Kernel Level Threads (KLT)

Lars Plum– Principal Systems/Software Engineer, HPE
April 2026

© 2026 Hewlett Packard Enterprise Development LP

Forward looking statements

This is a rolling (up to three year) roadmap and is subject to change without notice.

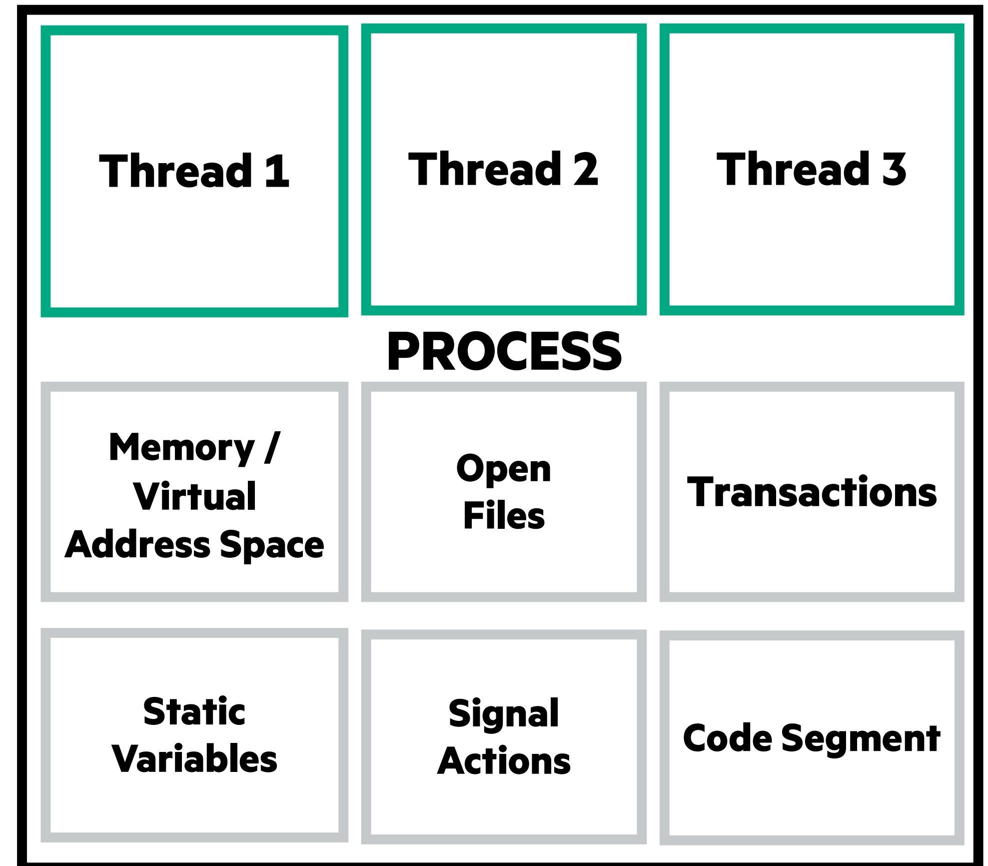
This document contains forward looking statements regarding future operations, product development, product capabilities and availability dates. This information is subject to substantial uncertainties and is subject to change at any time without prior notification. Statements contained in this document concerning these matters only reflect Hewlett Packard Enterprise's predictions and / or expectations as of the date of this document and actual results and future plans of Hewlett Packard Enterprise may differ significantly as a result of, among other things, changes in product strategy resulting from technological, internal corporate, market and other changes. This is not a commitment to deliver any material, code or functionality and should not be relied upon in making purchasing decisions.

Agenda

- **Multi-threading and prior thread models**
- **Parallel thread execution**
- **Application monitoring**
- **Migration**
- **Runtime and performance**

What is a multi-threaded application?

- Application divided into logically distinct execution paths within a process context
- Threads share process-level information with other threads
- Divides program flow into potentially parallel tasks
- Each thread has its own stacks (non-priv, priv, signal), registers and local variables

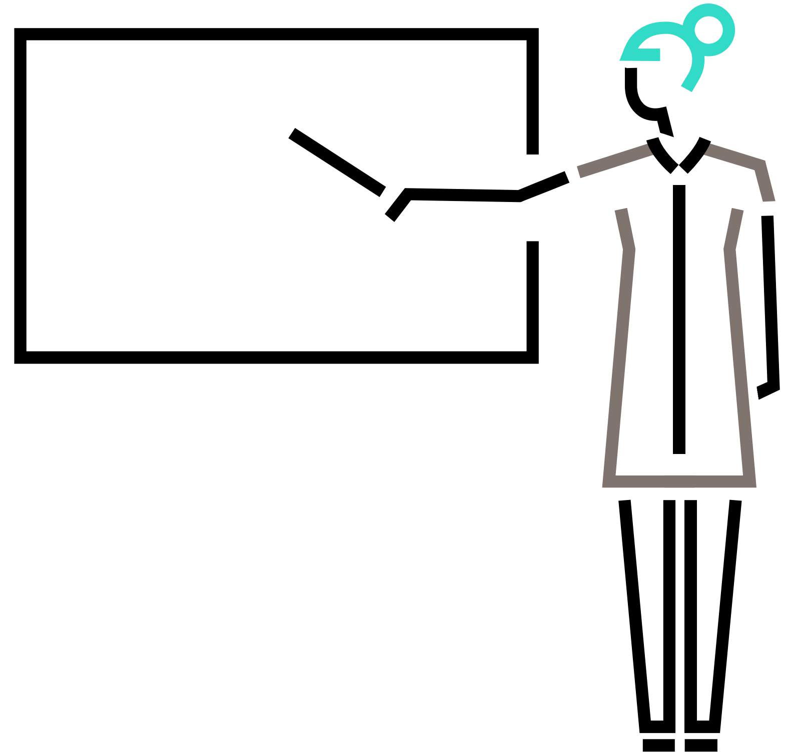


Thread model nomenclature

- 1x1: each user thread has a corresponding kernel thread (KLT)

- Mx1: multiple user threads are mapped to a single kernel thread (GTHREADs, SPT, PUT)

- MxN: multiple user threads are mapped to an equal or smaller set of kernel threads (uncommon)



Nonstop thread model history

Multi-threaded Servers

GTHREADs

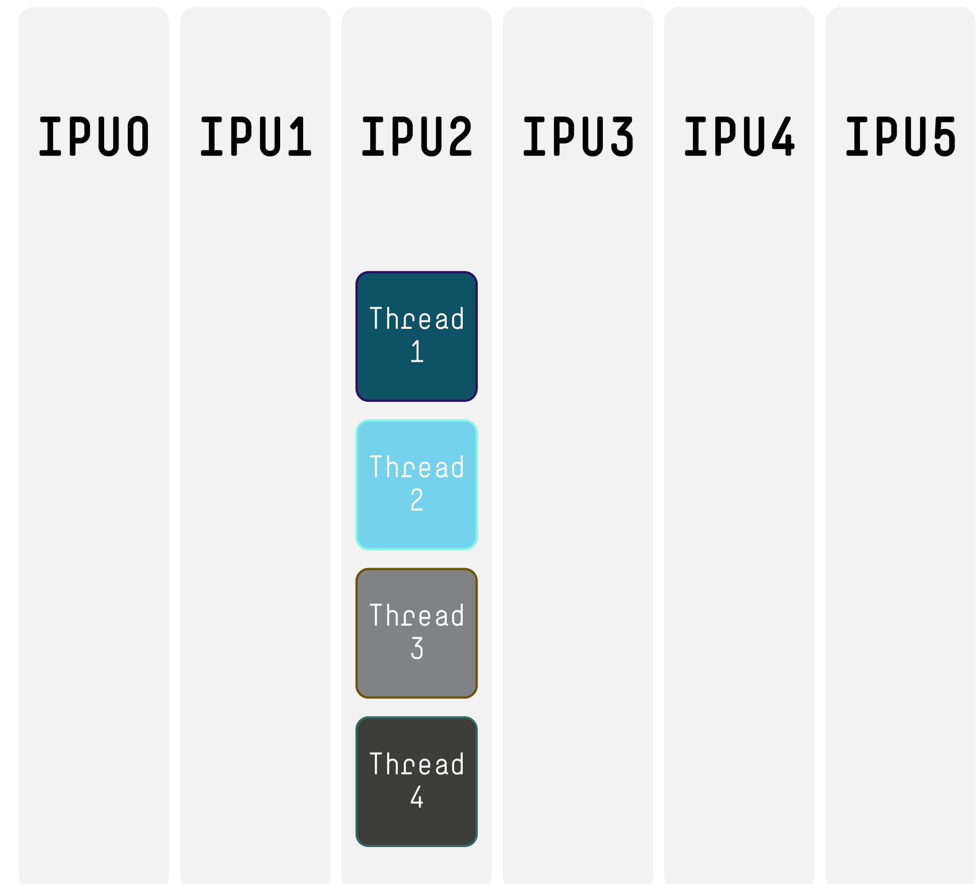
Standard POSIX Threads (SPT)

POSIX User Threads (PUT)

Kernel Level Threads (KLT) (L25.09 RVU)

POSIX User Thread Model (PUT) application (Mx1): cooperative execution

- OSS applications (32/64-bit)
- POSIX (IEEE 1003.1, 2004) compliant thread library (pthreads API)
- The PUT library controls thread creation, scheduling and management
- Cooperative execution model: threads execute one at a time in a single core
- Blocking calls (e.g. waited I/O) stall all threads
- Synchronizers: mutexes, condition variables
- Single underlying kernel thread
- Process-level performance limited to single core speed
- Nonstop OS not aware of the threads
- Same for GTHREADs and SPT
- Inefficient use of cores



Kernel Level Threads (KLT) application (1x1): parallel execution

- Nonstop OS provides native kernel support for threads (1x1)
- Shared process-level information
- Threads scheduled by Nonstop OS can execute in parallel across IPUs in a CPU
- Just like processes, threads on the same IPU run preemptively, and threads on different IPUs run in parallel
- Blocking call only affects calling thread
- Synchronizers: mutexes, condition variables, and read-write locks (new)
- POSIX (IEEE Std 1003.1, 2004) compliant thread library (pthreads API)
- More efficient use of cores

Shared and per-thread information

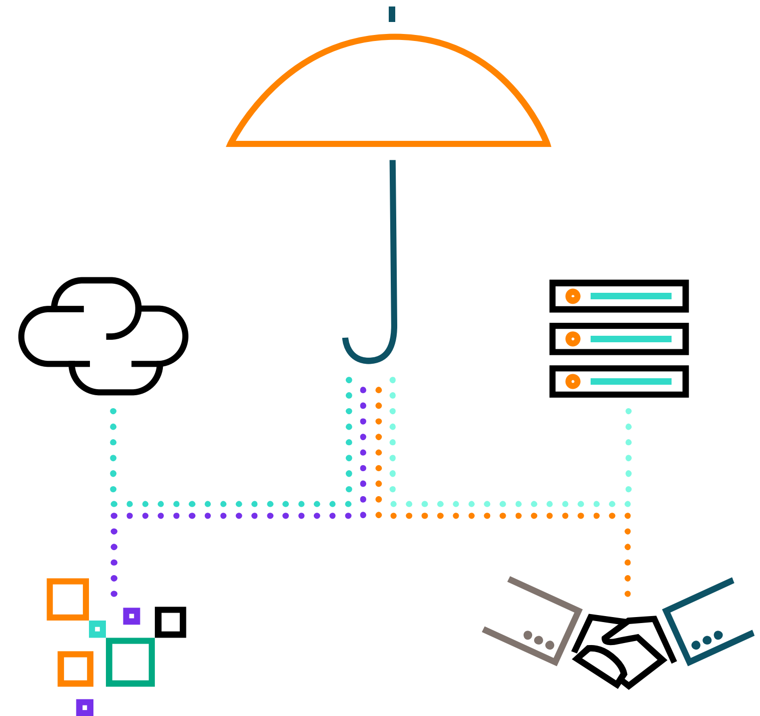
Defined by POSIX standard

- Shared by all threads
  - Thread Group ID (main thread PID)
  - Memory (data, heap, segments)
  - File system (open files, file descriptors, file state, file locks)
  - Static variables
  - SQL cursors
  - Environment variables
  - Active TMF transactions
  - Signal actions

- Per-thread information
  - Thread ID
  - Program counter
  - Register set
  - Stack
  - errno
  - Current transaction
  - Thread local storage (TLS)
  - Execution priority (managed by Nonstop OS; inherited) and 'nice' value
  - Signal mask

Characteristics of a KLT process

- Process identified by the main thread
- Subsequent threads: minor threads
- All threads in a given process are on the same CPU
- Minor threads initially launched in round-robin fashion on IPUs
- Each thread has a PIN and a PCB (Process Control Block)



OSS ps command: added options

-L (TPIN/NTHRD), -T (TID), -m (minor), -o (new columns), -W (BK column)

```
/home/suvit/ps_klt: ps -mLTf
UID                    PID        PPID        TID TPIN     C NTHRD TTY      STIME    TIME        CMD
SUPER.SUPER       16777218           1          - -        -     1 ?        Jul 22         00:00
SUPER.SUPER              -           -   16777218 0,495    -     - ?        Jul 22         00:00 -
SUPER.SUPER       16777220           1          - -        -     1 #zwn0002 Jul 22         00:00 -sh
SUPER.SUPER              -           -   16777220 3,500    -     - #zwn0002 Jul 22         00:00 -
SUPER.SUPER     2046820360           1          - -        -     1 #zwn0004 00:23:22       00:00 -sh
SUPER.SUPER              -           - 2046820360 3,544    -     - #zwn0004 00:23:22       00:00 -
SUPER.SUPER      704643077           1          - -        -     1 #zwn0005 00:24:36       00:00 -sh
SUPER.SUPER              -           -  704643077 3,572    -     - #zwn0005 00:24:36       00:00 -
SUPER.SUPER     1912602627  2046820360          - -        -    11 #zwn0004 00:25:14       00:00 ./thread10
SUPER.SUPER              -           - 1912602627 3,589    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   16777231 3,580    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   16777230 3,582    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           - 1862270982 3,586    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   16777232 3,590    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   50331657 3,591    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           - 1409286151 3,592    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   50331658 3,593    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   50331659 3,594    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   16777228 3,595    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER              -           -   16777229 3,596    -     - #zwn0004 00:25:14       00:00 -
SUPER.SUPER       50331665   704643077          - -        -     1 #zwn0005 00:25:32       00:00 ps -mLTf
SUPER.SUPER              -           -   50331665 3,599    -     - #zwn0005 00:25:32       00:00 -
/home/suvit/ps_klt:
```

Measure: PROCESS THREAD-VIEW option

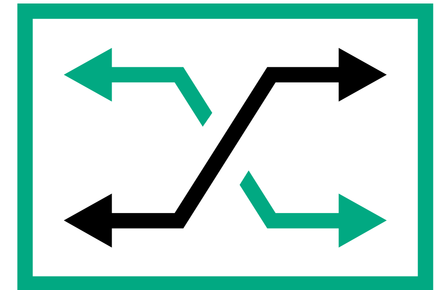
```
+ list process "/home/torgny/klt/kltbats", thread-view on
Process  3,578                Pri 149                 OSSPID:  16777222
                                             Thread Group ID:  16777222
Program  $OSS.ZYQ00001.Z0000ZAF:2181302867 (Native)
OSSPath: "/home/torgny/klt/kltbats"
Userid   255,255   Creatorid  255,255   Ancestor 3,575
Format Version:  L03  Data Version:  L03  Subsystem Version:  9
Local System \MEASCOG From  15 May 2020, 11:28:49   For    25 Seconds
------------ Processor -----------------------------------------------------
Cpu-Busy-Time                 0.02 %    Dispatches                   12.02 /s
Ready-Time                    0.02 %    Comp-Traps
Process-Launch-Qtime                    Process-Launches
Thread-Launch-Qtime                     Thread-Launches
Process-Launch-ART                      Thread-Launch-ART
Vsems                                   Ipu-Switches
Ipu-Num                          2 #
Initial-Priority-Start                  Initial-Priority-End           190 #
Current-Priority-Start                  Current-Priority-End           190 #
X-Interrupts                            Y-Interrupts
TLEs-Start                              TLEs-End
Ipu-Affinity-Class-Start                Ipu-Affinity-Class-End           2 #
```

PSTATE2: individual thread stacks

```
Call back trace for Thread: 2416
AsmBreakDispatchNoRelink + 0x2 (Milli)
$n_NSK_Event_Wait + 0xCA (Milli)
Timer_WaitOnEmbeddedTLE_ + 0xDE (SLr)
PROCESS_WAIT_int_(unsigned int, long long, bool) + 0x9D (SLr)
PROCESS_LONG_WAIT_ + 0x5B (SLr)
selectLP64_ + 0x61E (SLr)
$n_EnterPriv + 0x542 (Milli)
_TSL_select + 0x5D (DLL wcredll)
tandem_read(int, void *, unsigned long) + 0x1A3 (DLL libjvm.so)
pthread_mutex_lock + 0x96 (DLL wkltdll)
os::PlatformEvent::park(long) + 0x1BE (DLL libjvm.so)
JavaThread::sleep(long) + 0x114 (DLL libjvm.so)
JVM_Sleep + 0x1A2 (DLL libjvm.so)
java.lang.Thread::sleep(long) + 0xb7
```

Migration: Linux/SPT/PUT to KLT

- Build
  - OSS/Windows: -Wklt_model, -lklt (if separate link)
  - Guardian: KLT_MODEL
  - Process statically linked with KLT Model library (KLTDLL)
- Thread scheduling policy SCHED_OTHER, contention scope PTHREAD_SCOPE_SYSTEM
- Non-portable pthread APIs
  - pthread_getattr_np
  - pthread_getphandle_np
- Change "PUT_" calls to direct calls
- Not supported: pthread_attr_setstackaddr, pthread_attr_setstack
- Multistep OSS file I/O: use synchronizer
- Take advantage of read-write locks
- Test
  - CPU-intensive workloads benefit most
  - Parallel thread execution may reveal hidden scaling issues
- Liaison/concierge program

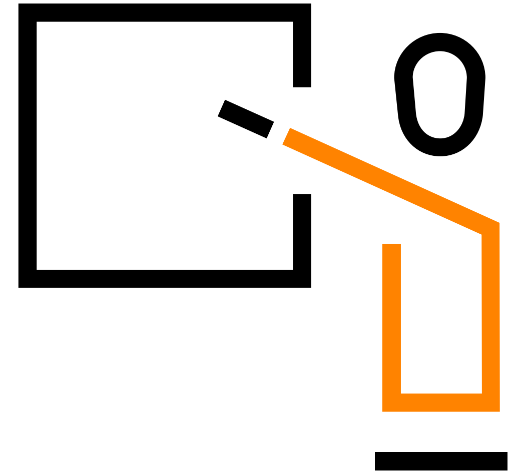


Runtime considerations

- Thread-safe and async-signal-safe functions
- Cancellation points
- Signal/assertion will lead to process termination
- Debugging: Native Inspect and NSDEE support thread debugging
  - New commands to list threads, select a thread, set thread-specific breakpoints, etc.
  - All threads suspended at debugging event such as a breakpoint
  - All threads resume when stepping or continuing
- Selectable segments not supported for KLT processes; use flat segments

Performance: Serialization

- Thread synchronization methods
  - Pthread mutex, condition variables, read-write locks, BINSEMs
  - Balance too few vs. too many
  - Orthogonal
  - Deadlocks
- File operations on shared file handles
  - `add_define =_KLT_FSPARALLELOP_EMS file=on`
  - Assign FDs to each thread or use nowait I/O
- SQL/MP and SQL/MX DB operations
- CRE heap operations
- Library calls



Performance: Shared memory

- Cache lines and coherency
- Naturally atomic operations on aligned scalars
  - Use SHARED8 field alignment, not SHARED2
- Take advantage of atomic functions (builtin.h)
- Favor atomics over mutexes
- Favor byte+ flags over bit flags

Performance summary

Application characteristics	Expectation
CPU-intensive application with minimal coordination between threads	Likely to see immediate benefit, including lower latency
CPU-intensive Java application	Likely to see benefit (with KLT-based JVM)
Application using parallel access to same file handle, other than $RECEIVE, opened for waited I/O	Likely to see degradation. Consider using nowait I/O or thread-specific file handles. Enable EMS messages to detect parallel file handle access.
Application using SQL/MX with JDBC Type 2 (T2) driver	Likely to see degradation

If you're running into performance issues but would like to take advantage of the portability benefits of KLT, consider pinning all of the threads in a KLT process to a single IPU

Possible workaround is to consider a hybrid model (more processes with fewer threads/process)

Nonstop subsystems updated for KLT

Nonstop Kernel	OSS File System	Standard Millicode	Compilers and runtime	Native Inspect and NSDEE
KLT DLL	TMF	Security	Measure	CIP
SQL/MP	SQL/MX	ODBC/MX	JVM	JDBC
OpenJDK Middleware	TS/MP	IMC	IOEDIT	TS/MP ACS

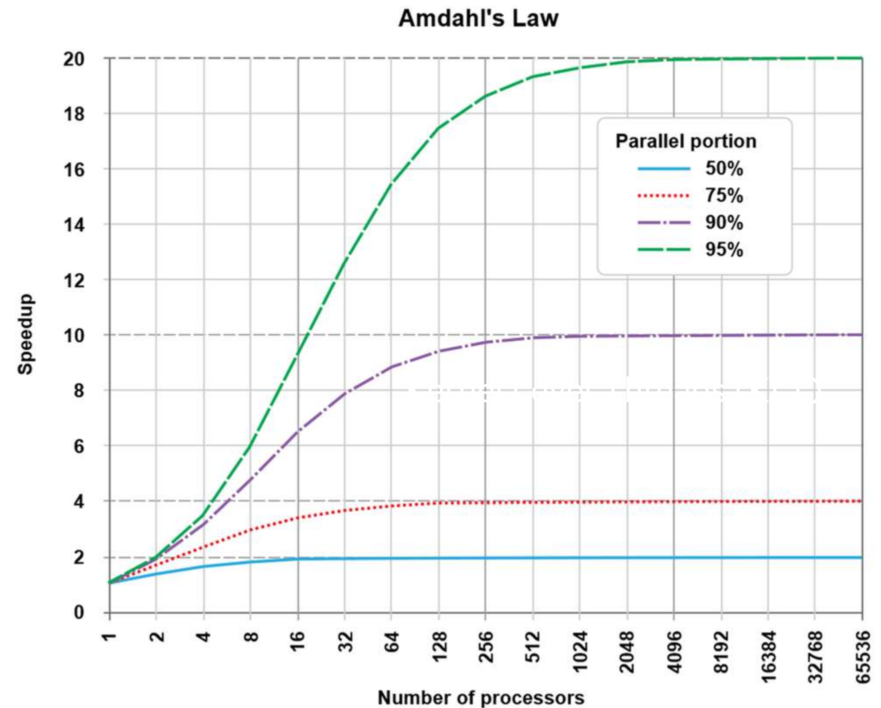
Java is a registered trademark of Oracle and/or its affiliates.

OpenJDK (the "Name") is a trademark of Oracle America, Inc. ("Oracle") (the "Trademark Owner")

Partner beta testing with KLT

What does KLT mean for you?

- KLT unlocks Nonstop multi-core architecture at the application level
- KLT-based JVM features increased performance, threaded GC
- Reduced application latency to support real-time applications
- Foundation for multi-threaded middleware and programming languages
- Ease porting of existing multi-threaded applications and languages to Nonstop
- GTHREAD and PUT libraries (including PUT-based JVM) remain available; SPT in Limited support



By Daniels220 at English Wikipedia, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=6678551

Nonstop partnership – it's a beautiful thing!

v2508

lars.plum@hpe.com

© 2026 Hewlett Packard Enterprise Development LP