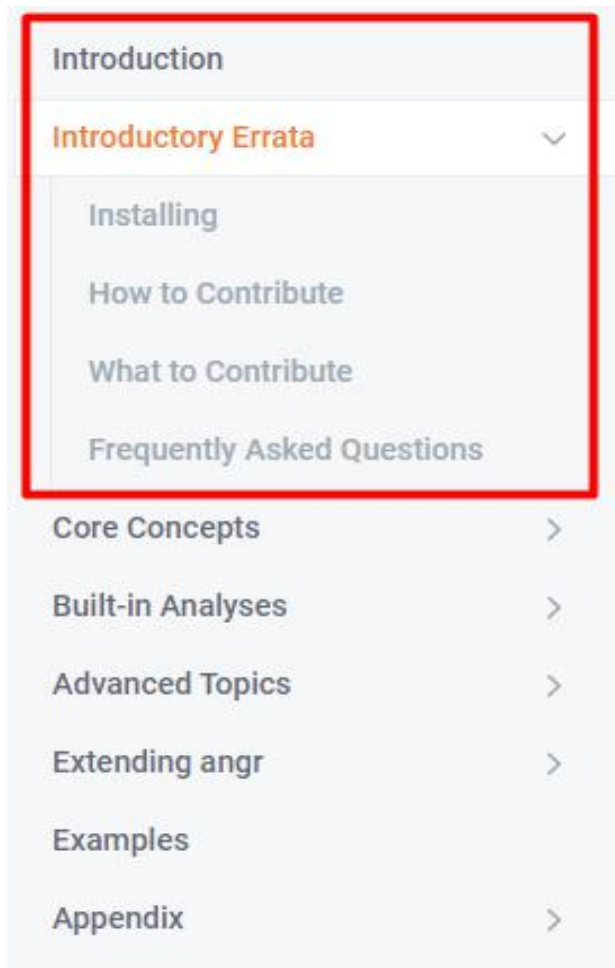


# Angr 文档介绍

刘凯

# Angr模块

分类	内容	说明
核心功能	符号执行,	参考文档和源码实现
库函数处理	SimProcedures	不全面, 且部分需要修复
搜索策略	dfs bfs tracer veritesting	方便辅助实现相关论文思路
程序表示	cfg cdg ddg vfg	cfg最基础, 生成与遍历方法
程序分析	Backward Slicing Forword Slicing Function Identifier Bindiff Disassembly ...	种类繁多, 官方项目会持续更新



# angr文档

<https://docs.angr.io/>

## ◆ 前两章

一. 介绍（引言）

二. 介绍性勘误

a) 安装

b) 如何贡献

c) 贡献什么

d) 经常问的问题

# 一、引言

## angr是什么

angr是一个多架构的二进制分析平台，具备对二进制文件的动态符号执行能力（例如Mayhem，KLEE等）和多种静态分析能力。

大概看来，要做到这些必须要克服一些问题：

- 装载二进制文件到分析平台
- 转换二进制文件为中间语言（intermediate representation）（IR）
- 转换IR为语义描述（即它做什么而不是它是什么）
- 执行真正的分析，这包括：
  - 部分或者全部的静态分析（即依赖分析，程序分片）
  - 对程序状态空间的符号探索（比如“我们能否一直执行它直到我们找到了一个溢出？”）
  - 对上述的情况的一些混合（比如“让我们执行一部分可导致内存写的内存切片，来发现一个溢出”）

angr拥有的组件能满足所有这些挑战。这本书将会向你解释每一个组件是如何工作的，以及如何使用它们来完成你的邪恶目标（原文：accomplish your evil goals）。

**程序分片：**主要应用在软件维护、程序调试、测试以及逆向工程等领域。

**出现：**在程序调试时，当错误出现时，我们希望通过一种手段，找到可能产生该错误的源代码部分。我们或许不能对这样的源代码准确定位，但通过剔除不可能产生该错误的部分，把可疑部分限制在一个较小的范围中。程序分片就是一种实现上述目的的技术。

**分类：**

1. 静态分片：以程序的静态信息为依据，使用数据流和控制流分析
2. 动态分片：依赖于程序的具体输入，每次结果不同
3. 条件分片：施加（去除）一定条件的动态分片

# 一、引言

## 引用angr

如果你想要在学术工作中使用angr，请引用这些论文：

```
@article{shoshitaishvili2016state,  
  title={SoK: (State of) The Art of War: Offensive Techniques in Binary Analysis},  
  author={Shoshitaishvili, Yan and Wang, Ruoyu and Salls, Christopher and Stephens, Nick and Polino, Mario and Dutcher},  
  booktitle={IEEE Symposium on Security and Privacy},  
  year={2016}  
}  
  
@article{stephens2016driller,  
  title={Driller: Augmenting Fuzzing Through Selective Symbolic Execution},  
  author={Stephens, Nick and Grosen, John and Salls, Christopher and Dutcher, Andrew and Wang, Ruoyu and Corbetta, Jac},  
  booktitle={NDSS},  
  year={2016}  
}  
  
@article{shoshitaishvili2015firmalice,  
  title={Firmalice - Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware},  
  author={Shoshitaishvili, Yan and Wang, Ruoyu and Hauser, Christophe and Kruegel, Christopher and Vigna, Giovanni},  
  booktitle={NDSS},  
  year={2015}  
}
```

# 一、引言

## 支持

---

如果想要获取关于angr的帮助，你可以通过：

- 邮件: [angr@lists.cs.ucsb.edu](mailto:angr@lists.cs.ucsb.edu)
- the slack channel: [angr.slack.com](https://angr.slack.com), for which you can get an account [here](#).
- 在github对应仓库上打开一个issue

## 二. 介绍性勘误

a) 安装

b) 如何贡献

c) 贡献什么

d) 经常问的问题



## ◆ 如何安装？

推荐使用python虚拟环境virtualenvwrapper来安装和使用angr

### 1. 安装依赖 (针对Linux)

```
sudo apt-get install python-dev libffi-dev build-essential virtualenvwrapper
```

### 2. 安装angr

```
mkvirtualenv angr && pip install angr
```

会在虚拟环境中安装一些python包

```
Jack_t0m@ubuntu:~$ mkvirtualenv angr && pip install angr
New python executable in /home/liukai/.virtualenvs/angr/bin/python
Installing setuptools, pip, wheel...done.
```

到此angr安装结束

```
(angr) Jack_t0m@ubuntu:~$
```

注：安装依赖时，可能出现问题。

### **virtualenvwrapper操作命令：**

1. 列出虚拟环境列表:workon, 也可以使用:lsvirtualenv
2. 新建虚拟环境:mkvirtualenv [虚拟环境名称], 环境创建之后, 会自动进入该目录, 并激活该环境。
3. 启动/切换虚拟环境:workon [虚拟环境名称]
4. 删除虚拟环境:rmvirtualenv [虚拟环境名称]
5. 离开虚拟环境:deactivate

```
(angr) Jack_t0m@ubuntu:~$deactivate  
Jack_t0m@ubuntu:~$workon  
angr 3.py      libc.so
```

```
Jack_t0m@ubuntu:~$workon  
angr  
Jack_t0m@ubuntu:~$workon angr  
(angr) Jack_t0m@ubuntu:~$
```

## 以docker的方式安装

为方便起见，上传了一个docker的镜像，99%的可能性保证能够工作。

通过以下步骤就能安装：

```
# #install docker
$curl -sSL https://get.docker.com/ | sudo sh
# #pull the docker image # #可以把想要的镜像拉下来直接使用
$sudo docker pull angr/angr
# #run it
$sudo docker run -it angr
```

## 开发安装

我们使用脚本创建了一个仓库来使一切对于angr的开发者变得更容易。 你可以把angr设置为开发模式通过：

```
git clone https://github.com/angr/angr-dev
cd angr-dev
mkvirtualenv angr
./setup.sh
```

这克隆了所有的仓库并且把它们安装成了可编辑模式。 `setup.sh` 甚至可以为你创建一个PyPy虚拟环境，这使得其有了更快的性能和更低的内存占用。

你可以对不同的模块进行创建分支 / 编辑 / 重新编译操作，产生的变化会自动映射到你的虚拟环境中。

## ◆ 贡献什么？

angr是一个庞大的项目，很难及时更新

### 1. 文档

angr的许多部分很少或根本没有文档。我们迫切需要这方面的社区帮助。

### 2.API

### 3.GitBook

缺少一些概念和生动的例子

### 4.课程

帮助学习

### 5.开发

IDA插件、支持更多架构、路径爆炸、多个并发进程分析的研究

### 6.angr GUI(angr-management)

.....

## ◆ 如何贡献？

报告错误：

如果发现了一些无法解决的问题并且似乎是一个错误，请告知。

● 如何告知？

1. 从 `angr / binaries` 和 `angr / angr` 创建一个 `fork`
2. 创建一个 `angr / binaries` 的 `pull` 请求，以及有问题的二进制文件
3. 创建一个 `angr/angr` 的 `pull` 请求，以及触发二进制程序的测试用例，放在

`angr/tests/broken_x.py` , `angr/tests/broken_y.py` 文件

测试用例格式：

```
1 def test_some_broken_feature():
2     p = angr.Project("some_binary")
3     result = p.analyses.SomethingThatDoesNotWork()
4     assert result == "what it should *actually* be if it worked"
5
6 if __name__ == '__main__':
7     test_some_broken_feature()
```

## 1. 提交的测试用例测试通过后：

当bug被修复后，贡献者提交的**broken\_x.py** 会被更名为**test\_x.py**，并且每次**angr**有改动，都会使用这些测试用例进行测试。

```
4     assert result == "what it should *actually* be if it worked"
```

## 2. 测试用例无效：

首先感谢，如果代码风格差的话，也可能被问候

## ◆ 代码风格

### PEP8 代码风格标准

 python™

» [PEP Index](#) > [PEP 8 -- Style Guide for Python Code](#)

ABOUT »

NEWS »

DOCUMENTATION »

DOWNLOAD »

下载 »

COMMUNITY »

FOUNDATION »

CORE DEVELOPMENT »

**PEP:** 8  
**Title:** Style Guide for Python Code  
**Version:** c451868df657  
**Last-Modified:** 2016-06-08 10:43:53 -0400 (Wed, 08 Jun 2016)  
**Author:** Guido van Rossum <guido at python.org>, Barry Warsaw <barry at python.org>,  
**Status:** Active  
**Type:** Process  
**Content-Type:** [text/x-rst](#)  
**Created:** 05-Jul-2001  
**Post-History:** 05-Jul-2001, 01-Aug-2013

Python Wiki  
Python Insider Blog  
Python 2 or 3?  
Help Fund Python

 or 



#### Contents

- [Introduction](#)
- [A Foolish Consistency is the Hobgoblin of Little Minds](#)
- [Code lay-out](#)
  - [Indentation](#)
  - [Tabs or Spaces?](#)
  - [Maximum Line Length](#)

- 避免使用**Tab**，改用**4**个空格  
因为混合使用制表符和空格缩进的合并代码非常糟糕
- 避免超长线（不易阅读）  
至多**120**个字符
- 避免很长很长的函数，最好分解它
- 使用单下划线定义私有类型

在**python**中定义私有变量只需要在变量名或函数名前加上 “**\_\_**” 两个下划线，那么这个函数或变量就会为私有的了。



## ◆ 经常被问的问题:

### 1. 为什么叫angr

The core of angr's analysis is on VEX IR, and when something is vexing, it makes you angry.

angr分析的核心是VEX IR, 当一些事情令人烦恼时, 它会让你生气。

vexing:令人烦恼的

## 中间语言

由于angr需要处理很多不同的架构, 所以它必须选择一种中间语言 (IR) 来进行它的分析。我们使用Valgrind的中间语言, VEX来完成这方面的内容。VEX中间语言抽象了几种不同架构间的区别, 允许在他们之上进行统一的分析:

For example, consider this x86 instruction:

```
addl %eax, %ebx
```

One Vex IR translation for this code would be this:

```
----- TMark(0x24E275, 7, 0) -----  
t3 = GET:I32(0)           # get %eax, a 32-bit integer  
t2 = GET:I32(12)          # get %ebx, a 32-bit integer  
t1 = Add32(t3,t2)         # addl  
PUT(0) = t1               # put %eax
```

IR的样式

## ◆ 经常被问的问题:

### 1. 为什么叫angr

The core of angr's analysis is on VEX IR, and when something is vexing, it makes you angry.

angr分析的核心是VEX IR, 当一些事情令人烦恼时, 它会让你生气。

vexing:令人烦恼的

### 2. angr的格式


永远小写, 即使位于开头。(反专有名词)


### 3. 如何获取诊断信息?


angr使用标准模块logging进行日志记录, 每个包和子模块都创建一个记录器。


```
1 import logging
2 logging.getLogger('angr').setLevel('DEBUG')
```

帮助文档: [https://github.com/a7vinx/angr-doc-zh\\_CN](https://github.com/a7vinx/angr-doc-zh_CN)

 12 commits

 1 branch

 0 releases

 1 contributor

Branch: master ▾


New pull request

Create new file





Upload files

Find file

Clone or download ▾

 a7vinx Add ir.md

Latest commit 5e0abdd on 17 Oct 2016

 docs	Add ir.md	2 years ago
 INSTALL.md	Fixed error in INSTALL.md	2 years ago
 README.md	Little improve in README.md	2 years ago
 SUMMARY.md	Added SUMMARY.md	2 years ago



ISDC

Sichuan Unviersity Information Security and Network Attack and Defense Community

 Sichuan University, Cheng...

 <https://www.scuisdc.org>

 [info@scuisdc.org](mailto:info@scuisdc.org)