The Industrial Age of Hacking

Timothy Nosco\textsuperscript{1}
Jared Ziegler\textsuperscript{2}
Zechariah Clark\textsuperscript{1}
Davy Marrero\textsuperscript{1}
Todd Finkler\textsuperscript{1}
Andrew Barbarello\textsuperscript{1}
W. Michael Petullo\textsuperscript{1}

\textsuperscript{1}United States Cyber Command
Fort Meade, Maryland USA

\textsuperscript{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?
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 Maryland
College Park, Maryland 20742
Email: {dvotipka,rstevens,eredmiles,jhu,mazurek}@cs.umd.edu
Targeting and information gathering

1. Target
2. Gather info.
3. Learn program
4. Evaluate attack surface
5. Explore
6. Discover Vulns.
7. Report
Program understanding and attack surface analysis

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

Target → Gather info. → Learn program → Evaluate attack surface → Explore

Discover Vulns. → Report

/bin/bash

Iterations: 398,052 [398.65k]
Mode: Feedback Driven Mode (2/2)
Target: '/httpd/httpd -X -f /home/jagger/fuzz/apache/dist/conf/h ...'
Threads: 8, CPUs: 8, CPU: 26% (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 cnp: 187,266

Crash (dup): '/SIGABRTocaust.7fffffff5ef10bb.STACK.18819c8652.CODE.-6.ADDR.(nil).INST R.mov___0x108(%rsp),%rcx.fuzz' already exists, skipping
 Persistent mode: Launched new persistent PID: 24529
Crash (dup): '/SIGABRT.rc.7fffffff5ef10bb.STACK.18819c8652.CODE.-6.ADDR.(nil).INST R.mov___0x108(%rsp),%rcx.fuzz' already exists, skipping
 Persistent mode: Launched new persistent PID: 25694
Size:296441 (i,b,hw,edge,ip,cnp): 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

\( R = \frac{T \times S}{L \times V} \)

**Increases** Risk:
- Projected **Time** investment
- Required **Skill** level

**Decreases** Risk:
- Likelihood of success
- Value of success

A deliberate risk formula
Our method: breadth-first search

- Write custom tools
- Heavily modify target
- Cutting-edge tools
- Tailor target to tool
- Use well-known tools
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

Selection

Applicants

Self Assessment

Orientation

Orientation Day

Training

Team Assignment

Skill Assessment

Execution

Week One

Team A

Depth

Team B

Breadth

Week Two

Skill Assessment

Breadth

Skill Assessment

Depth

Between-subjects tests

Individual skill differential

Within-subjects tests

Self Assessment

Within-subjects tests

Between-subjects tests

Individual skill differential
Target selection
Target selection

Something else entirely
Strict schedules
“Which VR method do you feel was more effective?”

Breadth-First (B=0.019)
“Which VR method do you think made the best use of your team’s skill?”

Breadth-First (B=0.003)
"Which VR method do you think is easier for a novice to contribute to?"

Breadth-First (B=2.400 × 10^-4)
“Did you learn any valuable skills during the experiment?”

Yes. (B=2.400 \times 10^{-4})
“How many unique bugs did you find during the experiment?”

At least one. (B=2.400 × 10⁻⁴)
## Results: bugs found

<table>
<thead>
<tr>
<th>Team</th>
<th>Method</th>
<th>Harnesses</th>
<th>T₀</th>
<th>T₁</th>
<th>T₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$S_D$</td>
<td>8</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>A</td>
<td>$S_B$</td>
<td>42</td>
<td>31</td>
<td>23</td>
<td>40</td>
</tr>
<tr>
<td>B</td>
<td>$S_B$</td>
<td>61</td>
<td>42</td>
<td>49</td>
<td>40</td>
</tr>
<tr>
<td>B</td>
<td>$S_D$</td>
<td>12</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
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