

# Introduction to Hacking Competitions CTF & CGC

Kir Chou @ Meetup Coffee with Science  
2017 Nov

# About me

Kir Chou

- Taiwanese
- SDE (Pythonista) @ Tokyo



[kirchou](#)



[note35](#)



[kir.chou](#)

# Outline

- i. What is CTF
- ii. Problem categories
- iii. Benefit from CTF
- iv. Culture of CTF
- v. What is CGC
- vi. CRS example

# What is CTF

## Capture the flag



# What is CTF

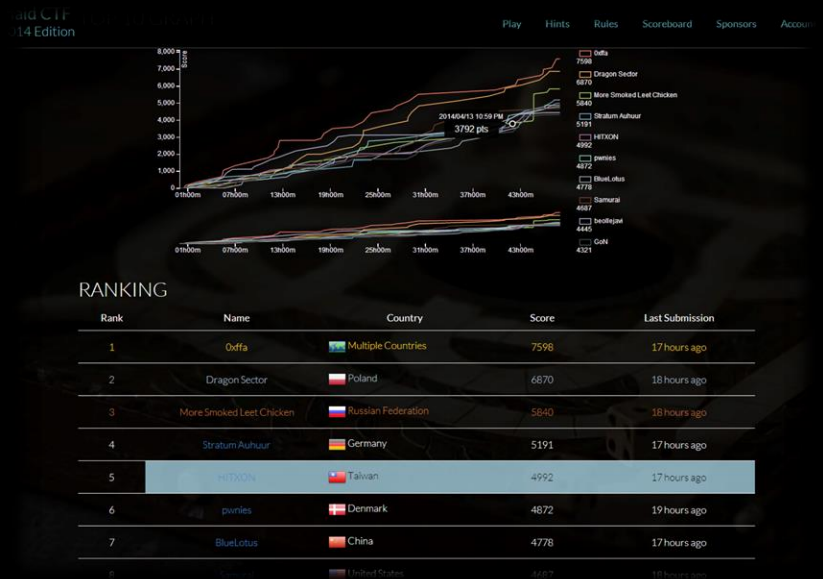
- CTF a.k.a Capture the Flag
  - A Computer security competition – CTF Time
  - For educational exercise and reward
  - Require several skills



# Styles of CTF

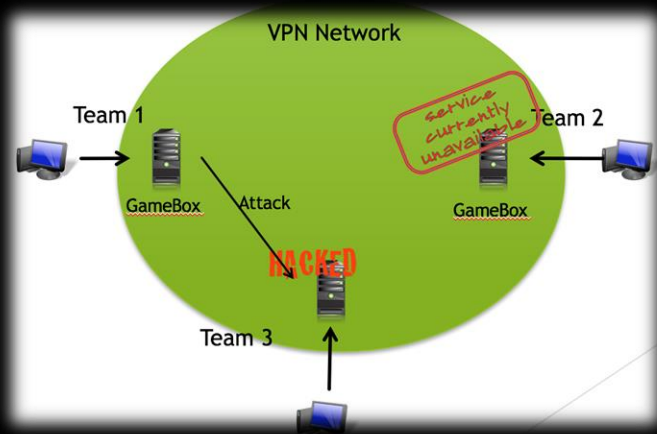
- Jeopardy (Common) - ジアパディー
  - Multiple categories of problems
  - Earn the most points in the time frame

grab bag	/urandom	binary l33tness	pwnables	forensics
100	100	100	100	100
200	200	200	200	200
300	300	300	300	300
400	400	400	400	400
500	500	500	500	500



# Styles of CTF

- Attack-Defense (Advance)
  - Given a machine (or a small network) to defend on an isolated network
  - Famous Competition: DEFCON | CSAW
  - [Game Record in DEFCON 2014 \[Src\]](#)




# DEFCON – Hacker World Cup

- [History](#)
  - Found in 1992 / CTF started from 1996
  - @Las Vegas in August
- How to enter?
  - Champion in seed CTF (Hitcon, Seccon...etc)
  - Top10 @ DEFCON Quals in May













# HITCON

- Found in 2005
- 2017 DEFCON 2<sup>nd</sup>
- 2016 DEFCON 4<sup>th</sup>
- 2014 DEFCON 2<sup>nd</sup>
- Top 1 @ CTFTIME Oct. 2017
- Why the name is 217?

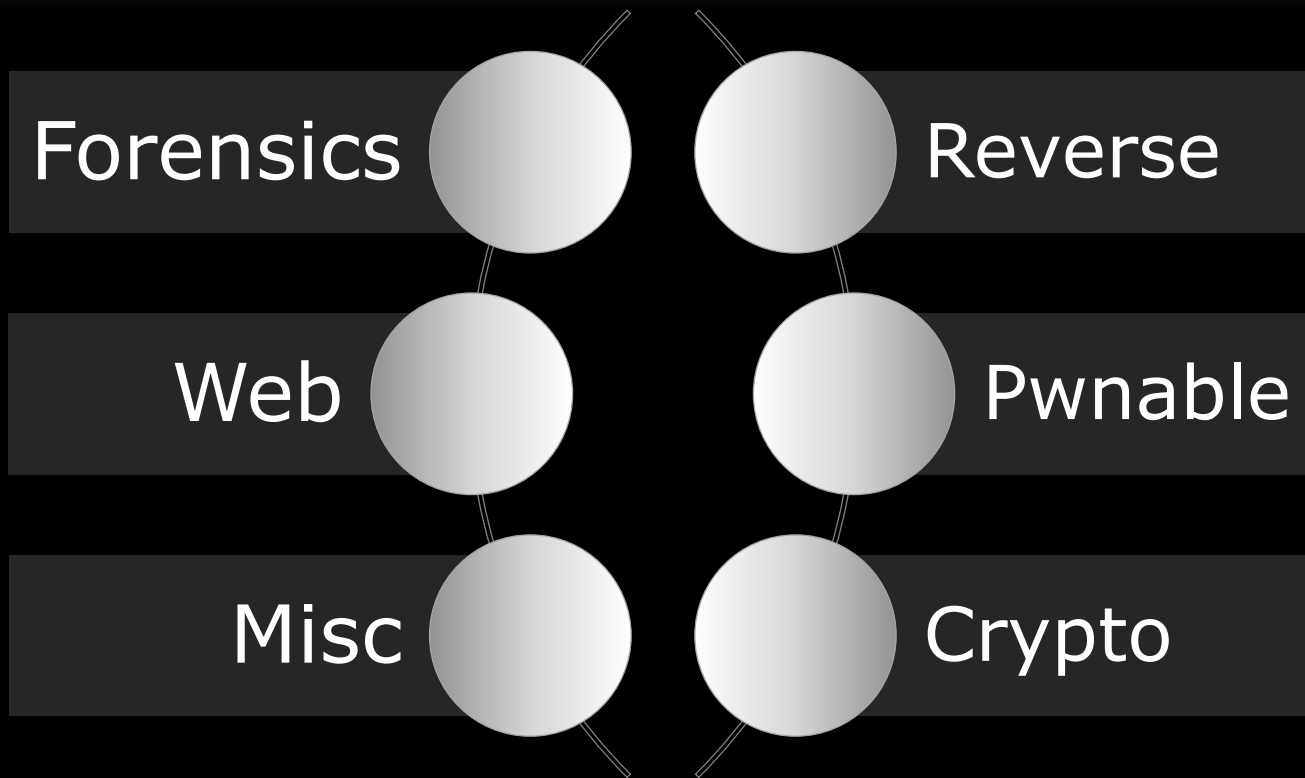


CTFs Upcoming Archive Calendar Teams FAQ

### Team rating

2017	2016	2015	2014	2013	2012	2011
Place	Team	Country	Rating			
1	217		710.143			
2	Plaid Parliament of Pwning		701.259			
3	LC&BC		498.595			
4	TokyoWesterns		484.055			
5	CodiSec		456.643			
6	dcua		455.243			
7	Bushwhackers		450.071			
8	Dragon Sector		445.447			
9	Shellphish		437.935			
10	Eat, Sleep, Pwn, Repeat		401.163			

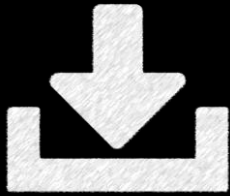
# Problem categories



# Reverse

Stereotype of typical hacker  
Some problems are relied on experience  
Some problems are like pwnable problem

Recommend any background  
0~ year



Download



Find Key

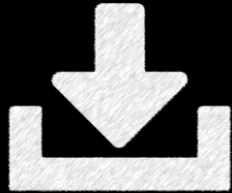


Earn Points

# Pwnable

once you learned, it's fun but need talent

Recommend CS background  
~1 year



Download

Some problems don't  
give you any file



Find exploitable  
vulnerability



Connect to server



Use exploitable  
vulnerability to get shell

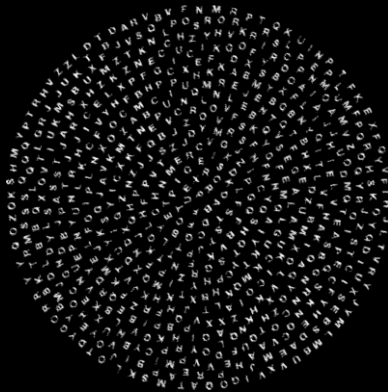


Earn Points

# Crypto

Very hard to learn  
Crypto are usually hard without background

Recommend Math/CS background  
4~ years



Various Source  
Web, File, String.  
Hardware...



Apply Math  
(Modern Cryptography)

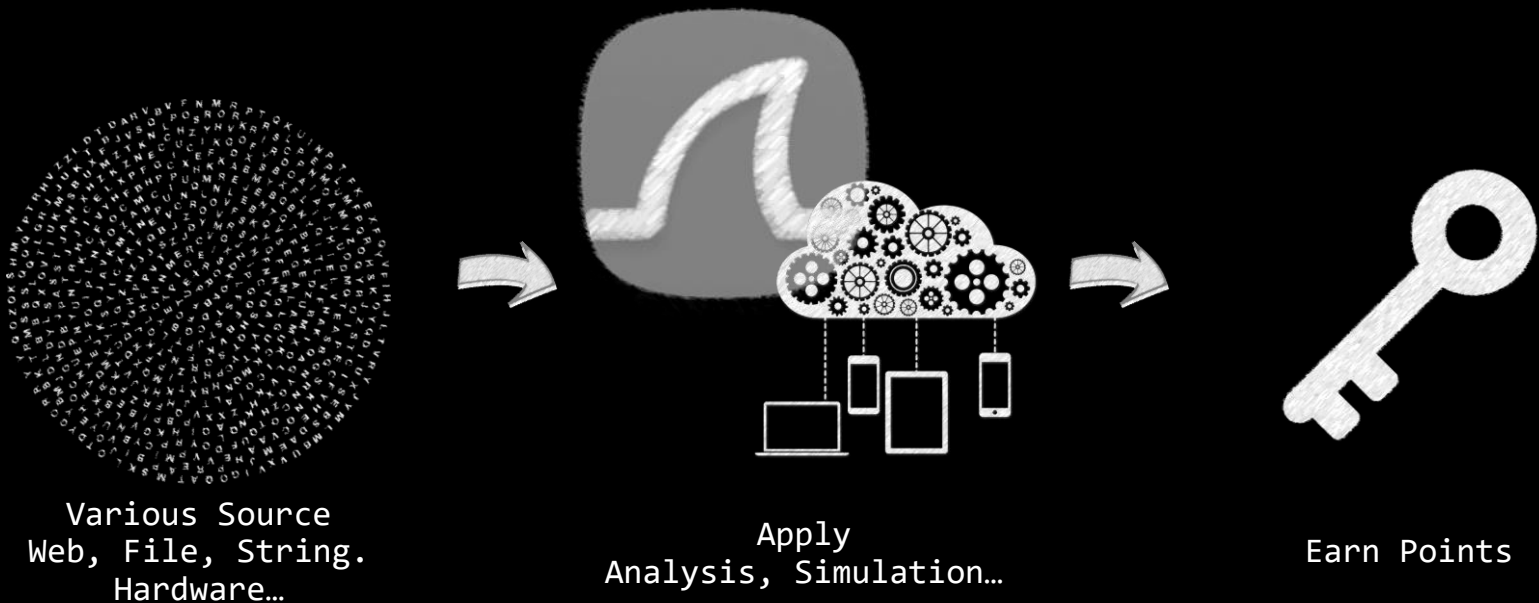


Earn Points

# Forensics

Some problems are rely on experience  
Most of problem need to learn tools

Recommend Any background  
0~ year



# Web

I have no idea how to explain this

Recommend for web geek  
 $\infty$  year

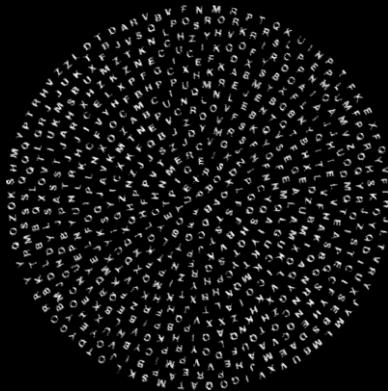




# Misc

No one need to learn how to play  
puzzle...right?

Recommend any background  
0 year



Various Source  
Web, File, String.  
Hardware...



Play with puzzle



Earn Points

Don't be addicted to this  
this won't help you become strong



# Benefit from CTF

- Digging knowledges
- Be bullied & Bullying
- Earn money

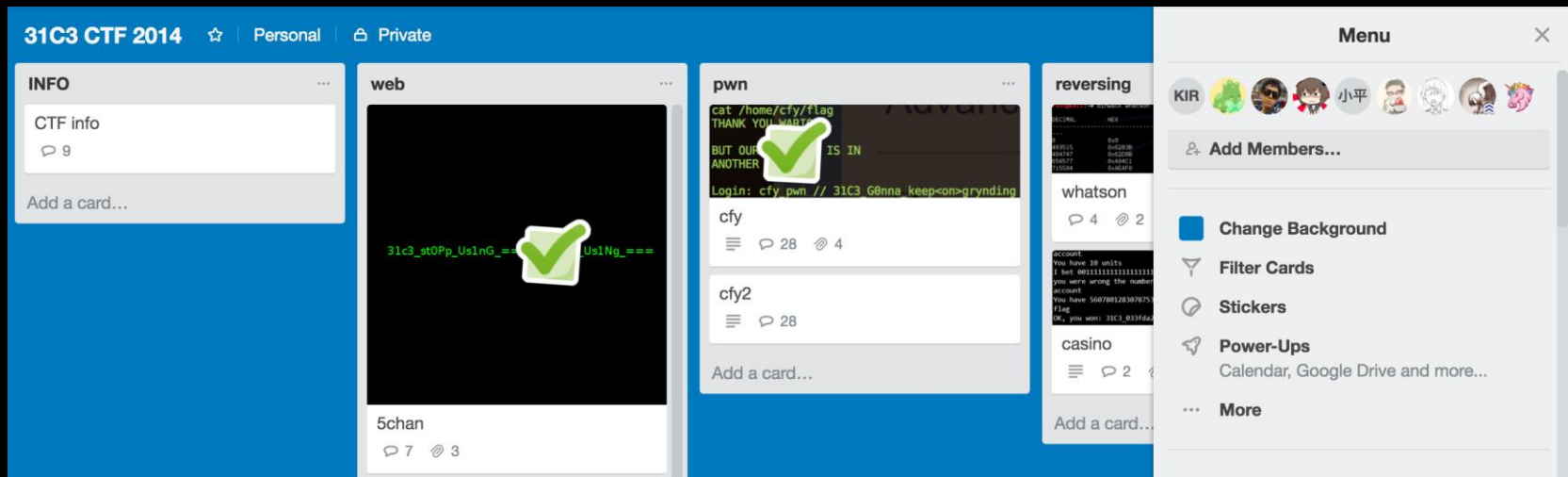


# Culture of CTF

- Strong is everything 強者至上主義
- Strong teams host famous CONF
- Strong teams host famous CTF
- Co-work workspace (eg. Trello, Slack)
- Write-up after ctf (Blog, SNS)
  - writing blog about how you solve problem



# Trello



<https://trello.com/>

# What is CGC

- CGC a.k.a. [Cyber Grand Challenge](#)
- Found by DARPA since 2014 (every 2 years)
- Make a CRS(Cyber Reasoning System) to attack and defense by system itself
  - Challenge Qualification Event (Standalone)
  - Challenge Final Event (Attack-defense)

## Techniques

**Pwnable + Reverse !**

Static  
Analysis

Dynamic  
Analysis

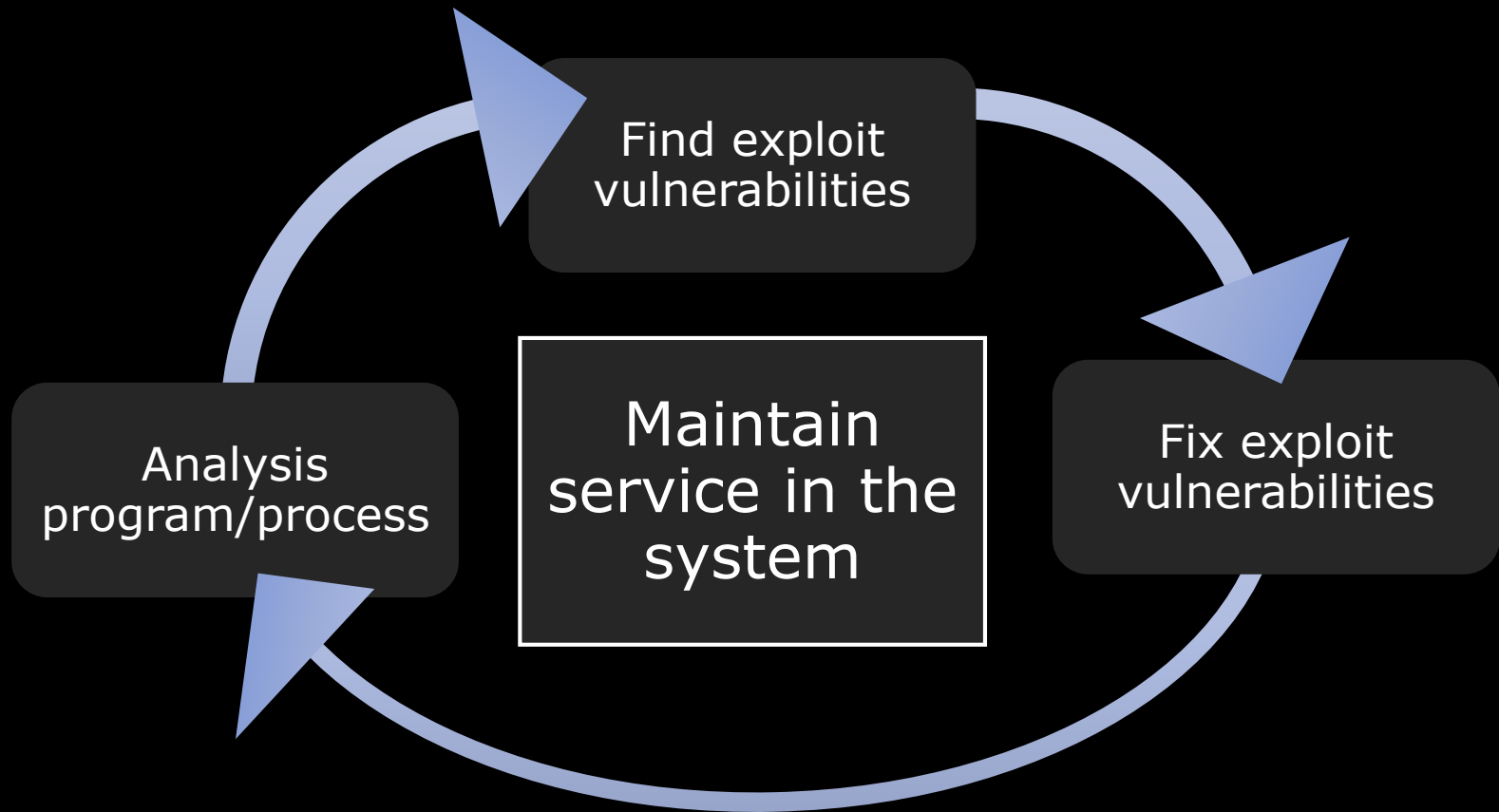
Symbolic  
Execution

Constraint  
Solving

Data Flow  
Tracking

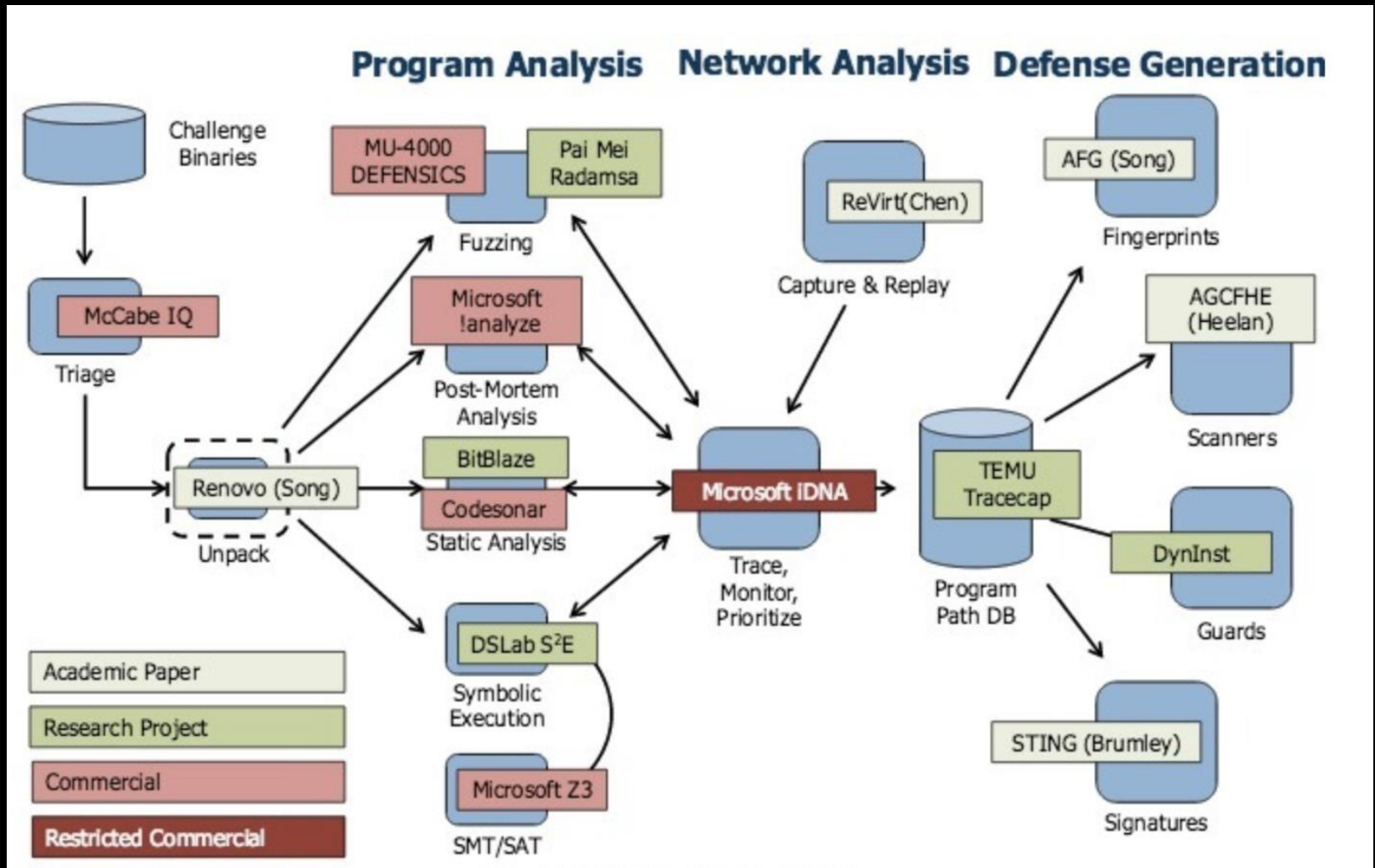
Fuzz  
Testing

# How does CRS work?



Finishing all of them automatically

# CRS Architecture



Thanks for listening



# Appendix

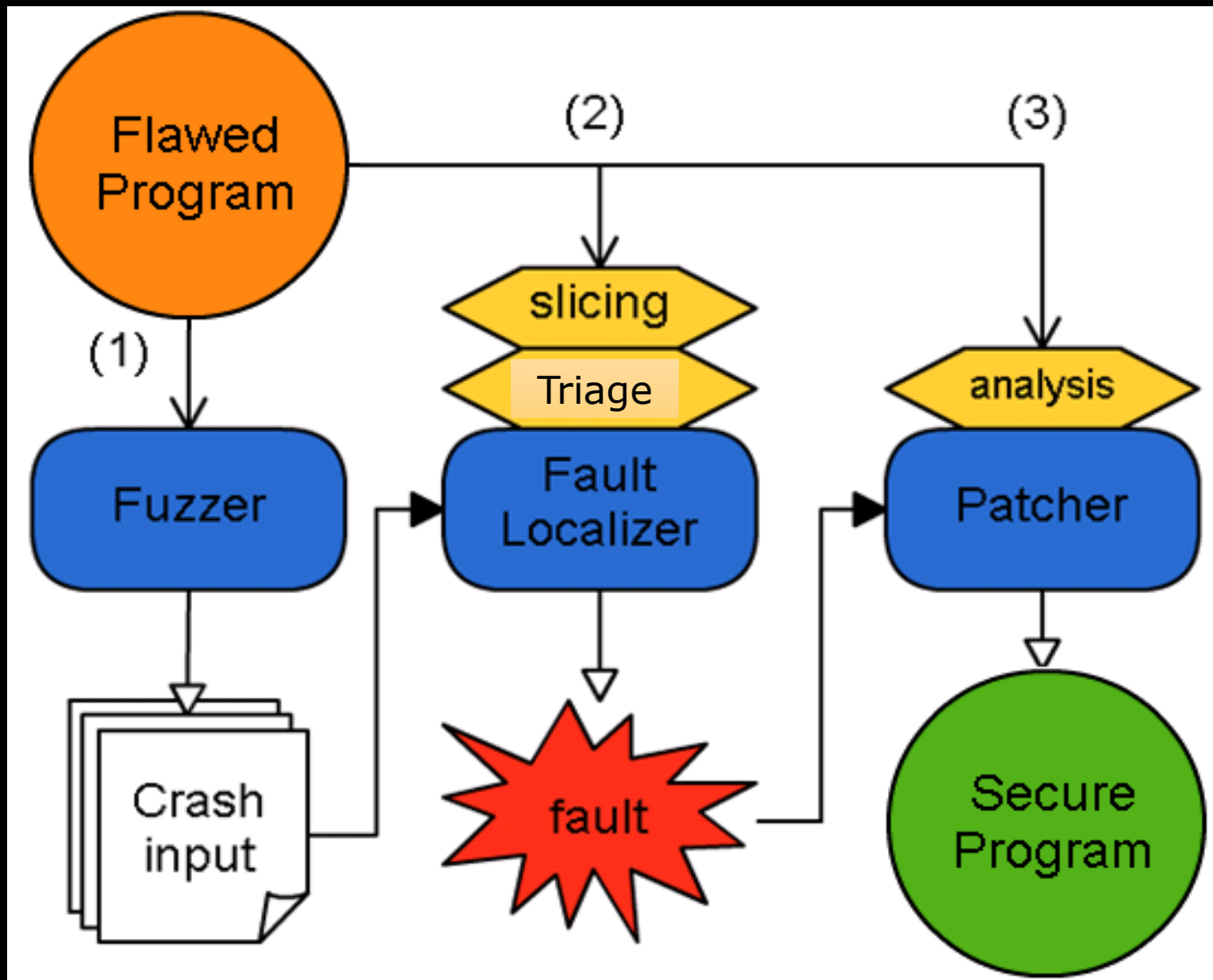
- An auto patching example of CRS



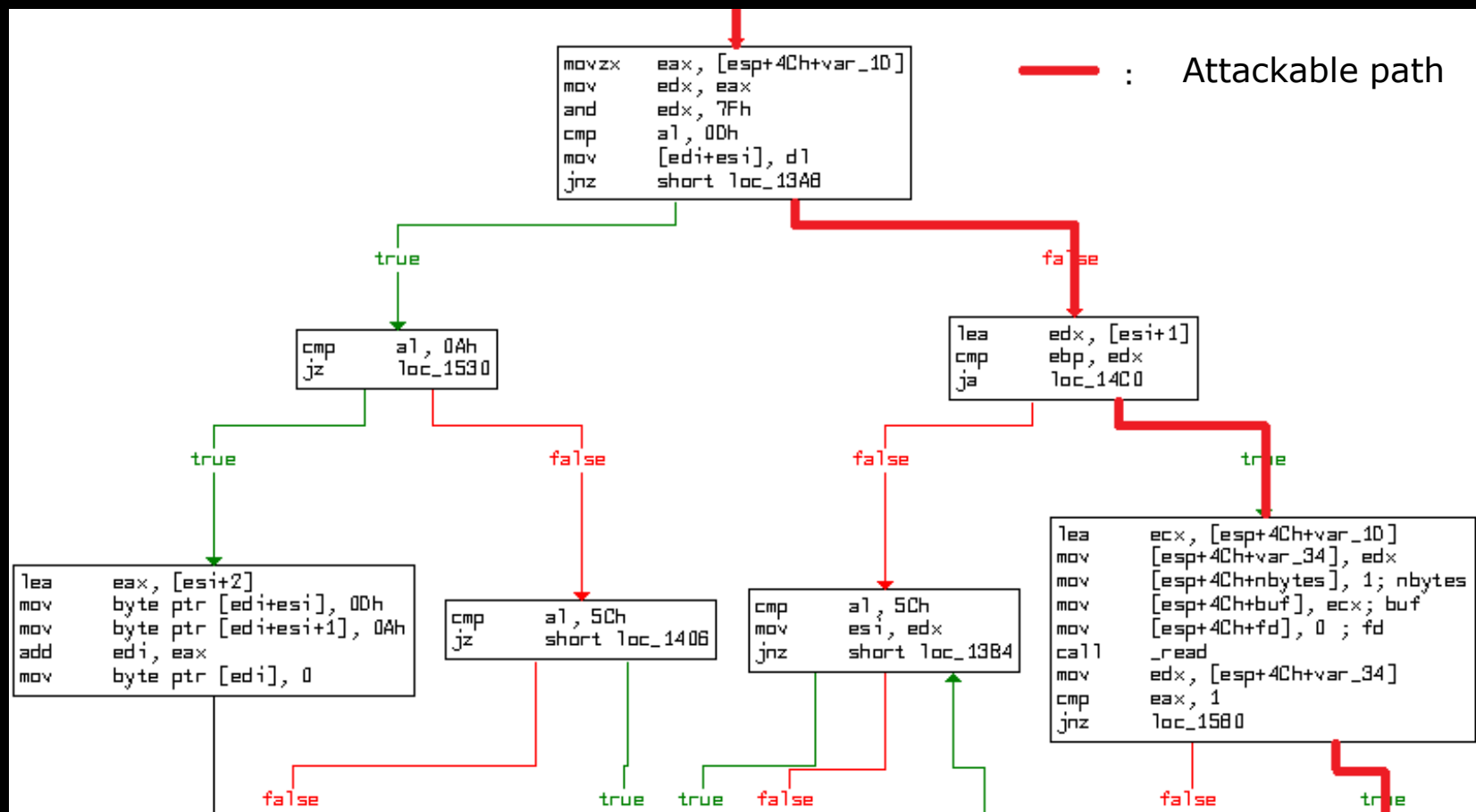
# Example Flawed Program

```
void foo(char* str) {  
    strcpy(str, "1234567890");  
}  
int main(void) {  
    char buf[5];  
    foo(buf);  
    return 0;  
}
```

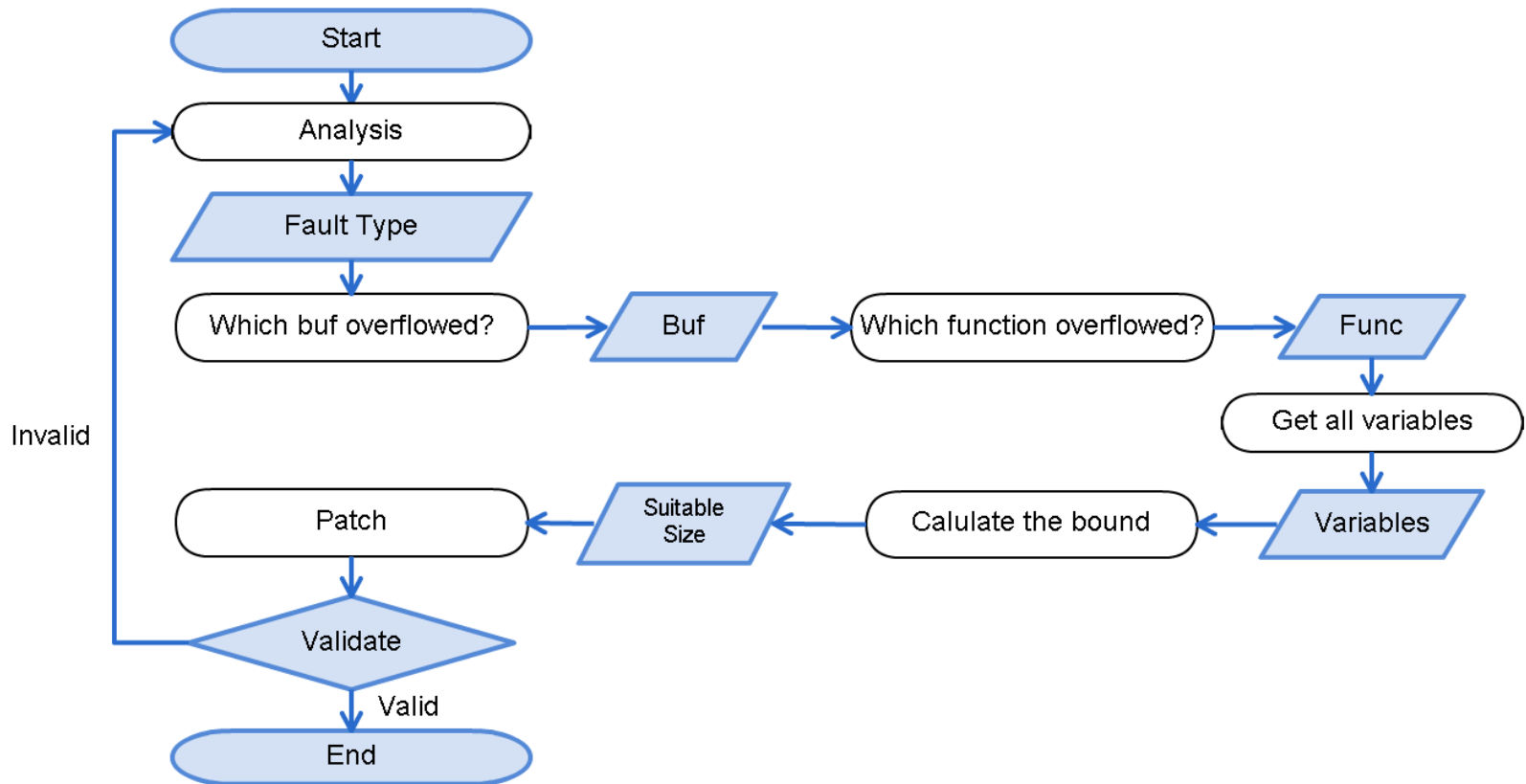
# Example CRS



# Fault Localization



# Patch Flow



# Patch Buffer Overflow

1. Decrease the bound to a suitable value

`strncpy(dst, src, 100) → strncpy(dst, src, 40)`

2. Increase the buffer size

`char buf[40] → char buf[100]`