

HangTrace

User Guide for Windows 'Not Responding' Diagnostics (CLI)

Applies to: HangTrace v1.5.x on Windows 10/11

Generated: January 09, 2026

HangTrace helps you identify what a hung application is waiting on, who is blocking it, and how to capture support-friendly artifacts (HTML/JSON report, optional ETW micro-trace, and optional dump).

Important: HangTrace is a diagnostic tool. Reports, ETW traces, and dumps may contain sensitive data. Use only on systems you own or have explicit permission to troubleshoot.

Contents

- 1. Overview
- 2. Quick start
- 3. Installation and requirements
- 4. Command-line reference
- 5. Recommended workflows
- 6. Interpreting results (HTML + JSON)
 - 6.1 Ping latency and message pump health
 - 6.2 Wait Chain Traversal (WCT) and blockers
 - 6.3 Stack samples and symbols
 - 6.4 Heuristics scoring (what it means)
 - 6.5 Close-path probe (before/after)
 - 6.6 ETW micro-trace (.etl) review in WPA
- 7. COM/RPC cross-process hang attribution
- 8. Safety, privacy, and best practices
- 9. Troubleshooting
- Appendix A. Common hang patterns
- Appendix B. Artifact checklist for support tickets
- 10. Disclaimer, license, and contact

1. Overview

When a Windows app shows “Not Responding”, it usually means the UI thread is not pumping messages. Windows may mark the window as unresponsive and, in many cases, create a “ghosted” UI to allow the user to move/close it.

HangTrace focuses on three questions:

- Is the UI thread actually unresponsive (measured latency, not guesswork)?
- What is it waiting on (Wait Chain Traversal) and which thread/process is the blocker?
- What evidence can we capture right now (stacks, before/after close probe, ETW micro-trace) to support a fix?

What HangTrace typically detects:

- CPU saturation or tight loops on the UI thread (high CPU, short waits)
- Blocking waits: critical sections, SRW locks, events, mutexes, I/O waits
- Cross-process dependencies: COM/RPC calls, ALPC, service hosts
- Close/shutdown stalls: WM_CLOSE triggers cleanup, flush, or long RPC calls

Outputs you can generate:

- HTML report for humans (shareable with support / ticket systems)
- JSON report for automation (stable schema version string)
- Optional minidump (for deep debugging; treat as sensitive)
- Optional ETW trace (.etl) via WPR micro-trace mode (WPA-friendly markers)

2. Quick start

Run HangTrace from an elevated command prompt if you need deep stacks or protected process access. Start with the smallest capture that answers your question, then escalate only if needed.

Basic report by process name:

```
HangTrace.exe --process notepad.exe --html hangtrace_report.html
```

Target a PID (recommended for precision):

```
HangTrace.exe --pid 1234 --html hangtrace_report.html --json hangtrace_report.json
```

Deep capture + dump (sensitive):

```
HangTrace.exe --pid 1234 --deep --dump hangtrace_dump.dmp --html hangtrace_report.html
```

ETW micro-trace (requires WPR / Windows Performance Toolkit):

```
HangTrace.exe --pid 1234 --etw hangtrace_trace.etl --html hangtrace_report.html
```

Close-path probe (opt-in) before/after:

```
HangTrace.exe --pid 1234 --probe-close --probe-delay 500 --html hangtrace_report.html
```

Self-test:

```
HangTrace.exe --selftest
```

Tip: If multiple processes match a name, use --pick to choose first/newest/oldest, or target by PID.

3. Installation and requirements

Supported OS: Windows 10 and Windows 11 (x64 recommended).

Permissions: Some diagnostics require Administrator privileges.

- Stack capture across processes may need `PROCESS_VM_READ` / `PROCESS_QUERY_INFORMATION` access.
- Minidump writing can fail without elevation, especially for protected processes.
- Listing services for service-host processes may require Service Control Manager access.

Optional components:

- WPR (Windows Performance Recorder) for ETW micro-trace mode. Typically installed with Windows Performance Toolkit (WPT).
- DbgHelp for stack walking and minidumps (present on most systems; can be shipped with your build if needed).

Artifacts and sensitivity: HTML/JSON are usually safe, but dumps/ETL can contain sensitive data.

4. Command-line reference

The exact flags may vary slightly by build. Common flags:

Area	Option	Purpose
Targeting	--pid / --process	Choose a process by PID or image name.
Selection	--pick first newest oldest	Resolve multiple matches when using --process.
Outputs	--html / --json	Write reports (recommended).
Stacks	--deep	Capture additional thread stacks and enrichment.
Dump	--dump	Write a minidump (treat as sensitive).
ETW	--etw	WPR-based micro-trace with markers (if available).
Close probe	--probe-close / --probe-delay	Opt-in WM_CLOSE probe, then capture after-state.
Diagnostics	--verbose / --selftest	Extra console output / validate environment.

Exit codes (recommended):

Code	Meaning
0	Success
2	Invalid arguments
3	Target not found
4	Access denied / insufficient rights
5	Output write failed
6	Selftest failed

5. Recommended workflows

Workflow A - Quick “what is it stuck on?”

Fast answer with minimal disruption. Produces ping latency, WCT chain, and baseline stacks.

```
HangTrace.exe --pid 1234 --html hangtrace_report.html
```

Workflow B - Deep capture for engineering

Adds more stacks and optional dump. Use for hard-to-reproduce hangs or engineering handoff.

```
HangTrace.exe --pid 1234 --deep --html hangtrace_report.html  
HangTrace.exe --pid 1234 --deep --dump hangtrace_dump.dmp --html hangtrace_report.html
```

Workflow C - Hang triggered during close (opt-in)

Use when the hang happens after the user clicks Close. Captures before/after state.

```
HangTrace.exe --pid 1234 --probe-close --probe-delay 500 --html hangtrace_report.html
```

Workflow D - ETW micro-trace for timing and contention

Creates a short ETW trace window around the capture, with WPA-friendly markers. Requires WPR.

```
HangTrace.exe --pid 1234 --etw hangtrace_trace.etl --html hangtrace_report.html
```

6. Interpreting results (HTML + JSON)

The HTML report is designed for quick human review. The JSON report is designed for automation and tooling. Both follow the same logical sections.

6.1 Ping latency and message pump health

HangTrace measures UI responsiveness using a lightweight “ping” (WM_NULL) to the target window thread. High round-trip latency indicates a real message pump stall (not just slow rendering).

- Low latency: UI thread is pumping (problem may be elsewhere).
- Moderate latency: intermittent stalls or heavy work on UI thread.
- High latency/timeouts: classic “Not Responding”. Capture stacks and WCT immediately.

6.2 Wait Chain Traversal (WCT) and blockers

WCT provides a best-effort chain of what the UI thread is waiting on and who is blocking it. It can identify deadlocks and cross-thread/process dependencies.

- If the chain ends in another thread: capture stacks for both UI and blocker threads.
- If the chain points to a PID only (PidOnly/PidOnlyRpcss): it is a strong hint of cross-process waits.
- If WCT fails: you can still rely on stacks + ETW micro-trace for the timing story.

6.3 Stack samples and symbols

Stack samples answer the question “where is the thread right now?”. For best results, ensure symbols are available (your own PDBs, Microsoft symbols, or local cache).

- If your app has PDBs, place them next to the EXE/DLL or configure a symbol path.
- Kernel/system frames are expected; the key is the topmost user-mode frames.
- Empty/partial stacks often indicates permissions or missing modules/symbols.

6.4 Heuristics scoring (what it means)

HangTrace summarizes findings into categories (scores). These are not proof: they are prioritized hints.

Category	What increases the score	Typical follow-up
CPU / busy loop	High CPU, short waits, repeatable stack frames.	Profile CPU or add ETW CPU sampling.
Lock contention	Waits on critical section/SRW; blocker thread identified.	Inspect lock owner stack; reduce lock scope.
I/O stall	Waits in file/network I/O; storage/network clues.	Check disk/network health; add timeout/cancel path.
COM/RPC	WCT shows COM/ALPC, PidOnlyRpcss, server PID hint.	Identify server process/service; capture ETW RPC/COM.
Close/shutdown	Before/after probe shows state shift on WM_CLOSE.	Move cleanup off UI thread; add async close.

6.5 Close-path probe (before/after)

If the hang appears during close, HangTrace can (opt-in) post WM_CLOSE to the main window and capture a second snapshot. The before/after delta is often extremely informative (new waits, COM calls, I/O flushes).

Use a short probe delay (300-800ms) to catch the transition without disturbing the target too much.

6.6 ETW micro-trace (.etl) review in WPA

ETW micro-trace mode uses WPR to create a short trace window around the capture. The trace includes clear markers so you can zoom into the exact time range in WPA.

Open the trace:

- 1) Launch Windows Performance Analyzer (WPA)
- 2) File -> Open -> hangtrace_trace.etl
- 3) Search for "HangTrace" markers (they bracket the capture window)

Recommended graphs:

- CPU Usage (Sampled)
- CPU Usage (Precise)
- Thread Activity
- Disk Usage / File I/O
- Networking (if relevant)

7. COM/RPC cross-process hang attribution

A common “Not Responding” root cause is a COM call from the UI thread to another process (often via RPC). HangTrace improves visibility by registering a COM callback for Wait Chain Traversal when available, and by enriching blocker nodes with process and service details.

What this looks like in practice:

- UI thread wait chain includes COM or ALPC nodes
- WCT status may show PidOnly / PidOnlyRpcss (limited visibility)
- Blocker enrichment identifies likely server process and related services

Interpretation tips:

- If the chain indicates COM/RPC and the server is a service host, list the services hosted by that PID.
- PidOnlyRpcss often means the call is routed via RPCSS; treat it as a strong hint of cross-process COM/RPC wait.
- If you can, re-run with elevation and ETW micro-trace to improve attribution and timing.

8. Safety, privacy, and best practices

HangTrace is intended for diagnostics and troubleshooting. Use it responsibly.

- Run only on systems you own or have explicit permission to troubleshoot.
- Start minimal (HTML/JSON), then escalate to --deep, --dump, and --etw only if needed.
- Treat dumps and ETL traces as confidential. Store securely and share carefully.
- Prefer short capture windows. Avoid long tracing sessions on production machines.
- Document the command used and timestamps in your incident/ticket.

Close probe caution: --probe-close posts WM_CLOSE to the target window. Use only if you explicitly want to test the close path.

9. Troubleshooting

Access denied / empty stacks: Re-run as Administrator. Some processes are protected and may still limit access.

ETW mode says WPR missing: Install Windows Performance Toolkit (WPT) or ensure wpr.exe is on PATH.

Multiple matches for --process: Use --pick newest (or oldest), or target by --pid.

Report files not created: Check working directory permissions; use full paths for --html/--json/--dump/--etw.

Stacks show only system frames: Configure symbols; ensure your PDBs are available for user-mode frames.

Appendix A. Common hang patterns

Pattern	What you often see	Confirm with
UI thread doing work	High CPU; stacks show app code at top.	ETW CPU sampling; move work off UI thread.
Deadlock	WCT cycle; multiple threads waiting on each other.	Capture stacks for all involved threads.
I/O stall	Waits in file/network APIs; high I/O time.	ETW disk/network graphs; check storage/network.
COM/RPC stall	PidOnlyRpcss; COM/ALPC nodes; server PID hint.	ETW micro-trace; identify server process/service.
Shutdown cleanup	After close probe, new long waits appear.	Before/after delta; add async close + timeouts.

Appendix B. Artifact checklist for support tickets

When filing a bug or support ticket, include:

- HangTrace command line used (exact flags)
- Windows version/build (winver) and whether HangTrace was run elevated
- HTML report + JSON report (redact sensitive paths if needed)
- If safe: ETL trace and/or DMP (only share within trusted channels)
- Steps to reproduce + whether hang happens during normal use or during close

10. Disclaimer, license, and contact

Provided AS IS. HangTrace is provided without warranty of any kind. You assume all risk arising from use of the tool.

If you distribute HangTrace publicly, include your LICENSE.txt and DISCLAIMER.txt, and consider providing a contact channel for security reports and bug reports.

Suggested contact fields: support email, issue tracker URL, and release download page.