Observations on Static Analysis to Detect Security Weaknesses

Paul E. Black

National Institute of Standards and Technology

http://www.nist.gov/

paul.black@nist.gov

- **Although we believe the material in this presentation to be accurate, everything should be treated as preliminary, not final.**

Outline

- General Impressions
- Repeated patterns
- Definition of false alarm
- Summary of warnings
- Definition of “one weakness”

Our Impressions of Tools

- **Tools mostly handled the code well – not trivial for the size of test cases**
- **Very impressed by the sophisticated analysis and reporting capabilities of tools**
- **Tools are useful in the quality "plains"**
  - **If code is terrible, don't bother with tools - educate your development and rewrite**
  - **For typical quality, tools can help**
  - **Tools only help in achieving the highest levels**
- **Not much overlap between tool reports**

Analysis phase “misused” tools

- **Tools help users answer the question**
  - **What code should be changed?**
- **We pose the harder question**
  - **Where are bugs, and what are they?**

- **SATE format limited information**
- **Many tools targeted toward general quality - not the same as security.**

Suggestions

- **Give more internal information, e.g., variable never < 0**
- **Use clearer weakness names**
- **Consistent severity (and other) ratings**
- **Postprocess reports to group them by location or type**

Suggestions for CWE

- **Clean up the most important weaknesses rather than working on them all**
  - **Consistent, meaningful names**
  - **Precise, even formal, definitions**
- **Add severity rating**

Outline

- General Impressions
- **Repeated patterns**
- **Definition of false alarm**
- Summary of warnings
- Definition of “one weakness”

Repeated Patterns

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

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

Repeated Patterns

- **strncpy Does Not Null-Terminate**

```
strncpy(variable,temp,sizeof(variable));
variable[sizeof(variable)-1]='\x0';
```

- **Our definition of false alarm was vague**
  - **This bit of code is fine.**
  - **The warning is correct as phrased.**

- **By the way, manual review found an error**

```
strncpy(input_buffer,url,sizeof(input_buffer)-1);
output_buffer[sizeof(input_buffer)-1]='\x0';
```

Outline

- General Impressions
- Repeated patterns
- Definition of false alarm
- **Summary of warnings**
- Definition of “one weakness”

Warnings by Severity

	Severity					
	1	2	3	4	5	All
Naim	299	155	227	660	407	1748
Nagios	652	538	1958	1783	1529	6460
Lighttpd	588	710	2032	403	153	3886
OpenNMS	1140	1266	4342	5564	9248	21560
MvnForum	113	1129	1406	2244	3174	8066
DSpace	381	384	2379	1386	1675	6205
All	3173	4182	12344	12040	16186	47925

Warnings by weakness type

	XSS	SQL inj.	Buffer	Info leak	Null ptr deref	Numeric error
Naim	0	0	624	4	128	155
Nagios	0	0	1684	766	94	559
Lighttpd	0	0	966	1092	293	243
OpenNMS	1748	179	0	180	1514	9
MvnForum	471	483	0	174	108	4
DSpace	417	53	0	273	308	0
All	2636	715	3274	2489	2445	970

Severity 1,2 Warnings Evaluated

	Warnings	Reduced to	TP	FP
Naim	454	180	52	90
Nagios	1190	291	37	205
Lighttpd	1298	383	13	325
OpenNMS	2406	2180	160	295
MvnForum	1242	329	264	49
DSpace	765	366	115	209
All	7355	3729	688	1198

Outline

- General Impressions
- Repeated patterns
- Definition of false alarm
- Summary of warnings
- **Definition of “one weakness”**

Need Better Def'n of Duplicate

- **Goal: group warnings that refer to the same weakness**
  - **To unify warnings from different tools**
  - **… or warnings from the same tool**
- **Same sink?**
- **Same source?**
- **Same path?**
- **Mitigation location(s)?**

So what is "the same weakness" ??

What is “the same weakness”?

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

    schedule_host_check(...){
        ...
2603    for(temp_event=event_list_low;temp_event!=NULL;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!=NULL;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!=NULL){
 819            if(temp_event->next==NULL){// 43522 43524
```