Georgia Institute of Technology

Course Syllabus: Advanced Malware Reverse Engineering

Summer 2021, CS/ECE-6747 Advanced Topics in Malware	School of Electrical and Computer Engineering, OMS Computer Science and OMS Cybersecurity, School of Computer Science, College of Computing
Delivery: 100% Web-Based, Asynchronous	Canvas for content delivery
Dates course will run: May 17 – August 5, 2021	

Instructor Information

Brendan Saltaformaggio, PhD	Email: brendan@ece.gatech.edu
Weekly Office Hours via Blue Jeans	Teaching Assistants: *TBD*

General Course Information

Description

This course covers advanced approaches for detecting the presence of vulnerabilities in binary software, the analysis of malicious software, and explores recent research and unsolved problems in software protection and forensics. The goal of this course is to engage in critical discussion around key research topics in software security and forensics. This course will cover: Binary Program Analysis Principles, Binary Software Security, Software Forensics and Cyber Attack Response. Students will be required to study published research papers from the top-tier academic venues in computer security and cyber forensics.

Why take this course? You are interested in learning the fundamental principles of dissecting malware, vulnerability finding/defense, and cyber-attack triage. You want to read cutting-edge research publications on these topics.

Pre- &/or Co-Requisites

There are no Pre-Requisites for this course. However, background knowledge in assembly will be helpful and programming experience in python or C is a must.

Course Goals and Learning Outcomes

After successfully completing this course, students should be able to:

1. Statically reverse engineer malware samples in a disassembler
2. Build static analysis tools to automate control flow recovery and identify intractable indirect jumps
3. Design and implement static analysis routines to perform automated data dependency tracking
4. Instrument binary programs and malware to collect dynamic instruction traces
5. Implement dynamic analysis tools to perform online control dependence tracking
6. Read and present cutting-edge research publications relating to malware analysis, vulnerability finding/defense, and cyber attack triage

Course Materials

Course Text

None. Instead we will study published research papers from the top-tier academic venues in computer security and cyber forensics. We will use a slide show to keep track of the papers we read in this class. Each paper will get at least 1 slide, and the slides must cover: “What problem is the paper focused on?”, “What solutions/techniques are proposed?”, “How did they evaluate their work?”, and “What future research opportunities can you think of?” These slides will be turned in for a grade at the end of the semester.

Additional Materials/Resources

The following books are recommended for additional background or more in-depth understanding of the topics discussed in class. Read these books only if you want to learn more! They will not be covered in lectures or on exams!

- **Practical Tools/Techniques For Malware Reverse Engineering:**
  Michael Sikorski, Andrew Honig. Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. No Starch Press, 2012. ISBN: 978-1593272906

- **Practical Tools/Techniques For Memory Forensics:**
  Michael Hale Ligh, Andrew Case, Jamie Levy, AAron Walters. The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory. Wiley, 2014. ISBN: 978-1118825099

- **Background On Low-Level Computer Systems Programming:**
  Randal E. Bryant, David R. O'Hallaron. Computer Systems: A Programmer's Perspective. Pearson (3rd Edition), 2015. Online: http://csapp.cs.cmu.edu/. ISBN: 978-0134092669

You may also need a copy of the Intel Developer’s manuals. These are free and available via this link: http://www.intel.com/content/www/us/en/processors/architectures-software-developer-manuals.html It's large, but the best PDF to get is the combined set, downloadable via the first link on that page. If you have an iPad or other tablet, drop this PDF on it and read it whenever you have spare time.

Course Website and Other Classroom Management Tools

This class will use Canvas to deliver course materials to online students. ALL course materials and activities will take place on Canvas.

Course Requirements, Assignments & Grading

Assignment Distribution and Grading Scale

Assignment	Release Date	Due Date EST 11:59pm	Weight
Six Mini-Projects			**90%**
Lab #1: Intro to GHIDRA	May 24	May 30	15%
Lab #2: Static Malware Reverse Engineering	May 31	June 20	15%
Lab #3: Basic Def Use GHIDRA Plugin	June 21	July 3	15%
Lab #4: Data Dependence GHIDRA Plugin	June 28	July 11	15%
Lab #5: Dynamic Control Flow Tracing	July 12	July 25	15%
Lab #6: Dynamic Control Dependence	July 19	August 5	15%
Reading Slides & Piazza Participation		August 5	**10%**

Grading Scale

Your final grade will be assigned as a letter grade according to the following scale:

A	90-100%
B	80-89%
C	70-79%
D	60-69%
F	0-59%

Description of Graded Components

Mini-Projects

There will be 6 mini-projects during the Binary Analysis Principles portion of the class. 3 of the projects will be static analysis with GHIDRA and 3 will be dynamic analysis with Pin. Each project will require careful time allocation to complete on time (1 or 2 week deadlines). Grades will be based on the results produced by your tool. For some mini-projects, we will schedule demos during office hours if needed.

The mini-projects will cover the following topics:

1. Intro. to Software Disassembly
2. Manual Static Malware Reverse Engineering
3. Automated Static Malware Analysis
4. Static Data Dependence Detection
5. Dynamic Control Flow Analysis
6. Dynamic Control Dependence Detection

Reading Slides

Each week's lesson is accompanied by published research papers from the top-tier academic venues in computer security and cyber forensics. Please read these research papers as pre-readings to prepare for each class. Each student will use a slide show to keep track of these papers as they read them. Each paper must get at least 1 slide, and the slides must cover the following for each paper: "What problem is the paper focused on?", "What solutions/techniques are proposed?", "How did they evaluate their work?", and "What future research opportunities can you think of?" These slides will be turned in for a grade at the end of the semester. Please keep it simple! 1 or 2 sentences for each question is sufficient. The grade is based on having a slide for all the papers and your understanding of each paper.

Piazza Participation

Students need to post at least once on Piazza for full participation credits.

Submitting Assignments

Mini projects will be available during the period described above. Each project will have details of the files/documents that need to be submitted. Again, these should be submitted by time they are due.

Sending assignments (projects etc.), whether early, on time, or late to the professors or TA is not permitted and will not be accepted. All projects must be completed and submitted within Canvas. If there are technical issues, please notify the help desk, as well as the professor immediately.

Assignment Due Dates

All assignments a will be due at the times listed on Canvas. The professor may extend deadlines when necessary, so please check back often. Please convert from EST to your local time zone using a Time Zone Converter.

Late and Make-up Work Policy

The Modules follow a logical sequence that includes knowledge-building and experience-building. Late submissions will only be accepted for extreme and documented unforeseen circumstances, on a case-by-case basis, via the professor's approval.

Grading and Feedback

Mini projects will be graded and provided with explanation for mistakes in two weeks after the due date.

Technology Requirements and Skills

Computer Hardware and Software

- High-speed Internet connection
- Laptop or desktop computer with a **minimum** of a 2 GHz processor and 4 GB of RAM
- Windows for PC computers OR Mac iOS for Apple computers
- Complete Microsoft Office Suite or comparable and ability to use Adobe PDF software (install, download, open and convert)
- Linux operating systems familiarity, including how system calls are used
- Virtualization software such as VirtualBox and ability to create and launch virtual machines
- Software development, compiling and debugging tools as required

Technology Help Guidelines

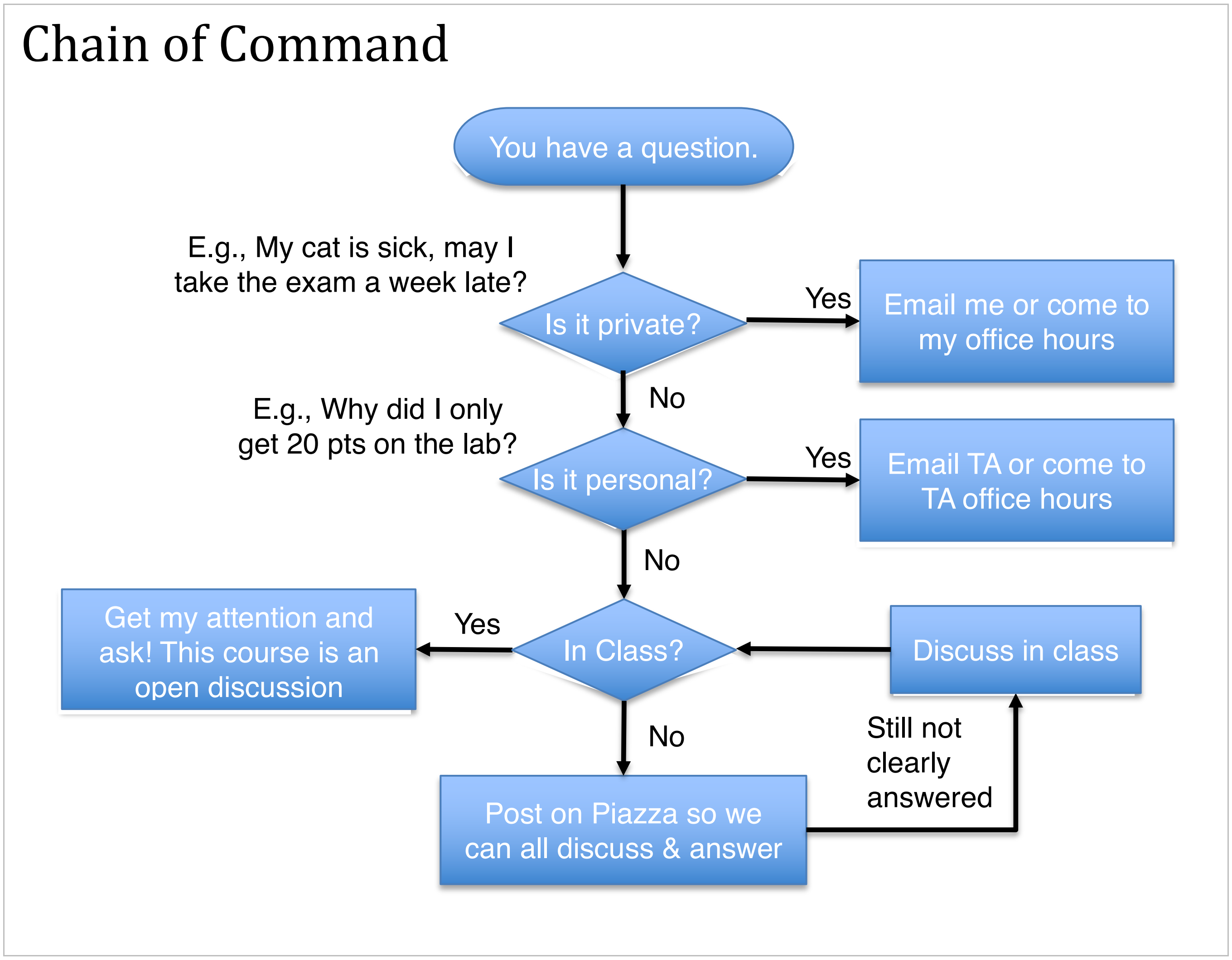
30-Minute Rule: When you encounter struggles with technology, give yourself 30 minutes to ‘figure it out.’ If you cannot, then post a message to the discussion board; your peers may have suggestions to assist you. You are also directed to contact the Helpdesk 24/7.

When posting or sending email requesting help with technology issues, whether to the Helpdesk, message board, or me use the following guidelines:

- Include a descriptive title for the subject field that includes 1) the name of course 2) the issue. Do NOT just simply type “Help” into the subject field or leave it blank.
- List the steps or describe the circumstance that preceded the technical issue or error. Include the exact wording of the error message.
- When possible, always include a screenshot(s) demonstrating the technical issue or error message.
- Also include what you have already tried to remedy the issue (rebooting, trying a different browser, etc.).

Course Policies, Expectations & Guidelines

Remember The Flow Chart From Course Introduction:


Chain of Command
You have a question.
E.g., My cat is sick, may I take the exam a week late?
Is it private?
Yes
Email me or come to my office hours
No
E.g., Why did I only get 20 pts on the lab?
Is it personal?
Yes
Email TA or come to TA office hours
No
Get my attention and ask! This course is an open discussion
Yes
In Class?
Discuss in class
No
Still not clearly answered
Post on Piazza so we can all discuss & answer

Communication Policy

- Email personal concerns, including grading questions, to the TAs or professor privately. Do NOT submit posts of a personal nature to the discussion board unless it is a private post on Piazza.
- Student Forum/Q&A discussion boards will be checked twice per day Monday through Friday; Saturday and Sunday, these discussion boards will be checked once per day. Piazza is absolutely the best way to get answers!
- The professor or TAs may even "redirect" you to ask a question on Piazza instead of email/office hours if it would benefit the entire class. We are all against the malware together!

- Email will be checked at least once per day, Monday through Friday. On Saturday and Sunday, email may be checked but there is no guarantee. During the week, I will respond to all emails within 24 hours; on weekends and holidays, allow up to 48 hours. If there are special circumstances that will delay my response, I will make an announcement to the class.
- The professor and TAs will hold virtual office hours using Bluejeans. Weekly office hour times are posted on Canvas. Weekly office hour times will not change unless an emergency comes up and we will make an announcement. Also, special office hours may be announced before exams. Such office hours will be announced in advance.

Online Student Conduct and (N)etiquette

Communicating appropriately in the online classroom can be challenging. In order to minimize this challenge, it is important to remember several points of **"internet etiquette"** that will smooth communication for both students and instructors:

1. *Read first, Write later*. Read the ENTIRE set of posts/comments on a discussion board before posting your reply, in order to prevent repeating commentary or asking questions that have already been answered.
2. *Avoid language that may come across as strong or offensive.* Language can be easily misinterpreted in written electronic communication. Review email and discussion board posts BEFORE submitting. Humor and sarcasm may be easily misinterpreted by your reader(s). Try to be as matter-of-fact and professional as possible.
3. *Follow the language rules of the Internet.* Do not write using all capital letters, because it will appear as shouting. Also, the use of emoticons can be helpful when used to convey nonverbal feelings. ☺️
4. *Consider the privacy of others.* Ask permission prior to giving out a classmate's email address or other information.
5. *Keep attachments small*. If it is necessary to send pictures, change the size to an acceptable 250kb or less (one free, web-based tool to try is picresize.com).
6. *No inappropriate material.* Do not forward virus warnings, chain letters, jokes, etc. to classmates or instructors. The sharing of pornographic material is forbidden.

NOTE*: The instructor reserves the right to remove posts that are not collegial in nature and/or do not meet the Online Student Conduct and Etiquette guidelines listed above.*

University Use of Electronic Email

A university-assigned student e-mail account is the official university means of communication with all students at Georgia Institute of Technology. Students are responsible for all information sent to them via their university-assigned e-mail account. If a student chooses to forward information in their university e-mail account, he or she is responsible for all information, including attachments, sent to any other e-mail account. To stay current with university information, students are expected to check their official university e-mail account and other electronic communications on a frequent and consistent basis. Recognizing that some communications may be time-critical, the university recommends that electronic communications be checked minimally twice a week.

Plagiarism & Academic Integrity

Georgia Tech aims to cultivate a community based on trust, academic integrity, and honor. Students are expected to act according to the highest ethical standards. All students enrolled at Georgia Tech, and all its campuses, are to perform their academic work according to standards set by faculty members, departments, schools and colleges of the university; and cheating and plagiarism constitute fraudulent misrepresentation for which no credit can be given and for which appropriate sanctions are warranted and will be applied. For information on Georgia Tech's Academic Honor Code, please visit http://www.catalog.gatech.edu/policies/honor-code/ or http://www.catalog.gatech.edu/rules/18/.

Any student suspected of cheating or plagiarizing on a quiz, exam, or assignment will be reported to the Office of Student Integrity, who will investigate the incident and identify the appropriate penalty for violations.

Copyright

The course readings include research papers that are available in the public domain or via the Georgia Tech library. As specified by publishers' copyright notices, the papers will be for individual use only. Similarly, course materials such as quiz and exam questions and project descriptions are for your use only and should not be published or disseminated.

Accommodations for Students with Disabilities

If you are a student with learning needs that require special accommodation, contact the Office of Disability Services at (404)894-2563 or http://disabilityservices.gatech.edu/, as soon as possible, to make an appointment to discuss your special needs and to obtain an accommodations letter. Please also e-mail me as soon as possible in order to set up a time to discuss your learning needs.

Collaboration & Group Work

You are encouraged to form virtual groups to discuss topics covered in class. Such discussion can enhance learning and could include clarifications of questions related to a topic or a project. However, individual work that you submit as part of an assessment and claim as yours must be yours. Also, you have the option to complete assignments with individually or as a team of two.

You are strongly urged to familiarize yourself with the **GT Student Honor Code (Links to an external site.) rules. Specifically, the following is not allowed:**

- Copying, with or without modification, someone else's work when this work is not meant to be publicly accessible *(e.g., a classmate's program or solution).*
- Submission of material that is wholly or substantially identical to that created or published by another person or persons, without adequate credit notations indicating authorship *(plagiarism).*
- Putting your projects on public Github. If a student in the future copies your code/reports, the student obviously violates the honor code but you will also be responsible for the violation.

Any public material that you use *(open-source software, help from a text, or substantial help from a friend, etc...)* should be acknowledged explicitly in anything you submit. If you have any doubt about whether something is allowed or not, please do check with the class Instructor or the TA.

Student-Faculty Expectations Agreement

At Georgia Tech we believe that it is important to strive for an atmosphere of mutual respect, acknowledgement, and responsibility between faculty members and the student body. See http://www.catalog.gatech.edu/rules/22/ for an articulation of some basic expectation that you can have of me and that I have of you. In the end, simple respect for knowledge, hard work, and cordial interactions will help build the environment we seek. Therefore, I encourage you to remain committed to the ideals of Georgia Tech while in this class.

Subject to Change Statement

The syllabus and course schedule may be subject to change. Changes will be communicated via the Canvas announcement tool and/edX bulk email and or the class Piazza discussion forum. It is the responsibility of students to stay current.

Course Schedule

Week	Date	Topic	Deliverables
			All assignments are due at 11:59 PM EST on the indicated dates.
Week 1	May 17	Module 1: Getting Started: Course Introduction, Assembly Language	• Extra Credit #1 Released • Extra Credit #1 Due May 23
Week 2	May 24	Module 2: Welcome to GHIDRA	• Lab #1 Released • **Lab #1 Due on May 30**
Week 3	May 31	Module 3: Static Malware Analysis Tools and Techniques	• Lab #2 Released
Week 4	June 7	Module 4: High Level Language Constructs in Assembly	• Lab #2 Continues
Week 5	June 14	Module 5: Software Representations	• **Lab#2 Due June 20**
Week 6	June 21	Module 6: How to Build and run GHIDRA Plug-ins	• Lab #3 Released
Week 7	June 28	Modules 7 & 8: Dynamic Analysis Tools and Techniques	• **Lab #3 Due July 3** • Lab#4 Released
Week 8	July 5	Module 9: Execution Tracing	• **Lab #4 Due July 11**
Week 9	July 12	Module 10: How to Access Malware Analysis Sandbox VM	• Lab #5 Released
Week 10	July 19	Module 11: Program Slicing	• **Lab #5 Due July 25** • Lab #6 Released
Week 11	July 26	Module 12: Symbolic Execution	• **Lab #6 Due August 5**

Reading List

Readings can be found in GT Library E-reserves at https://ereserves.library.gatech.edu/ares/. To access, you need to log in with your GT credentials.

Module 1 Pre-Readings:

- None

Module 2 Pre-Readings:

- Wressnegger, Christian & Yamaguchi, Fabian & Maier, Alwin & Rieck, Konrad. (2016). Twice the Bits, Twice the Trouble: Vulnerabilities Induced by Migrating to 64-Bit Platforms. 10.1145/2976749.2978403.
- Thompson, Ken. (2007). Reflections on trusting trust. 1983. 10.1145/1283920.1283940.
- Caballero, Juan & Grier, Chris & Kreibich, Christian & Paxson, Vern. (2011). Measuring Pay-per-Install: The Commoditization of Malware Distribution. Proc. 20th Usenix Conf. Security, Usenix Assoc.

Module 3 Pre-Readings:

- Ge, Xinyang & Payer, Mathias & Jaeger, Trent. (2017). An Evil Copy: How the Loader Betrays You. 10.14722/ndss.2017.23199.
- Garcia, Luis & Brasser, Ferdinand & Cintuglu, Mehmet & Sadeghi, Ahmad-Reza & Mohammed, Osama & Zonouz, Saman. (2017). Hey, My Malware Knows Physics! Attacking PLCs with Physical Model Aware Rootkit. 10.14722/ndss.2017.23313.
- Snow, Kevin & Rogowski, Roman & Werner, Jan & Koo, Hyungjoon & Monrose, Fabian & Polychronakis, Michalis. (2016). Return to the Zombie Gadgets: Undermining Destructive Code Reads via Code Inference Attacks. 954-968. 10.1109/SP.2016.61.

Module 4 Pre-Readings:

- Wang, Ruoyu & Shoshitaishvili, Yan & Bianchi, Antonio & Machiry, Aravind & Grosen, John & Grosen, Paul & Kruegel, Christopher & Vigna, Giovanni. (2017). Ramblr: Making Reassembly Great Again. 10.14722/ndss.2017.23225.
- Bichsel, Benjamin & Raychev, Veselin & Tsankov, Petar & Vechev, Martin. (2016). Statistical Deobfuscation of Android Applications. 343-355. 10.1145/2976749.2978422.
- Deaconescu, Razvan & Chiroiu, Mihai & Davi, Lucas & Enck, William & Sadeghi, Ahmad-Reza. (2016). SandScout: Automatic Detection of Flaws in iOS Sandbox Profiles. 704-716. 10.1145/2976749.2978336.

Module 5 Pre-Readings:

- Abadi, Martín & Budiu, Mihai & Erlingsson, Úlfar & Ligatti, Jay. (2009). Control-flow integrity: Principles, implementations, and applications. ACM Trans. Inf. Syst. Secur.. 13. 10.1145/1609956.1609960.
- Backes, Michael & Bugiel, Sven & Derr, Erik. (2016). Reliable Third-Party Library Detection in Android and its Security Applications. 356-367. 10.1145/2976749.2978333.
- Pawlowski, Andre & Contag, Moritz & van der Veen, Victor & Ouwehand, Chris & Holz, Thorsten & Bos, Herbert & Athanasopoulos, Elias & Giuffrida, Cristiano. (2017). MARX: Uncovering Class Hierarchies in C++ Programs. 10.14722/ndss.2017.23096.

Module 6 Pre-Readings:

- Lee, JongHyup & Avgerinos, Thanassis & Brumley, David. (2011). TIE: Principled Reverse Engineering of Types in Binary Programs.
- Slowinska, Asia & Stancescu, Traian & Bos, Herbert. (2011). Howard: A Dynamic Excavator for Reverse Engineering Data Structures..

- Sun, Mingshen & Wei, Tao & Lui, John C.s. (2016). TaintART: A Practical Multi-level Information-Flow Tracking System for Android RunTime. 331-342. 10.1145/2976749.2978343.

Module 7 Pre-Readings:

- Sharif, Monirul & Lanzi, Andrea & Giffin, Jonathon & Lee, Wenke. (2009). Automatic Reverse Engineering of Malware Emulators. Proceedings - IEEE Symposium on Security and Privacy. 94 - 109. 10.1109/SP.2009.27.
- Coogan, Kevin & Lu, Gen & Debray, Saumya. (2011). Deobfuscation of Virtualization-Obfuscated Software A Semantics-Based Approach. 275-284. 10.1145/2046707.2046739.
- Newsome, James & Song, Dawn. (2005). Dynamic Taint Analysis for Automatic Detection, Analysis, and SignatureGeneration of Exploits on Commodity Software..

Module 8 Pre-Readings:

- Shao, Yuru & Ott, Jason & Jia, Yunhan & Qian, Zhiyun & Mao, Zhuoqing. (2016). The Misuse of Android Unix Domain Sockets and Security Implications. 80-91. 10.1145/2976749.2978297.
- Zaidenberg, Nezer & Khen, Eviatar. (2015). Detecting Kernel Vulnerabilities During the Development Phase. 224-230. 10.1109/CSCloud.2015.91.
- Pan, Xiaorui & Wang, Xueqiang & Duan, Yue & Wang, XiaoFeng & Yin, Heng. (2017). Dark Hazard: Learning-based, Large-Scale Discovery of Hidden Sensitive Operations in Android Apps. 10.14722/ndss.2017.23265.

Module 9 Pre-Readings:

- He, Liang & Cai, Yan & Hu, Hong & Su, Purui & Liang, Zhenkai & Yang, Yi & Huang, Huafeng & Yan, Jia & Jia, Xiangkun & Feng, Dengguo. (2019). Automatically Assessing Crashes from Heap Overflows.
- Koppe, Philipp & Kollenda, Benjamin & Fyrbiak, Marc & Kison, Christian & Gawlik, Robert & Paar, Christof & Holz, Thorsten. (2019). Reverse Engineering x86 Processor Microcode.
- Shin, E.C.R. & Song, D. & Moazzezi, R.. (2015). Recognizing functions in binaries with neural networks. USENIX Security Symposium. 611-626.

Module 10 Pre-Readings:

- Xu, Jun & Mu, Dongliang & Chen, Ping & Xing, Xinyu & Wang, Pei & Liu, Peng. (2016). CREDAL: Towards Locating a Memory Corruption Vulnerability with Your Core Dump. 529-540. 10.1145/2976749.2978340.
- Cui, Weidong & Peinado, Marcus & Cha, Sang & Fratantonio, Yanick & Kemerlis, Vasileios. (2016). RETracer: triaging crashes by reverse execution from partial memory dumps. 820-831. 10.1145/2884781.2884844.
- Postmortem Program Analysis with Hardware-Enhanced Post-Crash Artifacts.
  - URL: https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-xu.pdf

Module 11 Pre-Readings:

- Halderman, J. & Schoen, Seth & Heninger, Nadia & Clarkson, William & Paul, William & Calandrino, Joseph & Feldman, Ariel & Appelbaum, Jacob & Felten, Edward. (2009). Lest We Remember: Cold-Boot Attacks on Encryption Keys. Commun. ACM. 52. 91-98. 10.1145/1506409.1506429.
- Lin, Zhiqiang & Rhee, Junghwan & Zhang, Xiangyu & Xu, Dongyan & Jiang, Xuxian. (2011). SigGraph: Brute Force Scanning of Kernel Data Structure Instances Using Graph-based Signatures.
- DSCRETE: Automatic Rendering of Forensic Information from Memory Images via Application Logic Reuse
  - URL: https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-saltaformaggio.pdf

Module 12 Pre-Readings:

- X-Force: Force-Executing Binary Programs for Security Applications
  - URL: https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-peng.pdf
- Blanket Execution: Dynamic Similarity Testing for Program Binaries and Components
  - URL: https://www.usenix.org/system/files/conference/usenixsecurity14/sec14-paper-egele.pdf
- Stephens, Nick & Grosen, John & Salls, Christopher & Dutcher, Andrew & Wang, Ruoyu & Corbetta, Jacopo & Shoshitaishvili, Yan & Kruegel, Christopher & Vigna, Giovanni. (2016). Driller: Augmenting Fuzzing Through Selective Symbolic Execution. 10.14722/ndss.2016.23368.