Debugging Tools

Additional Tools for Debugging in Hard Situations

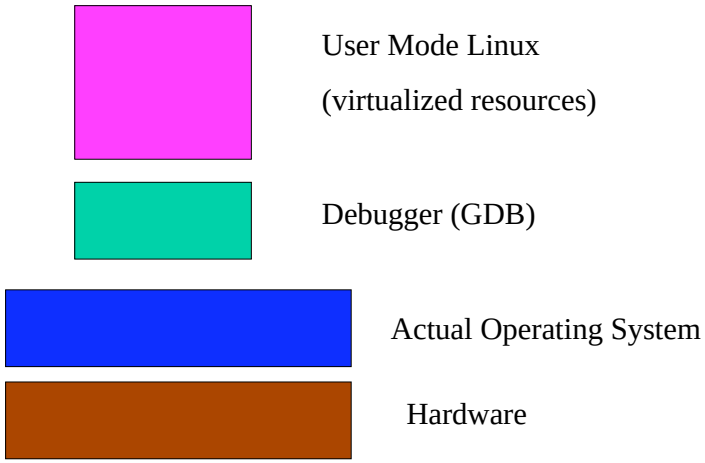
Guest Lecture by Brad Morrey

December 3, 2004

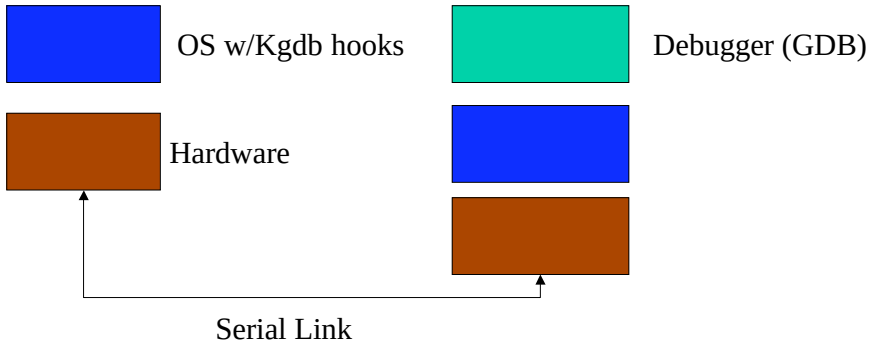
Kernel Debugging?

- Operating System sits on the Hardware
  - How to debug?
- User-Level Operating Systems
  - UML (http://user-mode-linux.sourceforge.net)
  - Not maintained
- Kernel Debuggers
  - Kgdb (http://oss.sgi.com/projects/kgdb)

User Mode Linux



Kgdb



Memory Leaks

- Operating System doesn't track memory operations during program execution
  - With C/C++ this is for speed
  - With Java the runtime does track memory but doesn't clear old objects still referenced
- Debugger doesn't track memory operations
  - It could, but this would further slow program execution
- Need a tool
  - Purify – IBM commercial tool
  - Valgrind (pronounced valgrinned)

GPL memory debugging and profiling tool

- Uninitialized memory use
- Using memory after freeing
- Access off the ends of malloc'd blocks
- Accesses to the stack
- Memory leaks
- Function call argument checking
- Mismatched use of malloc/free/new/delete

Valgrind details

- Intercepts calls to system memory allocation routines
- Tracks loads/stores
- Runs with binaries
- Reports errors as they occur with line number and stack trace

Limitations

- Works best with dynamically linked executables
- Has trouble with STL (STL uses its own allocation routines)
- Requires debugging information
- Can't handle bounds checking on static or stack allocated arrays

Conclusion

- Tools exist to make the programmer’s job easier
- None is a silver bullet
- But many can improve productivity when used in the proper circumstances