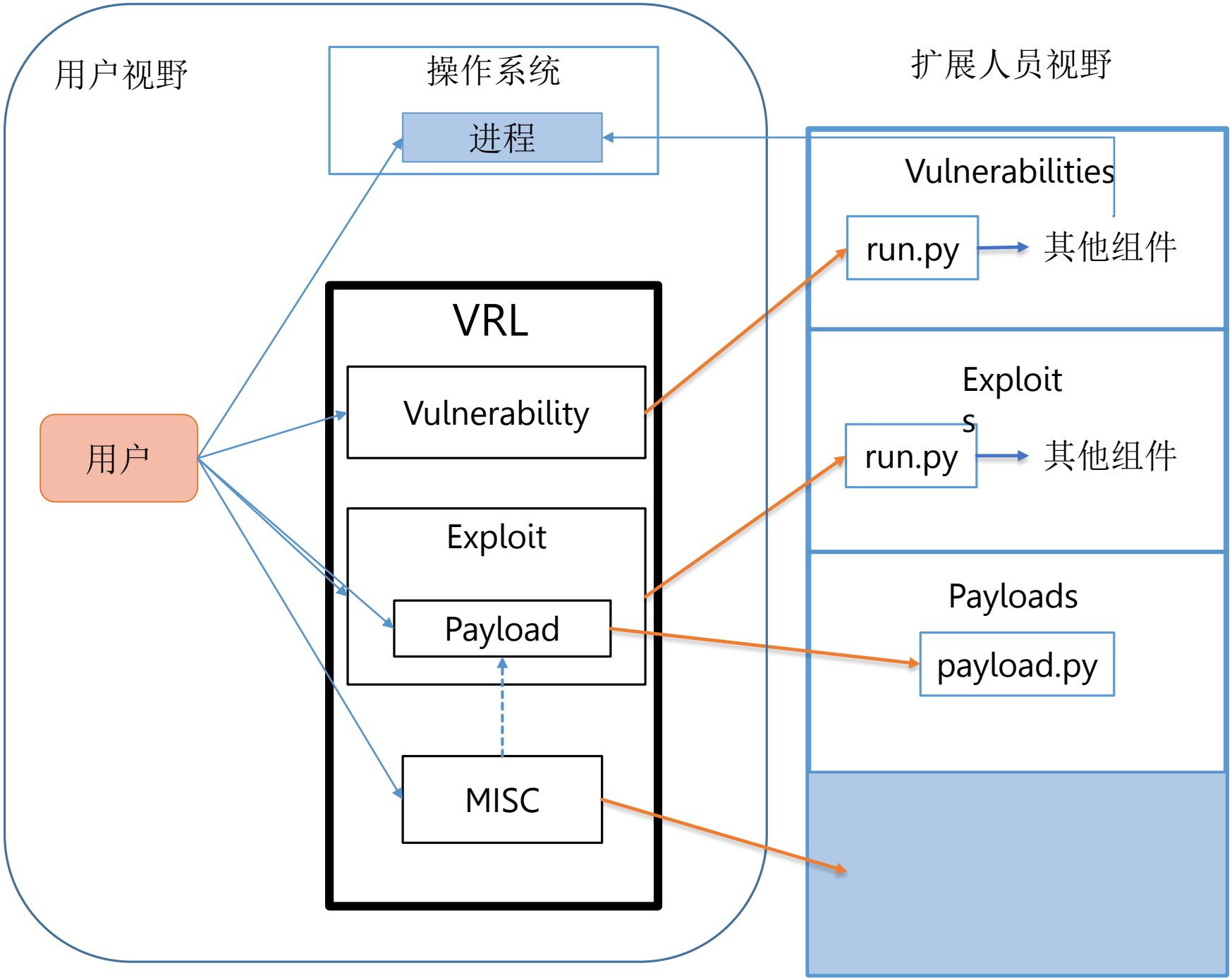
VRL平台的目的

1、尽可能提高漏洞攻击过程中代码复用性和便利性。
2、一个被研究过的漏洞攻击过程可以轻易的被其他学习者复现，学习，调试。
3、提供调试便利。
4、提供定制的扩展。


用户视野
操作系统
进程
扩展人员视野
Vulnerabilities
run.py
其他组件
VRL
Vulnerability
用户
Exploit
Exploits
run.py
其他组件
Exploit
Payload
Payloads
payload.py
MISC

整体架构

UI(VRL.py)
Vulnerabilities
Exploits
Payloads
MISC (其他，链接exploits和payload，payload编码加工，生成rop链工具*等)

UI

所有操作（漏洞程序，攻击程序，payload和其他）在这里统一完成。

在这个程序中可以调用所有需要的程序。(原test和vulserver)

支持各种使用过程：

简易：run name（使用默认的exploit, payload）

复杂：

1.show vul-> use vul-> show exp->use exp->
2.show options -> set options->
3.show payload -> use payload ->
4.(misc) ->run vul -> run exp

Vulnerability

由一个python脚本定义的子类(vulnerability)与框架交互。

包括options,默认options,默认的exploit,可用的exploit,不同防御版本,
Run方法：按照options开启程序
（make方法：在现有环境重新编译这一程序）

新加入漏洞程序的方法：
在vulnerability文件夹下建立一个文件夹，包括：
一个run.py记录上述功能；
编译好的可执行文件供脚本调用；
（src文件夹包含源码和make文档）

Exploit

类似于vulnerabilities，由python脚本和框架交互，简单的可以不需要其他文件。

Exploit的子类：
默认的目标程序(vulnerability名称)，options(与vulnerability对应名称的自动与vulnerability保持一致)，是否支持更换payload，支持的最长payload，payload要求，支持的payload列表。
标准的run方法。

加入新的exploit方法：
在exploit文件夹下建立子文件夹中建立run.py即可。

Payload

常用的payload，用python脚本或者.json文件保存(暂不支持json)即可，包括payload的长度，功能，适用平台。

支持在Exploit中选择默认payload自动载入。

MISC

其他工具：
编码payload工具：
一个工具来用其他工具组合、加工payload生成一个临时payload使用。例如，避免NULL字符的编码，unincode编码（生成rop链，对喷射等用的nop代码等等如果扩展放到这里）

更改系统设置工具：
ASLR，DEP等。

自动附加进程工具等。

工程进度

1. 基本功能完成：载入和使用Vulnerability，Exploit，Payload。
2. 支持自动载入对应的Vulnerability，Exploit，Payload。
3. 灵活的命令行 （多种格式支持，不区分大小写，自动补全）
4. 详细文档和扩展样例。
5. 简单的栈溢出样例。

完成/未完成的功能

已完成：

1、命令行命令(use, run, info, stop, make, show, set, etc.)。
2、扩展run.py自动检查，载入payload，运行。
3、调出GDB等简单工具。
4、自动载入默认vul、exp、pay。
5、run.py新终端开启函数，方便run.py编写。

未完成：

1、payload加工
2、JOP/ROP构建工具
3、auto attach（一键调试）
4、脚本自定义方法调用
5、打开/关闭系统ASLR、DEP
6、脚本名自动补全