

CS6265: Information Security Lab

Taesoo Kim

CS6265: Info. Security Lab

- A special course: supervised, **hands-on laboratory**
- Focusing on *reverse engineering* and *binary exploitation*
- Designed for seniors and above (including InfoSec MS, fresh PhDs)
 - **Prerequisite:** OS, system programming, architecture
 - **Background:** low-level programming (e.g., C, asm)

Learning via Capture-the-flag



CTF: Cyber War Game

- Jeopardy
- Attack and defense

Discover Our Unique Challenges Menu			
Amuse Bouche		Signature Dishes	
ELF Crumble		www	
warmup (Ordered by 368 teams)	102pt	prun (Ordered by 10 teams)	240pt
You Already Know		adamtune	
warmup (Ordered by 487 teams)	101pt	misc, ml (Ordered by 3 teams)	416pt
Easy Pisy		SAG?	
crypto, web (Ordered by 190 teams)	104pt	crypto, reverse (Ordered by 11 teams)	228pt
babypwn1805		stumbler	
prun (Ordered by 39 teams)	132pt	reversing (Ordered by 11 teams)	228pt
sbva		Ps-Secure	
web (Ordered by 99 teams)	110pt	reverse, x86-64 (Ordered by 7 teams)	291pt



Topics

- Reverse engineering
- Binary exploitation
- Bug finding
- Memory forensic
- etc.

Schedule: <https://tc.gts3.org/cs6265/2026-fall/cal.html>

Big Picture: Course Structure

- Total 9 labs (week/bi-weekly)
- **Event 1.** In-class, 24h TKCTF Dec 04 at 3:30pm (Fri) - Dec 05 at 3:30pm (Sat)
 - CS6265-hosted CTF event plus **Prizes (\$2,000)** (by SSLab)
 - Each team prepares one challenge for other teams
- **Event 2.** [NSA Codebreaker Challenge](#)

Event 1: TKCTF



Event 2: NSA Codebreaker (Lab 10)



PRESS RELEASE | April 15, 2025

Georgia Tech Wins NSA's Codebreaker Challenge for Fourth Consecutive Year

Weekly Structure (for Lab)

- **Fri** : Cover a single topic/theme (e.g., stack overflow)
- Optional recitations
 - **Mon/Wed** 3:30pm - 4:30pm
 - Location: Coda C1015 (Vinings)
- **Thu** : Deadline for the current week's problem set (i.e., 10 challenges)

In-class Meeting (on Fri)

- Fri 3:30pm - 6:15pm, Klaus 1447
- 30 min: discuss last week's challenges (you will be asked to explain)
- 30 min: cover this week's topic
- 30-60 min: **in-class tutorial (so bring your laptop!)**
- 30-60 min: TA-ing

Course Grading

- *100% Lab* (no single lab returned → F)
- No midterm/final exams
- 9 labs + 4-lab worth events = 13 labs
 - In-class CTF (2-lab worth)
 - NSA Codebreaker (2-lab worth)

Scoring in Each Lab (Game Rules)

- **10 challenges** (20pt x 10 = 200pt) + **1 in-class tutorial** (20pt) = 220pt
- Need to submit **flag** , **write-up w/ an exploit** of each challenge
- **Hint** : each challenge has 1-2 hints → -1pt x #hints revealed
- **Late policy** : 50% of the original points (one extra week)

Ref. Check [Submission Site](#)!

Grading Scheme (Expected)

- Grading Scheme (expected):
 - **A**: Average 7+ chals per lab ($7/10 \times 200\text{pt} + 13 * 20\text{pt} = \mathbf{1660\text{pt+}}$)
 - **B**: Average 6+ chals per lab ($6/10 \times 200\text{pt} + 13 * 20\text{pt} = \mathbf{1460\text{pt+}}$)
 - **C**: Average 5+ chals per lab ($5/10 \times 200\text{pt} + 13 * 20\text{pt} = \mathbf{1260\text{pt+}}$)
 - **D**: Average 5- chals per lab
 - **F**: Below or zero flag submitted for at least one lab.
- Expected distribution: 40%: A, 30-40%: B, 30-20%: C and below
- **If you don't turn in at least one flag for every lab, you will get an F**
- All tutorials are mandatory
- See [Game Rules!](#)

Online Competition

[Class](#) | [Problems](#) | [Scoreboard](#) | [Status](#) | [Chart](#)

[New api-key](#)

lab11

Name	Points	Release	Deadline	Solved	Flag	Exploits
sandbox-pttrace	20	11-18-2016 00:00:00	12-01-2016 00:00:00	9	Submit	Submit
sandbox-seccomp	20	11-18-2016 00:00:00	12-01-2016 00:00:00	4	Submit	Submit
sandbox-pttrace2	20	11-18-2016 00:00:00	12-01-2016 00:00:00	8	Submit	Submit
srop	20	11-18-2016 00:00:00	12-01-2016 00:00:00	7	Submit	Submit
simple-aeg	20	11-18-2016 00:00:00	12-01-2016 00:00:00	3	Submit	Submit
sandbox-pin	20	11-18-2016 00:00:00	12-01-2016 00:00:00	1	Submit	Submit
kproc-zeropage	20	11-18-2016 00:00:00	12-01-2016 00:00:00	2	Submit	Submit
kproc-bufovl	20	11-18-2016 00:00:00	12-01-2016 00:00:00	1	Submit	Submit
kproc-ret2dir	20	11-18-2016 00:00:00	12-01-2016 00:00:00	0	Submit	Submit
kproc-uaf	20	11-18-2016 00:00:00	12-01-2016 00:00:00	0	Submit	Submit

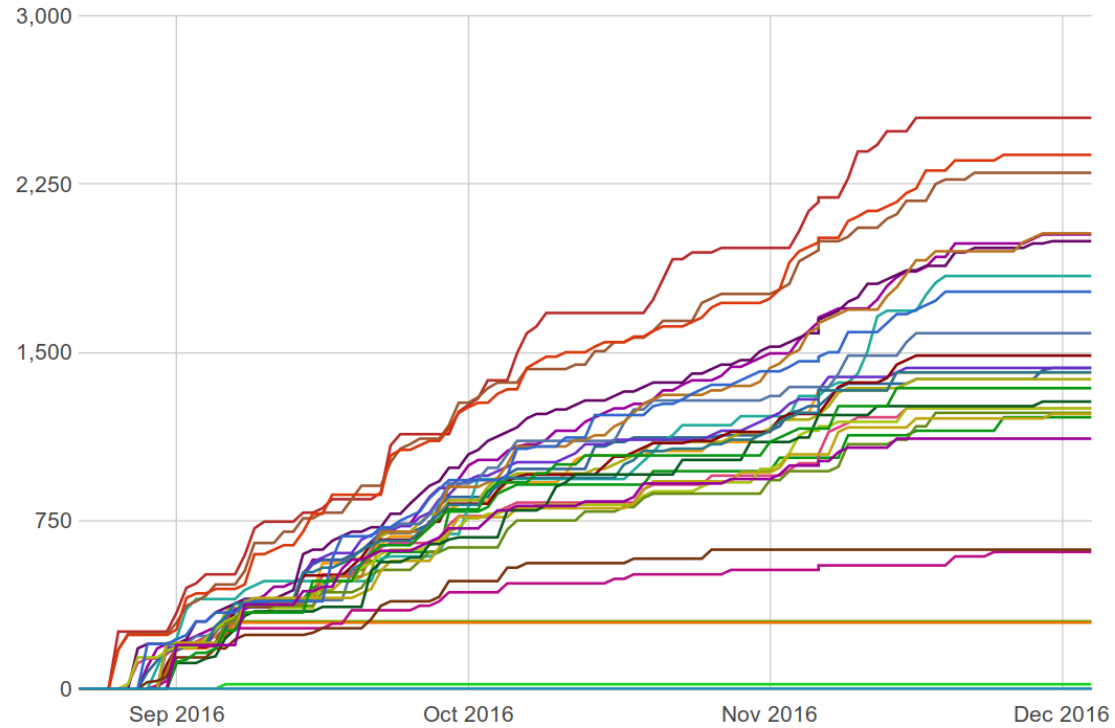
lab10

Name	Points	Release	Deadline	Solved	Flag	Exploits
dlmalloc	20	11-11-2016 00:00:00	12-01-2016 00:00:00	20	Submit	Submit
ptmalloc	20	11-11-2016 00:00:00	12-01-2016 00:00:00	14	Submit	Submit
uaf-basic	20	11-11-2016 00:00:00	12-01-2016 00:00:00	23	Submit	Submit
heap-spray	20	11-11-2016 00:00:00	12-01-2016 00:00:00	20	Submit	Submit

Online Competition

[Class](#) | [Problems](#) | [Scoreboard](#) | [Status](#) | [Chart](#)

Score Charts



Tips for CS6265

- Study in group (e.g., discussion)!
- Come to the recitation (Mon/Wed)!
- Understand your time budget!
- Tackle challenges in order!
- Learn basic tools next two weeks (e.g., editor, debugger, python)!

Misconduct Policy

- Cheating vs. collaboration
- Refer [GT's Academic Misconduct Policy](#)
- ***Never ever*** use/copy other students' code/write-up
- Please ***write down names*** of your collaborators

AI Policy

- AI tools are **allowed** to assist solving lab challenges
- You are expected to **understand your labs**: **do not one-shot challenges**
- Combine AI with the tools we teach (debugger, decompiler, exploit libs)
 - Archive what works as **Skills**, expose tools via **MCP**
- Write-ups **must be human-written**
 - But AI-generated **exploit code** is permitted
- **Never** run AI harnesses unsupervised on the lab servers

See [Use of AI Tools!](#)

About Course Material

- You should ***never*** share exploits/write-up online
- Once found → F (even after the semester is over)
- We are checking your submission against past years' submissions

Team



- Lecturer: Andrew Chin
- TA: Andrew Chin and Joshua Wang
- Instructor of Record: Brendan Saltaformaggio
- Contact: 6265-staff@cc.gatech.edu
- Website: <https://tc.gts3.org/cs6265/2026-fall/>
- Ed Discussion: <https://edstem.org/us/courses/102576/discussion>



Georgia Tech Students Claim Victory at DEF CON's Elite Hacking Competition

■ Aug 19, 2026

Year after year, some of the world's best hackers gather at DEF CON in Las Vegas to put their skills to the test. None is more prominent than the annual Capture the Flag (CTF) competition, where elite cybersecurity teams compete to find vulnerabilities, exploit systems, and defend their own infrastructure.

This year, Team Blue Water – which features nine Georgia Tech participants –came out on top after three days of intense competition.

"I've competed in DEF CON CTFs for four years now," said Ph.D. student **Kevin Stevens**. "After winning second place three years in a row, it's so rewarding to finally achieve first place with such a great team."

Blue Water finished with 578,103 points, well ahead of second-place team 0x4b52, which scored 454,797 points. SuperDiceCodeLovers took third place with 425,686 points.

The results highlighted Georgia Tech's strength in the competition. The first-, second-, and third-place teams all included Georgia Tech students or alumni.

Contact

John Popham
Communications Officer II at the School of Cybersecurity
and Privacy

Categories

Computer Science/Information Technology and
Security

Student and Faculty

Student Competition Winners (academic, innovation,

TA Rules

- Please come to the recitation (Mon/Wed 3:30-4:30pm, Coda C1015)
- Please post your questions on [Ed Discussion](#)
- Feel free to **answer other students' questions**
- Please proactively participate in the online discussion
- Contact 6265-staff@cc.gatech.edu as a last resort (slowest)!

Next Two Weeks

Monday	Tuesday	Wednesday	Thursday	Friday
Aug 24 First day of class (No class)	Aug 25	Aug 26	Aug 27	Aug 28 LEC: Warm-up: x86, Tools TUT: Tut01: GDB/x86 [video] Preparation: Read asm
Aug 31	Sep 01	Sep 02	Sep 03 DUE: Lab 01	Sep 04 LEC: Warm-up: x86_64, Shellcode, Tools TUT: Tut02: Pwndbg, Ghidra, Shellcode [video1], [video2], [video3] Preparation: Read x86_64

Today's Topics

- This week: Bomblab !
- Quick introduction to GDB
- In-class tutorial
 - Walk over x86 asm and tools
 - Be familiarized with GDB and x86 (32-bit)
 - Let's crack crackme0x00 – crackme0x03 binaries

Note on Flag

- Random looking bytes, but be careful. It is designed to include tons of information unique to you, so we can easily check plagiarism

```
$ cat /proc/flag
```

```
CB25682B33EF8BF23545A767562A1D5AA33C88EEACC1AE562D950CB9F1E5725D  
864725DB51460902ECBD52BA4CBED86A10F3A98A35F6FB71871019702A0E9199  
5BC59332C390A3C27D0EC2CE85BC13E956A6027E3171352F90467A8C12346D9A  
2A26EE914B3078ED031FDB14BB6224C3D743D79A733FB49EB4E9C1F383CF810E  
F6841EE935FE2DA2C57DB4804B6823884B36AE62B08848486918C120E4C2AA94  
E1D3F8A6E9E2251AC39E5F37971FB07DFF839E0BC1C4E6C1D4A24E0948F8751B  
25BFFE854CD84A8D8E28814398FF192CD9AD37150D83DA872E944DF1552F97DD
```

```
...
```

Note on Bomblab

```
$ ./bomb
```

```
Enter your api-key: <paste-your-api-key>
```

```
      ,---.!,
    __/  -* - |__ )
,d08b.  '|`  | _ \ / _ \ |' _ \ |' _ \ |' _ \ |' _ \
0088MM   | |_) | ( _) | | | | | | | |_) | | ( _) | |_) |
`9MMP'   |___/ \___/ | _| | _| | _| _| _| _| _| _| _|
          cs6265
```

Welcome to my fiendish little bomb. You have N? phases with
which to blow yourself up. See you alive!

(hint: security question)

>

Be Cautious!

```

      __, -~~/~          \ ---- .
    _/_ , --- (           ,       )
   __ /         <        /      ) \__
- - - - - == ; ; ; '====-----== ; ; ; == - - - - -
      \/ ~'~'~'~'~'~'\~'~')~'/
     (_ ( \ ( > \)
      \( _ <                >_>'
              ~ ` -i' ::>| -- '
                    I;|.|. |
                  <|i::|i|\` .
                 (` ^'' '- ''')

```

WARNING!

- **Don't send us email to restore scores!**
- Be extra cautious about what you are typing..
- But think about how to defeat? (i.e., cheating our server?)
- **ANY** techniques are acceptable and be creative!
- **Read the binary**, check how it works internally, tinker it locally?

DEMO: GDB Summary

- run/continue
- break/tbreak/rbreak/delete
- stepi/nexti/finish
- info reg/proc/break
- backtrace/examine
- gdbinit
- python
- etc.

Pwndbg

- Use `gdb - pwndbg` in the server
- GDB Commands (left side) are enough for Lab01/02

PWNCHEATSHEET

HTTPS://PWNCHEATSHEET.RE/

GDB COMMANDS

file <path>
load binary file to debug

run [<args>...]
run program (with args)

starti [<args>...]
start program and stop at its very first instruction

set args <args>...
set program arguments

break <where>
set a breakpoint

info breakpoints/threads/regs
list breakpoints/threads/register values

delete <breakpoint>
delete a breakpoint

next
go to next (source) line

step
go to next line stepping into functions

ni
go to next instruction

si
go to next instruction stepping into functions

finish
run until current function returns

PWNCHEATSHEET COMMANDS:

pwndbg [<topic>
print info about pwndbg commands

config
show pwndbg configuration

theme
show pwndbg theme configuration

tip [--all]
print tips that are shown during startup

CONTEXT DISPLAY

context [<section>
display context or a given context section (regs, disasm, args, code, stack, backtrace, expressions, ghidra, threads)

set context-sections [<sect1>] [<sect2>...]
set context to display only given sections

ctx-watch eval/execute <expression>
adds a given expression to be shown on context display

START COMMANDS

attachp <pid/name>
attach to given pid or process by part of its name

start [<args>...]
run and stop program at the first found symbol from: main, _main, start, _start, init, _init or entry

entry [<args>...]
run and stop program at its entrypoint address

STACK COMMANDS

retaddr
print return addresses on the stack

canary
print the global stack canary/cookie value and finds canaries on the stack

NAVIGATION

xuntil <where>
continue until an address or function

nextcall
continue to next call instruction

nextjmp
continue to next jump instruction

nextret
continue to next return-like instruction

stepret
step until a ret instruction is found

stopuntilasm <asm code>

p2p <mapping_names> [<mapping_names>...]
pointer to pointer chain search (e.g. p2p stack libc will look for pointers to libc on the stack)

xinfo <where>
show offsets of the specified address from various useful locations

tls
print thread local storage address

MISC COMMANDS

distance <where1> <where2>
compute difference between two addresses

patch <where> '<instructions>...'
patch given address with given code/bytes

patch_list
list all applied patches

patch_revert <patch>
revert a patch

symbol [...]
add, show, load, edit, or delete custom structures in plain C (so they can be used e.g. with print command)

plist [...]
dump elements of a linked list (see help plist)

procinfo
display process information

errno [<errno value>]
print libc's errno error code string

GLIBC HEAP HACKING

heap_config
show glibc allocator hacking configuration

heap

In-class Tutorial

- Step 1: Setup the game environment
 - <https://tc.gts3.org/cs6265/2026-fall/rules.html>
- Step 2: Tutorial (in CTF servers)
 - <https://tc.gts3.org/cs6265/tut/tut01-warmup1.html>

```
$ ssh lab01@52.201.10.159
password: xxxxxxxxx (find on Ed Discussion)
```

```
$ cat README
$ cd tut01-crackme
$ cat README
```

References

- [GDB tutorial](#)
- [x86 instructions](#)
- [x86 architecture](#)