Linux 故障排查工具速查手册

> **核心思路**：先定位是哪一类资源出了问题（CPU / 内存 / 磁盘 / 网络），再用对应的专项工具往下钻。

排查流程总览

步骤	目标	推荐工具
1	全景感知（定性）	htop / glances / dstat / vmstat
2	定位到进程	pidstat / iotop / nethogs
3	深入剖析	perf / eBPF / strace / ltrace / gdb
4	内核/底层确认	dmesg / journalctl / numactl

一、追踪剖析类

1. ltrace — 库函数调用追踪

> strace 追内核调用，ltrace 追用户态动态库函数，两者互补。

用途	命令
attach 到进程	`ltrace -p <PID>`
统计库函数调用次数和耗时	`ltrace -c ./your_app`

适用场景：怀疑程序在 malloc、字符串处理等库函数中死循环或性能瓶颈。

⚠️ **注意**：开销与 strace 类似，建议仅在测试环境或可接受短暂影响时使用。

---

2. eBPF 全家桶（bpftrace / bcc-tools）— 生产环境首选

> 基于内核 BPF 虚拟机，性能损耗极低，可在生产高并发环境直接使用。

bcc-tools 常用脚本：

工具	功能	典型场景
`execsnoop`	实时追踪新进程启动	异常启动、定时任务乱跑
`opensnoop`	追踪文件打开操作	频繁读文件、配置未缓存
`ext4slower 10`	揪出超过 10ms 的慢 I/O	磁盘 await 飙高时定位元凶
`biolatency`	磁盘 I/O 延迟直方图	整体 I/O 延迟分布
`tcpconnect`	追踪 TCP 主动连接	网络连接异常

bpftrace 自定义追踪示例：

```
# 统计各进程调用 read 的次数
bpftrace -e 'tracepoint:syscalls:sys_enter_read { @[comm] = count(); }'
```

安装：

```
yum install bcc-tools          # CentOS/RHEL
apt install bpfcc-tools        # Debian/Ubuntu
# 脚本位置：/usr/share/bcc/tools/
```

⚠️ **门槛**：内核版本需 ≥ 4.9。

---

3. perf — CPU 热点分析

用途	命令
实时 CPU 热点函数	`perf top`
采样 30 秒	`perf record -F 99 -p <PID> -g -- sleep 30`
查看采样结果	`perf report`

生成火焰图（强烈推荐）：

```
perf record -F 99 -p <PID> -g -- sleep 30
perf script | ./stackcollapse-perf.pl | ./flamegraph.pl > cpu.svg
# 用浏览器打开 cpu.svg，可交互点击放大
```

> 火焰图中越宽的函数占 CPU 越多，一目了然。

---

4. gdb — 进程卡死/死锁排查

```
gdb -p <PID>
(gdb) bt                       # 当前线程调用栈
(gdb) thread apply all bt      # 所有线程的栈（排查死锁利器）
```

⚠️ **注意**：gdb attach 会短暂暂停进程，生产慎用。

二、系统资源全景

5. htop / glances / dstat — 全局概览

工具	特点
`htop`	彩色、支持鼠标、单核负载可视
`glances`	CPU/内存/磁盘/网络/进程一屏全览
`dstat -tcmndy 1`	多指标横排滚动，观察趋势最佳

---

6. vmstat — 上下文切换和 Swap 的照妖镜

```
vmstat 1
```

重点关注列：

列	含义	异常信号
`cs`	上下文切换次数	异常高 → 线程过多 / 锁竞争
`in`	中断数	飙高 → 网络流量大 / 硬件问题
`si` / `so`	Swap 换入/换出	不为 0 → 内存不足，性能崩盘

---

7. pidstat — 精确到进程/线程

用途	命令
各进程 CPU	`pidstat 1`
各进程磁盘 I/O	`pidstat -d 1`
各进程内存	`pidstat -r 1`
线程级别	`pidstat -t -p <PID> 1`

三、磁盘 I/O 排查

8. iostat — 磁盘繁忙度

```
iostat -x 1
```

指标	含义	说明
`%util`	磁盘繁忙程度	接近 100% → 到极限（SSD/NVMe 需结合 await 判断）
`await`	I/O 平均响应时间	飙高 → 磁盘扛不住

---

9. iotop — 定位刷盘进程

```
iotop -oP
# -o 只显示有 I/O 的进程  -P 按进程显示
```

---

10. lsof — 幽灵文件 & 端口占用

经典场景：`df -h` 满了，`du` 找不到大文件。

```
lsof | grep deleted            # 找已删除但未释放空间的文件
lsof -i :8080                  # 谁占了 8080 端口
lsof -p <PID>                  # 进程打开了哪些文件
```

> 找到 deleted 文件对应的进程，kill 或重启后空间立即释放。

---

11. fuser — 强制解除目录占用

```
fuser -m /data                 # 谁在用这个挂载点
fuser -k /data                 # 直接杀掉占用进程
```

四、网络排查

12. ss — 替代 netstat

> ss 直接读内核，比 netstat 快几个数量级。

用途	命令
TCP 连接状态总览	`ss -s`
各状态连接数统计	`ss -tan | awk '{print $1}' | sort | uniq -c`
查看 TIME_WAIT 连接	`ss -tnp state time-wait`

重点关注：

- `TIME_WAIT` 过多 → 短连接太多，调内核参数优化
- `CLOSE_WAIT` 过多 → **程序 bug**，连接未正常关闭，必须查代码

---

13. tcpdump — 抓包分析

```
tcpdump -i eth0 host 10.0.0.1 and port 8080 -w capture.pcap
# 抓完用 Wireshark 打开分析
```

⚠️ **务必加过滤条件**，否则一秒能抓几个 G。

---

14. nethogs — 进程级带宽监控

```
nethogs eth0                   # 按进程显示实时带宽占用
```

---

15. ip / tc — 网络配置与故障模拟

```
ip a                        # 查看网卡地址（替代 ifconfig）
ip r                        # 查看路由

# 模拟 100ms 延迟 + 5% 丢包
tc qdisc add dev eth0 root netem delay 100ms loss 5%
# 测完务必删除！
tc qdisc del dev eth0 root
```

---

16. curl / nc — 连通性验证

```
curl -v http://xxx              # 查看完整 HTTP 过程
curl -w "@curl-format.txt" http://xxx  # 各阶段耗时
nc -zv 10.0.0.1 8080           # 测端口通不通
```

五、内核与底层

17. dmesg / journalctl -k — OOM 和硬件报错

```
dmesg -T | tail -50                    # -T 显示可读时间
journalctl -k --since "10 min ago"
```

常见发现：

- `Out of memory: Killed process xxx` → 服务被 OOM Killer 杀掉
- 网卡 Link Down、磁盘变只读、硬件报错

> 服务莫名消失，第一件事看 dmesg！

---

18. numactl / numastat — NUMA 架构性能问题

```
numastat              # 各 NUMA 节点内存分配
numactl --hardware    # NUMA 拓扑
```

适用场景：多路服务器上 CPU 不忙但性能上不去 → 可能是跨 NUMA 节点访问内存。

避坑清单

#	要点
1	❌ 别在高并发生产环境无脑上 strace/ltrace → ✅ 优先用 eBPF
2	❌ 别只看 CPU 使用率 → ✅ vmstat 看上下文切换和 swap
3	❌ 磁盘满了别 du 半天 → ✅ `lsof | grep deleted`
4	❌ `%util` 100% 别急下结论 → ✅ SSD/NVMe 结合 await 判断
5	❌ CLOSE_WAIT 堆积别调内核参数 → ✅ 一定是程序 bug，查代码
6	❌ tcpdump 不加过滤 → ✅ 务必指定 host/port 条件
7	❌ tc 模拟故障忘删规则 → ✅ 测完立即 `tc qdisc del`
8	❌ 服务被杀到处翻日志 → ✅ 先看 dmesg，八成是 OOM

按场景速查

故障现象	排查路径
CPU 飙高	`htop` → `pidstat` → `perf top` → 火焰图
内存不足 / 服务被杀	`vmstat`(看 si/so) → `dmesg`(OOM) → `pidstat -r`
磁盘 I/O 慢	`iostat -x` → `iotop -oP` → `ext4slower`(eBPF)
磁盘满但找不到文件	`df -h` → `lsof | grep deleted`
网络连接异常	`ss -s` → `tcpdump` → `curl -v`
带宽被打满	`nethogs eth0` → `tcpdump`
进程卡死不响应	`strace -p`(简单场景) → `gdb -p` → `thread apply all bt`
服务莫名消失	`dmesg -T` → 找 OOM Killer 日志
端口被占用	`ss -tnlp | grep <port>` 或 `lsof -i :<port>`
目录无法卸载	`fuser -m /path` → `fuser -k /path`