Android 动态逆向分析工具（四）——

Andbug 补充调试功能

anbingchun@163.com

新增了函数内设置断点，已经单步调试相关的功能。

1、在函数内部设置断点

1.1、定位要分析的函数

通过 classes 和 methods 命令，定位到要分析的函数，这里以 com.example.test.MainActivity 为例，可以对该函数的入口设置断点。

```
>> classes example
## Loaded Classes
   -- com.example.test.MainActivity$1
   -- com.example.test.MainActivity
>> method com.example.test.MainActivity
!! command not supported: "method."
>> methods com.example.test.MainActivity
## Methods Lcom/example/test/MainActivity;
   -- com.example.test.MainActivity.<clinit>()V
   -- com.example.test.MainActivity.<init>()V
   -- com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V
   -- com.example.test.MainActivity.onCreateOptionsMenu(Landroid/view/Menu;)Z
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I
   -- com.example.test.MainActivity.testConnectivityManager()V
>> break com.example.test.MainActivity onCreate
## Setting Hooks
   -- Hooked <536870924> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
>> break-list
## Active Hooks
   -- Hook <536870924> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
```

1.2、查看函数部的代码分布情况

通过 break com.example.test.MainActivity onCreate show 命令可以看到函数内各代码的

```
>> break com.example.test.MainActivity onCreate show
## Setting Hooks
   -- [24L, 25L, 27L, 29L, 31L, 46L]
```

与之对应的源代码为：

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        testConnectivityManager();

        Button btn = (Button)findViewById(R.id.test);
        //绑定匿名的监听器，并执行您所要在点击按钮后执行的逻辑代码
        btn.setOnClickListener(new View.OnClickListener()
        {
            public void onClick(View arg0)
            {
                Toast.makeText(MainActivity.this, "点击了按钮", Toast.LENGTH_LONG).show();
                byte tt[] = new byte[10];
                test("test", 20, 25L, MainActivity.this, tt);
                String a= new String(tt);
                Log.i("aaa", a);
            }
        });


    }
```

其中代码“super.onCreate(savedInstanceState);”位于第 24 行，onCreate 函数的最后一个大括号位于 46 行。
通过 show 我们可以得知，onCreate 中的代码分别位于第 24、25、27、29、31、46 行。
这里支持以源代码代码行粒度的断点设置，还不支持以 dilvik 汇编指令粒度的进行断点设置。

1.3、设置函数内的断点

如图所示，分别在 27、24、25 行设置了断点。

```
>> break-list
## Active Hooks
   -- Hook <536870924> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
>> break com.example.test.MainActivity onCreate show
## Setting Hooks
   -- [24L, 25L, 27L, 29L, 31L, 46L]
>> break com.example.test.MainActivity onCreate 27
## Setting Hooks
   -- Hooked <536870925> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:8 <class
      'andbug.vm.Location'>
>> break-list
## Active Hooks
   -- Hook <536870924> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
   -- Hook <536870925> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:8 <class
      'andbug.vm.Location'>
>> break com.example.test.MainActivity onCreate 24
## Setting Hooks
   -- Hooked <536870926> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
>> break com.example.test.MainActivity onCreate 25
## Setting Hooks
   -- Hooked <536870927> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:3 <class
      'andbug.vm.Location'>
>>
```

设置断点如下图所示，每个断点中的，0、8、0、3 是指设置断点位置相对与函数入口点，davlik 汇编指令的偏移位置。

```
>>
>> break-list
## Active Hooks
   -- Hook <536870924> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
   -- Hook <536870925> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:8 <class
      'andbug.vm.Location'>
   -- Hook <536870926> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:0 <class
      'andbug.vm.Location'>
   -- Hook <536870927> com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V:3 <class
      'andbug.vm.Location'>
>>
```

注：目前暂时不支持，以 dalvik 汇编指令粒度的断点设置

1.4、断点的出发

对函数 com.example.test.MainActivity.test 中的 65 行处设置断点，出发后如图所示。

```
>> break com.example.test.MainActivity test show
## Setting Hooks
   -- [50L, 51L, 52L, 53L, 54L, 56L, 58L, 62L, 63L, 64L, 65L, 66L, 67L, 68L, 70L]
>> break com.example.test.MainActivity test 65
## Setting Hooks
   -- Hooked <536870912>
      com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:3
      8 <class 'andbug.vm.Location'>
>> ## Breakpoint hit in thread <1> main    (running suspended), process suspended.
   --
      com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:3
      8
   -- com.example.test.MainActivity$1.onClick(Landroid/view/View;)V:26
   -- android.view.View.performClick()Z:18
   -- android.view.View$PerformClick.run()V:2
   -- android.os.Handler.handleCallback(Landroid/os/Message;)V:2
   -- android.os.Handler.dispatchMessage(Landroid/os/Message;)V:4
   -- android.os.Looper.loop()V:84
   -- android.app.ActivityThread.main([Ljava/lang/String;)V:56
   -- java.lang.reflect.Method.invokeNative(Ljava/lang/Object;[Ljava/lang/Object;Ljava/lang/Class;
      [Ljava/lang/Class;Ljava/lang/Class;IZ)Ljava/lang/Object; <native>
   -- java.lang.reflect.Method.invoke(Ljava/lang/Object;[Ljava/lang/Object;)Ljava/lang/Object;:17
   -- com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run()V:11
   -- com.android.internal.os.ZygoteInit.main([Ljava/lang/String;)V:70
   -- dalvik.system.NativeStart.main([Ljava/lang/String;)V <native>
```

2、单步调试功能

该部分代码为 FreedomCoder 实现，但一直没有被 swdunlop 合并进主干，为实习下面功能，作者将 FreedomCoder 的部分代码合并进了作者的 git 代码。

2.1 运行应用，并将应用停在设定位置

如下图所示：
加载调试目标，并分别在 test 函数的入口和 65 行代码处设置断点。

点击实际上的某个按钮，触发 test 函数，apk 运行中断在 test 函数入口处。

```
anbc@anbc-OptiPlex-780:~/test/do/Andbug_Freedom/AndBug$ PYTHONPATH=lib ./andbug  shell -p com.exam
ple.test

## AndBug (C) 2011 Scott W. Dunlop <swdunlop@gmail.com>
>> break com.example.test.MainActivity test
## Setting Hooks
   -- Hooked <536870913>
      com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:0
      : smali line: 50 <class 'andbug.vm.Location'>
>> break-list
## Active Hooks
   -- Hook <536870913> com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:0: smali line:
      50 <class 'andbug.vm.Location'>
>> break com.example.test.MainActivity test 65
## Setting Hooks
   -- Hooked <536870914> com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:38: smali
      line: 65 <class 'andbug.vm.Location'>
>> 536870913
## Breakpoint hit in thread <1> main    (running suspended), process suspended.
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:0: smali line: 50
   -- com.example.test.MainActivity$1.onClick(Landroid/view/View;)V:26
   -- android.view.View.performClick()Z:18
   -- android.view.View$PerformClick.run()V:2
   -- android.os.Handler.handleCallback(Landroid/os/Message;)V:2
   -- android.os.Handler.dispatchMessage(Landroid/os/Message;)V:4
   -- android.os.Looper.loop()V:84
   -- android.app.ActivityThread.main([Ljava/lang/String;)V:56
   -- java.lang.reflect.Method.invokeNative(Ljava/lang/Object;[Ljava/lang/Object;Ljava/lang/Class;[Ljava/lang/Class;Ljava/lang/Cl
      ass;IZ)Ljava/lang/Object; <native>
   -- java.lang.reflect.Method.invoke(Ljava/lang/Object;[Ljava/lang/Object;)Ljava/lang/Object;:17

>> s
## Step Over
```

2.2、step over 功能

执行下一步指令，使用命令：step over 命令。
如下图分别与行了第 51 行和 52 行的代码。
命令的全称是“stepover”，缩写是“s”

```
>> s
## Step Over
>> 536870915
## Single step complete in thread <1> main      (running suspended), suspended.
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:2: smali line: 51

>>
>> s
## Step Over
>> 536870916
## Single step complete in thread <1> main      (running suspended), suspended.
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:4: smali line: 52
```

2.3、step into 功能

进入子函数的单步执行命令“stepinto”，缩写为“si”
由于本例中，该代码处，不是一个函数调用，所以执行效果等同于“step over”

```
>> stepinto
## Step Into
>> 536870917
## Single step complete in thread <1> main      (running suspended), suspended.
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I:6: smali line: 53
```

单步执行后，代码停到了 53 行处。

2.4、stepout 功能

通过该功能可以实现，从当前函数中跳出来，停止在其父函数位置。

```
>>
>> stepout
## Step Out
>> 536870914
## Single step complete in thread <1> main      (running suspended), suspended.
   -- com.example.test.MainActivity$1.onClick(Landroid/view/View;)V:29
```

其父函数是“com.example.test.MainActivity$1.onClick”

下面为 test 函数的源码

```
public int test(String sArg, int iArg, long lArg, MainActivity activityArg, byte tt[])
{
    String sVar = "111111";
    int iVar = 30;
    long lVar = 5L;
    double dVar = 3.5;
    MainActivity mainActivityVar = activityArg;

    for (byte i=0; i<10; i++)
    {
        tt[i]='e';

    }

    Toast.makeText(MainActivity.this, "you test", Toast.LENGTH_LONG).show();
    Log.i("aaa", "111");
    Log.i("aaa", "112");
    Log.i("aaa", "113");
    Log.i("aaa", "114");
    Log.i("aaa", "115");
    Log.i("aaa", "116");

    return 22;

}
```

不支持的功能：

1、以 dalvik 汇编指令粒度的断点设置
2、Show 展示断点设置位置时，只显示数字行号偏移，不显示对应的代码，不利于操作。
3、断点与汇编或源码的关联功能没有实现，
4、bug 带$的类名会错。

```
   -- com.example.test.MainActivity.onCreateOptionsMenu(Landroid/view/Menu;)Z
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I
   -- com.example.test.MainActivity.testConnectivityManager()V
>> methods "com.example.test.MainActivity$1"
## Methods Lcom/example/test/MainActivity$1;
## TypeError: 'NoneType' object is not iterable
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/command.py", line 152, in perform
      act(self, *args, **kwargs)
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/cmd/methods.py", line 25, in
      methods
      for m in ctxt.sess.classes(cpath).methods(name=mname, jni=mjni):
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/data.py", line 83, in poolcall
      getattr(item, key)(*args, **kwargs) for item in self.items
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/data.py", line 83, in <genexpr>
      getattr(item, key)(*args, **kwargs) for item in self.items
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/vm.py", line 645, in methods
      return andbug.data.view(seq)
   -- File "/home/anbc/test/do/Andbug_swdunlop/AndBug/lib/andbug/data.py", line 71, in __init__
      self.items = list(items)
>> methods "com.example.test.MainActivity"
## Methods Lcom/example/test/MainActivity;
   -- com.example.test.MainActivity.<clinit>()V
   -- com.example.test.MainActivity.<init>()V
   -- com.example.test.MainActivity.onCreate(Landroid/os/Bundle;)V
   -- com.example.test.MainActivity.onCreateOptionsMenu(Landroid/view/Menu;)Z
   -- com.example.test.MainActivity.test(Ljava/lang/String;IJLcom/example/test/MainActivity;[B)I
   -- com.example.test.MainActivity.testConnectivityManager()V
>> br
```