The Industrial Age of Hacking

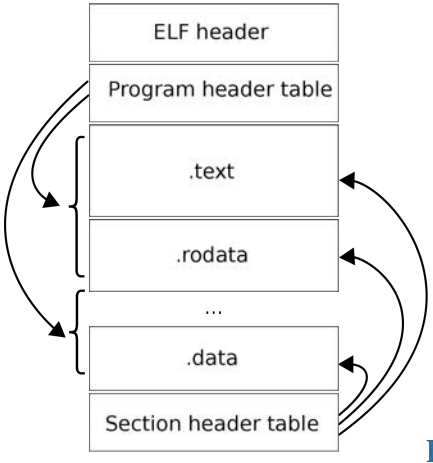
Timothy Nosco[1]
Jared Ziegler[2]
Zechariah Clark[1]
Davy Marrero[1]
Todd Finkler[1]
Andrew Barbarello[1]
W. Michael Petullo[1]

[1]United States Cyber Command
Fort Meade, Maryland USA

[2]National Security Agency
Fort Meade, Maryland USA

July 13, 2020

Wouldn't it be great if everyone knew all of this?

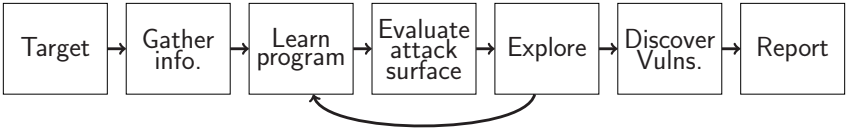


Is it all required?


CGC
CYBER
GRAND_CHALLENGE
CYBER
DARPA
CYBER
DARPA
CYBER


UNITED STATES CYBER COMMAND
24
Mission Support

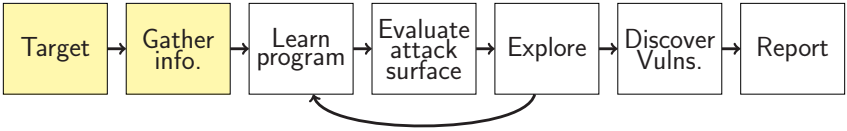
The hacking process

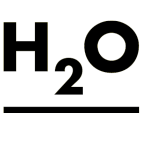


Hackers vs. Testers: A Comparison of Software Vulnerability Discovery Processes

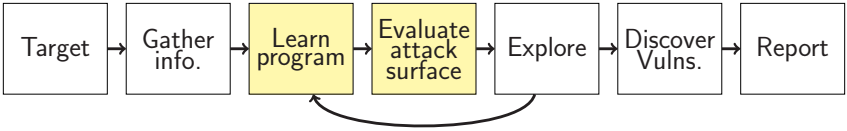
Daniel Votipka, Rock Stevens, Elissa M. Redmiles, Jeremy Hu, and Michelle L. Mazurek
Department of Computer Science
University of Maryalnd
College Park, Maryland 20742
Email: {dvotipka,rstevens,eredmiles,jhu,mmazurek}@cs.umd.edu

Targeting and information gathering



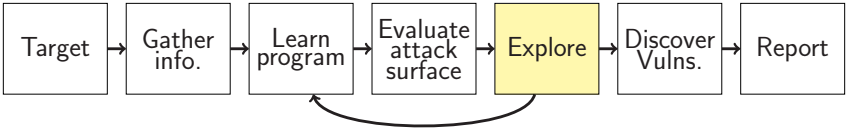


Program understanding and attack surface analysis



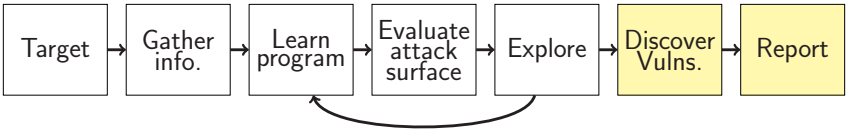
- Identify program's functionality.
- Rehost, emulate, or run.
- Prepare the program for fuzzing.

Exploration




```
/bin/bash
--------------------[  0 days 00 hrs 14 mins 00 secs ]-------/ honggfuzz 1.3 /-
  Iterations : 398,052 [398.05k]
        Mode : Feedback Driven Mode (2/2)
      Target : './httpd/httpd -X -f /home/jagger/fuzz/apache/dist/conf/h ...'
     Threads : 8, CPUs: 8, CPU%: 261% (32%/CPU)
       Speed : 323/sec (avg: 473)
     Crashes : 90 (unique: 1, blacklist: 0, verified: 0)
    Timeouts : [5 sec] 32
 Corpus Size : entries: 1,147, max size: 1,048,792, input dir: 8522 files
  Cov Update : 0 days 00 hrs 00 mins 05 secs ago
    Coverage : edge: 17,019 pc: 410 cmp: 187,266
---------------------------------- [ LOGS ] -----------------------------------
Crash (dup): './SIGABRT.PC.7ffff5ef10bb.STACK.18819c8652.CODE.-6.ADDR.(nil).INST
R.mov____0x108(%rsp),%rcx.fuzz' already exists, skipping
[2018-01-18T22:21:22+0100][W][3343] arch_checkWait():308 Persistent mode: PID 21
623 exited with status: SIGNALED, signal: 6 (Aborted)
Persistent mode: Launched new persistent PID: 24520
Crash (dup): './SIGABRT.PC.7ffff5ef10bb.STACK.18819c8652.CODE.-6.ADDR.(nil).INST
R.mov____0x108(%rsp),%rcx.fuzz' already exists, skipping
[2018-01-18T22:21:23+0100][W][3346] arch_checkWait():308 Persistent mode: PID 18
231 exited with status: SIGNALED, signal: 6 (Aborted)
Persistent mode: Launched new persistent PID: 25094
Size:296441 (i,b,hw,edge,ip,cmp): 0/0/0/0/0/1, Tot:0/0/0/17019/410/187266
```

Vulnerability recognition and reporting



- Explore corpus for bugs: crashes, ASan, valgrind errors.
- Prioritize, filter, and deduplicate.
- Write a report that indicates severity: likelihood of vulnerability, projected investment to convert bug into an exploit.

Combining hackers with machines

Human and machine working together, but how?

The prevailing method: depth-first search

CAUTION: Diamond Mining

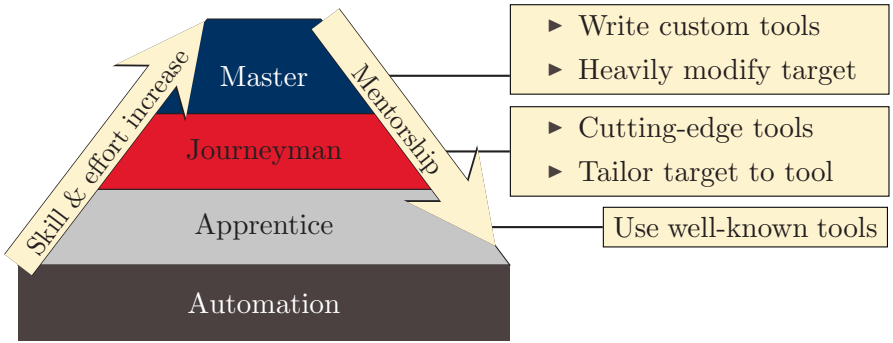
The problem

$$R = \frac{\mathbf{T} \times \mathbf{S}}{\mathbf{L} \times \mathbf{V}}$$

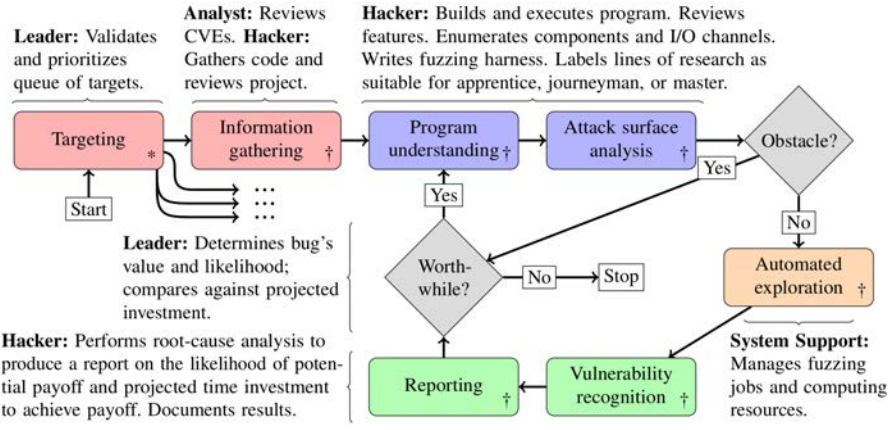
Increases **R**isk:	Decreases **R**isk:
Projected **T**ime investment	**L**iklihood of success
Required **S**kill level	**V**alue of success

A deliberate risk formula

Our method: breadth-first search

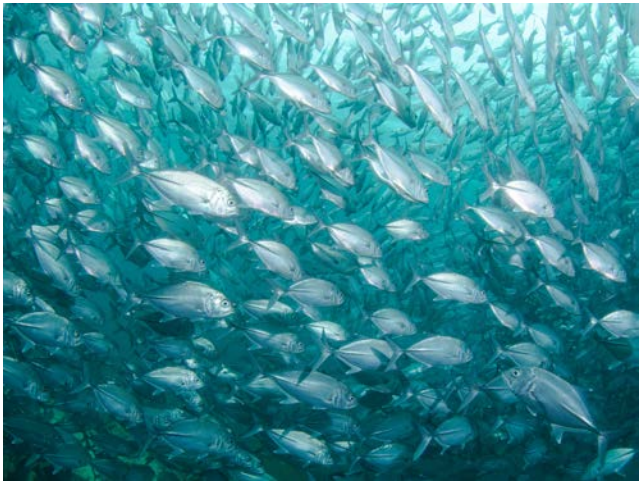


Our method: breadth-first search



Our vulnerability-discovery process adds targeting (*) to the steps of Votipka, et al. (†)

Metaphor: fishing For bugs



There are fish out there. How do we best catch them?

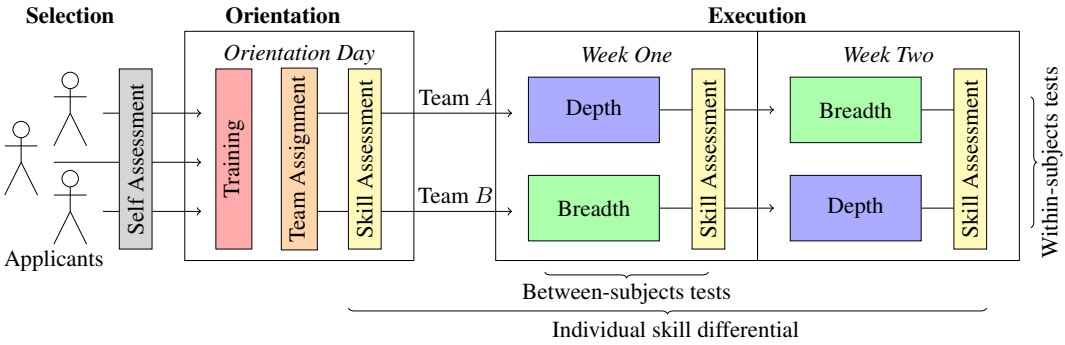
Metaphor: fishing For bugs

Larger holes in net $\implies$ less friction.

Metaphor: fishing For bugs

Some fish might escape, but we cover more area.

Experimental design



Target selection

Google

?

Target selection

Something else entirely

Strict schedules

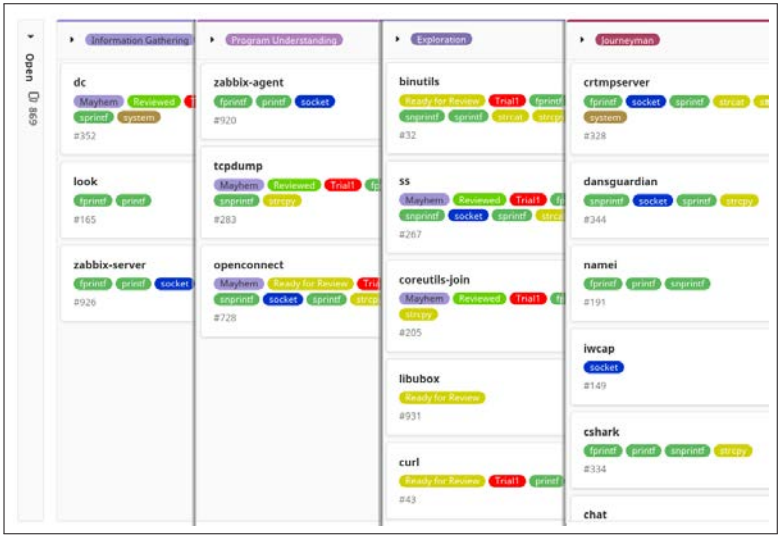
Workflow

Target Information gathering

Program understanding Attack surface

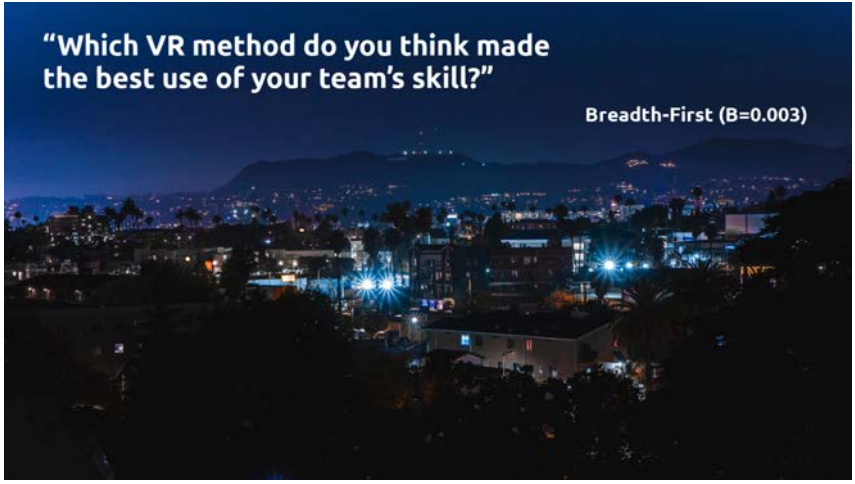
Automated exploration

Promote to journeyman



Results: surveys

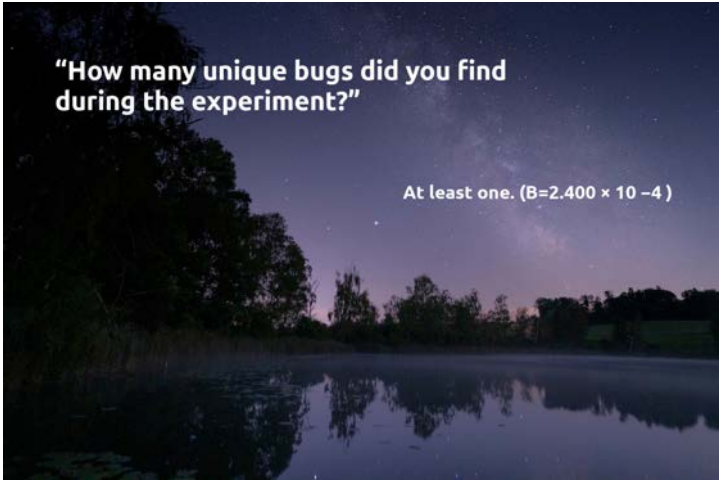
Results: surveys



Results: surveys

Results: surveys

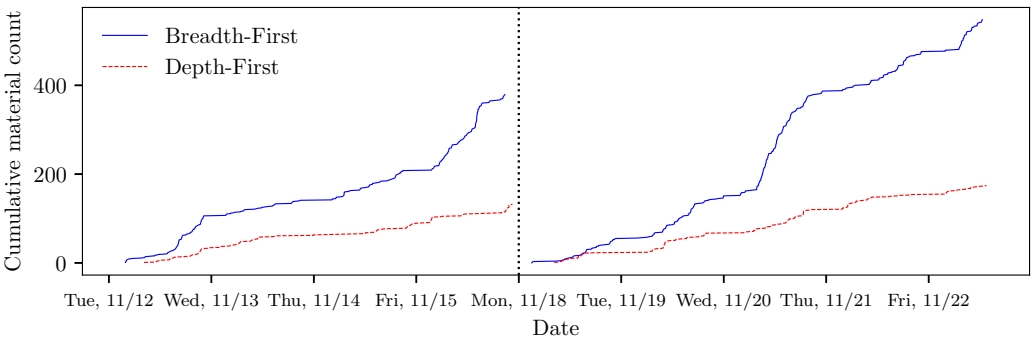
Results: surveys



Results: bugs found

Team	Method	Harnesses	T_0	T_1	T_2
A	S_D	8	3	2	3
A	S_B	42	31	23	40
B	S_B	61	42	49	40
B	S_D	12	4	4	4

Results: documentation produced



Conclusion

We described a repeatable experiment for measuring a novel workflow that:

- efficiently uses human resources, both novice and expert,
- **finds more bugs**,
- produces more documentation and learning resources,
- better applies automated bug-finding tools, and
- clearly defines work roles.

Tim Nosco: usenix@jocular.us