CS7038 - Malware Analysis - Wk01.1
Introduction

Coleman Kane
kaneca@mail.uc.edu

January 10, 2017

Me

- CSE Ph.D. Candidate, focus in Cyber Operations
- Senior Cyber Intelligence Analyst at GE Aviation
- Email: kaneca@mail.uc.edu
- Office hours: On request - *easiest for me is MWF around 15:30 (3:30pm)*
- Linux community, FreeBSD developer
- Research focus: Malware analysis (disassembly-level analysis)
- BSides Cincinnati conference organizer (*May 20 2017*)

Source: `http://www.deviantart.com/art/Nyan-Cat-207641828`

Malware

A brief definition: Malware is any unwanted software or code on a system that is intended to perform unauthorized actions on that system.

“Software or code” can be executable programs, or even files that cause controlled code execution to occur.

Common malware objectives:

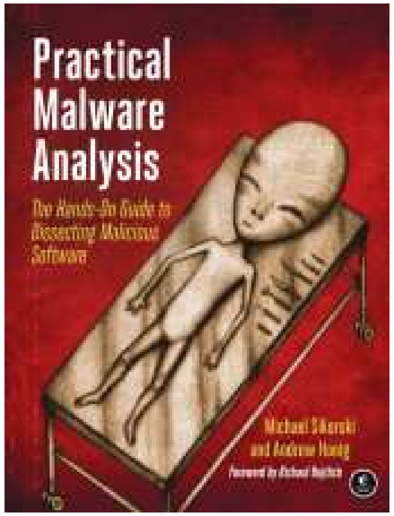
- Download a program to run on the system
- Delete files
- Encrypt & Delete files (“ransomware”)
- Remote access (e.g. backdoors)
- Prank system users
- Private information theft
- Anonymized attack source
- ...many many more...

Disclaimer

You will learn a number of techniques helpful for attacking systems, as well as other techniques that may facilitate other illegitimate goals. Many of these will be older, but still present concepts that continue to apply today. You are responsible for using this knowledge for academic and research purposes only, within the scope of the examples & assignments provided in this class and in your research efforts at The University. Abuse of these methods for illegal purposes is against policy. You will learn how attacks are employed, in order to become better analysts yourselves.

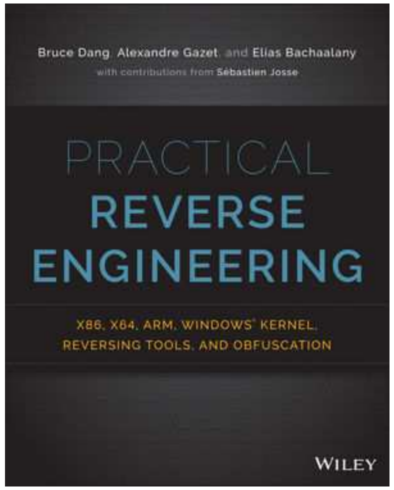
Recommended Texts

Majority of the course is provided from my experience. However, there are a couple books I may use from time to time, and recommend especially if you want third-party assistance with the class.


Practical Malware Analysis
The Hands-On Guide to Dissecting Malicious Software
Michael Sikorski and Andrew Honig
Foreword by Richard Bejtlich

Practical Malware Analysis

https://www.nostarch.com/malware


Bruce Dang, Alexandre Gazet, and Elias Bachaalany
with contributions from Sébastien Josse
PRACTICAL REVERSE ENGINEERING
X86, X64, ARM, WINDOWS KERNEL, REVERSING TOOLS, AND OBFUSCATION
WILEY

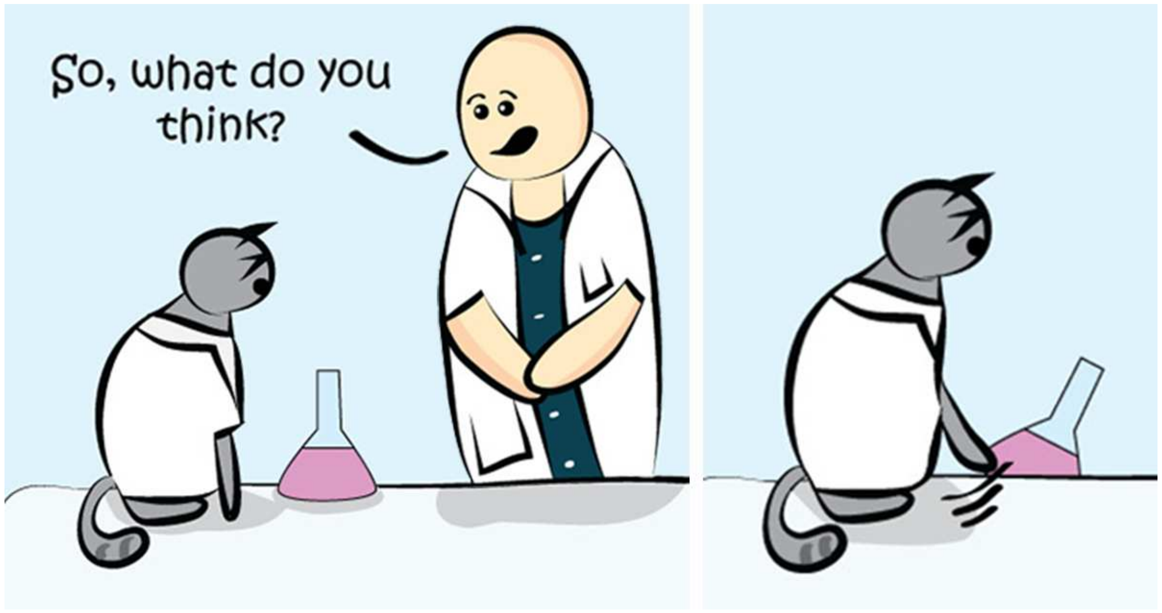
Practical Reverse Engineering

http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118787315,subjectCd-CSJ0.html

About This Class (1/2)

The class will largely be divided into two complementary phases of artifact analysis:

- **Static analysis**: Identifying malware through analysis of the contents of an artifact – *"How does it look?"*
- **Dynamic analysis**: Identifying malware through analysis of system effects produced by loading or running an artifact in a contained environment – *"How does it behave?"*



Source: http://www.boredpanda.com/science-cat-funny-comic-pearls-of-raw-nerdism/

About This Class (2/2)

The class will focus on applying techniques and concepts learned in class to the solution of a series of homework assignments weekly, as well as a final project.

It is important to familiarize yourself with the following technologies:

- Linux OS (I use Ubuntu, and will do so in many of my examples)
- Windows OS (Most common attack target)
- VirtualBox (`http://www.virtualbox.org`)
- IDA (`https://www.hex-rays.com/products/ida/`)
- Immunity Debugger (`http://debugger.immunityinc.com/`)
- Scripting programming language (**Python**, Perl, etc.)
- Yara (`http://virustotal.github.io/yara/`)

I'll make sure to add more to the list as the class develops

Tentative Schedule

This schedule is subject to change:

- **Week 01**: Intro, VirtualBox & Lab VMs
- **Week 02**: Example Cyber Attack
- **Week 03**: Malware Taxonomy/Terminology
- **Week 04**: Static Executable Analysis Overview
- **Week 05**: x86/amd64 assembly crash-course - IDA
- **Week 06**: Binary-level C data structure & construct analysis
- **Week 07**: PDF Document structure / analysis + PDFjavascript
- **Week 08**: OLE Document structure / analysis + VBA Macro
- **Week 09**: Obfuscation / exploits - carrier documents
- *Week 10: Spring break - NO CLASS*
- **Week 11**: Dynamic Analysis Overview
- **Week 12**: Run Time Debugging - Immunity Dbg - Memory
- **Week 13**: System level effects, breadcrumbs/footprints, etc.
- **Week 14**: Instrumentation, Tracing, Building detection / mitigation
- **Week 15**: Analysis Automation?
- ***Week 16: Finals week - Final project***

Pre-Requisites

There are no specific technical pre-requisites for registration for the course. However, depending on your experience, you may need to take extra time in order to teach yourself or expand your understanding on some topics.

Some of the base requirements are below. A very brief refresher of these, if any, will be included in the course:

- Windows OS Internals (Filesystem Layout, Registry, Process memory space, PE file format, Kernel entities)
- Linux OS Internals (syscalls, POSIX API, filesystem layout, file types, bash scripting)
- Programming languages (some C, C++, and Python - reading and writing)
- Assembly Language (primarily x86 and amd64)

If you aren't strong in one of the above topics, I strongly recommend you find some resources to help you improve your knowledge and skill set in any of these areas. The UC Courses "Operating Systems" and "Assembly Language" provide adequate foundations.

Other Odds and Ends

Labs will be assigned more-or-less weekly. I do not have them all ahead of time, so I cannot provide all of them ahead of time. Will publish as I have time.

Homework submissions will occur through UC BlackBoard, unless specified otherwise. Assignment instructions will likely be posted publicly.

Planning to host a website for the course on campus, and record and host lecture videos there. URL is forthcoming.