Vulnerability Discovery

Dr. Edward J. Schwartz

eschwartz@cert.org

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

© 2015 Carnegie Mellon University

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Copyright 2015 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon® and CERT® are registered marks of Carnegie Mellon University.

DM-0002802

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Vulnerability Discovery Project

Increase **assurance** of 1st and 3rd party DoD software through **enhanced vulnerability discovery techniques**

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Team

Software Engineering Institute

- Edward Schwartz, PhD, CERT
- David Warren, CERT
- Allen Householder, CERT

Collaborators

∀.Secure

- David Brumley, PhD
- Thanassis Avgerinos, PhD
- Tyler Nighswander

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Agenda

Towards vulnerability discovery as a science

Intelligent fusion of vulnerability discovery techniques

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

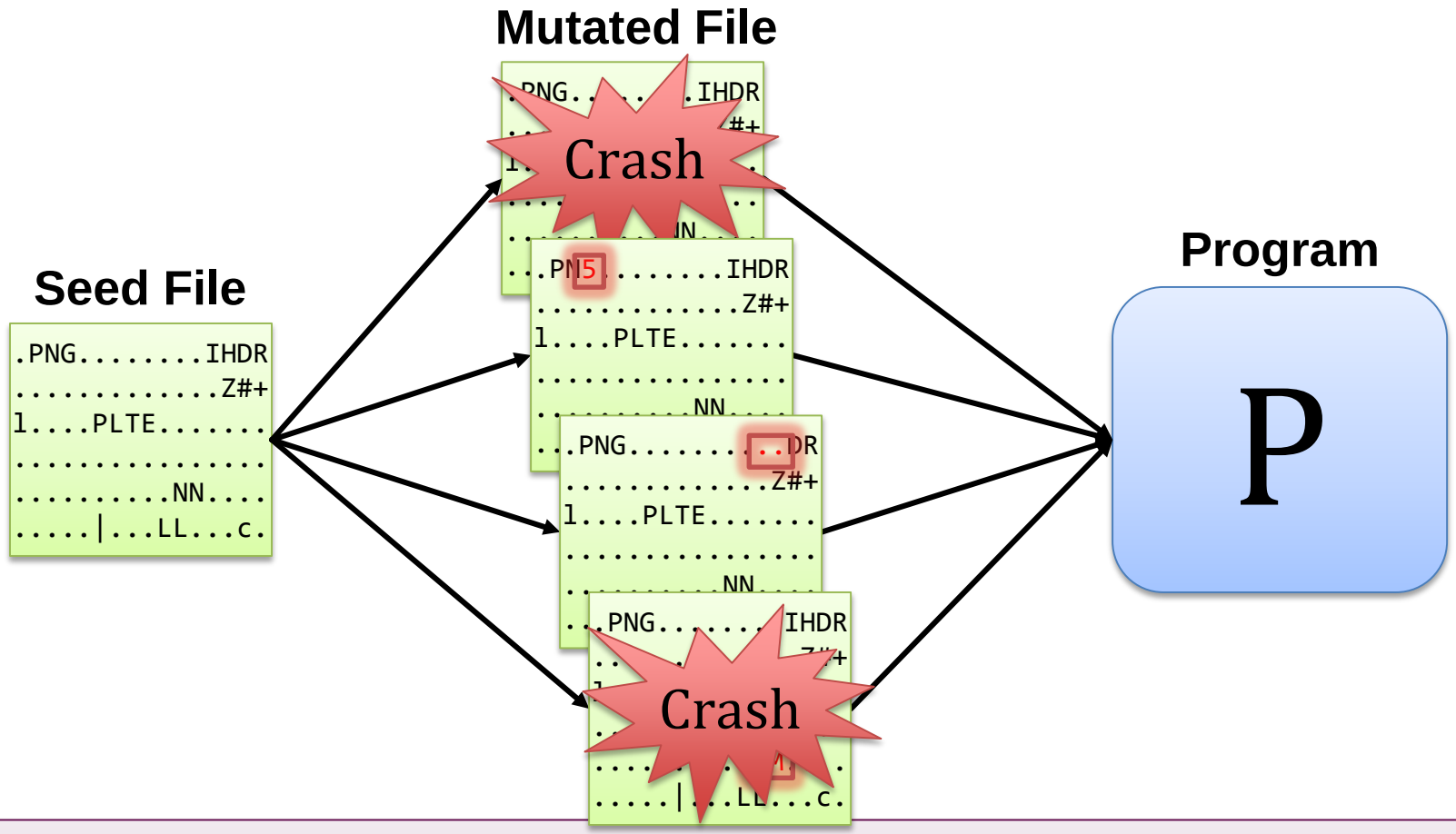
Towards Vulnerability Discovery as a Science

Software Engineering Institute | Carnegie Mellon University

© 2015 Carnegie Mellon University

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

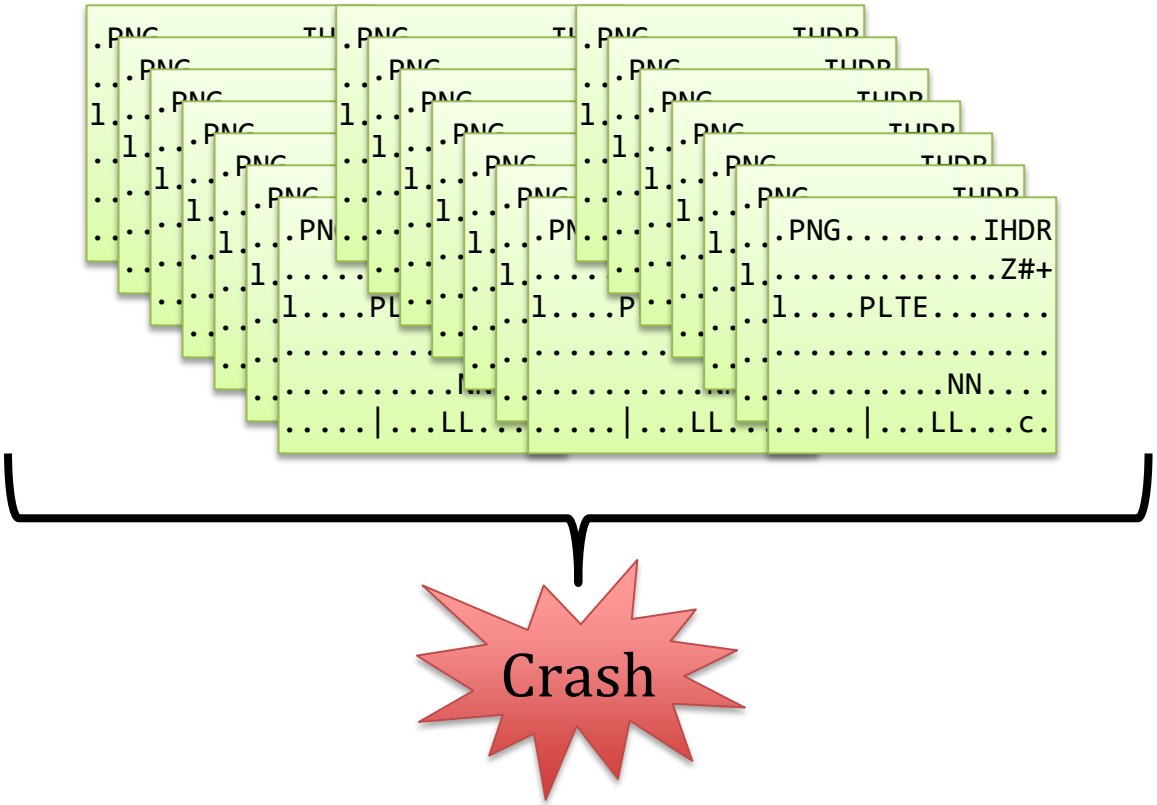
Background: Mutational Fuzzing of Software



Testing of programs by randomly mutating program inputs (seeds)

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Challenge: How Many Software Vulnerabilities are There?



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Problem: Distinguishing One Vulnerability From Another

I don't know how to *specify* a vulnerability, but I know how to *fix* one

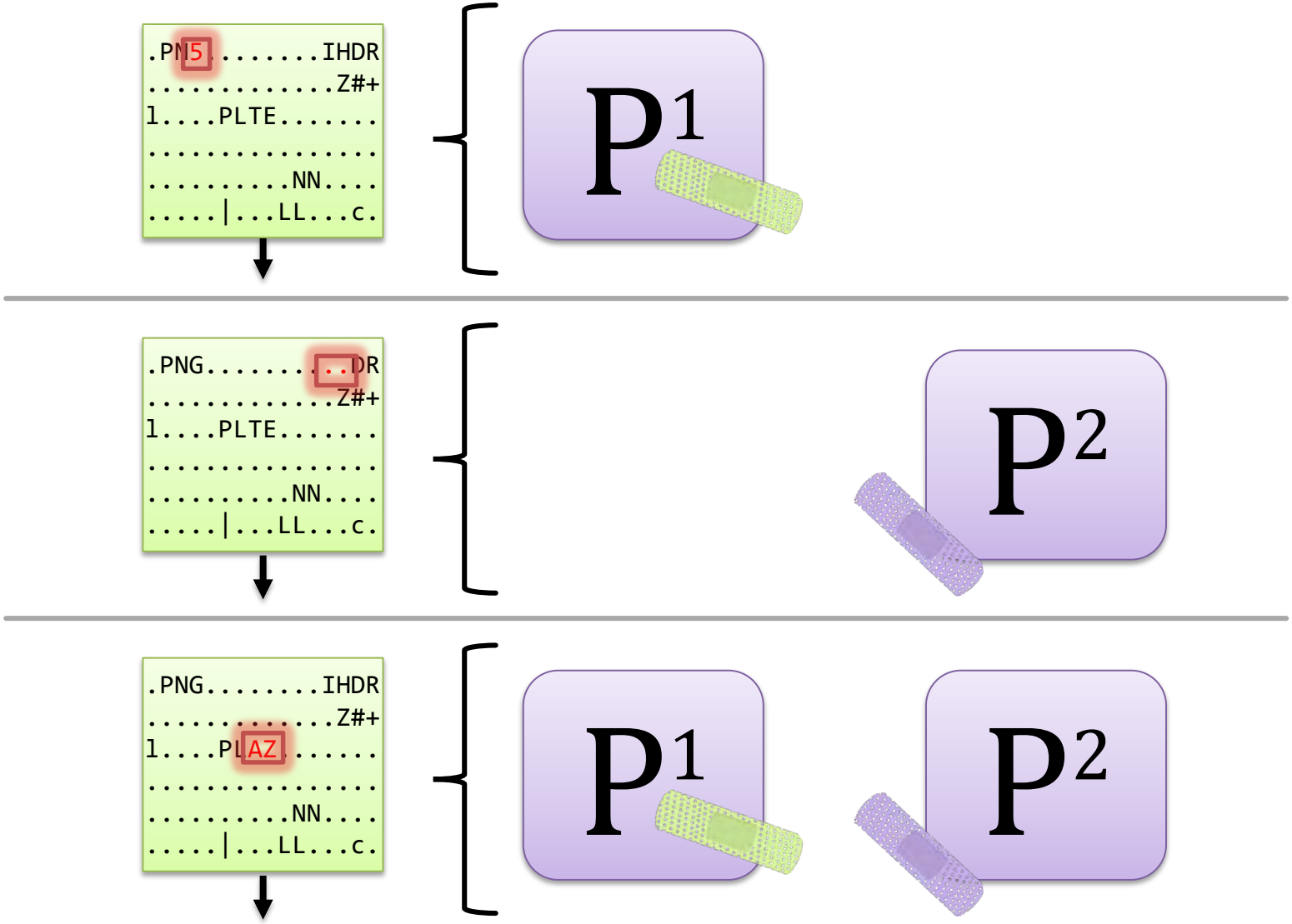
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The Idea: Patches Define Vulnerabilities

Any crash that is fixed by the patch is also affected by vulnerability V

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Example Ground Truth



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Patching ImageMagick

Fuzzed old ImageMagick with the CERT BFF fuzzer

- 1 week
- 130,000 crashes found

Manually patched all vulnerabilities

- Took approximately one month
- 31 patches/vulnerabilities

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

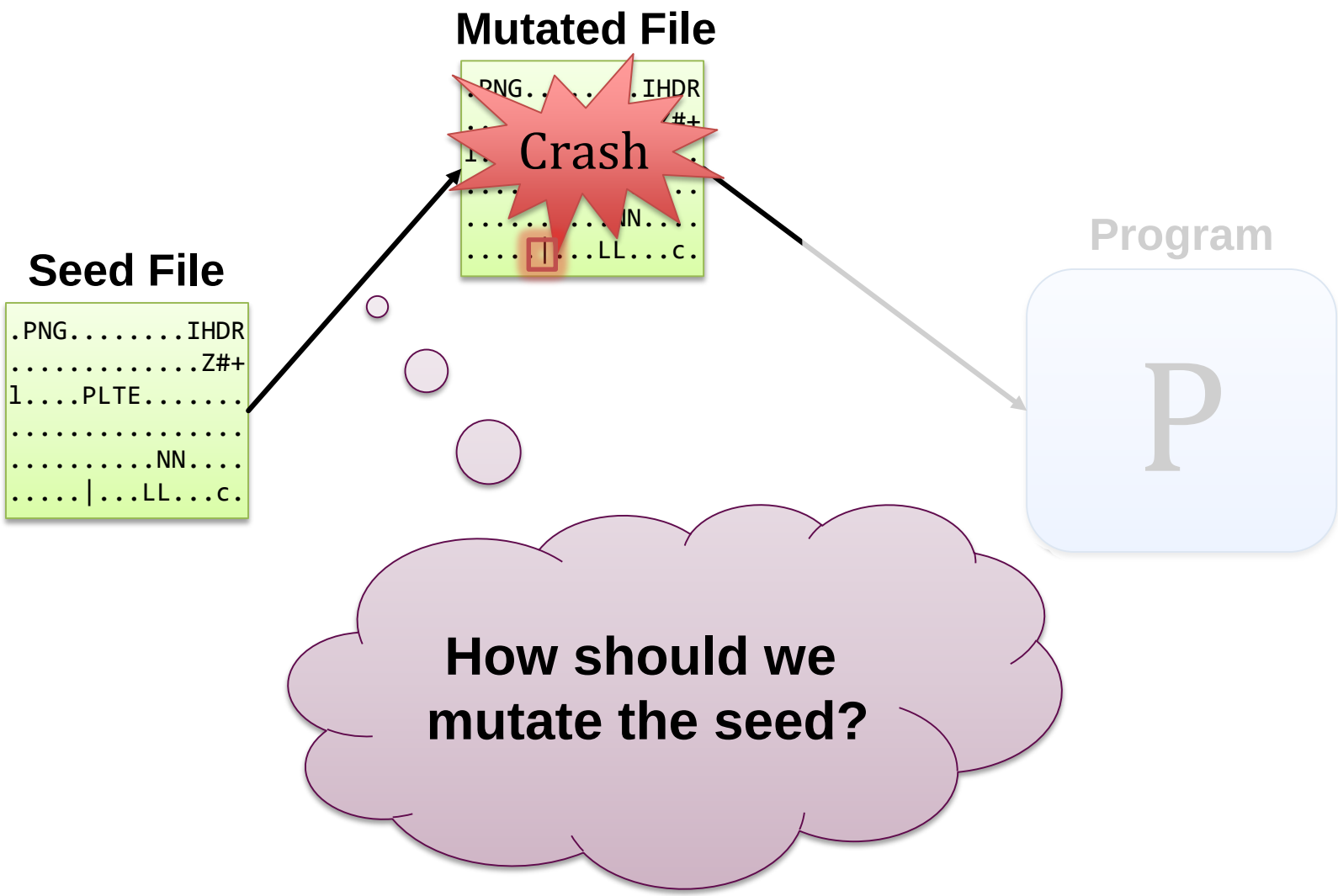
Vulnerability Discovery Science

Analyze fuzzing parameters

- What mutators work best?
- When should we stop fuzzing?
- What effect do compiler settings have?
- …
- Paper submitted to NDSS 2016

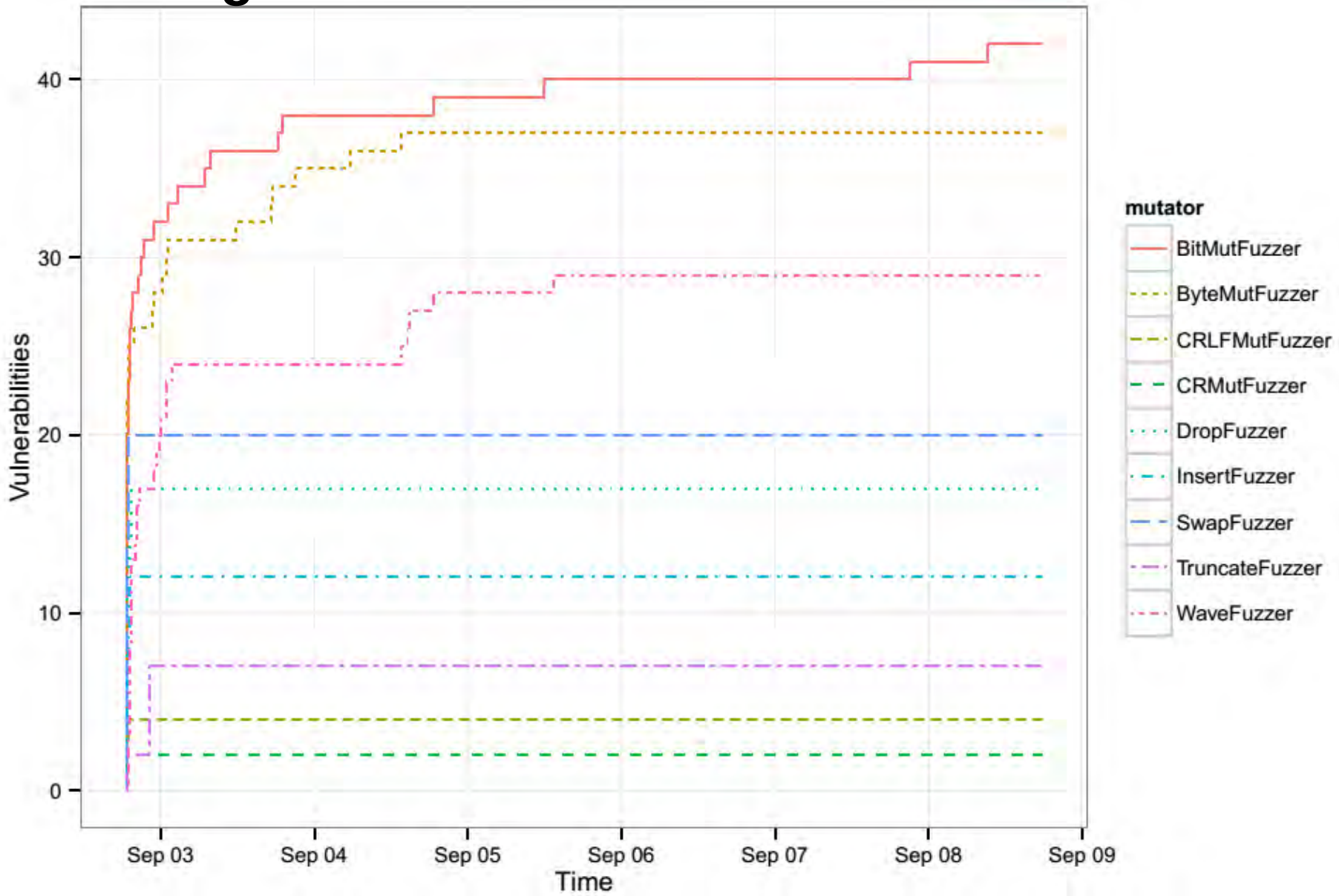
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Background: Mutational Fuzzing of Software



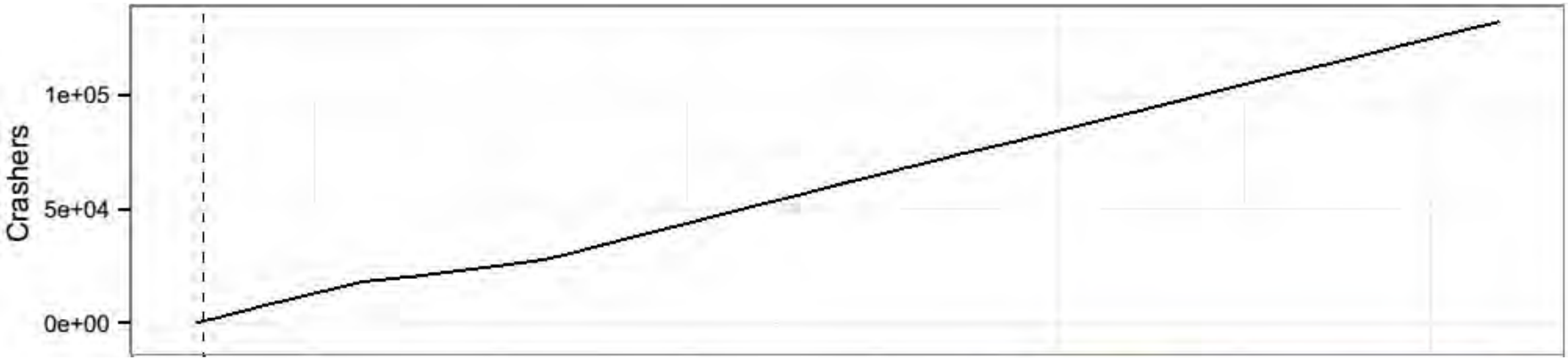
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Fuzzing Mutators



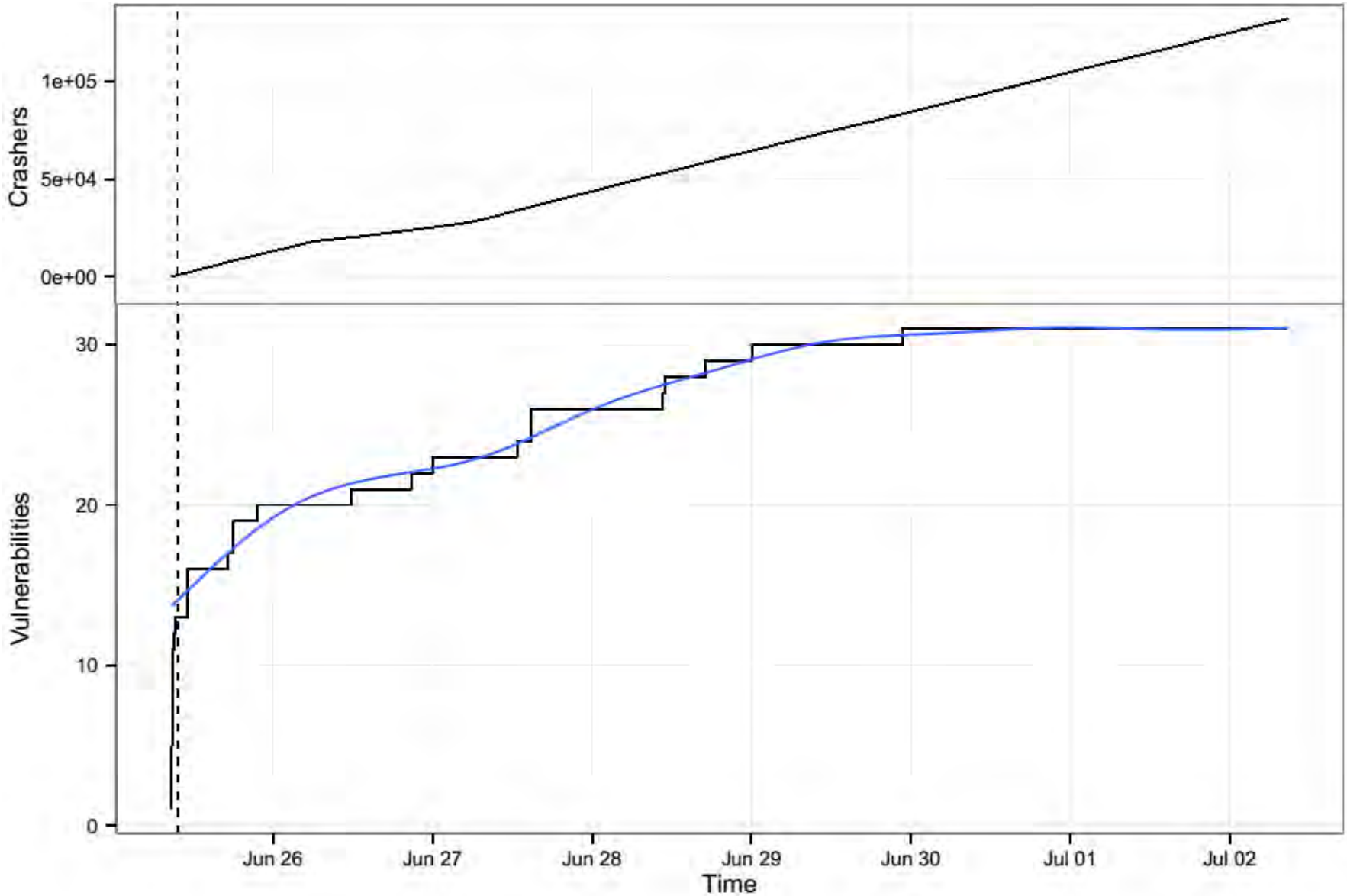
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

When to Stop Fuzzing?



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

When to Stop Fuzzing?



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Collaboration with ForAllSecure

© 2015 Carnegie Mellon University

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

∀.Secure

CMU Spinoff of David Brumley's research group

Dr. David Brumley

Dr. Thanassis Avgerinos

Alex Rebert

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

∀.Secure

Expertise

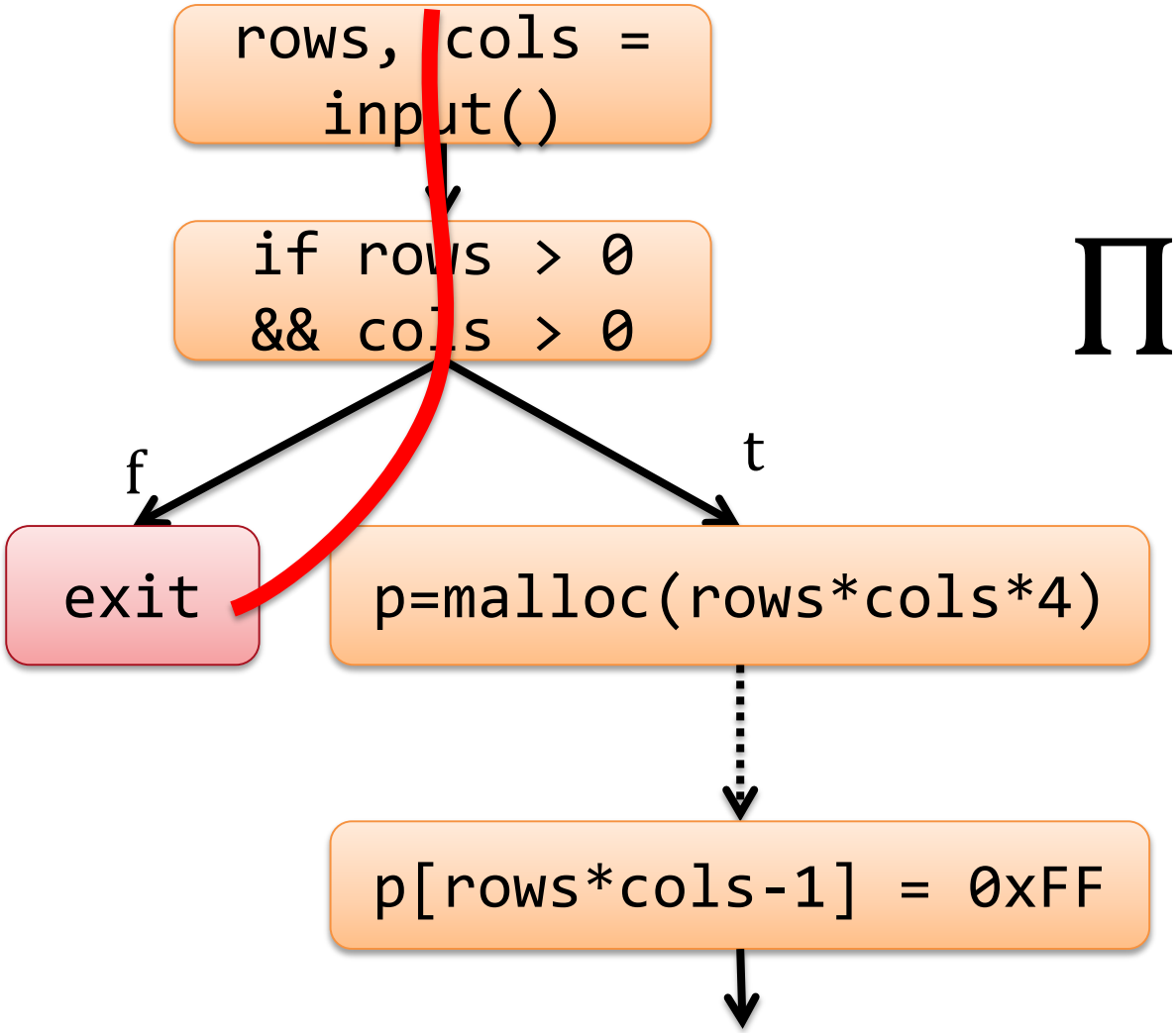
- Concolic execution
- Automatic exploitation
- Binary analysis
- Complements SEI's expertise in fuzzing

Previous collaboration

- With same group at CMU

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Background: Concolic Execution



Π

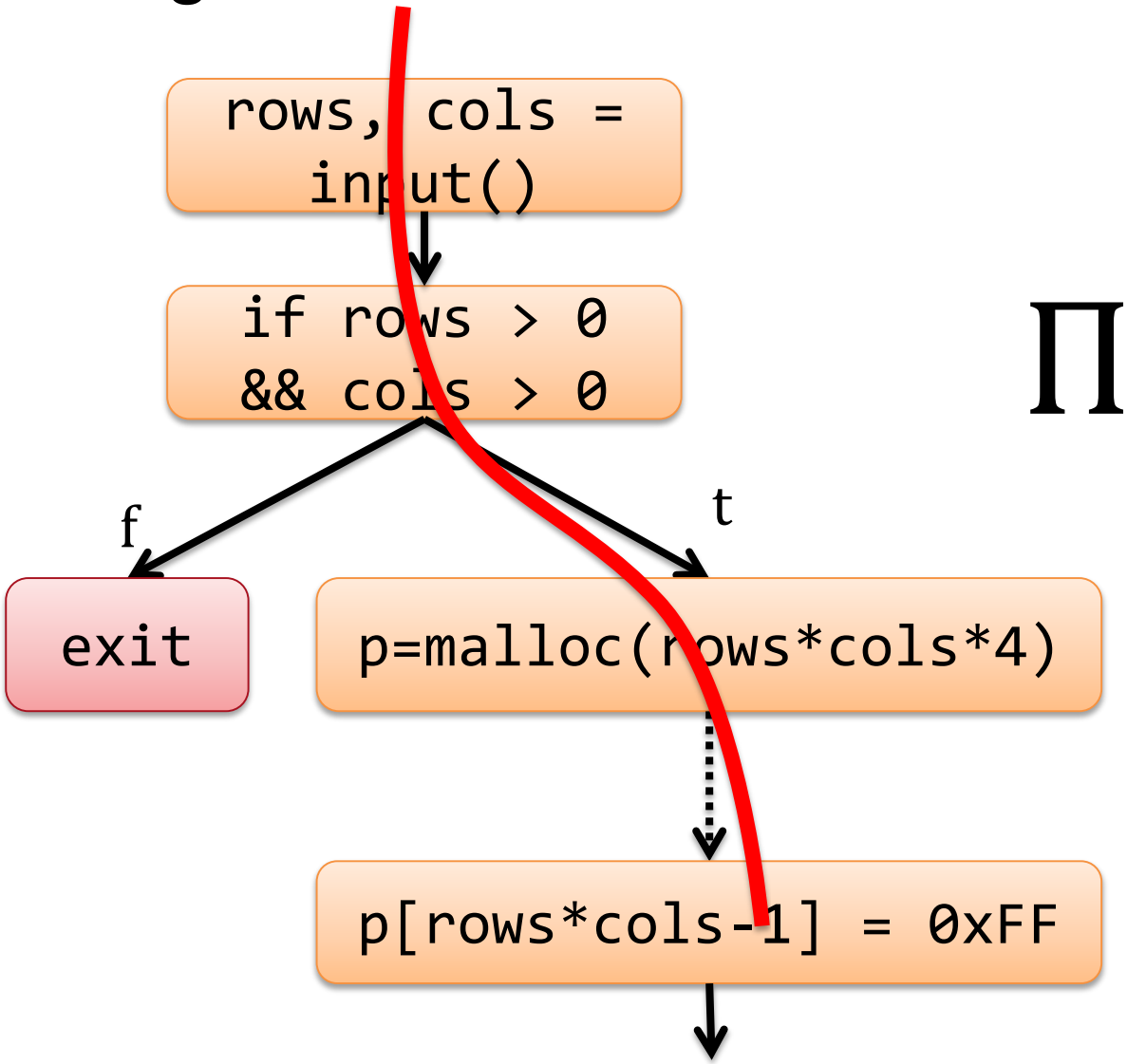
rows=0
cols=0

✖ (rows > 0 && cols > 0)

rows=5
cols=10

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Background: Concolic Execution



Π

rows=0
cols=0

✖ (rows > 0 && cols > 0)

rows=5
cols=10

Overflow?

rows*cols-1 >= rows*cols*4

rows=0x11111112
cols=0x0000000f

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Background: Fuzzing

```
rows, cols =
input()
```

```
if rows > 0
&& cols > 0
```

f

t

```
exit
```

```
p=malloc(rows*cols*4)
```

```
p[rows*cols-1] = 0xff
```

Most mutations do not trigger the overflow vulnerability

```
rows=0x00000014
cols=0x00000014
```

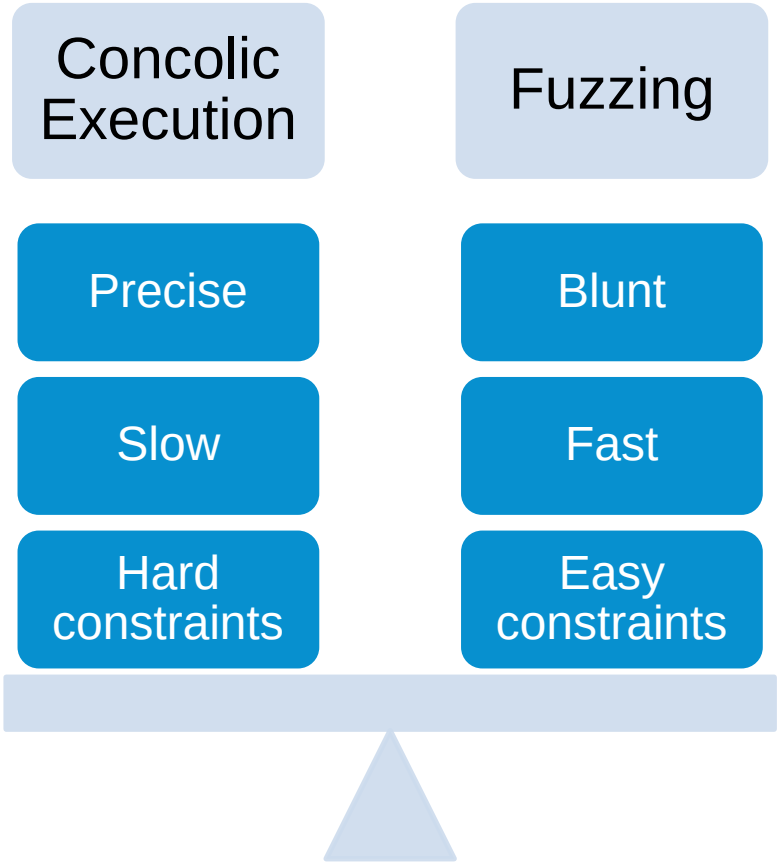
Overflow?

```
0xd040f >=
0x341040
```

No

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Concolic Execution vs. Fuzzing



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

SMART

The Synergistic Mayhem AFL Research Tool

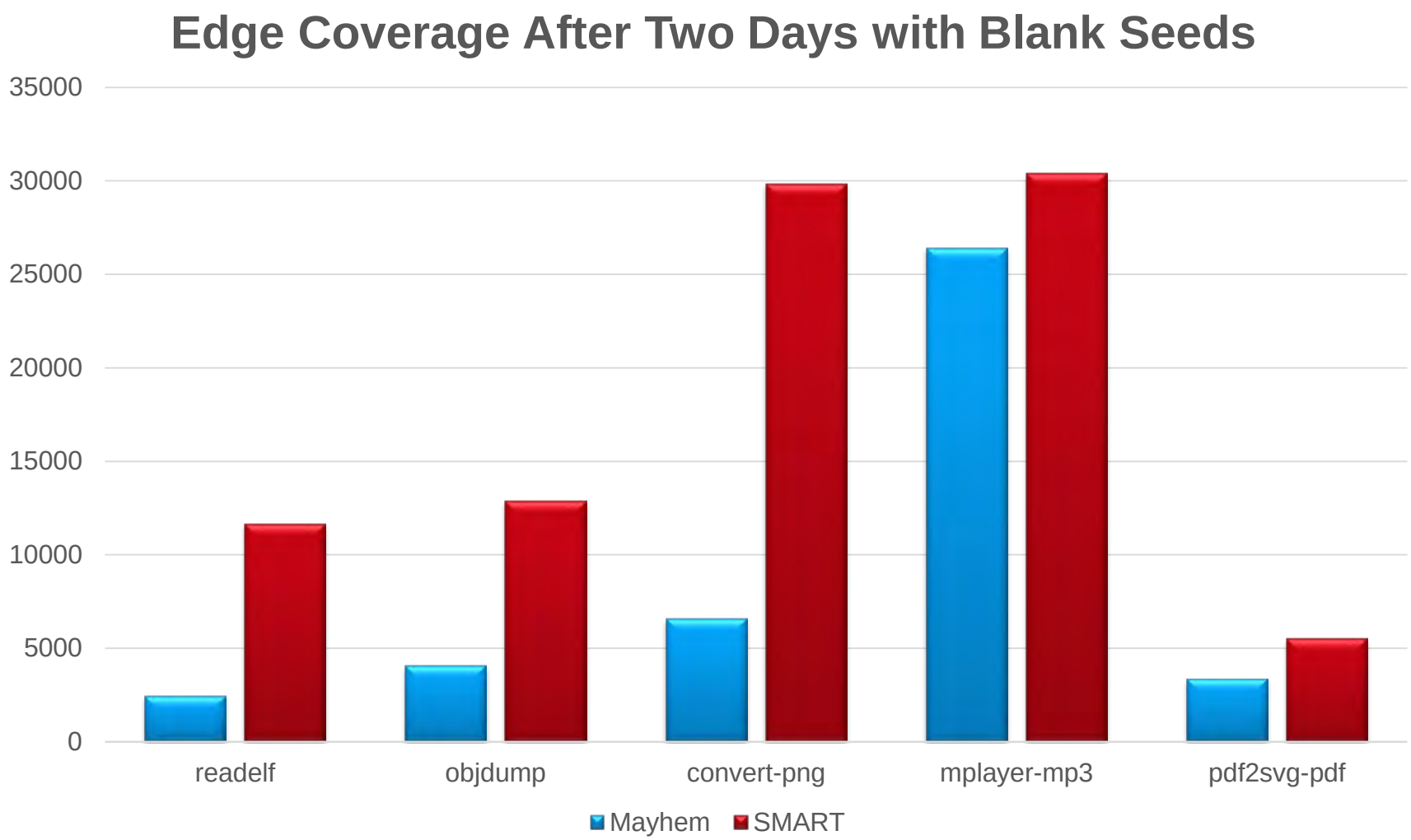
- Concolic execution: Mayhem (ForAllSecure+SEI)
- Fuzzing: AFL
- Periodically synchronize seed files between them

Challenges

1. Where to go?
   - We don’t know the location of vulnerabilities
2. How much should we use concolic execution?
   - ~10^4 times slower than fuzzing
   - Brute force vs. high cost

© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

SMART Evaluation



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Summary

- Developing new techniques for discovering and mitigating vulnerabilities in the DoD
- Developed vulnerability uniqueness model and used ground truth to explore the effect of fuzzing parameters
- ForAllSecure: Hybrid fuzzing and concolic tester

Team Members

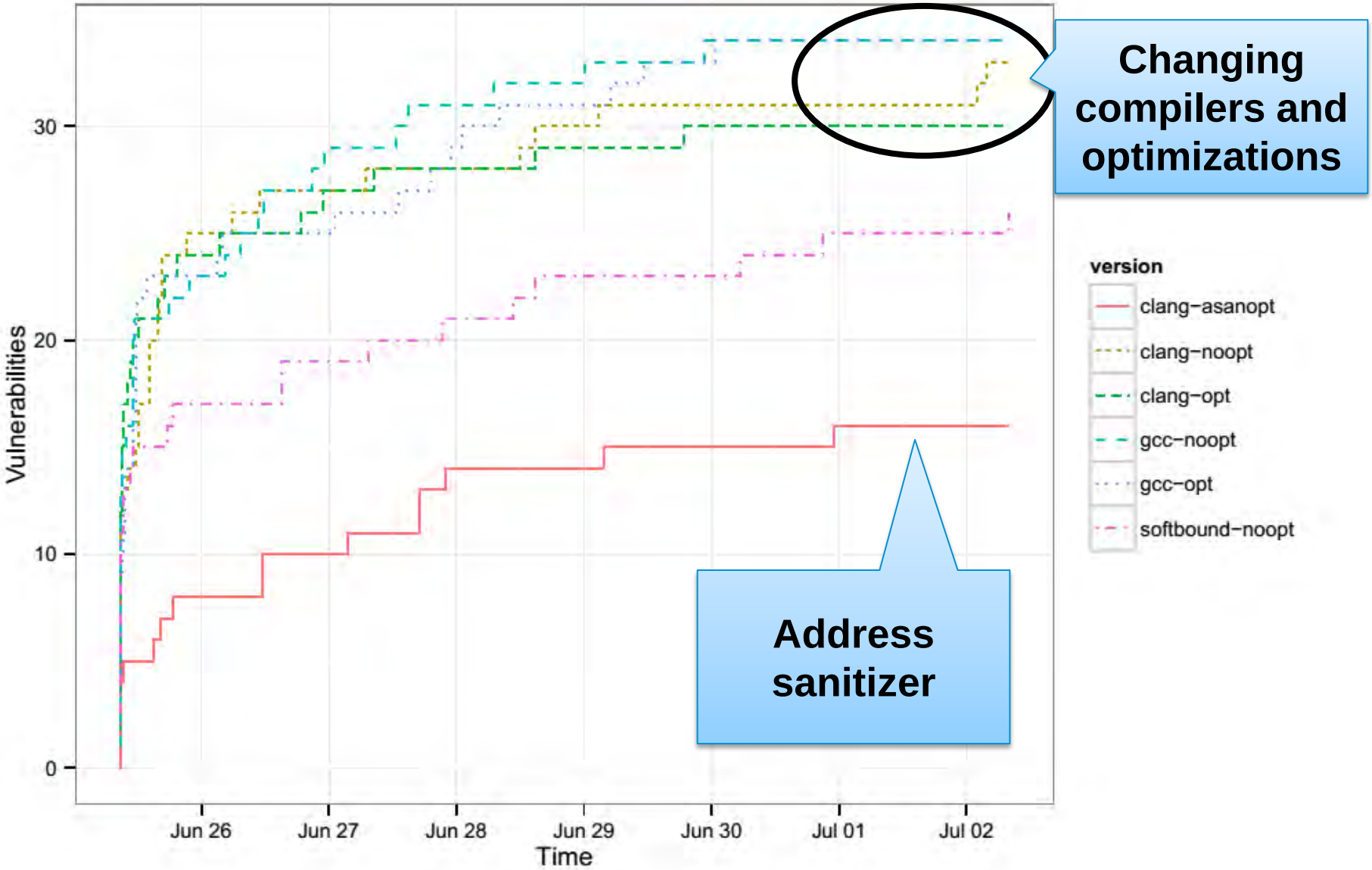
- Edward Schwartz, PhD, CERT
- David Warren, CERT
- Allen Householder, CERT

ForAllSecure, Inc.:

- David Brumley, PhD
- Thanassis Avgerinos, PhD
- Tyler Nighswander

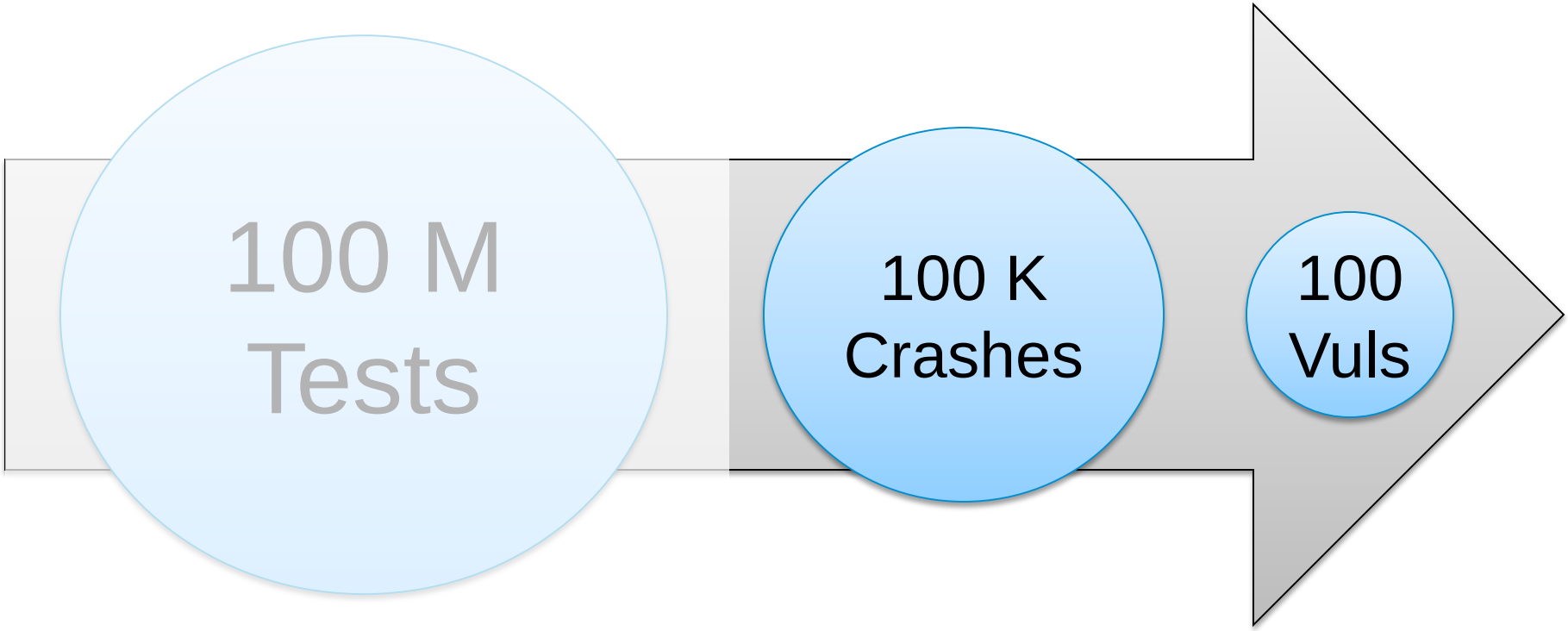
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Compiler Flags and Settings



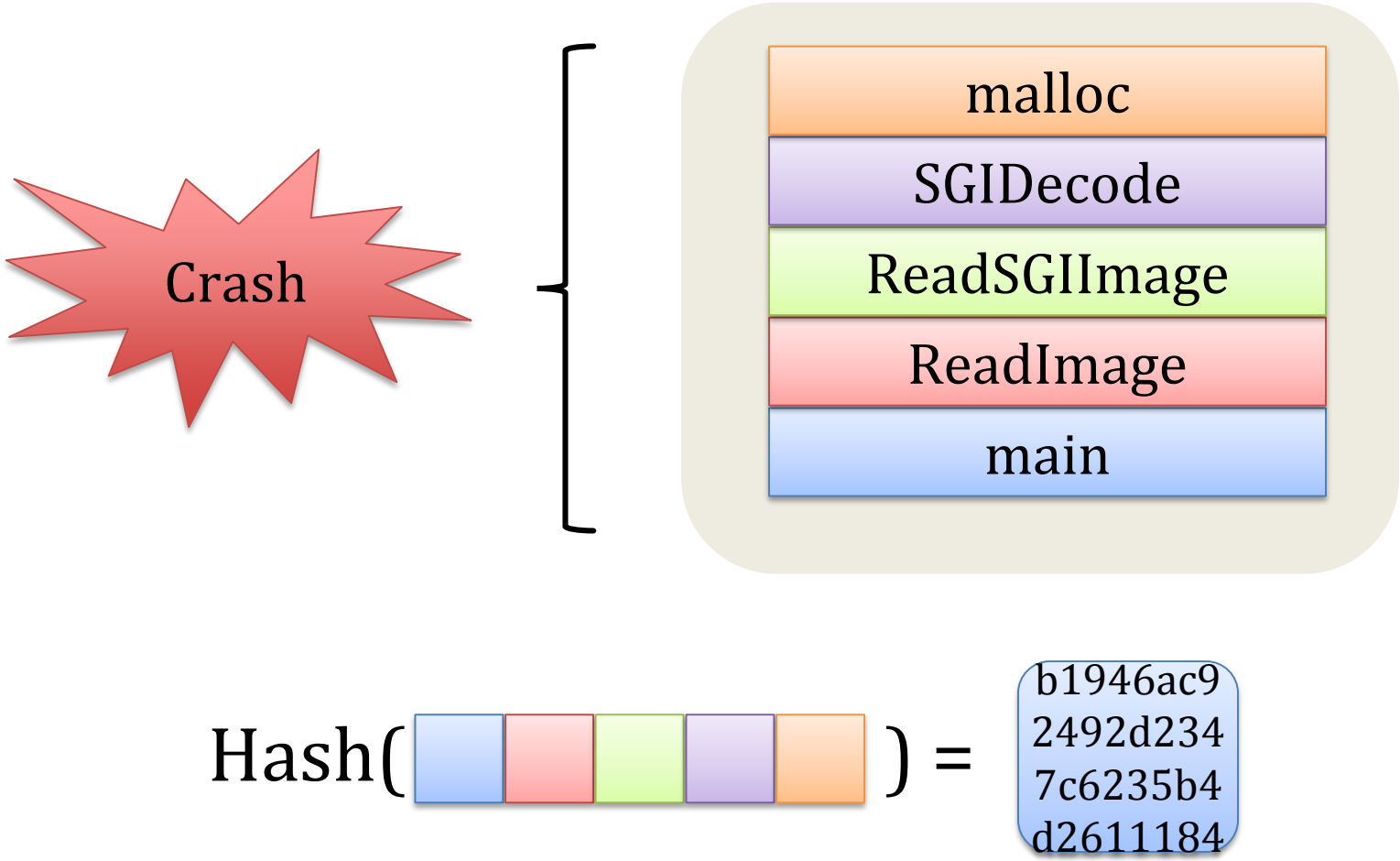
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The Crash Uniqueness Problem



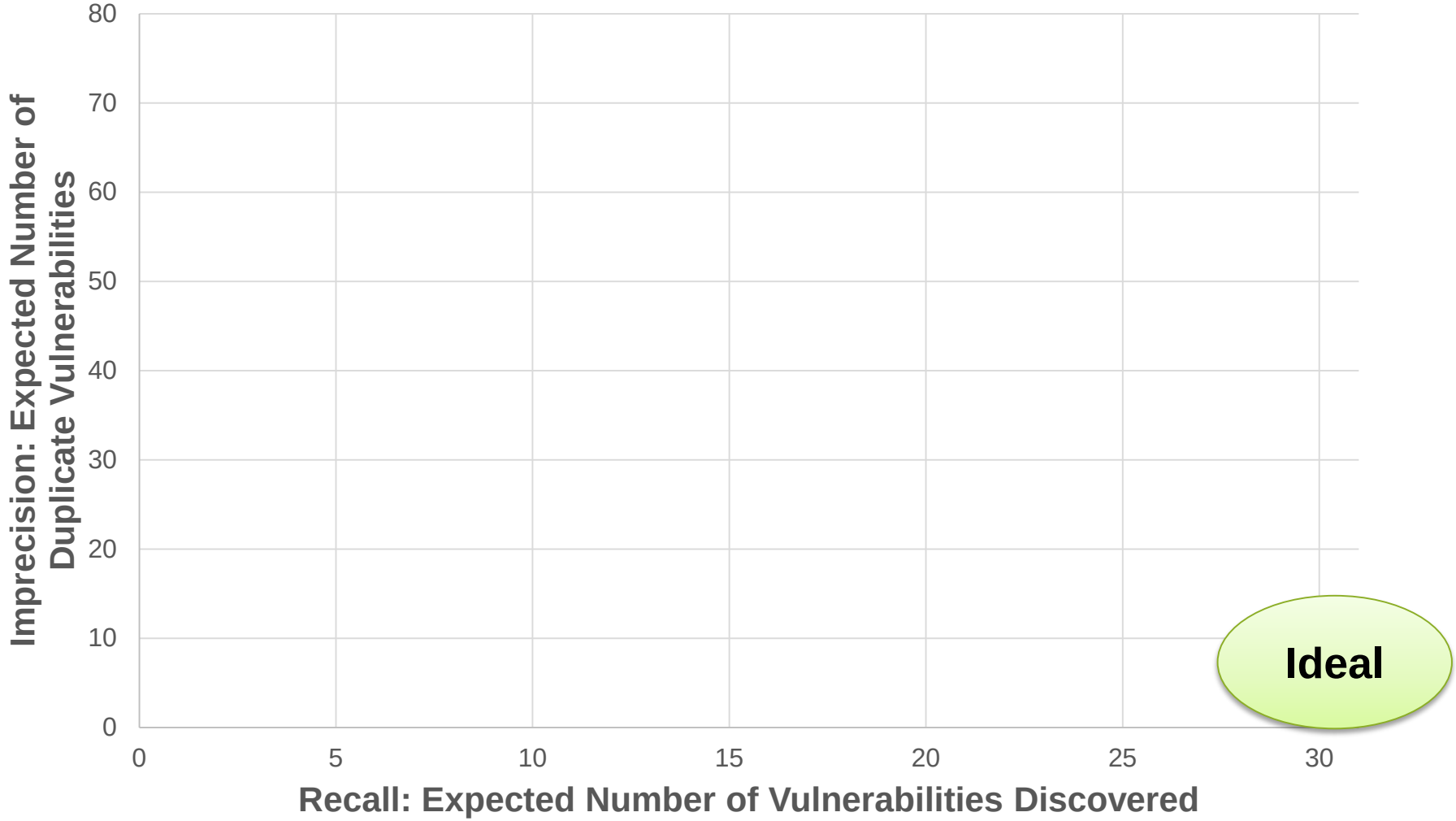
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The State of the Art: Stack Hashing



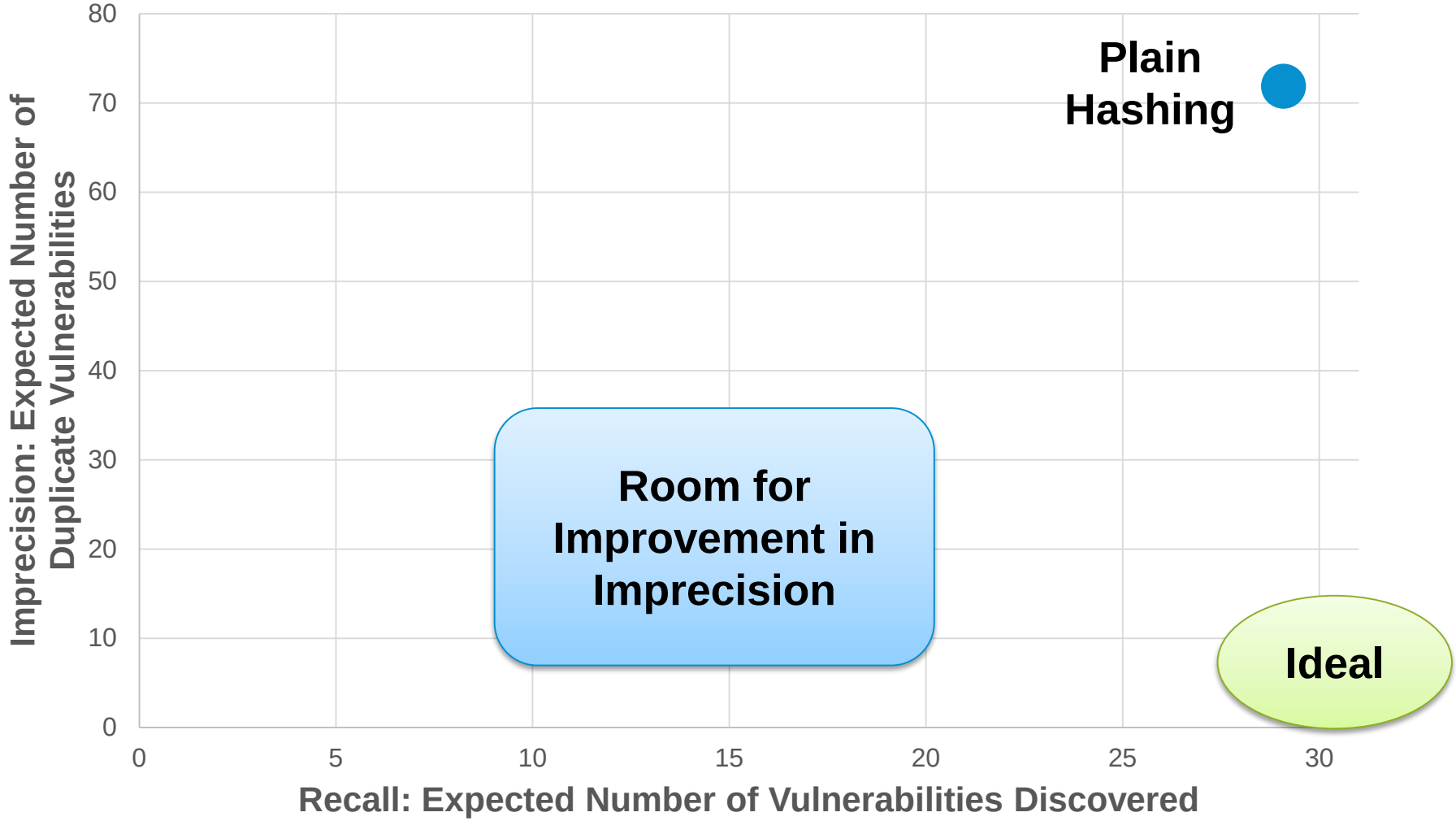
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Does Stack Hashing Work?



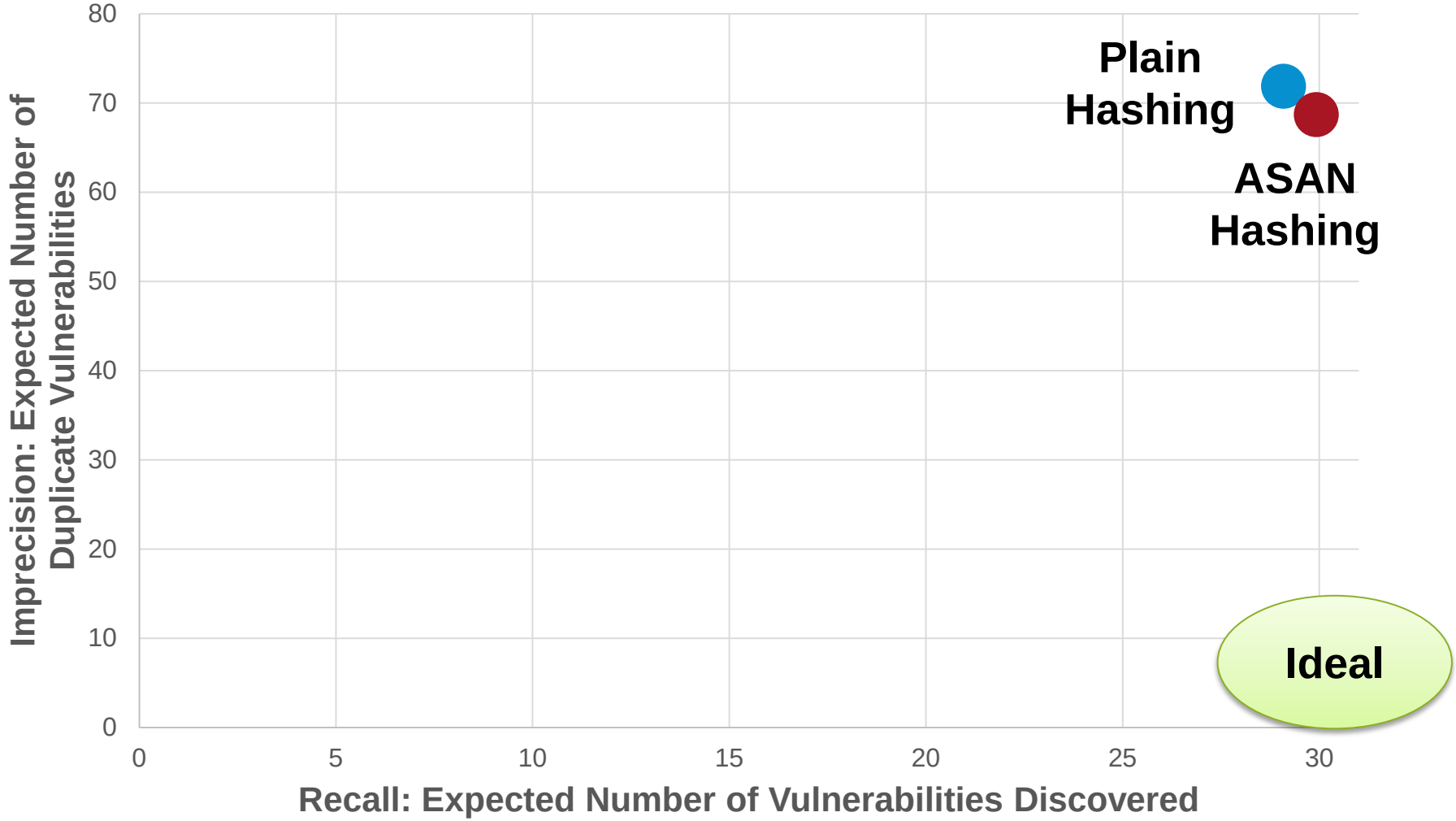
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Does Stack Hashing Work?



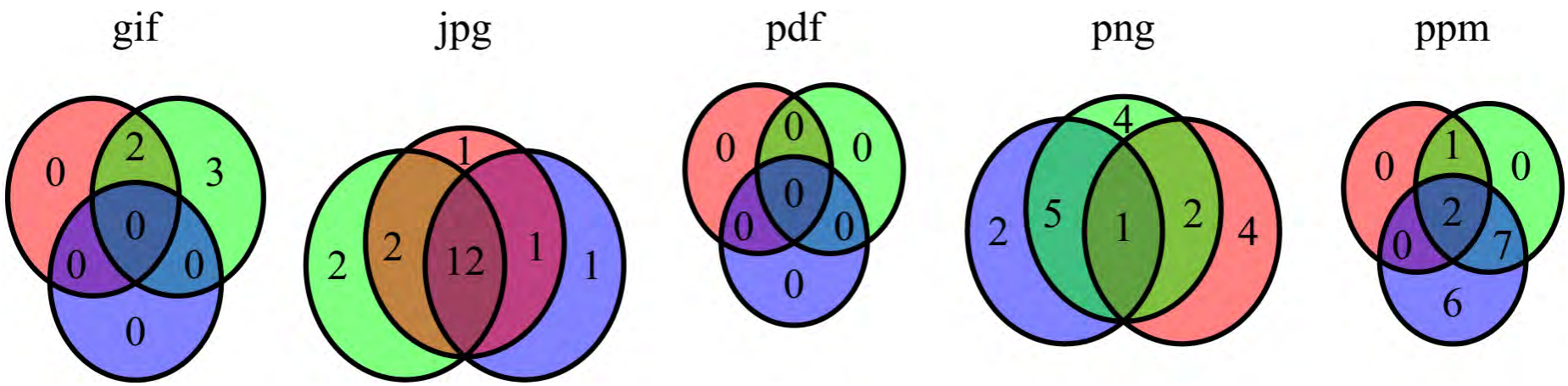
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Does Stack Hashing Work?



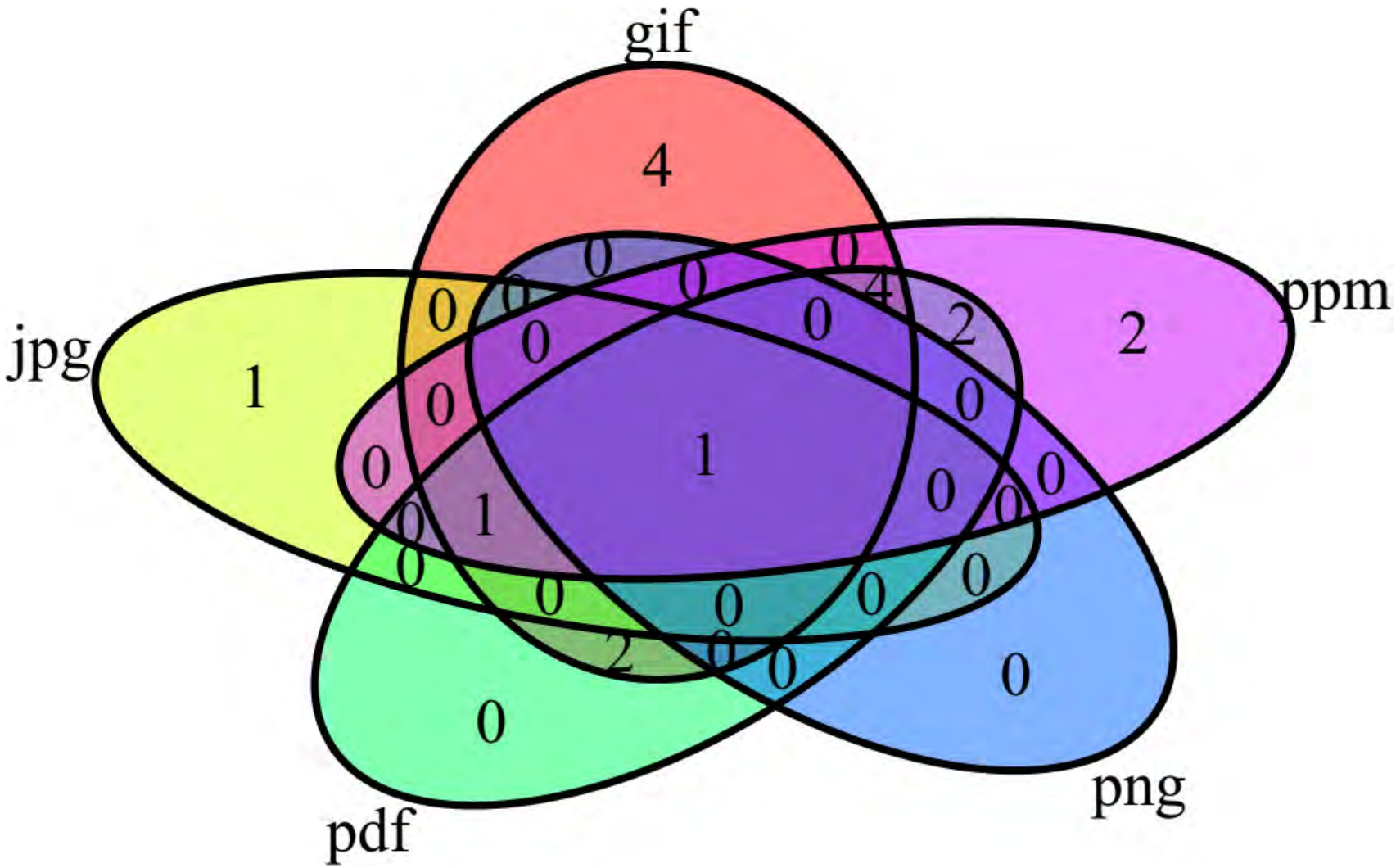
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Importance of Seed Selection



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Importance of Seed Selection



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release;
Distribution is Unlimited

Challenge: Multiple Vulnerabilities

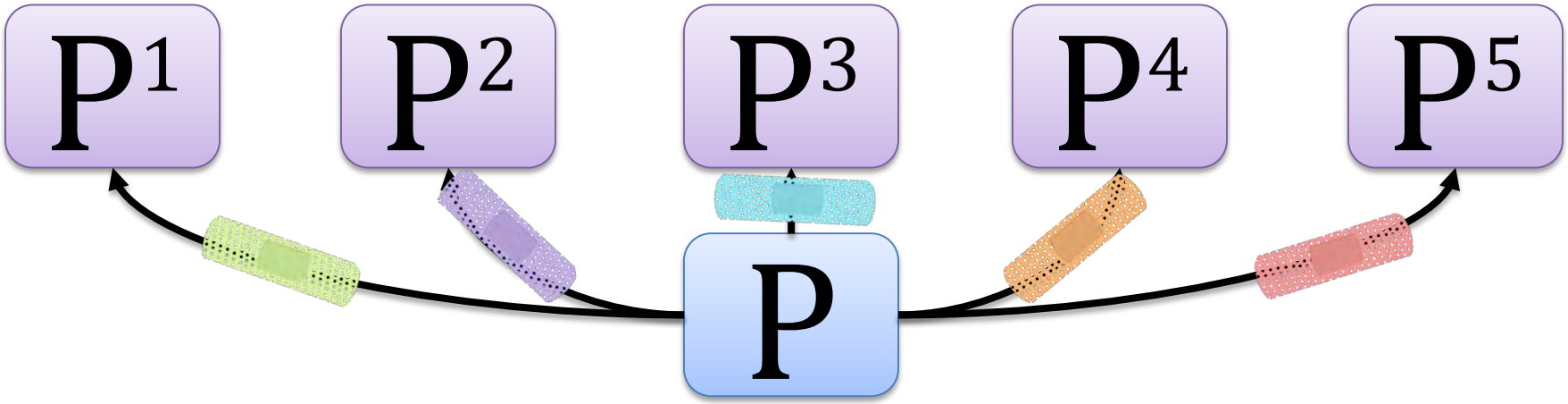
```
int main(int argc, char* argv[]) {
  int x = atoi(argv[1]);
  if (x&1) vulA(1);
  if (x&2) vulB(1);
}
```

Which vulnerability causes `main(3)`?

Vuls	1	2	3	4	5
Crashes	45859	**79626**	6860	21	1

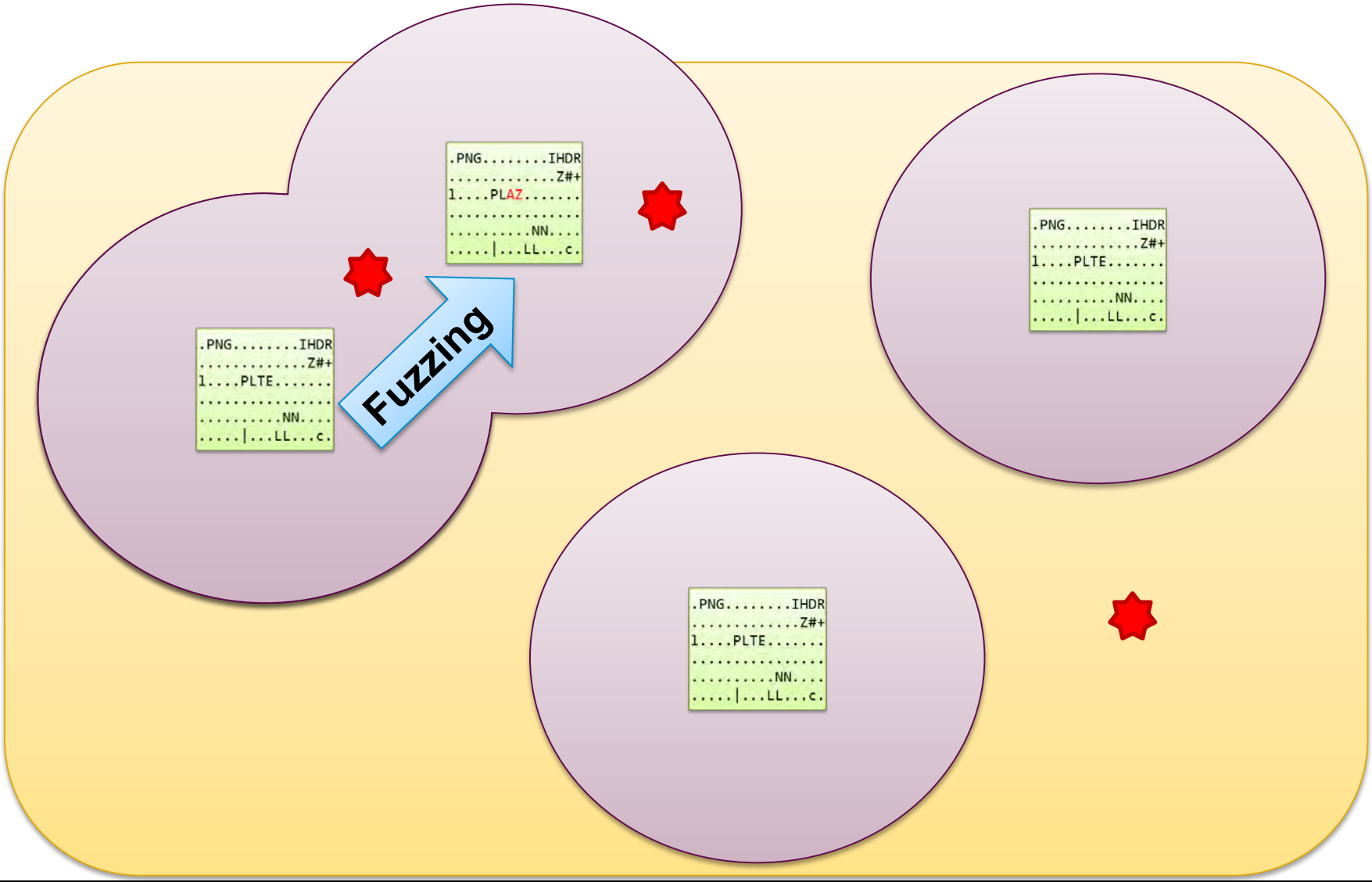
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

The Patch Tree:
Ability to Test Patches Independently



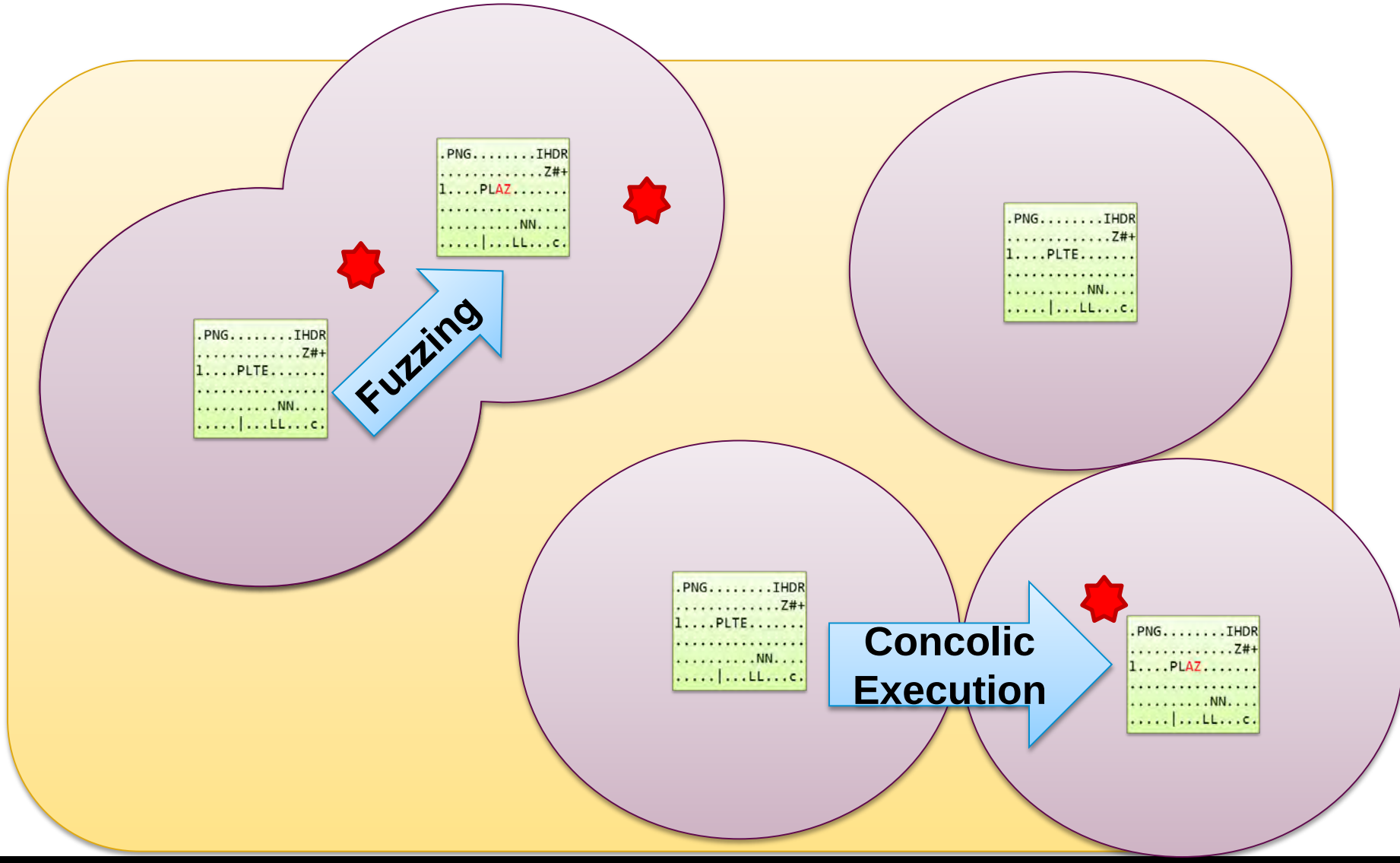
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Guided Fuzzing



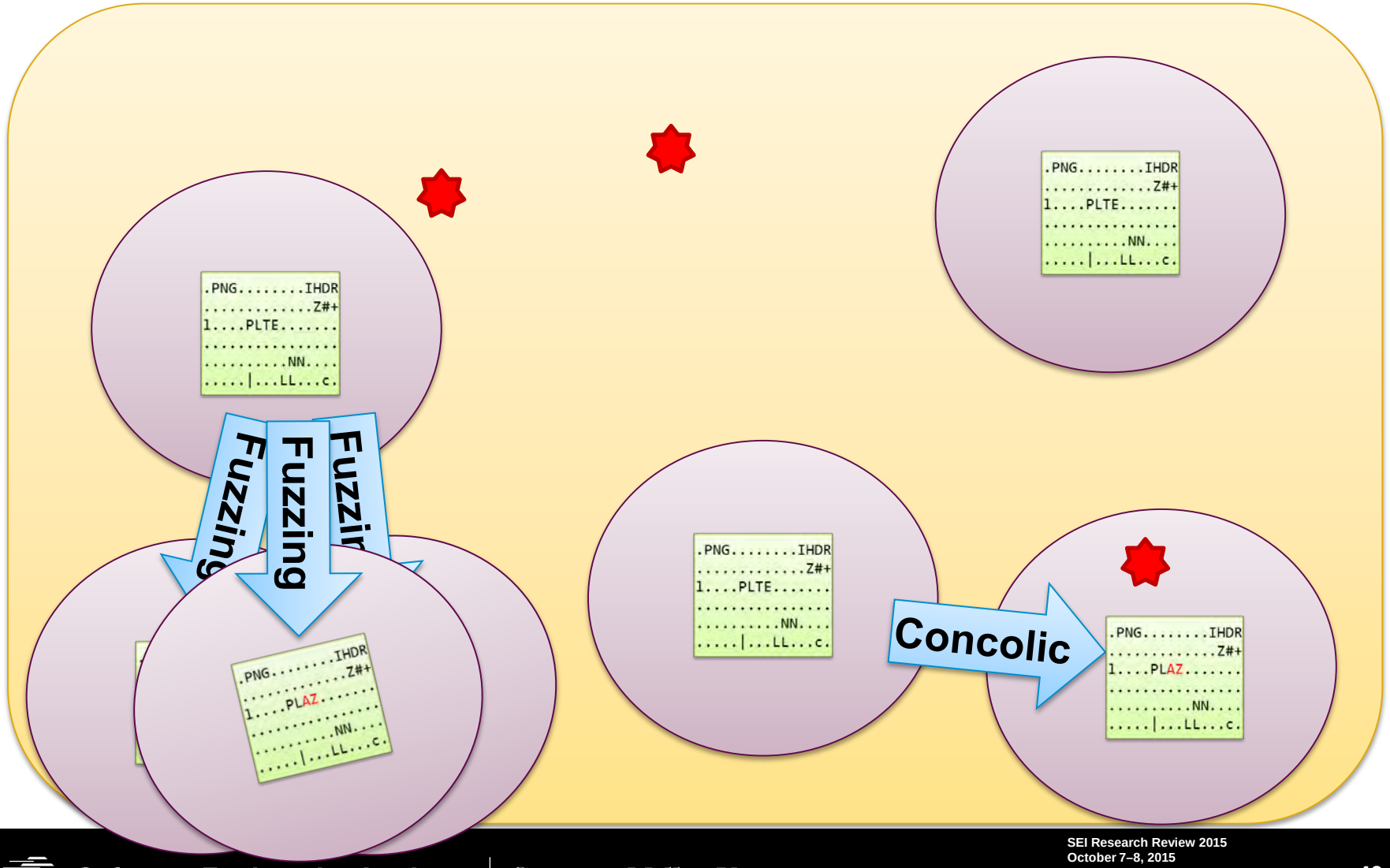
© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Fuzzing vs. Concolic Execution



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Combining Fuzzing and Concolic Execution



© 2015 Carnegie Mellon University
Distribution Statement A: Approved for Public Release; Distribution is Unlimited