Linux Debug Training

Presenters: John O'Sullivan and Manas Marawaha

Document source: https://github.com/SpecialistLinuxTraining/linux-debug-training/tree/main/Slides

Part-1 Demonstrations: https://www.udemy.com/course/linux-debug-training-part-1/?referralCode=6534D858AEF9555AB23F

Part-2 Demonstrations: https://www.udemy.com/course/linux-debug-training-part-2/?referralCode=1FAF95060F47194A895F

Presenters Introduction

John O'Sullivan

John O'Sullivan is a software architect with over 30 years of experience, specializing in the development and optimization of Linux-based embedded systems. He has extensive hands-on expertise in hardware-software integration and product development across a wide range of industries.

Email LinkedIn

Manas Marawaha

Manas Marawaha is a Principal Engineer with over 13 years of experience in embedded Linux software development. With a strong understanding of Linux internals, kernel programming, and device drivers, he brings extensive hands-on expertise in product development across audio, video, and networking domains.

Email LinkedIn

License Information

You are free to:

- **Share** - copy and redistribute the material in any medium or format for any purpose, even commercially.
- **Adapt** - remix, transform, and build upon the material for any purpose, even commercially.
- The licensor cannot revoke these freedoms as long as you follow the license terms.

License Information (Cont.)

Under the following terms:

- **Attribution** - You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
- **ShareAlike** - If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
- **No additional restrictions** - You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Course information

- Example code demonstrated in the course is available on the GitHub repository.
- For document source, please refer to this link.
- Demonstrations in this part of the course use Ubuntu (OS 22.04), Arch Linux (2025.08.01) and a Buildroot based installation on the Raspberry-Pi.
- Please review the instructions for building the course examples before proceeding.
- We value your feedback and suggestions! Please share them via email at specialistlinuxtraining@gmail.com.
- For corrections on the training material, feel free to raise issues and pull request in the GitHub repository.

Course Outline

Linux Debug Training (Part-1)

Module 1: Linux Operating System Architecture

Module 2: Basic Linux Analysis and Observability Tools

Module 3: Application Debugging

Module 4: Memory Issues in Linux Applications

Linux Debug Training (Part-2)

Module 5: Tracing in Linux

Module 6: Profiling in Linux

Module 7: Linux Kernel Debugging

Part-1 Synopsis

- Module 1 began with the Linux OS system itself, discussing the fundamental architecture and structure of the Linux operating system.
- In the second module of this course, we explored the Basic Linux Analysis and Observability Tools. We demonstrated how these foundational tools served as the first level of debugging and how they provided insights into the system's state and behavior.
- The third module of this course looked at how we could dissect application binaries with tools like binutils and how we could employ powerful debugging applications like GDB for more in-depth analysis.
- The fourth module was dedicated to addressing common memory issues in user-space applications, along with exploring related tools such as valgrind and sanitizers which are designed to detect and resolve these issues before the code is deployed.

Part-2 Synopsis

- Module five takes a look at tracing in Linux. We cover userspace tools like strace, ltrace, uprobe and perf. We also look at kernel space tools like Kprobe, Perf, ftrace, eBPF, and LTTng.
- In the sixth module of this course, we will explore profiling in Linux looking at tools like massif, heaptrack and memusage to profile memory. We will investigate the use of callgrind, cachegrind and perf-stat for CPU and hardware profiling. We will also look at stacktrace profiling using eBPF profile, gprof and sysprof. And, we will look at how data visualization tools can complement our analysis.
- In the seventh and final module we take a comprehensive look at Kernel debugging, investigating Kernel OPPS, reviewing logging and SysRq, and using tools like KGDB. We will show Kernel recovery using Kexec and Kdump. And we will also examine many of the new tools that have emerged in recent years like: UBSAN, KCSAN and KASAN.