# Read Papers 2

| | |
|---|---|
| Assign | |
| Date Created | @Jan 11, 2021 2:53 PM |
| Due Date | |
| Priority | Low |
| Property | |
| Status | Completed |

### Fuzzing学习资源汇总

https://scubsrgroup.github.io/BinaryDatabase/Fuzzing-学习资源汇总.html#qsym-2018

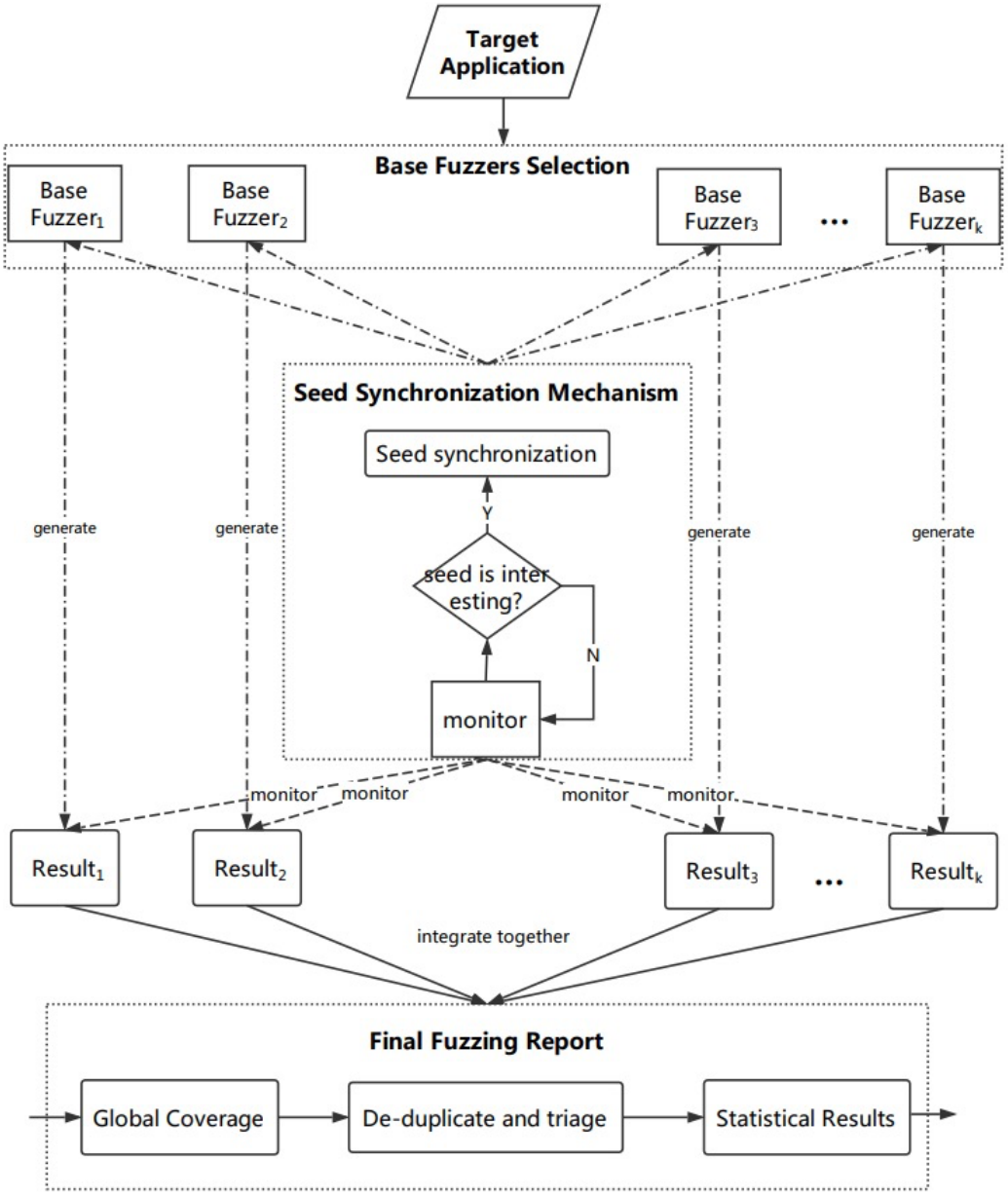
问就是只看中文。

看了一遍，用到再从里面找吧，什么样的fuzzer都堆在一起。

### EnFuzz: Ensemble Fuzzing with Seed Synchronization among Diverse Fuzzers

https://arxiv.org/pdf/1807.00182.pdf

However, in industrial practice, it is found that the performance of those well-designed fuzzing strategies is challenged by the complexity and diversity of real-world applications. 确实。




文章把一些好用的fuzzer，然后Ensemble Fuzzing，ensemble是合奏的意思。Evaluation写了一大堆，也不算长吧，附录比较长。其中fuzz realworld：

Table 12: Unique previously unknown bugs detected by each tool within 24 hours on some real-world applications.

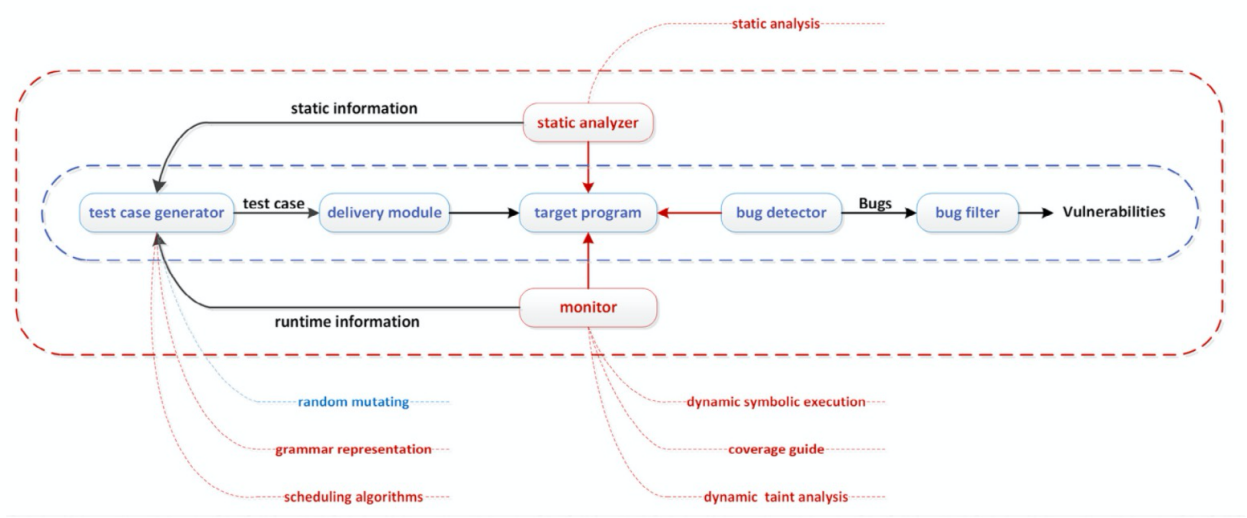
| Project | AFL | AFLFast | FairFuzz | LibFuzzer | QSYM | EnFuzz |
|---|---|---|---|---|---|---|
| Bento4_mp4com | 5 | 4 | 5 | 5 | 4 | **6** |
| Bento4_mp4tag | 5 | 4 | 4 | 5 | 4 | **7** |
| bitmap | 1 | 1 | 1 | 0 | 1 | **2** |
| cmft | 1 | 1 | 0 | 1 | 0 | **2** |
| ffjpeg | 1 | 1 | 1 | 0 | 1 | **2** |
| flif | 1 | 1 | 1 | 2 | 1 | **3** |
| imageworsener | **1** | 0 | 0 | 0 | **1** | **1** |
| libjpeg-05-2018 | 3 | 3 | 3 | 4 | 3 | **5** |
| libiec61850 | 3 | 2 | 2 | 1 | 2 | **4** |
| libpng-1.6.34 | 2 | 1 | 1 | 1 | 2 | **3** |
| libwav_wavgain | 3 | 2 | 3 | 0 | 2 | **5** |
| libwav_wavinfo | 2 | 1 | 2 | 4 | 2 | **5** |
| LuPng | 1 | 1 | 1 | 3 | 1 | **4** |
| pbc | 5 | 5 | 6 | 7 | 6 | **9** |
| pngwriter | 1 | 1 | 1 | 1 | **2** | **2** |
| total | 35 | 28 | 31 | 34 | 32 | **60** |

**FUZZIFICATION: Anti-Fuzzing Techniques**

https://securitygossip.com/blog/2019/11/08/fuzzification-anti-fuzzing-techniques/

这个有意思，还是尊贵的中文：

> 有效的模糊化技术应启用以下三个功能：
>
> - 首先，它应该有效地阻止现有的模糊测试工具，使其在固定时间内发现更少的错误；
> - 其次，受保护的程序在正常使用下仍应有效运行；
> - 最后，不应被分析技术轻易地识别保护代码或将其从受保护的二进制文件中删除。




## InternalBlue – Bluetooth Binary Patching and Experimentation Framework

https://arxiv.org/pdf/1905.00631.pdf

InternalBlue is a versatile framework and we demonstrate its abilities by implementing tests and demos for known Bluetooth vulnerabilities. Moreover, we discover a novel critical security issue affecting a large selection of Broadcom chipsets that allows executing code within the attacked Bluetooth firmware. We further show how to use our framework to fix bugs in chipsets out of vendor support and how to add new security features to Bluetooth firmware.

通过逆向攒了个蓝牙交互框架，可以验证、挖掘、修复漏洞。

## SymQEMU: Compilation-based symbolic execution for binaries

http://s3.eurecom.fr/docs/ndss21_symqemu.pdf

在哪里看过好像，不用source-base，可以在qemu translate的时候操作。对符号执行并不了解，也许因为符号执行总是被吐槽停留在学术界?

## Understanding Android VoIP Security: A System-level Vulnerability Assessment

https://daoyuan14.github.io/papers/TR19_VoIPFuzzing.pdf

这篇论文系统性地研究了Android系统VoIP协议——这一与手机上打电话强相关的功能在代码实现上的脆弱性，创新性地结合设备侧的Intent/系统API Fuzzing、网络侧的数据包Fuzzing和对存在不一致性的代码定向审计等研究方法，首次披露了存在于Android 7.0至9.0的8个Android VoIP的0 day漏洞，此类型的漏洞可造成永久拒绝服务、敏感信息泄漏、号码仿冒、未授权拨号、远程代码执行等各种危害。 目前，论文中所涉及的漏洞均已被厂商修复。

VoIP（Voice over Internet Protocol）即首先数字化语音信号并压缩成帧，转换为IP数据包在网络上传输，以此完成语音通话的业务，是一种利用IP协议传输语音数据的、新兴的通信技术。

随着我国三网融合的推进，VoIP与IPTV（Interactive Personality TV）一起成为这一庞大工程的重要标志。

VoIP是一项新技术，可通过数据包交换IP网络向移动设备提供语音、传真、SMS、语音消息，并且VoIP还支持包括 VoLTE 和 VoWiFi 标准的Android 。

攻击面的网络协议，解码之类的。

- SIP (Session Initiation Protocol): Android's SIP implementation directly uses the nist-sip library, which was developed by National Institute of Science of Technology (NIST). It is a purely Java based SIP

implementation, and provides API classes (e.g., SipSession and SipProfile) via the android.net.sip package.

- SDP (Session Description Protocol): Similar to SIP, Android's SDP also uses the NIST implementation (gov.nist.javax.sdp), and provides a hidden API class called SdpSessionDescription.
- RTP (Real-time Transport Protocol): Android implements RTP in a C/C++ dynamic link library called librtp-jni.so. It also provides a few API classes via the android.net.rtp package.
- Audio or Video Codec: Android VoIP supports only a handful of codecs, including PCM (Pulse-Code Modulation) type A and type U codec, AMR (Adaptive MultiRate) codec, and GSM EFR (Enhanced Full Rate) codec. Supporting these codecs relies on libstagefright.
- SIP UA (User Agent): Android VoIP implements its UA into the system phone app (com.android.phone). It is a high-privilege app under the Linux user group of radio. Hence, it can not only access typical phonerelated permissions (e.g., accessing user contacts and making a phone call) but also low-level resources in the Telephone Manager and Radio Interface Layer (RIL). Additionally, displaying VoIP caller numbers is handled by the system dialer app (com.android.dialer).

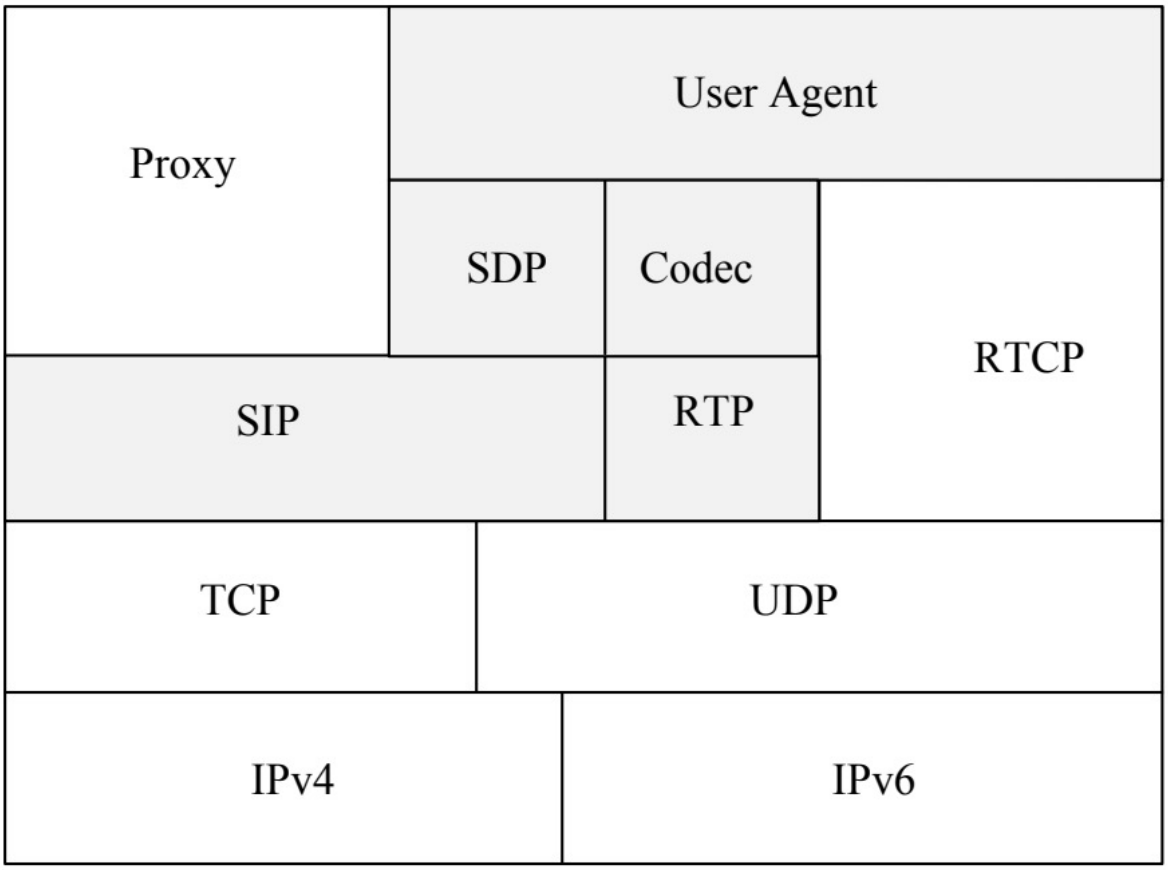
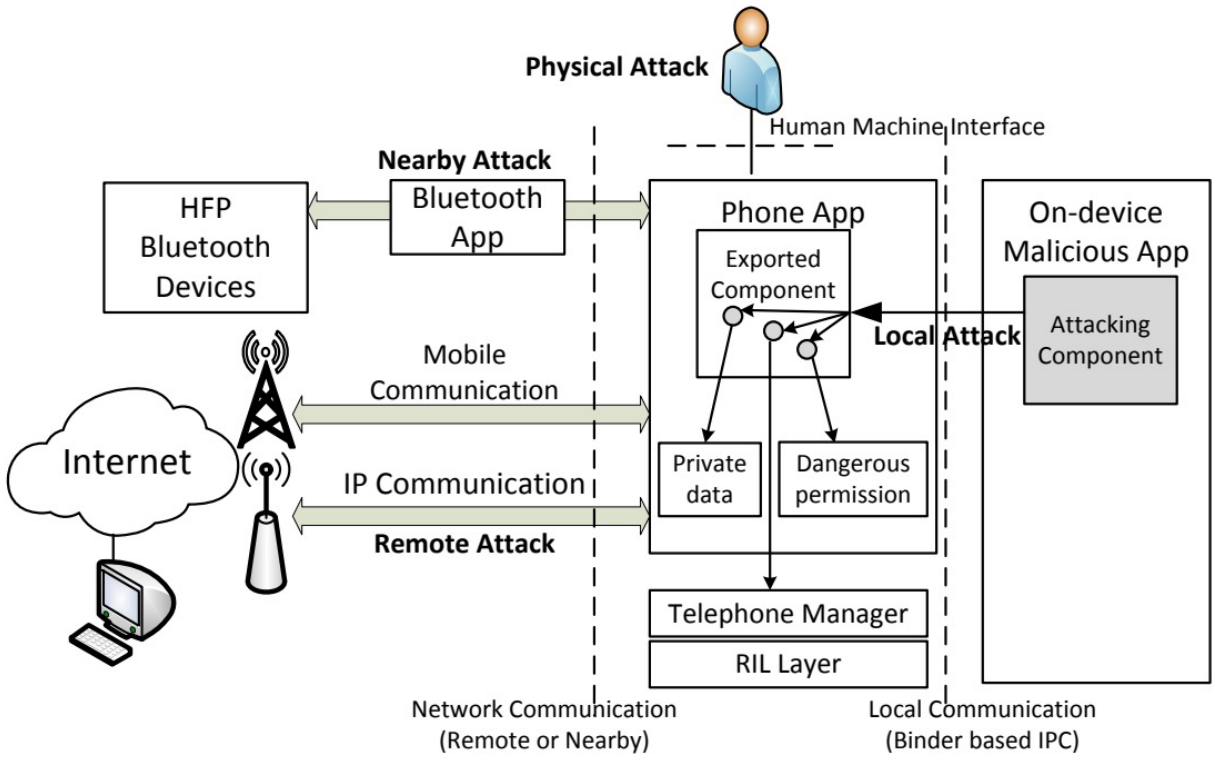



**Fig. 2: Android's integration of VoIP protocol stack.**

- 用SIP/SDP/RTP远程打电话；
- 蓝牙靠近打VoIP；

- Local IPC攻击；
- 物理攻击，改配置；




用Drozer fuzz的模样。

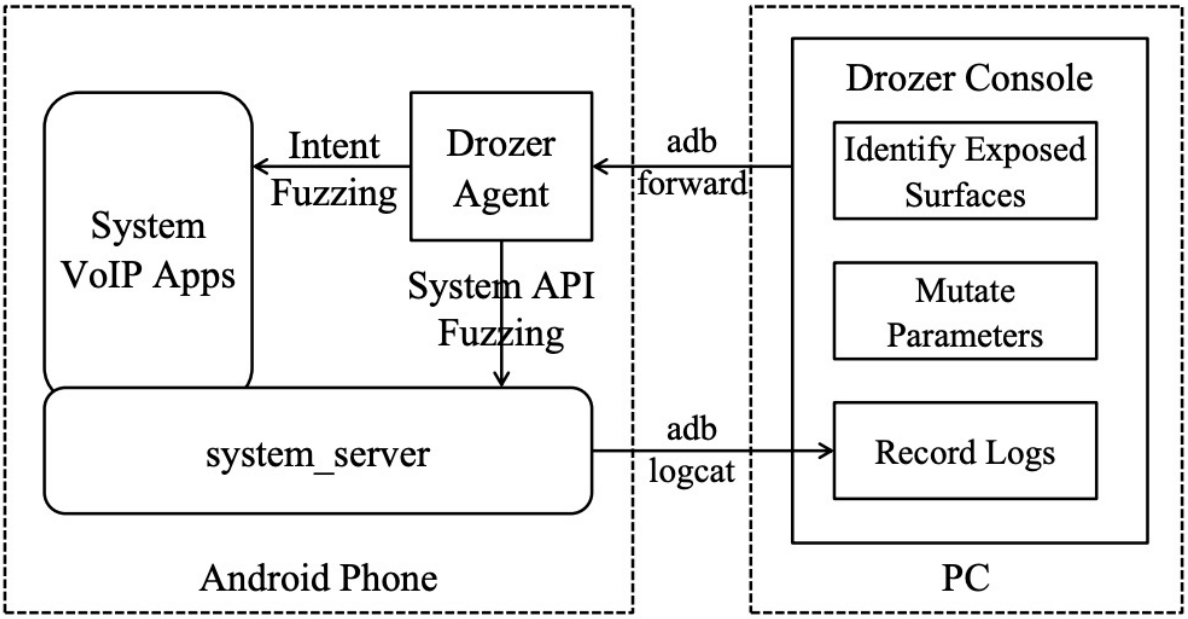



**Fig. 4: The on-device fuzzing framework.**

- On-device Fuzzing是那个Drozer。
- Network-side Fuzzing是弄了脚本往内发，去fuzz。
- Code Auditing
  - Log-driven auditing. 通过log报错定位信息去审计；
  - Protocol specification based auditing. 审计一些特殊特性，'&'之类的；For example, special attributes, e.g., the call transfer splitting character “&” and the phone number prefix “phone-context”, in PSTN may have different behaviors in VoIP, which we will illustrate later.

下面这些包名都是什么东西啊，有的都没听说过，也没逆过，好像可以逆逆看看。看了看洞是一些名字啊电话号过长之类的溢出，RCE或者DoS，还有一些审计出来的逻辑漏洞。

**TABLE I: Zero-day Android VoIP vulnerabilities discovered in our work.**

| Discovery Method | ID | CVE/AID | Attack Vector | Vulnerable Entry Component | Affected Android | Severity Level | Security Consequence |
|---|---|---|---|---|---|---|---|
| On-device Fuzzing | V1 | H1-#386144 | Local | `com.vkontakte.android` | All | Low | Triggering a call without user's consent |
| | V2 | CVE-2017-11042 | Local | `org.codeaurora.ims` | $\leq$ 7.1.2 | Moderate | Unauthorized setting of call transfer |
| Network-side Fuzzing | V3 | A-31823540-1 | Remote | `com.android.dialer` | $\leq$ 7.1.1 | High | Undeniable VoIP call spam |
| | V4 | CVE-2017-0394 | Remote | `com.android.phone` | $\leq$ 7.1.1 | High | Remote DoS once accepting a call |
| | V5 | CVE-2018-9475 | Remote* | `com.android.bluetooth` | $\leq$ 9.0 | Critical | Remote code execution due to overflow |
| | V6 | A-79431031 | Remote* | `com.android.bluetooth` | $\leq$ 9.0 | High | Remote DoS once receiving a call |
| Code Auditing | V7 | CVE-2016-6763 | Physical | `com.android.phone` | $\leq$ 7.0 | High | Sensitive data leak; Permanent DoS |
| | V8 | A-31823540-2 | Remote | `com.android.dialer` | $\leq$ 7.1.1 | High | Caller ID spoofing |
| | V9 | A-32623587 | Remote | `com.android.dialer` | $\leq$ 7.1.1 | High | Caller ID spoofing |

* These two remote vulnerabilities could be triggered only when the phone is connected with a *nearby* Bluetooth-based HFP (Hands-Free Profile) device.

### Chizpurfle: A Gray-Box Android Fuzzer for Vendor Service Customizations

http://wpage.unina.it/roberto.natella/papers/natella_androidfuzzing_issre2017.pdf

灰盒fuzzer，frida插桩，遗传算法，fuzz service。用法长下面这样。这不frida结合afl跑在Android上。也支持黑盒直接怼。

```
./chizpurfle -h
cat chizpurfle.shell
>usage: chizpurfle
> -bb,--black-box                                uses a blackbox approach
> -e,--extract                                   Extract the model from the smartphone
> -f1,--blocks-counter-fitness-evaluator         uses the blocks counter fitness evaluator (default)
> -f2,--branch-execution-fitness-evaluator       uses the blocks branch execution evaluator
> -f3,--coarse-branch-hit-fitness-evaluator      uses the blocks coarse branch hit evaluator
> -h,--help                                      show help
> -n,--max-generation <arg>                      the number of generations the populations should pass through (default is 20)
> -process,--process-name <arg>                  the name of the process to trace
> -s1,--fitness-proportionate-selection          uses a fitness proportionate selection algorithm (default)
> -s2,--ranking-selection                        uses a ranking selection algorithm
> -s3,--tournament-selection                     uses a tournament selection algorithm
> -service,--service-name <arg>                  the name of the service under test
> -method,--method-name <arg>                    the name of the method under test
>
>Thank you for feeding me!
```

## Evolutionary Fuzzing of Android OS Vendor System Services

https://arxiv.org/pdf/1906.00621.pdf

Evolutionary Chizpurfle。跟上面那篇父子篇，还是看上面那篇，这篇不是两栏的，看不惯。

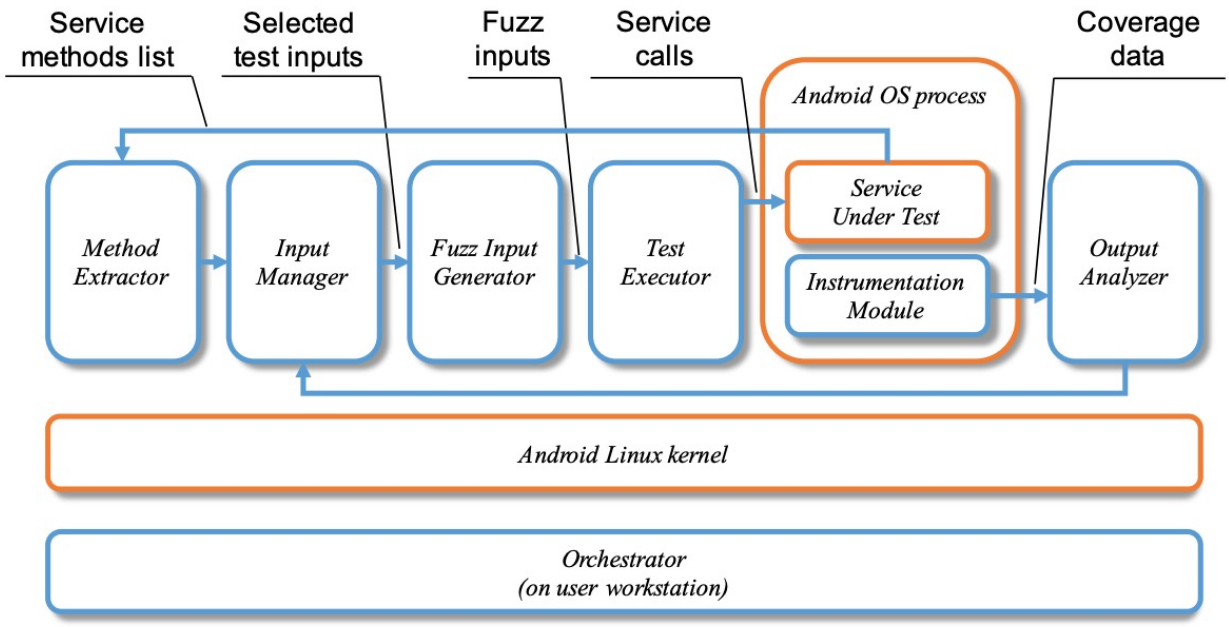
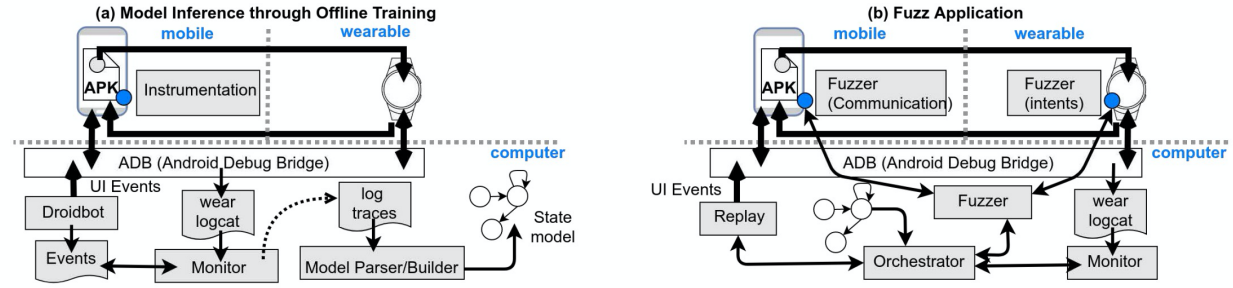



**Fig. 2** Overview of Chizpurfle components.

## Vulcan: Lessons on Reliability of Wearables through State-Aware Fuzzing

还是fuzz Android那些玩意。

- Intent Injection.
- Communication Fuzzing.
- Automated Intent Specification Generation.




### 枫聆大佬的知乎专栏

https://www.zhihu.com/column/c_1272166623609475072

睡前看完了，也不知道记住多少，看起来很高级的样子，有很多数学知识在里面。

## Others

可以用gty式快速阅读法读，了解到GoSSIP那样差不多了，我是嘤文太烂一篇文章看半天。

问就是中文真爽：https://securitygossip.com/

下面这两个，太长了。我原称之为读不完的Paper，主要也是看一个Abstract，然后看看标题，吐槽吐槽，这里应该不好写，直接新开一个markdown吧。

### awesome fuzzing

https://github.com/cpuu/awesome-fuzzing

看完了。

### Paper Collection

https://github.com/0xricksanchez/paper_collection

大致看完了。

### Software Security Paper List

https://github.com/AdaLogics/software-security-paper-list

大致看完了。

### Some Papers About Fuzzing

https://github.com/bsauce/Some-Papers-About-Fuzzing

这个是带阅读整理的。看了一遍了。