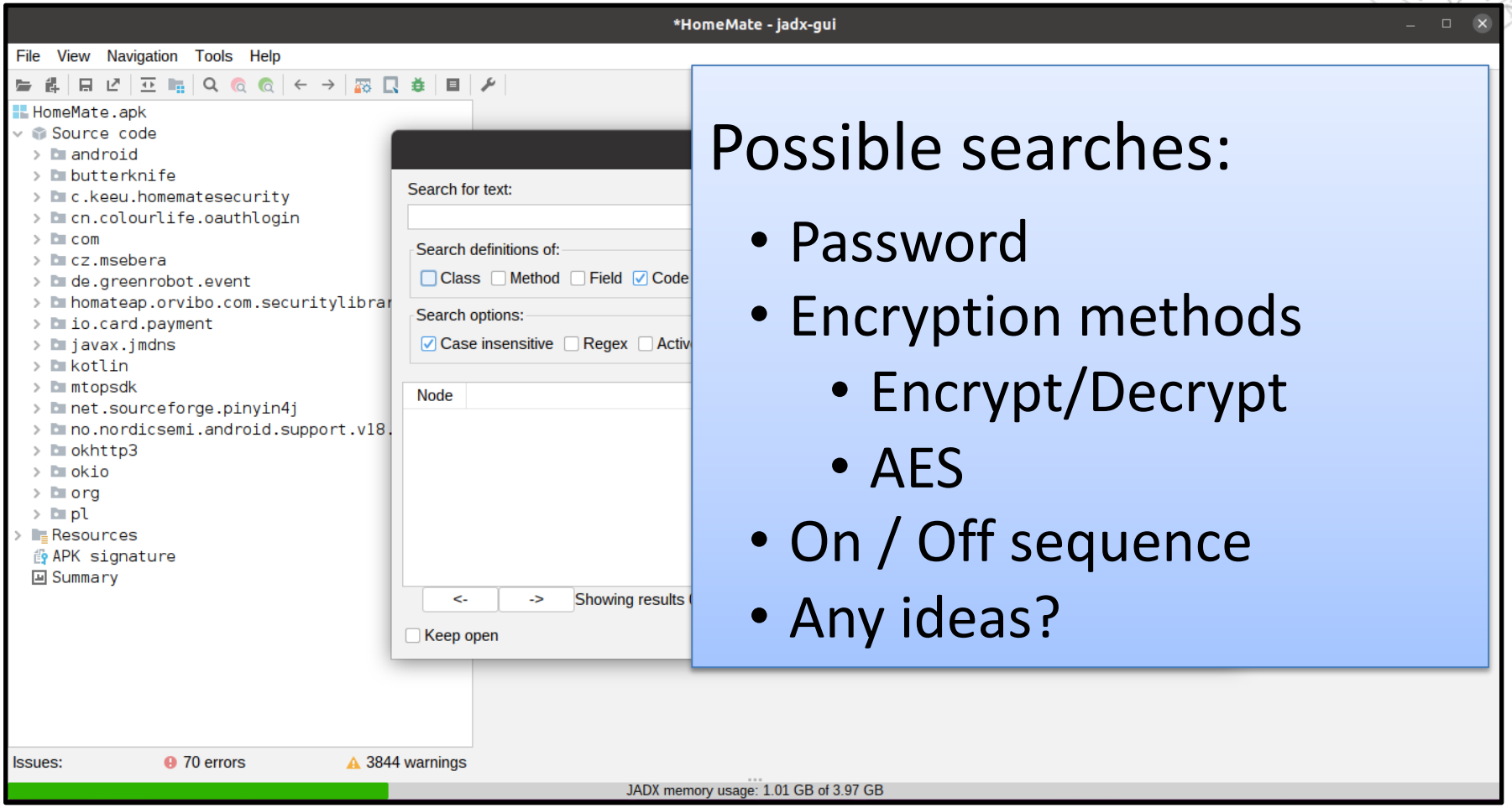
jadx-gui: Search



jadx-gui: Search



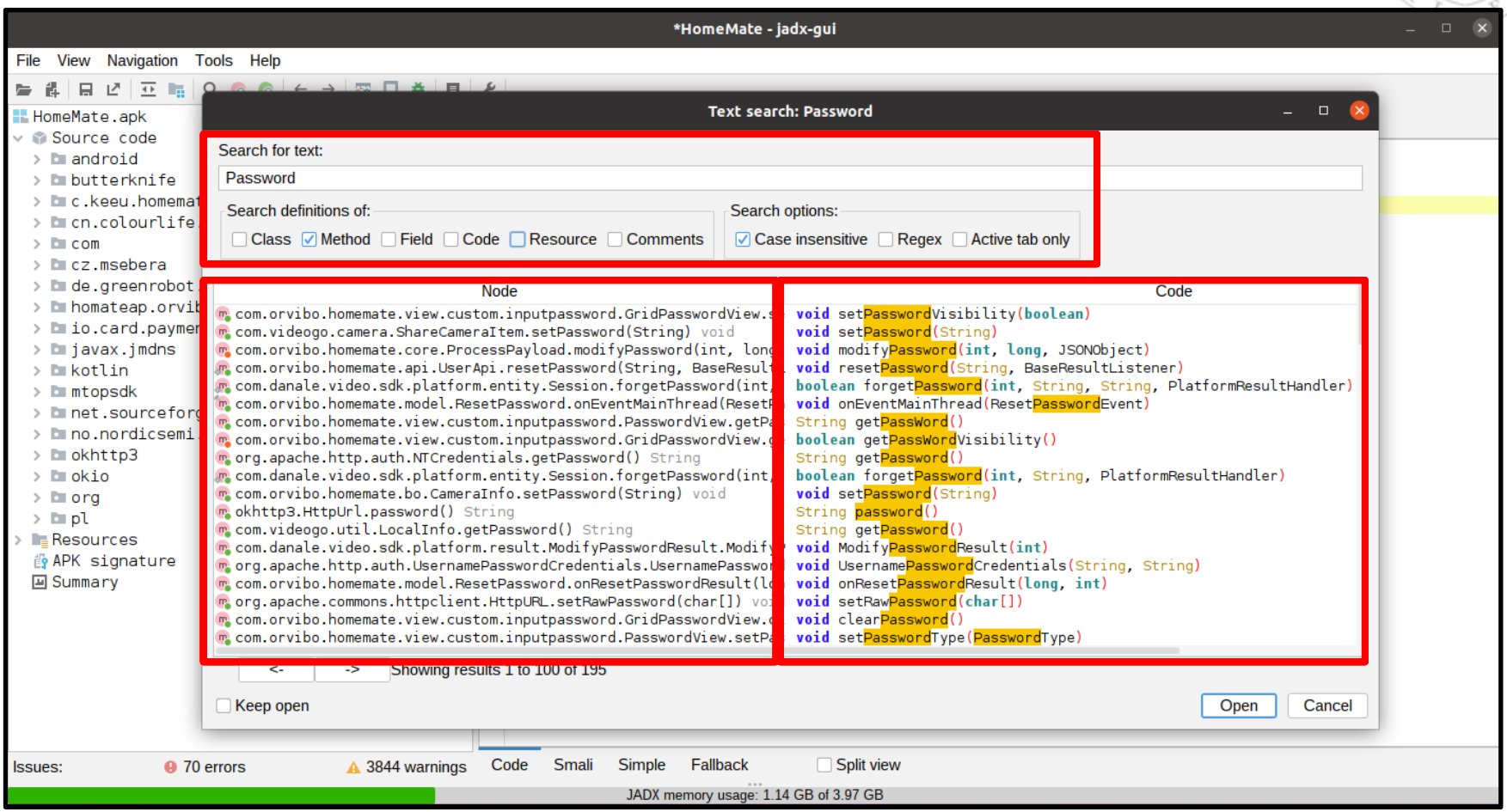
jadx-gui: Search



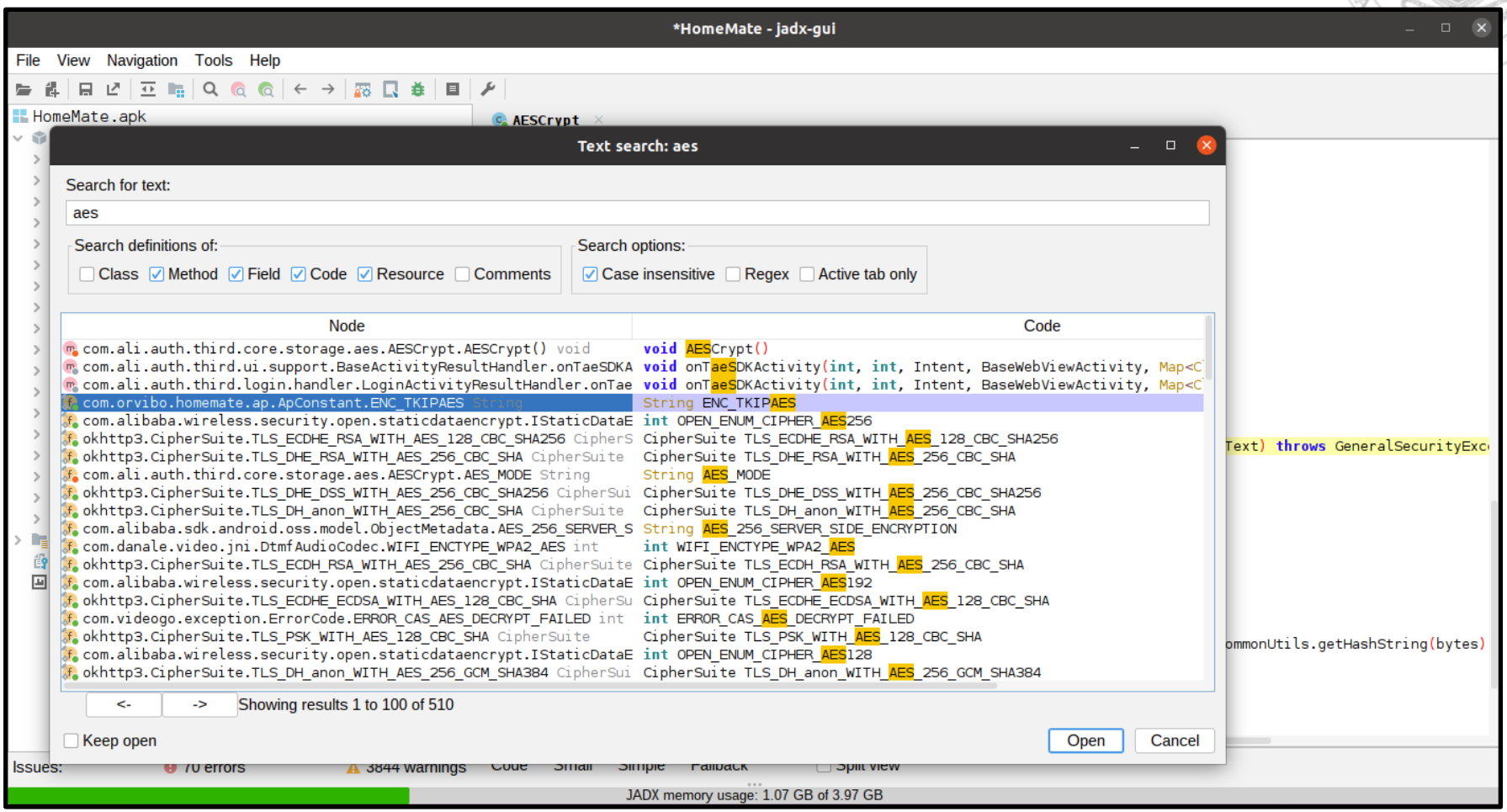
Possible searches:

- Password
- Encryption methods
  - Encrypt/Decrypt
  - AES
- On / Off sequence
- Any ideas?

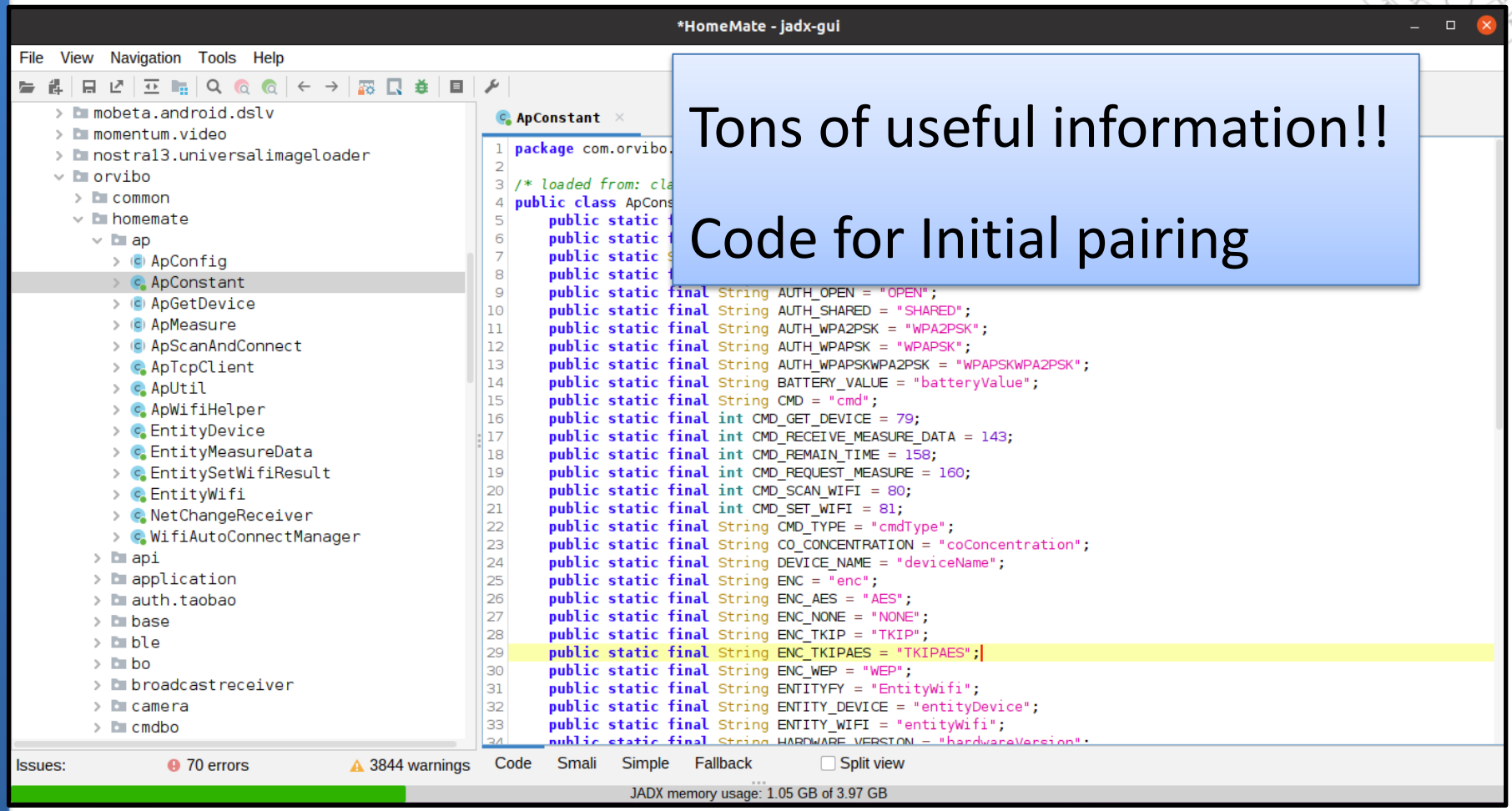
jadx-gui: Password



jadx-gui: AES


*HomeMate - jadx-gui
File View Navigation Tools Help
HomeMate.apk
AESCrypt
Text search: aes
Search for text:
aes
Search definitions of:
Class Method Field Code Resource Comments
Search options:
Case insensitive Regex Active tab only
Node
Code
com.ali.auth.third.core.storage.aes.AESCrypt.AESCrypt() void
void AESCrypt()
com.ali.auth.third.ui.support.BaseActivityResultHandler.onTaeSDKA
void onTaeSDKActivity(int, int, Intent, BaseWebViewActivity, Map<C
com.ali.auth.third.login.handler.LoginActivityResultHandler.onTae
void onTaeSDKActivity(int, int, Intent, BaseWebViewActivity, Map<C
com.orvibo.homemate.ap.ApConstant.ENC_TKIPAES String
String ENC_TKIPAES
com.alibaba.wireless.security.open.staticdataencrypt.IStaticDataE
int OPEN_ENUM_CIPHER_AES256
okhttp3.CipherSuite.TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 CipherS
CipherSuite TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
okhttp3.CipherSuite.TLS_DHE_RSA_WITH_AES_256_CBC_SHA CipherSuite
CipherSuite TLS_DHE_RSA_WITH_AES_256_CBC_SHA
com.ali.auth.third.core.storage.aes.AESCrypt.AES_MODE String
String AES_MODE
okhttp3.CipherSuite.TLS_DHE_DSS_WITH_AES_256_CBC_SHA256 CipherSui
CipherSuite TLS_DHE_DSS_WITH_AES_256_CBC_SHA256
okhttp3.CipherSuite.TLS_DH_anon_WITH_AES_256_CBC_SHA CipherSuite
CipherSuite TLS_DH_anon_WITH_AES_256_CBC_SHA
com.alibaba.sdk.android.oss.model.ObjectMetadata.AES_256_SERVER_S
String AES_256_SERVER_SIDE_ENCRYPTION
com.danale.video.jni.DtmfAudioCodec.WIFI_ENCTYPE_WPA2_AES int
int WIFI_ENCTYPE_WPA2_AES
okhttp3.CipherSuite.TLS_ECDH_RSA_WITH_AES_256_CBC_SHA CipherSuite
CipherSuite TLS_ECDH_RSA_WITH_AES_256_CBC_SHA
com.alibaba.wireless.security.open.staticdataencrypt.IStaticDataE
int OPEN_ENUM_CIPHER_AES192
okhttp3.CipherSuite.TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA CipherSu
CipherSuite TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
com.videogo.exception.ErrorCode.ERROR_CAS_AES_DECRYPT_FAILED int
int ERROR_CAS_AES_DECRYPT_FAILED
okhttp3.CipherSuite.TLS_PSK_WITH_AES_128_CBC_SHA CipherSuite
CipherSuite TLS_PSK_WITH_AES_128_CBC_SHA
com.alibaba.wireless.security.open.staticdataencrypt.IStaticDataE
int OPEN_ENUM_CIPHER_AES128
okhttp3.CipherSuite.TLS_DH_anon_WITH_AES_256_GCM_SHA384 CipherSui
CipherSuite TLS_DH_anon_WITH_AES_256_GCM_SHA384
<- -> Showing results 1 to 100 of 510
Keep open
Open Cancel
Issues:
JADX memory usage: 1.07 GB of 3.97 GB

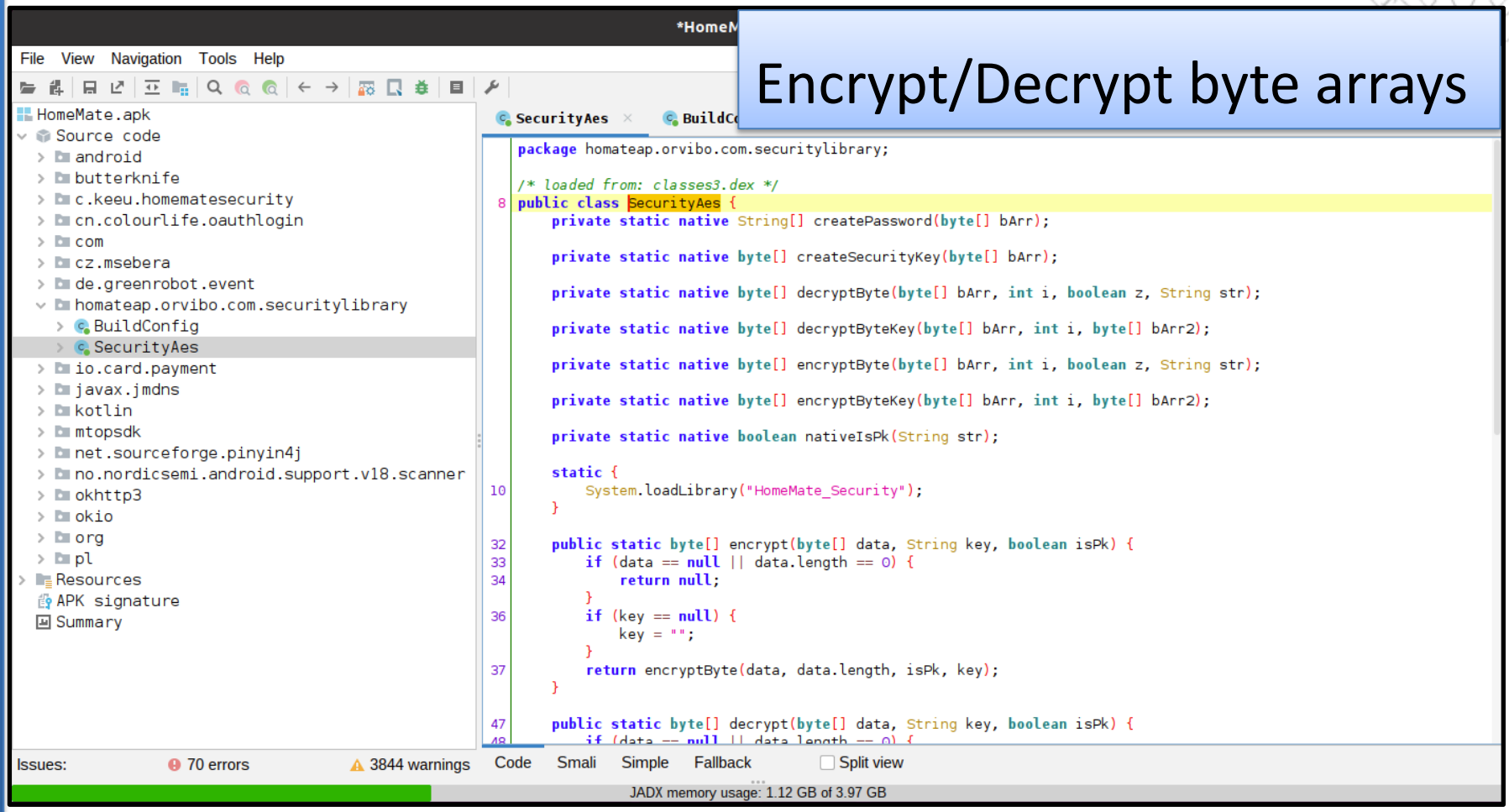
com.orvibo.homemate.ap.ApConstant



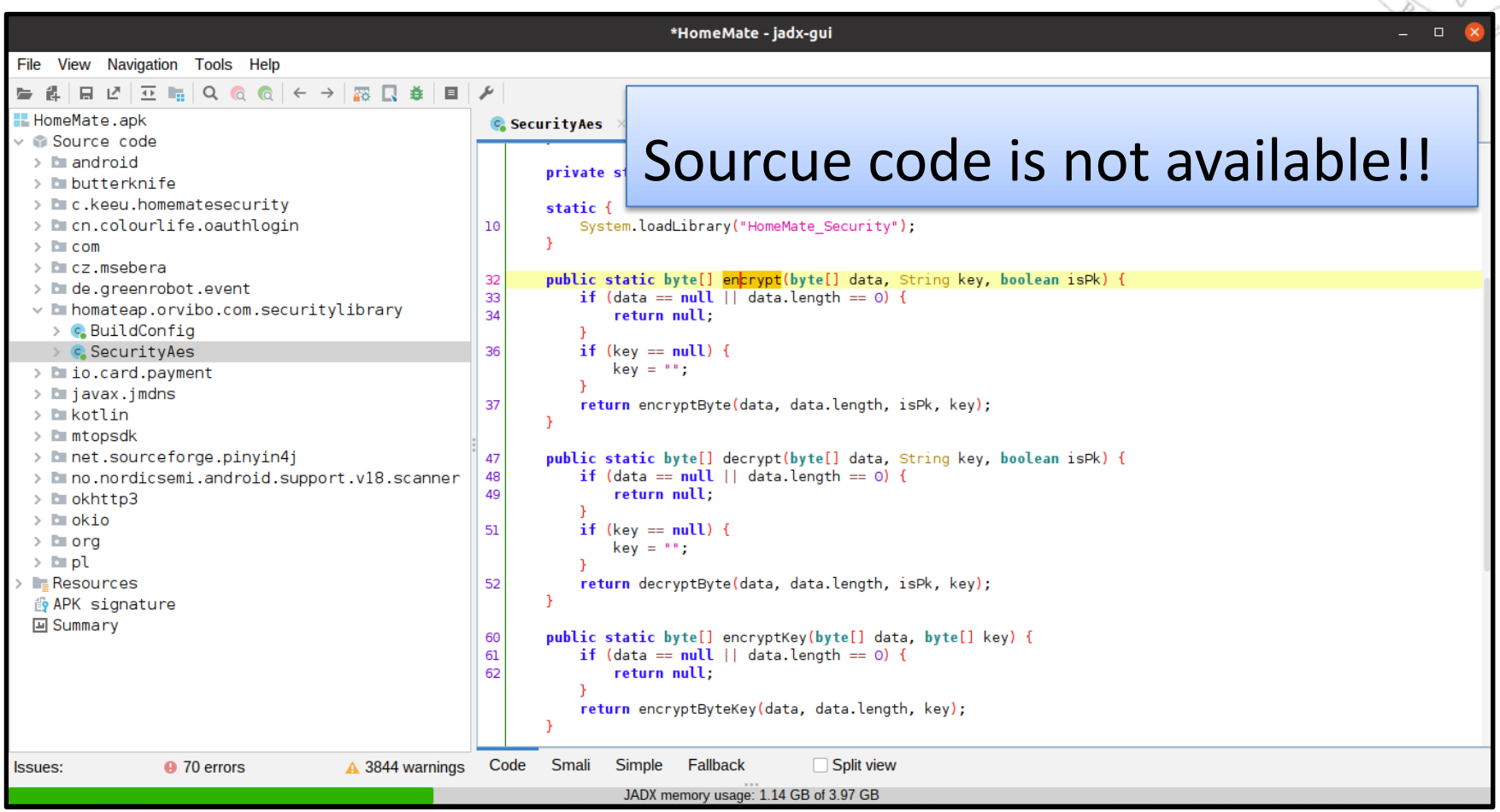
Tons of useful information!!

Code for Initial pairing

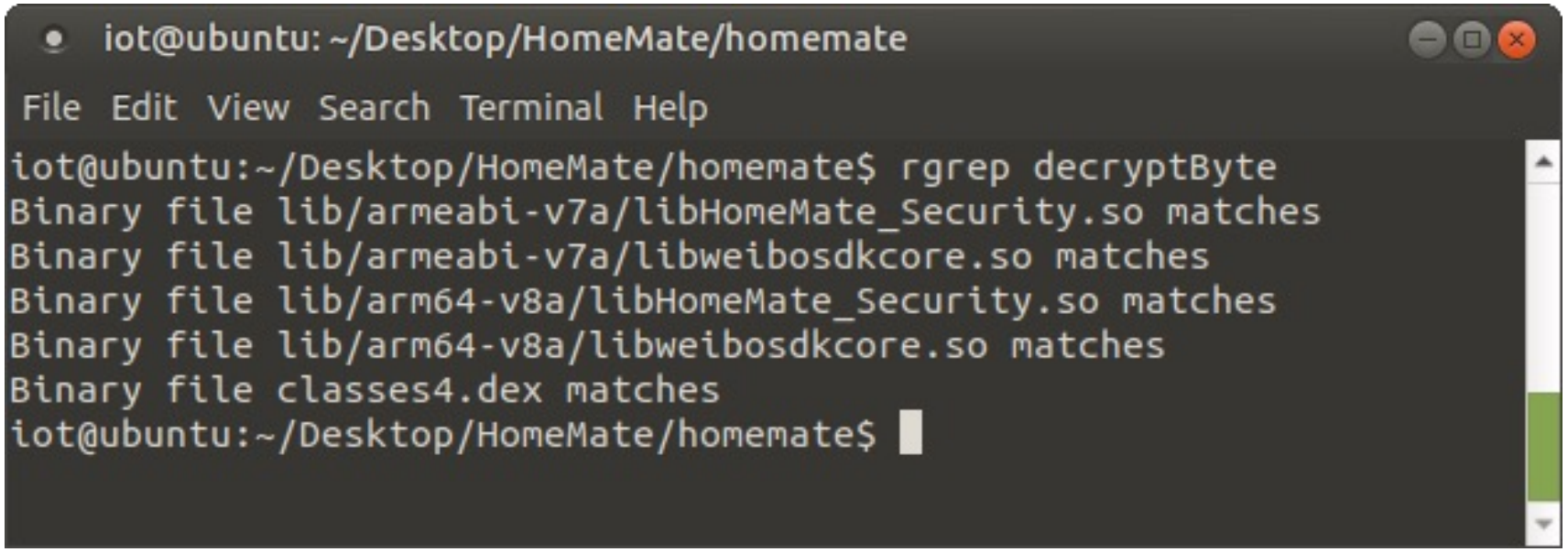
com.orvibo.homemate.ap.ApConstant



com.orvibo.homemate.ap.ApConstant



Looking for decryptByte



We find it in a binary 😭

Binary Ninja

- Interactive disassembler, decompiler, and binary analysis platform for reverse engineers, malware analysts, vulnerability researchers, and software developers that runs on Windows, macOS, Linux.

Disassemble

Disassemble executables and libraries from multiple formats, platforms, and architectures.

Decompile

Decompile code to C or BNIL for any supported architecture - including your own

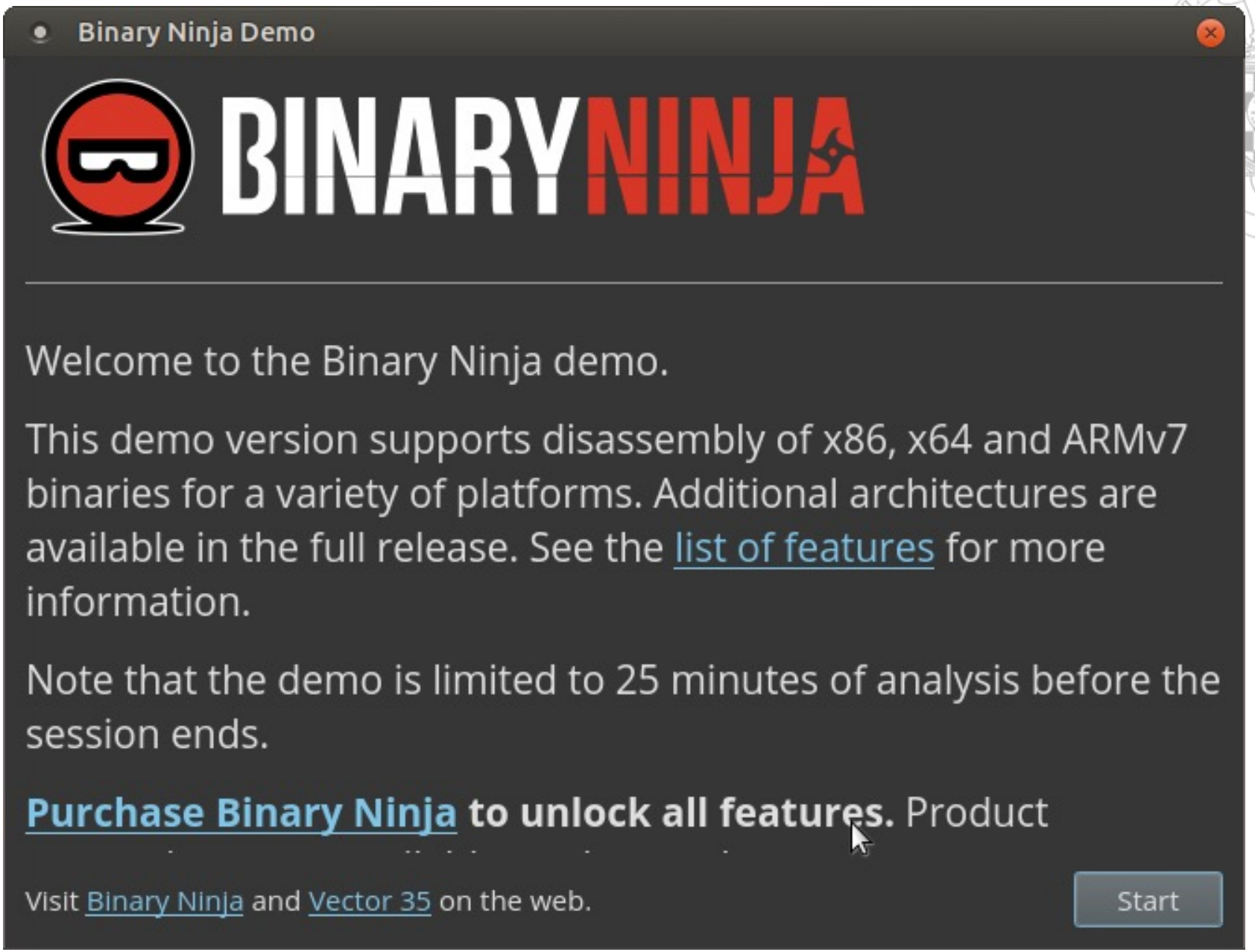
Analyze

Visualize control flow and navigate through cross-references interactively.

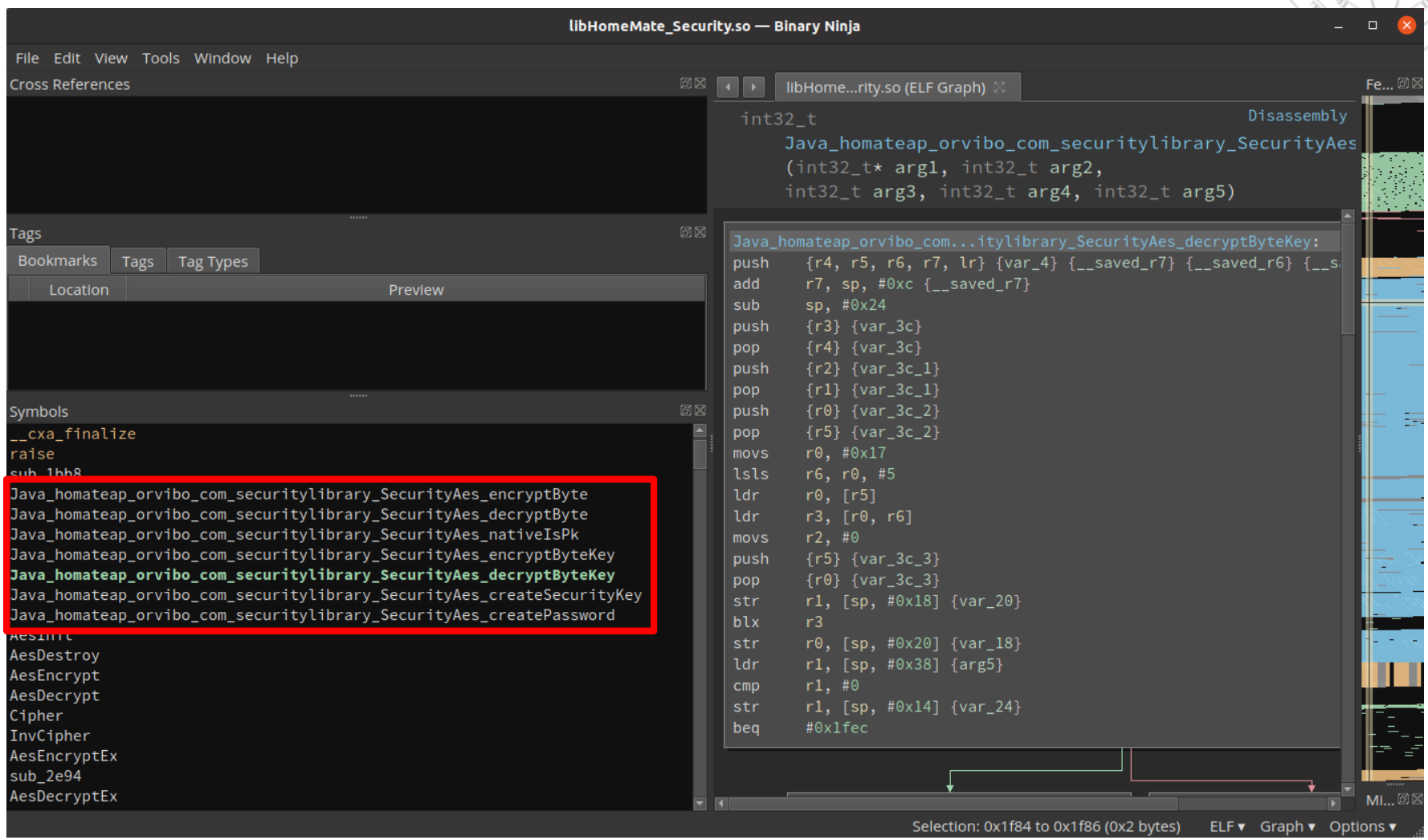
Annotate

Name variables and functions, apply types, create structres, and add comments.

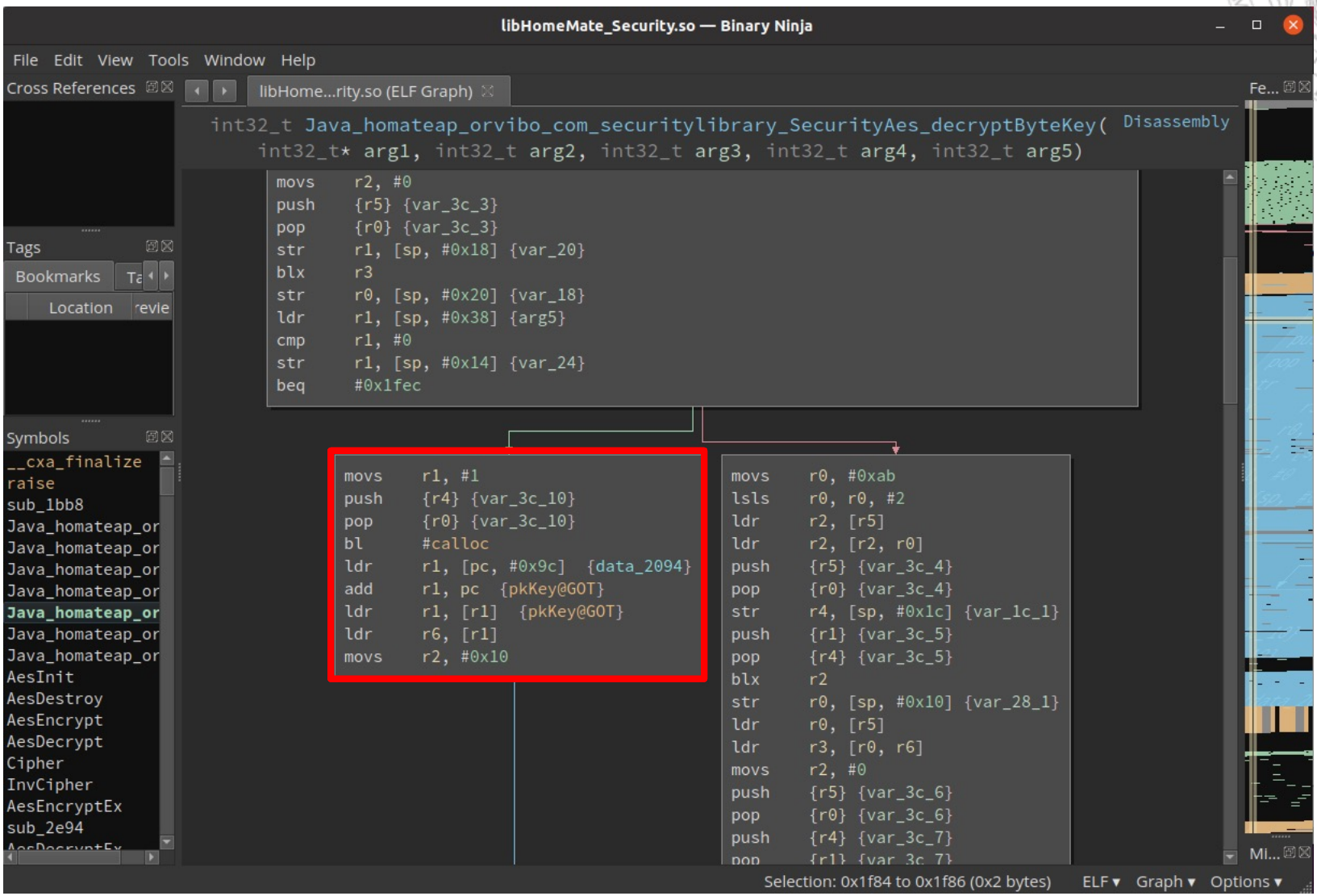
https://binary.ninja/


Binary Ninja Demo
BINARYNINJA
Welcome to the Binary Ninja demo.
This demo version supports disassembly of x86, x64 and ARMv7 binaries for a variety of platforms. Additional architectures are available in the full release. See the list of features for more information.
Note that the demo is limited to 25 minutes of analysis before the session ends.
Purchase Binary Ninja to unlock all features. Product
Visit Binary Ninja and Vector 35 on the web.
Start

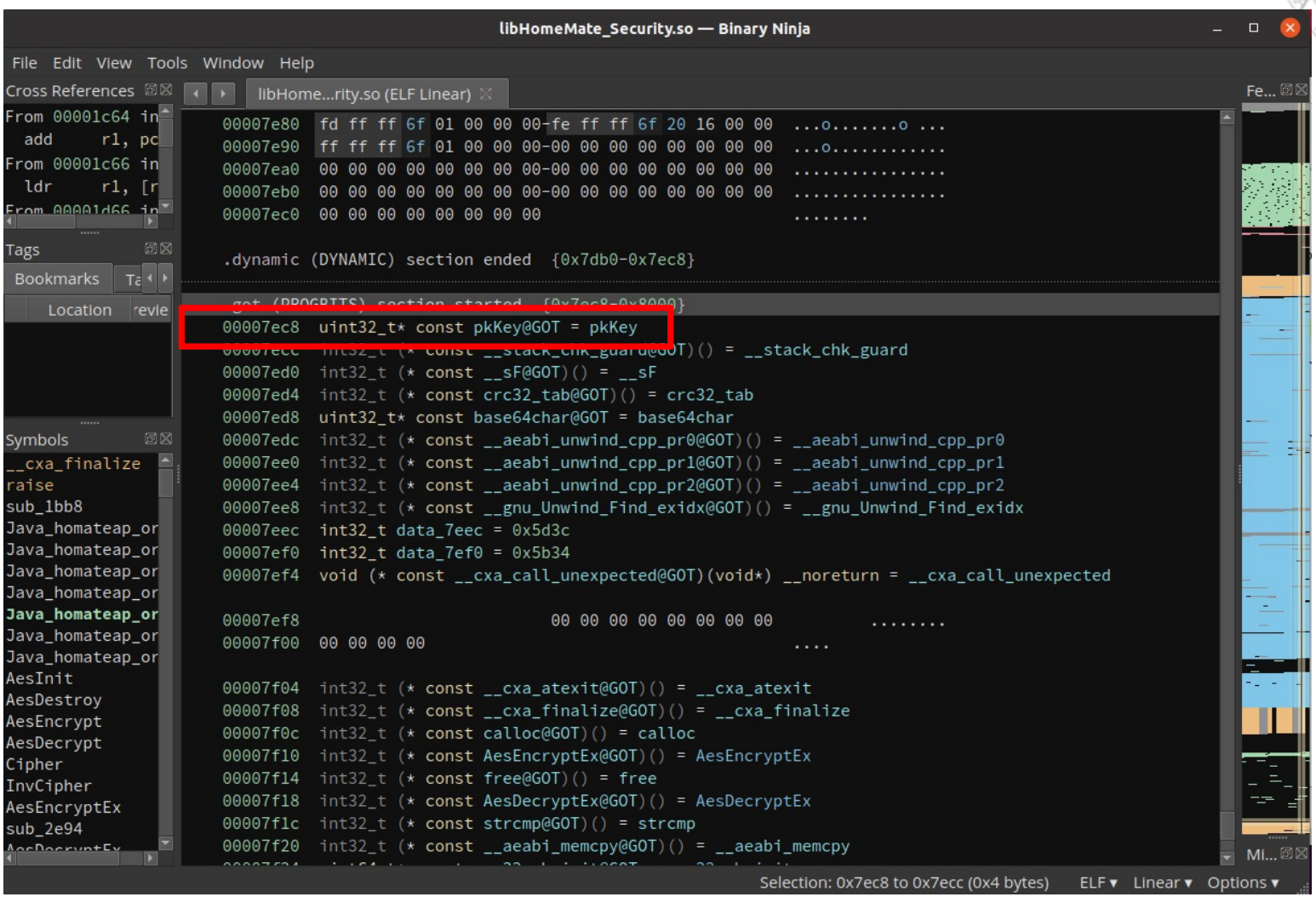
Binary Ninja: dedryptByte



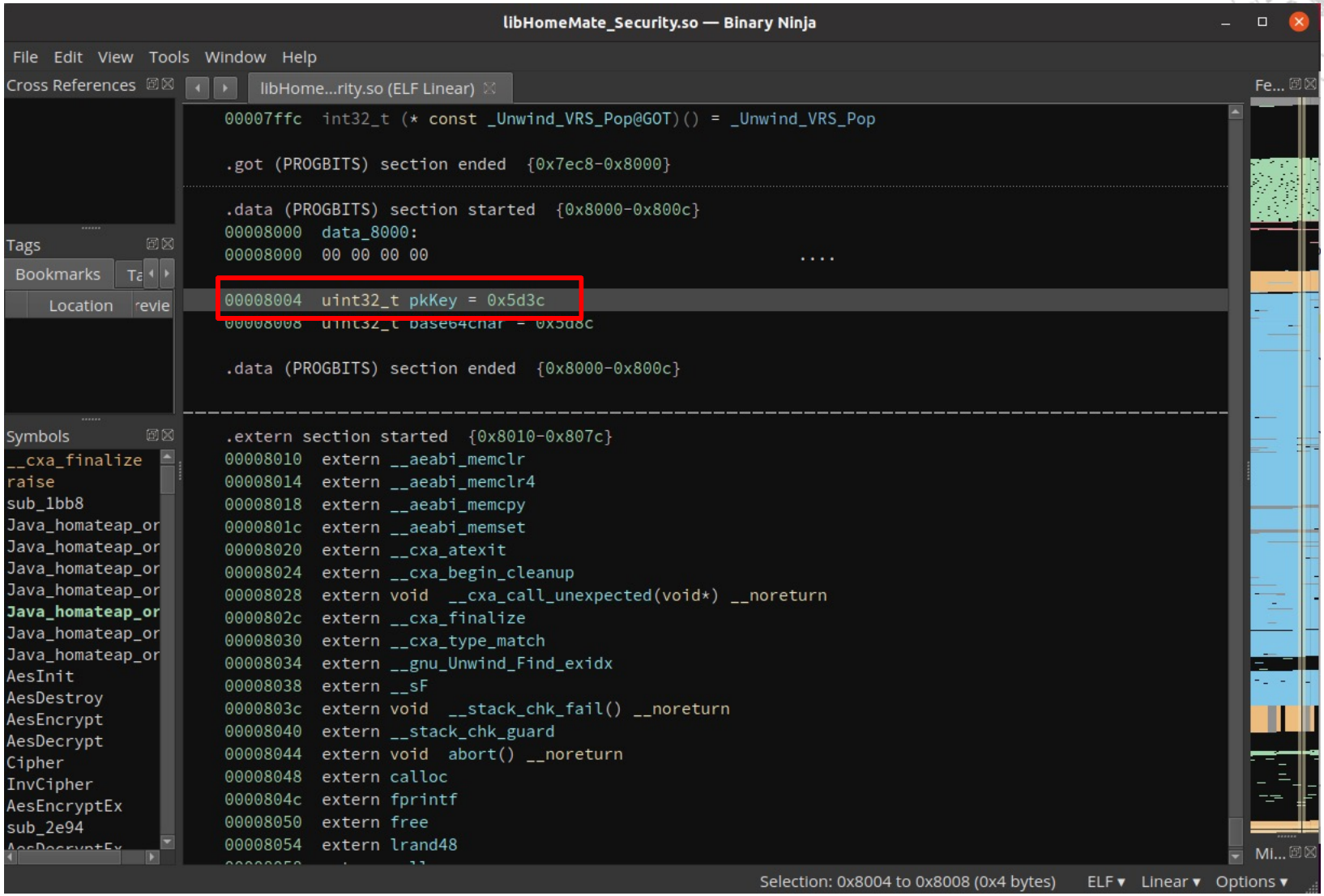
Binary Ninja: dedryptByte



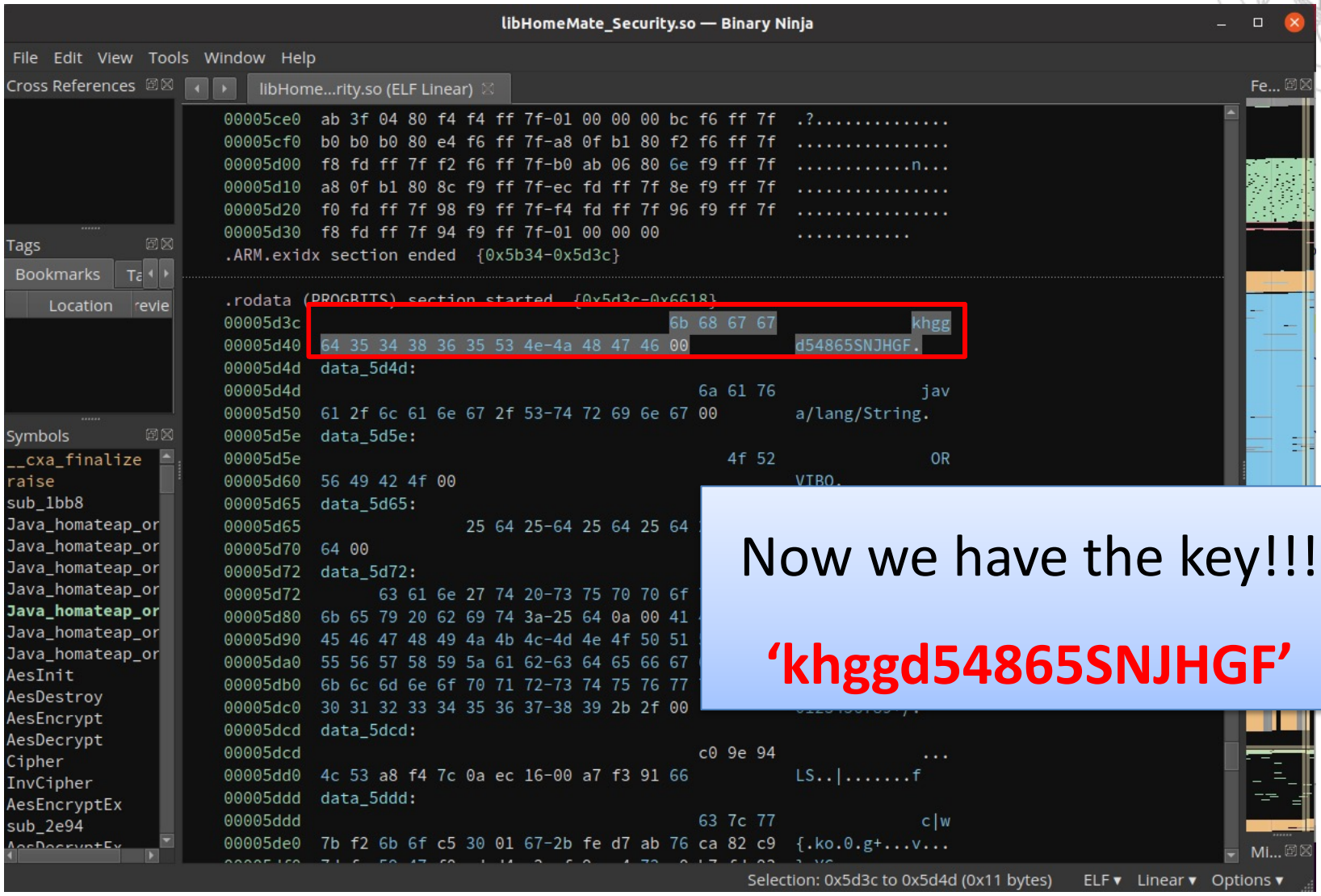
Binary Ninja: pkKey



Binary Ninja: pkKey



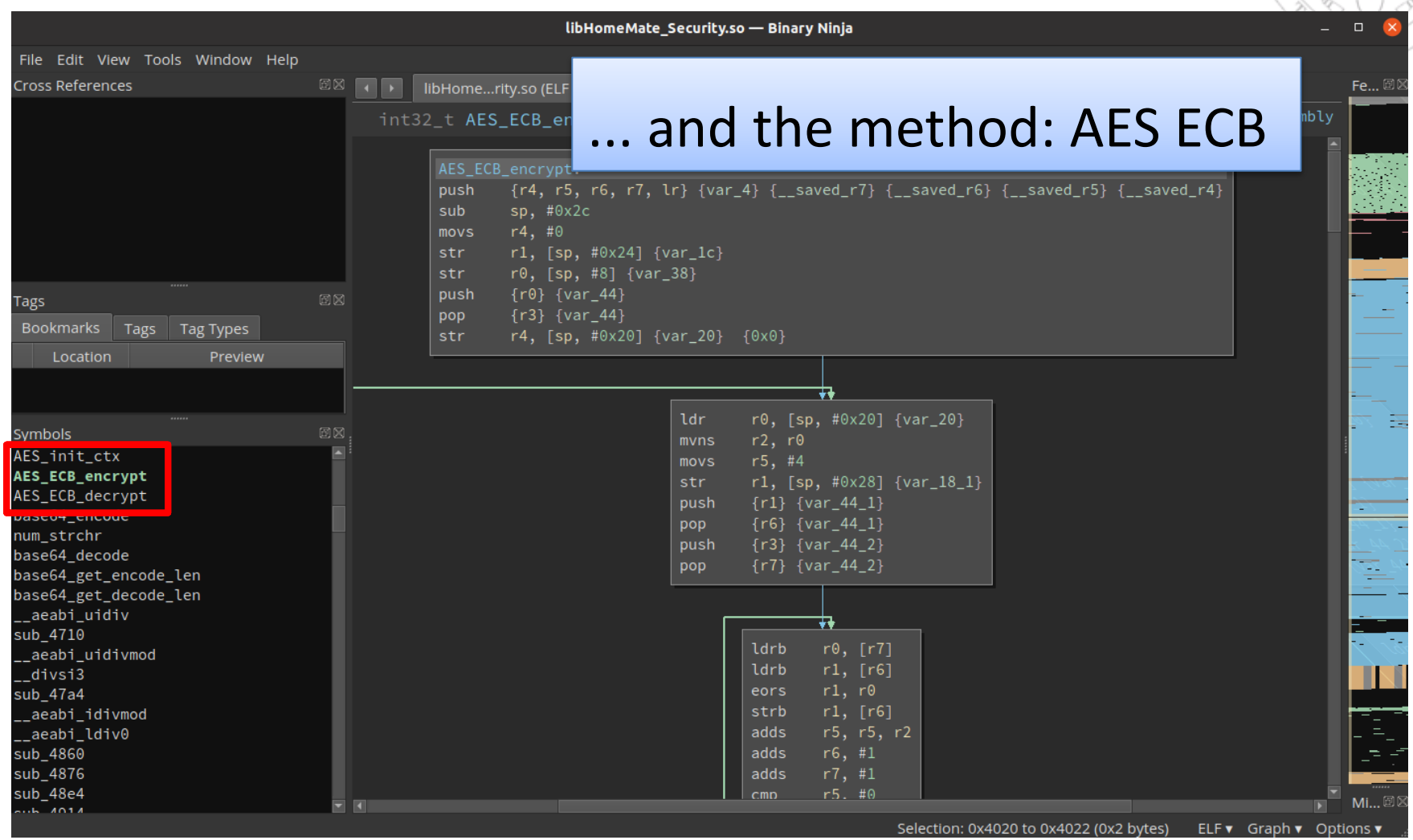
Binary Ninja: pkKey



Now we have the key!!!

‘khggd54865SNJHGF’

Binary Ninja: AES-ECB



... and the method: AES ECB

Next steps

- Decrypt and understand the messages
  - Get the payload from the On/Off sequence
  - Use python to decrypt its content
  - Try to understand the logic
  - Take control of the device