# Computer Virology

1. **Course number and name:** 020VIRES5 Computer Virology

2. **Credits and contact hours:** 4 ECTS credits, 35 contact hours (lectures + labs)

3. **Instructor's or course coordinator's name:** Maroun Chamoun

4. **Text book**:
   a. **Other supplemental materials**:
      Handouts posted on the Web.

5. **Specific course information**
   a. **Catalog description**:
      Introduction: The taxonomy of malware and its capabilities, History of malware - Reverse engineering: tools, obfuscation, packers, anti-debug techniques, x86 and x64 Assembly, Binary Code Analysis – Buffer overflows: Memory Corruption Bugs, Stack Overflow, Format String Attack, Integer Overflow, Fuzzing, Exploitation and Mitigation Techniques, Protection Mechanisms - The theory of malware: Turing Machine, The Halting Problem and Decidability, Adleman's proof of the undecidability of the presence of a virus, Cohen's experiments on detectability and self-obfuscation – Self-reproducing Malware: script and macro-virus, executable file virus, system virus and rootkit, Antivirus: Antivirus techniques, Antivirus Relay, Protection techniques, Antivirus Benchmarking and Testing – SPAM: Common techniques of SPAM and SPAM filtering.

   b. **Prerequisites:**

   c. **Required**: Elective for CCE students; required for CCE software engineering option students

6. **Specific goals for the course**
   A critical element of a complete education for the graduating professional computer scientists must include knowledge about viruses, their nature, and their destruction.

   a. **Specific outcomes of instruction**
      - Prepare the newest computer professionals with the expertise needed to work in a computing environment which includes computer viruses and other forms of malware.
      - Have specific reversing skills in the deconstruction of various x86 assembler obfuscation tricks used by malware in order to be an expert malware reverser.
      - Identify and describe major programming errors and ways to mitigate the impact of discovered vulnerabilities.
      - Identify key characteristics of malware and ways to mitigate the threat of malware.

**b. KPI addressed by the course:**

| KPI | a1 | a2 | g1 | k2 | k3 |
|---|---|---|---|---|---|
| Covered | x | x | | x | x |
| Assessed | | x | x | x | |

**7. Topics and approximate lecture hours**:

- Introduction: The taxonomy of malware and its capabilities: viruses, Trojan horses, rootkits, backdoors, worms, targeted malware; History of malware (2 lectures)
- Reverse engineering: Hex Editors, Disassemblers, Debuggers, obfuscation, packers, anti-debug techniques, x86 and x64 Assembly, memory organization, Binary Code Analysis (3 lectures)
- Lab: Reverse engineering tools and cracking software (1:15 lab hours)
- Homework1: De-obfuscation of an obfuscated perl code (6 hours of mini-project)
- Homework2: Cracking a serial number (8 hours of mini-project)
- Buffer overflows: Memory Corruption Bugs, Stack Overflow, Heap Overflow, Format String Attack, Integer Overflow, Fuzzing, Exploitation and Mitigation Techniques, Protection Mechanisms (3 lectures)
- Lab: Format string attack (2:30 lab hours)
- Homework3: Fuzzing and exploiting a vulnerable server (10 hours of mini-project)
- The theory of malware: Turing Machine, The Halting Problem and Decidability, Adleman's proof of the undecidability of the presence of a virus, Cohen's experiments on detectability and self-obfuscation (2 lectures)
- Script and macro-virus: VBA and Microsoft office virus, propagate macro-virus by infecting Normal.dot template, macro-worms, VBscript and Javascript worms, shell script virus (2 lectures)
- Lab: Shell script virus on Linux (1:15 lab hours)
- Executable file virus: PE and ELF executable file format, Adding Viral Code: Appenders and Prependers, Code Interlacing Infection, Companion Viruses, Virus algorithm (3 lectures)
- Lab: Writing Linux executable virus using C and Assembly language (3:45 lab hours)
- System Virus and Rootkit: Computer bootstrapping, File system structure, Boot structure viruses, Windows Kernel architecture, Behavioral Viruses, Anti-Antiviral Techniques, User-mode and Kernel-mode Rootkit (3 lectures)
- Lab: Writing a Linux Kernel-mode Rootkit (1:15 lab hours)
- Anti-virus: Protecting Against Viral Infections, Antiviral Techniques: Scanning; Spectral analysis; Heuristic analysis; File integrity checking; Behavior Monitoring; Code emulation, Antivirus Relay, Assessing of the Cost of Viral Attacks, Computer "Hygiene Rules", What To Do in Case of a Malware Attack (3 lectures)
- Lab: Implementing an Anti-virus Relay (1:15 lab hours)
- SPAM:. opting -in and –out, Spammers' Strategies: Consolidation; Outsourcing; Affiliation based models, Technical countermeasures: Closing the "open relays"; Blacklists; Whitelists; Filtering systems; Teergrubing; Greylisting; Turing Test (2 lectures)