

Model Checking and Security

Amin Vahdat

CSE 123b

June 1, 2006

Annoucements

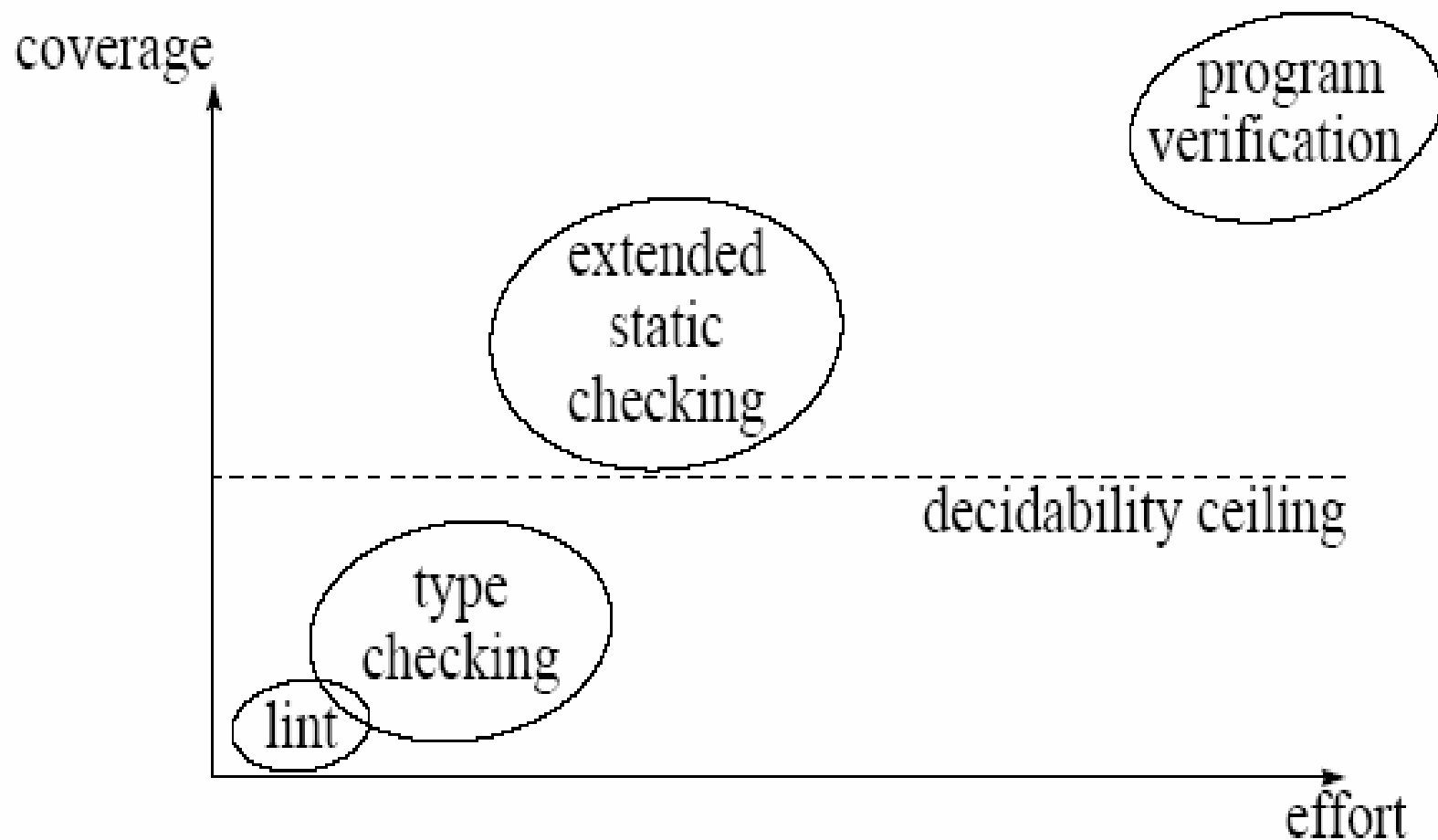
- Third assignment
 - Due date June 9, 5 pm
- Final exam, June 14, 11:30-2:30

A System and Language for Building System Specific, Static Analyses

Background

- What is model checking?
 - Why is it hard?
- What is static analysis?
 - Why is it hard?
- Concerns
 - *Soundness*
 - *Completeness*

Solution Space



MetaCompilation Approach

- Metal language
 - Specifies correctness rules for a given program
 - Goals: easy to use and flexible enough to express a broad range of rules
 - State machine approach for specifying expected behavior
- xgcc
 - Begin with specification of correct behavior, explore code (statically)
 - Exploring different potential values for branch paths

Sample Rule

```
1: state decl any_pointer v;  
2:  
3: start: { kfree(v) } ==> v.freed;  
4:  
5: v.freed: { *v } ==> v.stop,  
6: { err("using %s after free!",  
    mc_identifier(v)); }  
7: | { kfree(v) } ==> v.stop,  
8: { err("double free of %s!",  
    mc_identifier(v)); }  
9: ;
```

Sample Target Code

```
1:int contrived(int *p, int *w, int x) {
2:  int *q;
3:
4:  if(x)
5:  {
6:      kfree(w);
7:      q = p;
8:      p = 0;
9:  }
10: if(!x)
11:     return *w; // safe
12: return *q; // using 'q' after free!
13:}
14:int contrived_caller (int *w, int x, int *p) {
15: kfree (p);
16: contrived (p, w, x);
17: return *w; // using 'w' after free!
18:}
```

Lock Checker

```
state decl { lock_t } l;  
start:  
{trylock(l) != 0} ==> true=l.locked, false=l.stop  
| {trylock(l) == 0} ==> true=l.stop, false=l.locked  
| {lock(l);} ==> l.locked  
| {unlock(l);} ==>  
{ err("%s is not locked", mc_identifier (l)); }  
;  
l.locked:  
{lock(l);} || {trylock(l)} ==>  
{ err("dbl. lock of %s", mc_identifier (l)); }  
| {unlock(l);} ==> l.unlocked  
| $end_of_path$ ==>  
{ err("path ends with lock held"); }  
;
```

Model Checking

- Static checking must consider all possible paths through a piece of code
 - I.e., for all possible inputs and error conditions, will the program behave correctly
 - This can only scale so far
- Model checking runs the code with a particular input
 - User specifies “transition points” where multiple paths should be explored
 - Often, these transition points correspond to inter-leavings of events across multiple programs
 - May correspond to particular error conditions
 - At each transition point, determine whether there have been any *safety* or *liveness* violations

Model Checking TCP

- Fully encapsulate TCP implementation in the linux kernel
- For a particular sequence of message transmissions, consider inter-leaving of a variety of events
 - Messages being dropped
 - Nodes failing
 - Sockets being reset
- Determine whether the state of the system after any high level transition (e.g., a message being received and processed) matches some specification (TCP RFC) of correct behavior

Security Issues

(notes from Tom Leighton keynote address at
USENIX/ACM NSDI, May 2005)

Pharming

- Other ways to hijack traffic and identities
- DNS is not authenticated and so it is possible to alter resolutions through cache poisoning
 - Diverting traffic to an unintended location
- Confidential data can be copied from a compromised server
- Compromised browsers can be made to send traffic to the wrong location

Phishing

- The Problem is Large and Rapidly Growing
 - One percent of US Households experienced identity theft in the first half of 2004, with losses in excess of \$400M
 - There were 13,141 unique phishing e-mail messages reported to the Anti-Phishing Working Group (APWG)

Virus and Worm Proliferation

- Over 10% of PCs were infected by viruses in 2003
- Hostile traffic originated from 40% of networks controlled by Fortune 100 companies
 - Despite the fact that these companies have taken a variety of protective measure
- 92% of organizations experienced "virus disasters" in 2003 (ICSA Labs)
- 83% of financial institutions experienced compromised systems in 2003
 - More than double the rate in 2002 (Deloitte)

DOS and BotNet Attacks

- Denial-of-service attacks have become the second most costly type of computer crime, after computer viruses (2004 CSI and FBI survey)
- 17% of 100 companies surveyed reported being the target of cyberextortion (CMU-Information Week)
- In 2004, there was a 400% increase in the number of botnet attacks against commercial Internet sites (Symantec)
- Reports of attacks at up to 8 Gb/s of sustained attack from botnets
 - Almost no origin infrastructure can handle this