DEXCALIBUR

AUTOMATE YOUR ANDROID APP REVERSE

Or hooking for dummies

https://github.com/FrenchYeti/dexcalibur.git

WHO AM I ?

GEORGES-B. MICHEL

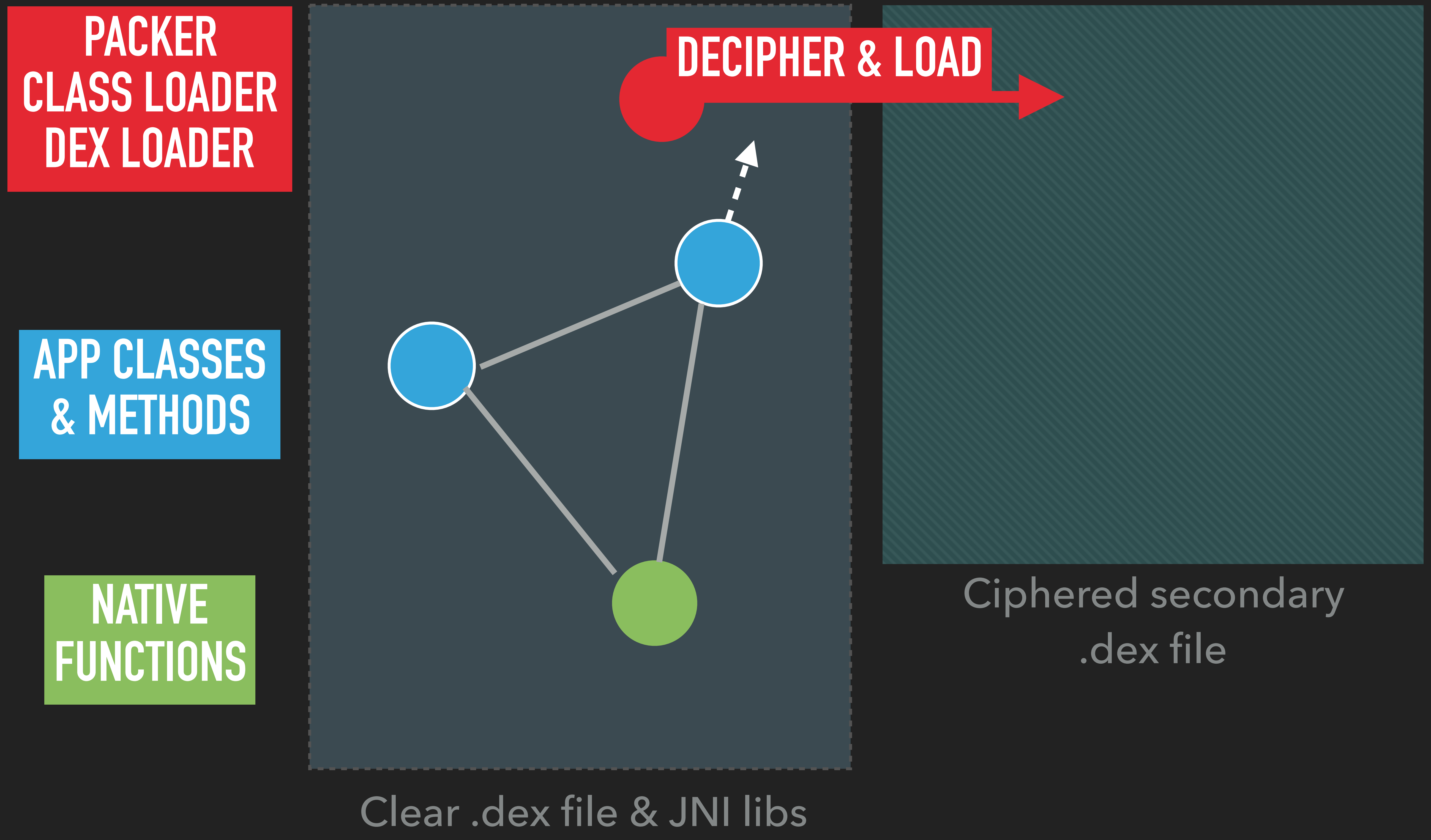
- @FrenchYeti
- yeti@0xff.ninja
- Software Security Evaluator at Thales

Aka @FrenchYeti

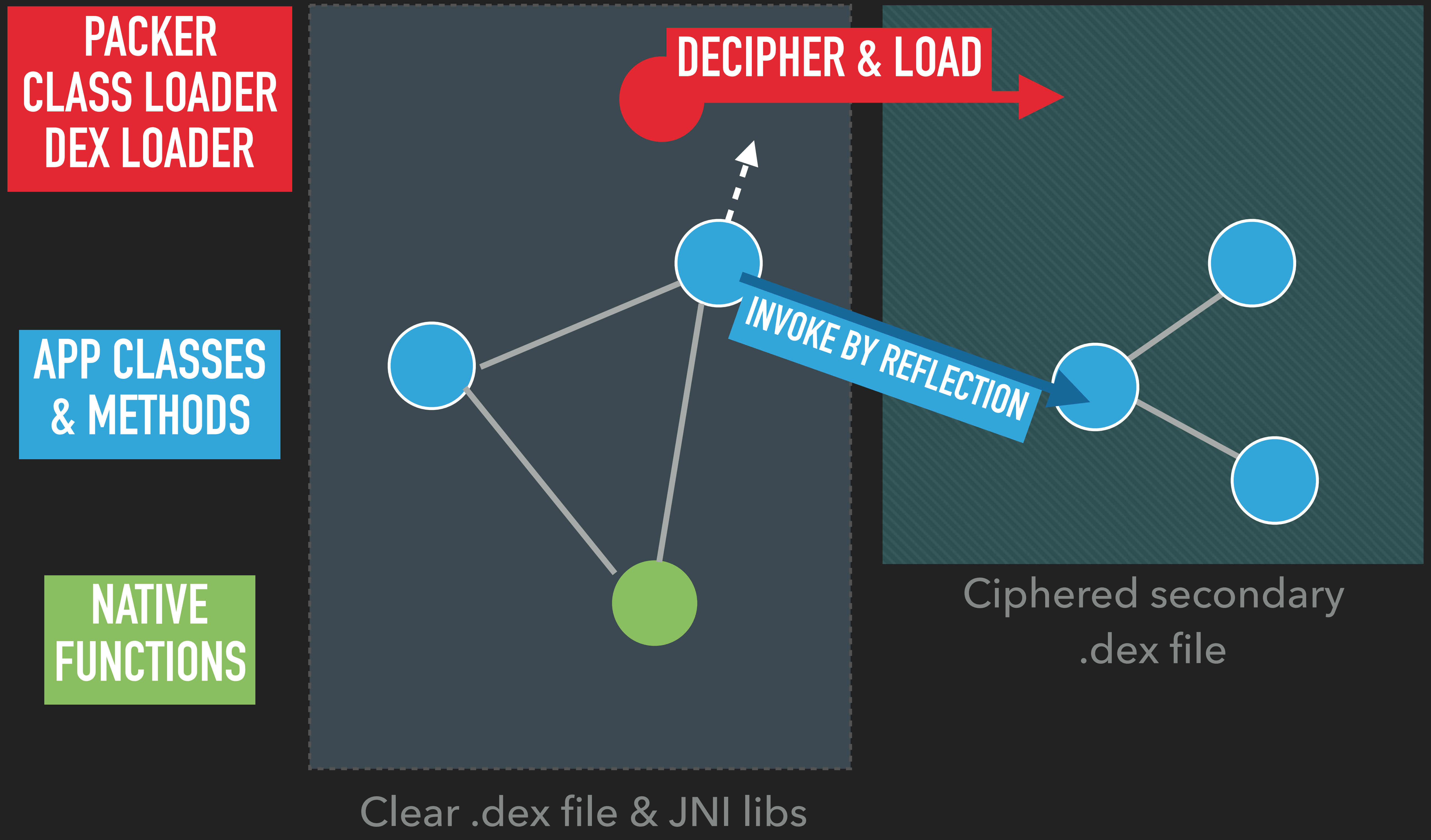
- Day : Reverse engineering (Android + TEE) apps
  - HCE Payment applications, Trusted Applications, ARM binaries
- Night : Develop reverse / pentest / appsec tools
  - Frida addict ❤️

EXAMPLE OF AN OBFUSCATED ANDROID APPLICATION

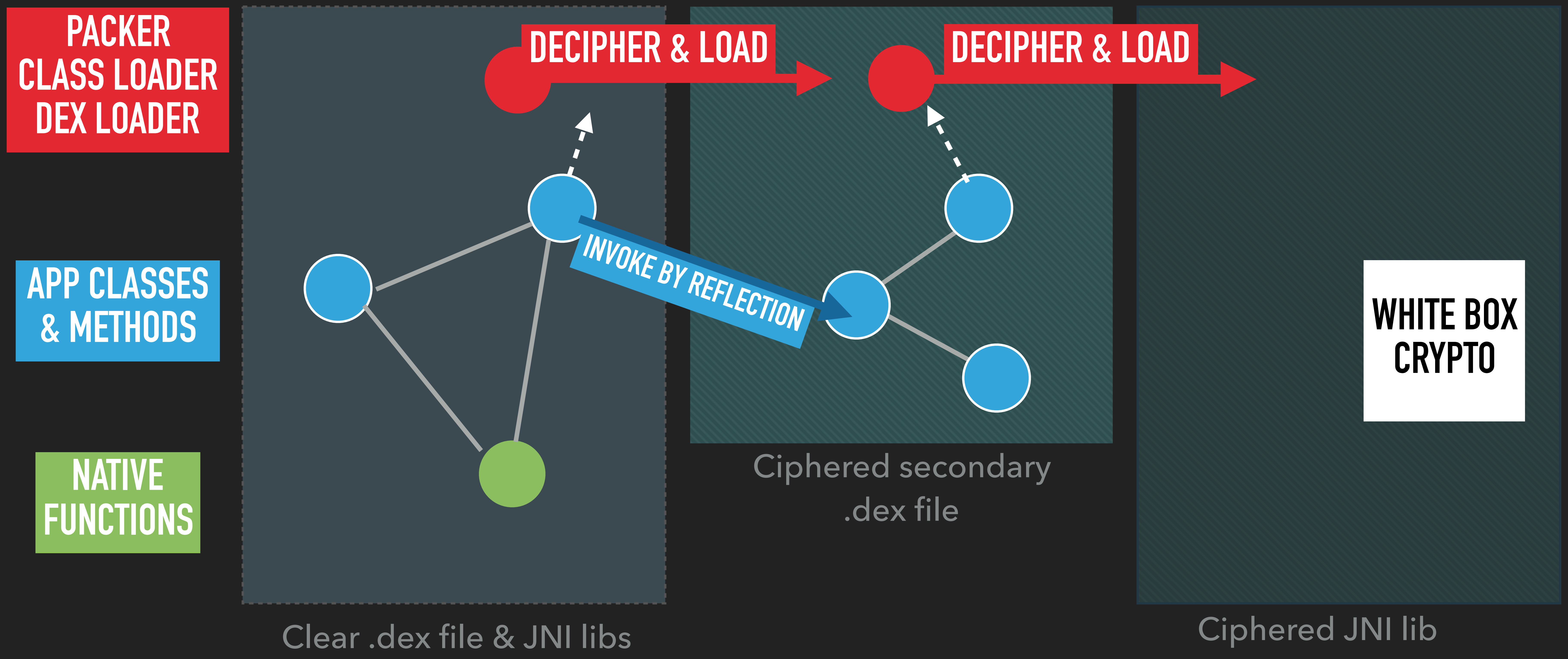
LET'S IMAGINE AN OBFUSCATED MULTI-DEX APPLICATION


PACKER CLASS LOADER DEX LOADER
APP CLASSES & METHODS
NATIVE FUNCTIONS
DECIPHER & LOAD
Ciphered secondary .dex file
Clear .dex file & JNI libs

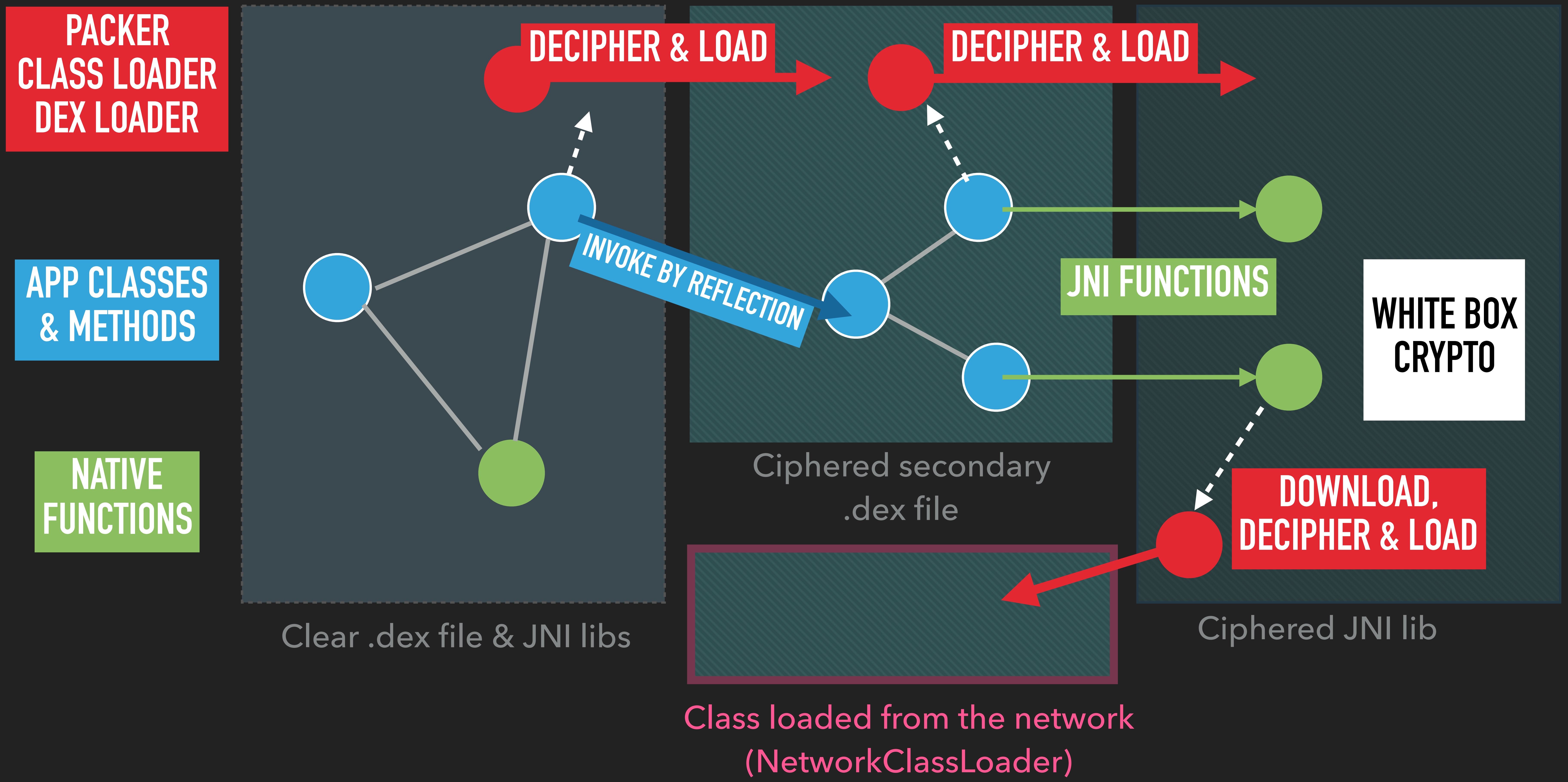
LET'S IMAGINE AN OBFUSCATED MULTI-DEX APPLICATION


PACKER CLASS LOADER DEX LOADER
APP CLASSES & METHODS
NATIVE FUNCTIONS
DECIPHER & LOAD
INVOKE BY REFLECTION
Ciphered secondary .dex file
Clear .dex file & JNI libs

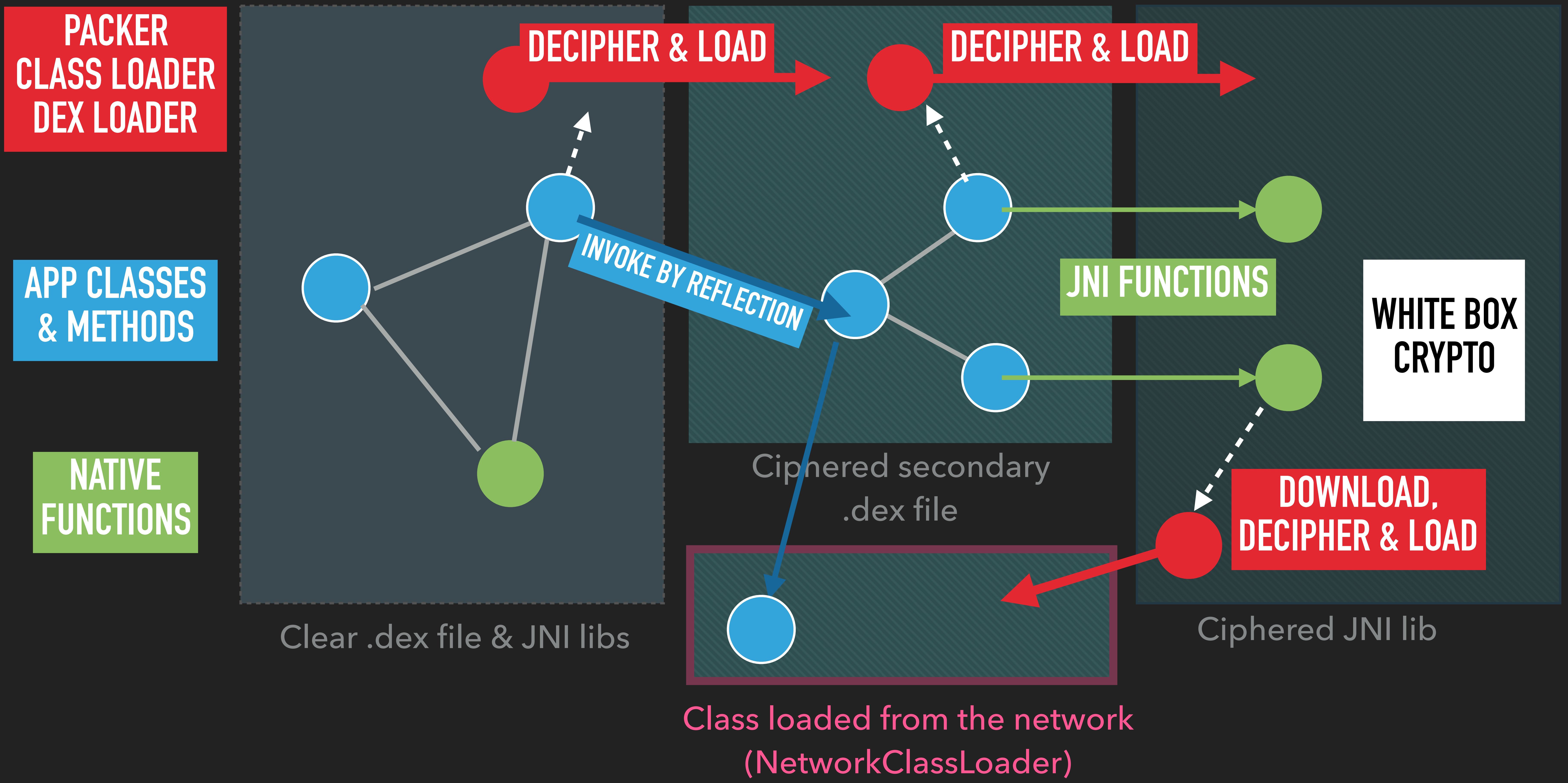
LET'S IMAGINE AN OBFUSCATED MULTI-DEX APPLICATION



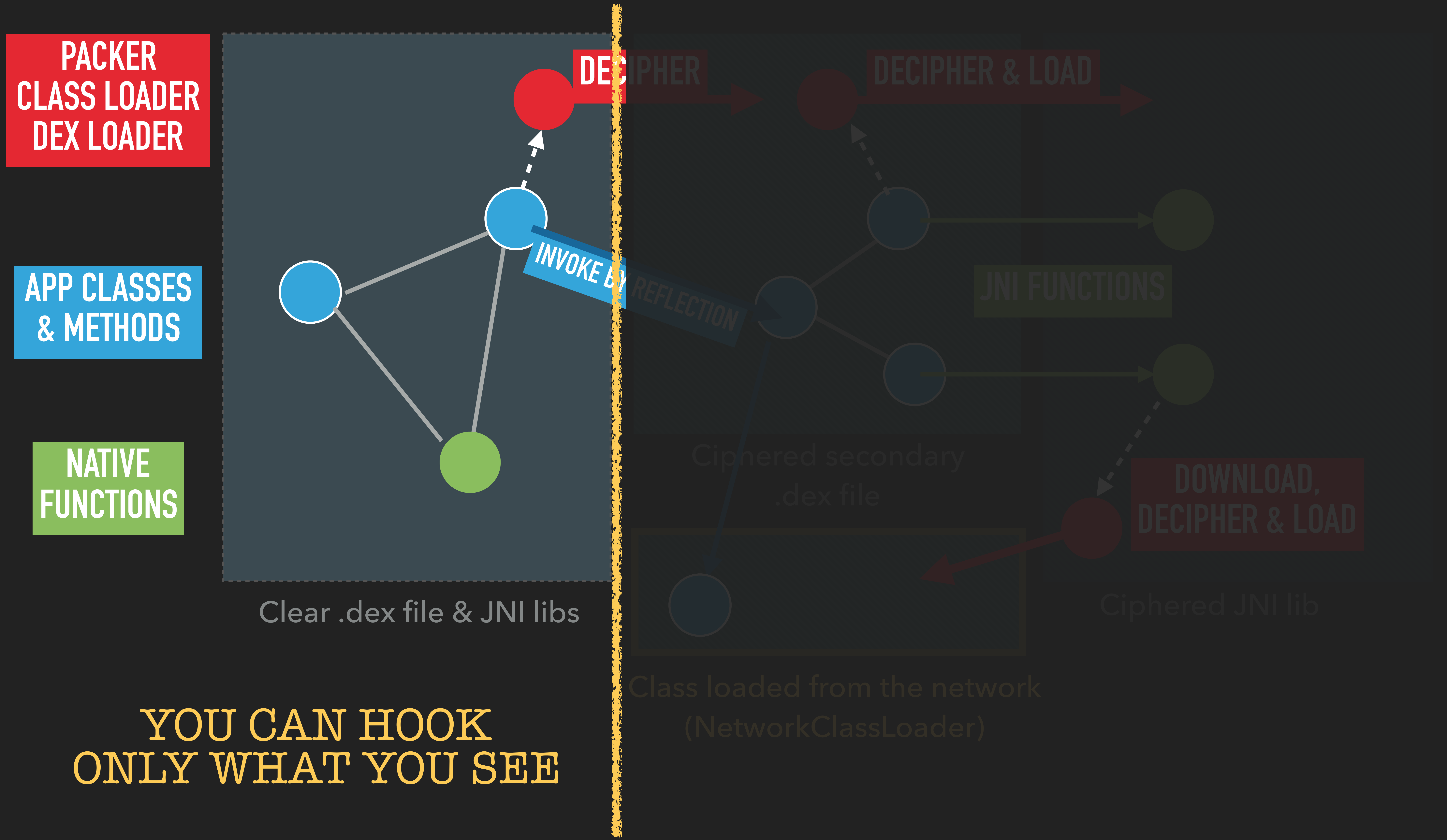
LET'S IMAGINE AN OBFUSCATED MULTI-DEX APPLICATION


PACKER CLASS LOADER DEX LOADER
APP CLASSES & METHODS
NATIVE FUNCTIONS
DECIPHER & LOAD
DECIPHER & LOAD
INVOKE BY REFLECTION
JNI FUNCTIONS
WHITE BOX CRYPTO
DOWNLOAD, DECIPHER & LOAD
Clear .dex file & JNI libs
Ciphered secondary .dex file
Ciphered JNI lib
Class loaded from the network (NetworkClassLoader)

LET'S IMAGINE AN OBFUSCATED MULTI-DEX APPLICATION

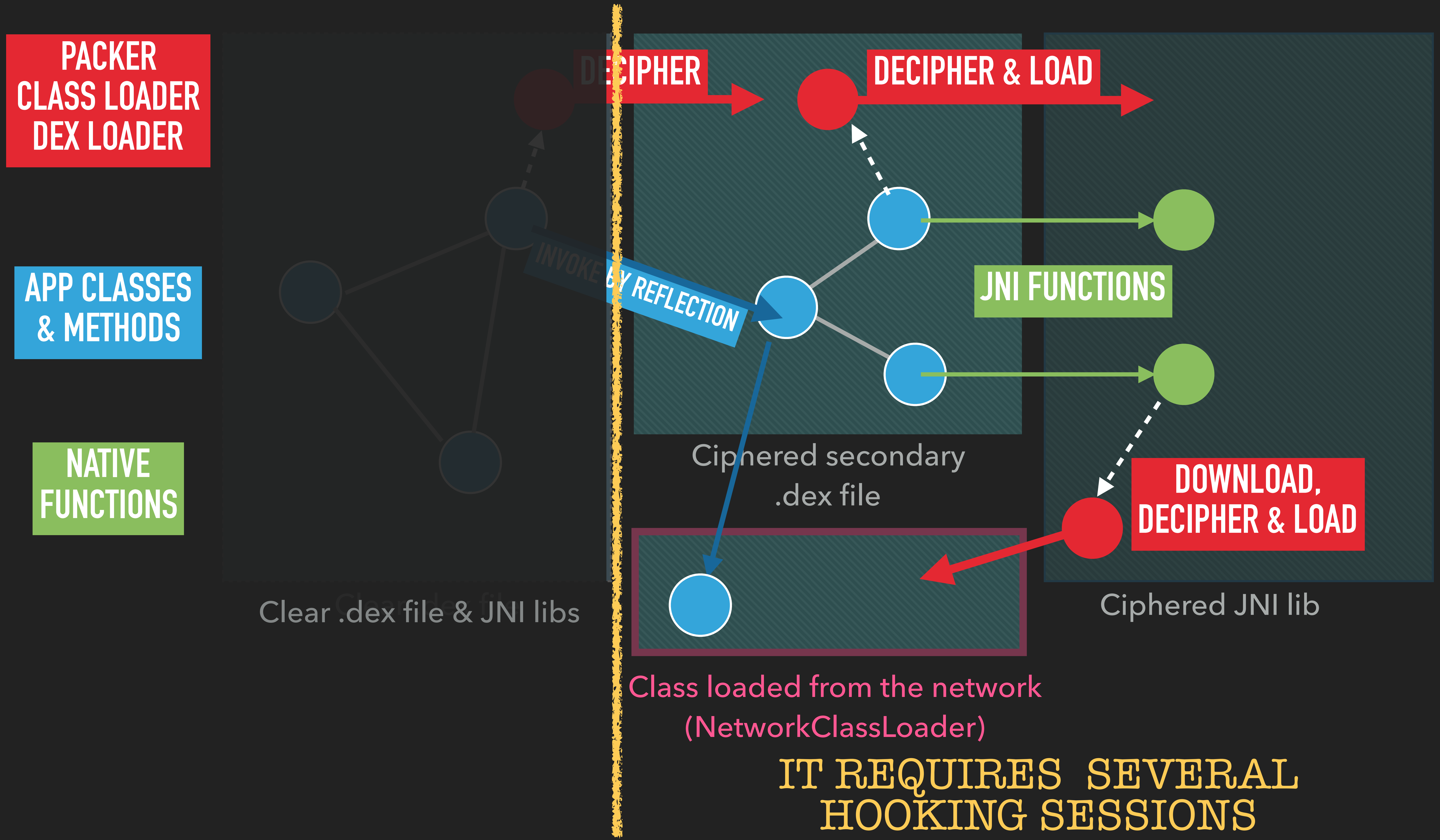

PACKER CLASS LOADER DEX LOADER
APP CLASSES & METHODS
NATIVE FUNCTIONS
DECIPHER & LOAD
DECIPHER & LOAD
INVOKE BY REFLECTION
JNI FUNCTIONS
WHITE BOX CRYPTO
DOWNLOAD, DECIPHER & LOAD
Clear .dex file & JNI libs
Ciphered secondary .dex file
Ciphered JNI lib
Class loaded from the network (NetworkClassLoader)

WHAT CAN I HOOK ?



YOU CAN HOOK
ONLY WHAT YOU SEE

WHAT IS INTERESTING TO HOOK ?



IT REQUIRES SEVERAL HOOKING SESSIONS

MOTIVATION

MOTIVATION

- Deobfuscate ➡ waste of time

MOTIVATION

- Deobfuscate ➡ waste of time
- Manage hooks ➡ not so easy

MOTIVATION

- Deobfuscate ➡ waste of time
- Manage hooks ➡ not so easy
- Manual tasks ➡ can be automated (start App, …)

MOTIVATION

- Deobfuscate ➡ waste of time
- Manage hooks ➡ not so easy
- Manual tasks ➡ can be automated (start App, …)
- Several devices ➡ hooked simultaneously

MOTIVATION

- Deobfuscate → waste of time
- Manage hooks → not so easy
- Manual tasks → can be automated (start App, …)
- Several devices → hooked simultaneously
- Application size → explore bytecode/libs is boring

CHRISTMAS WISH LIST 1/2 :

- Show functions invoked dynamically as « xrefs »
- Discover automatically classes & bytecode loaded dynamically (DexFile ..)
- Generate hook with a single click on the function
- Debug a single hook while others are active
- Enable/disable hook without lose or pollute the source code

CHRISTMAS WISH LIST 2/2 :

- Multi-user : share the same instrumentation with my friends
- Instrumente several devices and merge hook logs (Workflow / IoT)
- Be able to run with rooted & non-rooted devices
- Offer user-friendly GUI and API,
- **Free & open-source ! (license APACHE 2)**

WHAT IS DEXCALIBUR ?

NOT JUST A TOOLBOX

DEX DISASSEMBLER *Baksmali*

NOT JUST A TOOLBOX

DEX DISASSEMBLER *Baksmali*

FILE IDENTIFIERS & PARSERS

NOT JUST A TOOLBOX

- DEX DISASSEMBLER *Baksmali*
- FILE IDENTIFIERS & PARSERS
- STATIC BYTECODE ANALYZER
- DYNAMIC BYTECODE ANALYZER

NOT JUST A TOOLBOX

DEX DISASSEMBLER *Baksmali*

FILE IDENTIFIERS & PARSERS

STATIC BYTECODE ANALYZER

DYNAMIC BYTECODE ANALYZER

INSTRUMENTATION TOOL FRIDA

NOT JUST A TOOLBOX

DEX DISASSEMBLER *Baksmali*

FILE IDENTIFIERS & PARSERS

STATIC BYTECODE ANALYZER

DYNAMIC BYTECODE ANALYZER

INSTRUMENTATION TOOL FRIDA

MODULAR HEURISTIC & SEARCH ENGINE

WHAT IS DEXCALIBUR ?

NOT JUST A TOOLBOX

DEX DISASSEMBLER *Baksmali*

FILE IDENTIFIERS & PARSERS

STATIC BYTECODE ANALYZER

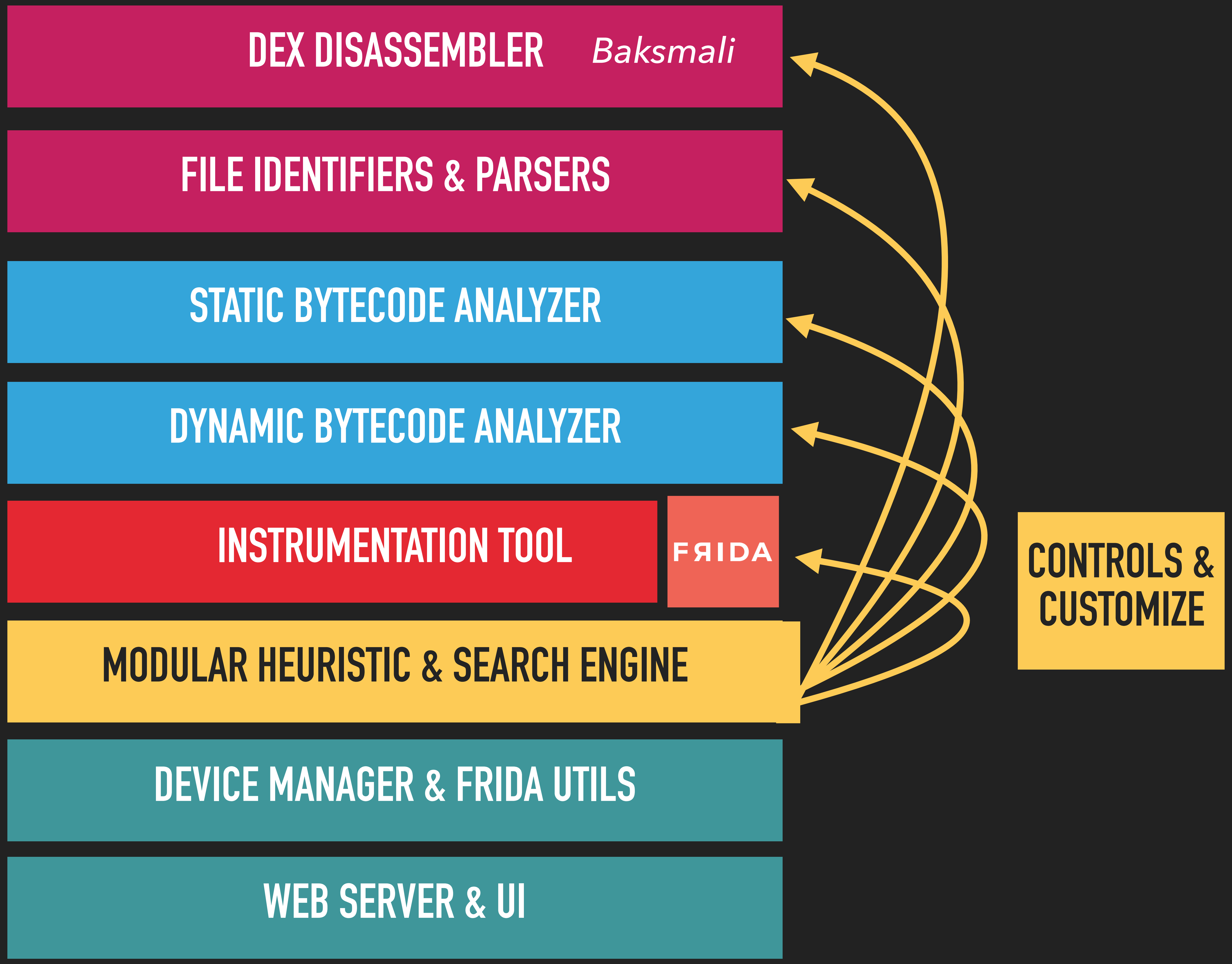
DYNAMIC BYTECODE ANALYZER

INSTRUMENTATION TOOL FRIDA

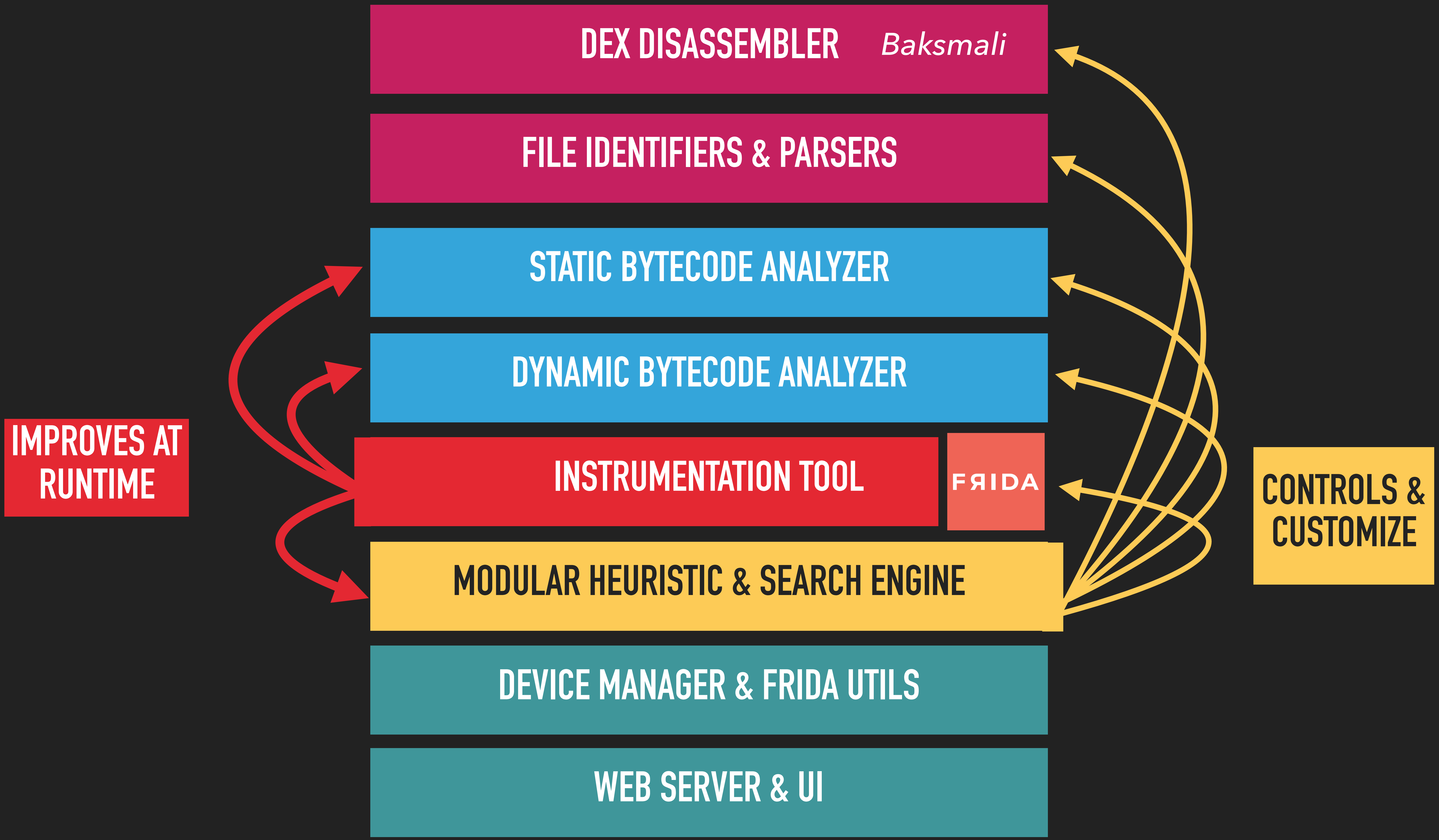
MODULAR HEURISTIC & SEARCH ENGINE

DEVICE MANAGER & FRIDA UTILS

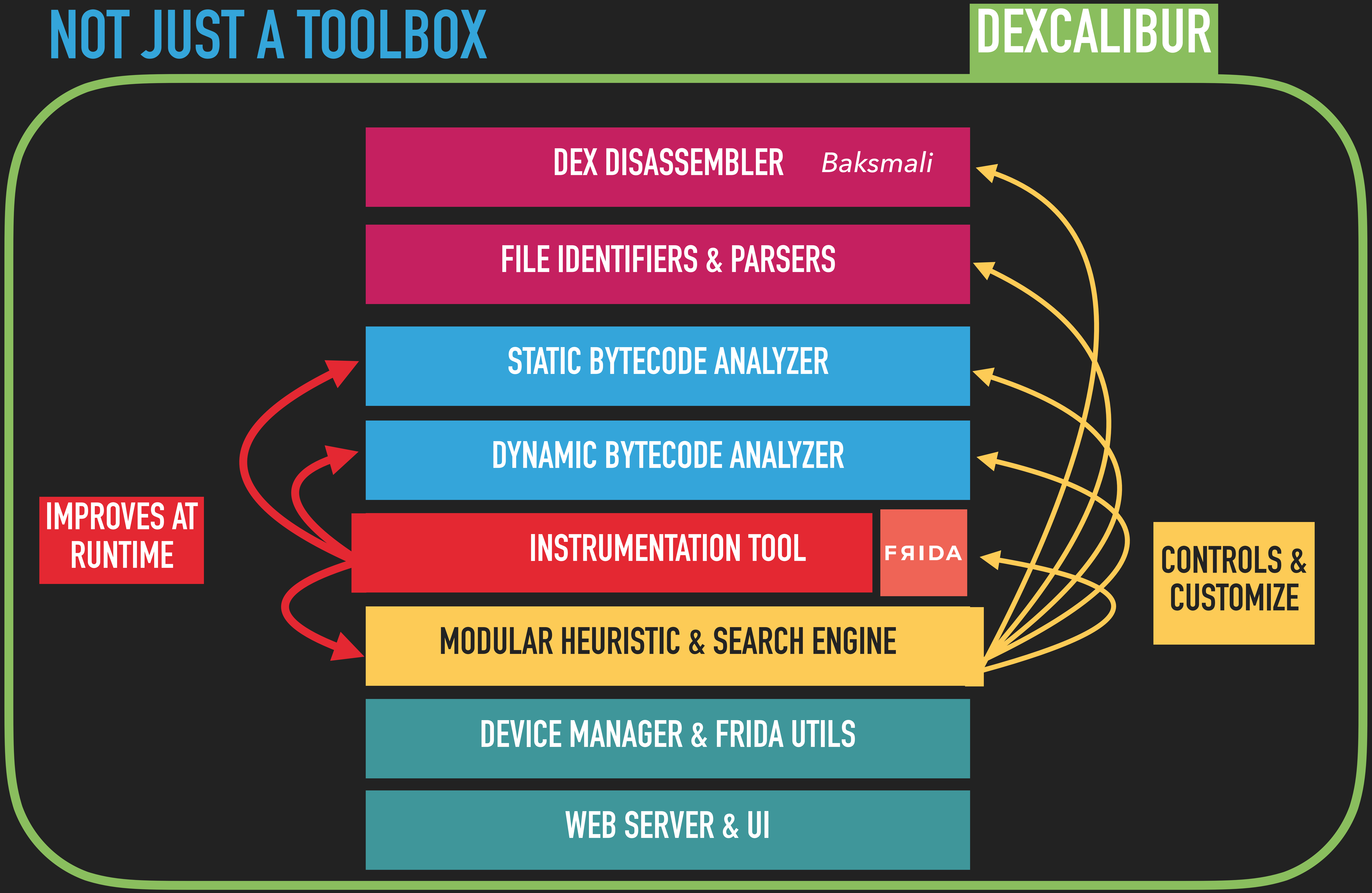
NOT JUST A TOOLBOX


DEX DISASSEMBLER
Baksmali
FILE IDENTIFIERS & PARSERS
STATIC BYTECODE ANALYZER
DYNAMIC BYTECODE ANALYZER
INSTRUMENTATION TOOL
FRIDA
MODULAR HEURISTIC & SEARCH ENGINE
DEVICE MANAGER & FRIDA UTILS
WEB SERVER & UI
CONTROLS & CUSTOMIZE

NOT JUST A TOOLBOX



NOT JUST A TOOLBOX



POWERED BY … NICE TOOLS :-)

FRIDA

APKTOOL + BAKSMALI

Today

NATIVE HOOK CANNOT BE GENERATED
NO BYTECODE SYMBOLIC EXEC

Functions contained into JNI/native libs can be hooked, but decompilers/analyzers dont support it. So, native hook cannot be generated.

POWERED BY … NICE TOOLS :-)

AND MORE !

SMALI VM

Z3 SOLVER

FRIDA

APKTOOL + BAKSMALI

ANDROID SDK

R2

LIEF

RetDec

Today

NATIVE HOOK CANNOT BE GENERATED
NO BYTECODE SYMBOLIC EXEC

Functions contained into JNI/native libs can be hooked, but decompilers/analyzers dont support it. So, native hook cannot be generated.

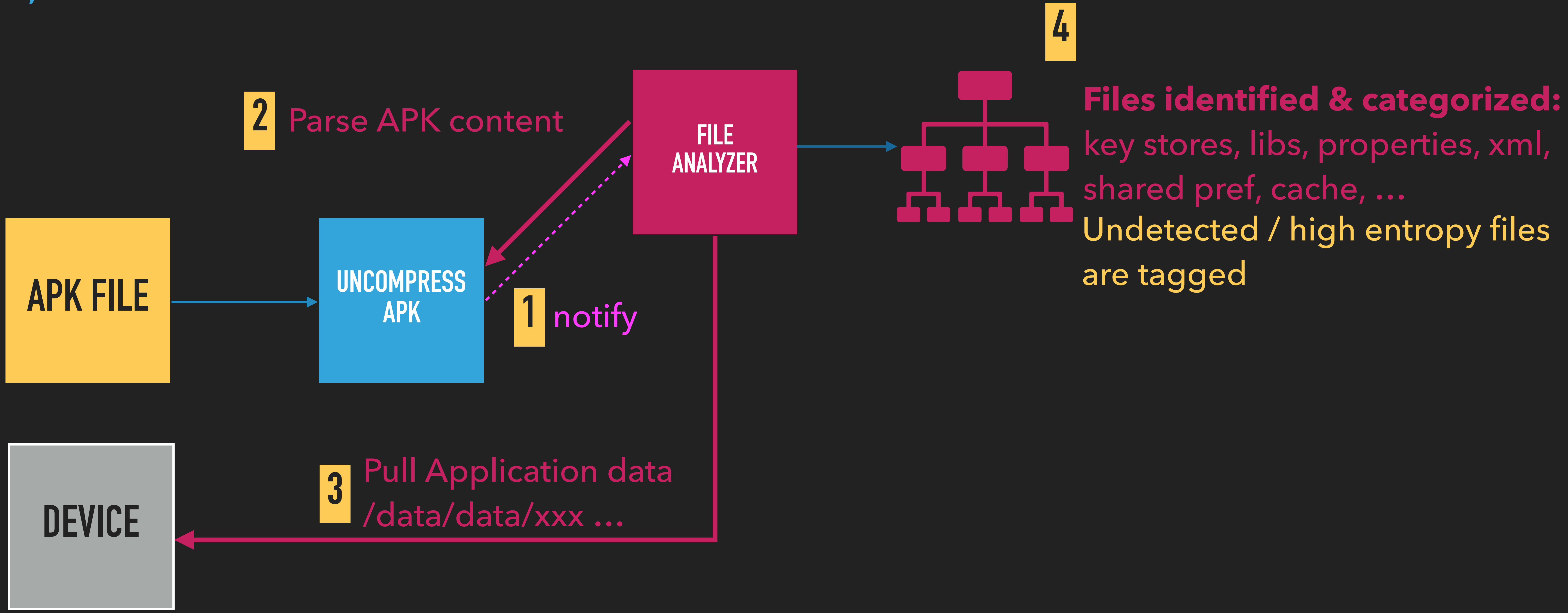
Tomorrow

ADD NATIVE LIBRARIES SUPPORT
SMALI SYMBOLIC EXEC

DEMO #1

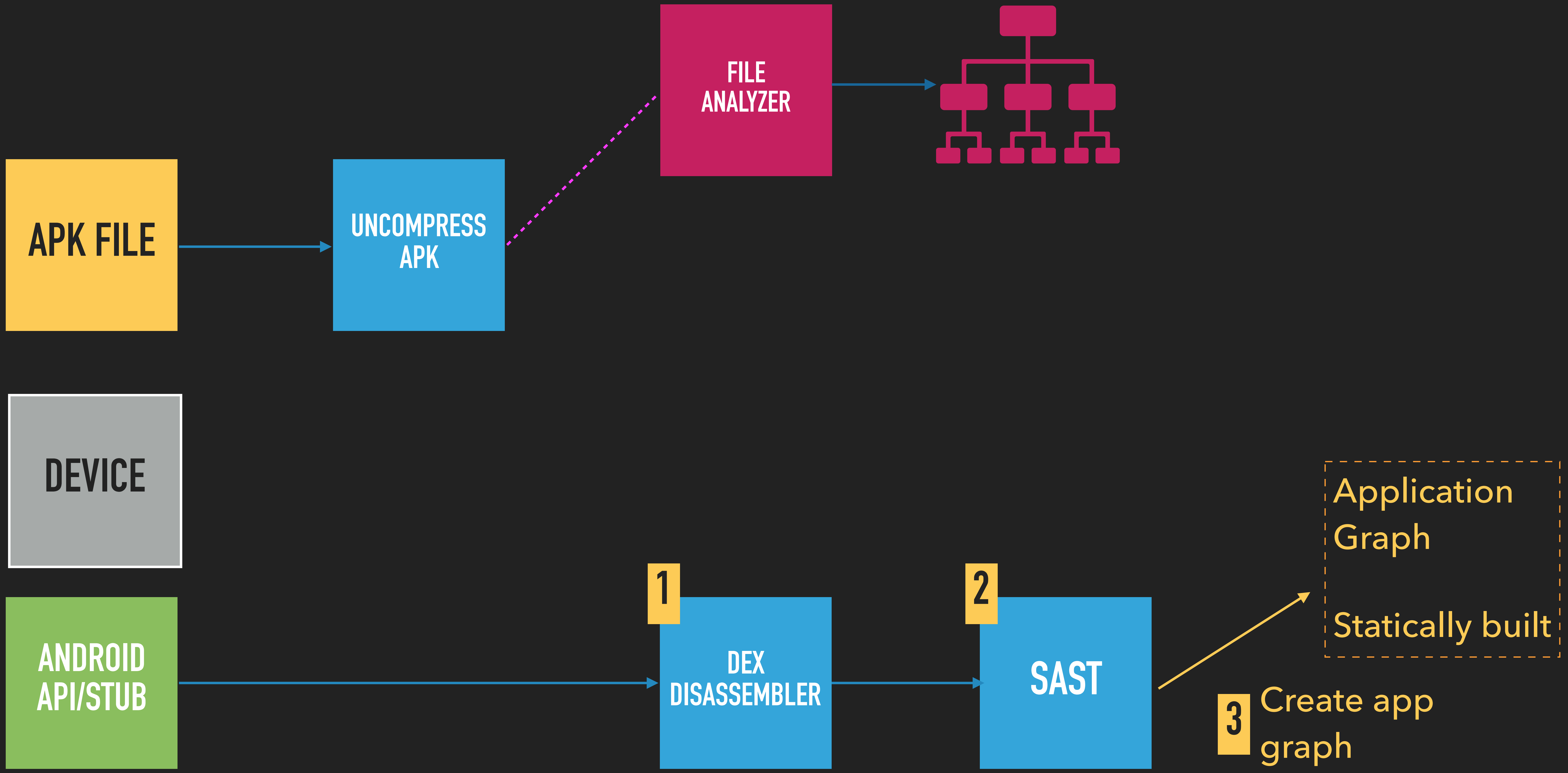
HOW IT WORKS ?

1) START PHASE – FILE ANALYSIS

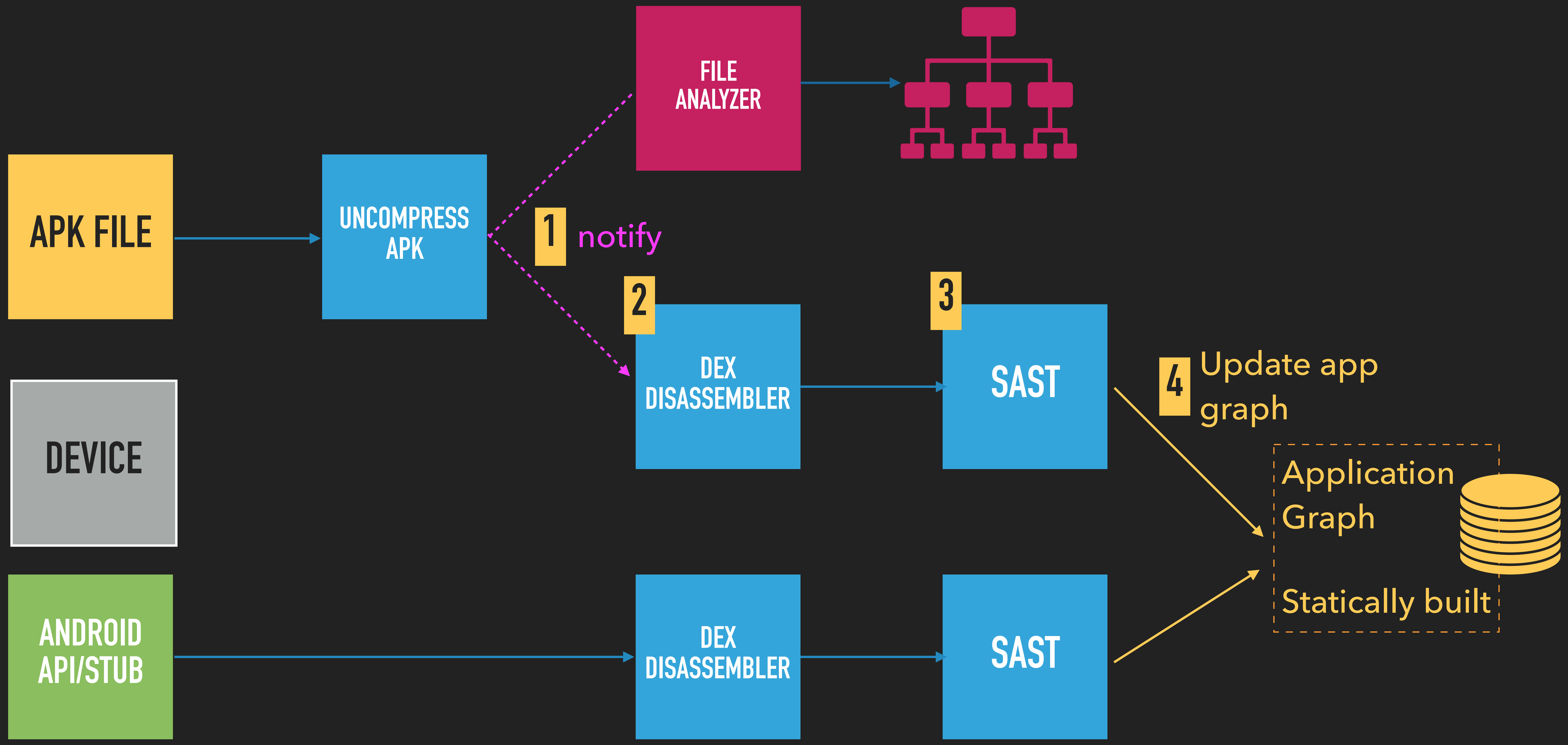


Files identified & categorized:
key stores, libs, properties, xml, shared pref, cache, …
Undetected / high entropy files are tagged

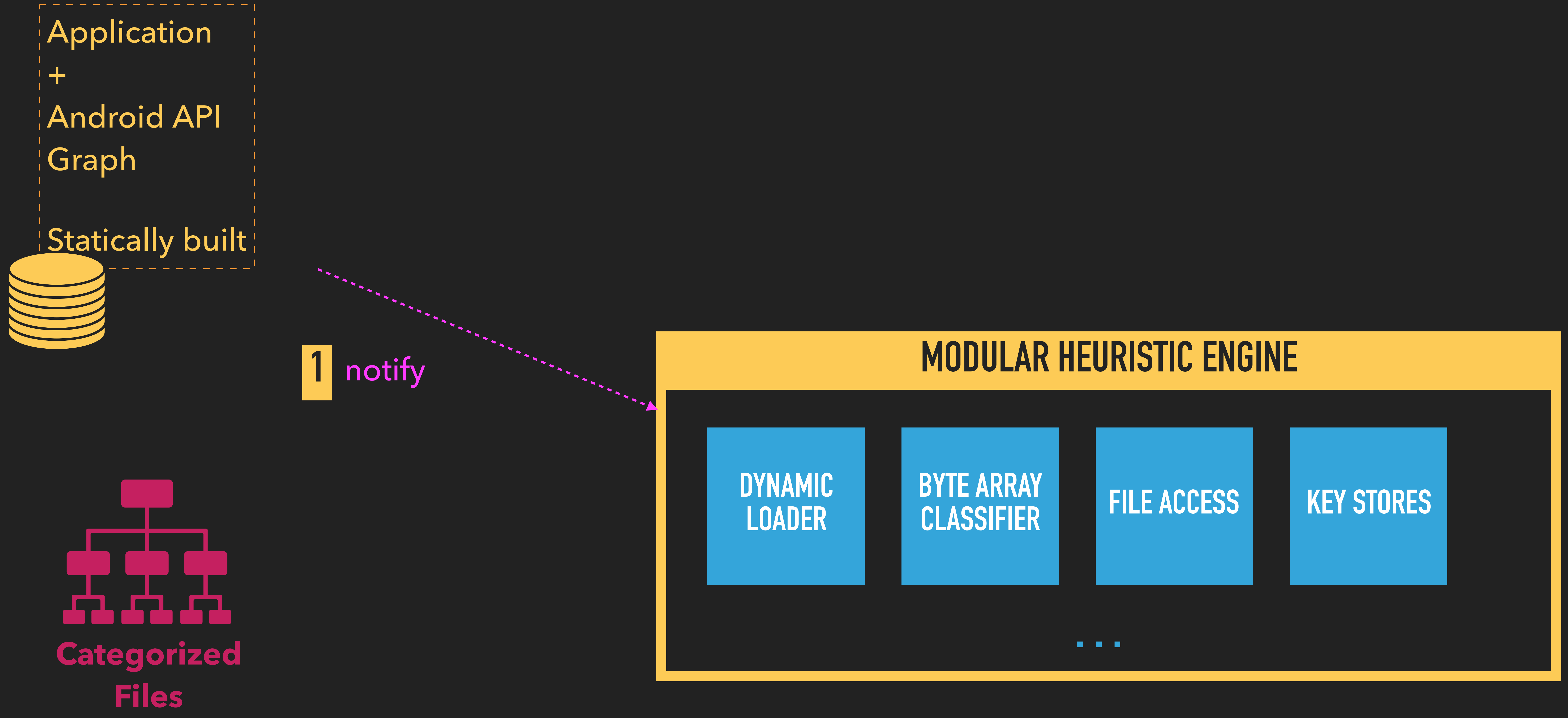
1) START PHASE – ANDROID API ANALYSIS



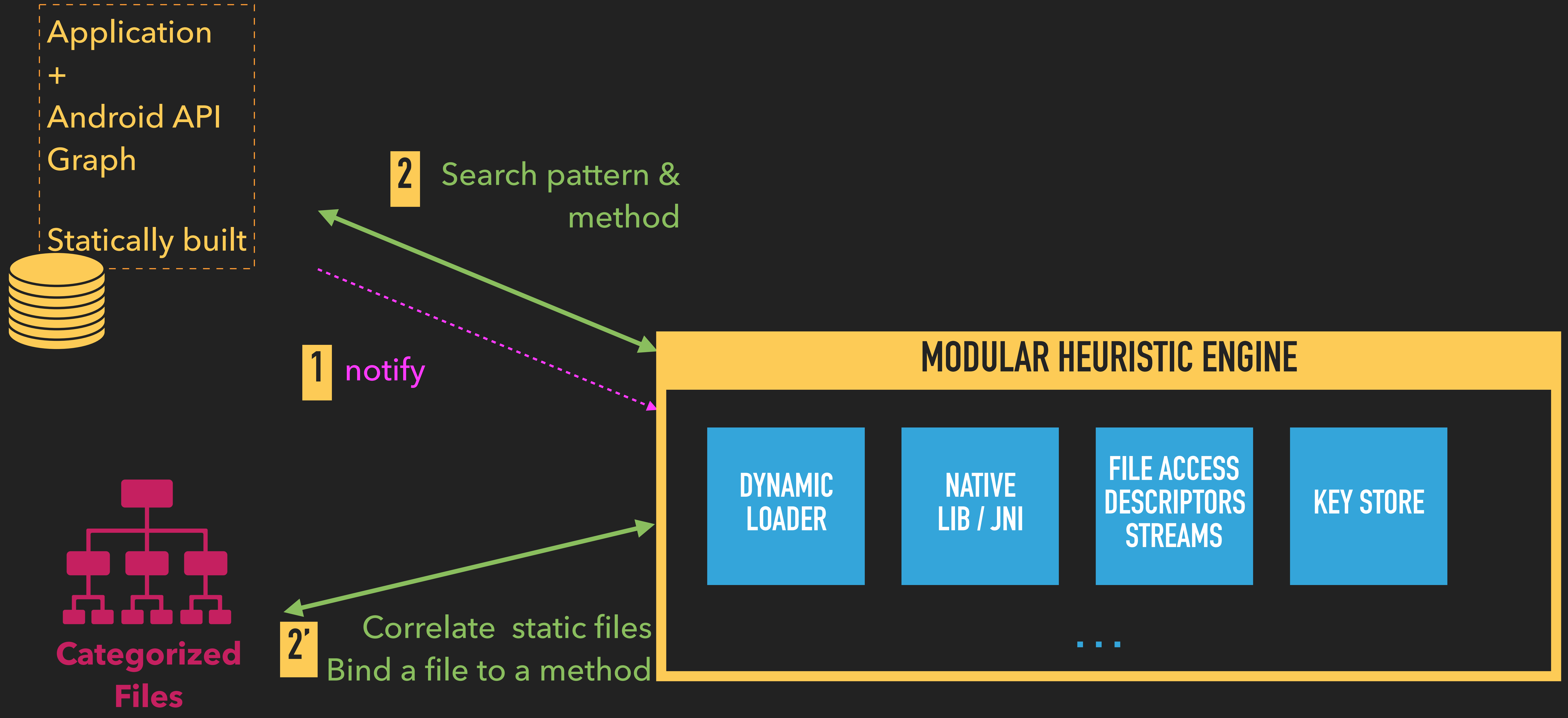
1) START PHASE - APPLICATION BYTE CODE ANALYSIS



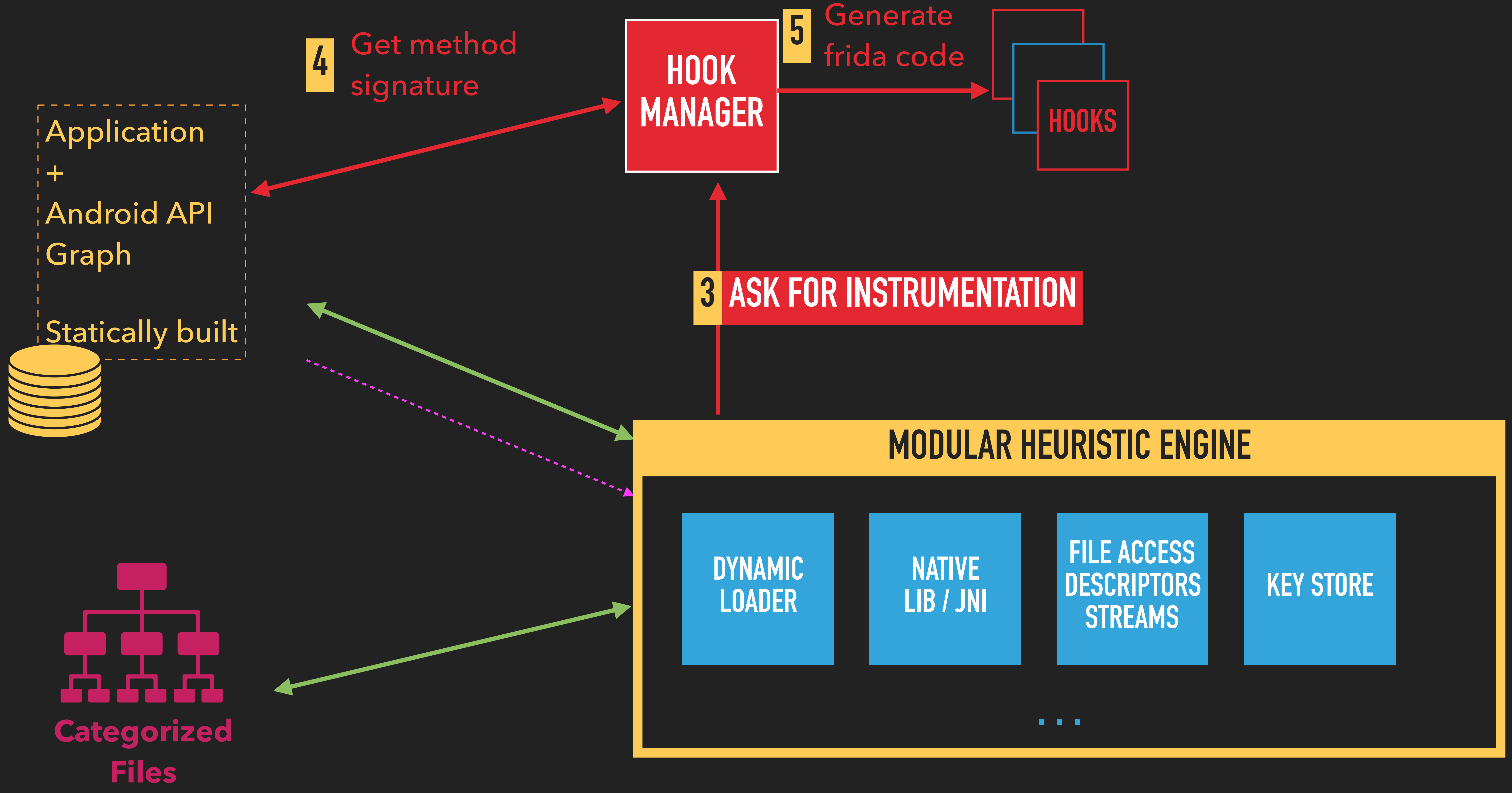
2) INSTRUMENTATION PHASE – BEFORE RUN



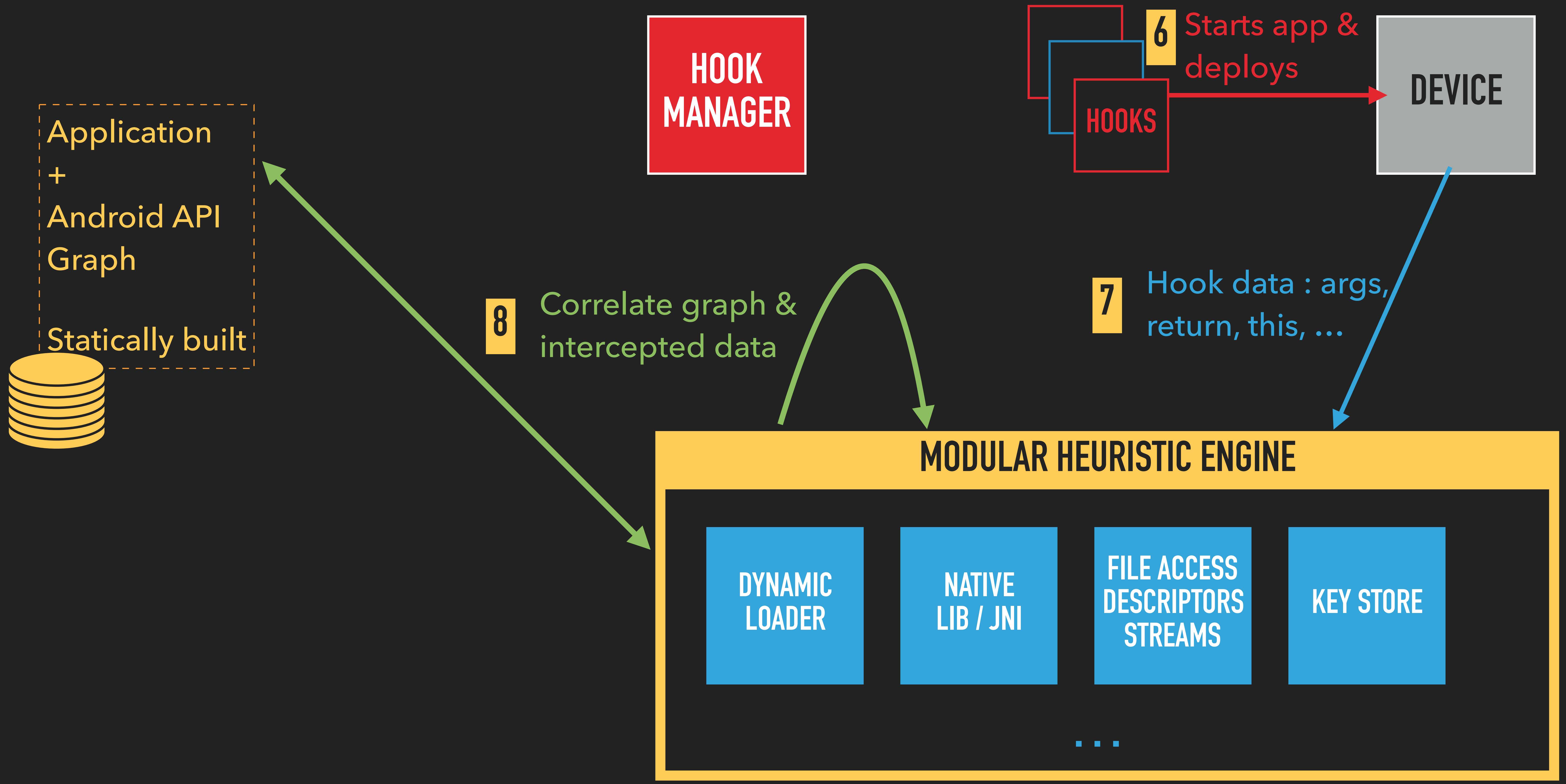
2) INSTRUMENTATION PHASE - BEFORE RUN



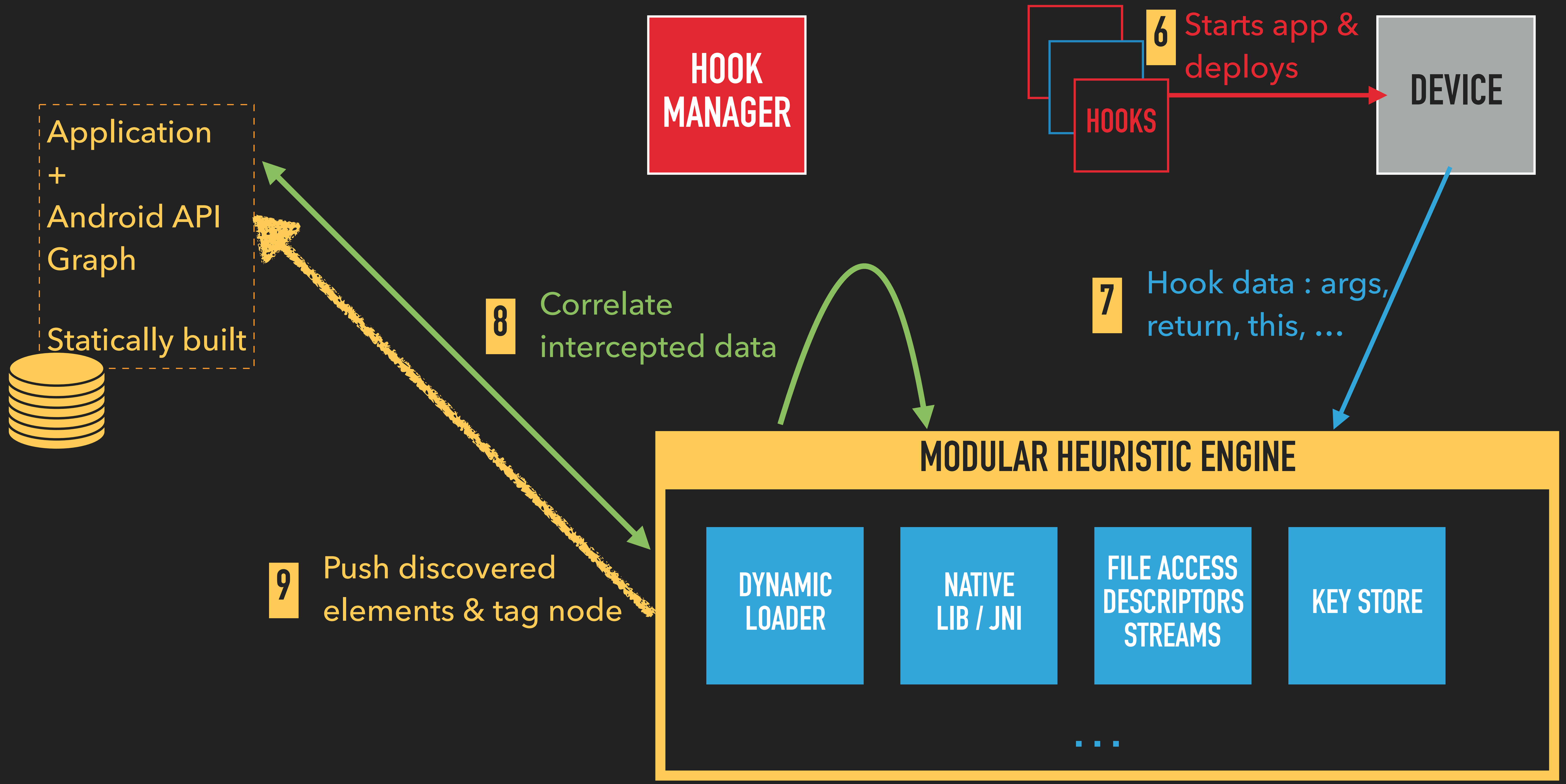
2) INSTRUMENTATION PHASE – BEFORE RUN



2) INSTRUMENTATION PHASE – RUNTIME

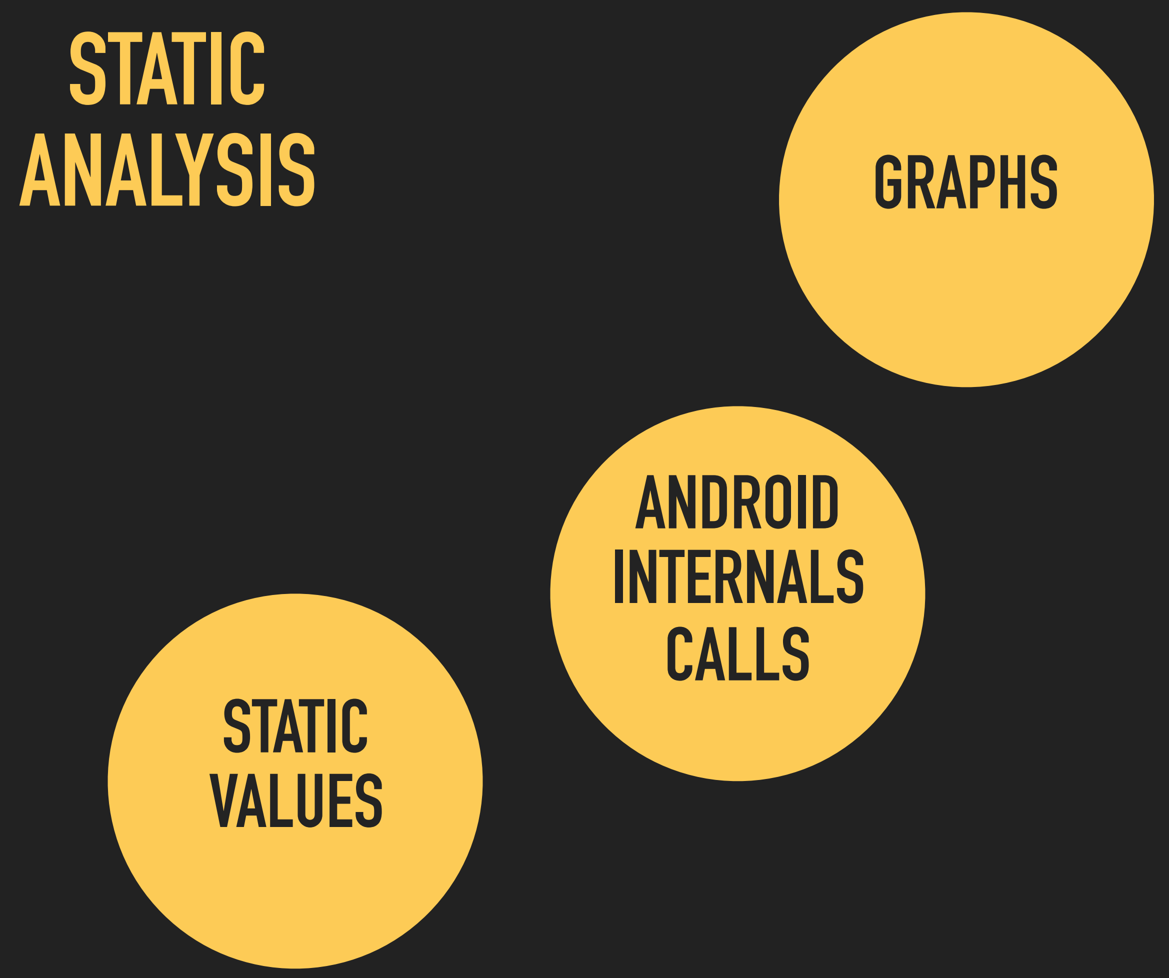

HOOK MANAGER
HOOKS
6 Starts app & deploys
DEVICE
Application + Android API Graph
Statically built
8 Correlate graph & intercepted data
7 Hook data : args, return, this, …
MODULAR HEURISTIC ENGINE
DYNAMIC LOADER
NATIVE LIB / JNI
FILE ACCESS DESCRIPTORS STREAMS
KEY STORE
…

2) INSTRUMENTATION PHASE - RUNTIME

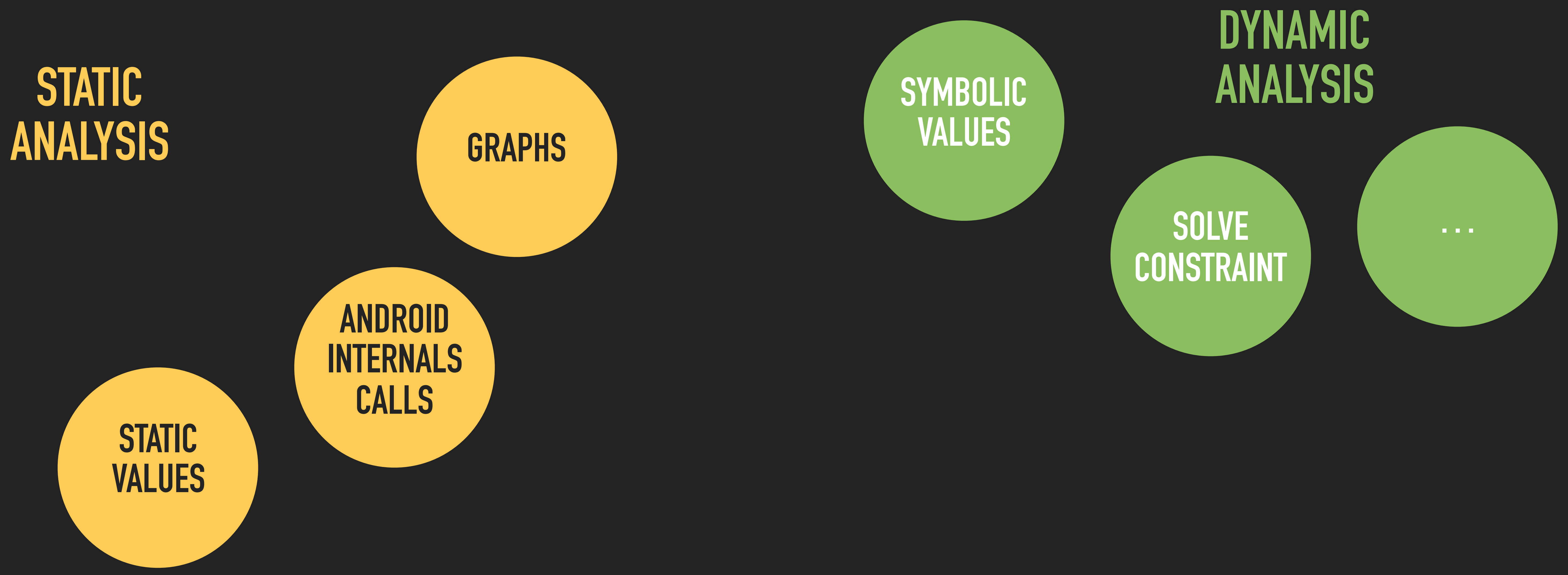

HOOK MANAGER
HOOKS
6 Starts app & deploys
DEVICE
Application + Android API Graph
Statically built
7 Hook data : args, return, this, …
8 Correlate intercepted data
9 Push discovered elements & tag node
MODULAR HEURISTIC ENGINE
DYNAMIC LOADER
NATIVE LIB / JNI
FILE ACCESS DESCRIPTORS STREAMS
KEY STORE
…

« HEY !
GIVE ME THE MOST
COMPLETE PICTURE OF
THE APPLICATION »

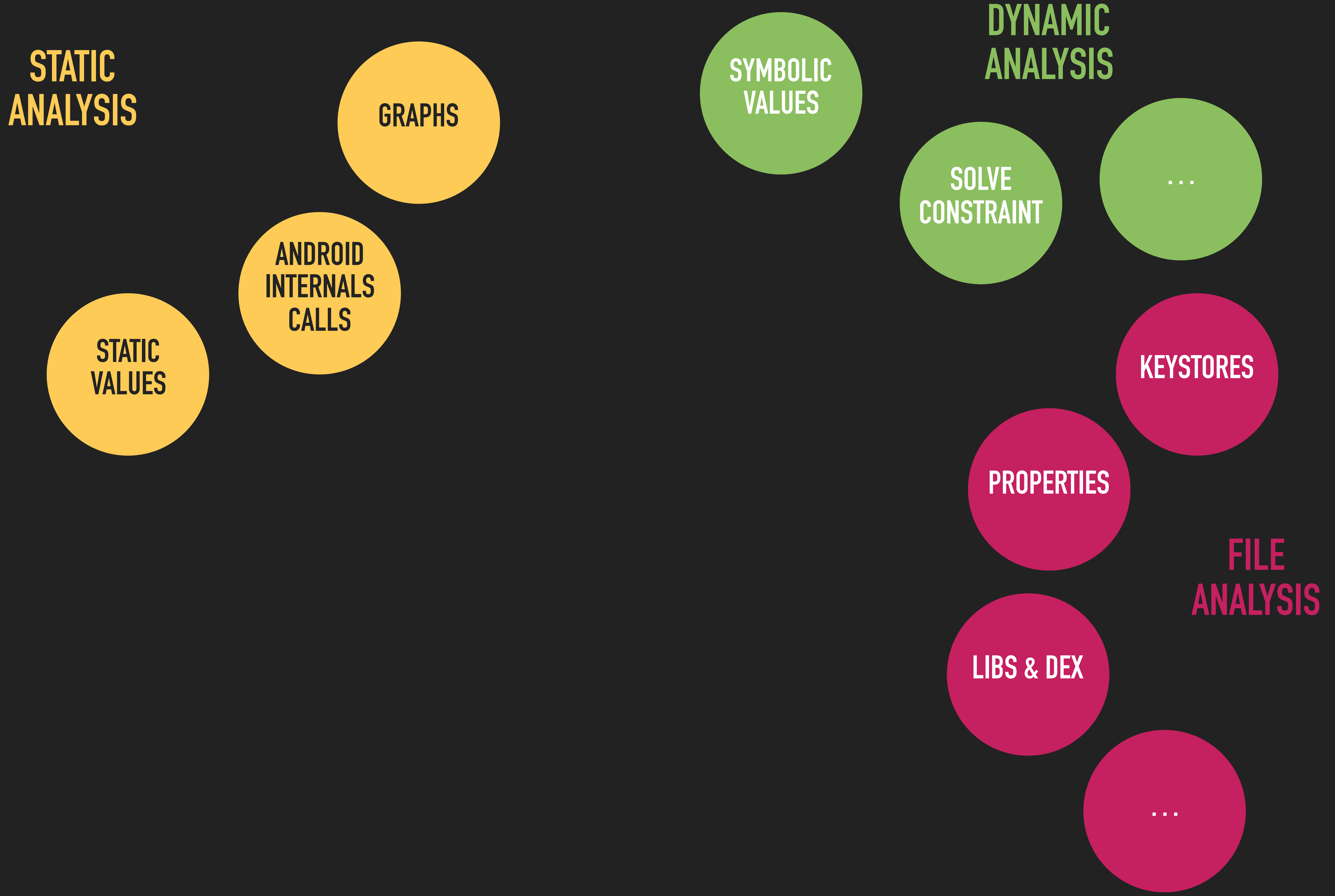
MIX * ANALYSIS WITH INSTRUMENTATION RESULTS



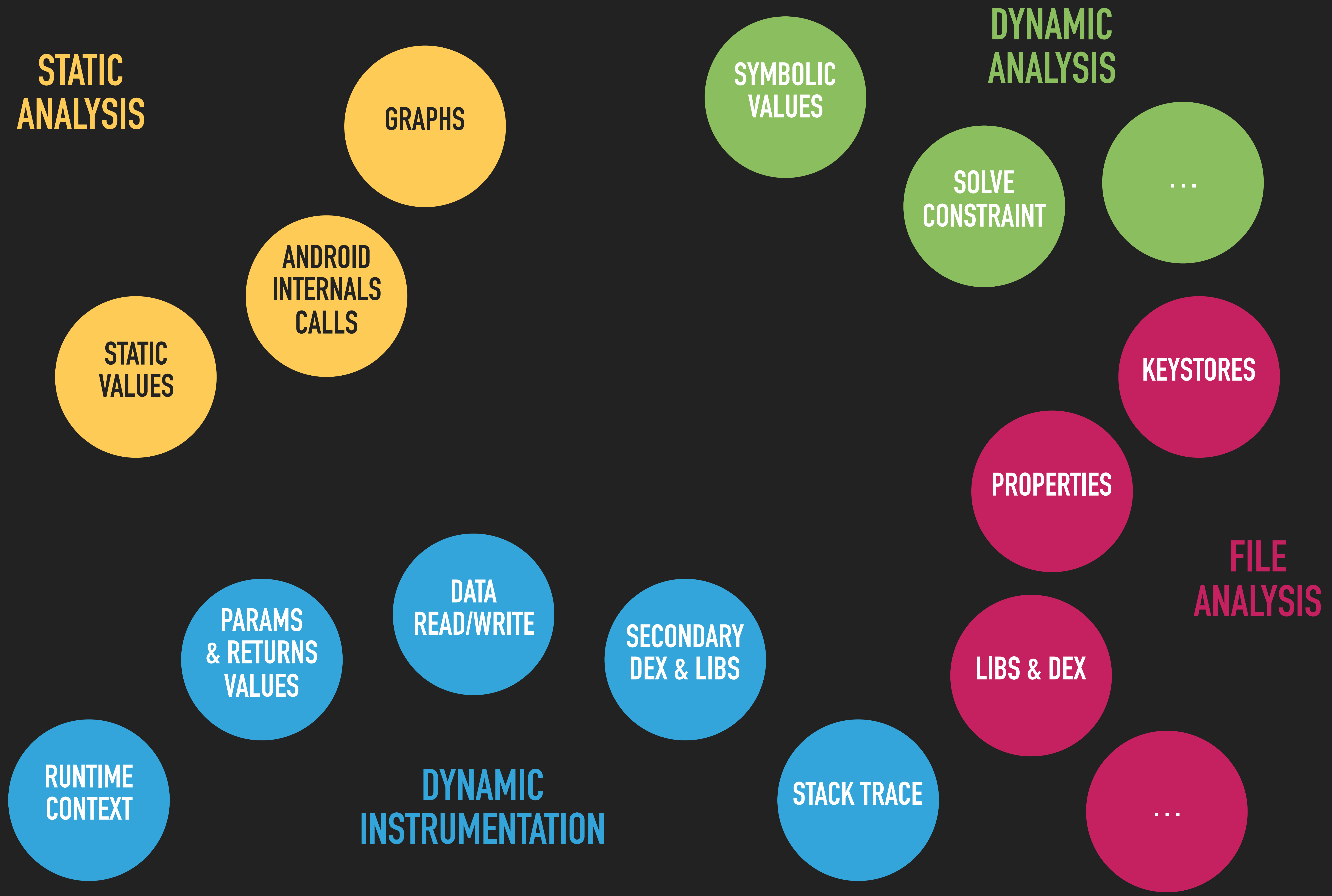
MIX * ANALYSIS WITH INSTRUMENTATION RESULTS



MIX * ANALYSIS WITH INSTRUMENTATION RESULTS



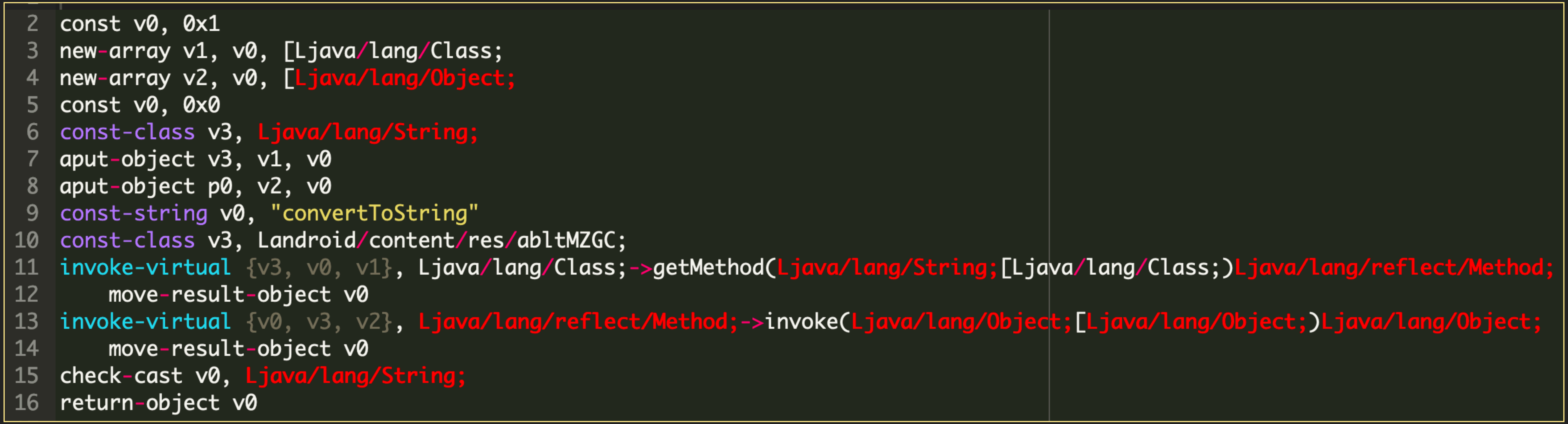
MIX * ANALYSIS WITH INSTRUMENTATION RESULTS



CASE #1

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS

METHOD INVOKED DYNAMICALLY



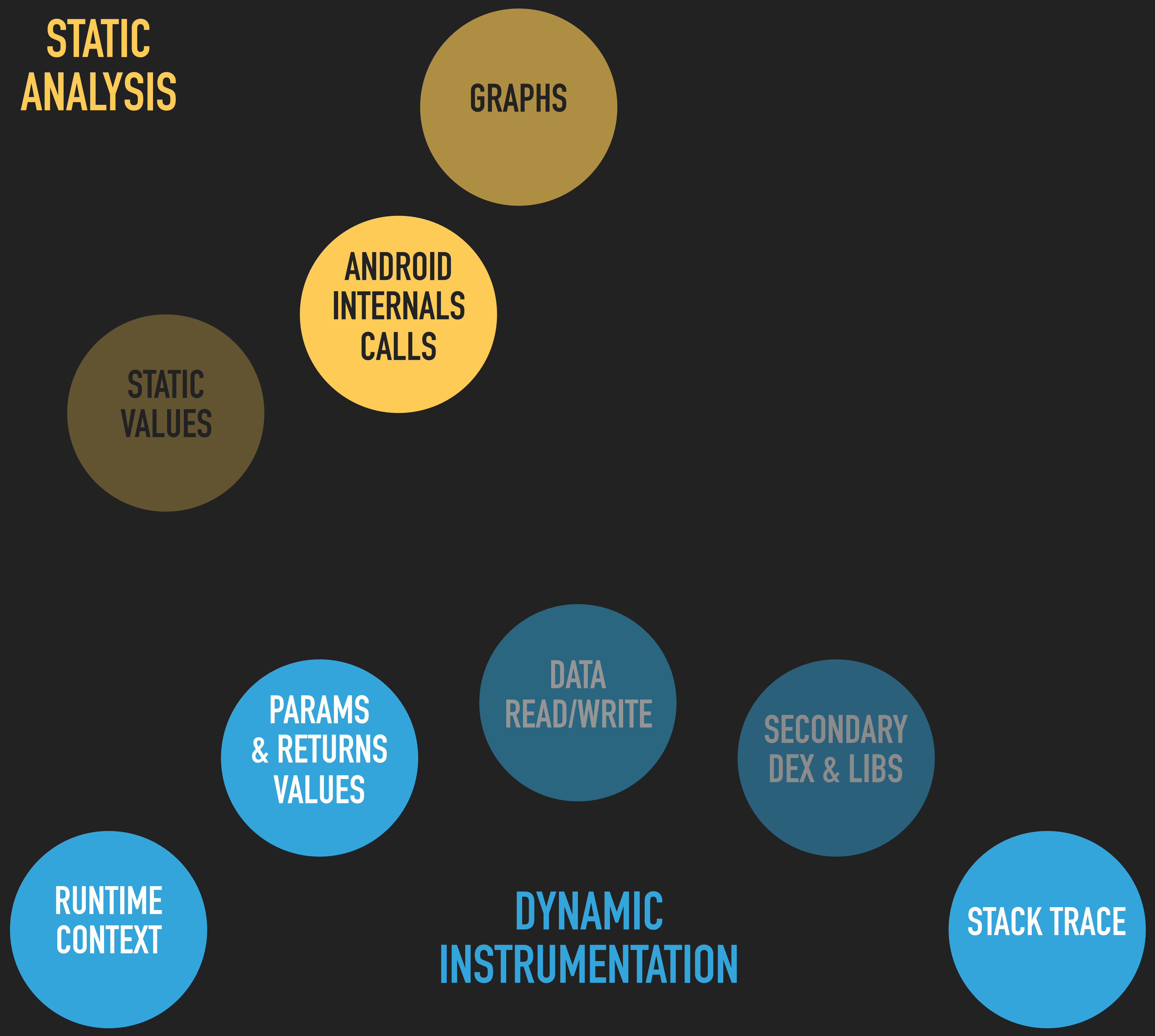
```
const v0, 0x1
new-array v1, v0, [Ljava/lang/Class;
new-array v2, v0, [Ljava/lang/Object;
const v0, 0x0
const-class v3, Ljava/lang/String;
aput-object v3, v1, v0
aput-object p0, v2, v0
const-string v0, "convertToString"
const-class v3, Landroid/content/res/abltMZGC;
invoke-virtual {v3, v0, v1}, Ljava/lang/Class;->getMethod(Ljava/lang/String;[Ljava/lang/Class;)Ljava/lang/reflect/Method;
    move-result-object v0
invoke-virtual {v0, v3, v2}, Ljava/lang/reflect/Method;->invoke(Ljava/lang/Object;[Ljava/lang/Object;)Ljava/lang/Object;
    move-result-object v0
check-cast v0, Ljava/lang/String;
return-object v0
```

Smali code

From a static point-of-view only two methods are called :

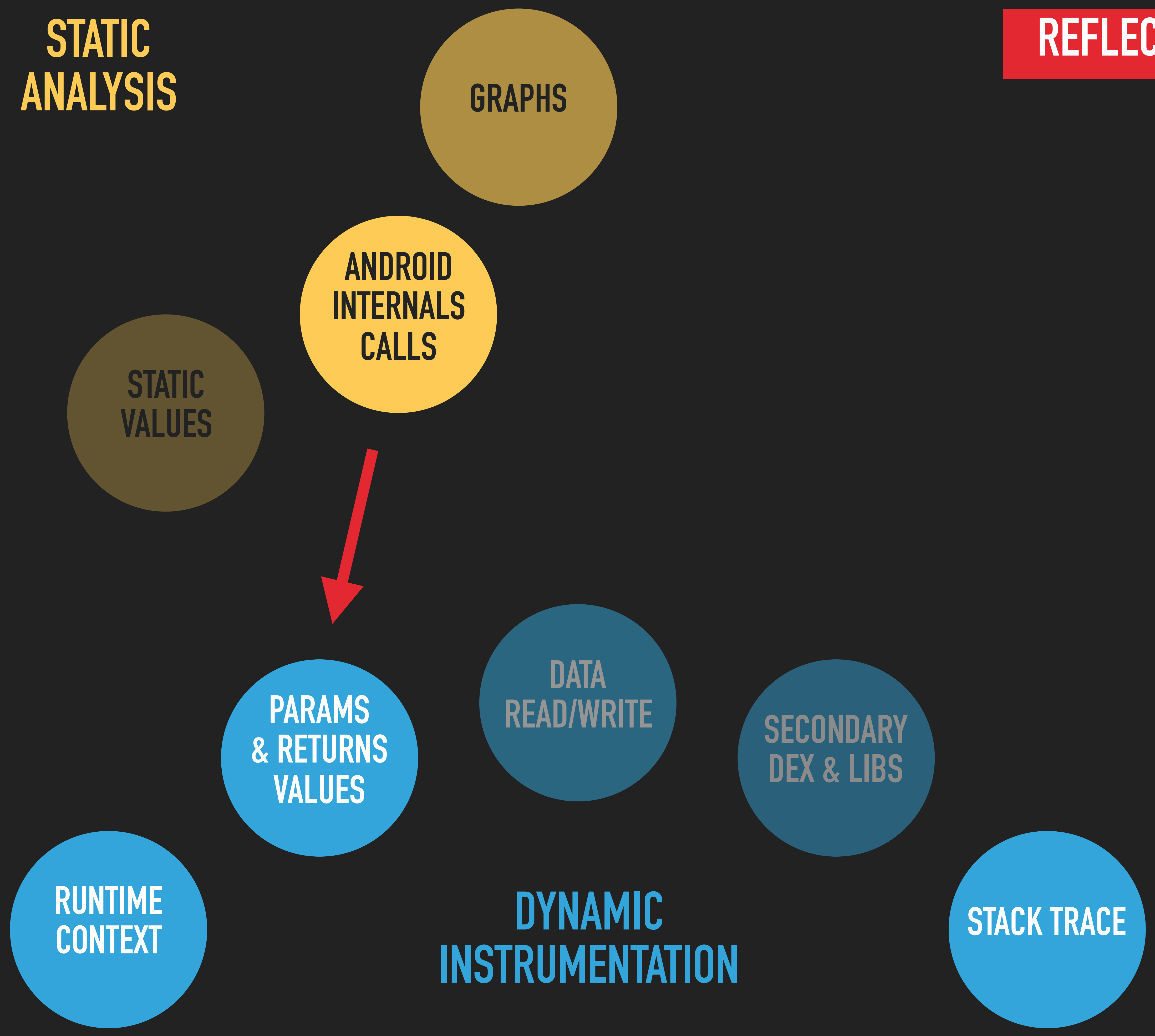
- Class.getMethod()
- Method.invoke()

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS

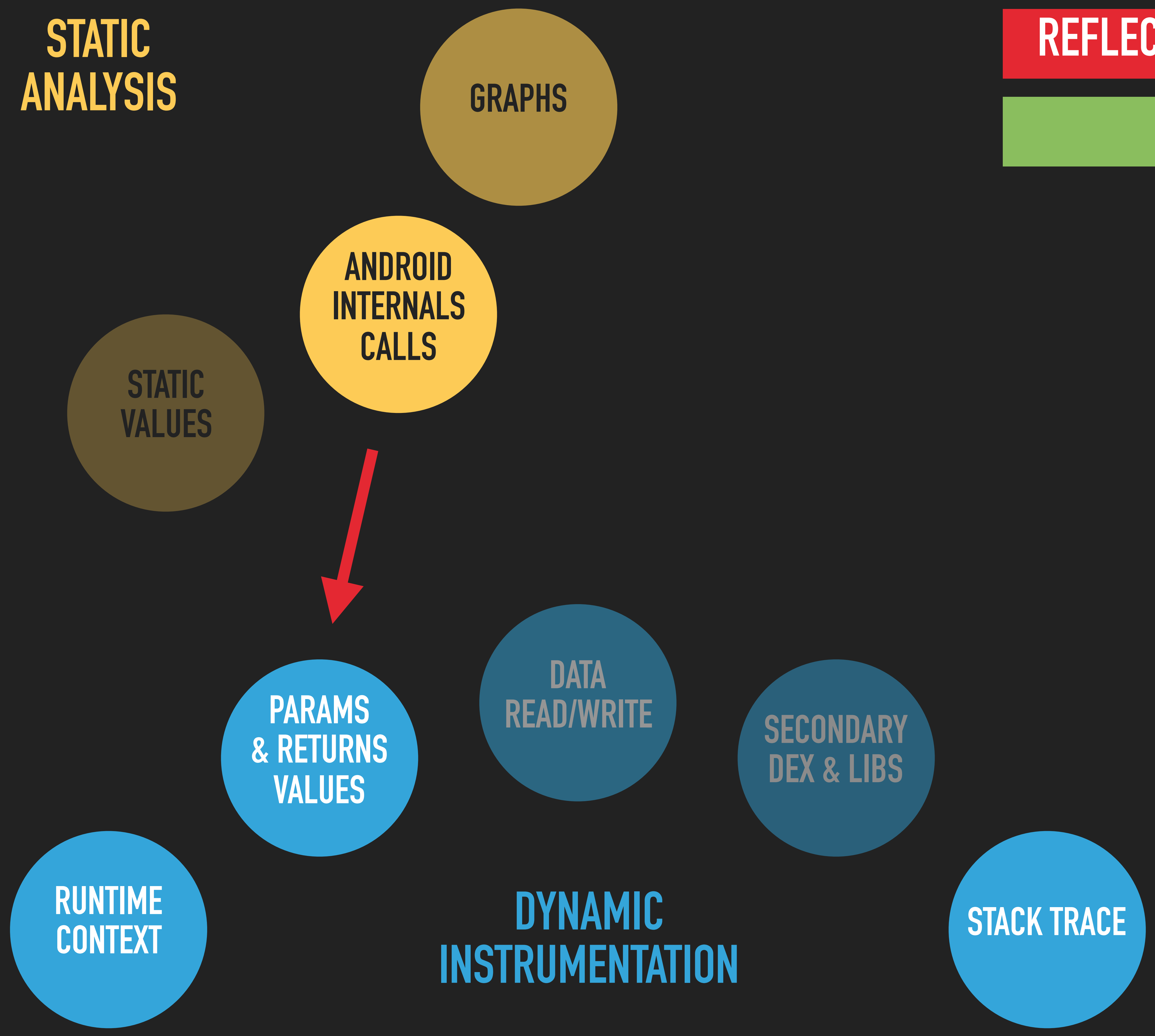


DYNAMIC UPDATE OF XREF WITH INVOKED METHODS

REFLECTION API INSTRUMENTED



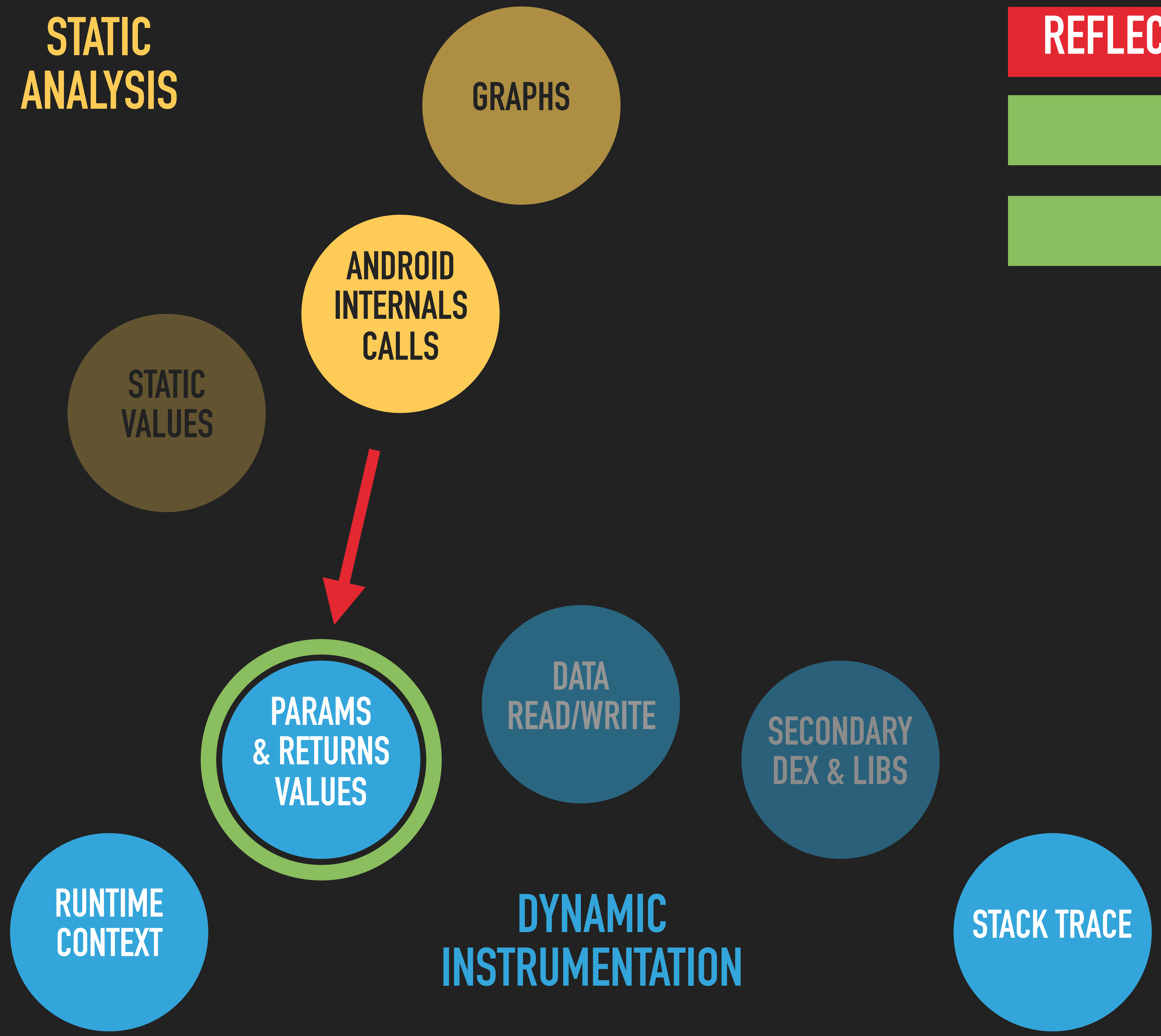
DYNAMIC UPDATE OF XREF WITH INVOKED METHODS



REFLECTION API INSTRUMENTED

START APP

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS

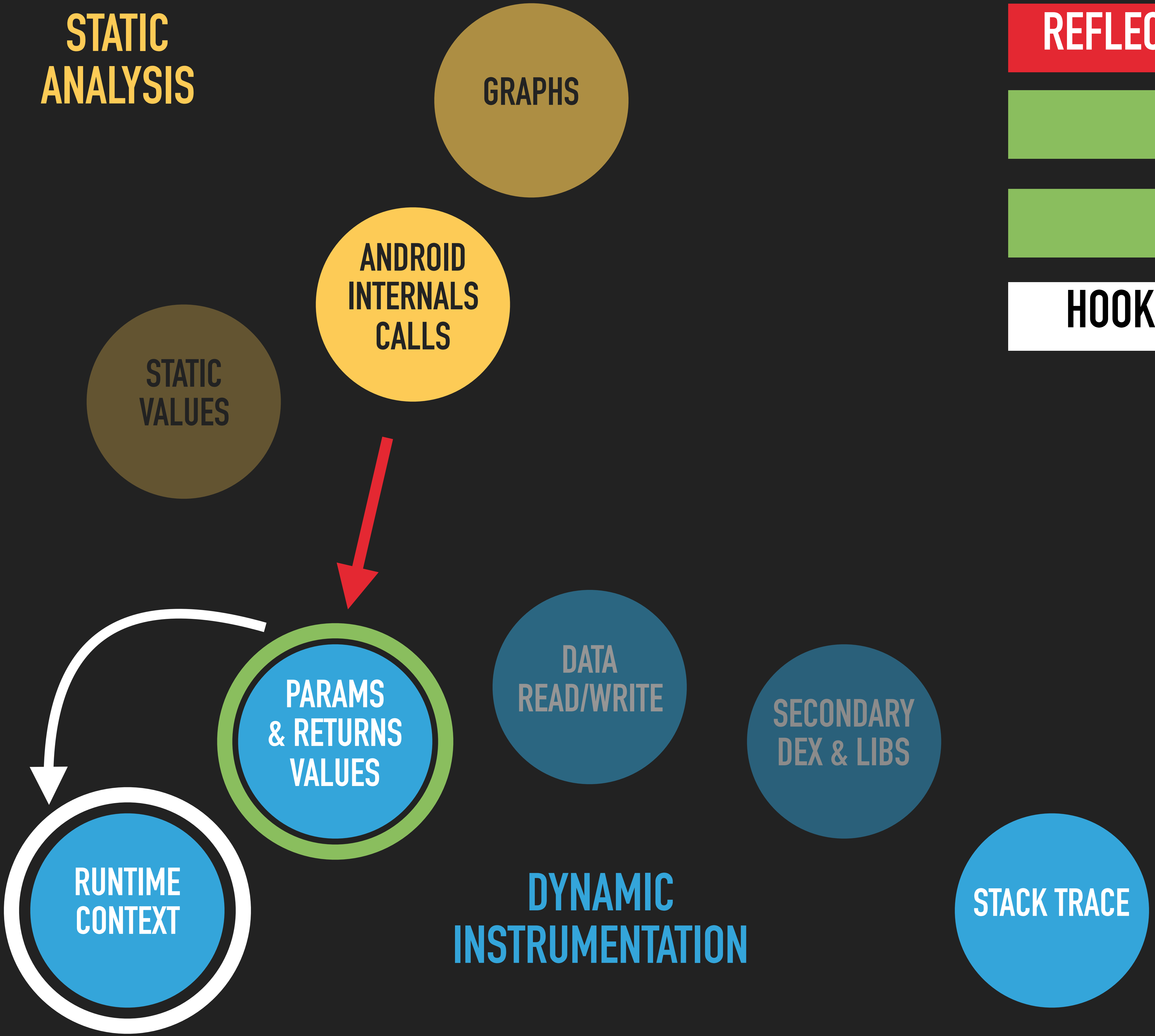


REFLECTION API INSTRUMENTED

START APP

HOOK TRIGGED

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS



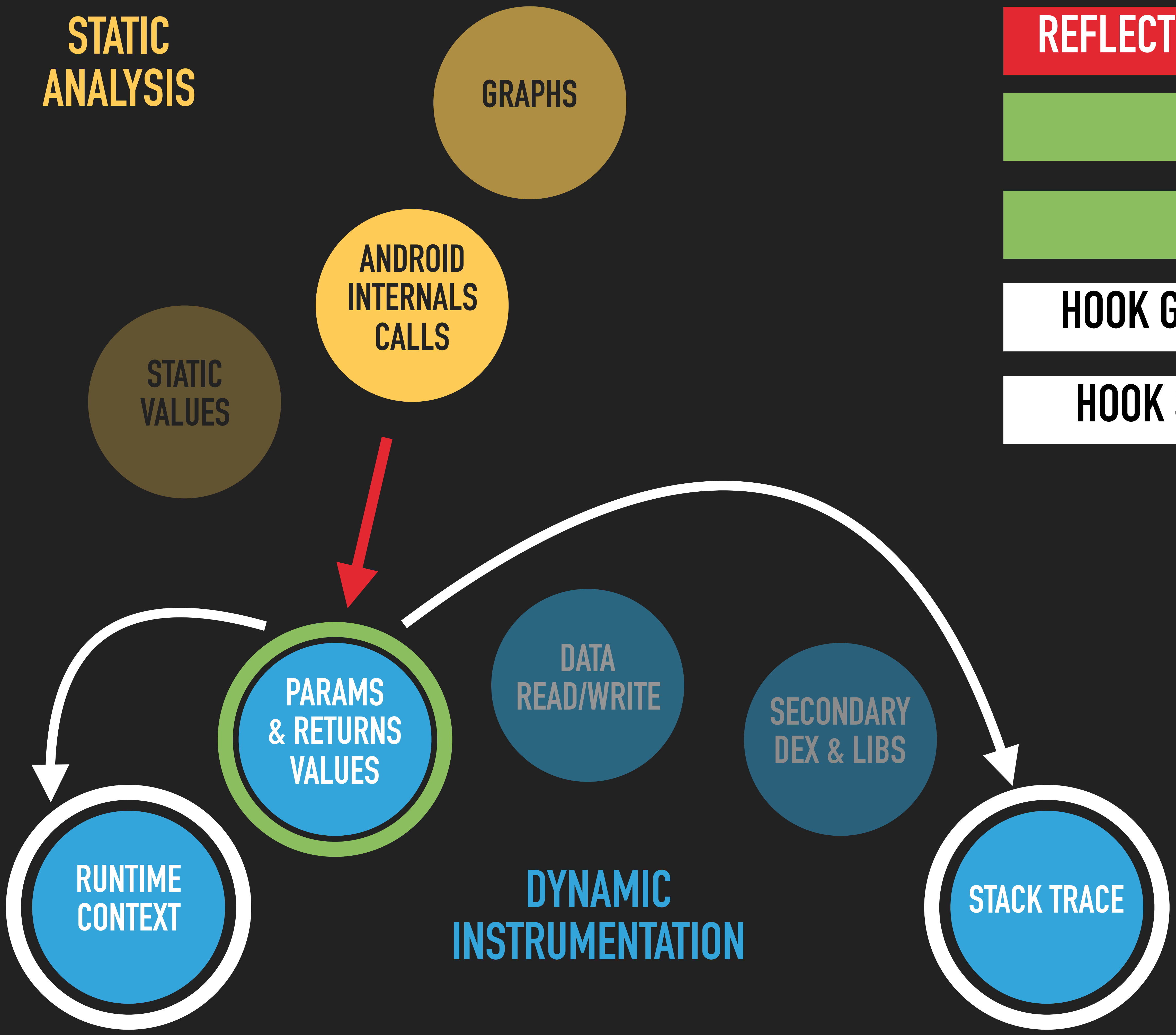
REFLECTION API INSTRUMENTED

START APP

HOOK TRIGGED

HOOK GATHERS METHOD INFO

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS



REFLECTION API INSTRUMENTED

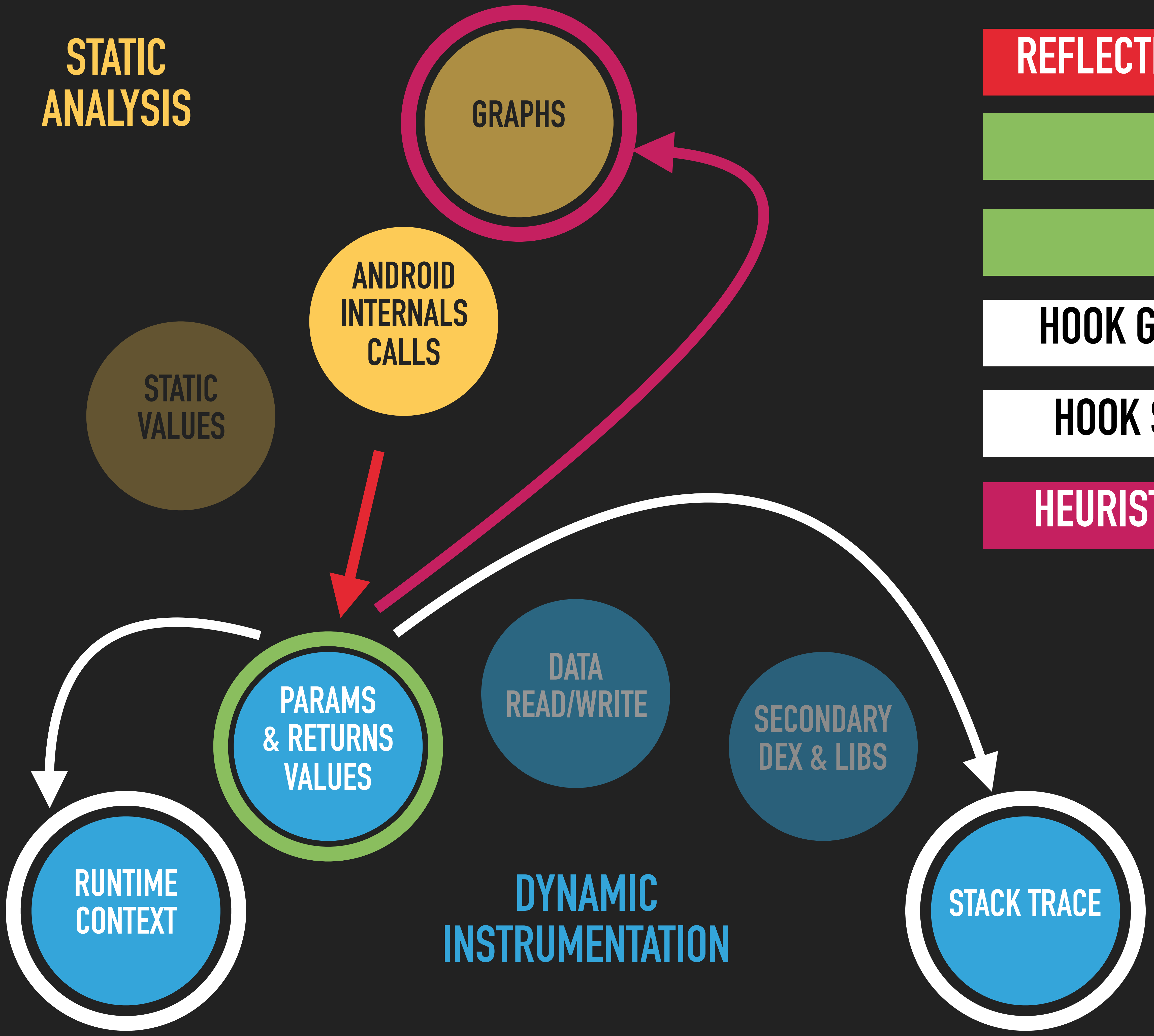
START APP

HOOK TRIGGED

HOOK GATHERS METHOD INFO

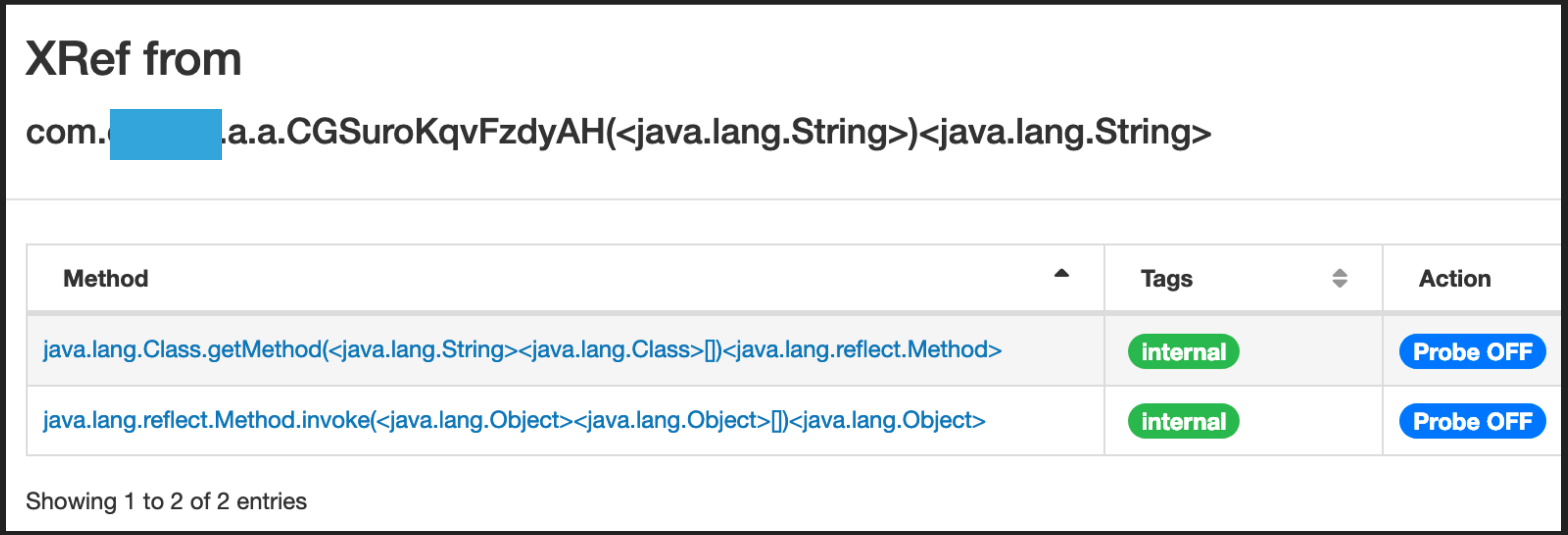
HOOK SHOWS STACK TRACE

DYNAMIC UPDATE OF XREF WITH INVOKED METHODS



- REFLECTION API INSTRUMENTED
- START APP
- HOOK TRIGGED
- HOOK GATHERS METHOD INFO
- HOOK SHOWS STACK TRACE
- HEURISTIC ENGINE UPDATE DB

METHOD INVOKED DYNAMICALLY



XRef from

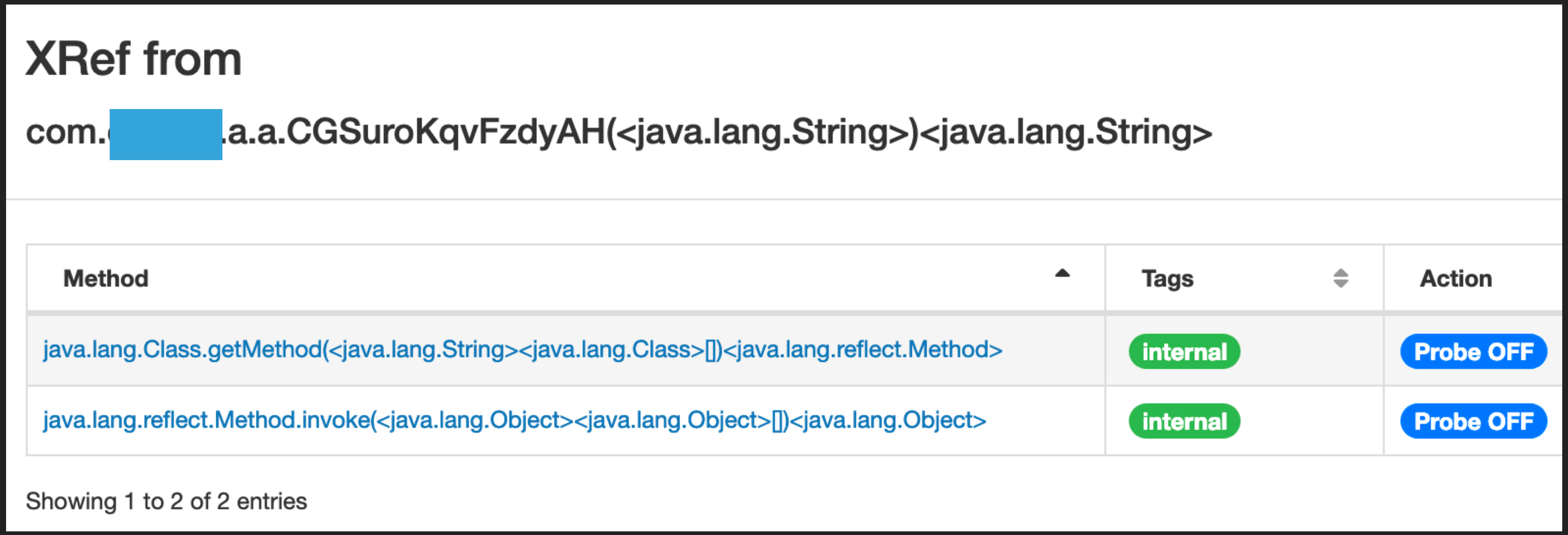
com.[redacted].a.a.CGSuroKqvFzdyAH(<java.lang.String>)<java.lang.String>

Method	Tags	Action
java.lang.Class.getMethod(<java.lang.String><java.lang.Class>[])<java.lang.reflect.Method>	internal	Probe OFF
java.lang.reflect.Method.invoke(<java.lang.Object><java.lang.Object>[])<java.lang.Object>	internal	Probe OFF

Showing 1 to 2 of 2 entries

BEFORE RUNTIME

METHOD INVOKED DYNAMICALLY



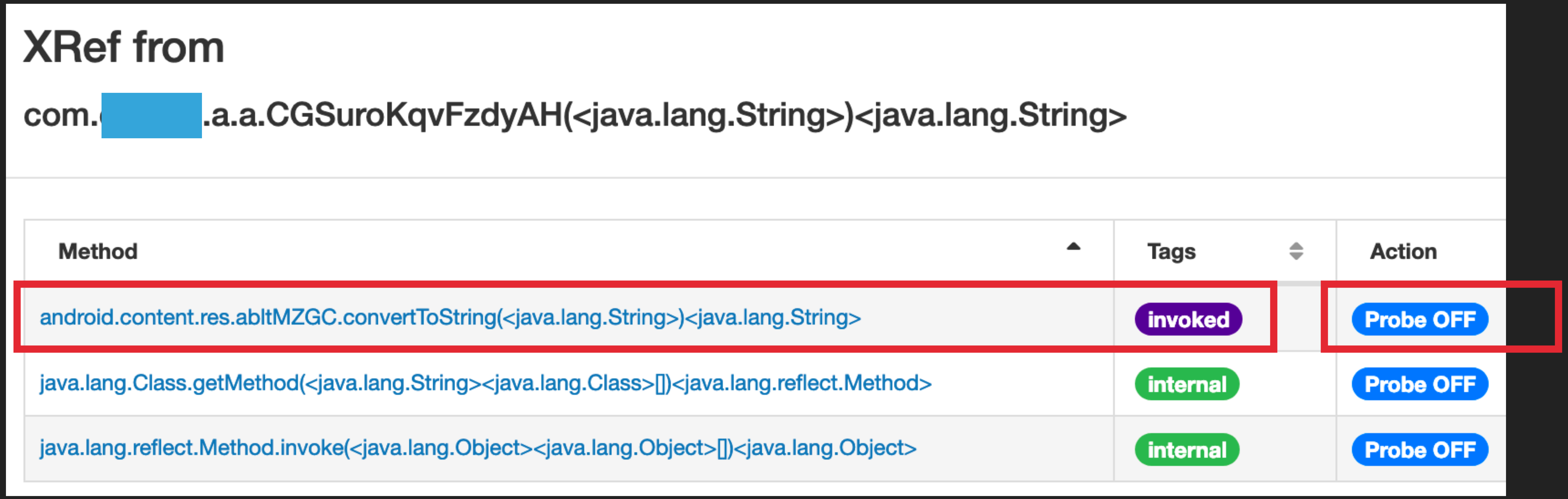
BEFORE RUNTIME

XRef from

com.[redacted].a.a.CGSuroKqvFzdyAH(<java.lang.String>)<java.lang.String>

Method	Tags	Action
java.lang.Class.getMethod(<java.lang.String><java.lang.Class>[])<java.lang.reflect.Method>	internal	Probe OFF
java.lang.reflect.Method.invoke(<java.lang.Object><java.lang.Object>[])<java.lang.Object>	internal	Probe OFF

Showing 1 to 2 of 2 entries



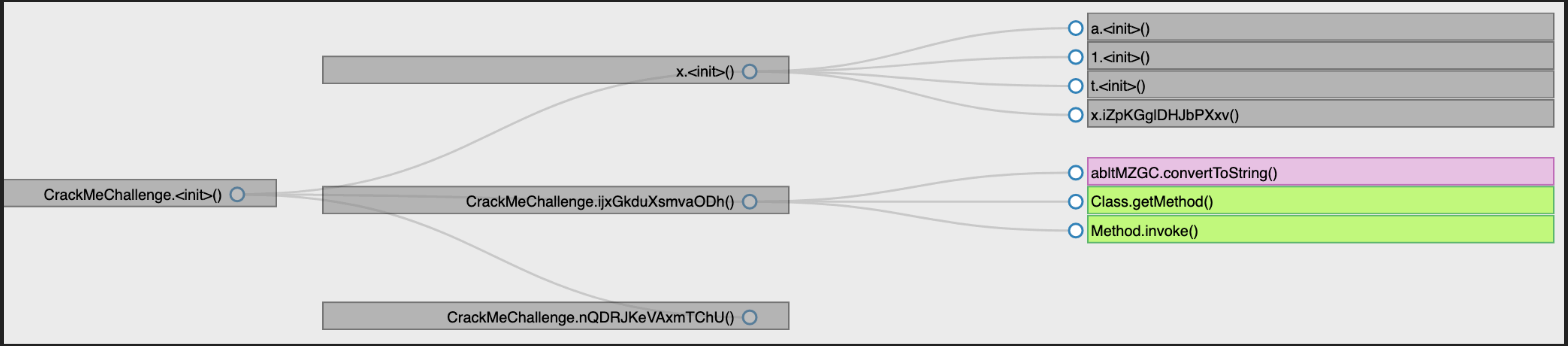
AFTER RUNTIME

XRef from

com.[redacted].a.a.CGSuroKqvFzdyAH(<java.lang.String>)<java.lang.String>

Method	Tags	Action
android.content.res.abltMZGC.convertToString(<java.lang.String>)<java.lang.String>	invoked	Probe OFF
java.lang.Class.getMethod(<java.lang.String><java.lang.Class>[])<java.lang.reflect.Method>	internal	Probe OFF
java.lang.reflect.Method.invoke(<java.lang.Object><java.lang.Object>[])<java.lang.Object>	internal	Probe OFF

UPDATE OF THE CALL GRAPH


CrackMeChallenge.<init>()
x.<init>()
CrackMeChallenge.ijxGkduXsmvaODh()
CrackMeChallenge.nQDRJKeVAxmTChU()
a.<init>()
1.<init>()
t.<init>()
x.iZpKGglDHJbPXxv()
abltMZGC.convertToString()
Class.getMethod()
Method.invoke()

Gray nodes have been discovered statically

Green nodes are internal Android or Java methods

Pink node are invoked dynamically and not discovered statically

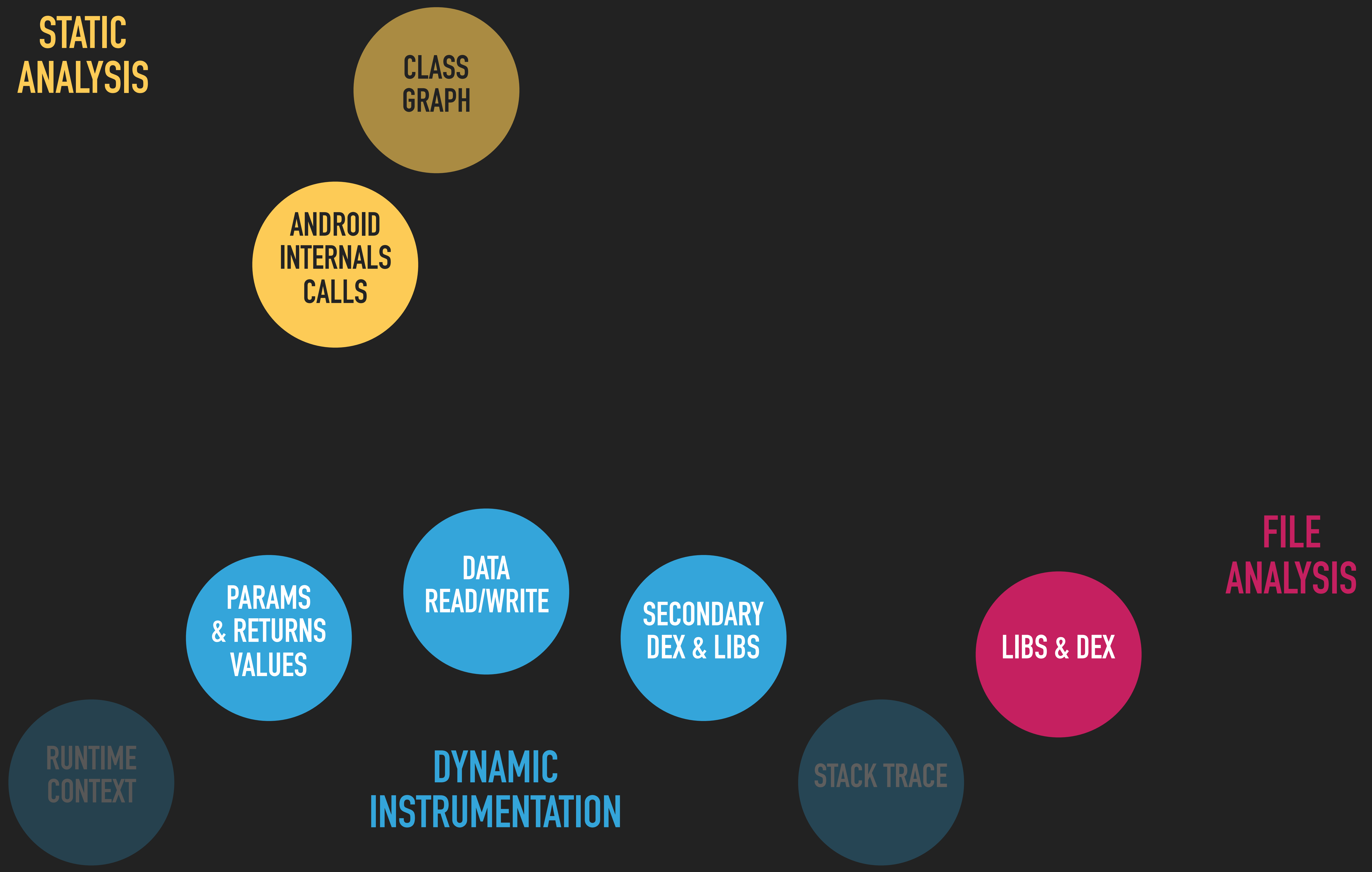
DEMO #2

DYNAMIC UPDATE OF XREFS
WITH INVOKED METHODS

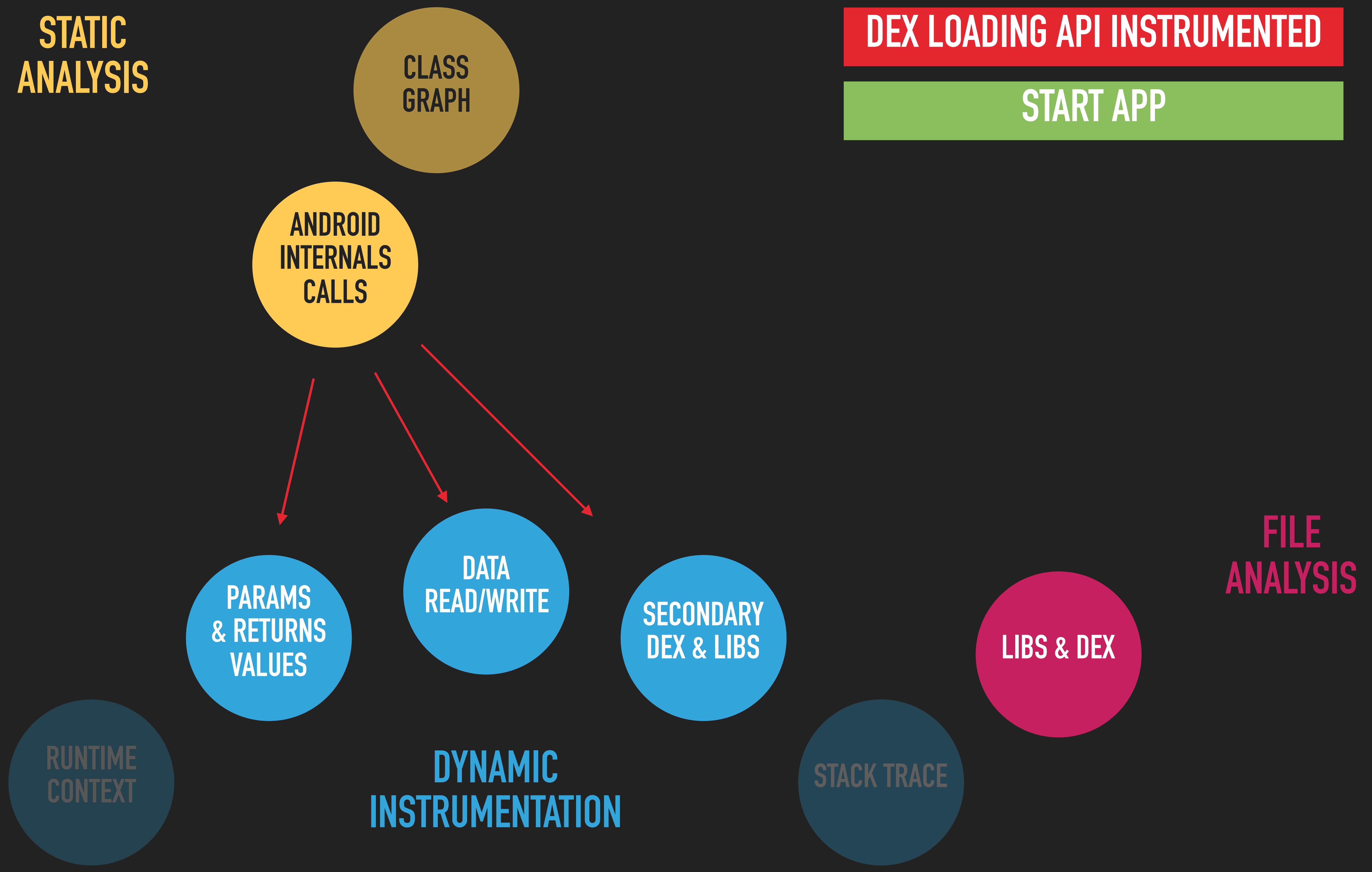
CASE #2

ANALYZE DEX FILE LOADED DYNAMICALLY

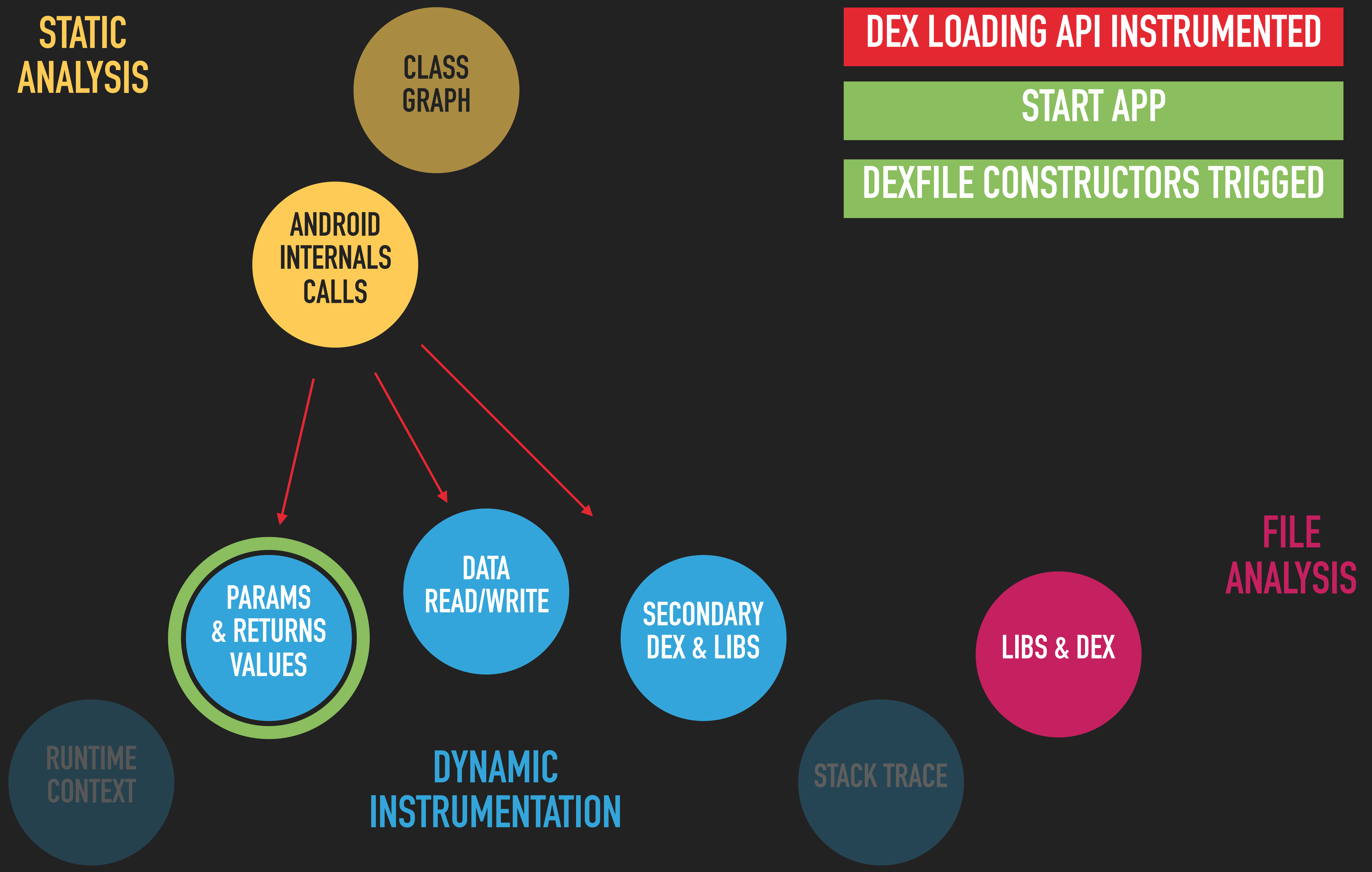
ANALYZE DEX FILE LOADED DYNAMICALLY



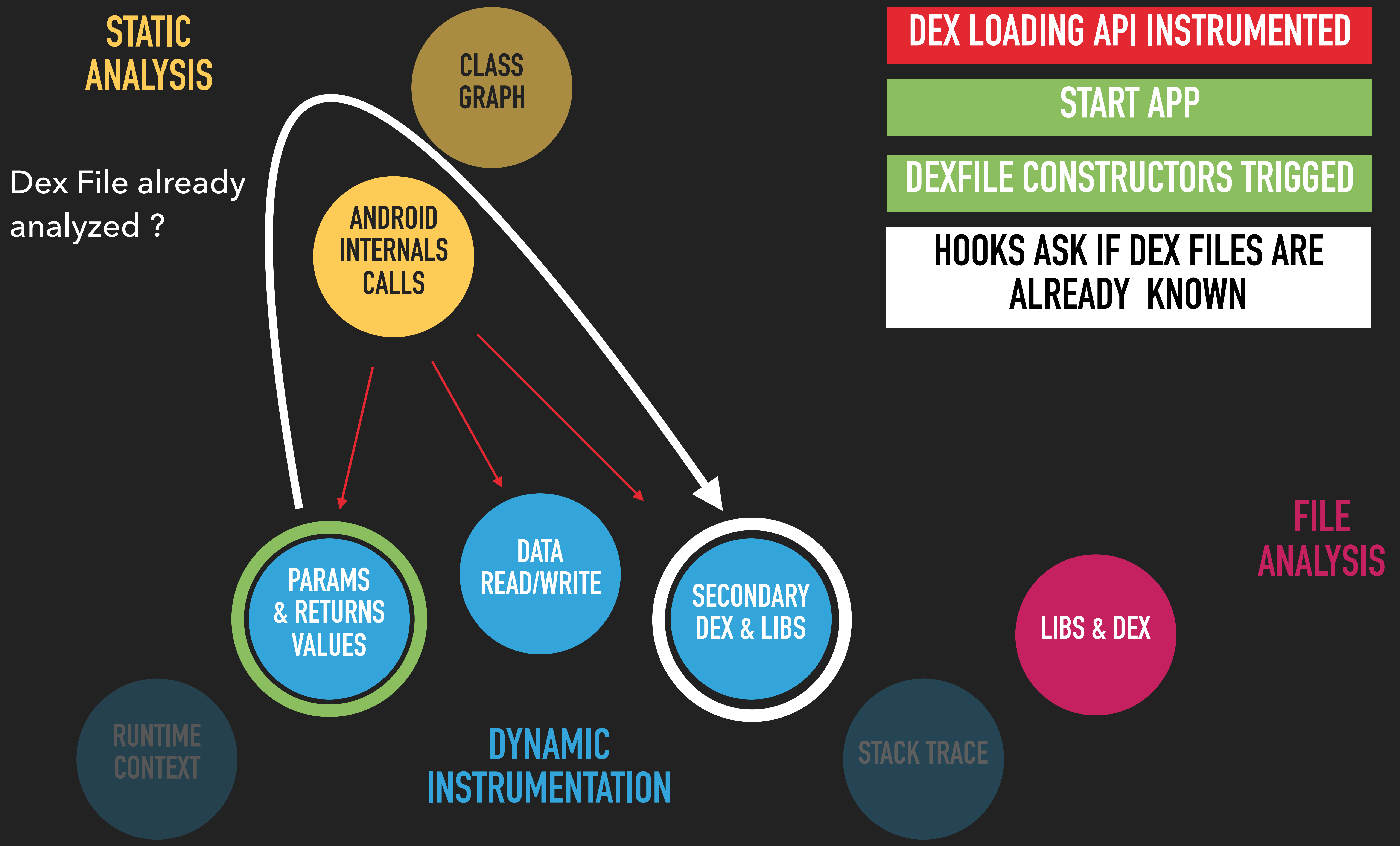
ANALYZE DEX FILE LOADED DYNAMICALLY



ANALYZE DEX FILE LOADED DYNAMICALLY



ANALYZE DEX FILE LOADED DYNAMICALLY



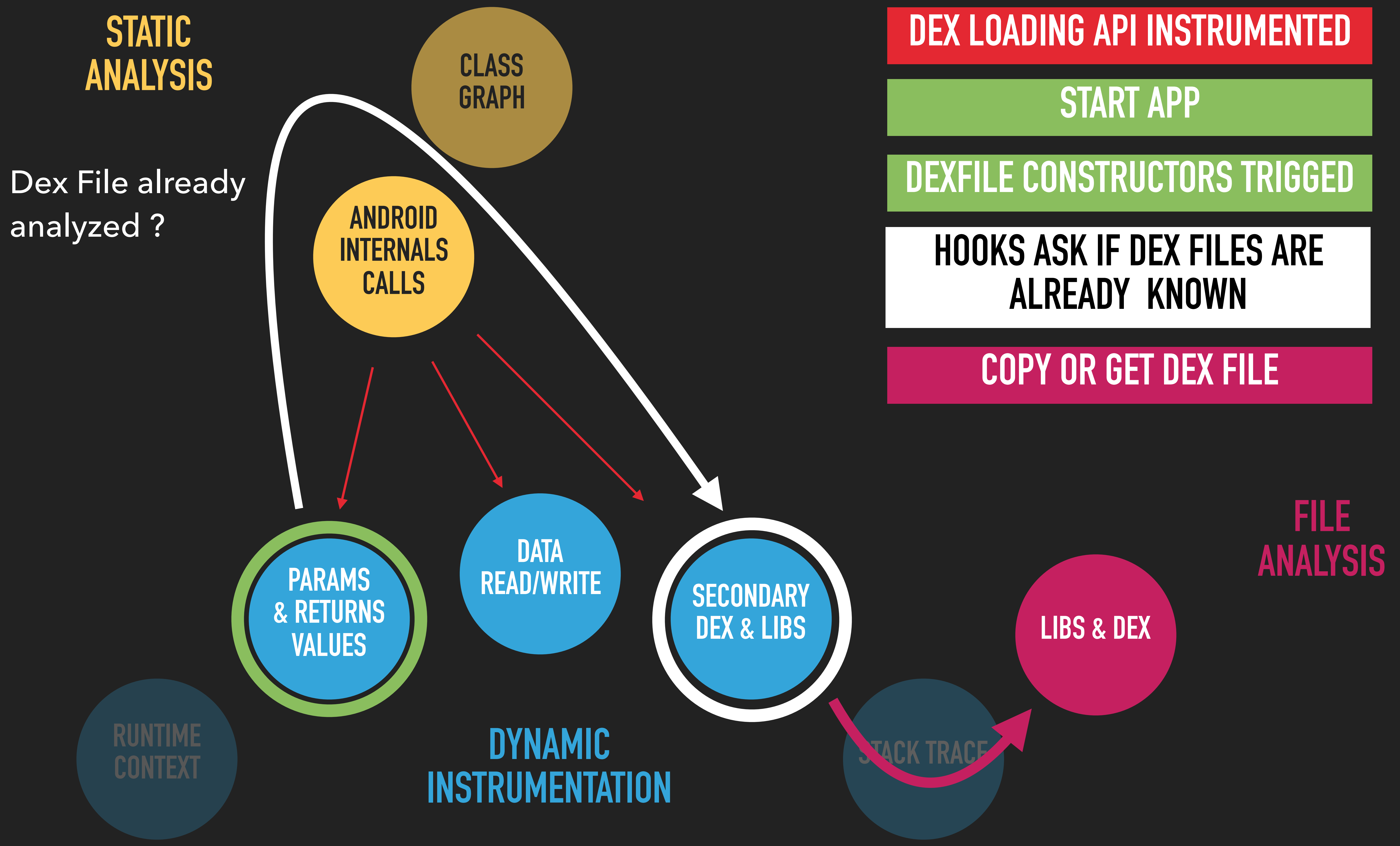
DEX LOADING API INSTRUMENTED

START APP

DEXFILE CONSTRUCTORS TRIGGED

HOOKS ASK IF DEX FILES ARE ALREADY KNOWN

ANALYZE DEX FILE LOADED DYNAMICALLY



DEX LOADING API INSTRUMENTED

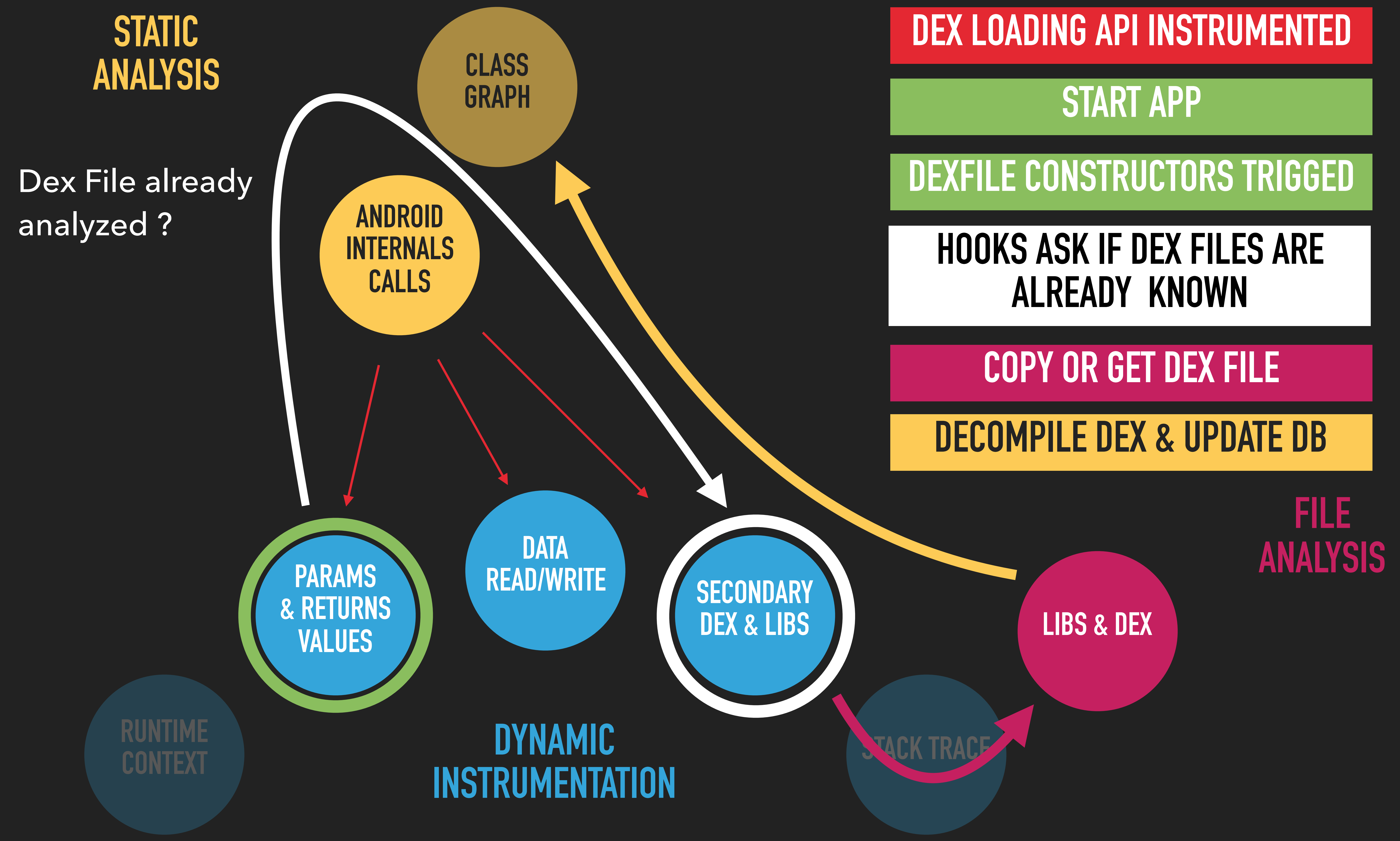
START APP

DEXFILE CONSTRUCTORS TRIGGED

HOOKS ASK IF DEX FILES ARE ALREADY KNOWN

COPY OR GET DEX FILE

ANALYZE DEX FILE LOADED DYNAMICALLY



DEX LOADING API INSTRUMENTED

START APP

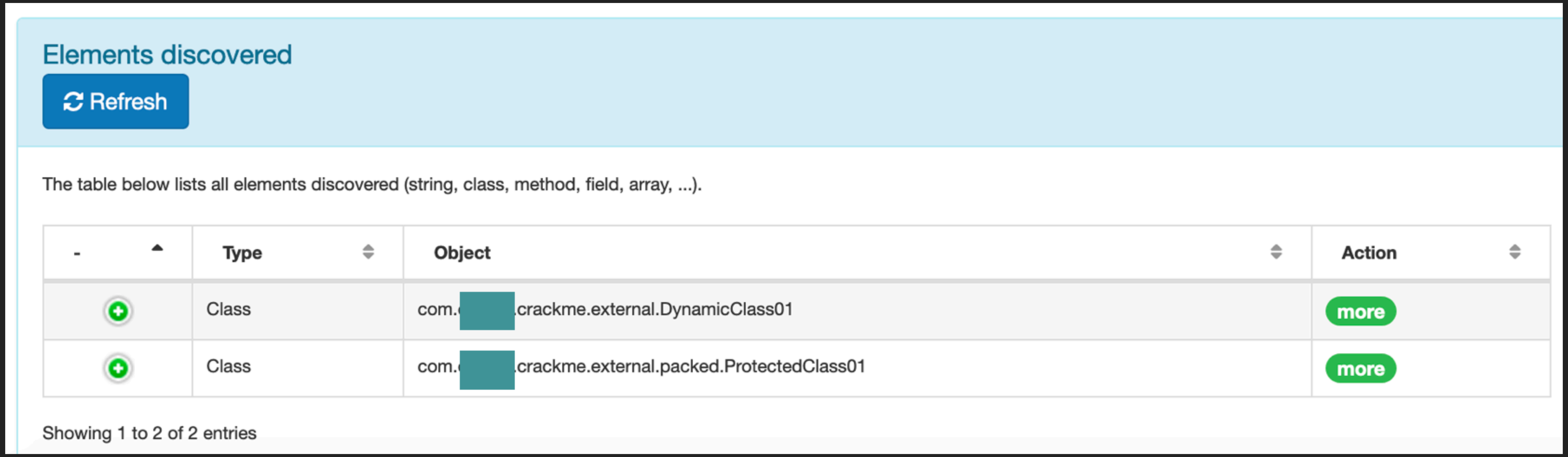
DEXFILE CONSTRUCTORS TRIGGED

HOOKS ASK IF DEX FILES ARE ALREADY KNOWN

COPY OR GET DEX FILE

DECOMPILE DEX & UPDATE DB

ANALYZE DEX FILE LOADED DYNAMICALLY



Elements discovered

The table below lists all elements discovered (string, class, method, field, array, ...).

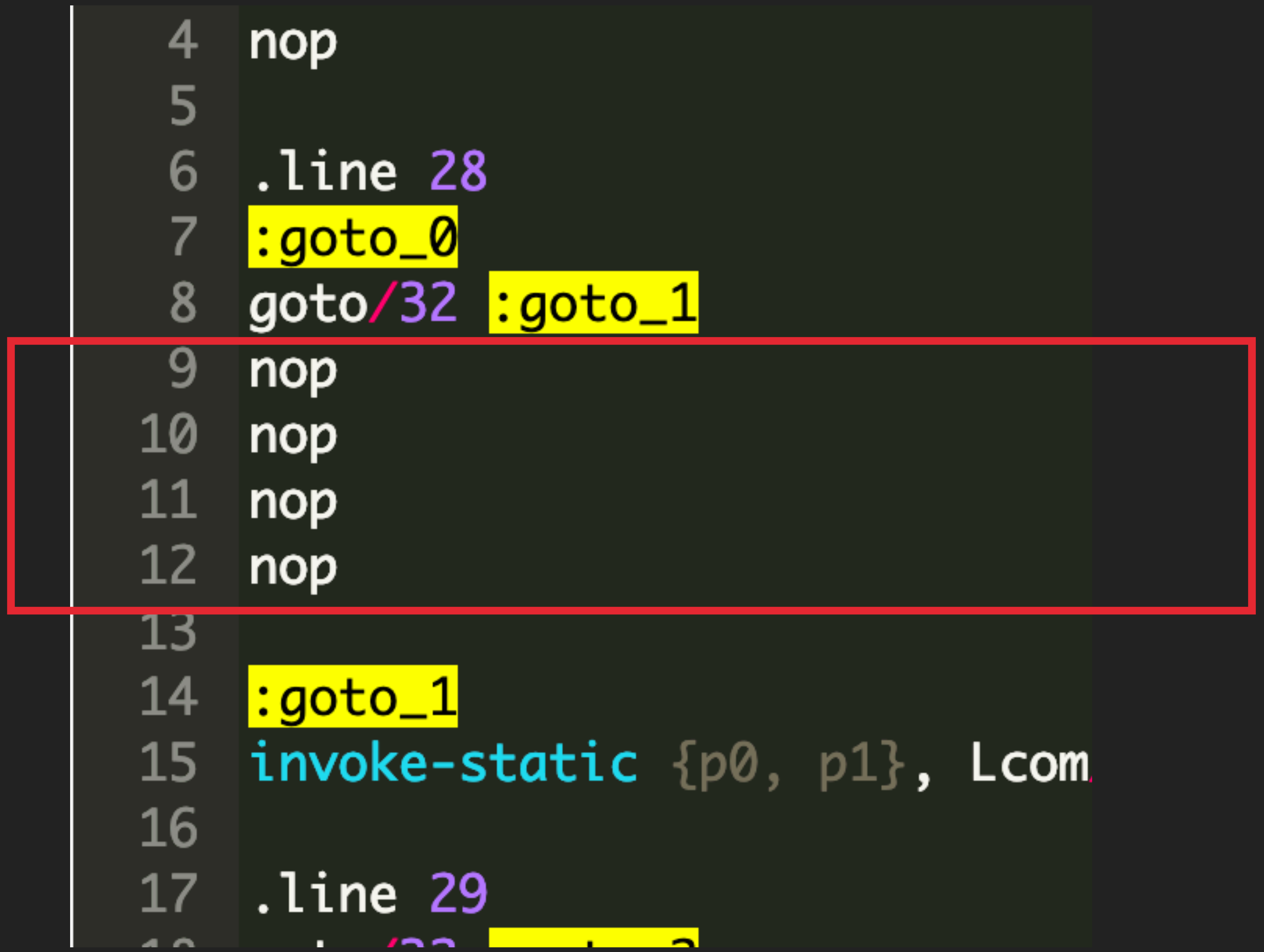
-	Type	Object	Action
⊕	Class	com.[redacted].crackme.external.DynamicClass01	more
⊕	Class	com.[redacted].crackme.external.packed.ProtectedClass01	more

Showing 1 to 2 of 2 entries

CASE #3

BYTECODE CLEANER

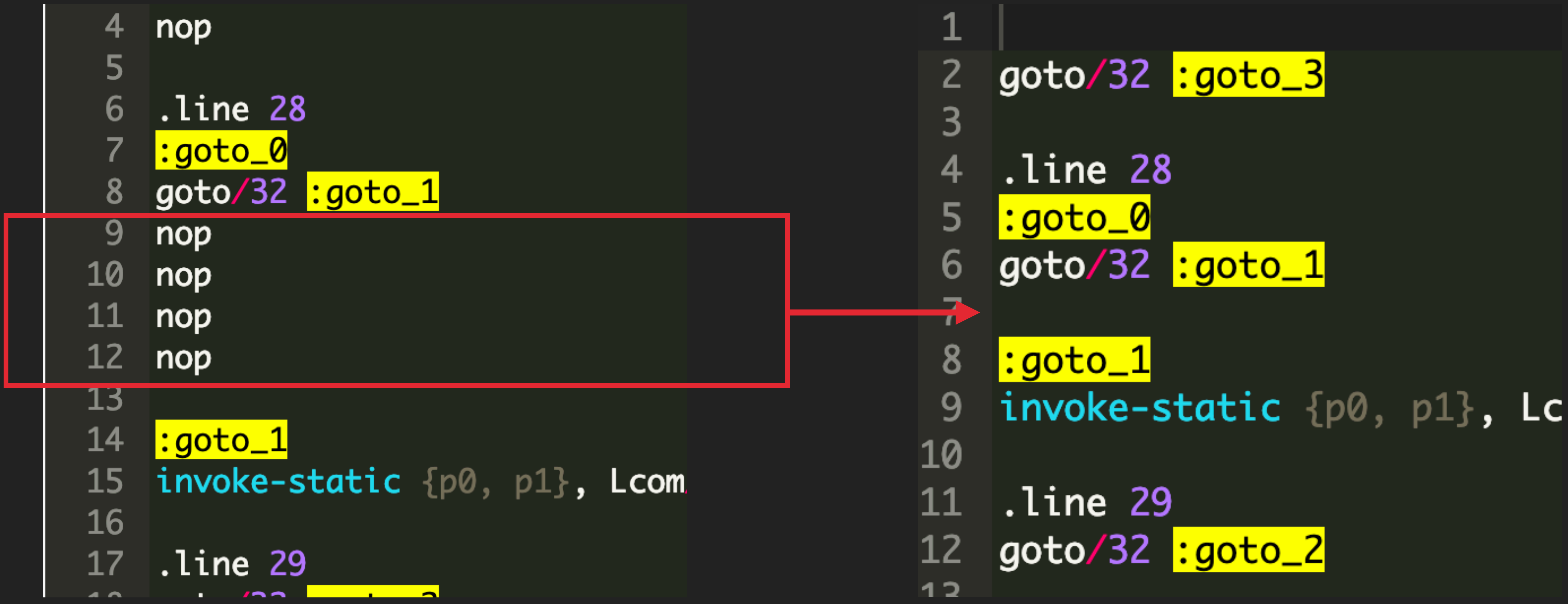
BYTE CODE CLEANER : REMOVE NOP



```
nop

.line 28
:goto_0
goto/32 :goto_1
nop
nop
nop
nop

:goto_1
invoke-static {p0, p1}, Lcom

.line 29
```

BEFORE

BYTE CODE CLEANER : REMOVE NOP



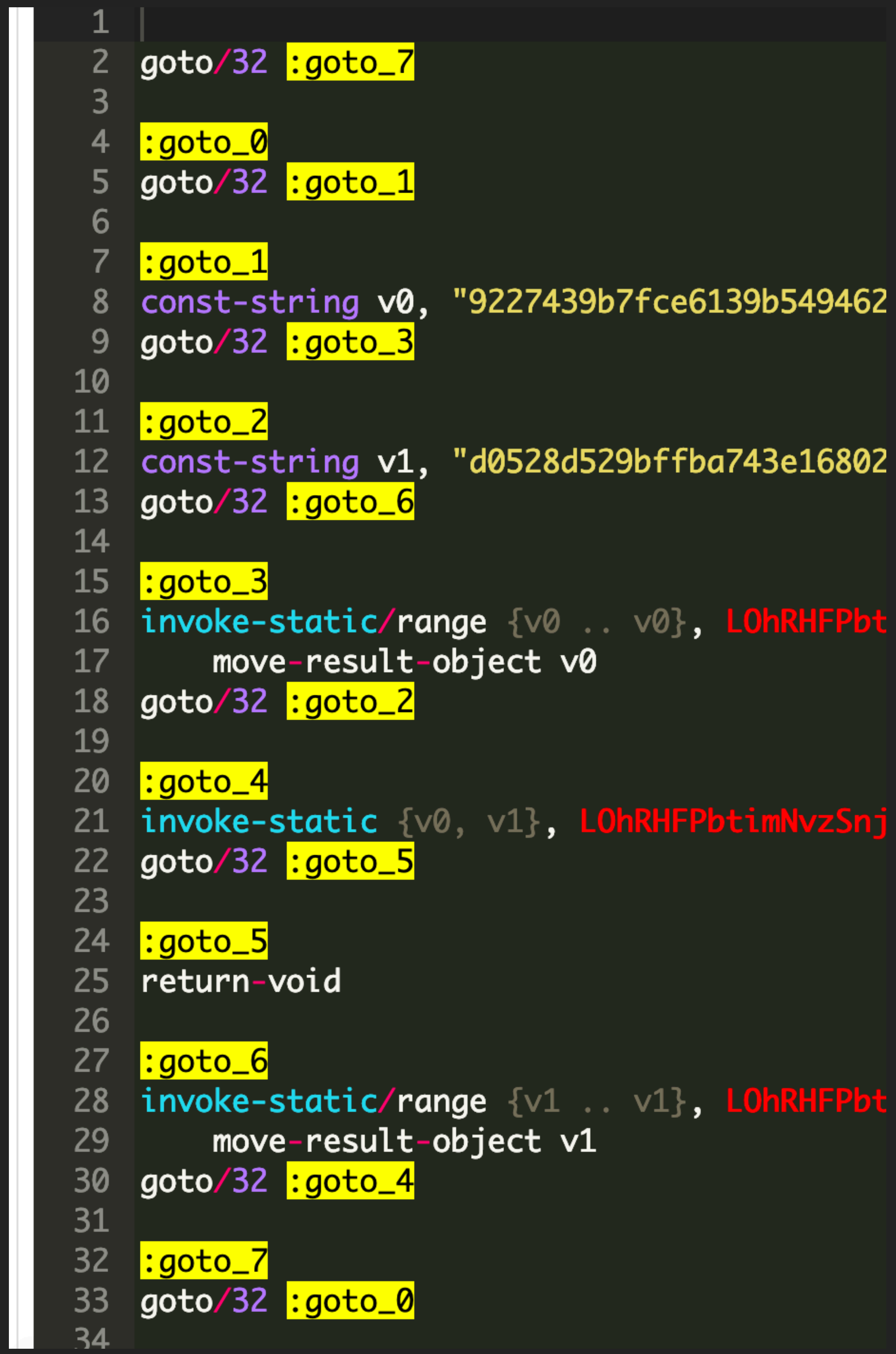
```
nop

.line 28
:goto_0
goto/32 :goto_1
nop
nop
nop
nop

:goto_1
invoke-static {p0, p1}, Lcom

.line 29
```

BEFORE

```

goto/32 :goto_3

.line 28
:goto_0
goto/32 :goto_1

:goto_1
invoke-static {p0, p1}, Lc

.line 29
goto/32 :goto_2
```

AFTER

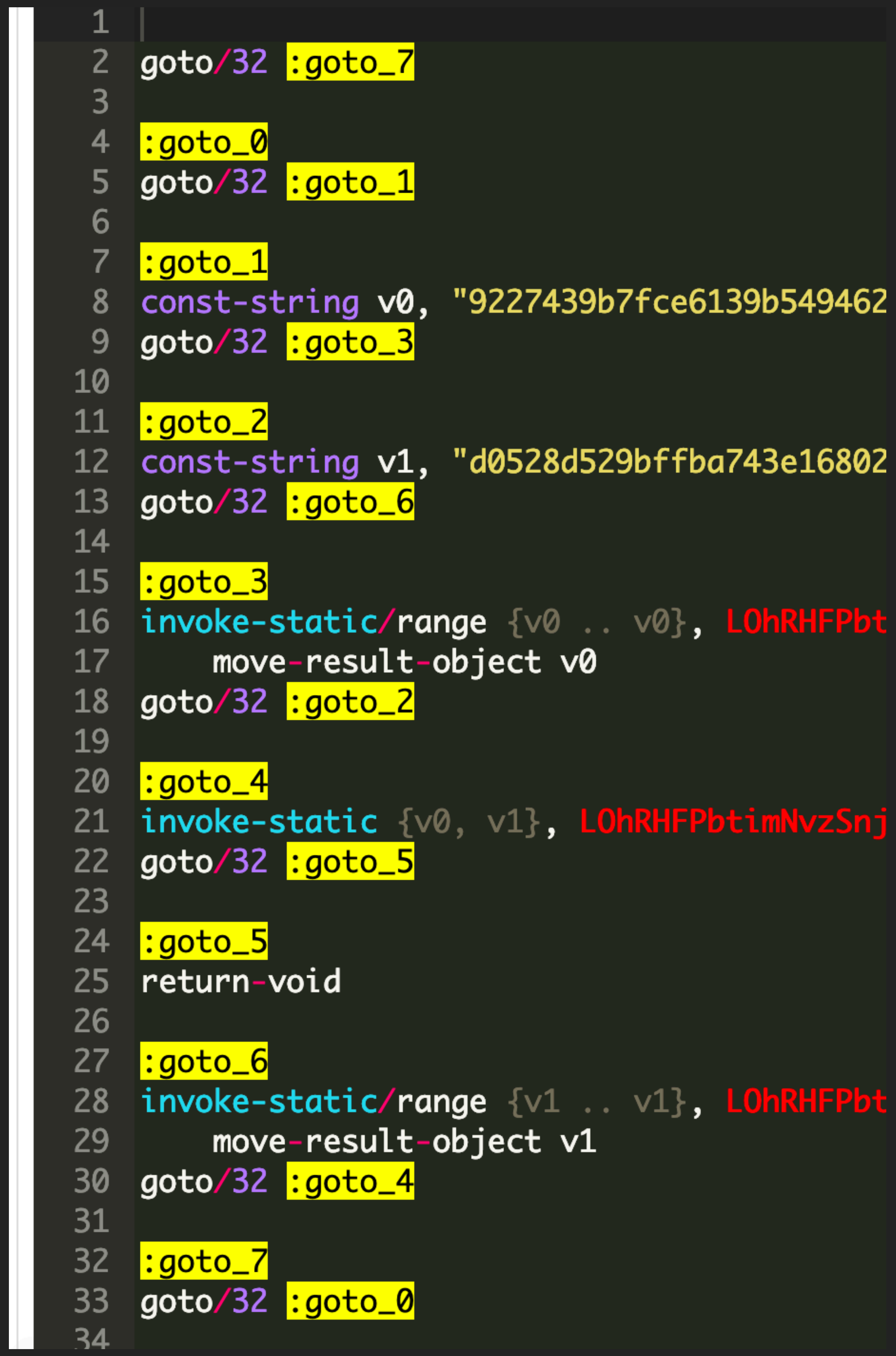
REMOVE USELESS GOTO



```

goto/32 :goto_7

:goto_0
goto/32 :goto_1

:goto_1
const-string v0, "9227439b7fce6139b549462
goto/32 :goto_3

:goto_2
const-string v1, "d0528d529bffba743e16802
goto/32 :goto_6

:goto_3
invoke-static/range {v0 .. v0}, LOhRHFPbt
    move-result-object v0
goto/32 :goto_2

:goto_4
invoke-static {v0, v1}, LOhRHFPbtimNvzSnj
goto/32 :goto_5

:goto_5
return-void

:goto_6
invoke-static/range {v1 .. v1}, LOhRHFPbt
    move-result-object v1
goto/32 :goto_4

:goto_7
goto/32 :goto_0

```

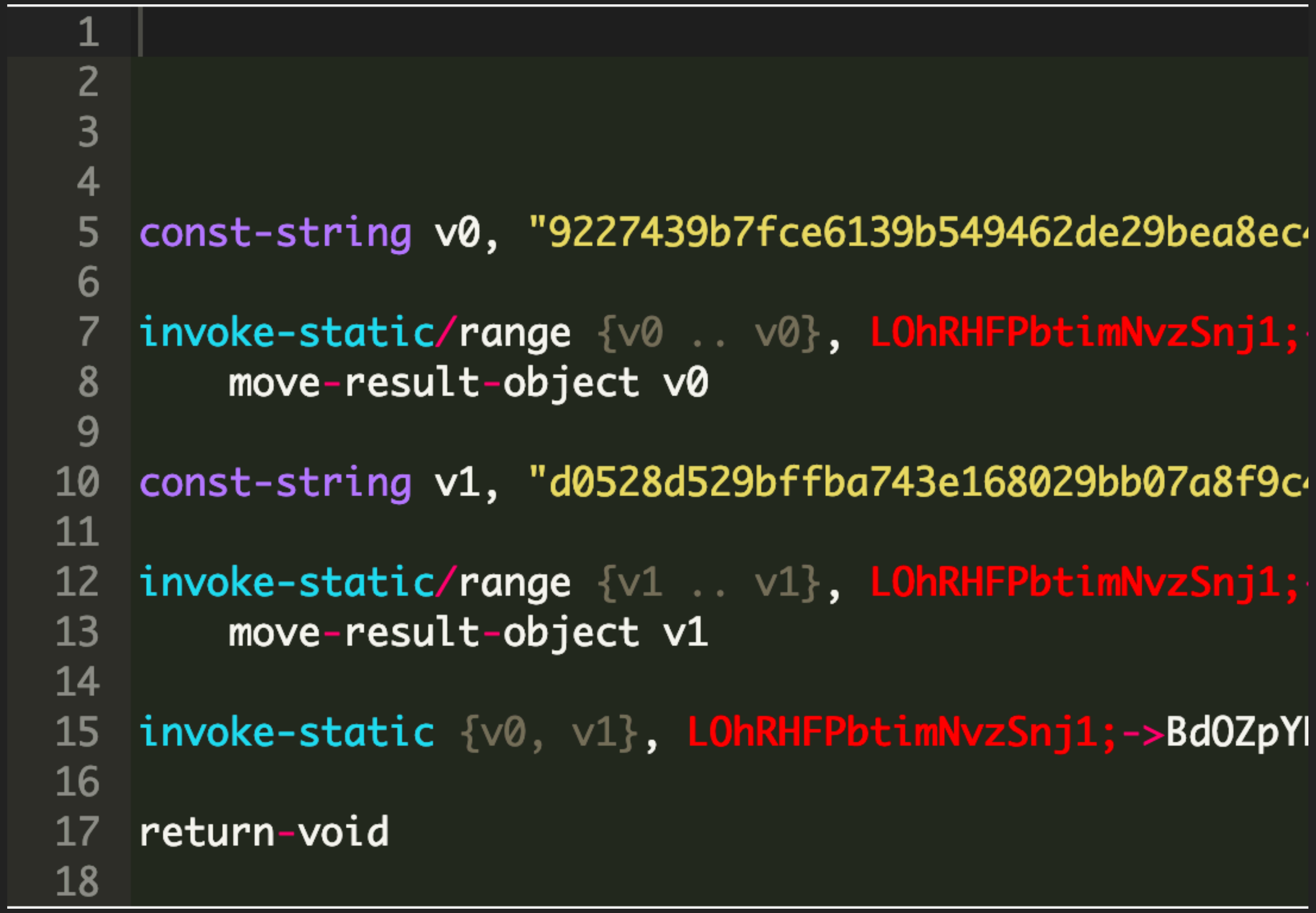
BEFORE

REMOVE USELESS GOTO



```
goto/32 :goto_7

:goto_0
goto/32 :goto_1

:goto_1
const-string v0, "9227439b7fce6139b549462
goto/32 :goto_3

:goto_2
const-string v1, "d0528d529bffba743e16802
goto/32 :goto_6

:goto_3
invoke-static/range {v0 .. v0}, LOhRHFPbt
    move-result-object v0
goto/32 :goto_2

:goto_4
invoke-static {v0, v1}, LOhRHFPbtimNvzSnj
goto/32 :goto_5

:goto_5
return-void

:goto_6
invoke-static/range {v1 .. v1}, LOhRHFPbt
    move-result-object v1
goto/32 :goto_4

:goto_7
goto/32 :goto_0
```

BEFORE



```
const-string v0, "9227439b7fce6139b549462de29bea8ec

invoke-static/range {v0 .. v0}, LOhRHFPbtimNvzSnj1;
    move-result-object v0

const-string v1, "d0528d529bffba743e168029bb07a8f9c

invoke-static/range {v1 .. v1}, LOhRHFPbtimNvzSnj1;
    move-result-object v1

invoke-static {v0, v1}, LOhRHFPbtimNvzSnj1;->BdOZpY

return-void
```

AFTER

IMPROVEMENTS

- Use my own customizable Dex Decompiler (or use LIEF)?
- Add r2 binding and native hooks
- HTTP communications & Intent grabbing
- Bytecode & native symbolic exec (Z3) ?
- Bytecode emulation (SmaliVM @CalebFenton)?
- Offers native instruction hooking (QBDI)?
- And fuzz (afl-fuzz params + feedback given by hooking)?

Thanks

Q&A

ANNEXES

HOW TO INSTALL ?

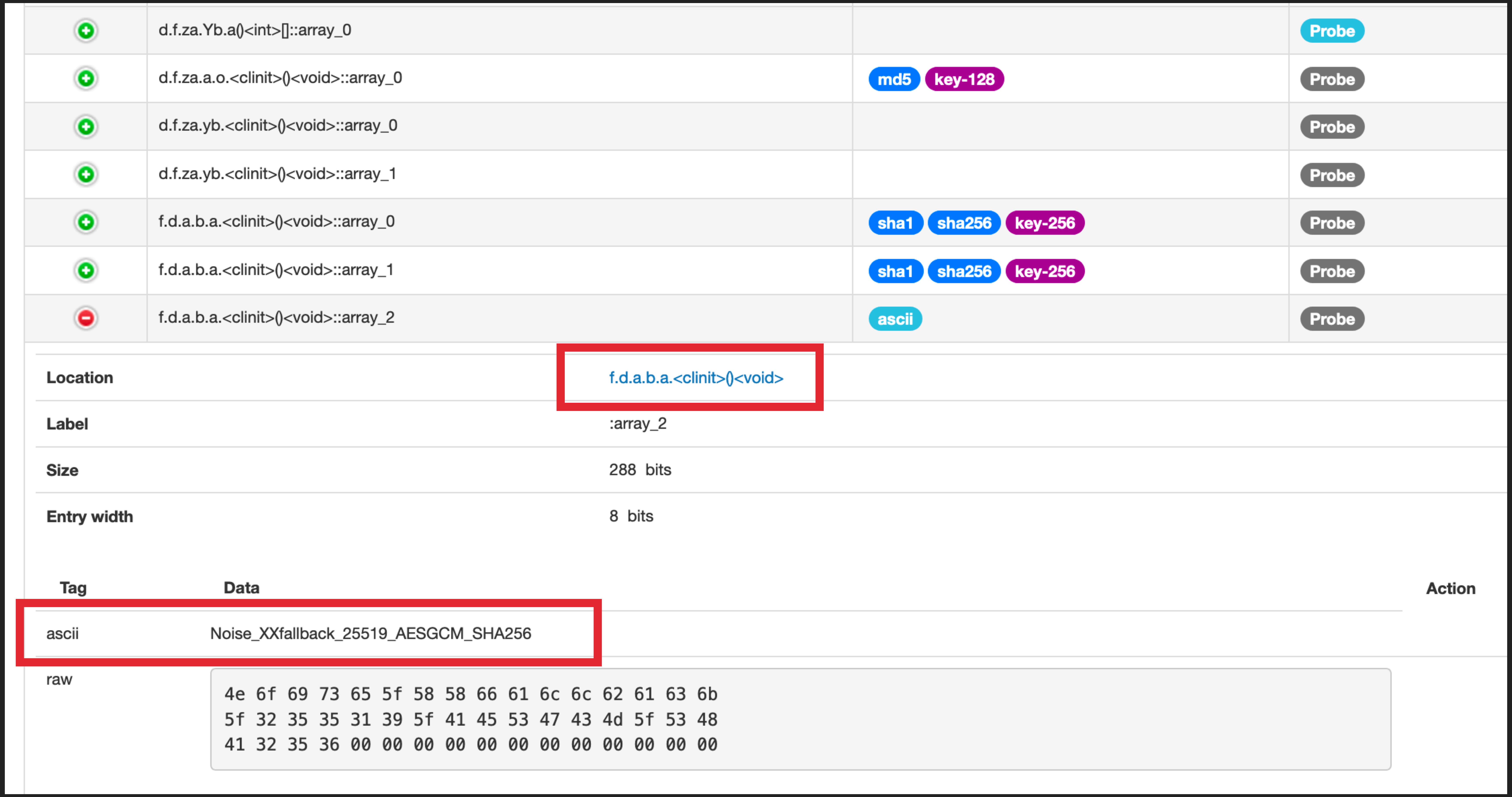
- Ensure you have the requirements (Frida, NodeJS, apktool)

```
git clone https://github.com/FrenchYeti/dexcalibur.git
cd dexcalibur
npm install
```

- Or install from DockerHub

```
docker pull frenchyeti/dexcalibur
docker run -it \
   -v <workspace>:/home/dexcalibur/workspace \
   -p 8080:8000 —dev=<device> \
   frenchyeti/dexcalibur
```

SEARCH BYTE ARRAY



⊕	d.f.za.Yb.a()<int>[]::array_0		Probe
⊕	d.f.za.a.o.<clinit>()<void>::array_0	md5 key-128	Probe
⊕	d.f.za.yb.<clinit>()<void>::array_0		Probe
⊕	d.f.za.yb.<clinit>()<void>::array_1		Probe
⊕	f.d.a.b.a.<clinit>()<void>::array_0	sha1 sha256 key-256	Probe
⊕	f.d.a.b.a.<clinit>()<void>::array_1	sha1 sha256 key-256	Probe
⊖	f.d.a.b.a.<clinit>()<void>::array_2	ascii	Probe

Location	f.d.a.b.a.<clinit>()<void>
Label	:array_2
Size	288 bits
Entry width	8 bits

Tag	**Data**	**Action**
ascii	Noise_XXfallback_25519_AESGCM_SHA256	
raw		

```
4e 6f 69 73 65 5f 58 58 66 61 6c 6c 62 61 63 6b
5f 32 35 35 31 39 5f 41 45 53 47 43 4d 5f 53 48
41 32 35 36 00 00 00 00 00 00 00 00 00 00 00 00
```