brian Brian Luczkiewicz Roon Labs Founder
Nov '17

I've spoken a lot about RAAT's goals before…but let me state a few of them again here.

- Create an ecosystem of certified products that meet a centrally enforced quality standard
- Provide a solution for DIY users and computer audiophiles
- Create consistency of user experience and mutual compatibility across devices from many manufacturers without requiring them to cooperate with each other
- Provide reliable audio streaming for high-resolution music listening
- Provide a control-channel for non-audio aspects like Volume Control, Convenience Switching, Standby, and other device controls.
- Reliable autodiscovery
- Stable on Ethernet and WiFi networks in the homes of real people
- Suitable for hardware devices at a wide range of cost levels
- Accomodate a large variety of existing, in-pipeline, and future products without dictating their hardware/platform choices

So the first thing to understand is–RAAT is very different in design goals and scope than the other protocols. Some of the other protocols have some of these other goals, but as a general rule, we are concerned a lot more about the end-to-end user experience than the people making the protocols you asked about–they are building infrastructure, and we are building product.

You may be thinking "but I asked about the technology" and yes I will get to that–but looking at RAAT as a streaming technology in isolation is missing important context. RAAT isn't just plumbing–it's a product that executes on the goals stated above.

From a user experience standpoint

It's almost not worth comparing RAAT with the others. They don't solve the quality assurance problem. They don't create a consistency of experience. They don't handle out-of-band concerns like volume control or standby or convenience switching without out-of-band extensions (like what Merging have done on the NADAC). They don't work on WiFi, …

On a technical level

To fully extract the benefits of AES67/DANTE/RAVENNA, you need a well-engineered ethernet network, and controlled computing environments on the sending/receiving side.

In return for that, you get low latency capabilities and extremely scalable matrix mixing. This is important for some applications, but most homes (meaning, 99.99%) are not large enough to need unlimited matrix scalability and most music listening does not require low latency. With RAAT, we've made a different set of tradeoffs. We've sacrificed some things that are not so important in return for properties that are better for our application.

AES67+Friends are streaming in real-time through a relatively small fixed-sized buffer. Maybe 1ms, maybe 500ms–pretty short either way. Packets are sent in real time. When you press "pause" you wait for the buffer to drain. When you press "play" you wait for the buffer duration to pass before hearing anything. The protocols behave sort of like a speaker cable with a built in delay.

RAAT is different. When you press play, Roon blasts a couple of seconds of audio into a buffer on the endpoint as quickly as possible. Often this can be done in just a few milliseconds, since computers and networks move faster than audio streams. Then playback starts, and Roon continues filling the endpoint buffer in a faster-than-realtime manner until it is full (currently that means 10 seconds of audio in the endpoint).

Once playback starts, the endpoint drains this buffer using its own playback clock, and Roon's job is to replace data as it is consumed. Once full, it takes a pretty large network failure to have that whole 10 second buffer drain before it can be replenished.

Of course, you wouldn't want to pipe TV Audio through a buffering scheme like this. But for music listening, streaming latency doesn't matter because we are free to pre-fetch data out of your files–and we can solve user experience latency while keeping our large buffer sizes by buffering faster than real time, so even though we have 10 seconds of buffer in the chain, playback still starts in a few hundred milliseconds or less.

Why the huge buffer? Because it makes RAAT stable on networks that were not installed by professionals, including WiFi networks.

Compared to the other protocols, RAAT is a lot more involved in the discipline of playback. When you press pause, we simply stop the buffer from flowing and retain the audio data on the endpoint until you unpause. With the others, the buffer drains on "pause" and then must be refilled when you unpause. This is all the result of RAAT being designed around music listening, and not simply for moving audio around.

RAAT tries to "tread lightlly" on the network. We use TCP to move the audio instead of UDP because that's what 99% of peoples' home network usage looks like–web pages, Netflix, Youtube, and Spotify also use TCP. So if you have a network that supports those, RAAT will probably work.

Finally, RAAT is built to evolve in place without firmware updates. This means that many of the details I described above are not actually details of RAAT, but rather details of Roon. 6 months ago, RAAT was a UDP based protocol with a 2.5 second buffer. Now it's a TCP based protocol with with a 10 second buffer. We reworked part of the "flow" of starting a new stream a couple of releases ago because some newer WiFi devices were having trouble with the old "flow". All of these changes rolled out without modifying any devices or having people wait for their manufacturers to "catch up".

These changes are delivered via Roon without device firmware updates–so in that sense, RAAT is much more of a living protocol. I have some ideas about how to improve clock synchronization on high-latency WiFi networks, and how to improve sound quality during multi-zone playback. When we find the time to work on those, those will result in changes to the protocol and the benefits will roll out uniformly to all of our devices.

In this regard, RAAT is more of a vehicle for delivering Roon's current idea of state-of-the-art audio streaming to your devices, rather than a protocol in and of itself.

In closing

RAAT is a streaming product targeted at music listening on home networks. AES67/RAVENNA/DANTE are infrastructure for moving audio around controlled/managed environments. They are really for different purposes, so they work totally differently. It's not a matter of "better" or "worse". Just different tradeoffs and different goals.

If I wanted to build ethernet-based matrix switching for a huge home that was going to be used for routing real-time sources like TV audio, I wouldn't use RAAT. If I want to stream music and internet radio to the WiFi speaker in my kitchen, the other protocols wouldn't even be an option.

From:

https://community.roonlabs.com/t/raat-vs-aoip-protocols/33573/6?u=ermos