Competitive Computer Security Stuco

Chris Lambert (chrislambert@cmu.edu)

Welcome to 98-212, Competitive Computer Security!

This is a technical course focused on offensive computer security, particularly in the context of the computer security competitions called “Capture the Flag” competitions. Its primary purpose is to teach the tools and tricks used in offensive security, both for the purpose of honing your skills for these competitions and also for use in better understanding what an attacker might actually be capable of.

I'm Chris Lambert—I'm a sophomore in Computer Science and a member of the “Plaid Parliament of Pwning”—which we usually refer to as the slightly-less-ridiculous “PPP.” PPP is Carnegie Mellon's CTF team, as well as a security research group. PPP is ranked as one of the best teams in the world, consistently taking home first place finishes, most recently at RealWorldCTF in Beijing, China and DEF CON CTF in Las Vegas.

CTFs are competitions held both online and in-person at conferences that are competed in by many people active in information security. Many companies send unofficial teams - for example, Raytheon SI, ManTech and Google are represented. Many universities also have teams, such as CMU, MIT, UC Berkeley, UC Santa Barbara, Georgia Tech, Boston University, and RPI.

This course will be held on Mondays from 7:00 to 8:20 pm in DH 2315 (for registered on-campus students only) and on Zoom at

`https://cmu.zoom.us/j/93468517525?pwd=WC84dmNZQ0Q3aW10WGgxUTgzK1R4UT09`

We will cover topics ranging throughout many aspects of computer security, including web security, cryptography, reverse engineering and binary exploitation.

1 Course Policies

The StuCo system requires that I take attendance, so I will. You are allowed 2 unexcused absences, and “excused absences” as needed. There are two requirements to be excused according to the Stuco committee. First, if you have something come up toss me an email and everything will be fine. Second, you have to complete the “homework checkpoint” that is posted for that week. The homework *will be hard* if you do not come to lecture, and will be *easy* if you do.

Grades will be determined by completion of the homework. Time will be provided in class to complete it.

2 Tentative Schedule

The plan is currently to follow roughly the following order. With any lecture day, real examples of challenges from CTFs are provided, with a basic challenge being the assigned homework for the week. Time will be provided in class to work on it, and just like in real CTFs collaboration is *encouraged.*

Week 1	Introduction, course overview. "What is a CTF?", tooling setup, useful resources, and getting set up with the online platform for the course
Week 2	Free-run of selected challenges from actual CTFs. Specifically, interesting challenges that don't require much ahead-of-time knowledge.
Week 3	Introduction to reverse engineering (re): getting comfortable with static reverse engineering tools, using debuggers to confirm your hypotheses
Week 4	Introduction to binary exploitation (pwn): a quick overview of how x86 works, then looking at shellcode and ROP. Introducing pwntools
Week 5	Introduction to web exploitation (web): SQL injections, common PHP mistakes, covering some of the variety of things in this category
Week 6	Introduction to cryptography (crypto): overview of popular cryptosystems (RSA, ElGamal, DHKE)—should be review from 251 for CS students. Covering common attacks on RSA and how to find papers and implement them using Sage.
⋮	⋮

After this point, progress will largely be determined by seeing how fast the course should go. It's the first time this course is being taught in a long time, so I'm not sure how long lectures should last nor how fast they should go. Ideally, I would like to cover the following topics (and probably more) in another round of the categories or depending on what people are most interested in:

- non-x86 architectures (ARM, MIPS, etc.)
- reverse engineering interpreted/VM languages (Java, Lua, Python)
- exploitation on the heap (looking at glibc malloc internals)
- exploiting function pointers (mostly vtables)
- challenges with XSS and CSRF against web clients
- SSRF on web services
- dynamic typing issues with web languages (PHP, Javascript, Ruby)
- symmetric cryptography (block ciphers, stream ciphers, padding)

- demos/working through interesting challenges from old CTFs
- using fuzzers to get starting points for bugs
- history of PPP