AES67

with pipewire

15.01.2025

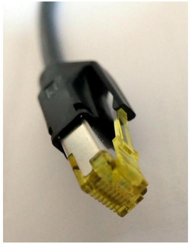
Florian “Florob” Zeitz

Outline

1 Network Audio

Motivation

→



Benefits

- transmit large amounts of channels in a tree topology
- bidirectional
- flexible channel routing
  - multiple simultaneous sinks
  - rerouting in software
- consolidated clocking architecture
- reuse existing cabling

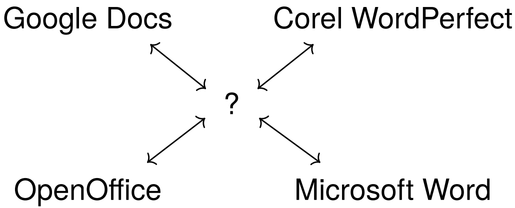
Available Solutions

- many competing protocols
- some directly layered on Ethernet, some on IP
- Audio over Ethernet (AoE)
  - CobraNet: proprietary (Cirrus Logic)
  - AVB/TSN and Milan: open standard (IEEE and Avnu)
- Audio over IP (AoIP)
  - Dante: proprietary (Audinate)
  - Q-LAN: proprietary (QSC)
  - RAVENNA: open standard* (Lawo)
  - Livewire+: open standard* (Telos Alliance)

*most documentation via partner program

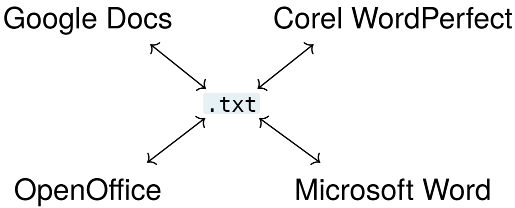
2 AES67

AES67 Is Like _ for Word



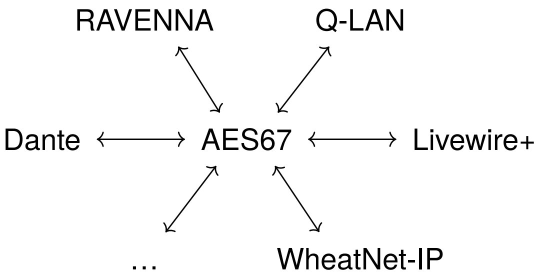
- in the early 2000s there were various word processors
- each with its own file format

AES67 Is Like .txt for Word



- raw audio can be transmitted
- no control
- no way to create connections
- multiple ways to discover streams, **none** mandatory

Motivation



- most AoIP protocols are very similar
- the Audio Engineering Society (AES) wanted to define an exchange protocol
  - use existing standards where possible
  - use AES67 between otherwise incompatible “islands”
- interoperability between
  - Dante
  - RAVENNA
  - Q-LAN
  - Livewire+
  - WheatNet-IP
  - …

Properties

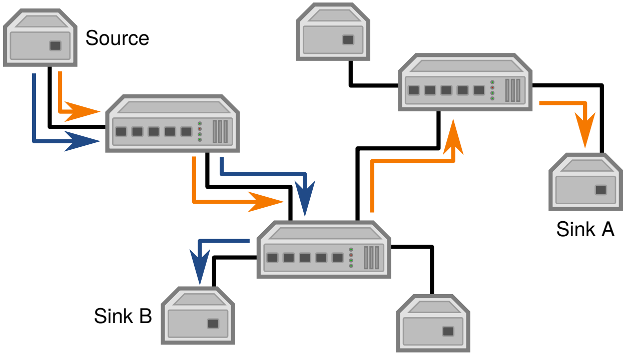
- latency between 125 µs and 4 ms
- up to 96 kHz (some implementations 48 kHz only, e. g. Dante)
- 16 bits or 24 bits
- up to 120 channels per link

Implementation Aspects

- delivery of streams
  - where/how to send the data
  - prioritization of stream delivery
- clock synchronization
  - make sure devices use the same sampling rate
  - with the same phase
- stream discovery
  - finding streams on the network

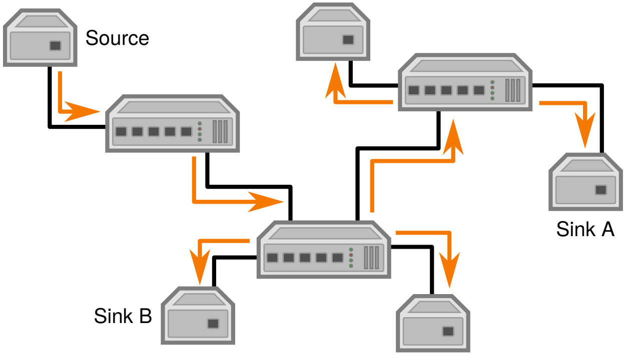
3 Stream Delivery

Unicast



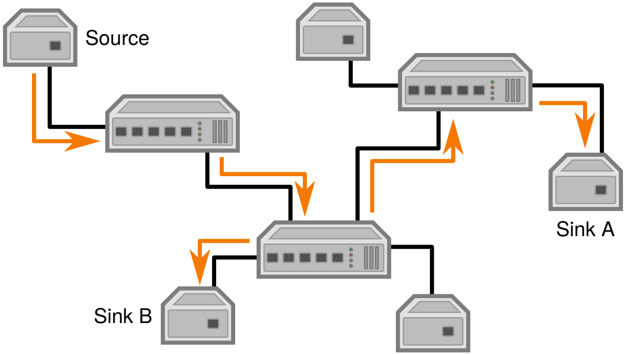
- each stream is send to each recipient separately
- requires redundant packets
- wastes bandwidth and therefore potential channels

Broadcast



- each stream is delivered to all devices on the network
- wastes bandwidth on nonparticipating links

Multicast



- each stream is delivered only to interested devices
- use multicast MAC and IP addresses
- requires support from bridges (IGMP snooping), otherwise degrades to broadcast

Real-time Transport Protocol (RTP)

Protocol used deliver audio and video over IP networks

- RFC 3550
- AES67 uses RTP to deliver audio
- can be send as unicast or multicast traffic
- includes timestamps of transmitted samples

IGMP Snooping

Internet Group Management Protocol (IGMP)

Protocol that allows hosts to request multicast traffic from adjacent routers

- multicast is primarily the concern of routers
- IGMP signals which network wants which multicast traffic
- bridges can restrict where packets are forwarded by listening in (**IGMP snooping**)
- requires a multicast router in the network
- bridges can implement an **IGMP querier** to get around this requirement

Quality of Service

- sort traffic into different queues
- can avoid streams being “starved” by other traffic
- improves latency
- avoids drops
- based on IP DiffServ field
- AES67 uses
  - Expedited Forwarding (EF, DSCP 46): PTPv2 (clock)
  - Assured Forwarding 41 (AF41, DSCP 34): RTP (audio)
- example: Luminex GigaCore Mapping Table

4 Clock Synchronization

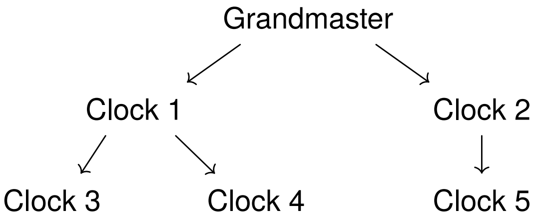
Synchronization

- establish a synchronized network clock
- network clock runs significantly faster than word clock
- word clock based on network clock
  - e. g. 48 kHz word clock ticks every ~21 µs network time
- (ideally) requires hardware support
  - adjustable hardware clock
  - timestamping of packets according to that clock

PTPv2

Precision Time Protocol (PTP)

Protocol for time synchronization in computer networks



- IEEE 1588-2002 specifies PTPv1
- IEEE 1588-2008 specifies PTPv2
- IEEE 1588-2019 specifies PTPv2.1 (backward compatible with v2)
- AES67 uses PTPv2
- transmitted as unicast or multicast
- creates a tree of clocks all synchronized to one Grandmaster
- role of each clock is automatically determined

BTCA

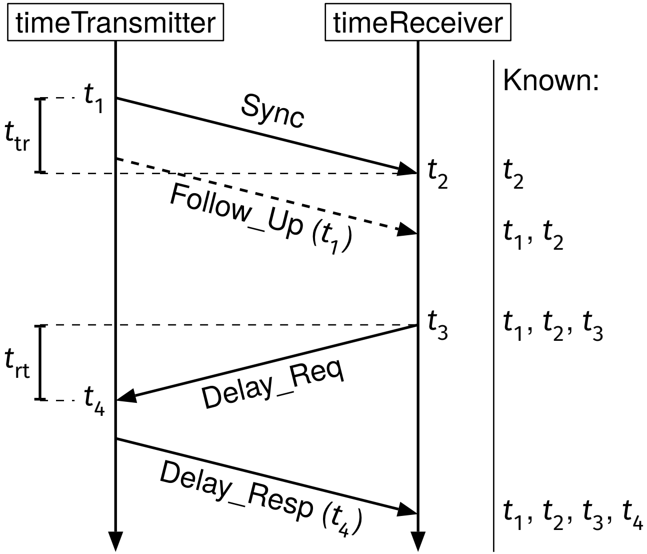
Best timeTransmitter Clock Algorithm (BTCA)

Algorithm that determines which network clock will act as Grandmaster.

Compares attributes in order to determine the best clock:

- **priority1**: user-configurable priority
- **clockClass**: type of clock, synchronized to reference (e. g. GPS), free-running, etc.
- **clockAccuracy**: accuracy of the clock
- **offsetScaledLogVariance**: stability of the clock
- **priority2**: user-configurable priority
- **clockIdentity**: tie-breaker based on unique instance identifier

PTP opperation



delay

$$\frac{(t_2-t_1)+(t_4-t_3)}{2}$$

time offset

$$(t_2 - t_1) - \text{delay} = \frac{(t_2-t_1)-(t_4-t_3)}{2}$$

- devices exchange messages timestamped with their local clocks
- allows adjusting time and frequency
- assumes symmetric link delay $t_{rt} = t_{tr}$
  - short time between messages
  - switching delays can cause issues

Clock Types

Ordinary Clock

Single network port, either Grandmaster or timeReceiver

Transparent Clock

Forwards messages, updating timestamps with the time required to transit the device.
Typically a bridge. Improves accuracy by accounting for switching delays.

Boundary Clock

Multiple network ports, one timeReceiver, others timeTransmitter.
Does not forward messages, only generates its own.
Typically a bridge or router. Improves accuracy and reduces network traffic.

Demo

- two Raspberry Pi CM4
- Broadcom Ethernet with time stamping (PTP v2.0 only)
- pulse-per-second output
  - outputs a pulse every full second
  - can be used to check quality of synchronization
- Luminex GigaCore 10 configured with different PTP modes
- check PPS phases on an oscilloscope

5 Stream Discovery

Discovery Methods

zeroconf

Set of technologies for discovering services on the local network. Avahi implements this for Linux.

Session Announcement Protocol (SAP)

Protocol for advertising multicast sessions. Transmits Session Description Protocol (SDP) descriptions of RTP sessions.

- AES67 recommends two main ways to discover streams, neither mandatory
- most implementations support only one of them or neither
- **zeroconf**: used by RAVENNA
- **SAP**: used by Dante and pipewire-aes67
- Windows tool RAV2SAP converts between them providing interoperability

6 Pipewire Setup

Hardware Requirements

Timestamping capable interface

check `ethtool -T <interface>` contains:

- `hardware-transmit`
- `hardware-receive`
- `PTP Hardware Clock: N`

if you need to buy something:

- PCIe: Intel i210
- USB: ASIX AX88279†

†Requires out-of-tree vendor module

Software Requirements

linuxptp 4.0 or newer

Open source implementation of PTPv2. Version 4.0 allows requesting current state.

Pipewire 1.1 or newer

Includes AES67 and automatically requests values from linuxptp.

linuxptp

Place the following in e. g. `aes67.conf`:

```
[global]
# Avoid becoming the Grandmaster
priority1 255
# Converge faster when time jumps
step_threshold 1

## AES67 Profile options
# Send Sync messages more often
logSyncInterval -3
# QoS
dscp_event 46
dscp_general 34
```

- run `ptp4l -mq -i <interface> -f aes67.conf`
- ensure `/dev/ptpN` is readable: udev rule

pipewire-aes67

- copy `/usr/share/pipewire/pipewire-aes67.conf` to `~/.config/pipewire/`
- edit the file according to the comments within
- minimally: `sed -i s/eth0/<interface>/ ~/.config/pipewire/pipewire-aes67.conf`
- run `pipewire-aes67`
- you should now see a stereo output stream and any stream on the network advertised via SAP
- more documentation in the Pipewire Wiki

Questions?

https://babelmonkeys.de/~florob/talks/OSAMC-2025-01-15-pipewire-aes67.pdf

Thank you for your attention. Any questions?