Musical Control Through The Internet of Musical Things

Extending MIDI for Secure Transmission Over The Internet

Damien McEvoy
Prof Joseph Timoney
Dr Iain McCurdy

Abstract

The Internet of Musical Things (IoMusT) is a growing research area which brings together Music technology, Network Technology, The Internet of Things (IoT) and Interactive Systems. Large portions of the research in this area concerns itself with the transmission of audio via video call apps or similar software solutions.
The principle aim of this project is to investigate the requirements of an Internet enabled, remote network collaborative environment. However the aim is to forego sending audio directly and look at sending symbolic musical event data, such as MIDI.

00

The beginning of this research looked at the then ‘vogue’ for musical collaborations via video call apps. This scene rose from lock-down prohibiting people working in close quarters, something musicians were used to. The use of video call apps introduced large amounts of latency, band limited frequency response (creating degraded timbre of instruments or sounds), and glitches and compression of audio, as well as video.

01

Investigating symbolic music event data led to choosing MIDI for its ubiquity, longevity and supported open standard.
There have been several successful project which “did” send MIDI over the internet, but investigation of these solution highlighted the changing standard of the internet infrastructure itself since this projects were finished.

02

Research has settled on looking at RTP MIDI and its limitations. Currently RTP, also known as AppleMIDI, comes pre-install as part of Apple OS and uses Address Resolution Protocol (ARP) to connect users via

mapping Bonjour/mDNS and IP and MAC addresses. RTP-MIDI foregoes elaborate security verification by relying on access to these address on the LAN.

However, ARP is suppressed or ignored on large LAN. Also, as Internet Service Providers (ISPs) block incoming transmissions in home networks or WAN, connection outside of institutions is also unreliable.

RTP-MIDI also is hard coded to use the User Datagram Protocol (UDP). Therefore RTP-MIDI does not need to set up a prior connection to send data. Although this makes UDP “faster” than TCP, UDP does have some drawbacks in the contemporary context. The current version of AppleMIDI/RTP-MIDI fails to connect, and does not indicate as to why. Also, although RTP queues data packets for transmission, if lost UDP does not guarantee the retrieval of the packet. Finally, RTP-MIDI does not have scope for encryption or secure connections or validation of certificates.

03

Innovation on tis theme was to take an existing stable library for each stage and combine them to create a prototype client server model.
There are compatibility issues with c++ and contemporary Mac OS versions (deprecated libraries, missing linked libraries etc).
The research has settled upon python as the coding language and created a stable Conda environment to experiment.

04

The prototype framework is as follows:
The Mido library will handle receiving the MIDI messages data stream coming from software or hardware routed to the client.
The data then will be encrypted using Advanced encryption Standard (AES) in CTR (Counter) Mode.
This data is then wrapped in an RTP packet (NOT RTP-MIDI).
A secure connection is made using SSL validation.
Data is transmitted via TCP.

This allows us to build on the benefits of:
MIDI - symbolic musical data, time stamped, ubiquitous
Encryption - needed for privacy on the contemporary internet
RTP - queue data for transmission and receiving.
SSL - Security certificate validation
TCP - Packet Data Loss retrieval.

05

Next steps are to move the server code to an Azure cloud server space and SSH into that server and test the certificate validation. Ideally, an instance of Csound should run on the server to play the MIDI events received. Then to evolve the code to be more robust and move to SSL/TCP. Finally, transmit to a receiving client machine with the server in between.