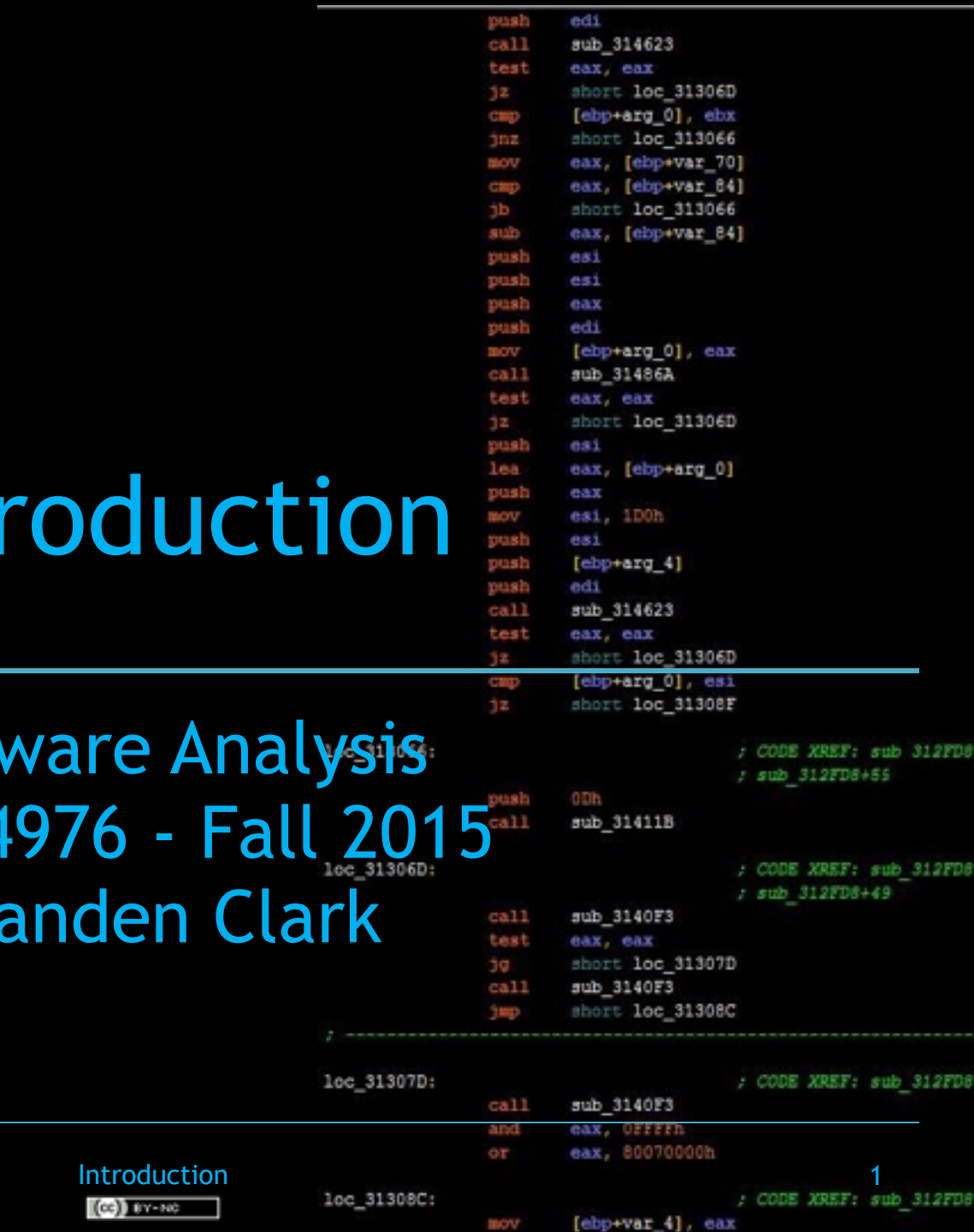
Introduction

Malware Analysis
CSCI 4976 - Fall 2015
Branden Clark

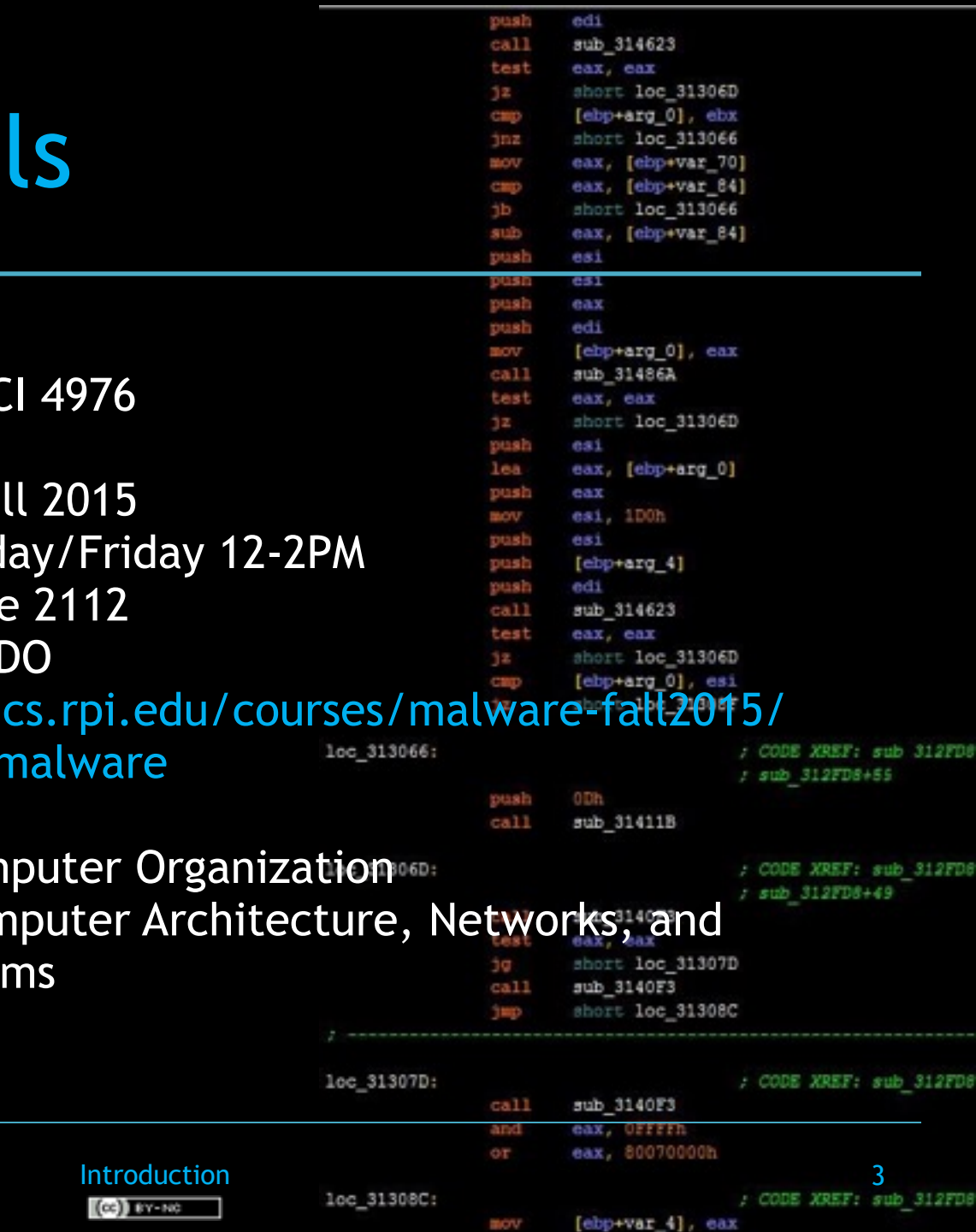


Lecture Overview

1. Syllabus
2. Course Overview
3. Basic Analysis

Course Details

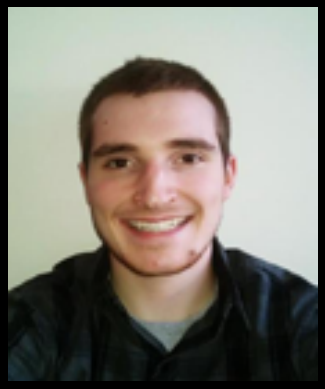
- Malware Analysis
  - Course Number: CSCI 4976
  - Credit Hours: 4
  - Semester / Year: Fall 2015
  - Meeting Days: Tuesday/Friday 12-2PM
  - Room Location: Sage 2112
  - Course Website: TODO
    - http://security.cs.rpi.edu/courses/malware-fall2015/
    - http://rpis.ec/malware
  - Prereqs:
    - CSCI 2500 - Computer Organization
    - ECSE 2660 - Computer Architecture, Networks, and Operating Systems



Instructor

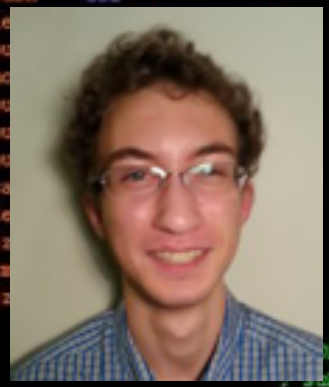
- Instructor: Dr. Bülent Yene
  - Office: Lally 310
  - Email: yener@cs.rpi.edu

Malware Mentors



Branden
(Clark)

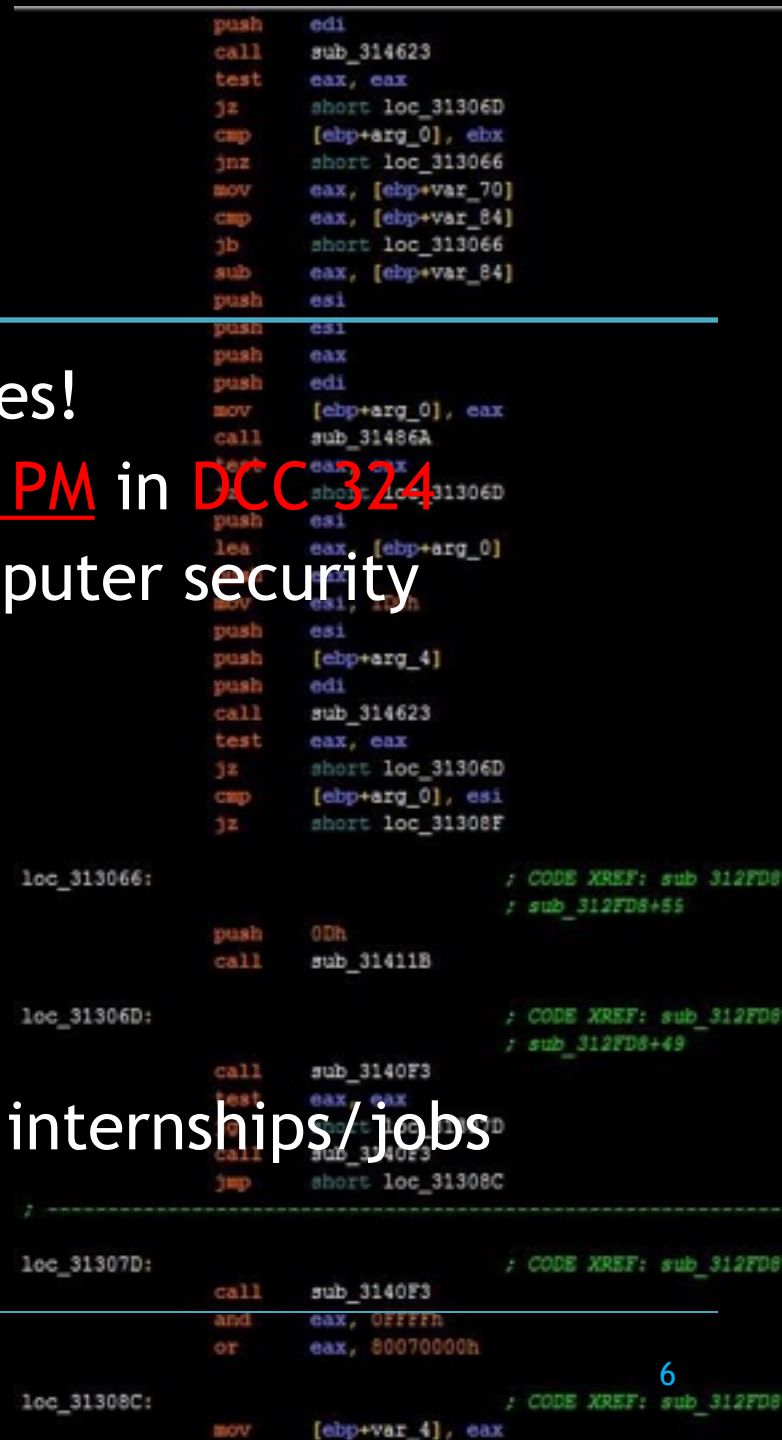
Aaron
(Aidielse)



Austin
(Lense)

RPISEC

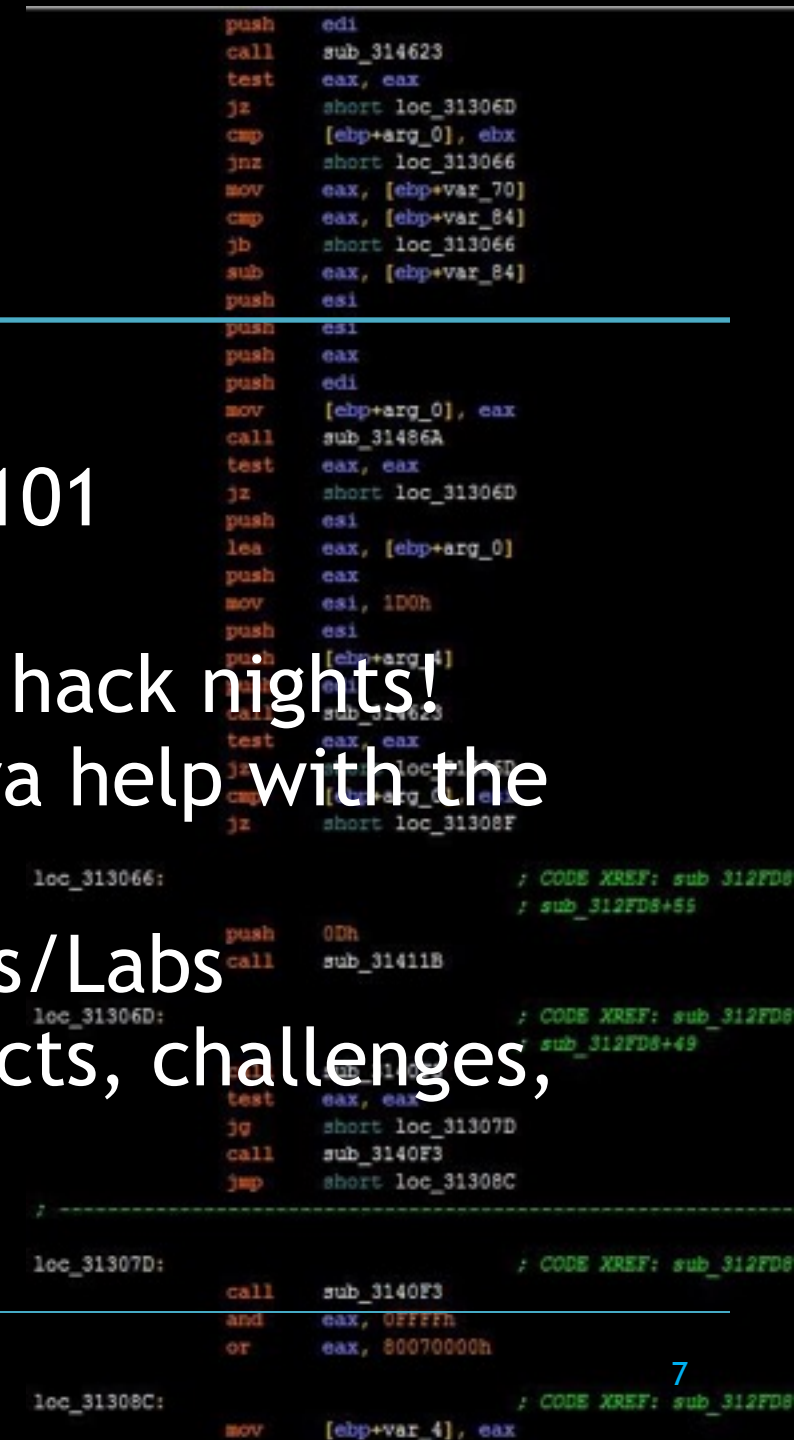
- Good to see lots of familiar faces!
- RPISEC meetings are Friday 5-7 PM in DCC 324
- Come learn other topics in computer security
  - Web hacking
  - Malware analysis
  - Reverse Engineering
  - Digital Forensics
  - So so much more
- Meet people from industry, get internships/jobs
- Read more - http://rpis.ec



Office Hours

- Office hours:
  - Wed 7-10 PM @ Sage 5101

- Come hang out at RPISEC hack nights!
  - Ask questions, get extra help with the class
  - Collaborate on Projects/Labs
  - Work on security projects, challenges, etc



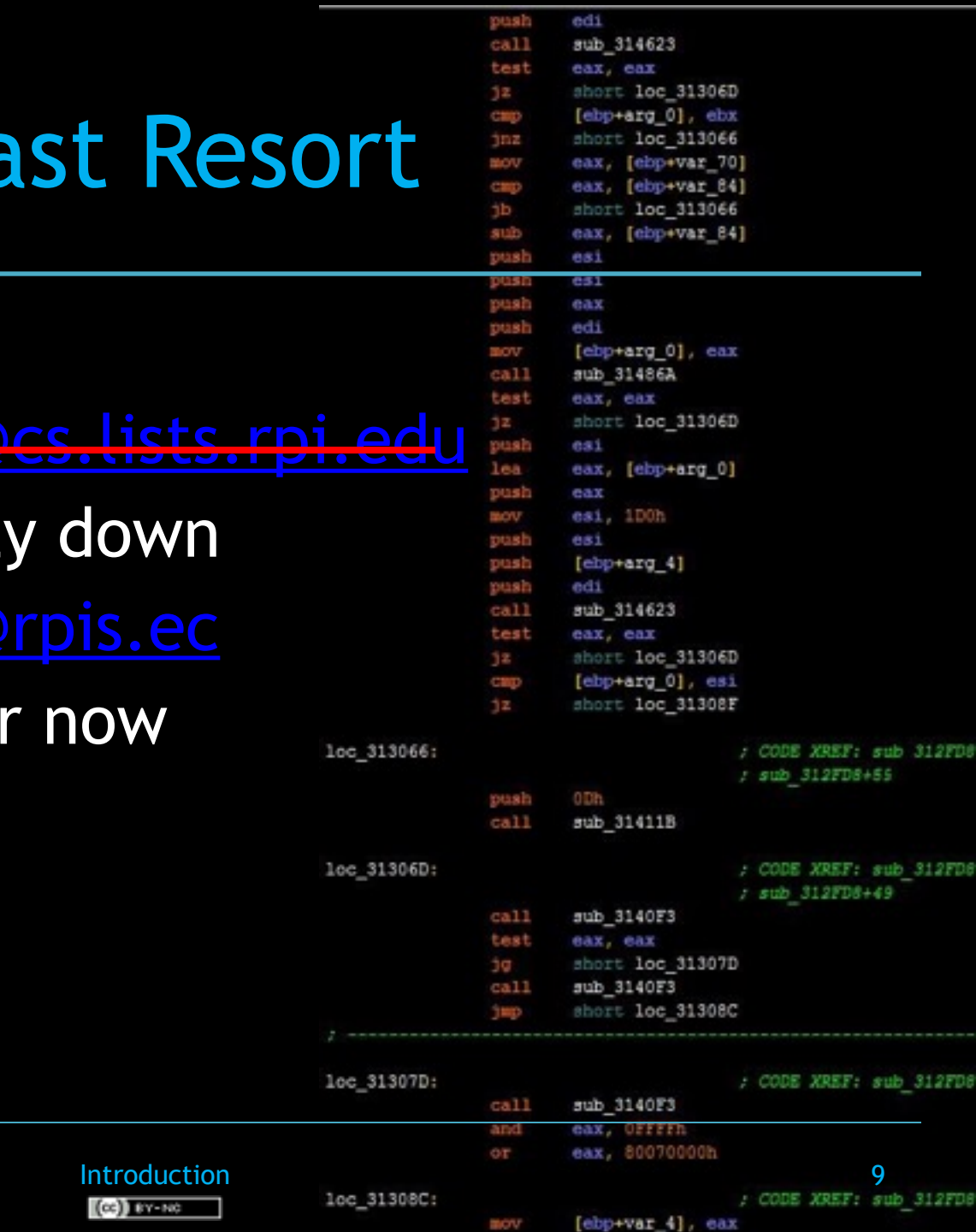
Digital Office Hours (IRC)

- The RPISEC IRC - http://rpis.ec/irc
  - server: irc.rpis.ec
  - port: 6667 (6697 for SSL)
  - room: #rpisec

- Way faster than emailing back and forth
- Some of us are usually on at ridiculous hours
  - basically a 24/7 channel

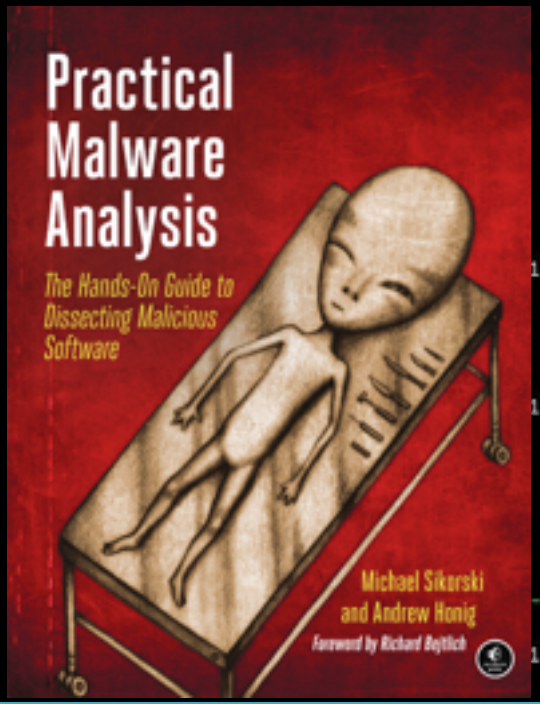
Options of Last Resort

- Email us
  - malware_ta@cs.lists.rpi.edu
    - temporarily down
  - malware_ta@rpis.ec
    - use this for now



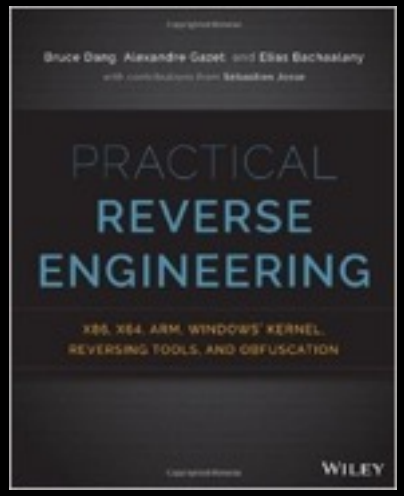
Required Textbooks

- Practical Malware Analysis by Michael Sikorski and Andrew Honig
  - ISBN 978-1593272906



Suggested Textbooks

- Practical Reverse Engineering by Dang, Gazet, Bachaalany
  - ISBN 978-1118787311
- Rootkits: Subverting the Windows Kernel by Hoglund, Butler
  - ISBN 978-0321294319

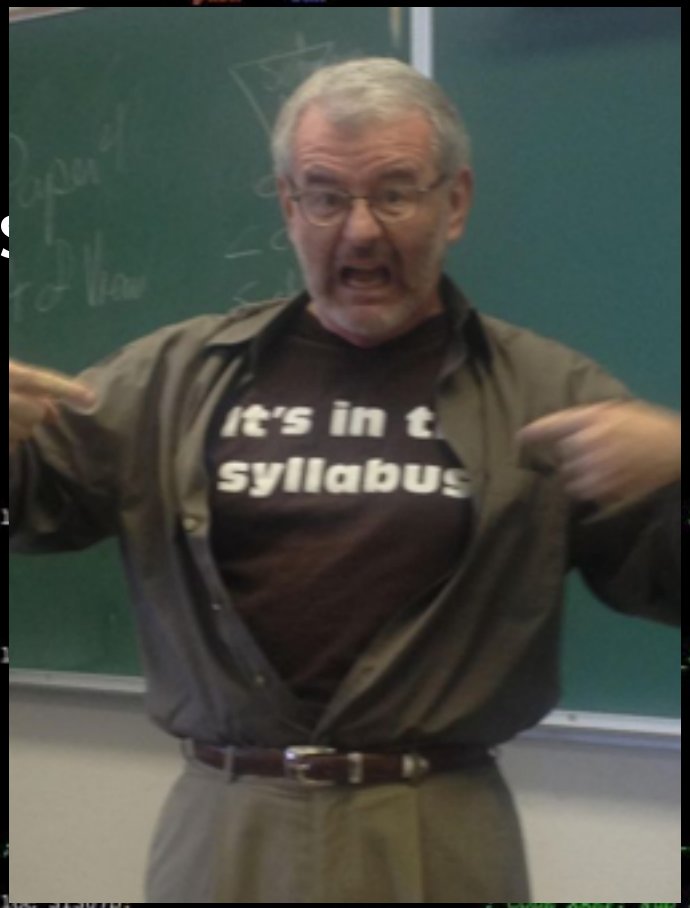


Grade Breakdown

- Labs - 48%
  - 12 labs @ 4% each
  - Lab attendance is MANDATORY as the first part is due and must be checked off in person
- Malware Analysis - 42%
  - 3 Projects @ 10% each
  - Final Project @ 12%
  - Like a big lab, but over a few weeks
- Quizzes - 10%
  - 10 quizzes @ 1% each
  - Small, quick, easy, from the reading

Syllabus

- READ THE SYLLABUS
- Well written, full of details
- It's on the course website rpis.ec/malware



Lecture Overview

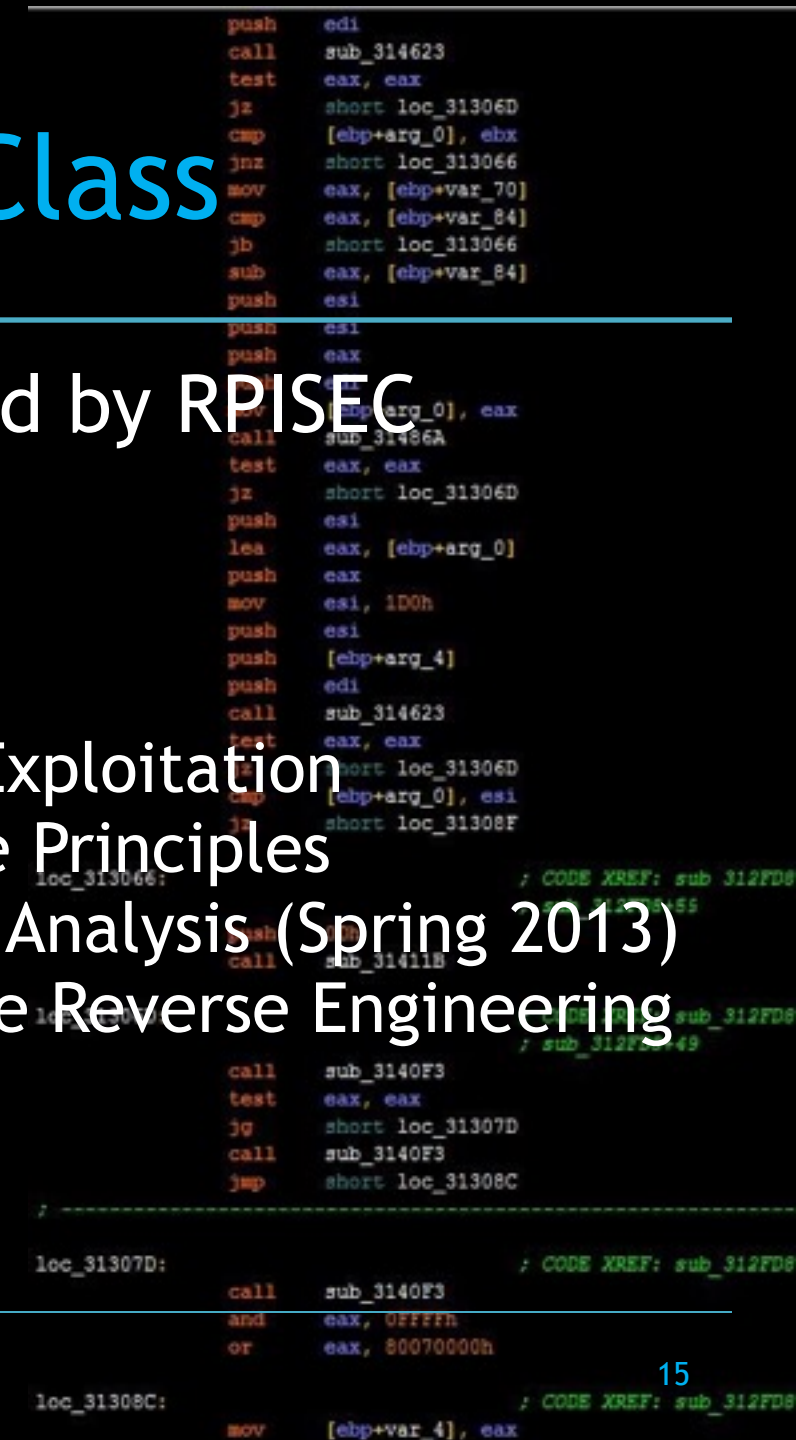
1. Syllabus
2. Course Overview
3. Basic Analysis

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```

A typical (RPISEC) Class

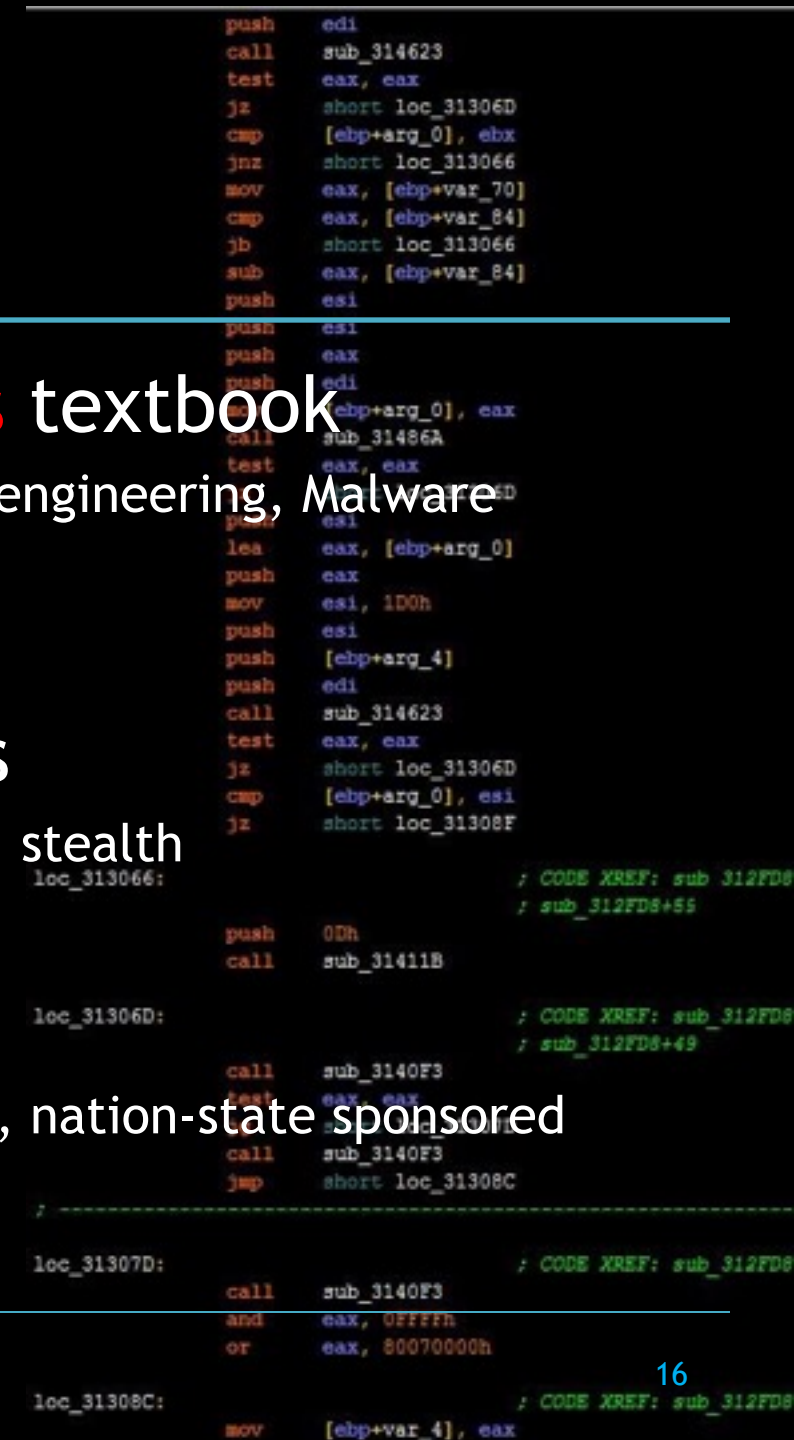
- Designed and orchestrated by RPISEC (students)

- Other courses
  - CSCI 4968 Modern Binary Exploitation
  - CSCI 4971 Secure Software Principles
  - CSCI 4972 / 6963 Malware Analysis (Spring 2013)
  - CSCI 4974 / 6974 Hardware Reverse Engineering



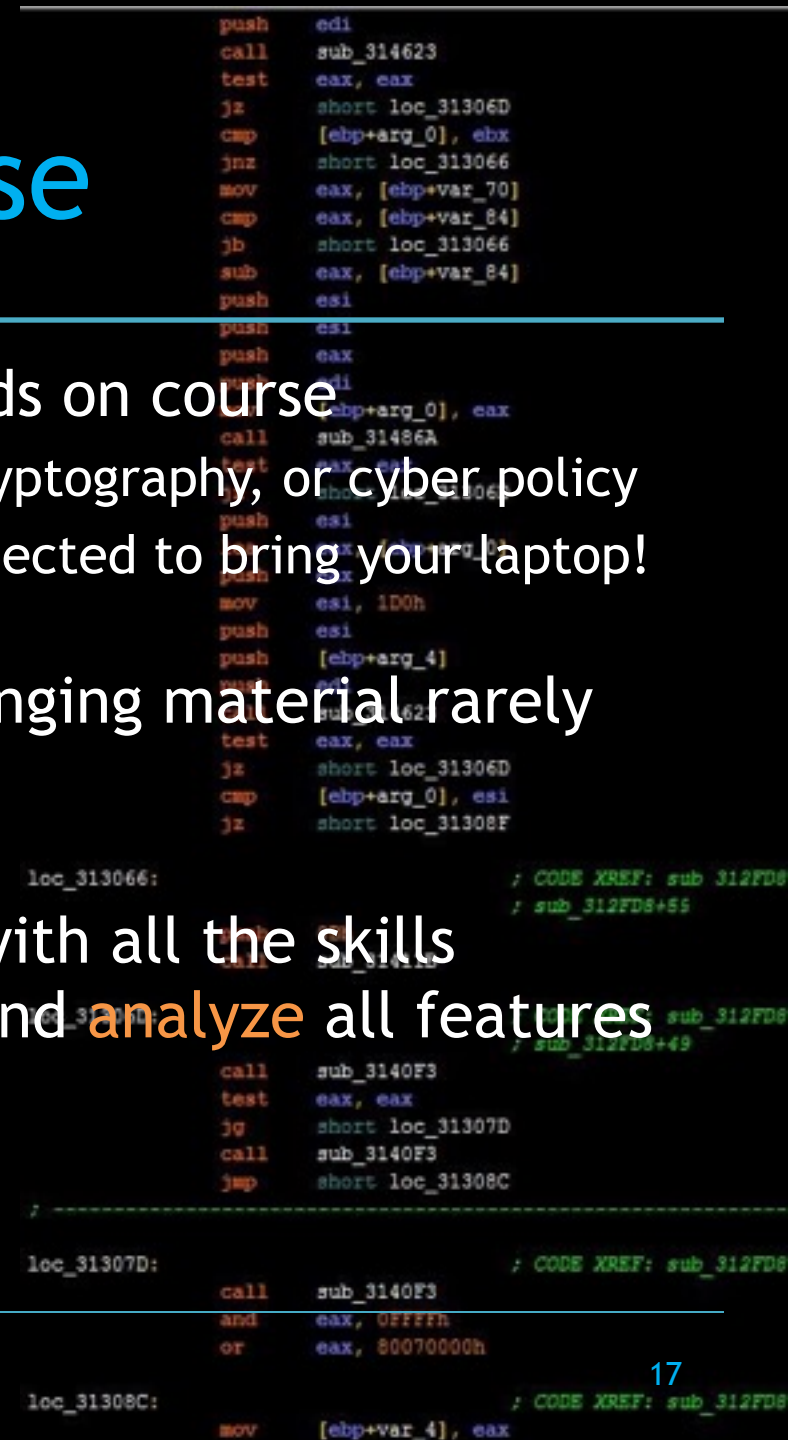
Course Roadmap

- Practical Malware Analysis textbook
  - Basic analysis, debugging, reverse engineering, Malware behavior, Windows internals
- Windows Kernel + Rootkits
  - kernel basics, debugging, behavior, stealth
- Modern malware threats
  - APTs (Advanced Persistent threats), nation-state sponsored



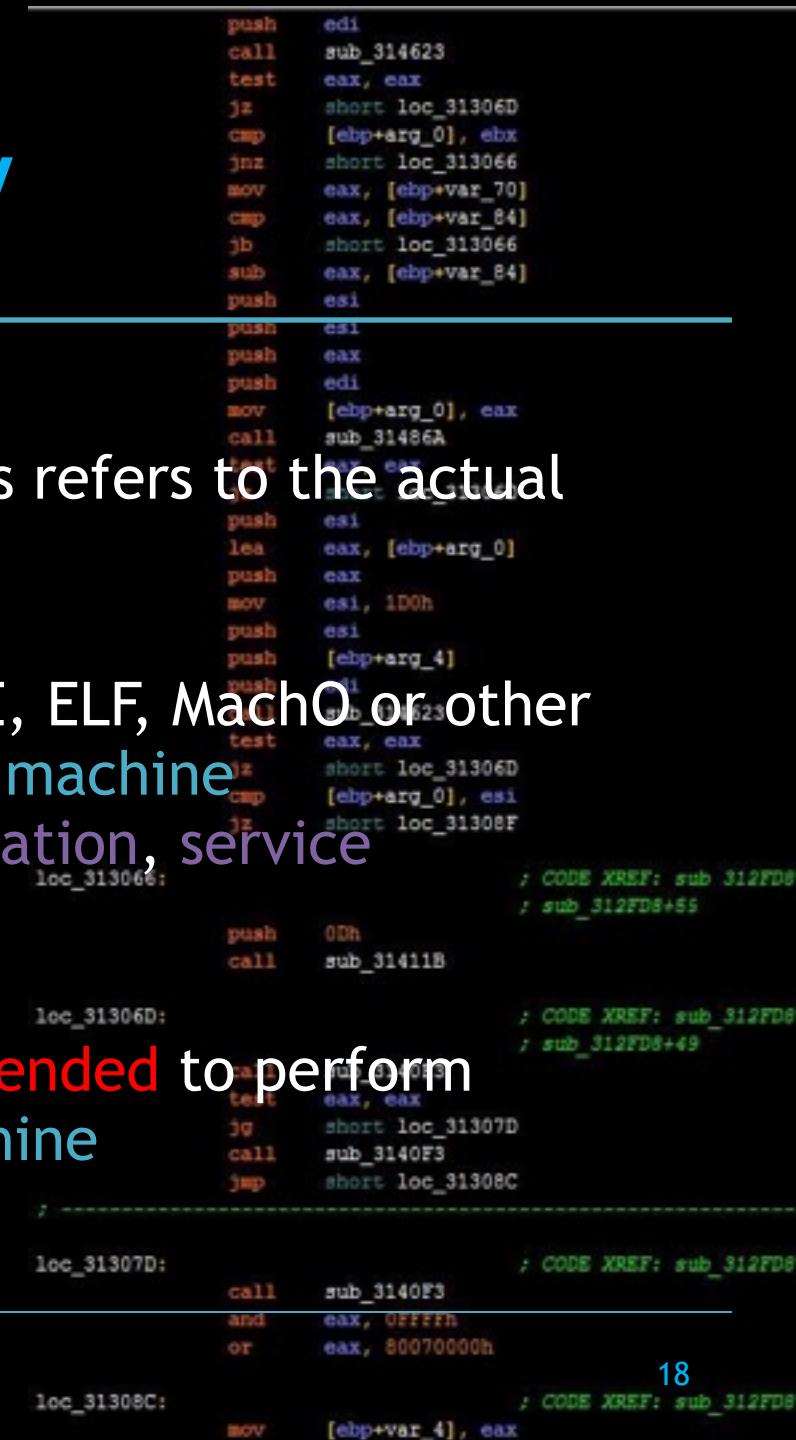
Goals for This Course

- This will be a very applied, hands on course
  - No data structures, algorithms, cryptography, or cyber policy
  - Every lecture after this you're expected to bring your laptop!
- We will cover technically challenging material rarely touched upon in other classes
- As an individual you will leave with all the skills necessary to identify, extract, and analyze all features of modern malicious software.



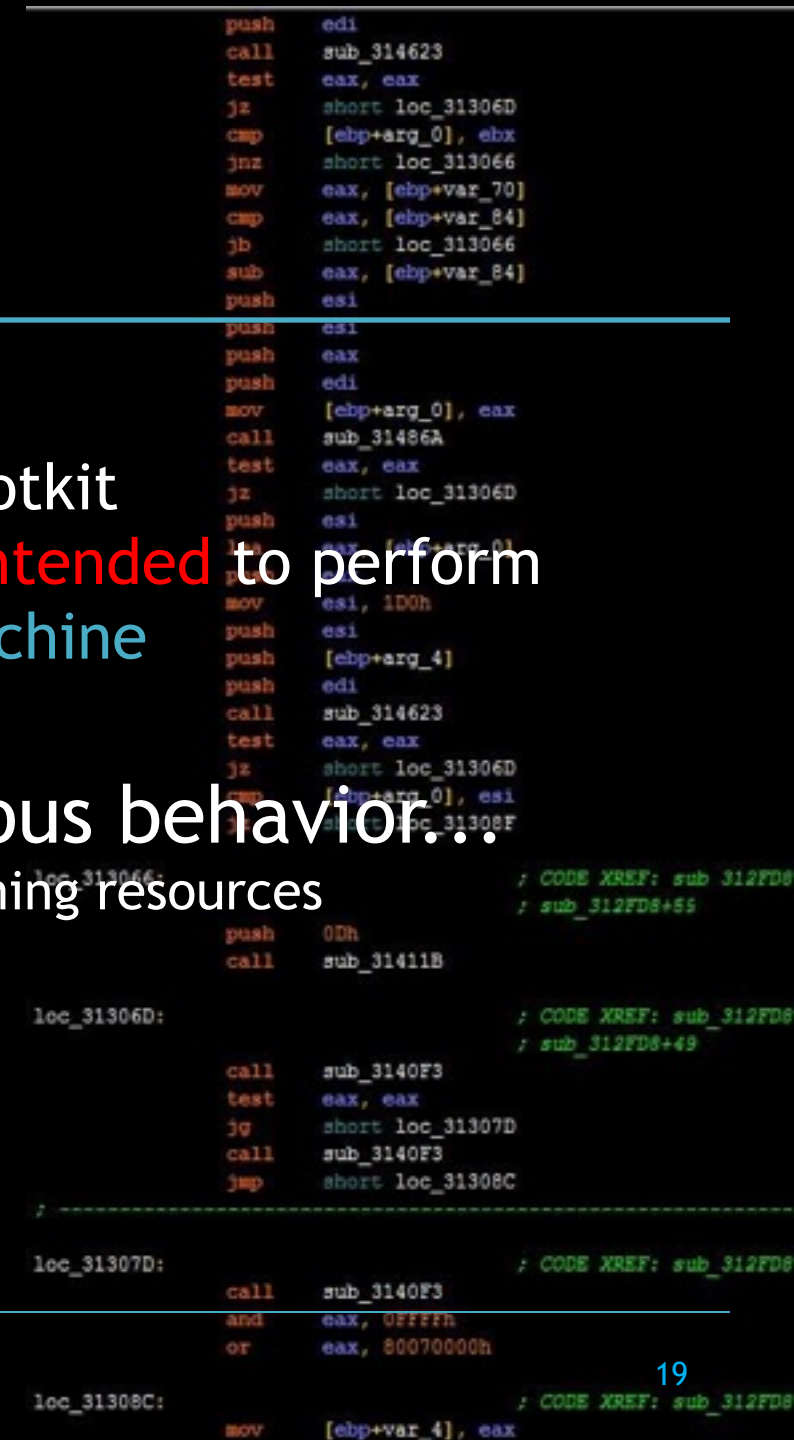
Course Terminology

- Machine
  - A computer, server, sometimes refers to the actual CPU
- Binary
  - An executable such as an .EXE, ELF, MachO or other code containers that run on a machine
  - Other names: program, application, service (sometimes)
- Malware
  - A piece of software that is intended to perform unwanted activities on a machine
- More as we go along!



What is malware?

- Some common names…
  - Trojan, virus, worm, RAT, rootkit
  - A piece of software that is intended to perform unwanted activities on a machine
- Some examples of malicious behavior…
  - Serving ads, stealing data, consuming resources
  - Others?



Why do people write malware?

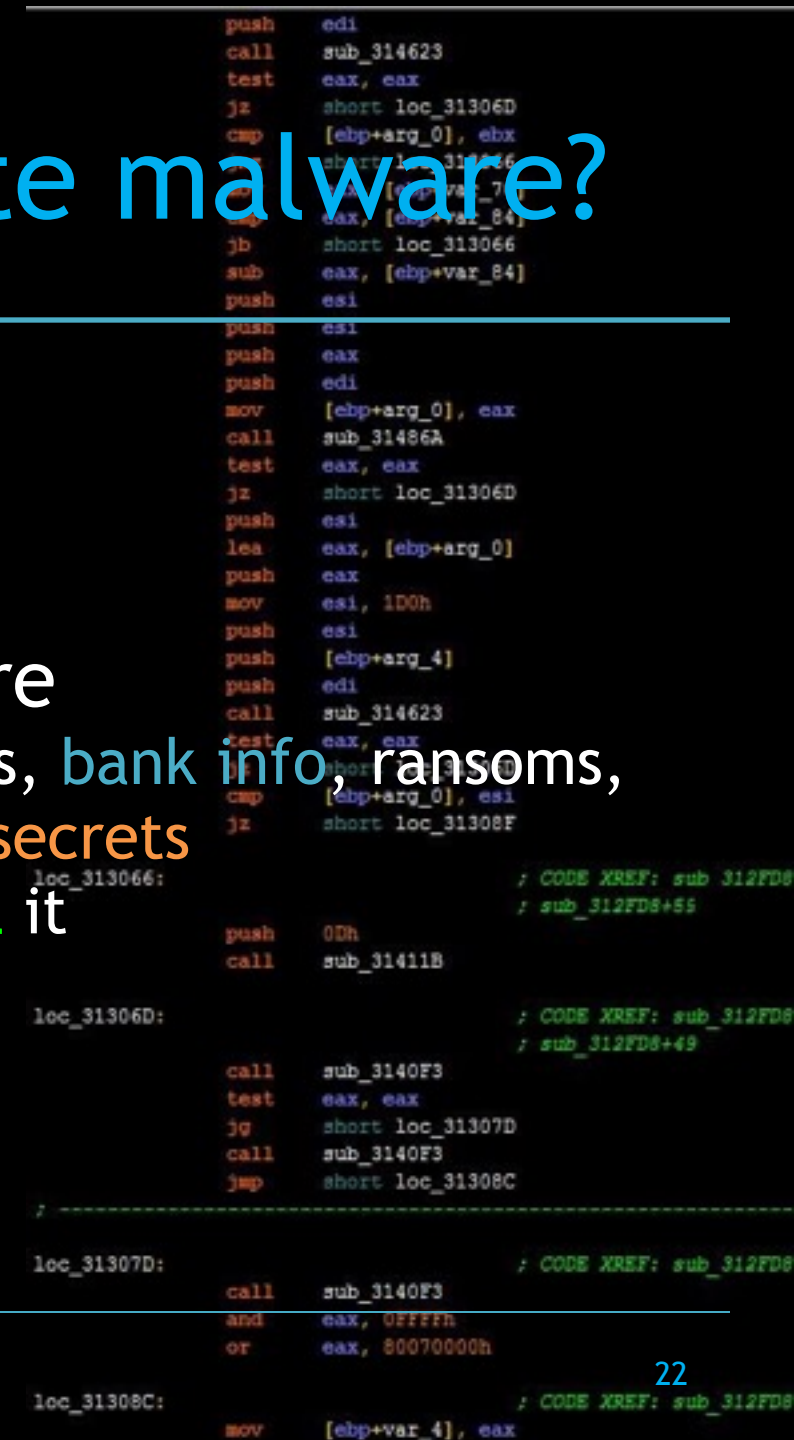
- Morris Worm
  - On accident
    - Purpose: “gauge the size of the internet”
    - What happened: Fork bomb

Why do people write malware?

- In the 90s
  - For the lulz / glory
  - Spread to other machines & display a message

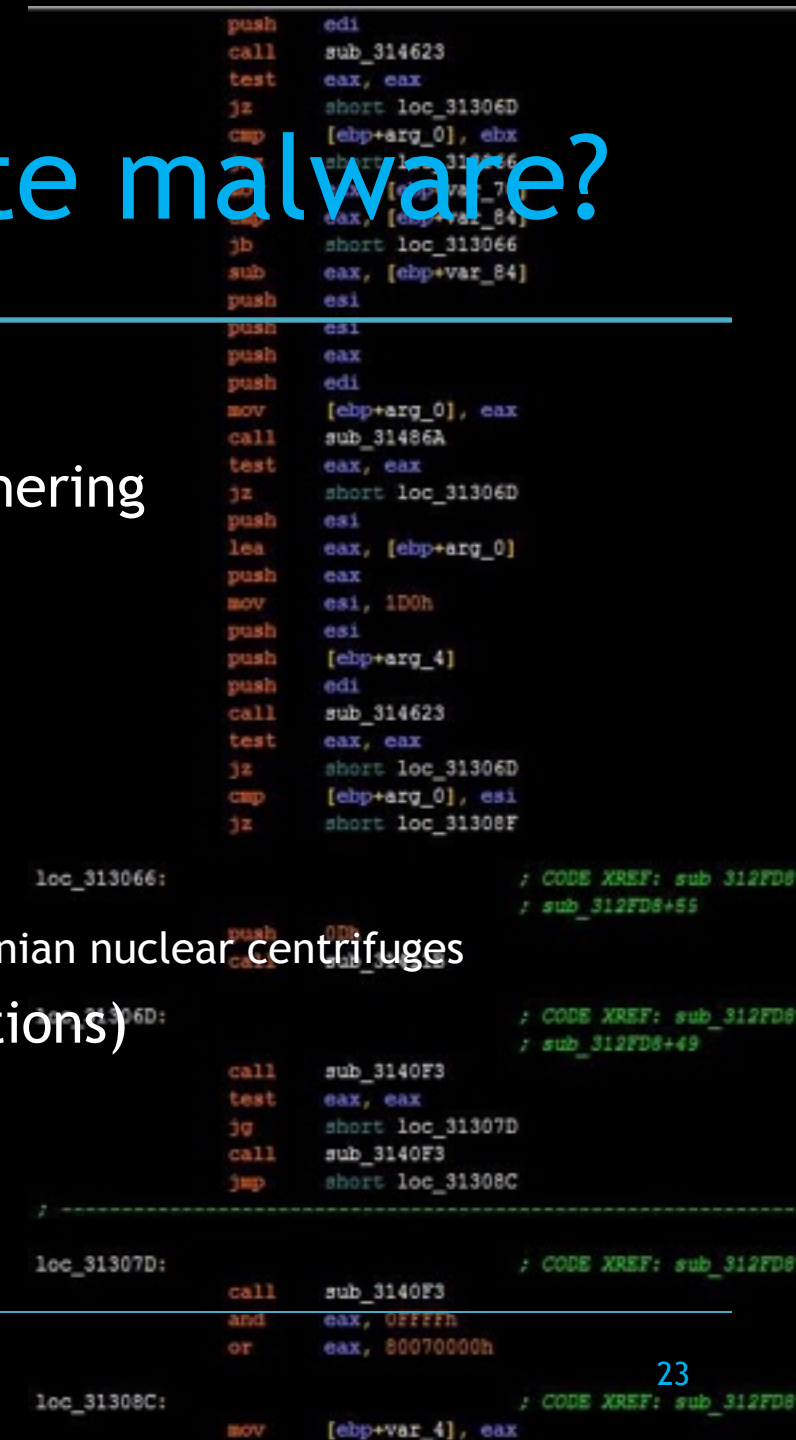
Why do people write malware?

- Today
  - $$$
- Organizations buy malware
  - Steal passwords, credit cards, bank info, ransoms, intellectual property, trade secrets
  - They can use this info or sell it



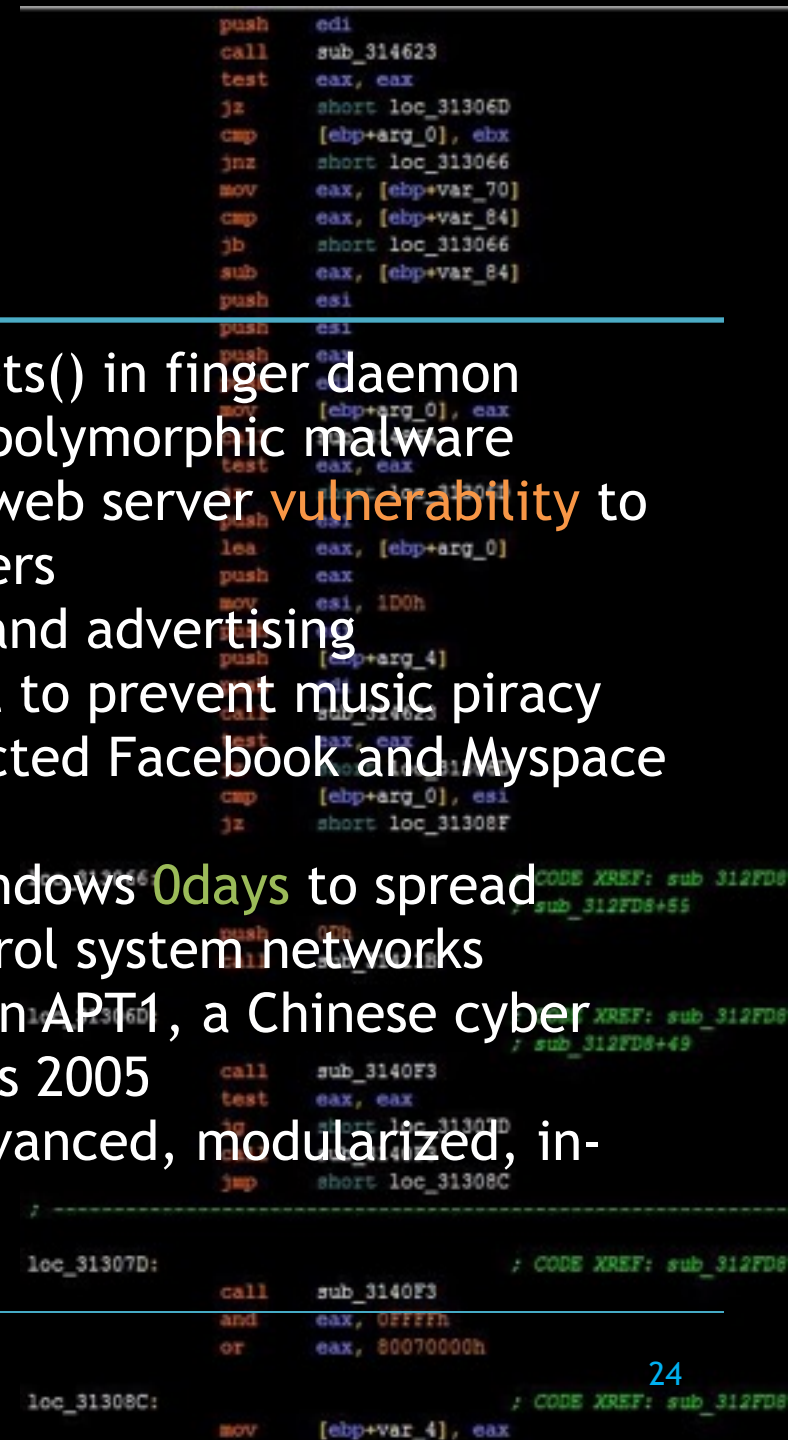
Why do people write malware?

- Future?
  - Cyber warfare, intelligence gathering
- Nation-states
  - Stuxnet
    - Highly advanced
    - Multiple Windows 0-days
    - Targeted and physically destroyed Iranian nuclear centrifuges
  - CNO (Computer Network Operations)
    - CND - Defense
    - CNE - Exploitation
    - CNA - Attack



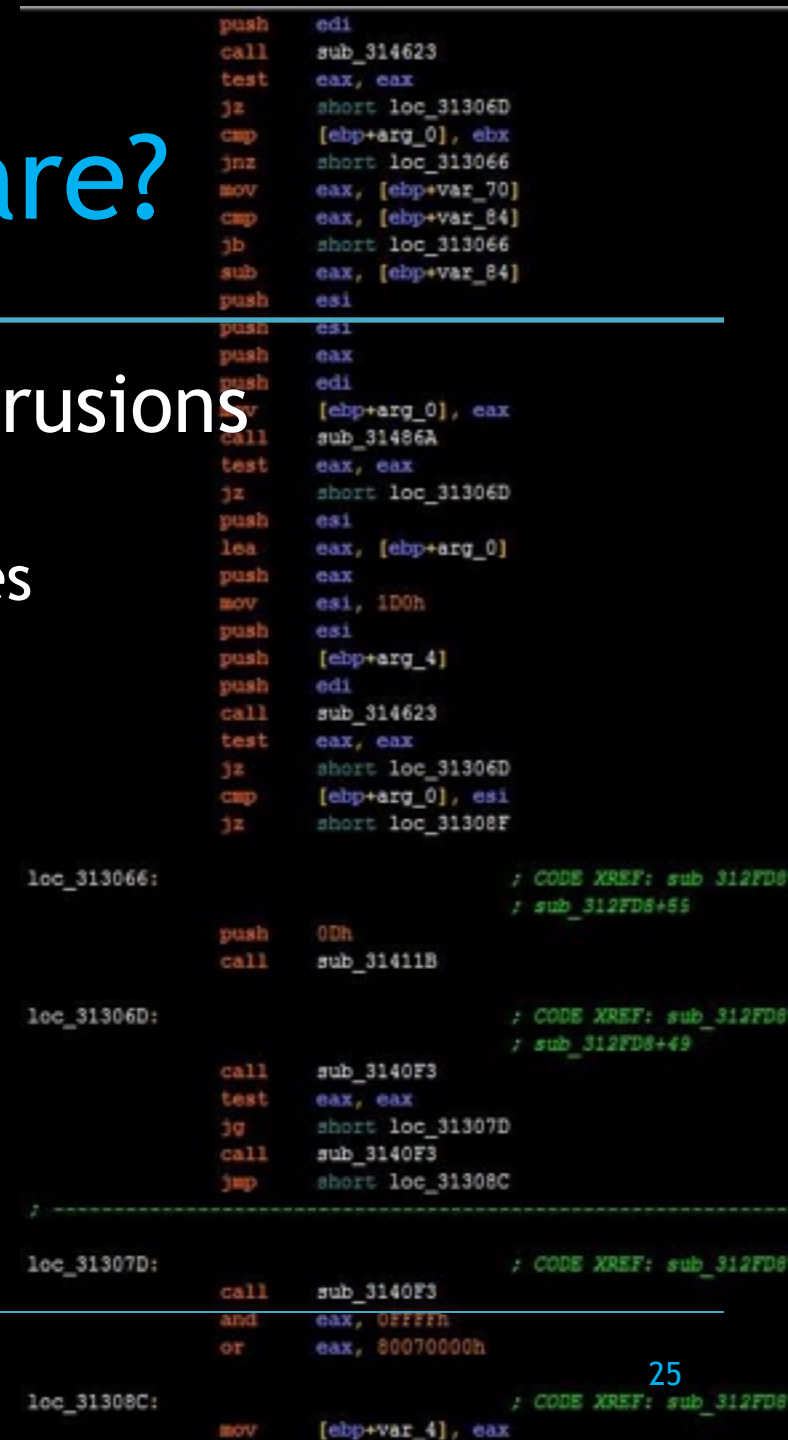
Malware over time

- 1988 - Morris Worm exploits use of gets() in finger daemon
- 1990 - Mark Washburn develops first polymorphic malware
- 2001 - Code Red worm exploits a MS web server vulnerability to hit hundreds of thousands of computers
- 2004 - Vundo trojan displays popups and advertising
- 2005 - Sony infects CDs with a rootkit to prevent music piracy
- 2008 - Koobface RAT spreads via infected Facebook and Myspace profiles
- 2008-2010 - Stuxnet employs four Windows 0days to spread through Iranian nuclear refinery control system networks
- 2013 - Mandiant publishes evidence on APT1, a Chinese cyber espionage campaign dating as early as 2005
- 2015 - Duqu2 targets McAfee with advanced, modularized, in-memory only malware



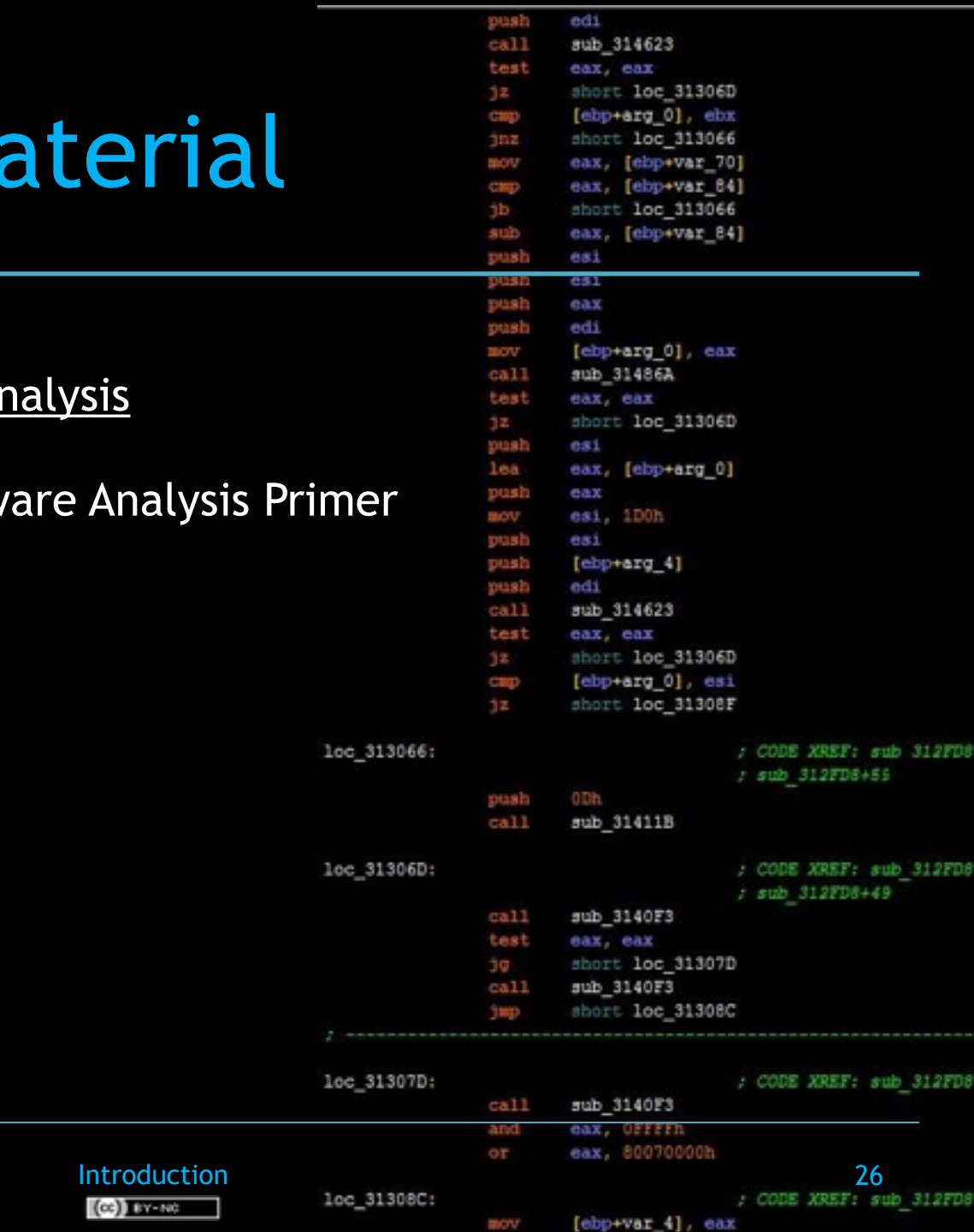
Why analyze malware?

- Detect and respond to intrusions
  - Threat analysis
    - Host & Network signatures
    - What's the damage?
      - Who/What is infected?
  - Threat prevention
  - Threat removal



Additional Material

- Related Readings:
  - Practical Malware Analysis
    - Introduction
    - Chapter 0. Malware Analysis Primer



Lecture Overview

- Syllabus
- Course Overview
- Basic Analysis

```
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], ebx
                jnz     short loc_313066
                mov     eax, [ebp+var_70]
                cmp     eax, [ebp+var_84]
                jb      short loc_313066
                sub     eax, [ebp+var_84]
                push    esi
                push    esi
                push    eax
                push    edi
                mov     [ebp+arg_0], eax
                call    sub_31486A
                test    eax, eax
                jz      short loc_31306D
                push    esi
                lea     eax, [ebp+arg_0]
                push    eax
                mov     esi, 1D0h
                push    esi
                push    [ebp+arg_4]
                push    edi
                call    sub_314623
                test    eax, eax
                jz      short loc_31306D
                cmp     [ebp+arg_0], esi
                jz      short loc_31308F

loc_313066:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+55
                push    0Dh
                call    sub_31411B

loc_31306D:                             ; CODE XREF: sub_312FD8
                                        ; sub_312FD8+49
                call    sub_3140F3
                test    eax, eax
                jg      short loc_31307D
                call    sub_3140F3
                jmp     short loc_31308C
; ---------------------------------------------------------------------------

loc_31307D:                             ; CODE XREF: sub_312FD8
                call    sub_3140F3
                and     eax, 0FFFFh
                or      eax, 80070000h

loc_31308C:                             ; CODE XREF: sub_312FD8
                mov     [ebp+var_4], eax
```