静态分析：

不运行代码的情况下，阅读反汇编代码掌握程序功能

两种方法：

1. 阅读 Dalvik 字节码(通过 baksmali 反编译 dex 文件生成 samli 文件)
2. 阅读 java 代码(通过 dex2jar 生成 jar 文件，在 jd-gui 阅读 jar 文件)

常用步骤：

1. 反编译 apk
2. 通过 AndroidManifest.xml 查找主 Activity

```
<activity android:label="@string/title_activity_main" android:name=".MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

标题 类名 主Activity 通过该Activity启动

3. 查看程序的入口函数：主 Activity 的 onCreate()
4. 查看 Application 类(全局类，早于其他类启动)的 onCreate()函数，该函数通常用作授权检测

定位关键代码常用方法：

1. 信息反馈发：运行时信息
2. 特征函数法：运行时行为
3. 顺序查看法：执行流程
4. 代码注入法：添加 Log

Smali 代码格式：

MainActivity.smali

```
.class public Lcom/droider/crackme0502/MainActivity;
.super Landroid/app/Activity;
.source "MainActivity.java"


# instance fields
.field private btnAnno:Landroid/widget/Button;

.field private btnCheckSN:Landroid/widget/Button;

.field private edtSN:Landroid/widget/EditText;


# direct methods
.method public constructor <init>()V
    .locals 0

    .prologue
    .line 19
    invoke-direct {p0}, Landroid/app/Activity;-><init>()V

    return-void
.end method
```

内部类的表示：

MainActivity$1.smali：匿名内部类，多用于程序中的响应

MainActivity$SNChecker.smali：成员内部类

MainActivity.smali：外部类

this$0 是内部类自动保留的一个指向所在外部类的引用。this 表示父类的引用，右边的 0 便是引用的层数，例如：ThirdInner 是 this

```
public class Outer{            //this$0
    public class FirstInner{       //this$1
        public class SecondInner{  //this$2
            public class ThirdInner{
        }
        }
    }
}
```

this$X 型字段都被指定了 synthetic(合成的、编译器生成的)属性，表明他们是被编译器合成的、虚构的、非 java 代码指定的字段。

构造函数执行步骤：

1. 保存外部类的引用到本类的一个 synthetic 字段中
2. 调用内部类的父类构造函数
3. 内部类初始化

```
.class public Lcom/droider/crackme0502/MainActivity$SNChecker;
.super Ljava/lang/Object;
.source "MainActivity.java"


# instance fields
.field private sn:Ljava/lang/String;

.field final synthetic this$0:Lcom/droider/crackme0502/MainActivity;


# direct methods
.method public constructor <init>(Lcom/droider/crackme0502/MainActivity;Ljava/lang/String;)V
    .locals 0
    .param p2, "sn"    # Ljava/lang/String;

    .prologue
    .line 83
    #将外部类引用赋给p1
    iput-object p1, p0, Lcom/droider/crackme0502/MainActivity$SNChecker;->this$0:Lcom/droider/crackme0502/MainActivity;

    #调用SNCheck的基类Object的构造函数
    invoke-direct {p0}, Ljava/lang/Object;-><init>()V

    .line 84
    #调用SNCheck自身的构造函数
    iput-object p2, p0, Lcom/droider/crackme0502/MainActivity$SNChecker;->sn:Ljava/lang/String;

    .line 85
    return-void
.end method
```