Git-based CTF: A Simple and Effective Approach to Organizing In-Course Attack-and-Defense Security Competition

SeongIl Wi, Jaeseung Choi, Sang Kil Cha
KAIST
Capture The Flag (CTF)

CTF: Cybersecurity competition that involves capturing a flag
Types of CTF

Attack-and-Defense Style
- Real time, Realistic

Jeopardy Style
- One-way, Problem solving
Number of CTF Events in 2018

- Attack-and-Defense Style: 8 times
- Jeopardy Style: 73 times
In-Course Attack and Defense CTF

• Class Capture-the-Flag Exercises, *USENIX 3GSE ’14*

• Build It, Break It, Fix It: Contesting Secure Development, *ACM CCS ’16*

Difficult to organize!
Attack & Defense CTF Infrastructures

Game Server

Team1 VM

Team5 VM

Teaching Assistant

Flag
Challenges

C1: Interactivity Challenge

Team1 VM

Team5 VM

Game Server

Teaching Assistant
Challenges

C1: Interactivity Challenge

C2: Configuration Challenge
Challenges

C1: Interactivity Challenge

C2: Configuration Challenge

C3: Monitoring Challenge

Teaching Assistant

Game Server

Team1 VM

Team5 VM
Challenges

C1: Interactivity Challenge

C2: Configuration Challenge

C3: Monitoring Challenge

C4: Contents Creation Challenge
Challenges

C1: Interactivity Challenge

C2: Configuration Challenge

C3: Monitoring Challenge

C4: Contents Creation Challenge
Recent Researches:

- **C1: Interactivity Challenge**
  - SWPAC, USENIX ASE '17
- **C2: Configuration Challenge**
  - InCTF, USENIX ASE '16
- **C3: Monitoring Challenge**
  - VM-based Framework, USENIX 3GSE '15
- **C4: Contents Creation Challenge**
  - BIBIFI, ACM CCS '16
  - SecGen, USENIX ASE '17

Handle only a subset of the challenges
Previous Work: BIBIFI, ACM CCS ’16

Build-It → Break-It → Fix-It

Attack

C1: Interactivity Challenge

Defense

C3: Monitoring Challenge
Challenges in BIBIFI

C1: Interactivity Challenge

C2: Configuration Challenge

C3: Monitoring Challenge

C4: Contents Creation Challenge
Can we handle all the challenges?
We propose Git-based CTF
GitHub as a CTF Framework
GitHub = DB

Service

Local Repository

Remote Repository

Push

Pull

Push

Pull

Push

Pull

Push

Pull

Defense

Local Repository

Team 2

Organize

Local Repository

Organizer

Team 1

Local Repository

Attack
Our Goal: Handle All the Challenges

C1: Interactivity Challenge
C2: Configuration Challenge
C3: Monitoring Challenge
C4: Contents Creation Challenge

Team1
Team5
Team1 VM
Team5 VM
Handle Interactivity Challenge

- Real time attack and defense

C2: Configuration Challenge

Team1 VM

Team5 VM

C3: Monitoring Challenge

Teaching Assistant

C4: Contents Creation Challenge
Real Time Attack and Defense

Periodically award points until it is fixed by the defending team
Repository as Scoreboard

Scoreboard

Remote Repository

Push

Pull

Check score

Team 1

Local Repository

Evaluate

Local Repository

Organizer

Check score

Team 2

Local Repository

Pull

Pull
Handle Configuration Challenge

C1: Interactivity Challenge
C2: Configuration Challenge
C3: Monitoring Challenge
C4: Contents Creation Challenge

Real time attack and defense
Git-based infrastructure

Teaching Assistant

Team1
Team5

Team1 VM
Team5 VM
Git-based Infrastructure

Network, VM, DB, Scoreboard, etc.
Git-based Infrastructure
Git Repository as Service
Distributed System

1. **Clone**
   - Attacker clones a GitHub repository.

2. **Find**
   - Attacker finds a vulnerable program within the cloned repository.

3. **Submit Exploit**
   - Attacker submits the exploit as a GitHub issue.

GitHub Repository

Cloned GitHub Repository

Exploit

Vulnerable Program

Vulnerable Program
Handle Monitoring Challenge

- Real time attack and defense
- Automated verification system
- Git-based infrastructure
- Git Hub
- Team 1
- Team 5 VM
- C4: Contents Creation Challenge

- Teaching Assistant

C1: Interactivity Challenge
C2: Configuration Challenge
C3: Monitoring Challenge
C4: Contents Creation Challenge
Automated Verification System

- Teaching Assistant

 Execute

```
> _
```

- Verify exploit in each round
- Manage the game score
Automated Exploit Verification

Docker Container

Vulnerable Program

Copy & Execute
Random String

Flag
Exploit

Docker Container

Exploit

Execution Result
Public Verification

Publicly Accessible!

Team 1
Local Repository

Team 2
Local Repository

Remote Repository

Service

Verify exploit

Verify exploit

Verify exploit

Local Organizer Repository

Push
Pull

Push
Pull

Push
Pull

Push
Pull

Push
Pull

Push
Pull

Push
Pull

Push
Pull
Handle Contents Creation Challenge

Real time attack and defense

Git-based infrastructure

Team1 VM

Team5 VM

Automated verification system

Teaching Assistant

Shifting creation burden to student

C2: Configuration Challenge

C1: Interactivity Challenge

C3: Monitoring Challenge

C4: Contents Creation Challenge

Git - based infrastructure

Software Security
KAIST
Shifting Creation Burden to Student (BIBIFI, ACM CCS ’16)

Hands-on Development

Prepared Program
Injecting Vulnerabilities

Prepared Program

GitHub Repository

Intended Vulnerability

Injection

Unintended Vulnerability

Vulnerable Program

GitHub Repository
Our Goal: Handle All the Challenges

- Real time attack and defense
- Git-based infrastructure
- Automated verification system
- Shifting creation burden to student
- Teaching Assistant

C1: Interactivity Challenge
C3: Monitoring Challenge
C4: Contents Creation Challenge

Need to invent new problems for every competition.
Evaluation Setup (Preliminary)

- **IS521 Information Security Laboratory** 2018 in KAIST, Korea

- 21 students (11 of them had no experience in CTF), 6 teams

- Preparation: Develop a simple secure messaging application (use either C or C++)

- Injection (Individual): Inject at least one vulnerability

- Exercise (Individual): Report unintended vulnerabilities or functionality bugs
Diversity of Injected Vulnerabilities

The students introduced 28 vulnerabilities in the 6 distinct applications

Help the instructors prepare a diverse set of CTF challenges
Exercise with Unintended Vulnerability

• 14 vulnerabilities and 18 functionality bugs were reported

• Each team had at least one unintended vulnerability

• Unintended vulnerabilities are found mostly by experienced students
Discussion: Binary-Only CTF

GitHub Repository

<source/>
Prepared Source Code

Compile

01011101
00111011
10010010
Compiled Binary
Discussion: Binary-Only CTF

Player

Attack & Defense

GitHub Repository

Compiled Binary

01011101
00111011
10010010
Open Science

https://github.com/SoftSec-KAIST/GitCTF
Demo
Demo Scenario

(1) Turn on evaluator

(2) Attack

Team 1 (Attacker)

Local Repository

Push

Pull

Service

Remote Repository

Push

Pull

(3) Defense

Team 2 (Defender)

Local Repository

Push

Pull

Local Repository

Organizer
Open Science

https://github.com/SoftSec-KAIST/GitCTF
Question?