

# Practical Linux Performance and Application Troubleshooting

by Tanel Poder | <https://blog.tanelpoder.com/seminar>

## Training overview

The emphasis of this class is on *practical application troubleshooting* and Linux OS performance tuning techniques. We'll be using scripts and tools that you can use on any Linux machine whenever a problem happens, without having to first install a complex layer of monitoring tools.

At the same time, we will go way beyond what the classic sysadmin tools (like `vmstat`, `iostat`, `top`) have to offer. We will use a process/thread-level sampling approach instead of system wide averages and drill down into application activity from there.

There is "application troubleshooting" in this class title - in addition to GNU/Linux OS, we are going to spend lots of time troubleshooting Linux-based applications (Java apps, database engines, web servers etc.) with OS tools. This is useful for augmenting the application's built-in instrumentation and logging, or in cases where the app doesn't have decent logging enabled at all.

While we won't spend time on non-practical tasks like compiling Linux kernels, we will regularly examine kernel activity using Linux built-in diagnostic & tracing facilities like *strace*, *perf*, *ftrace*, *blktrace* and the latest *eBPF* tracing tools. We won't write Linux kernel source code but will learn how to find and read the relevant source files of interest to aid practical troubleshooting and avoid guesswork.

My favorite starting point is to examine the process state from `/proc` filesystem directly if the traditional tools are not enough. We will use this technique throughout the class. Here's a blog entry of mine from past that illustrates how useful this simple and non-intrusive technique can be:

- <https://blog.tanelpoder.com/2013/02/21/peeking-into-linux-kernel-land>

In addition to application troubleshooting, we will cover all key areas of kernel/OS interaction: disk IO, network IO, memory allocation, CPU usage & scheduling, filesystems & buffering, spinlocks, IPC and process crashes/kernel panics too.

## Training Duration and Details

|              |   |
|--------------|---|
| Duration:    | <b>5 days</b> (10 x 4h online training sessions spread over 2 weeks)  |
| Format:      | Online webinar (8am-12pm Pacific Time on each day)  |
| Audience:    | Application admins, Linux sysadmins, DevOps folks, DBAs, system programmers, full-stack engineers & senior developers |
| Skill level: | Intermediate to Advanced  |

## Learning Objectives

- Navigate around in Linux process state & performance metrics like a boss
- Learn the systematic process-level troubleshooting approach all the way to OS kernel
- Familiarize yourself with scripts and tools that can quickly & safely be used in production
- Tune the operating system settings to improve your application performance
- Find application bottlenecks, inefficiencies and suboptimal patterns of OS API usage

## Non-Objectives

- This is not a Linux *administration* class
- This is not a typical OS-only tuning & configuration class aimed just for sysadmins
- While we look into measuring Java, C/C++, python apps, this is not a developer coding class
- While we look into database engines at OS level, this is not a RDBMS or SQL tuning class
- While we troubleshoot OS-level network activity, this is not a network administration class

## Example troubleshooting scenarios in this class

- Why is everything slow in this machine?
- Where is my process/application spending its time or is hanging?
- Which PATH & configuration files is my application really using?
- Where are all the logfiles this application writes to?
- Why does my program fail with an OS error on startup?

## Example questions answered in this class

- What does the Linux system load really measure?
- Why is the system load high, while CPU activity is low?
- How to know how much free memory is really left?
- How to know how much physical RAM this application is using?
- How to break down I/O latency by Linux OS layer?

## Recommended Prerequisites

- Basic understanding of OS concepts: processes, threads, files & file systems, devices
- Know how to use Unix/Linux command line tools (from basics to *ps*, *top*, *df*, etc.)
- Ability to read shell scripts (and optionally Python, C)

## Is This Training for Me?

- As with all my classes, if you like my blog entries, you'll like this training too:
- <https://blog.tanelpoder.com/categories/linux/>

## Table of Contents

- Download the Table of Contents and get more info here:
- <https://blog.tanelpoder.com/seminar/>

## Further Questions:

- [seminars@poderc.com](mailto:seminars@poderc.com)