自动化生成漏洞复现环境
与大规模高精确度的漏洞信息获取

华中科技大学 - 网络空间安全学院

宋静怡 - 2024.5.25

https://github.com/hust-open-atom-club/S2VulnHub

漏洞复现是缓解漏洞的关键步骤

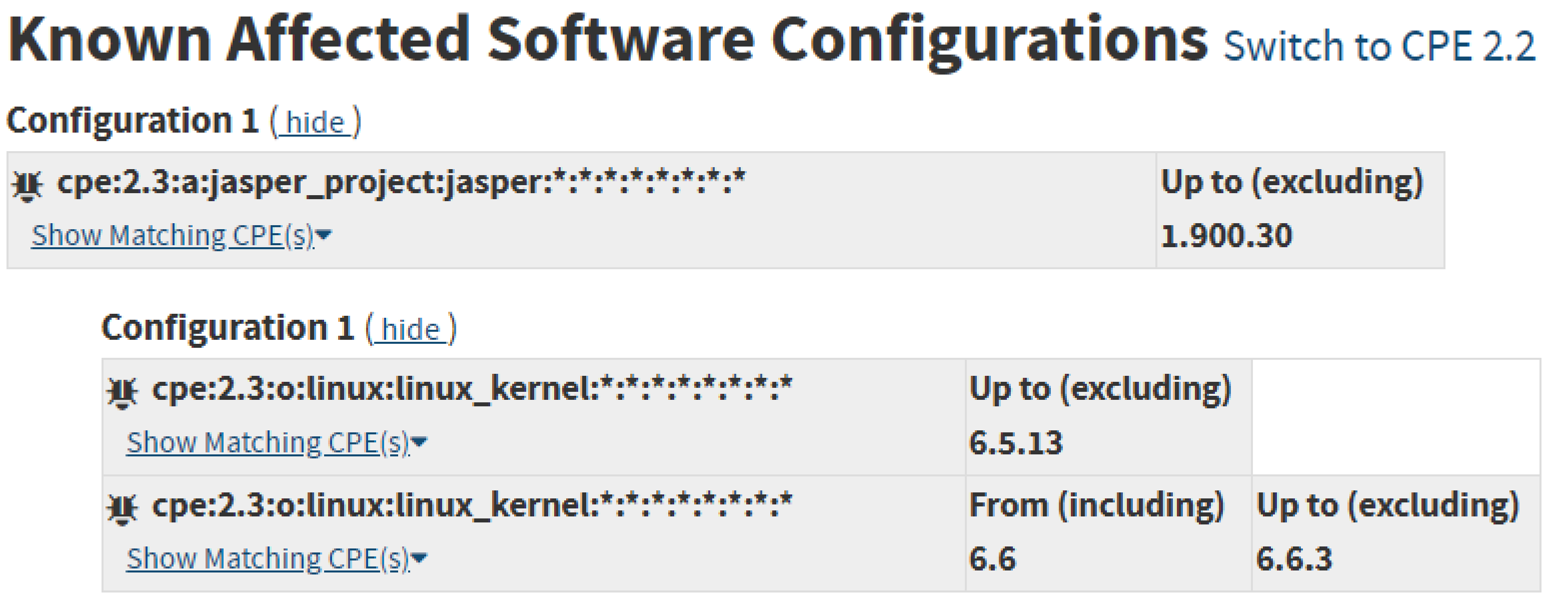
漏洞生命周期

- 发现异常
- 漏洞分类和优先级确定
- 漏洞解决，发布安全补丁
- 发布最终安全报告

漏洞复现

- 确定漏洞存在最准确的方式
- 详细了解漏洞的触发条件和潜在后果
    - 生成安全补丁
    - 评估漏洞危险性
    - 促进漏洞检测和漏洞防御

漏洞复现提升漏洞数据库质量



Known Affected Software Configurations Switch to CPE 2.2

Configuration 1 (hide)

cpe:2.3:a:jasper_project:jasper:*:*:*:*:*:*:*:* Show Matching CPE(s)▾	Up to (excluding) 1.900.30	

Configuration 1 (hide)

cpe:2.3:o:linux:linux_kernel:*:*:*:*:*:*:*:* Show Matching CPE(s)▾	Up to (excluding) 6.5.13	
cpe:2.3:o:linux:linux_kernel:*:*:*:*:*:*:*:* Show Matching CPE(s)▾	From (including) 6.6	Up to (excluding) 6.6.3

- 确定漏洞影响的软件版本信息，但静态方法误报率较高
- 提升开源漏洞数据库质量
- 辅助 SBOM 和 SCA 等软件供应链安全保障技术

漏洞复现需要人力和专业知识

用户态漏洞

- 漏洞软件获取
- 软件依赖获取
- 编译命令获取
- PoC 获取

内核态漏洞

- 内核源码获取
- 设置编译选项并编译内核
- PoC 获取

[1] Mu, Dongliang, et al. "Understanding the reproducibility of crowd-reported security vulnerabilities." *27th USENIX Security Symposium (USENIX Security 18)*. 2018.

Docker — 最方便的漏洞复现方式

- Docker 可以很好地解决环境配置问题。容器不是模拟一个完整的操作系统，而是对进程进行隔离。

- Vulhub 是一个面向大众的开源漏洞靶场，无需 docker 知识，简单执行一条命令即可编译、运行一个完整的漏洞靶场镜像。
- Vulhub 的漏洞集中于Web应用，而我们希望关注 Linux 中用户态软件和 Linux 内核的内存漏洞。

[2] https://vulhub.org/

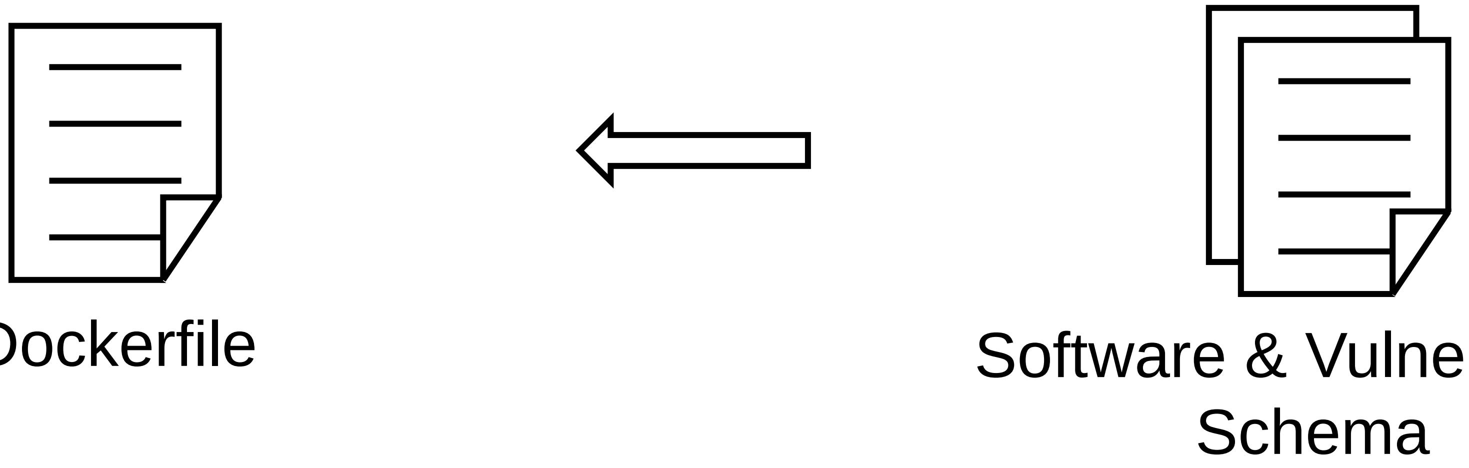
Docker — 最方便的漏洞复现方式

如何降低漏洞复现的门槛?

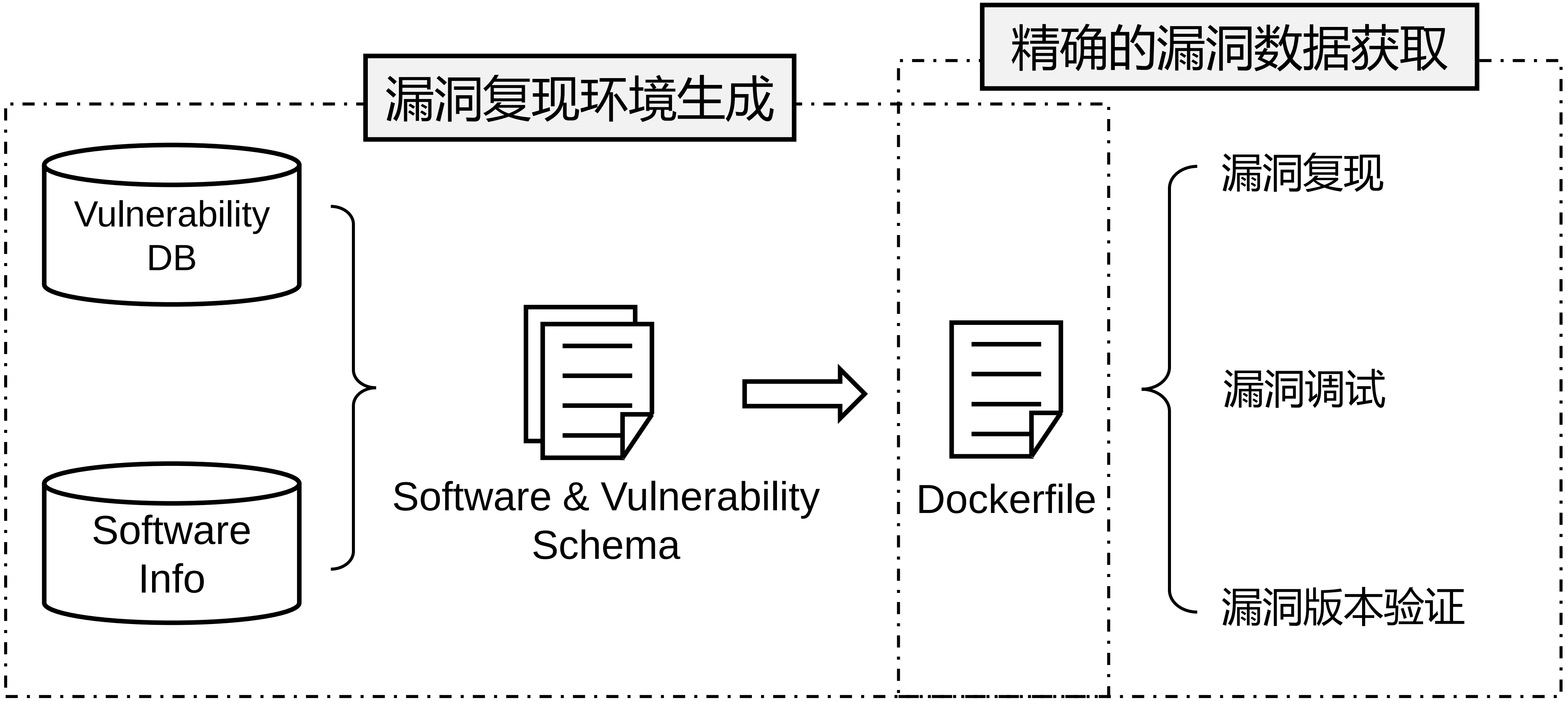
- 将漏洞环境打包为 docker

如何自动化 Dockerfile 的生成?

- 将必要信息存储到模板



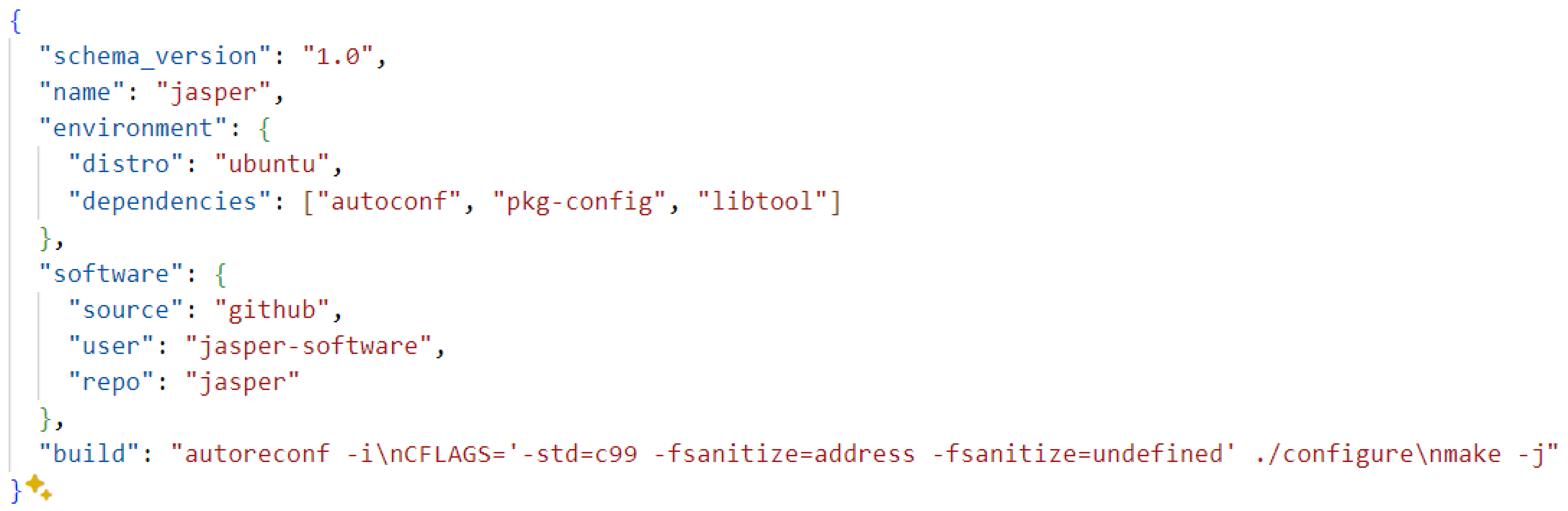
S2VulHub 系统设计



[3] https://github.com/hust-open-atom-club/S2VulnHub

Software Schema

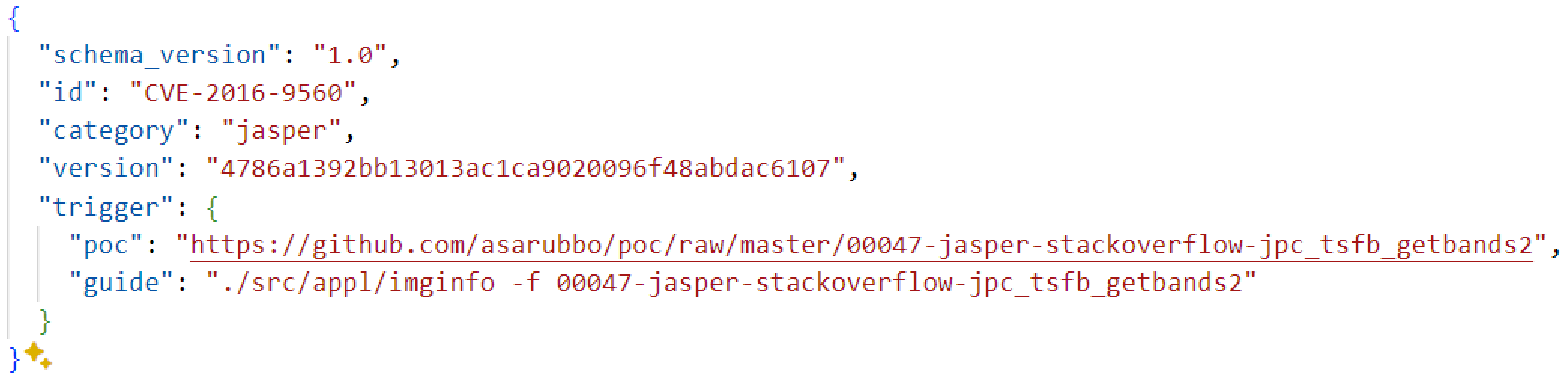
- 软件依赖，软件获取，编译命令



```json
{
  "schema_version": "1.0",
  "name": "jasper",
  "environment": {
    "distro": "ubuntu",
    "dependencies": ["autoconf", "pkg-config", "libtool"]
  },
  "software": {
    "source": "github",
    "user": "jasper-software",
    "repo": "jasper"
  },
  "build": "autoreconf -i\nCFLAGS='-std=c99 -fsanitize=address -fsanitize=undefined' ./configure\nmake -j"
}
```

Vulnerability Schema

- CVE ID，漏洞软件，软件版本，PoC



```
{
  "schema_version": "1.0",
  "id": "CVE-2016-9560",
  "category": "jasper",
  "version": "4786a1392bb13013ac1ca9020096f48abdac6107",
  "trigger": {
    "poc": "https://github.com/asarubbo/poc/raw/master/00047-jasper-stackoverflow-jpc_tsfb_getbands2",
    "guide": "./src/appl/imginfo -f 00047-jasper-stackoverflow-jpc_tsfb_getbands2"
  }
}
```

Dockerfile 生成

```
FROM ubuntu:16.04
RUN sed -i "s@http://.*archive.ubuntu.com@http://mirrors.ustc.edu.cn/@g" /etc/apt/sources.list
RUN sed -i "s@http://.*security.ubuntu.com@http://mirrors.ustc.edu.cn/@g" /etc/apt/sources.list
ARG DEBIAN_FRONTEND=noninteractive
RUN apt update && apt install -y iputils-ping wget git vim build-essential cmake unzip
RUN apt install -y autoconf pkg-config libtool
WORKDIR /root
RUN git clone https://github.com/jasper-software/jasper
WORKDIR /root/jasper
RUN git checkout 4786a1392bb13013ac1ca9020096f48abdac6107
RUN echo -n YXV0b3JlY29uZiAtaQpDRkxBR1M9Jy1zdGQ9Yzk5IC1mc2FuaXRpemU9YWRkcmVzcyAtZnNhbml0aXplPXVuZ
RUN wget https://github.com/asarubbo/poc/raw/master/00047-jasper-stackoverflow-jpc_tsfb_getbands2
RUN echo -n Li9zcmMvYXBwbC9pbWdpbmZvIC1mIDAwMDQ3LWphc3Blci1zdGFja292ZXJmbG93LWpwY190c2ZiX2dldGJhb
RUN bash build.sh
CMD ["/bin/bash"]
```

build.sh

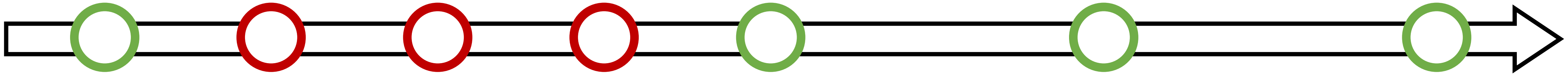
trigger.sh

漏洞版本验证

自动启动容器并运行 PoC，根据容器退出状态和错误输出判断漏洞存在性

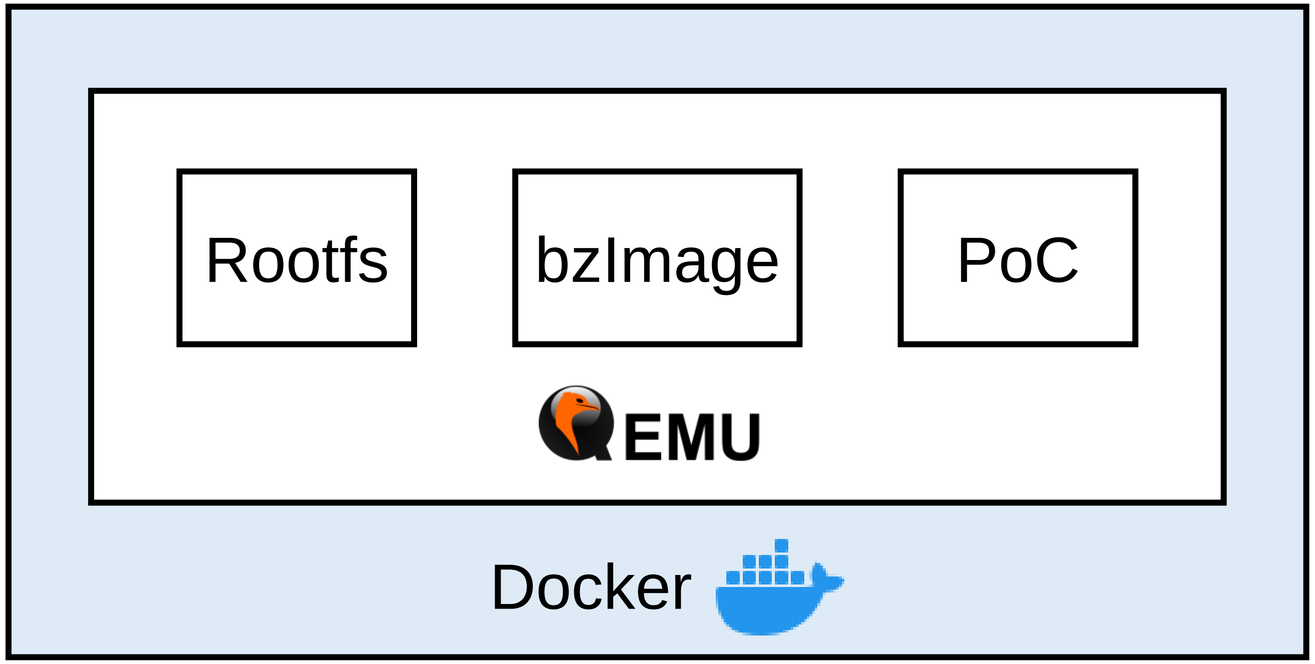
- 输入：CVE-ID，软件版本
- 输出：指定版本软件是否存在漏洞

二分法确定漏洞影响版本 (Bisection)



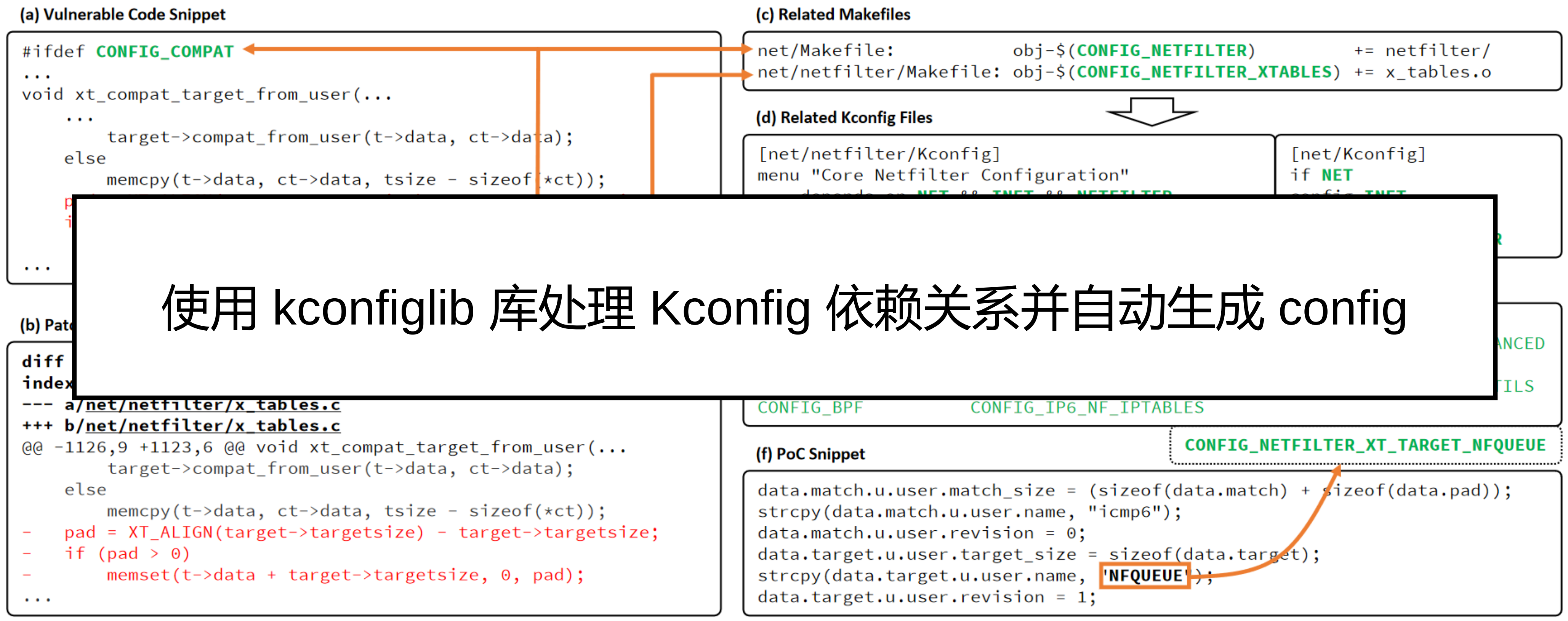
内核态漏洞复现

- QEMU 是一款开源的模拟器及虚拟机监管器，可以模拟启动 Linux 系统。
- Rootfs 是操作系统在启动和运行时使用的主要文件系统，是系统能够启动、运行并提供基本功能的基础。



内核编译选项

bzImage = source code + config，确保漏洞模块被编译进内核



(a) Vulnerable Code Snippet

```
#ifdef CONFIG_COMPAT
...
void xt_compat_target_from_user(...
    ...
        target->compat_from_user(t->data, ct->data);
    else
        memcpy(t->data, ct->data, tsize - sizeof(*ct));
...
```

(b) Pat

```
diff
index
--- a/net/netfilter/x_tables.c
+++ b/net/netfilter/x_tables.c
@@ -1126,9 +1123,6 @@ void xt_compat_target_from_user(...
        target->compat_from_user(t->data, ct->data);
    else
        memcpy(t->data, ct->data, tsize - sizeof(*ct));
-   pad = XT_ALIGN(target->targetsize) - target->targetsize;
-   if (pad > 0)
-       memset(t->data + target->targetsize, 0, pad);
...
```

(c) Related Makefiles

```
net/Makefile:            obj-$(CONFIG_NETFILTER)          += netfilter/
net/netfilter/Makefile:  obj-$(CONFIG_NETFILTER_XTABLES)  += x_tables.o
```

(d) Related Kconfig Files

```
[net/netfilter/Kconfig]
menu "Core Netfilter Configuration"
```

```
[net/Kconfig]
if NET
```

```
ANCED
TILS
CONFIG_BPF          CONFIG_IP6_NF_IPTABLES
```

CONFIG_NETFILTER_XT_TARGET_NFQUEUE

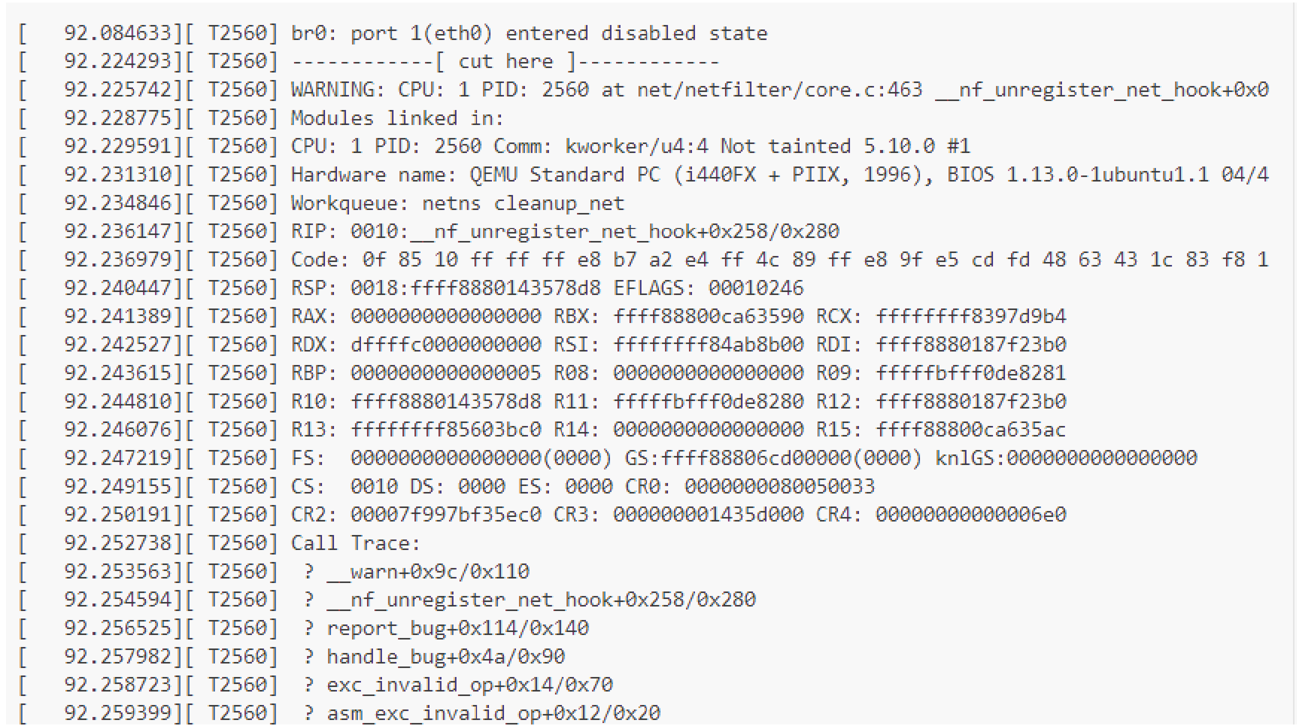
(f) PoC Snippet

```
data.match.u.user.match_size = (sizeof(data.match) + sizeof(data.pad));
strcpy(data.match.u.user.name, "icmp6");
data.match.u.user.revision = 0;
data.target.u.user.target_size = sizeof(data.target);
strcpy(data.target.u.user.name, "NFQUEUE");
data.target.u.user.revision = 1;
```

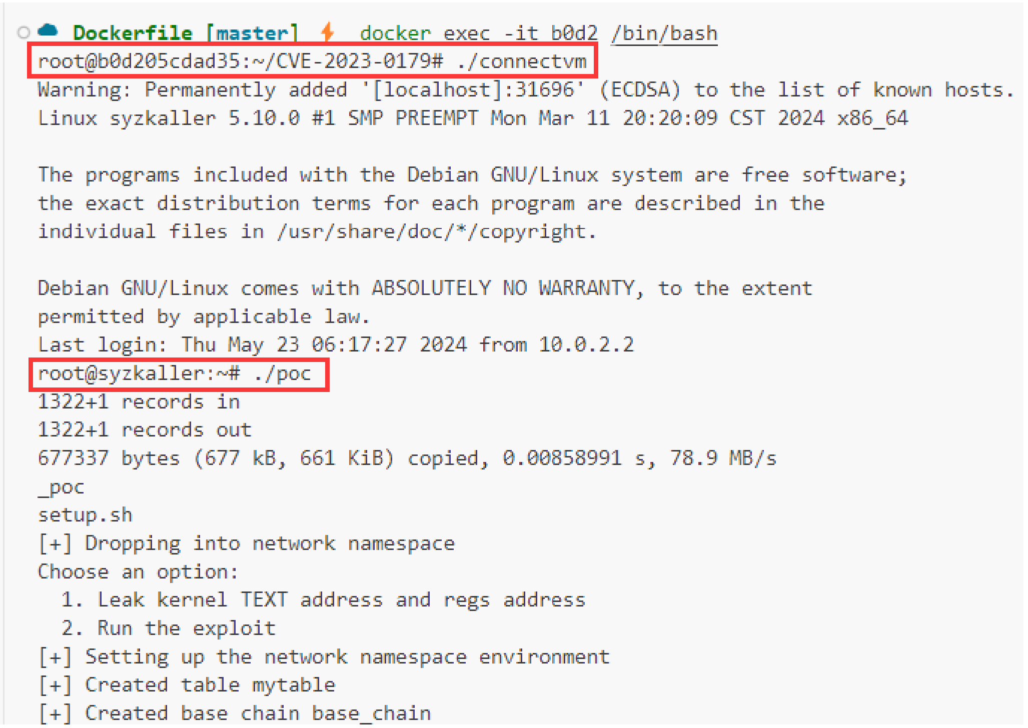
使用 kconfiglib 库处理 Kconfig 依赖关系并自动生成 config

内核漏洞复现

编译内核 + 启动QEMU + 运行 PoC

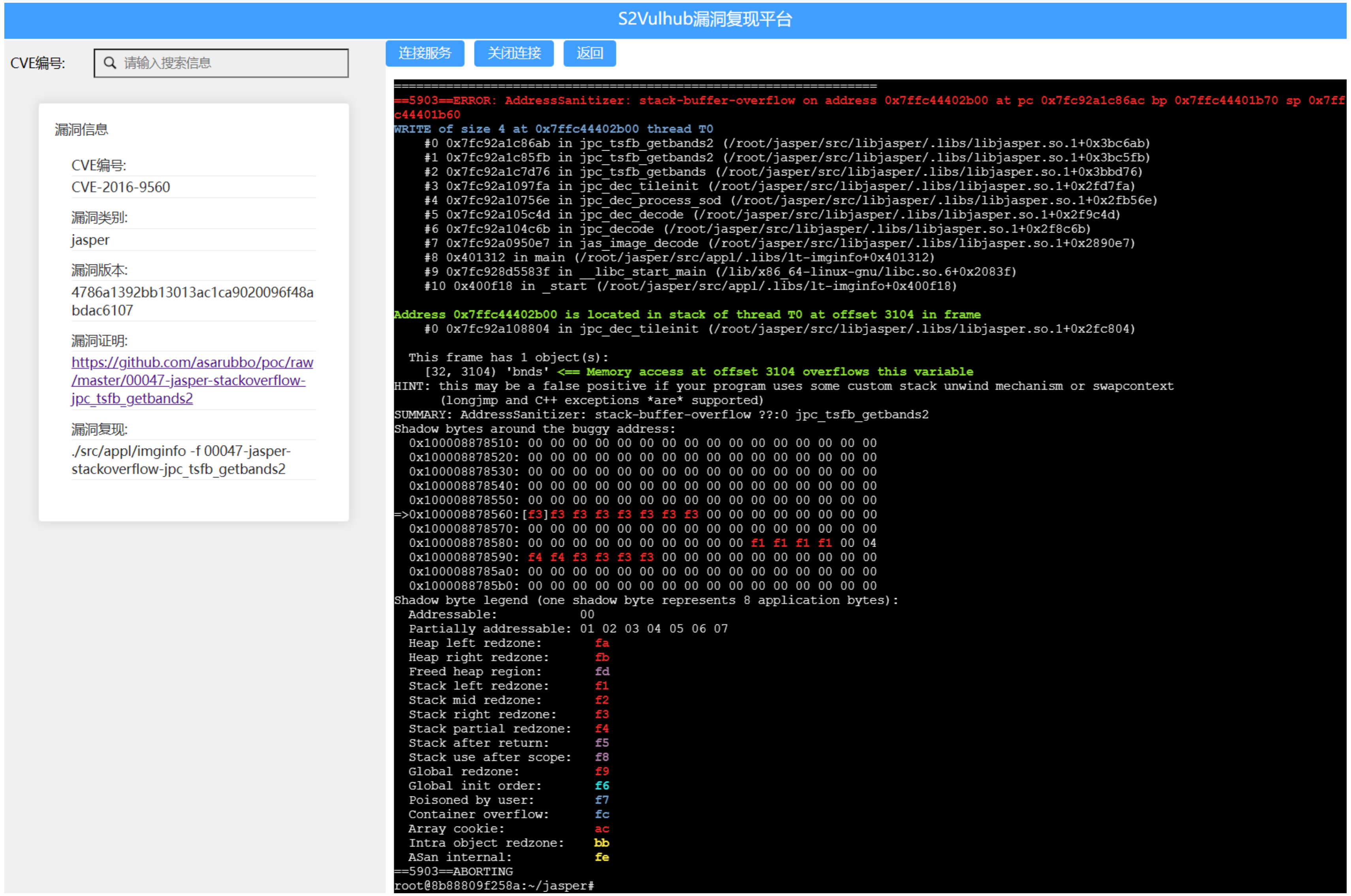


```
[   92.084633][ T2560] br0: port 1(eth0) entered disabled state
[   92.224293][ T2560] ------------[ cut here ]------------
[   92.225742][ T2560] WARNING: CPU: 1 PID: 2560 at net/netfilter/core.c:463 __nf_unregister_net_hook+0x0
[   92.228775][ T2560] Modules linked in:
[   92.229591][ T2560] CPU: 1 PID: 2560 Comm: kworker/u4:4 Not tainted 5.10.0 #1
[   92.231310][ T2560] Hardware name: QEMU Standard PC (i440FX + PIIX, 1996), BIOS 1.13.0-1ubuntu1.1 04/4
[   92.234846][ T2560] Workqueue: netns cleanup_net
[   92.236147][ T2560] RIP: 0010:__nf_unregister_net_hook+0x258/0x280
[   92.236979][ T2560] Code: 0f 85 10 ff ff ff e8 b7 a2 e4 ff 4c 89 ff e8 9f e5 cd fd 48 63 43 1c 83 f8 1
[   92.240447][ T2560] RSP: 0018:ffff8880143578d8 EFLAGS: 00010246
[   92.241389][ T2560] RAX: 0000000000000000 RBX: ffff88800ca63590 RCX: ffffffff8397d9b4
[   92.242527][ T2560] RDX: dffffc0000000000 RSI: ffffffff84ab8b00 RDI: ffff8880187f23b0
[   92.243615][ T2560] RBP: 0000000000000005 R08: 0000000000000000 R09: fffffbfff0de8281
[   92.244810][ T2560] R10: ffff8880143578d8 R11: fffffbfff0de8280 R12: ffff8880187f23b0
[   92.246076][ T2560] R13: ffffffff85603bc0 R14: 0000000000000000 R15: ffff88800ca635ac
[   92.247219][ T2560] FS:  0000000000000000(0000) GS:ffff88806cd00000(0000) knlGS:0000000000000000
[   92.249155][ T2560] CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
[   92.250191][ T2560] CR2: 00007f997bf35ec0 CR3: 000000001435d000 CR4: 00000000000006e0
[   92.252738][ T2560] Call Trace:
[   92.253563][ T2560]  ? __warn+0x9c/0x110
[   92.254594][ T2560]  ? __nf_unregister_net_hook+0x258/0x280
[   92.256525][ T2560]  ? report_bug+0x114/0x140
[   92.257982][ T2560]  ? handle_bug+0x4a/0x90
[   92.258723][ T2560]  ? exc_invalid_op+0x14/0x70
[   92.259399][ T2560]  ? asm_exc_invalid_op+0x12/0x20
```



```
○   Dockerfile [master]    docker exec -it b0d2 /bin/bash
root@b0d205cdad35:~/CVE-2023-0179# ./connectvm
Warning: Permanently added '[localhost]:31696' (ECDSA) to the list of known hosts.
Linux syzkaller 5.10.0 #1 SMP PREEMPT Mon Mar 11 20:20:09 CST 2024 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu May 23 06:17:27 2024 from 10.0.2.2
root@syzkaller:~# ./poc
1322+1 records in
1322+1 records out
677337 bytes (677 kB, 661 KiB) copied, 0.00858991 s, 78.9 MB/s
_poc
setup.sh
[+] Dropping into network namespace
Choose an option:
  1. Leak kernel TEXT address and regs address
  2. Run the exploit
[+] Setting up the network namespace environment
[+] Created table mytable
[+] Created base chain base_chain
```

Web 服务 — 漏洞信息 + 复现环境展示



漏洞信息

CVE编号:
CVE-2016-9560

漏洞类别:
jasper

漏洞版本:
4786a1392bb13013ac1ca9020096f48abdac6107

漏洞证明:
https://github.com/asarubbo/poc/raw/master/00047-jasper-stackoverflow-jpc_tsfb_getbands2

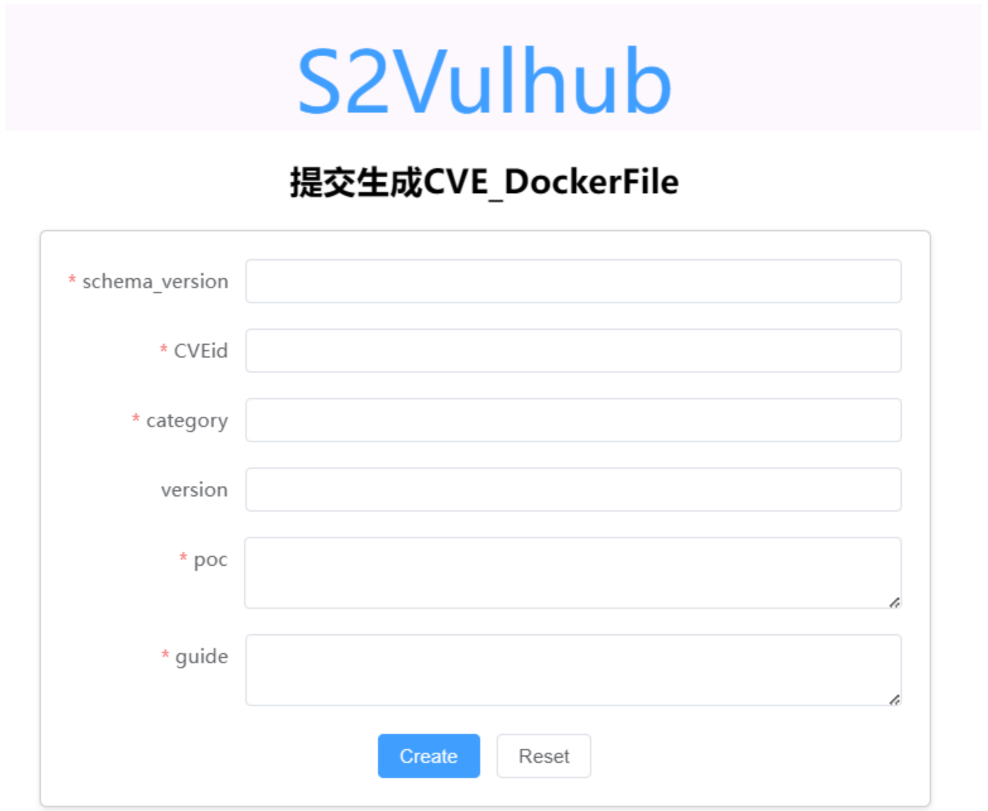
漏洞复现:
./src/appl/imginfo -f 00047-jasper-stackoverflow-jpc_tsfb_getbands2

```
=================================================================
==5903==ERROR: AddressSanitizer: stack-buffer-overflow on address 0x7ffc44402b00 at pc 0x7fc92a1c86ac bp 0x7ffc44401b70 sp 0x7ff
c44401b60
WRITE of size 4 at 0x7ffc44402b00 thread T0
    #0 0x7fc92a1c86ab in jpc_tsfb_getbands2 (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x3bc6ab)
    #1 0x7fc92a1c85fb in jpc_tsfb_getbands2 (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x3bc5fb)
    #2 0x7fc92a1c7d76 in jpc_tsfb_getbands (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x3bbd76)
    #3 0x7fc92a1097fa in jpc_dec_tileinit (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2fd7fa)
    #4 0x7fc92a10756e in jpc_dec_process_sod (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2fb56e)
    #5 0x7fc92a105c4d in jpc_dec_decode (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2f9c4d)
    #6 0x7fc92a104c6b in jpc_decode (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2f8c6b)
    #7 0x7fc92a0950e7 in jas_image_decode (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2890e7)
    #8 0x401312 in main (/root/jasper/src/appl/.libs/lt-imginfo+0x401312)
    #9 0x7fc928d5583f in __libc_start_main (/lib/x86_64-linux-gnu/libc.so.6+0x2083f)
    #10 0x400f18 in _start (/root/jasper/src/appl/.libs/lt-imginfo+0x400f18)

Address 0x7ffc44402b00 is located in stack of thread T0 at offset 3104 in frame
    #0 0x7fc92a108804 in jpc_dec_tileinit (/root/jasper/src/libjasper/.libs/libjasper.so.1+0x2fc804)

  This frame has 1 object(s):
    [32, 3104) 'bnds' <== Memory access at offset 3104 overflows this variable
HINT: this may be a false positive if your program uses some custom stack unwind mechanism or swapcontext
      (longjmp and C++ exceptions *are* supported)
SUMMARY: AddressSanitizer: stack-buffer-overflow ??:0 jpc_tsfb_getbands2
Shadow bytes around the buggy address:
  0x100008878510: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x100008878520: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x100008878530: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x100008878540: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x100008878550: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
=>0x100008878560:[f3]f3 f3 f3 f3 f3 f3 f3 00 00 00 00 00 00 00 00
  0x100008878570: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x100008878580: 00 00 00 00 00 00 00 00 00 00 f1 f1 f1 f1 00 04
  0x100008878590: f4 f4 f3 f3 f3 f3 00 00 00 00 00 00 00 00 00 00
  0x1000088785a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  0x1000088785b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Shadow byte legend (one shadow byte represents 8 application bytes):
  Addressable:           00
  Partially addressable: 01 02 03 04 05 06 07
  Heap left redzone:       fa
  Heap right redzone:      fb
  Freed heap region:       fd
  Stack left redzone:      f1
  Stack mid redzone:       f2
  Stack right redzone:     f3
  Stack partial redzone:   f4
  Stack after return:      f5
  Stack use after scope:   f8
  Global redzone:          f9
  Global init order:       f6
  Poisoned by user:        f7
  Container overflow:      fc
  Array cookie:            ac
  Intra object redzone:    bb
  ASan internal:           fe
==5903==ABORTING
root@8b88809f258a:~/jasper#
```

开源贡献

HUST OPEN ATOM

网站提交

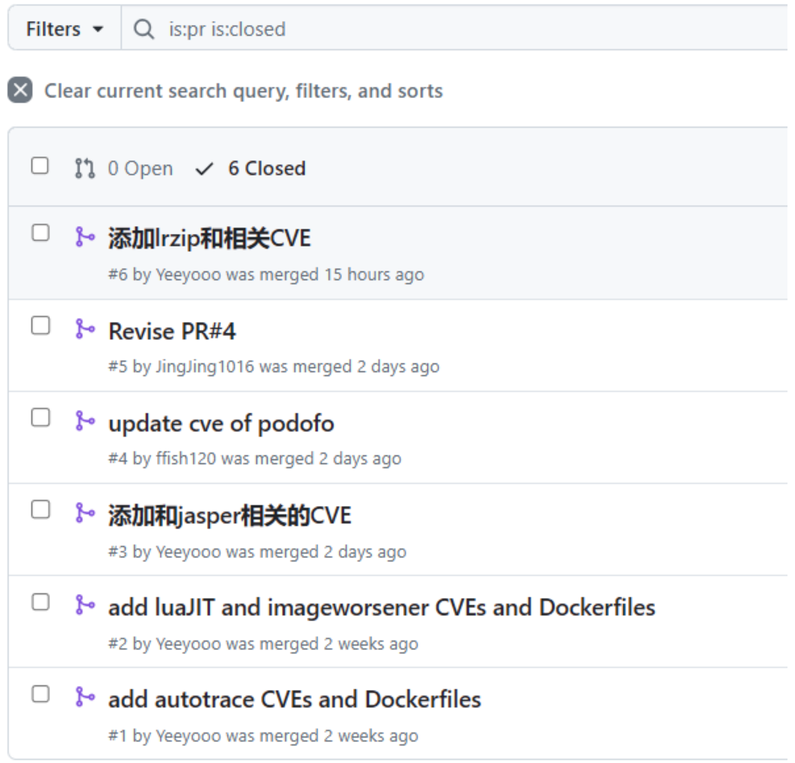

S2Vulhub

提交生成CVE_DockerFile

* schema_version	
* CVEid	
* category	
version	
* poc	
* guide	

Github 提交 PR

https://github.com/hust-open-atom-club/S2VulnHub



is:pr is:closed

Clear current search query, filters, and sorts

0 Open ✓ 6 Closed

添加lrzip和相关CVE
#6 by Yeeyooo was merged 15 hours ago

Revise PR#4
#5 by JingJing1016 was merged 2 days ago

update cve of podofo
#4 by ffish120 was merged 2 days ago

添加和jasper相关的CVE
#3 by Yeeyooo was merged 2 days ago

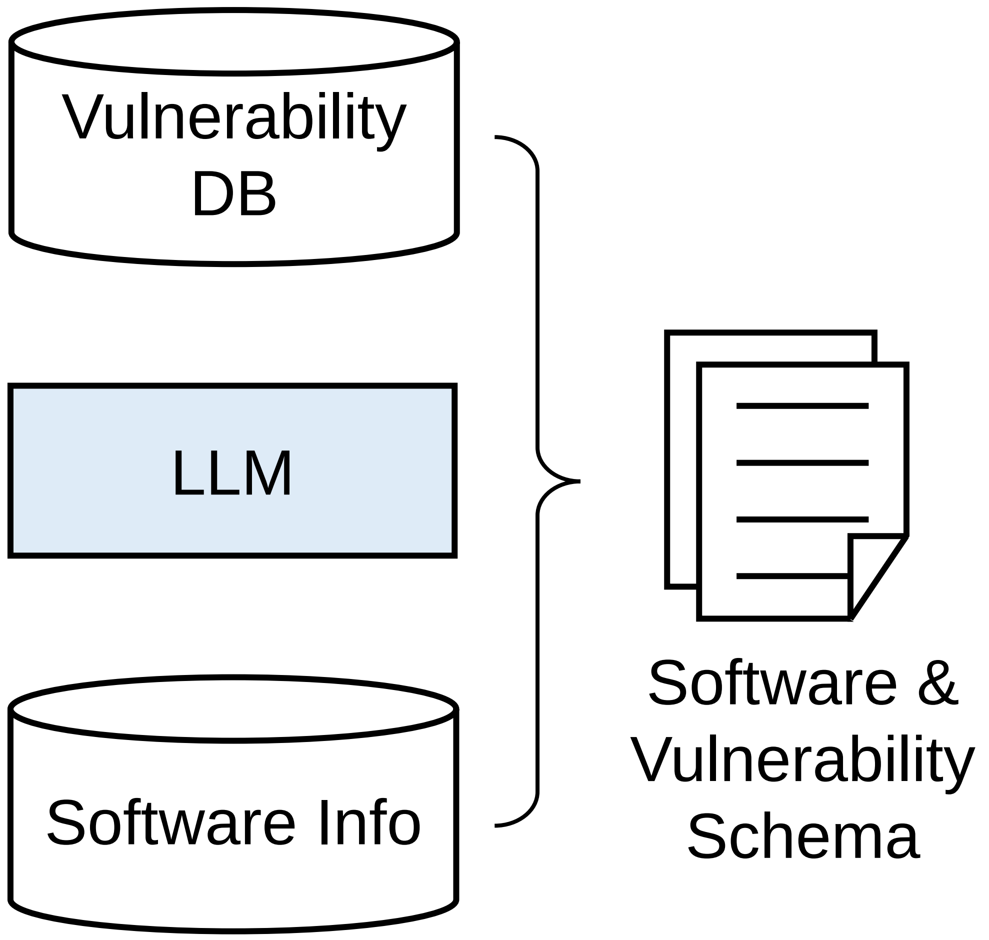
add luaJIT and imageworsener CVEs and Dockerfiles
#2 by Yeeyooo was merged 2 weeks ago

add autotrace CVEs and Dockerfiles
#1 by Yeeyooo was merged 2 weeks ago

应用LLM

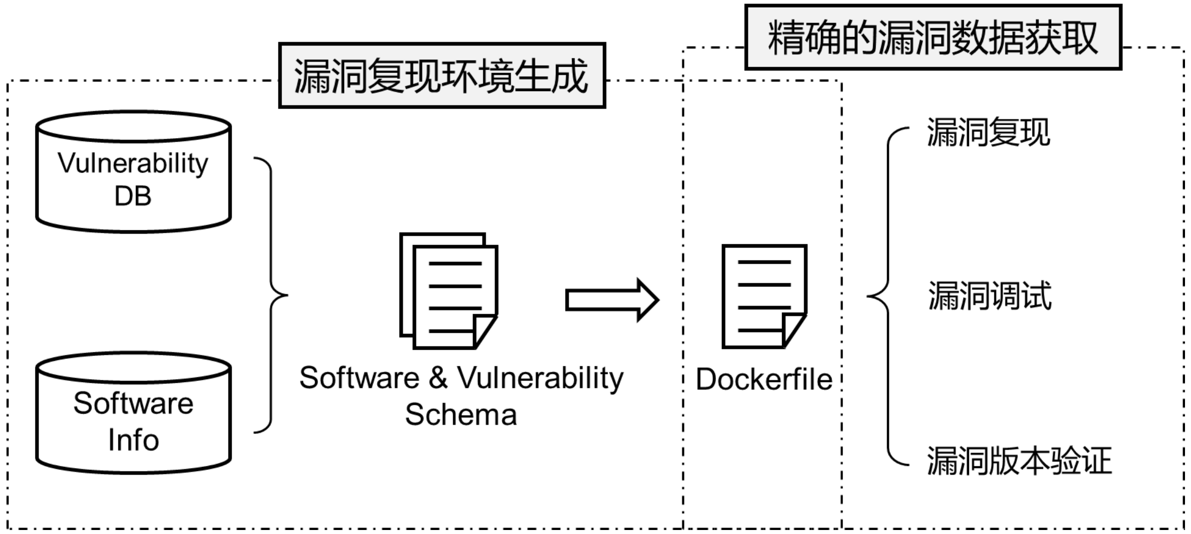
向大模型提问得到漏洞相关信息

- 漏洞的描述信息，漏洞类型，软件版本
- 软件依赖
- 漏洞 PoC



总结

- 用户态、内核态漏洞自动复现，大模型辅助
- 验证漏洞存在性，确定漏洞影响范围
- 精确漏洞数据库信息，保障软件供应链安全



https://github.com/hust-open-atom-club/S2VulnHub

谢谢！