SCTP User Message Interleaving Integration and Validation

Felix Weinrank
Michael Tüxen
Irene Rüngeler
Erwin P. Rathgeb

Fachhochschule
Münster University of
Applied Sciences

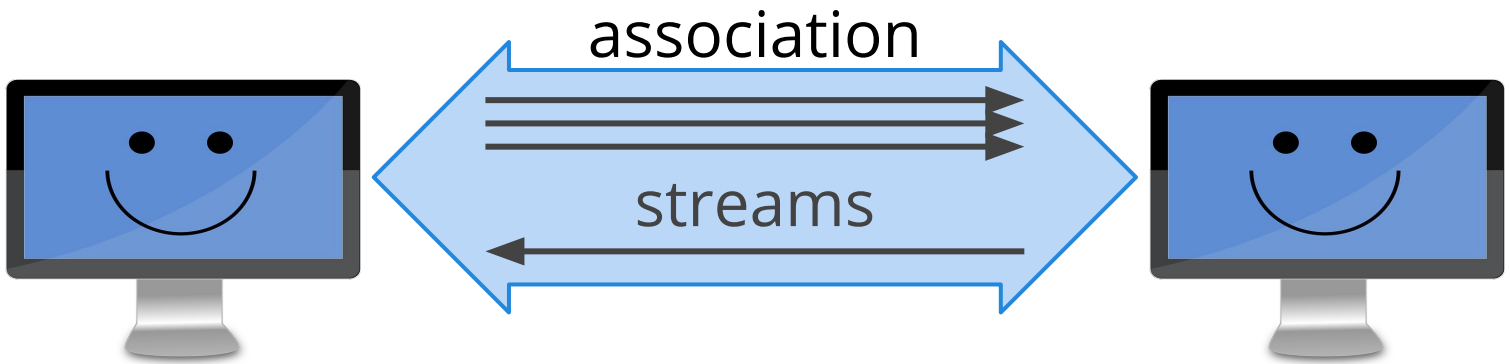
Outline

- Brief introduction to SCTP
- Message interleaving and stream scheduler
- Integration and validation
- Measurements and results
- Outlook and future work

SCTP Overview

Stream Control Transmission Protocol

- Layer 4 protocol like TCP/UDP
- Message oriented and multihomed
- Originally designed for small messages
- Used for WebRTC data channels



Interleaving and Scheduling

Interleaving - why?

Sender side Head-of-line Blocking

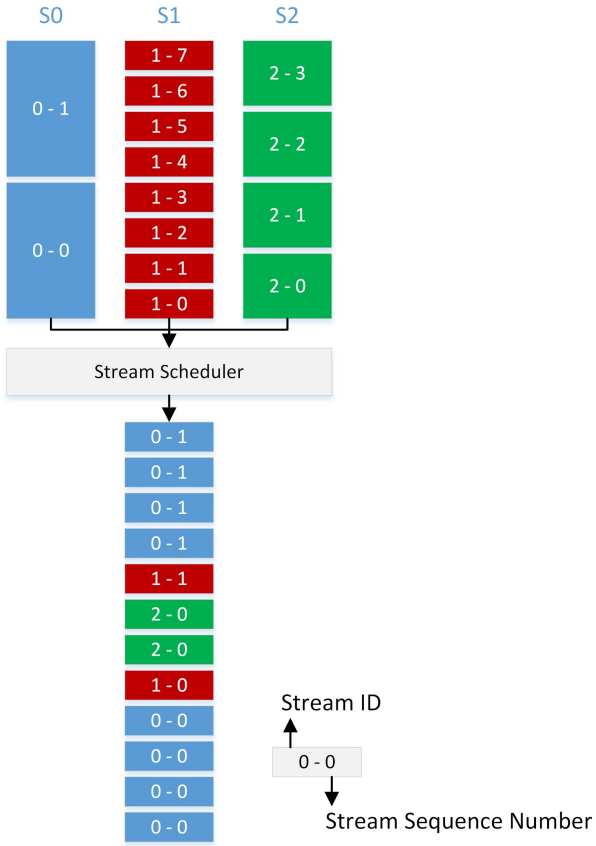
- A large SCTP user message blocks all other messages in any stream until completely sent

Message interleaving

- Reduces Head-of-line Blocking
- Specified by IETF draft *

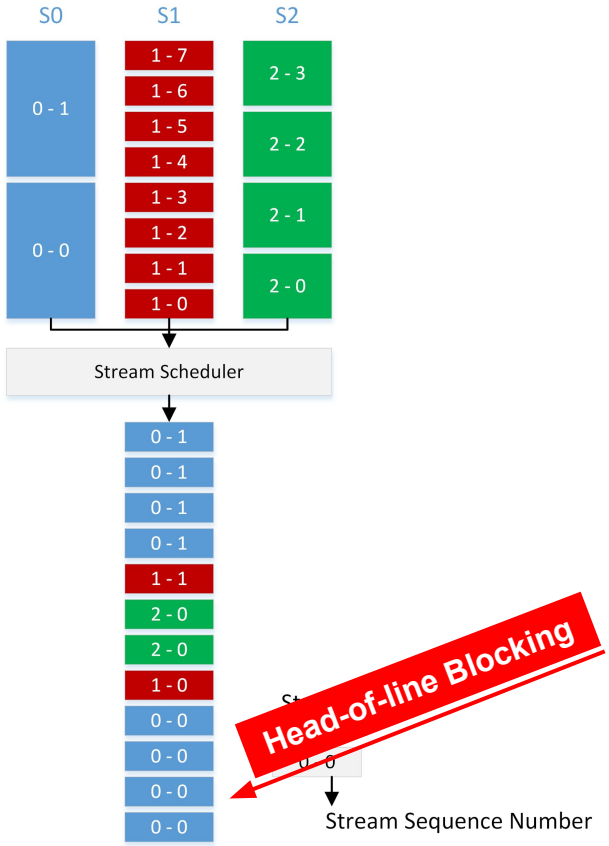
*https://tools.ietf.org/html/draft-ietf-tsvwg-sctp-ndata-07

Data transfer - non-interleaving



1. Stream scheduler selects stream
2. Optional message fragmentation
3. Stream scheduler keeps locked on stream until all fragments of a single message have been sent

Data transfer - non-interleaving

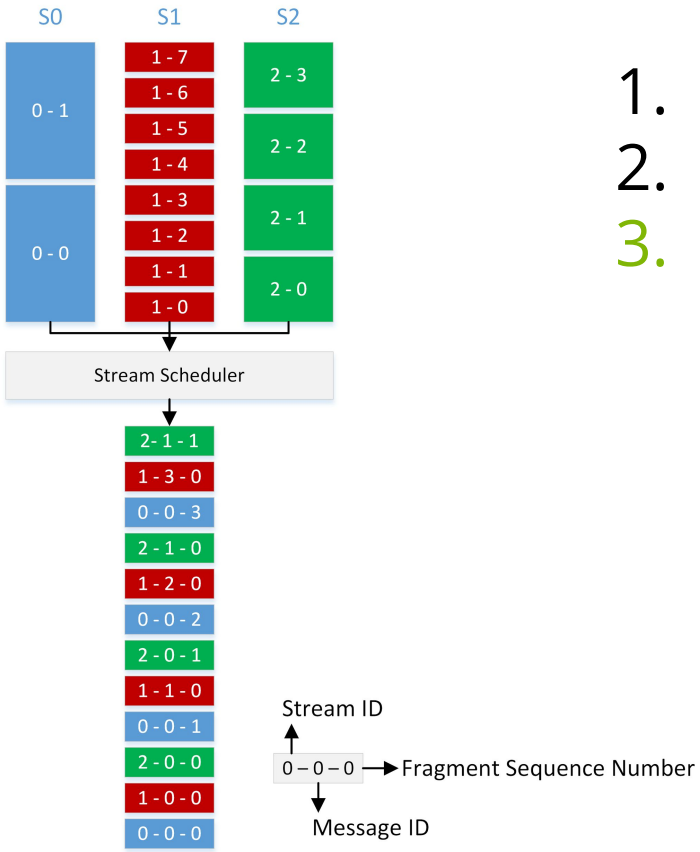


1. Stream scheduler selects stream
2. Optional message fragmentation
3. Stream scheduler keeps locked on stream until all fragments of a single message have been sent

Example: WebRTC chat application

- File transfer blocks chat messages

Data transfer - interleaving



1. Stream scheduler selects stream
2. Optional message fragmentation
3. Stream scheduler selects next stream

Integration and Validation

Integration

- Integration follows IETF draft
- `iData` parameter enables interleaving support
- Interleaving is used if both peers announce the extension support in the 4-way-handshake

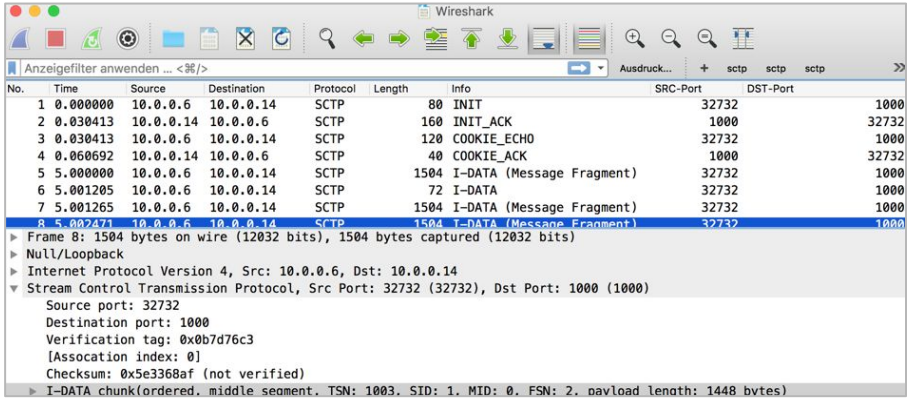
```
double fairStop @unit(s) = default(0s);
string streamsToPaths = default("");
int startEndToEndDelay = default(0);
int stopEndToEndDelay = default(0);
double throughputInterval = default(1);
bool iData = default(false);    // announce support for interleaving (iDATA)

//#====== Chunk Authentication ========================================
bool auth = default(false);
string chunks = default("");
bool padding = default(false);
```

Validation

Wireshark

- Packet flow inspection
- I-Data support added



Packetdrill

- Script based testing tool for transport protocols
- Currently more than 120 interleaving specific tests
- Same tests for OMNeT++/INET and FreeBSD

Validation

External Interface

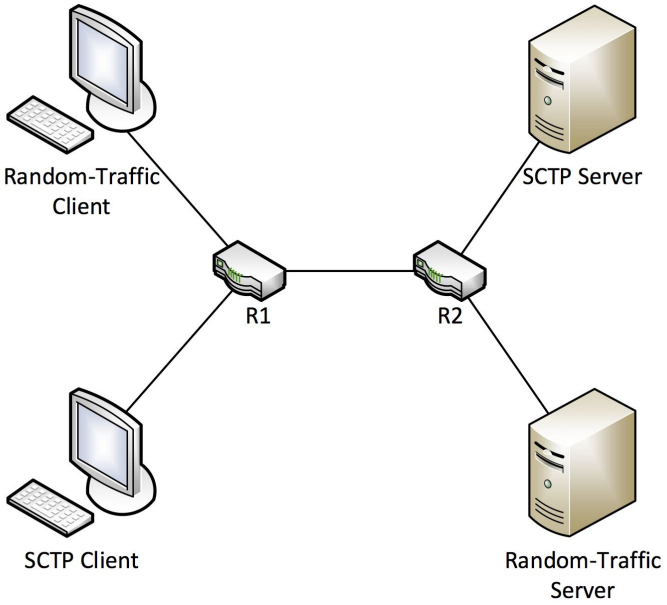
- Interoperability tests between FreeBSD’s SCTP implementation and OMNeT++/INET model

FreeBSD®

Measurements - scenario

Bottleneck scenario

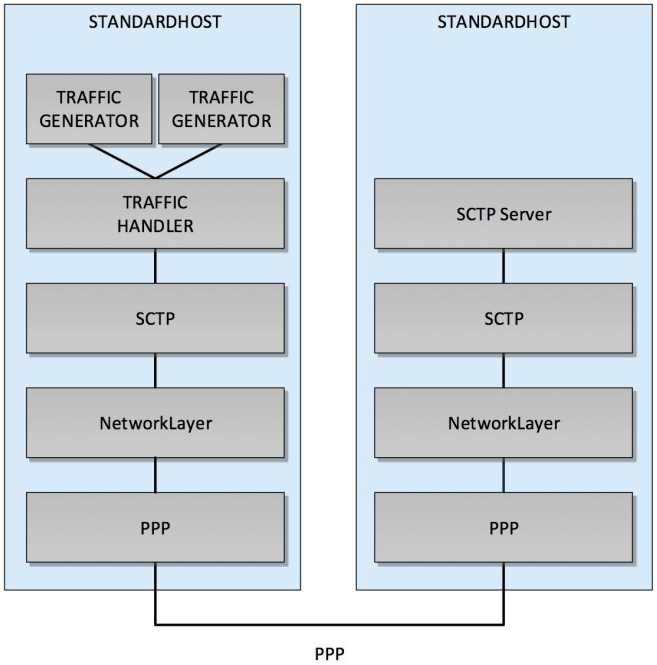
- SCTP server and client
- Random UDP background traffic



Measurements - scenario

Two competing streams

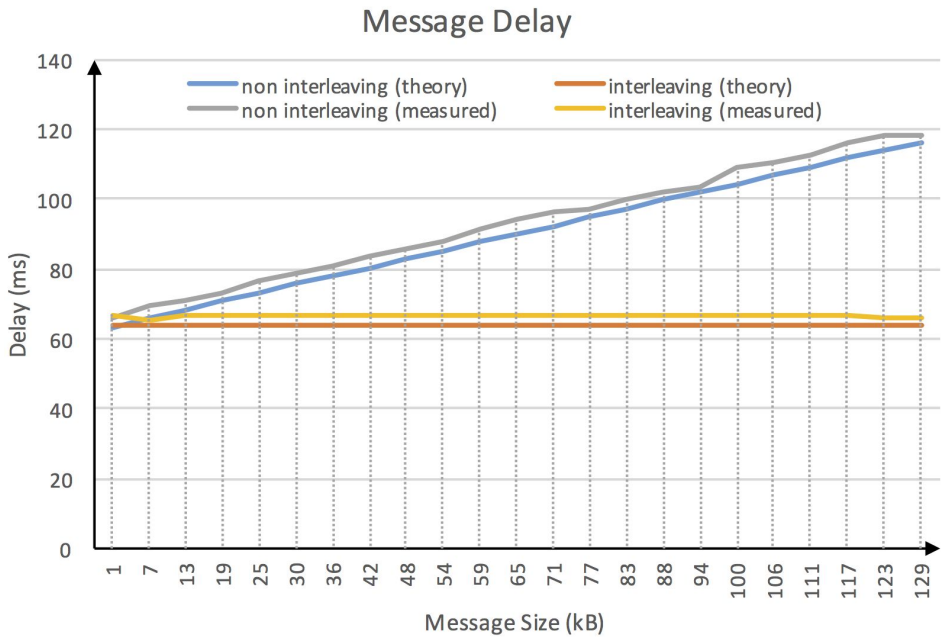
- Stream 1
  - Saturated
  - Large messages (1 - 128kB)
  - Low priority
- Stream 2
  - Unsaturated
  - Small messages (8 - 16B)
  - High priority



Measurements - results

$$d_n(s_{\text{msg}}) = \frac{s_{\text{msg}} + s_{\text{hdr}} \cdot \left\lceil \frac{s_{\text{msg}}}{s_{\text{frag}}} \right\rceil}{2 \cdot \text{bw}} + d_{\text{link}} + d_{\text{buffer}}$$

$$d_i(s_{\text{msg}}) = \frac{\text{mtu}}{2 \cdot \text{bw}} + d_{\text{link}} + d_{\text{buffer}}$$



Conclusion and Outlook

Conclusion

Conclusion

- Message interleaving reduces head-of-line-blocking for fragmented messages
- Wireshark, Packetdrill and the external interface are great tools to validate protocol operation

Outlook

- Buffering improvements
- New stream schedulers (e.g. weighted-fair-queueing)