Walkthrough: What's New in TotalView 2023.2

Bill Burns | July 2023

Senior Director of Software Engineering and Product Manager

Presenter

Bill Burns

Senior Director of Software Engineering and Product Manager

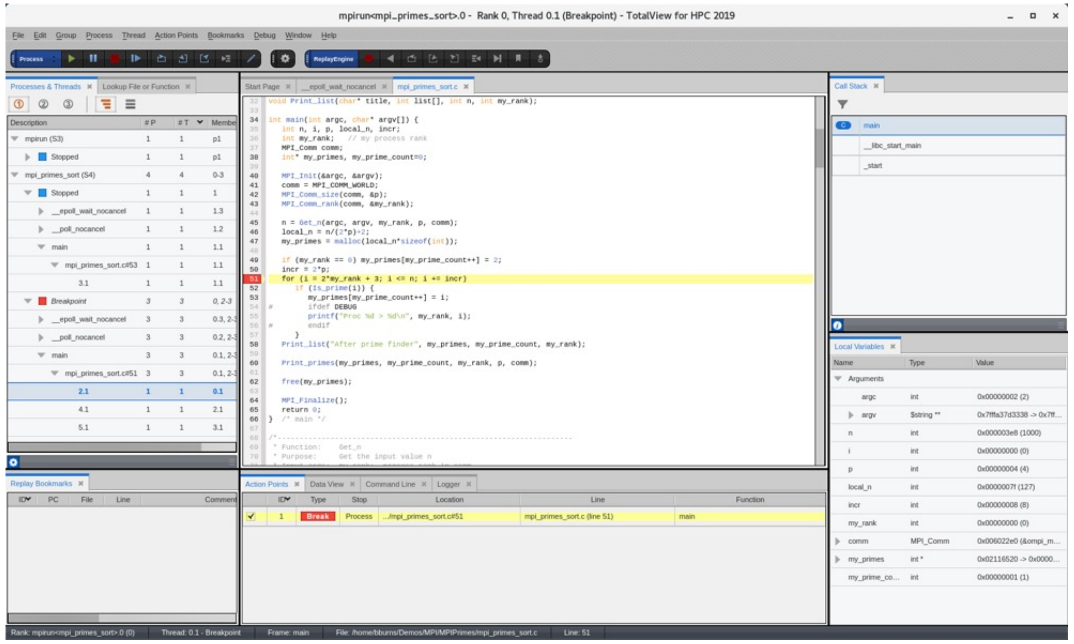
Agenda

- TotalView Performance Improvements
- GPU Advancements
- New Memory Debugging Features
- Array Visualization Advancements
- Other UI Enhancements
- Platform / Compiler / Other Updates
- Apple M1/M2 Beta
- Leveraging TotalView’s Debugging Technologies
- Q&A

HPC Debugging and Dynamic Analysis With TotalView

Debugging Complex Applications With TotalView

- Comprehensive multi-process/thread dynamic analysis and debugging
- Debug Hybrid MPI/OpenMP applications
- Advanced C, C++ and Fortran support
- NVIDIA CUDA and AMD ROCm GPU debugging support
- Integrated reverse debugging
- Mixed language C/C++ and Python debugging
- Memory leak detection
- Batch/unattended debugging



Description	# P	# T	Members
mpirun (S3)	1	1	p1
Stopped	1	1	p1
mpi_primes_sort (S4)	4	4	0-3
Stopped	1	1	1
__epoll_wait_nocancel	1	1	1.3
__poll_nocancel	1	1	1.2
main	1	1	1.1
mpi_primes_sort.c#53	1	1	1.1
3.1	1	1	1.1
Breakpoint	*3*	*3*	*0, 2-3*
__epoll_wait_nocancel	3	3	0.3, 2-3
__poll_nocancel	3	3	0.2, 2-3
main	3	3	0.1, 2-3
mpi_primes_sort.c#51	3	3	0.1, 2-3
2.1	1	1	0.1
4.1	1	1	2.1
5.1	1	1	3.1

```
void Print_list(char* title, int list[], int n, int my_rank);

int main(int argc, char* argv[]) {
   int n, i, p, local_n, incr;
   int my_rank;   // my process rank
   MPI_Comm comm;
   int* my_primes, my_prime_count=0;

   MPI_Init(&argc, &argv);
   comm = MPI_COMM_WORLD;
   MPI_Comm_size(comm, &p);
   MPI_Comm_rank(comm, &my_rank);

   n = Get_n(argc, argv, my_rank, p, comm);
   local_n = n/(2*p)+2;
   my_primes = malloc(local_n*sizeof(int));

   if (my_rank == 0) my_primes[my_prime_count++] = 2;
   incr = 2*p;
   for (i = 2*my_rank + 3; i <= n; i += incr)
      if (Is_prime(i)) {
         my_primes[my_prime_count++] = i;
#        ifdef DEBUG
         printf("Proc %d > %d\n", my_rank, i);
#        endif
      }
   Print_list("After prime finder", my_primes, my_prime_count, my_rank);

   Print_primes(my_primes, my_prime_count, my_rank, p, comm);

   free(my_primes);

   MPI_Finalize();
   return 0;
}  /* main */

/*------------------------------------------------------------------
 * Function:    Get_n
 * Purpose:     Get the input value n
```

ID	PC	File	Line	Comment

	ID	Type	Stop	Location	Line	Function
✓	1	Break	Process	../mpi_primes_sort.c#51	mpi_primes_sort.c (line 51)	main

Name	Type	Value
Arguments		
argc	int	0x00000002 (2)
argv	$string **	0x7fffa37d3338 -> 0x7ff...
n	int	0x000003e8 (1000)
i	int	0x00000000 (0)
p	int	0x00000004 (4)
local_n	int	0x0000007f (127)
incr	int	0x00000008 (8)
my_rank	int	0x00000000 (0)
comm	MPI_Comm	0x006022e0 (&ompi_m...
my_primes	int *	0x02116520 -> 0x0000...
my_prime_co...	int	0x00000001 (1)

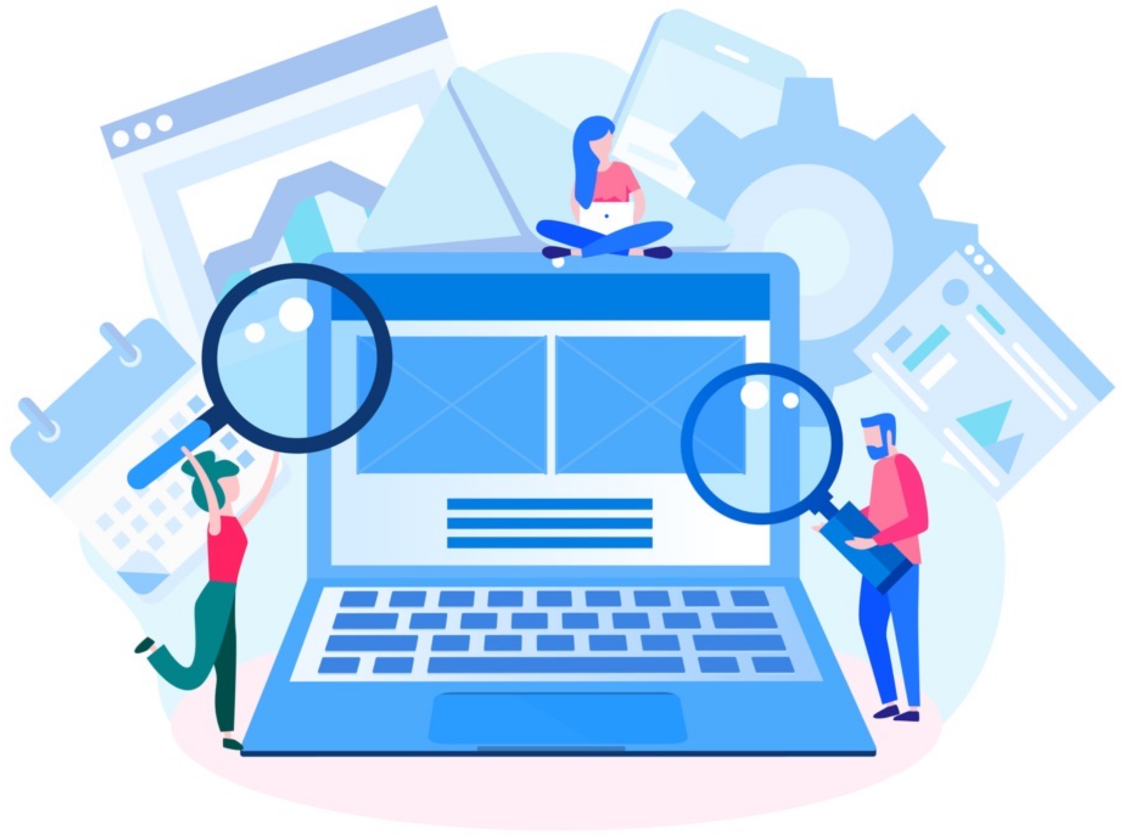
Rank: mpirun<mpi_primes_sort> 0 (0) Thread: 0.1 - Breakpoint Frame: main File: /home/bburns/Demos/MPI/MPIPrimes/mpi_primes_sort.c Line: 51

Supported Technologies…

LANGUAGES	OPERATING SYSTEMS	APPLICATIONS	PLATFORMS
Ftn		MPI, NVIDIA CUDA, OpenMP, OpenACC	intel, NVIDIA, arm, AMD, Hewlett Packard Enterprise

Debuggers – More Than Just a Tool to Find Bugs

- Understand complex code
- Improve developer efficiency
- Collaborate with team members
- Improve code quality
- Shorten development time



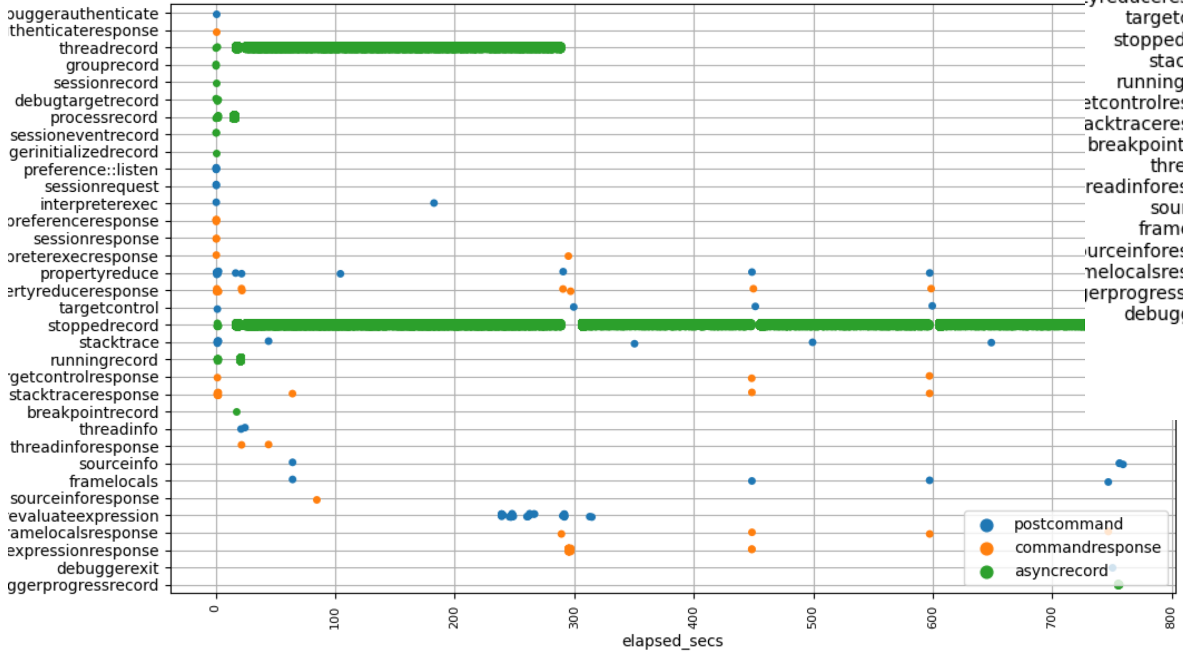
TotalView Performance Improvements

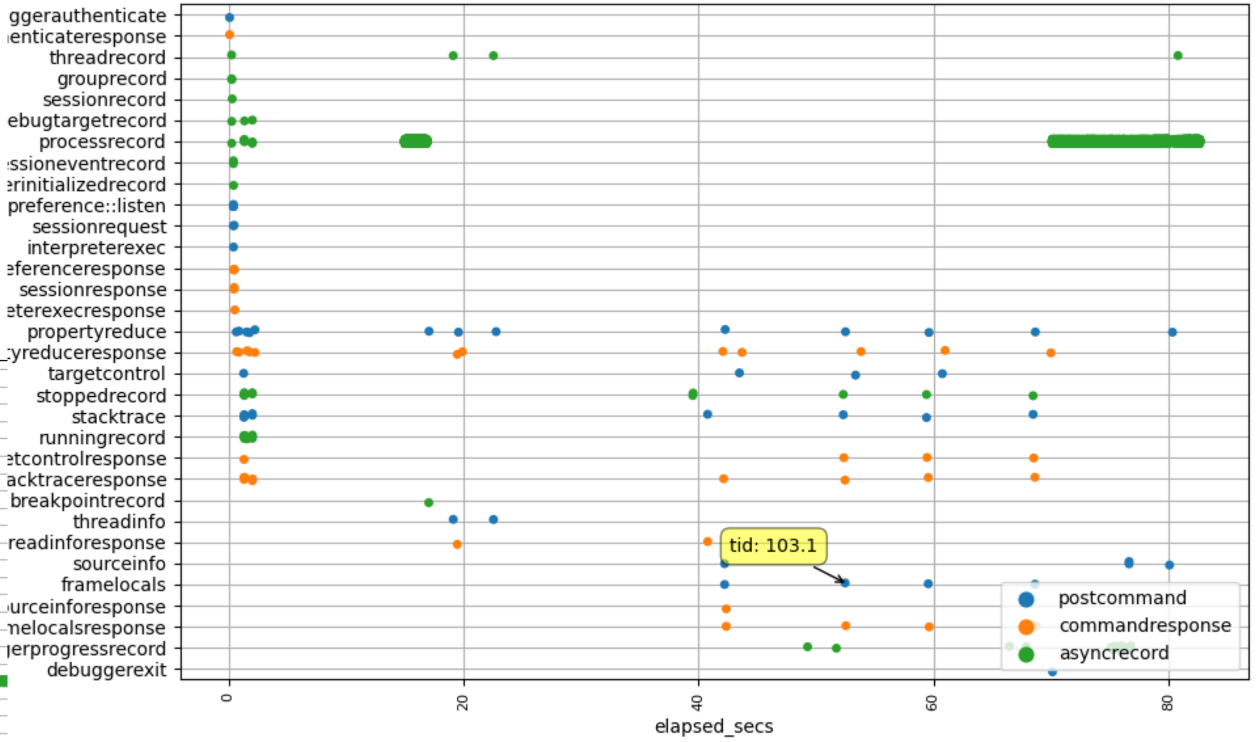
PERFORCE

TotalView UI Performance at 5,000+ Ranks

UI Performance Improvements

- Scalability
- Responsiveness – low and high scales
- Improved front-end process/thread states





TotalView GPU Advancements

PERFORCE

GPU Debugging with TotalView

AMD GPUs

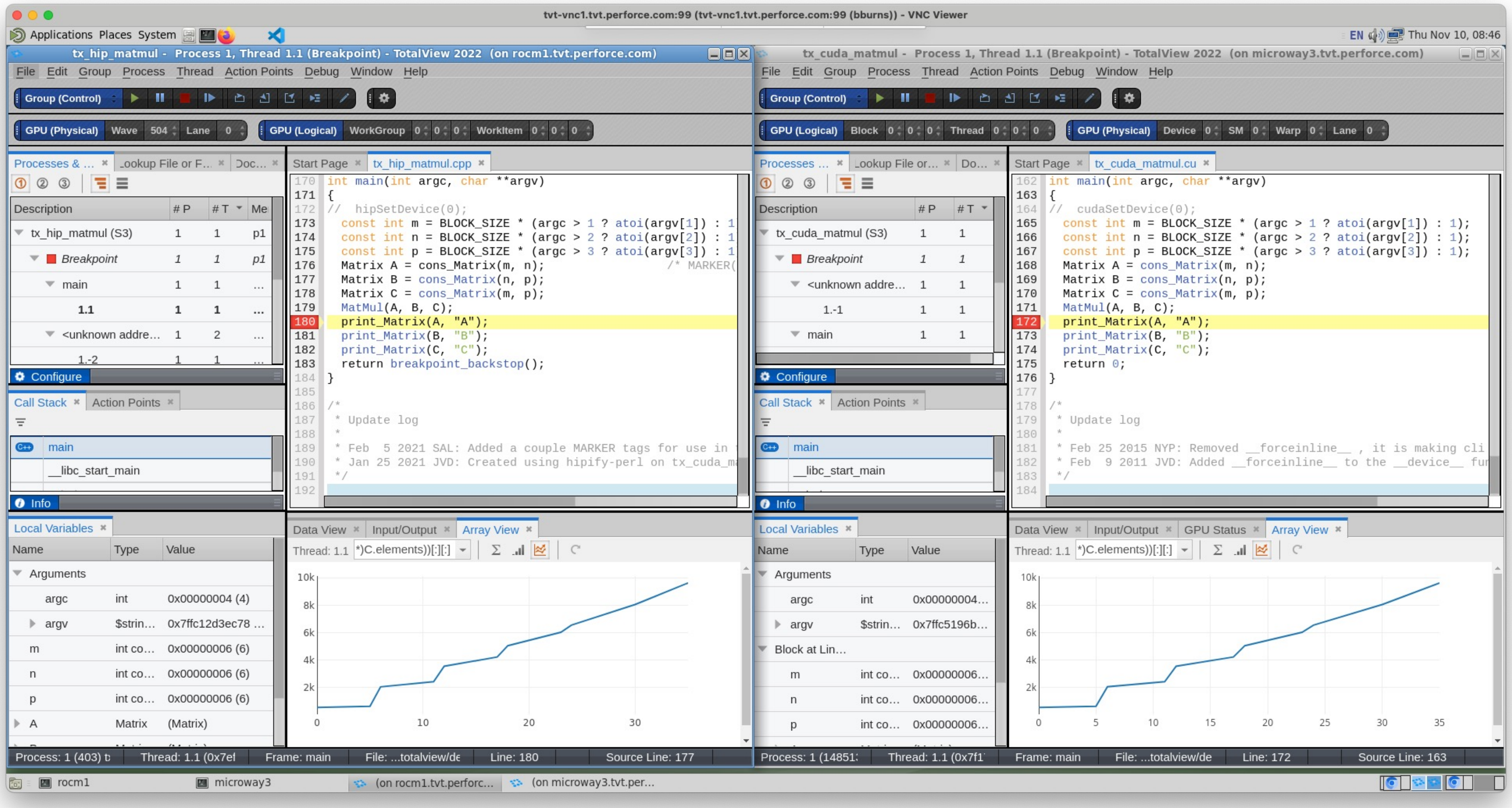
- AMD MI50, MI100, and MI200 series of GPUs
- ROCm 5.1 and 5.2, 5.4, and 5.5
- Debug HIP (Heterogeneous Interface for Portability) and MPI
- Debugging Features:
    - Process launch, attach, and detach
    - Viewing scalar, vector, general, and special AMD GPU registers
    - Instruction disassembly
    - Breakpoint creation and deletion on AMD GPU code
    - Single-stepping and fast smart-stepping
    - Stack unwinding, including inlined functions
    - Navigation controls for changing the logical workgroup / work-item focus or physical agent, queue, dispatch, wave, and lane focus
    - Variable display with the ROCm 5.1+ compilers
    - Data watchpoints on global memory variables

NVIDIA GPUs

- NVIDIA Tesla, Fermi, Kepler, Pascal, Volta, Turing, Ampere, **Hopper coming in 2023**
- NVIDIA CUDA 9.2, 10 and 11
    - With support for Unified Memory
- Debugging 64-bit CUDA programs
- Features and capabilities include
    - Support for dynamic parallelism
    - Support for MPI based clusters and multi-card configurations
    - Flexible Display and Navigation on the CUDA device
        - Physical (device, SM, Warp, Lane)
        - Logical (Grid, Block) tuples
    - GPU Status View reveals what is running where
    - Support for types and separate memory address spaces
    - Leverages CUDA memcheck

TotalView GPU Debugging Enhancements

- AMD GPUs
  - ROCm 5.4 and 5.5
- NVIDIA GPUs
  - CUDA 11.8


tvt-vnc1.tvt.perforce.com:99 (tvt-vnc1.tvt.perforce.com:99 (bburns)) - VNC Viewer
Applications Places System
Thu Nov 10, 08:46
tx_hip_matmul - Process 1, Thread 1.1 (Breakpoint) - TotalView 2022 (on rocm1.tvt.perforce.com)
tx_cuda_matmul - Process 1, Thread 1.1 (Breakpoint) - TotalView 2022 (on microway3.tvt.perforce.com)
File Edit Group Process Thread Action Points Debug Window Help
Group (Control)
GPU (Physical) Wave 504 Lane 0
GPU (Logical) WorkGroup 0 0 0 WorkItem 0 0 0
GPU (Logical) Block 0 0 0 Thread 0 0 0
GPU (Physical) Device 0 SM 0 Warp 0 Lane 0
Start Page
tx_hip_matmul.cpp
tx_cuda_matmul.cu
Configure
Call Stack
Action Points
main
__libc_start_main
Info
Data View
Input/Output
GPU Status
Array View
Thread: 1.1 *)C.elements))[:][:]
rocm1
microway3
(on rocm1.tvt.perforc...
(on microway3.tvt.per...

Description	# P	# T	Me
tx_hip_matmul (S3)	1	1	p1
Breakpoint	1	1	p1
main	1	1	...
1.1	1	1	...
<unknown addre...	1	2	...
1.-2	1	1	...

```
int main(int argc, char **argv)
{
//  hipSetDevice(0);
  const int m = BLOCK_SIZE * (argc > 1 ? atoi(argv[1]) : 1
  const int n = BLOCK_SIZE * (argc > 2 ? atoi(argv[2]) : 1
  const int p = BLOCK_SIZE * (argc > 3 ? atoi(argv[3]) : 1
  Matrix A = cons_Matrix(m, n);                         /* MARKER(
  Matrix B = cons_Matrix(n, p);
  Matrix C = cons_Matrix(m, p);
  MatMul(A, B, C);
  print_Matrix(A, "A");
  print_Matrix(B, "B");
  print_Matrix(C, "C");
  return breakpoint_backstop();
}

/*
 * Update log
 *
 * Feb  5 2021 SAL: Added a couple MARKER tags for use in
 * Jan 25 2021 JVD: Created using hipify-perl on tx_cuda_ma
 */
```

Name	Type	Value
Arguments		
argc	int	0x00000004 (4)
argv	$strin...	0x7ffc12d3ec78 ...
m	int co...	0x00000006 (6)
n	int co...	0x00000006 (6)
p	int co...	0x00000006 (6)
A	Matrix	(Matrix)

```
Process: 1 (403) t   Thread: 1.1 (0x7ef   Frame: main   File: ...totalview/de   Line: 180   Source Line: 177
```

Description	# P	# T
tx_cuda_matmul (S3)	1	1
Breakpoint	1	1
<unknown addre...	1	1
1.-1	1	1
main	1	1

```
int main(int argc, char **argv)
{
//  cudaSetDevice(0);
  const int m = BLOCK_SIZE * (argc > 1 ? atoi(argv[1]) : 1);
  const int n = BLOCK_SIZE * (argc > 2 ? atoi(argv[2]) : 1);
  const int p = BLOCK_SIZE * (argc > 3 ? atoi(argv[3]) : 1);
  Matrix A = cons_Matrix(m, n);
  Matrix B = cons_Matrix(n, p);
  Matrix C = cons_Matrix(m, p);
  MatMul(A, B, C);
  print_Matrix(A, "A");
  print_Matrix(B, "B");
  print_Matrix(C, "C");
  return 0;
}

/*
 * Update log
 *
 * Feb 25 2015 NYP: Removed __forceinline__ , it is making cli
 * Feb  9 2011 JVD: Added __forceinline__ to the __device__ fu
 */
```

Name	Type	Value
Arguments		
argc	int	0x00000004...
argv	$strin...	0x7ffc5196b...
Block at Lin...		
m	int co...	0x00000006...
n	int co...	0x00000006...
p	int co...	0x00000006...

```
Process: 1 (14851:   Thread: 1.1 (0x7f1   Frame: main   File: ...totalview/de   Line: 172   Source Line: 163
```

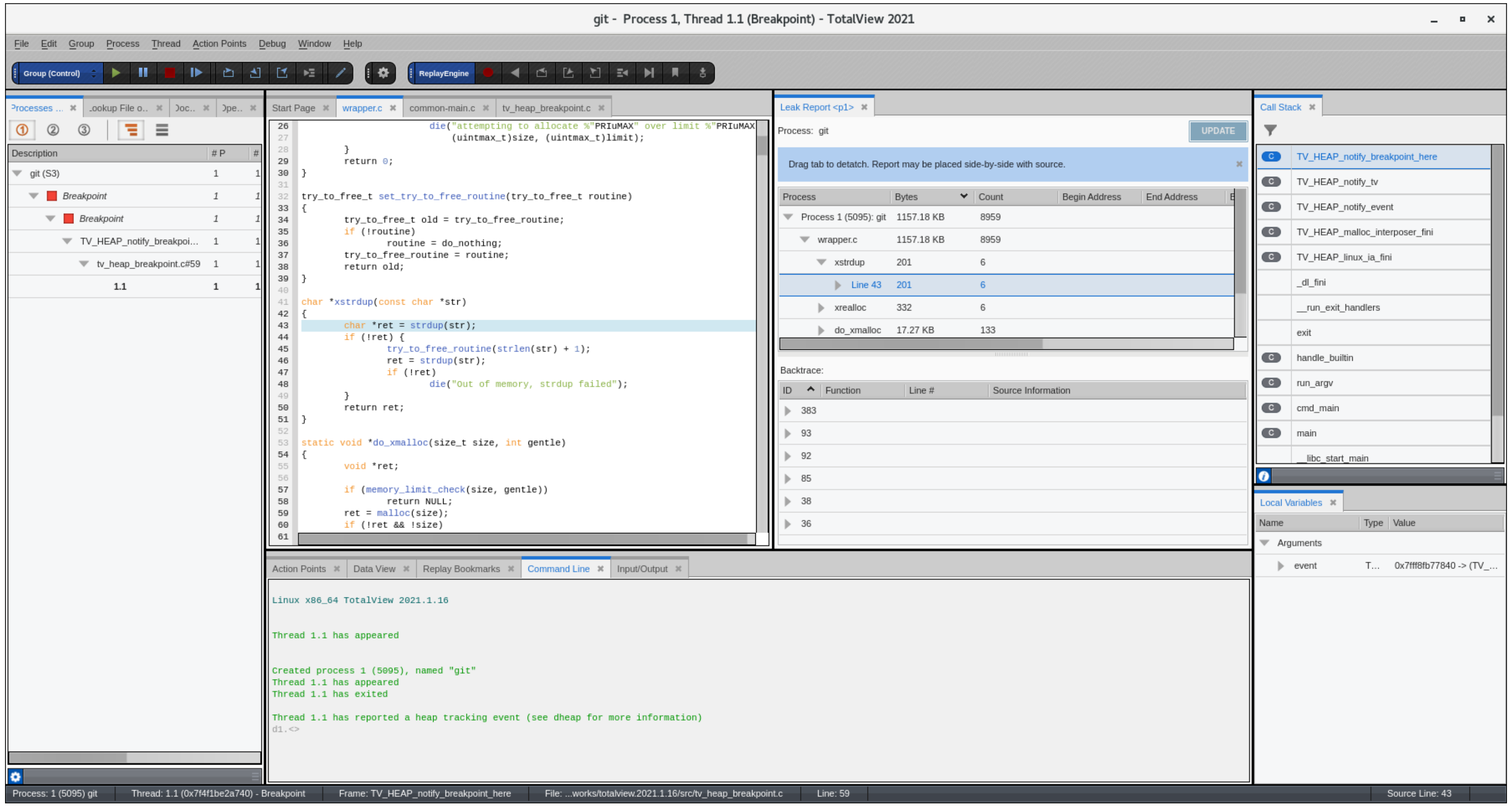
TotalView New Memory Debugging Features

PERFORCE

Memory Debugging with TotalView

Memory Debugging Features

- Leak detection
- Dangling pointer detection
- Heap status
- Automatically detect allocation problems
- Memory Corruption Detection
- Memory Block Painting
- Memory Hoarding
- Memory Comparisons between processes
- Light weight memory block tracking



git - Process 1, Thread 1.1 (Breakpoint) - TotalView 2021

Description	# P	#
git (S3)	1	1
Breakpoint	1	1
Breakpoint	1	1
TV_HEAP_notify_breakpoi...	1	1
tv_heap_breakpoint.c#59	1	1
1.1	1	1

```
                        die("attempting to allocate %"PRIuMAX" over limit %"PRIuMAX
                            (uintmax_t)size, (uintmax_t)limit);
        }
        return 0;
}

try_to_free_t set_try_to_free_routine(try_to_free_t routine)
{
        try_to_free_t old = try_to_free_routine;
        if (!routine)
                routine = do_nothing;
        try_to_free_routine = routine;
        return old;
}

char *xstrdup(const char *str)
{
        char *ret = strdup(str);
        if (!ret) {
                try_to_free_routine(strlen(str) + 1);
                ret = strdup(str);
                if (!ret)
                        die("Out of memory, strdup failed");
        }
        return ret;
}

static void *do_xmalloc(size_t size, int gentle)
{
        void *ret;

        if (memory_limit_check(size, gentle))
                return NULL;
        ret = malloc(size);
        if (!ret && !size)
```

Leak Report <p1>

Process: git

Drag tab to detatch. Report may be placed side-by-side with source.

Process	Bytes	Count	Begin Address	End Address
Process 1 (5095): git	1157.18 KB	8959		
wrapper.c	1157.18 KB	8959		
xstrdup	201	6		
Line 43	201	6		
xrealloc	332	6		
do_xmalloc	17.27 KB	133		

Backtrace:

ID	Function	Line #	Source Information
383			
93			
92			
85			
38			
36			

Call Stack

- TV_HEAP_notify_breakpoint_here
- TV_HEAP_notify_tv
- TV_HEAP_notify_event
- TV_HEAP_malloc_interposer_fini
- TV_HEAP_linux_ia_fini
- _dl_fini
- __run_exit_handlers
- exit
- handle_builtin
- run_argv
- cmd_main
- main
- __libc_start_main

Local Variables

Name	Type	Value
Arguments		
event	T...	0x7fff8fb77840 -> (TV_...

```
Linux x86_64 TotalView 2021.1.16


Thread 1.1 has appeared


Created process 1 (5095), named "git"
Thread 1.1 has appeared
Thread 1.1 has exited

Thread 1.1 has reported a heap tracking event (see dheap for more information)
d1.<>
```

Process: 1 (5095) git | Thread: 1.1 (0x7f4f1be2a740) - Breakpoint | Frame: TV_HEAP_notify_breakpoint_here | File: ...works/totalview.2021.1.16/src/tv_heap_breakpoint.c | Line: 59 | Source Line: 43

New Memory Debugging Features in TotalView

Painting

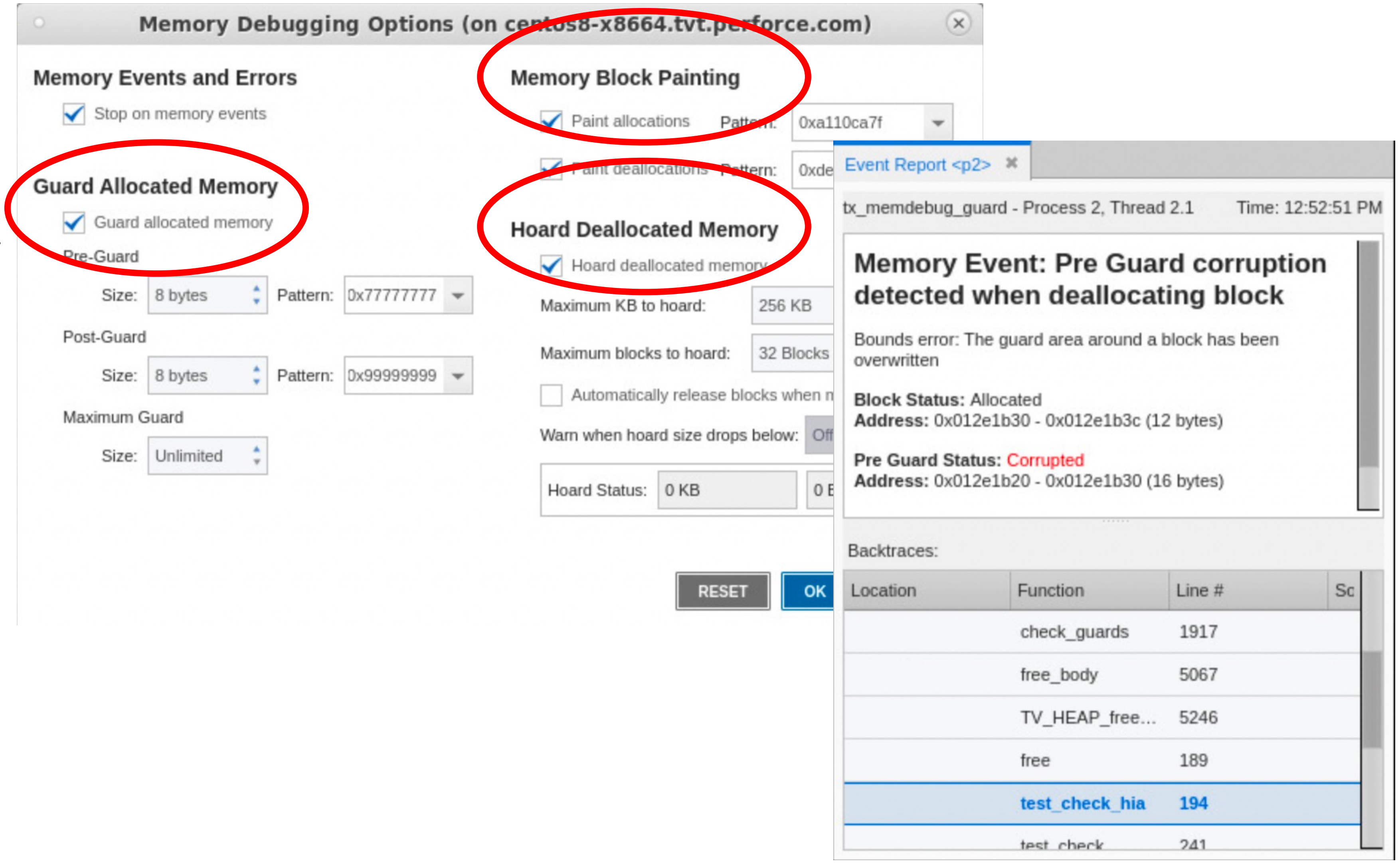
- Initialize new block
- Invalidate freed block
- Provide consistent, invalid memory contents

Hoarding

- Hold onto freed memory without asking OS to free it
- Debug dangling pointer problems

Memory Overwrite Detection

- Notified when memory bounds are overwritten
- Use Reverse Debugging and Watchpoints to find overwrite location



Memory Debugging Options (on centos8-x8664.tvt.perforce.com)

Memory Events and Errors

✓ Stop on memory events

Guard Allocated Memory

✓ Guard allocated memory

Pre-Guard

Size: 8 bytes Pattern: 0x77777777

Post-Guard

Size: 8 bytes Pattern: 0x99999999

Maximum Guard

Size: Unlimited

Memory Block Painting

✓ Paint allocations Pattern: 0xa110ca7f

✓ Paint deallocations Pattern: 0xde

Hoard Deallocated Memory

✓ Hoard deallocated memory

Maximum KB to hoard: 256 KB

Maximum blocks to hoard: 32 Blocks

☐ Automatically release blocks when m

Warn when hoard size drops below: Off

Hoard Status: 0 KB 0 E

Event Report <p2>

tx_memdebug_guard - Process 2, Thread 2.1 Time: 12:52:51 PM

Memory Event: Pre Guard corruption detected when deallocating block

Bounds error: The guard area around a block has been overwritten

Block Status: Allocated
Address: 0x012e1b30 - 0x012e1b3c (12 bytes)

Pre Guard Status: Corrupted
Address: 0x012e1b20 - 0x012e1b30 (16 bytes)

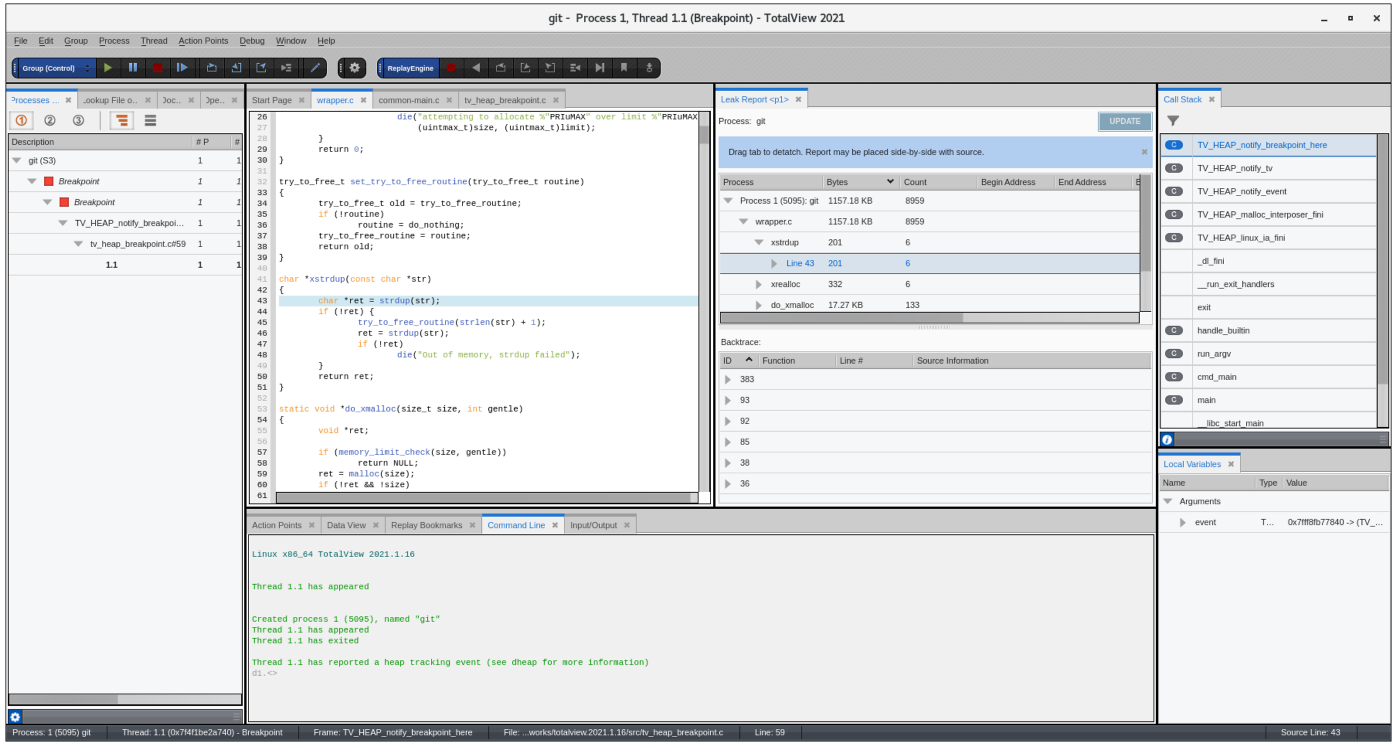
Backtraces:

Location	Function	Line #	Sc
	check_guards	1917	
	free_body	5067	
	TV_HEAP_free...	5246	
	free	189	
	test_check_hia	**194**	
	test_check	241	

Solving Tough Memory Bugs with TotalView Blog

Upcoming TotalView 2023.2 Blog

- Upcoming blog that finds a tough dangling pointer / memory corruption bug with TotalView
- Uses Memory Debugging, Painting, Hoarding, Reverse Debugging, Watchpoints, and Data Debugging features of TotalView
- https://totalview.io/blog



git - Process 1, Thread 1.1 (Breakpoint) - TotalView 2021
File Edit Group Process Thread Action Points Debug Window Help
Group (Control)
ReplayEngine
Processes ... Lookup File o.. Doc.. Ope..
Start Page wrapper.c common-main.c tv_heap_breakpoint.c
Leak Report <p1>
UPDATE
Call Stack
Local Variables
Action Points Data View Replay Bookmarks Command Line Input/Output

Description	# P	#
git (S3)	1	1
Breakpoint	1	1
Breakpoint	1	1
TV_HEAP_notify_breakpoi...	1	1
tv_heap_breakpoint.c#59	1	1
1.1	1	1

```c
                        die("attempting to allocate %"PRIuMAX" over limit %"PRIuMAX
                            (uintmax_t)size, (uintmax_t)limit);
        }
        return 0;
}

try_to_free_t set_try_to_free_routine(try_to_free_t routine)
{
        try_to_free_t old = try_to_free_routine;
        if (!routine)
                routine = do_nothing;
        try_to_free_routine = routine;
        return old;
}

char *xstrdup(const char *str)
{
        char *ret = strdup(str);
        if (!ret) {
                try_to_free_routine(strlen(str) + 1);
                ret = strdup(str);
                if (!ret)
                        die("Out of memory, strdup failed");
        }
        return ret;
}

static void *do_xmalloc(size_t size, int gentle)
{
        void *ret;

        if (memory_limit_check(size, gentle))
                return NULL;
        ret = malloc(size);
        if (!ret && !size)
```

Process: git

Drag tab to detatch. Report may be placed side-by-side with source.

Process	Bytes	Count	Begin Address	End Address
Process 1 (5095): git	1157.18 KB	8959		
wrapper.c	1157.18 KB	8959		
xstrdup	201	6		
Line 43	201	6		
xrealloc	332	6		
do_xmalloc	17.27 KB	133		

Backtrace:

ID	Function	Line #	Source Information
383			
93			
92			
85			
38			
36			

C	TV_HEAP_notify_breakpoint_here
C	TV_HEAP_notify_tv
C	TV_HEAP_notify_event
C	TV_HEAP_malloc_interposer_fini
C	TV_HEAP_linux_ia_fini
	_dl_fini
	__run_exit_handlers
	exit
C	handle_builtin
C	run_argv
C	cmd_main
C	main
	__libc_start_main

Name	Type	Value
Arguments		
event	T...	0x7fff8fb77840 -> (TV_...

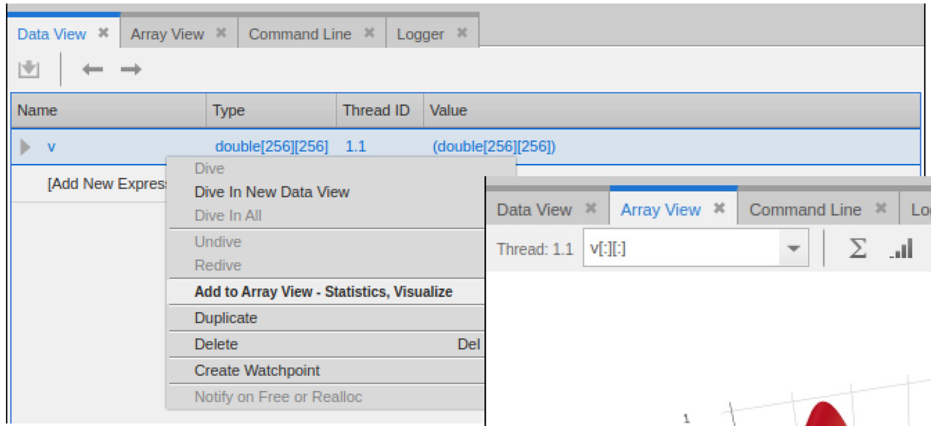
```
Linux x86_64 TotalView 2021.1.16


Thread 1.1 has appeared


Created process 1 (5095), named "git"
Thread 1.1 has appeared
Thread 1.1 has exited

Thread 1.1 has reported a heap tracking event (see dheap for more information)
d1.<>
```

Process: 1 (5095) git | Thread: 1.1 (0x7f4f1be2a740) - Breakpoint | Frame: TV_HEAP_notify_breakpoint_here | File: ...works/totalview.2021.1.16/src/tv_heap_breakpoint.c | Line: 59 | Source Line: 43

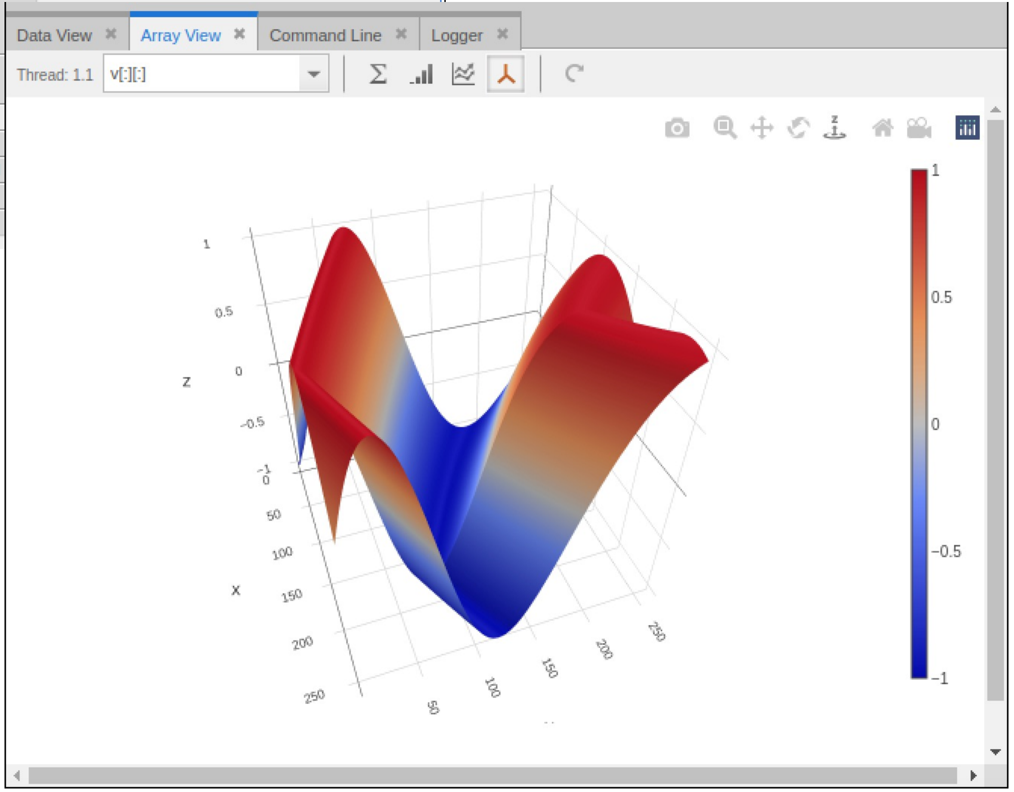
TotalView Array Visualization Advancements

TotalView Array Visualization Advancements

- Array View provides array specific debugging capabilities
  - Array statistics
  - Histogram
  - 2D Plot
- TotalView 2023.2 adds Surface Plot visualization



Name	Type	Thread ID	Value
v	double[256][256]	1.1	(double[256][256])
[Add New Expres			



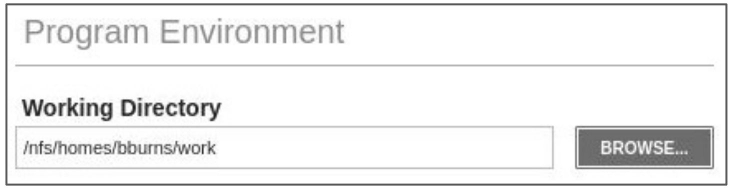
TotalView Other UI Enhancements

PERFORCE

Other UI Enhancements in TotalView

Debug Session Working Directory

- Specify the working directory that TotalView should run your program from



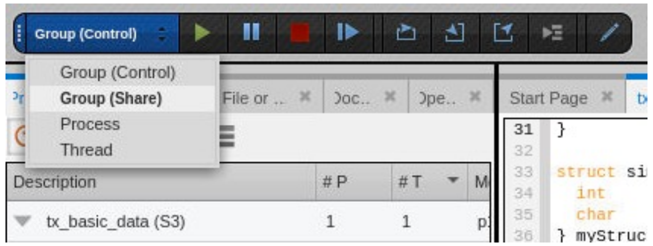
Program Environment

Working Directory

/nfs/homes/bburns/work

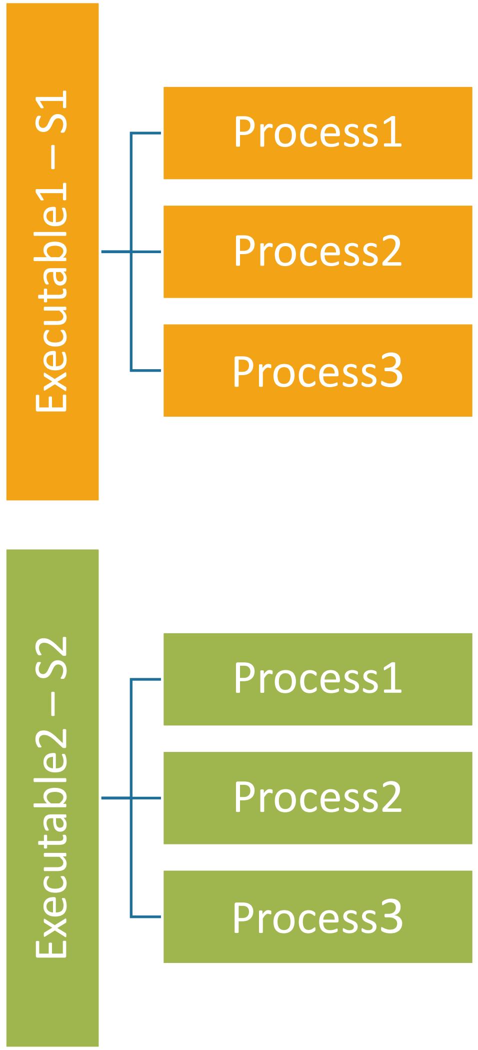
Share Group Support

- TotalView organizes processes associated by executables into Share groups
- Set breakpoints, issue debugger commands by Share group
- Upcoming blog details using Share Groups for MPMD debugging



Description	# P	# T
tx_basic_data (S3)	1	1

```
}

struct si
  int
  char
} myStruc
```



Python 3.10 Mixed Language Debugging support

- TotalView 2023.1 added support for Python 3.10

TotalView Platform and Other Updates

PERFORCE

TotalView Platform and Other Updates

Platform / Compiler Updates

- macOS Ventura (Intel)
- Ubuntu 22.04
- Fedora 35 / 36
- AIX 7.2 and 7.3 - IBM Open XL C/C++ 17.1
- Rocky Linux

Other Updates

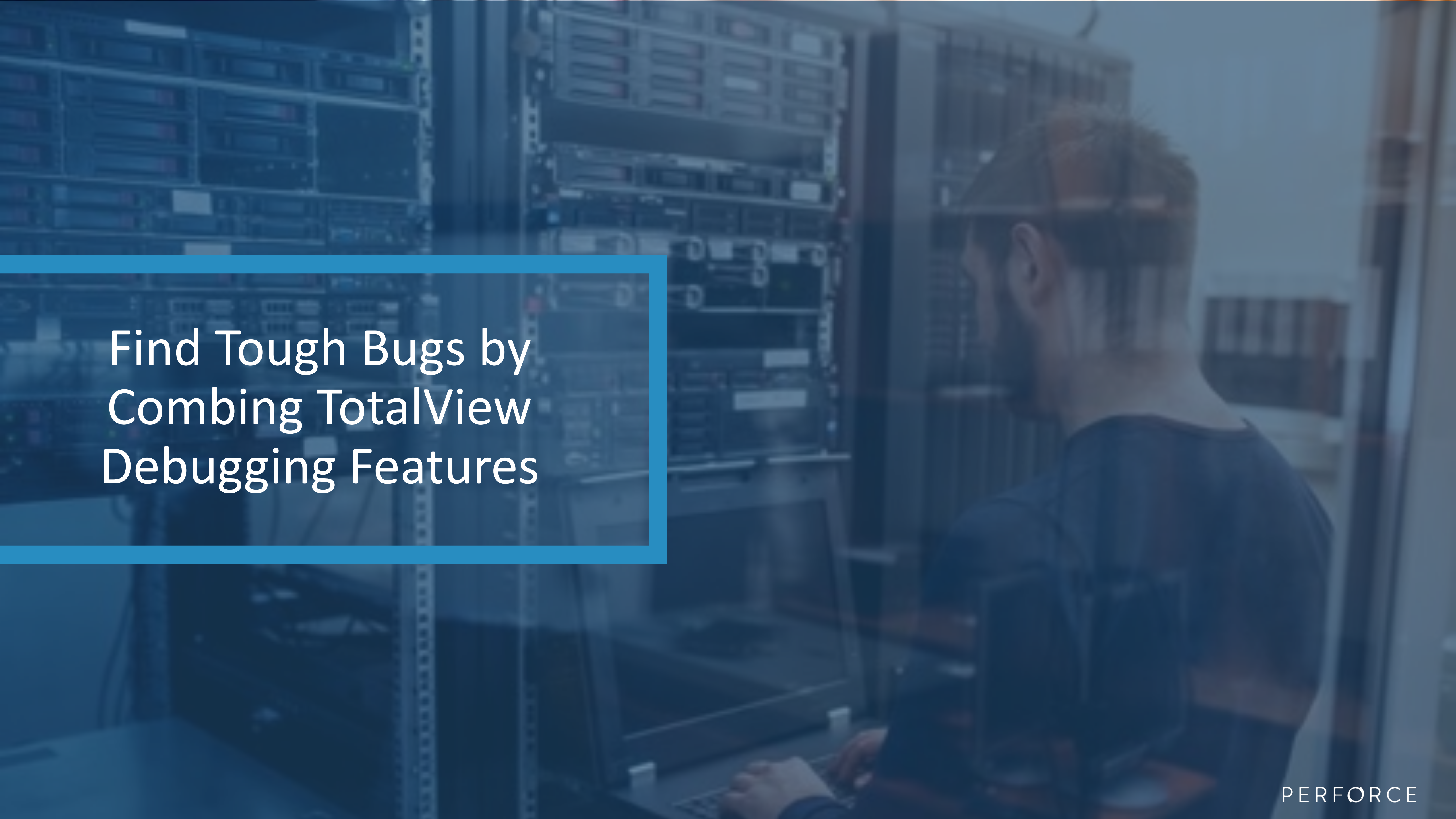
- Various bug fixes and other minor enhancements
- Third-party open-source package updates - security

TotalView Apple M1/M2 Beta

PERFORCE

TotalView support for Apple M1/M2

- TotalView 2023.3 will add support for Apple M1/M2 ARM64 Silicon
- Has required significant modifications and improvements to TotalView
- Early beta for M1/M2 support available starting in late July
- If interested in participating contact
  - Bill Burns – bburns@perforce.com

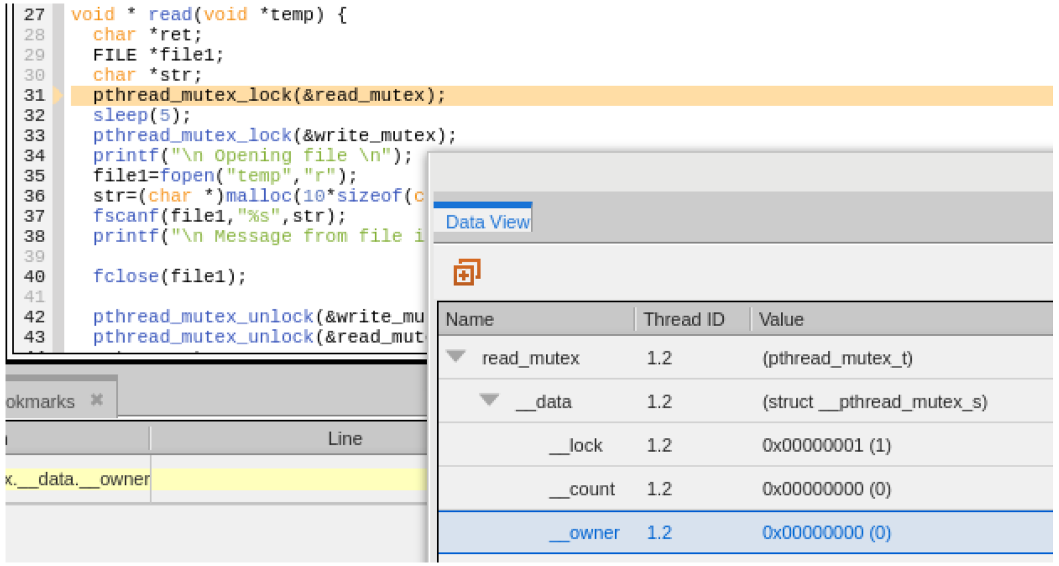


Find Tough Bugs by Combing TotalView Debugging Features

PERFORCE

Combine Multiple Debugging Features

- Find where a mutex lock was acquired
    - Combine reverse debugging and watchpoints
    - Run backwards until pthread_mutex_t __owner changes
- Mix source **code** debugging, **reverse** debugging and **memory** debugging
    - Find memory allocations and leaks during your debugging session
- Use TotalView's **Remote UI** for efficient debugging using all TotalView's features from your laptop

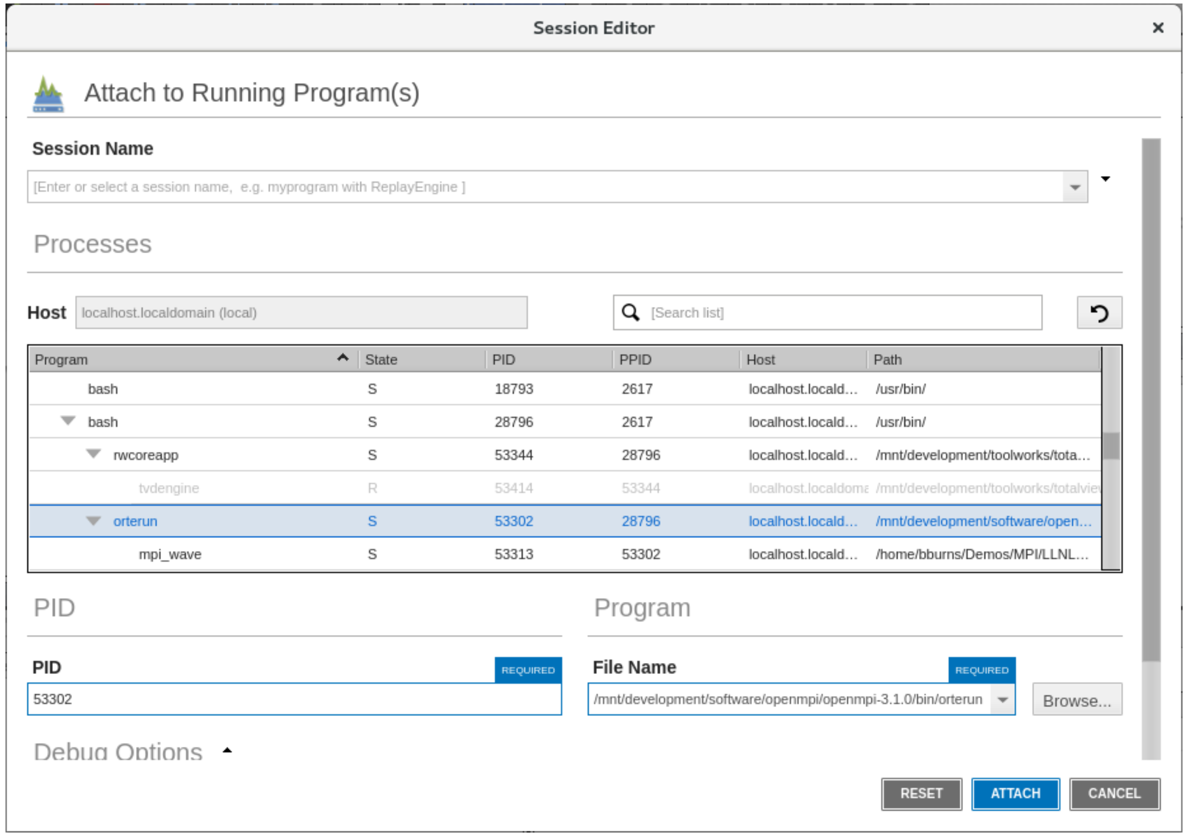


```
void * read(void *temp) {
  char *ret;
  FILE *file1;
  char *str;
  pthread_mutex_lock(&read_mutex);
  sleep(5);
  pthread_mutex_lock(&write_mutex);
  printf("\n Opening file \n");
  file1=fopen("temp","r");
  str=(char *)malloc(10*sizeof(c
  fscanf(file1,"%s",str);
  printf("\n Message from file i

  fclose(file1);

  pthread_mutex_unlock(&write_mu
  pthread_mutex_unlock(&read_mut
```

Name	Thread ID	Value
read_mutex	1.2	(pthread_mutex_t)
__data	1.2	(struct __pthread_mutex_s)
__lock	1.2	0x00000001 (1)
__count	1.2	0x00000000 (0)
__owner	1.2	0x00000000 (0)

Attach and Detach From a Parallel Job

- Peek at the state of your parallel job
- Use TotalView's attach and detach capabilities to examine the job and then let it continue to run
- Attaching to starter process enables TotalView to discover and attach to all (or a subset) of the ranks



Session Editor

Attach to Running Program(s)

Session Name

[Enter or select a session name, e.g. myprogram with ReplayEngine]

Processes

Host localhost.localdomain (local)

[Search list]

Program	State	PID	PPID	Host	Path
bash	S	18793	2617	localhost.locald...	/usr/bin/
bash	S	28796	2617	localhost.locald...	/usr/bin/
rwcoreapp	S	53344	28796	localhost.locald...	/mnt/development/toolworks/tota...
tvdengine	R	53414	53344	localhost.localdom	/mnt/development/toolworks/totalvie
orterun	S	53302	28796	localhost.locald...	/mnt/development/software/open...
mpi_wave	S	53313	53302	localhost.locald...	/home/bburns/Demos/MPI/LLNL...

PID

PID REQUIRED

53302

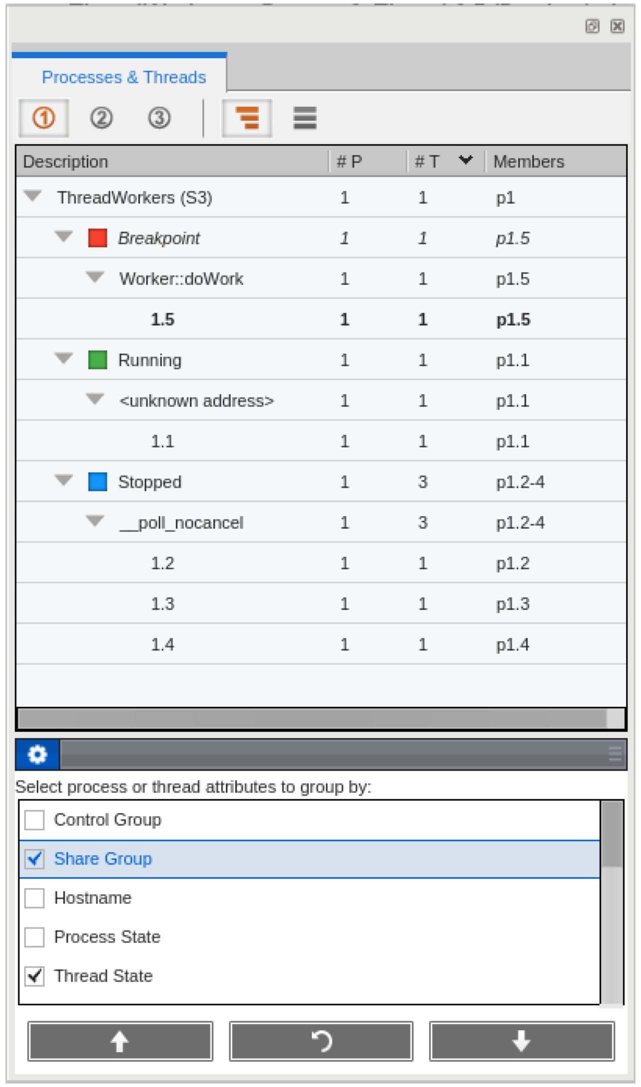
Program

File Name REQUIRED

/mnt/development/software/openmpi/openmpi-3.1.0/bin/orterun

Process/Thread Aggregation

- Aggregate process and thread state to quickly understand the state of the job
- Find outliers quickly
- Views allow different configuration to be easily switched



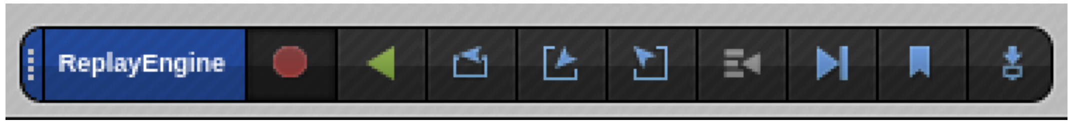
Description	# P	# T	Members
ThreadWorkers (S3)	1	1	p1
🟥 *Breakpoint*	*1*	*1*	*p1.5*
Worker::doWork	1	1	p1.5
1.5	**1**	**1**	**p1.5**
🟩 Running	1	1	p1.1
<unknown address>	1	1	p1.1
1.1	1	1	p1.1
🟦 Stopped	1	3	p1.2-4
__poll_nocancel	1	3	p1.2-4
1.2	1	1	p1.2
1.3	1	1	p1.3
1.4	1	1	p1.4

Select process or thread attributes to group by:

- [ ] Control Group
- [✓] Share Group
- [ ] Hostname
- [ ] Process State
- [✓] Thread State

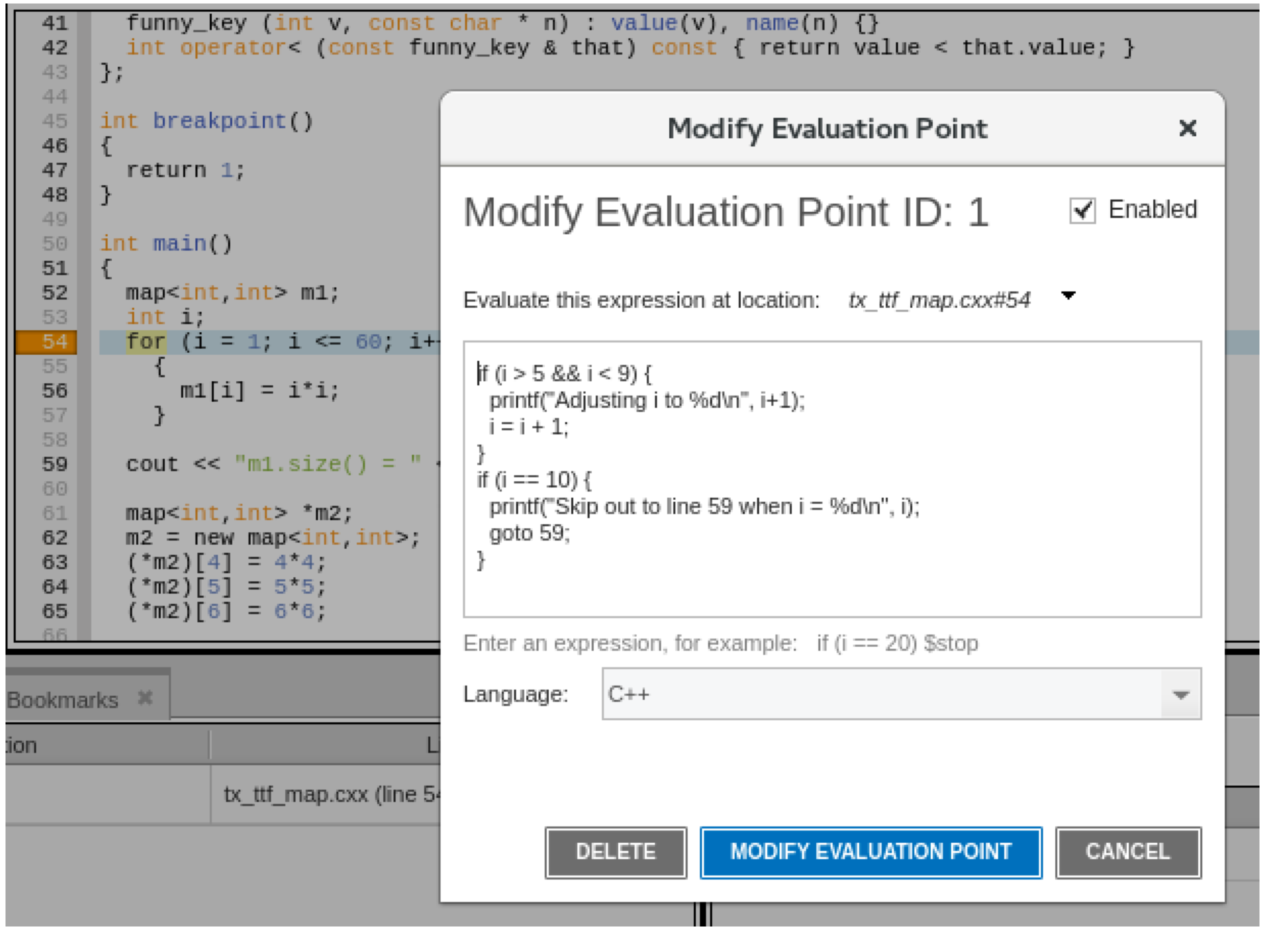
Record Only What You Need To

- Reverse debugging can be enabled on the fly
- Adjust the size of the circular recording buffer
- Record only the processes/ranks that you need



Patch Code with Evaluation Points

- Evaluation points allow a segment of code to be run at a line number
- Patch code on the fly
- Use special directives such as $stopthread and $stopprocess to control threads and processes



```
41     funny_key (int v, const char * n) : value(v), name(n) {}
42     int operator< (const funny_key & that) const { return value < that.value; }
43   };
44
45   int breakpoint()
46   {
47     return 1;
48   }
49
50   int main()
51   {
52     map<int,int> m1;
53     int i;
54     for (i = 1; i <= 60; i+
55       {
56         m1[i] = i*i;
57       }
58
59     cout << "m1.size() = " 
60
61     map<int,int> *m2;
62     m2 = new map<int,int>;
63     (*m2)[4] = 4*4;
64     (*m2)[5] = 5*5;
65     (*m2)[6] = 6*6;
66
```

Bookmarks ✖

ion | L

tx_ttf_map.cxx (line 5

Modify Evaluation Point ×

Modify Evaluation Point ID: 1 ✓ Enabled

Evaluate this expression at location: *tx_ttf_map.cxx#54*

```
if (i > 5 && i < 9) {
  printf("Adjusting i to %d\n", i+1);
  i = i + 1;
}
if (i == 10) {
  printf("Skip out to line 59 when i = %d\n", i);
  goto 59;
}
```

Enter an expression, for example: if (i == 20) $stop

Language: C++

DELETE | MODIFY EVALUATION POINT | CANCEL

TotalView Resources and Documentation

PERFORCE

TotalView Resources and Documentation

- TotalView website:
  https://totalview.io
- TotalView documentation:
  - https://help.totalview.io
  - User Guides: Debugging, Memory Debugging and Reverse Debugging
  - Reference Guides: Using the CLI, Transformations, Running TotalView
- Blog:
  https://totalview.io/blog
- Video Tutorials:
  https://totalview.io/support/video-tutorials

TotalView Debugging Feature References

Getting Started with TotalView

- https://totalview.io/video-tutorials/getting-started-totalview

How to Use Remote User Interface Debugging

- https://totalview.io/video-tutorials/how-use-remote-user-interface-debugging

Controlling Execution with Evaluation Points

- https://totalview.io/video-tutorials/controlling-execution-evaluation-points

Reverse Debugging

- https://totalview.io/video-tutorials/reverse-debugging

Debugging Python and C++ Mixed Language Applications

- https://totalview.io/video-tutorials/debugging-python-and-c-mixed-language-applications

Debugging the Toughest Challenges with NVIDIA and AMD GPUs

- https://totalview.io/resources/debugging-toughest-challenges-nvidia-and-amd-gpus

Q&A

Questions

- Any questions or comments?
  - Don’t hesitate to reach out to me directly with any questions or comments!
  - **Email:** bburns@perforce.com

- **Thank you for your time today!**