Problems Counting Weaknesses from Static Analysis Tool Exposition (SATE)

Paul E. Black

paul.black@nist.gov

http://samate.nist.gov/

Outline

- Overview of SATE 2008
- The very idea of distinct weaknesses
- Possible useful ideas

SATE 2008 Overview

- **Static Analysis Tool Exposition (SATE) goals:**
  - **Enable empirical research based on large test sets**
  - **Encourage improvement of tools**
  - **Speed adoption of tools by objectively demonstrating their use on real software**

- ***NOT*** **to choose the "best" tool**

- **http://samate.nist.gov/index.php/SATE.html**

SATE 2008 Events

- **Telecons, etc. to come up with procedures, goals**
- **Choose 6 open source C & Java programs with security implications.**
- **Provide them to tool makers (15 Feb)**
- **Tool makers run tools and return reports (29 Feb)**
- **(Try to) find “ground truth” (15 Apr)**
- **Rounds of critique and update with some tool makers (13 May)**
- **Share observations in workshop (12 June)**
- **Final report and all data available Q2 2009**

SATE 2008 Observations

- **Tools reported 13 of SANS Top 25 CWEs (21 if related CWEs count)**
- **Tools reported some 200 different kinds of weaknesses**
  - **Buffer errors still very frequent in C**
  - **Many XSS errors in Java**
- **Coding without security in mind leaves LOTS of weaknesses**

- **In SATE 2009 we will use the latest (“beta”) version for more benefit to developers**

Tools Useful in Quality "Plains"

Tararua mountains and the Horowhenua region, New Zealand
Swazi Apparel Limited   www.swazi.co.nz used with permission

- **Tools are not enough to achieve the highest "peaks" of quality.**
- **In the "plains" of typical quality, tools can help.**
- **If code is adrift in a "sea" of chaos, train developers.**

Outline

- Overview of SATE 2008
- The very idea of distinct weaknesses
- Possible useful ideas

Are There Distinct Weaknesses?

- No; the idea of “one weakness” does not (and cannot) have a well-defined meaning in most cases of production code.
  - Only 1/8 to 1/3 of weaknesses are simple.
- The notion breaks down when
  - weakness classes are related,
  - data or control flows are intermingled, or
  - there are many instances of one syndrome.
- Even “location” is nebulous.

Weakness Classes are Related

- **Hierarchy**
  - **Cross-Site Scripting (CWE-79) is a child (subset) of Improper Input Validation (CWE-20)**
- **Chains**
  - **Validate-Before-Canonicalize (CWE-180) allows Relative Path Traversal (CWE-23)**

```
lang=%2e./%2e./%2e/etc/passwd%00
```

- **Composites**
  - **Symlink Following (CWE-61) occurs because of several weaknesses, including Race Conditions (CWE-362), Predictability (CWE-340), and Permissions (CWE-275)**

from “Chains and Composites”,Steve Christey, MITRE

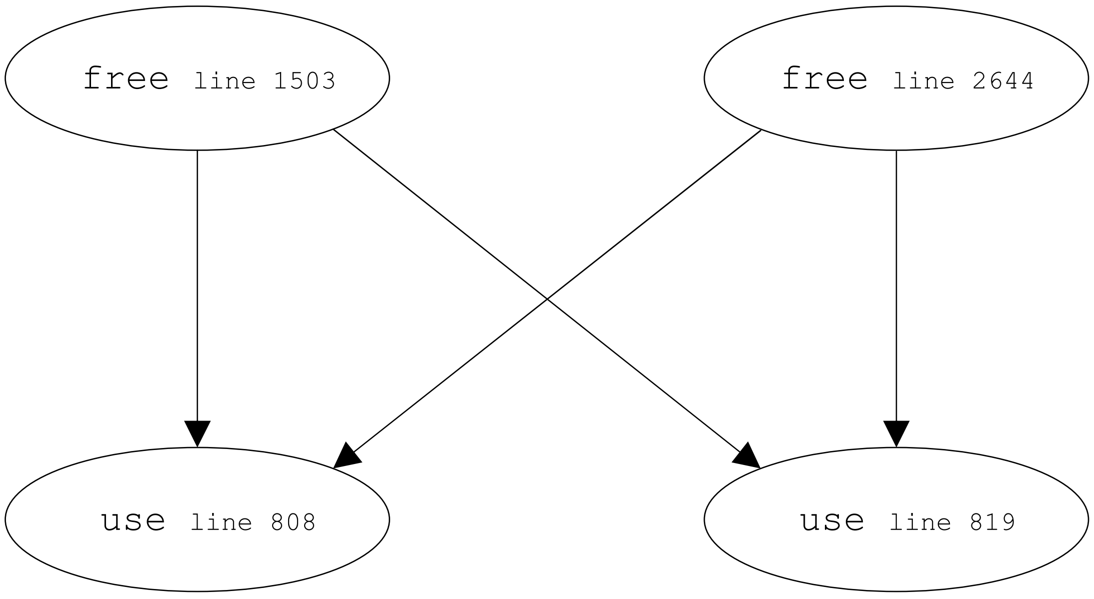
http://cwe.mitre.org/data/reports/chains_and_composites.html

Many Instances of a Syndrome

- **Because of coding habits, the same construct may occur many times.**
  - **Double Unlock vulnerability**

```
while(1){
        pthread_mutex_lock(&buffer_lock);
                … other stuff…
        pthread_mutex_unlock(&buffer_lock);
                … a lot of other stuff…
}
```

Intermingled Flow:
2 sources, 2 sinks, 4 paths
How many weaknesses?



Intermingled Flows

```
1462   for(temp_event=event_list_low; temp_event; temp_event=temp_event->next){
       …
       }
       …
    remove_event(temp_event,&event_list_low);
    free(temp_event);
    …
    reschedule_event(new_event,&event_list_low);
```

```
2603   for(temp_event=event_list_low; temp_event; temp_event=temp_event->next){
       …
       }
    …
    remove_event(temp_event,&event_list_low);
    free(temp_event);
    …
    reschedule_event(new_event,&event_list_low);
```

2 sources, 2 sinks, 4 paths How many weaknesses?

```
2603   for(temp_event=event_list_low; temp_event; temp_event=temp_event->next){
       …
       }
     …
     remove_event(temp_event,&event_list_low);
     free(temp_event);
     …
     reschedule_event(new_event,&event_list_low);

  reschedule_event(…,timed_event **event_list){
     …
     add_event(event,event_list);

  add_event(…,timed_event **event_list){
     first_event=*event_list;
     …
 808   else if(event->run_time < first_event->run_time){// 43523 43525
       …
     else{
       temp_event=*event_list;
       while (temp_event) {
 819        if (temp_event->next==NULL) {// 43522 43524
```

Even Locations are not Definite

- **Source or sink?**
- **Caller or callee?**
- **Data path? Enclosing function?**
- **Regions - Dead Code (CWE-561) or Leftover Debug Code (CWE-489)**
- **Missing function/property - Session Doesn’t Expiration (CWE-613)**

Outline

- Overview of SATE 2008
- The very idea of distinct weaknesses
- **Possible useful ideas**

What Concepts *Are* Useful?

- **Weakness class, e.g., CWE**
- **Vulnerability**
- **Attacks or exploits**

Additional Useful Concepts

- **Source, sink**
- **Fault - when program state first goes bad**
- **Path, data or control**
  - **the set of all paths such that …**
- **Region - lines with "bad" state**
  - **session not closed, resource not freed, etc.**

Even More Concepts

- **Error (i.e., human mistake)**
- **Code fixes needed (minimum? best?)**
- **# weaknesses = min(sources, sinks)**
  - – **Why? #code fixes ≤ #weaknesses (usually …)**