More Transport, Please!

Kory Draughn
Software Developer
iRODS Consortium

June 9-12, 2020
iRODS User Group Meeting 2020
Virtual Event

The Interface

```
namespace irods::experimental::io
{
    template <typename CharT, typename Traits>
    class transport
    {
    public:
        virtual ~transport();

        virtual auto open(path, openmode) -> bool;

        virtual auto open(path, replica_number, openmode) -> bool;

        virtual auto open(path, resource_name, openmode) -> bool;

        virtual auto close() -> bool;

        virtual auto send(buffer, buffer_size) -> streamsize;

        virtual auto receive(buffer, buffer_size) -> streamsize;

        virtual auto seekpos(offset, seekdir) -> pos_type;

        virtual auto is_open() -> bool;

        virtual auto file_descriptor() -> int;
    };
}
```

It enables wizardly things!

For example:

- Compression
- Encryption
- Statistics / Diagnostics
- New Protocol Support (e.g. RDMA)

And more ...

Example: Compression w/ Snappy!

From **snappy_transport.hpp**

```
#include <snappy-c.h>
#include <vector>

// ... Boilerplate ...

auto send(char_type* buffer, streamsize buffer_size) -> streamsize override
{
    auto output_length = snappy_max_compressed_length(_buffer_size);
    std::vector<char> output(output_length);
    snappy_compress(_buffer, _buffer_size, output.data(), &output_length);
    return tp_->send(output.data(), output_length);
}

auto receive(char_type* buffer, streamsize buffer_size) -> streamsize override
{
    const auto bytes_read = tp_->receive(_buffer, _buffer_size);

    // Uncompress the buffer if the buffer can be uncompressed.
    if (snappy_validate_compressed_buffer(_buffer, bytes_read) == SNAPPY_OK) {
        std::size_t output_length;
        snappy_uncompressed_length(_buffer, bytes_read, &output_length);

        std::vector<char_type> compressed(bytes_read);
        std::copy(_buffer, _buffer + bytes_read, std::begin(compressed));
        snappy_uncompress(compressed.data(), compressed.size(), _buffer, &output_length);

        return output_length;
    }

    return bytes_read;
}

// ... Boilerplate ...
```

Example: Compression w/ Snappy! (cont.)

```cpp
#include <irods/dstream.hpp>
#include <irods/transport/default_transport.hpp>
#include <irods/transport/snappy_transport.hpp>

int main()
{
    namespace io = irods::experimental::io;

    auto large_buffer = read_a_whole_lot_of_data();

    // No compression here.
    io::client::default_transport dtp{conn};

    if (io::odstream out{dtp, "/tempZone/home/rods/foo.txt"}; out) {
        out.write(large_buffer.data(), large_buffer.size());
    }

    // Want compression? Just decorate the original transport with
    // another transport that provides compression!
    io::client::snappy_transport stp{conn, dtp};

    if (io::odstream out{stp, "/tempZone/home/rods/foo.txt"}; out) {
        out.write(large_buffer.data(), large_buffer.size());
    }
}
```

Example: Compression w/ Snappy! (cont.)

Using **tc** to emulate a delay of **200ms** RTT.

- File Size = 72971843 bytes (73MB)
- Buffer Size = 4000000 bytes (4MB)
- Single Stream Object
- Custom iRODS server supporting compression and decompression

- Write Speeds roughly 40% faster
- Read Speeds roughly 36% faster

Results vary depending on the network, file size, and buffer size.