Comparisons of Underwater Acoustic Network Protocol Stacks

Yibo Zhu, Lina Pu, Zigeng Wang, Xiaoyan Lu, Robert Martin, Yu Luo, **Zheng Peng**, and Jun-Hong Cui

Underwater Sensor Network Lab
University of Connecticut

Objectives

- To answer the following questions:
  - Types of protocol stacks?
  - Representative protocol stacks?
  - Differences between them?
  - How to evaluate their performance?

Outline

- Introduction
- Underwater protocol stacks
- Performance criteria
- Case study
- Conclusions

Underwater protocol stack

- A computer networking software platform
  - Accommodate a number of networking protocols
  - Provide interfaces for different networking layers
  - Similar to Internet protocol stack

- Consider restrictions and challenges to underwater system design
  - Limited computing resources
    - Memory-constrained
    - Less powerful CPU
  - Battery-powered

Representative platforms for underwater networks

- Simulator based platform
  - DESERT, SUNSET
- OS-based platform
  - SeaLinx
- AF-based platform
  - UnetStack

Representative platforms for underwater networks

- Simulator based platform
  - DESERT, SUNSET
- OS-based platform
  - SeaLinx
- AF-based platform
  - UnetStack

Problem: how to evaluate the performance of underwater network platform?

Performance Criteria

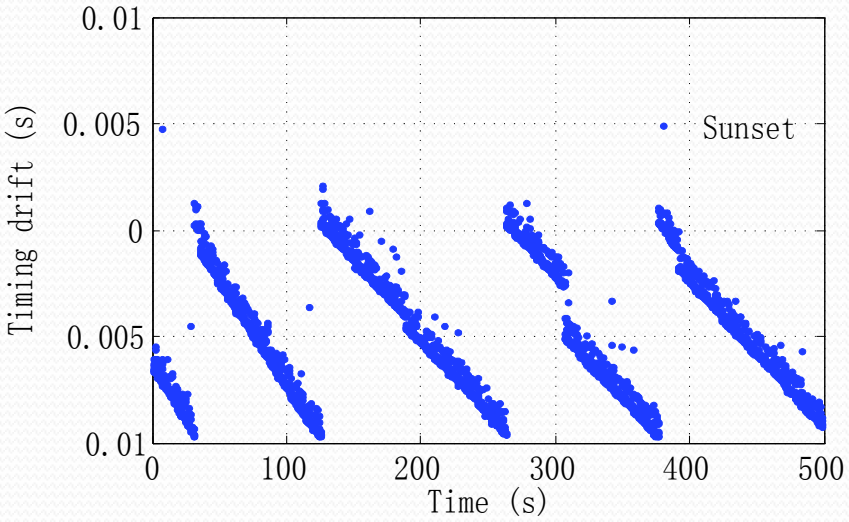
- Timing accuracy
  - Timing accuracy affects system performance
- Memory usage
  - Underwater nodes have limited memory
- Power consumption
  - Underwater nodes are powered by battery
- Support of simulation and emulation
  - Provide seamless transitions from simulations to field tests
- Learning curve
  - Reduce the development cycle

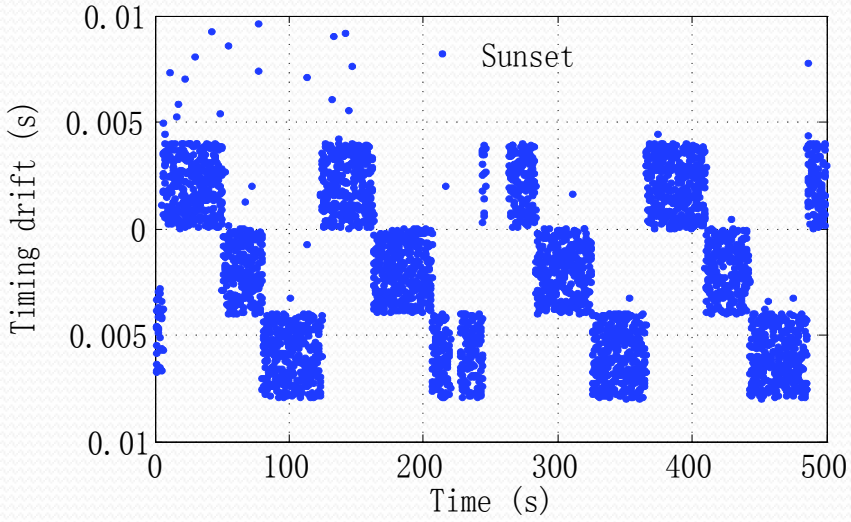
Comparing SUNSET and SeaLinx

- SUNSET
  - Based on the architecture of ns-2, with enhanced real-time scheduler and new I/O related modules
  - Support both simulation and emulation
  - A similar timing scheme as in ns-2
- SeaLinx
  - An OS-based protocol stack, built from scratch.
  - Support both simulation (SeaLinx-Mate) and emulation

Behavior of SUNSET real-time scheduler (1)

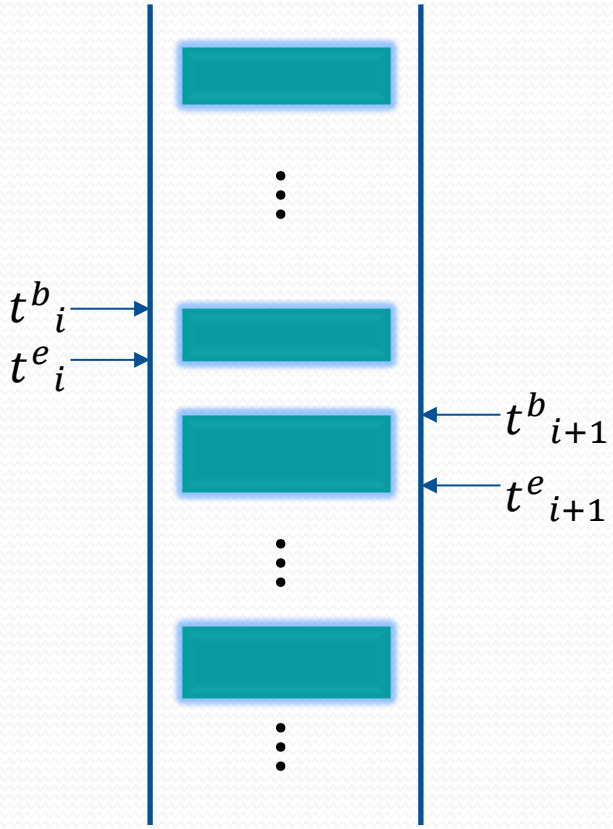
Drift of non-overlapping events' start time in SUNSET



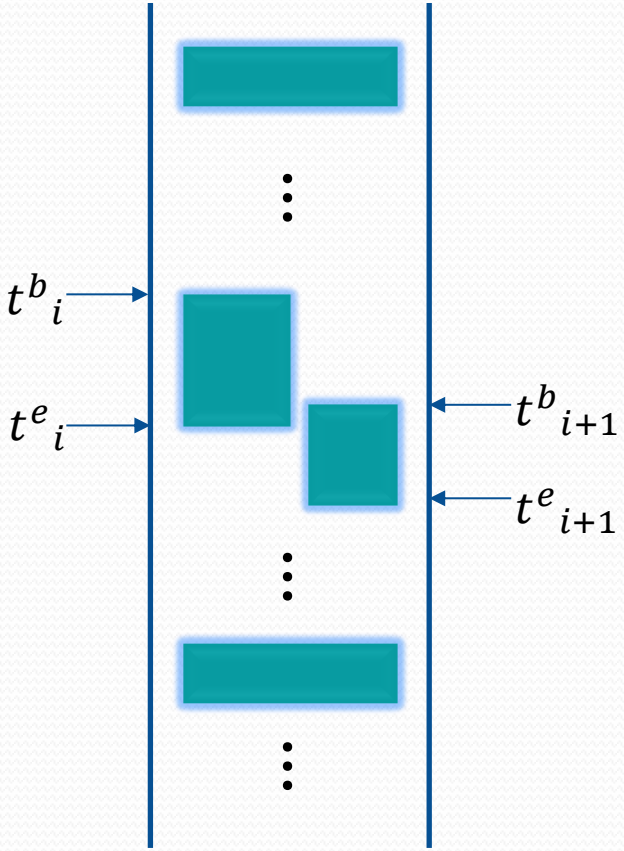


- Sunset's real-time scheduler may have adopted a special method to decide when to invoke events

Behavior of SUNSET real-time scheduler (2)

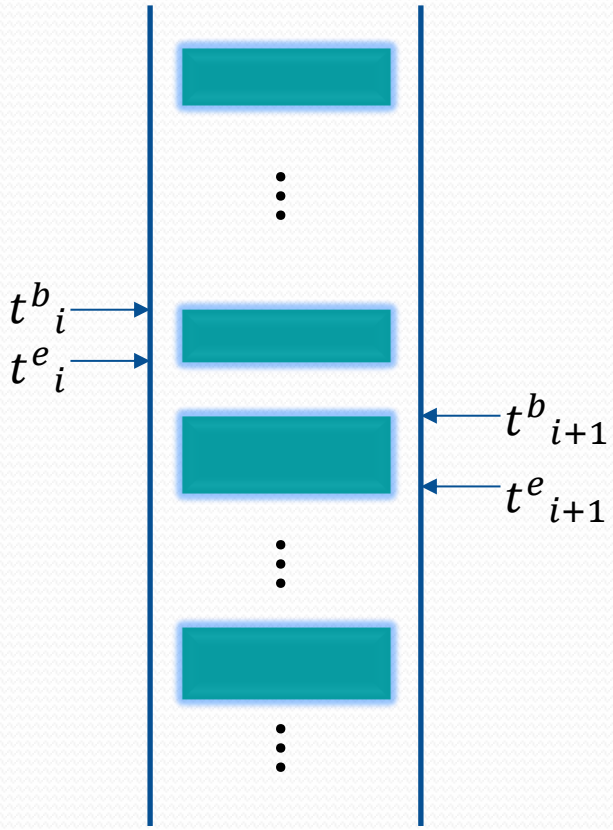


Non-overlapping event

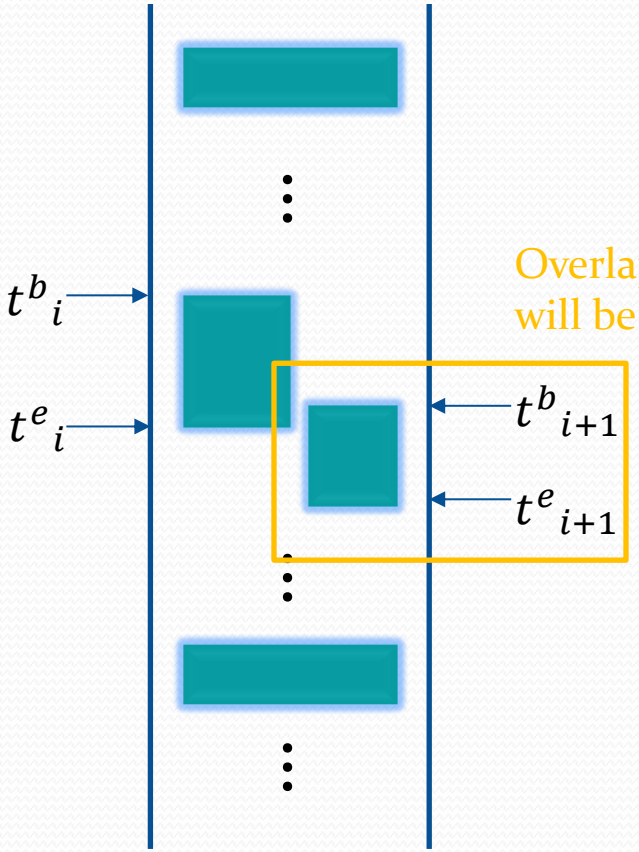


Overlapping event

Behavior of SUNSET real-time scheduler (2)



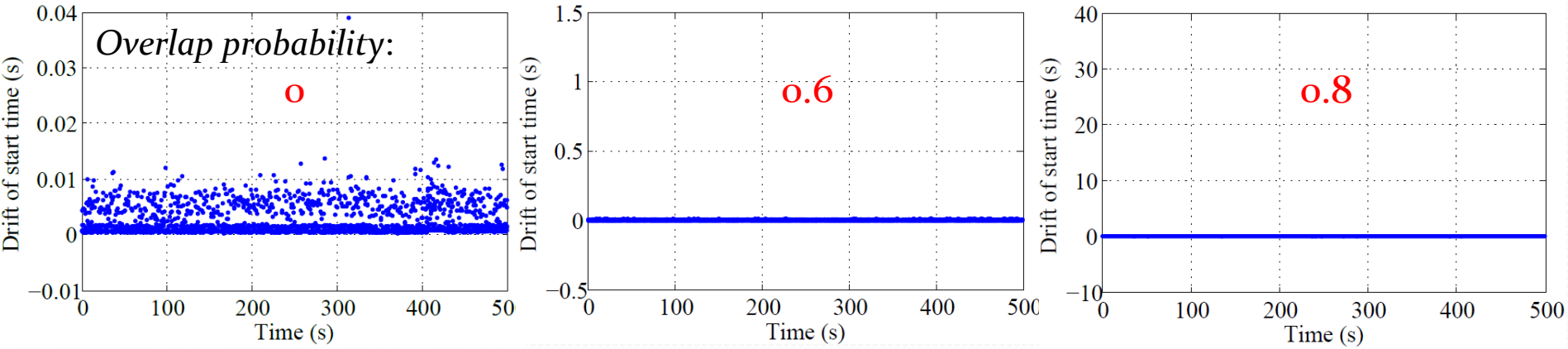
Non-overlapping event



Overlapping event

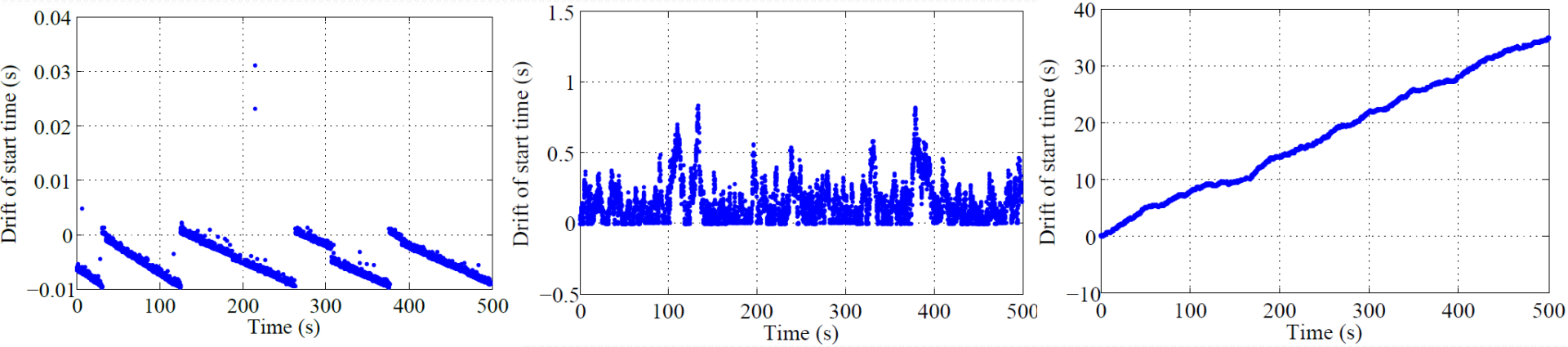
Handling overlapping events

SeaLinx



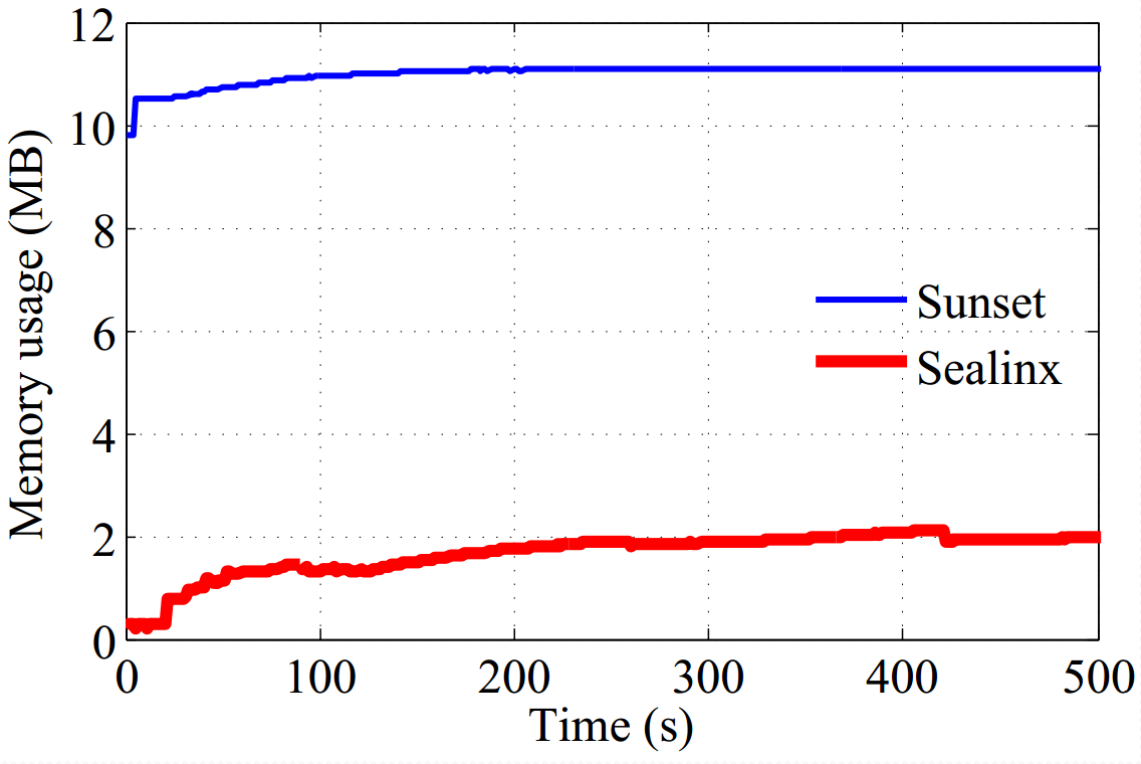
The actual start time of overlapping events has a small and stable drift in SeaLinx

SUNSET



The actual start time of overlapping events could be postponed in SUNSET

Memory usage



- OS-based protocol stack has small memory footprint
- Simulator-based protocol stack can have high memory usage

Platform comparison: OS-Based V.S. simulator-based

	OS-Based (SeaLinx)	Simulator-Based (SUNSET)
Time drift	Low, depends on OS system timer	High, due to discrete event queue & single threaded
Memory usage	Very low base memory usage, several KB	High base memory usage, about 10MB
Stack feature	Multiple process and multi-threaded	single-threaded
Support of Simulation and emulation	Yes. By using a dedicated simulator.	Yes. It has both simulation mode and emulation mode.
Result consistency	Yes, consistent	No, due to the time drift in simulation mode

Platform comparison: OS-Based V.S. simulator-based (cont.)

	OS-Based (SeaLinx)	Simulator-Based (SUNSET)
Code reuse	Yes	Yes, but may cause problems if NS2 centralized modules are used
Simulation time	Actual time	Different from actual time
Modem support	Currently limited, but can support other modems	Currently support a number of popular acoustic modems
Remote control	Yes	Yes
Learning curve	General Linux programming	Need to master NS-2 framework first

Conclusions

- Discussed different types of networking platforms
- Proposed several performance criteria
- Evaluated two representative protocol stacks as a case study

Thank you!