# Our products and services

## Tools

<table>
<thead>
<tr>
<th>Klocwork</th>
<th>On-the-fly static code analysis for app security</th>
</tr>
</thead>
<tbody>
<tr>
<td>CodeDynamics</td>
<td>Commercial dynamic analysis</td>
</tr>
<tr>
<td>OpenLogic Support</td>
<td>Enterprise-grade SLA support</td>
</tr>
<tr>
<td>OpenLogic Audits</td>
<td>Detailed open source license and security risk guidance</td>
</tr>
</tbody>
</table>

## Libraries

<table>
<thead>
<tr>
<th>SourcePro</th>
<th>OS, database, network, and analysis abstraction for C++</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visualization</td>
<td>Real-time data visualization at scale</td>
</tr>
<tr>
<td>PV-WAVE</td>
<td>Visual data analysis</td>
</tr>
<tr>
<td>IMSL Numerical Libraries</td>
<td>Scalable math and statistics algorithms</td>
</tr>
<tr>
<td>HydraExpress</td>
<td>SOA/C++ modernization framework</td>
</tr>
<tr>
<td>HostAccess</td>
<td>Terminal emulation for Windows</td>
</tr>
<tr>
<td>Zend Guard</td>
<td>PHP encoding and obfuscation</td>
</tr>
</tbody>
</table>

---

Stingray MFC GUI components
How does Rogue Wave help in HPC?

TotalView for HPC

- Source code debugger for C/C++/Fortran
  - Visibility into applications
  - Control over applications
- Scalability
- Usability
- Support for HPC platforms and languages
TotalView Overview
Mid-1980’s Bolt, Berenak, and Newman (BBN) Butterfly Machine
An early ‘Massively Parallel’ computer
How do you debug a Butterfly?

- TotalView project was developed as a solution for this environment
  - Able to debug multiple processes and threads
  - Point and click interface
  - Multiple and Mixed Language Support
- Core development group has been there from the beginning and have been/are involved in defining MPI interfaces, DWARF, and lately OMPD (Open MP debugging interface)
Other capabilities added

- Support for most types of MPI
- Linux
- Lightweight Memory Debugging
- Type transformations – STL and user containers
- Memscript and tvscript
- Reverse Debugging - only on Linux x86-64
- Remote Display Client
- GPU debugging
- Intel Xeon Phi – Including KNL
- Currently looking at ARM64 port
Key features of TotalView

• Interactive Debugging
• Interactive Memory Debugging
• Reverse Debugging
• Unattended Debugging

Serial, Parallel and Accelerated applications
Memory Analysis

How do you identify buffer overflows?

Runtime Memory Analysis : Eliminate Memory Errors

- Detects memory leaks *before* they are a problem
- Explore heap memory usage

Features

- Detects
  - Malloc API misuse
  - Memory leaks
  - Buffer overflows
- Low runtime overhead
- Easy to use
  - Works with vendor libraries
  - No recompilation
  - No instrumentation
Reverse debugging

- How do you isolate an intermittent failure?
  - Without TotalView,
    - Set a breakpoint in code
    - Realize you ran past the problem
    - Re-load
    - Set breakpoint earlier
    - Hope it fails
    - Keep repeating
  - With TotalView
    - Start recording
    - Set a breakpoint
    - See failure
    - Run backwards/forwards in context of failing execution

- Reverse Debugging
  - Re-creates the context when going backwards
  - Focus down to a specific problem area easily
  - Saves days in recreating a failure
memscript and tvscript

- Command line invocation to run TotalView and Memoryscape unattended
- tvscript can be used to set breakpoints, take actions at those breakpoints and have the results logged to a file. It can also do memory debugging
  - tvscript -create_actionpoint “method1=>display_backtrace show_arguments” \ -create_actionpoint “method.c#342=>print x” myprog -a dataset 1
- memscript can be used to run memory debugging on processes and display data when a memory event takes place. Exit is ALWAYS an event
  
  Memscript -event_action \
  "alloc_null=list_allocations,any_event=check_guard_blocks” \ 
  -guard_blocks -maxruntime "00:30:00” -display_specifiers \ 
  "noshow_pc,noshow_block_address,show_image”\ 
  myProgram -a myProgramArg1
- Memscript data can be saved in html, memory debug file, text heap status file
Remote Display Client (RDC)

- Push X11 bits and events across wide networks can be painful. The RDC can help
The RDC setup

1. Enter the Remote Host to run your debug session:
   - Remote Host: vesta.alcf.anl.gov
   - User Name: thompson

2. As needed, enter hosts in access order to reach the Remote Host:
   - Host 1: Access By: User Name
   - Host 2: Access By: User Name

3. Enter settings for the debug session on the Remote Host:
   - Path to TotalView on Remote Host: /soft/debuggers/totalview/bin/totalview
   - Arguments for TotalView:
   - Your Executable (path & name): runjob
   - Arguments for Your Executable: -p 1 --np 512 --block ${COBALT_PARTNAME}:ALLc2
   - Submit Job to Batch Queueing System: Custom

4. Enter batch submission settings for the Remote Host:
   - Submit Command: qsub
   - Script to execute via Submit Command: ./tv_PBS.csh
   - Additional Submit Command Options: -q ATPESC2015 -t 60 -n 512 --mode script --O LOG
TotalView for the NVIDIA® GPU Accelerator

- NVIDIA CUDA 6.5, 7.0, 7.5, (8.0)
- 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
- CUDA device window reveals what is running where
- Support for CUDA Core debugging
- Leverages CUDA memcheck
- Support for OpenACC
Supports All Major Intel Xeon Phi Coprocessor Configurations

- Native Mode
  - With or without MPI
- Offload Directives
  - Incremental adoption, similar to GPU
- Symmetric Mode
  - Host and Coprocessor
- Multi-device, Multi-node
- Clusters
- KNL Support – Just works like a normal node
  - AVX2 support being added

User Interface

- MPI Debugging Features
  - Process Control, View Across, Shared Breakpoints
- Heterogeneous Debugging
  - Debug Both Xeon and Intel Xeon Phi Processes

Memory Debugging

- Both native and symmetric mode
Knights Landing Memory

- KNL will have on-board High Bandwidth Memory (HBM) which can be accessed much faster than going out to main memory.
  - Cache
  - Explicitly managed for placement of frequently accessed data

- MemoryScape will be able to track allocations made both the standard heap and the on-chip HBM

- Optimization may include making sure that the right data structures are available to the processor in HBM
  - MemoryScape can show you data structure usage and placement

- KNL machines starting to come online
Linux OpenPower (LE) support

- Support for OpenPower (Linux power LE)
  - All major functionality
  - Support for CUDA Debugging on GPU Accelerators
TotalView’s Memory Efficiency

- TotalView is lightweight in the back-end (server)
- Servers don’t “steal” memory from the application
- Each server is a multi-process debugger agent
  - One server can debug thousands of processes
  - Not a conglomeration of single process debuggers
  - TotalView’s architecture provides flexibility (e.g., P/SVR)
  - No artificial limits to accommodate the debugger (e.g., BG/Q 1 P/CN)
- Symbols are read, stored, and shared in the front-end (client)
- Example: LLNL APP ADB, 920 shlibs, Linux, 64 P, 4 CN, 16 P/CN, 1 SVR/CN

<table>
<thead>
<tr>
<th>Process</th>
<th>VSZ (largest, MB)</th>
<th>RSS (largest, MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV Client</td>
<td>4,469</td>
<td>3,998</td>
</tr>
<tr>
<td>MRNet CP</td>
<td>497</td>
<td>4</td>
</tr>
<tr>
<td>TV Server</td>
<td>304</td>
<td>53</td>
</tr>
</tbody>
</table>
New UI Framework – aka CodeDynamics
For HPC we have two methods to start the debugger

The ‘classic’ method

- totalview –args mpiexec –np 512 ./myMPIprog myarg1 myarg2
- This will start up TotalView on the parallel starter (mpiexec, srun, runjob, etc) and when you hit ‘Go’ the job will start up and the processes will be automatically attached. At that point you will see your source and can set breakpoints.

• Some points to consider…
  - You don’t see your source at first, since we’re ‘debugging’ the mpi starter
  - Some MPI’s don’t support the process acquisition method (most do, but might be stripped of symbols we need when packaging)
  - In general more scalable than the next method...
Starting TotalView

The ‘indirect’ method
• Simply ‘totalview’ or ‘totalview myMPIprog’ and then you can choose a parallel system, number of tasks, nodes, and arguments to the program.

• With this method the program source is available immediately
• Less dependent on MPI starter symbols
• May not be as scalable as some ‘indirect’ methods launch a debug server per process
The New UI for HPC

MPI debugging with the new UI requires starting in ‘classic’ mode with the –newUI argument:
totalview –newUI –args mpiexec –np 4 ./cpi
Starting MemoryScape and ReplayEngine

• MemoryScape must be enabled at the start of the program or your program can be linked against the Heap Interposition Agent (HIA) to have memory debugging always enabled. It can’t be turned on when the program has been started.

• ReplayEngine can be enabled at the start, or at any point during debugging. Once started, it can’t be disabled.
Running TotalView on Vesta
In order to use memory debugging on the BG/Q, you must link against the agent. For instance, I first set up an environment variable:

- export TVLIB=/soft/debugger/totalview/linux-power/lib
- mpixlc_r -g -o ALLc2.mem ALLc2.c -L$TVLIB \n- -Wl,@$TVLIB/tvheap_bgqs.ld

Memory Debugging Options
Select your preferred level of debugging below or press Advanced Options for more control.

- Enable memory debugging

Levels of Debugging

- Low
  Provides event notifications and leak detection. It allows the best performance for your process.

- Medium
  Adds corrupted memory detection by applying guard blocks. Performance may degrade slightly.

- High
  Provides memory over run alerts by monitoring Red Zone violations. Your memory consumption will increase significantly.

- Extreme
  Enables all options. There is a risk that performance may suffer and you will use more memory.
One of many reports available
More Information

- Product documentation on website:
  http://www.roguewave.com/help-support/documentation/totalview
- Contact sales@roguewave.com with any inquiries about our future plans with regard to TotalView product.
Questions
Thanks!

- Visit the website
  - Documentation
  - Sign up for an evaluation
  - Contact customer support & post on the user forum