The Link Layer service in a Quantum Internet
draft-dahlberg-ll-quantum-02

Abstract

In a classical network the link layer is responsible for transferring a datagram between two nodes that are connected by a single link, possibly including switches. In a quantum network however, the link layer will need to provide a robust entanglement generation service between two quantum nodes which are connected by a quantum link. This service can be used by higher layers to produce entanglement between distant nodes or to perform other operations such as qubit transmission, without full knowledge of the underlying hardware and its parameters. This draft defines what can be expected from the service provided by a link layer for a Quantum Network and defines an interface between higher layers and the link layer.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 17, 2019.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents
The most important fundamental operation in a quantum network is the generation of entanglement between nodes. Short-distance entanglement can be used to generate long-distance entanglement with the use of an operation called entanglement swap [1] (also formalised in [2]). If nodes A and B share an entangled pair and similarly for B and C, B can perform a so called Bell measurement [3] and send the measurement outcome (2 bits) over a classical channel to A or C such that in the end A and C share an entangled pair. Furthermore, long-distance entanglement does in turn enable long-distance qubit transmission by the use of quantum teleportation [3] (also formalised in [2]). Node A can teleport an unknown qubit state to B by consuming an entangled pair between A and B and sending two classical bits to B. For an overview of quantum networking and its applications we refer to [5].

Long lived entanglement between distant nodes capable of storing such entanglement has been demonstrated over a distance of up to 1.3 km [4], in a proof-of-principle experiment. This entanglement was also heralded, that is, there exits a so-called heralding signal that indicates success in entanglement production without consuming such entanglement. Short lived and non-heralded entanglement has been observed from a satellite over a distance of 1200 km [6] in a proof of principle experiment. The next step towards a quantum network is
to turn ad-hoc experiments that produce entanglement into a reliable service. This is the role of the link layer, which turns an ad-hoc physical setup to a reliable entanglement generation service. Reliable here means that the higher layers can (unless a timeout or other critical failures occur) rely in deterministic entanglement production. In particular, this means that since the underlying physical process is often probabilistic but entanglement generation can be confirmed using the heralding signal, one of the main tasks of the link layer is to manage re-tries in producing entanglement at the physical layer. Once an entangled pair has been generated, the nodes need to be able to agree on which qubits are involved in which entangled pair in order to use it, thus another main task of the link layer is to provide an entanglement identifier.

2. Scope

This draft is meant to define the service and interface of a link layer of a quantum network. Further considerations that motivate this definition can be found in [7]. It does not present a protocol realising this service. However a protocol that indeed does this have been proposed in [7], together with more details on use cases and design decisions in forming a quantum network stack.

3. Desired service

This section defines the service that a link layer provides in a quantum network. The interface and header specification is defined in the next section.

A link layer between two nodes A and B of a quantum network must provide the following minimal features (see [7] for an extended feature set):

- Allow both node A and B to initialize entanglement generation.
- Allow the initializing node to specify a desired minimum fidelity[3] and maximum waiting time.
- Notify both nodes of success or failure of entanglement generation before the requested maximum waiting time has passed since the request was initialized.
- If success is notified, the generated entangled pair has with high confidence higher (or equal) fidelity than the desired minimum fidelity.
For a successful request, provide an entanglement identifier to allow higher layers to use identify the entangled pair in the network without the need for further communication.

4. Interface

This section describes the interface between higher layers and the link layer in a quantum network, along with header specifications for the type of messages. The interface consists of a single type of message from the higher layers to the link layer, which is the CREATE message for requesting entanglement generation. Response messages from the link layer to the higher layers take either the form of an ACK, an OK message or one of many error messages. The ACK is sent back directly upon receiving a CREATE if the link layer supports the request and contains a CREATE ID such that the higher layer can associated the subsequent OK messages to the correct request. It is assumed that the nodes in the network are assigned a unique ID in the network, which is used in the Remote Node ID parameters of the messages below.

4.1. Higher layers to link layer

The higher layers can send a CREATE message to the link layer to request the generation of entanglement. Along with other parameters, as specified below the higher layers can specify a minimum fidelity, a maximum waiting time and the number of entangled pairs to be produced.

4.1.1. Header specification

The CREATE message contains the following parameters:

- Remote Node ID (32 bits): Used if the node is directly connected to multiple nodes. Indicates which node to generate entanglement with.

- Minimum fidelity (16 bits): The desired minimum fidelity, between 0 and 1, of the generated entangled pair. A binary value encoding the integer ‘n’ represents the fidelity ‘n’ divided by \(2^{16}-1\).

- Max Time (16 bits): The maximum time in milliseconds that the higher layer is willing to wait for the request to be fulfilled. Represented as a binary16 float as specified in [8].

- Purpose ID (16 bits): Allows the higher layer to tag the request for a specific purpose. If the request is from an application this can be thought of as a port number. The purpose ID can also be used by a network layer to specify that this entanglement
request is part of long-distance entanglement generation over a specific path.

- **Number (16 bits):** The number of entangled pairs to generate.

- **Priority (3 bits):** Can be used to indicate if this request is of high priority and should ideally be fulfilled early. Higher means faster service.

- **Type of request (TPE) (1 bit):** Either create and keep (K) or measure directly (M), where K stores the generated entanglement in memory and M measures the entanglement directly.

- **Atomic (ATO) (1 bit):** A flag that indicates that the request should be satisfied as a whole without interruption by other requests.

- **Consecutive (CON) (1 bit):** A flag indicating an OK is returned for each pair made for a request. Otherwise, an OK is sent only when the entire request is completed (more common in application use cases). For K type requests, this means all pair should be in memory at the same time.

- **Random Basis Choice (RBC) (2 bits):** Choose to measure uniformly randomly in either
  - 00: Ignored
  - 01: X or Z basis (BB84)
  - 10: X, Y or Z basis (six state)
  - 11: CHSH rotated bases, Z basis rotated by angles +/- pi/4 around Y axis.

  - **Rotation of measurement basis in the case of M types of requests.** Three rotations from the defaults Z basis are performed, first a rotation around the X-axis (ROTX1), then a rotation around the Y-axis (ROTY) and finally a rotation again around the X-axis. Note that arbitrary rotations can be composed as these three rotations, see <https://en.wikipedia.org/wiki/Euler_angles>. If all three fields are 00000000, the qubits are measured in the Z basis. If RBC is not 00, these three fields (ROTX1, ROTY and ROTX2) are ignored.

    - Measurement rotation around X (ROTX1) (8 bits): Measurement to be performed in the case of M types of request. Default is Z.
measurement. Specified measurement to be rotated around the X axis by angle of 2 \( \pi/256 \) * ROTX1

* Measurement rotation around Y (ROTY) (8 bits): Measurement to be performed in the case of M types of request. Default is Z measurement. Specified measurement to be rotated around the Y axis by an angle of 2 \( \pi/256 \) * ROTY

* Measurement rotation around X (ROTX2) (8 bits): Measurement to be performed in the case of M types of request. Default is Z measurement. Specified measurement to be rotated around the X axis by an angle of 2 \( \pi/256 \) * ROTX2

The complete header specification of the CREATE message is given in Figure 1.

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 1: CREATE message header format

4.2. Link layer to higher layers

When receiving a CREATE message from higher layers the link layer will directly respond and notify the higher layer whether requests will be scheduled for generation. If so the link layer responds with an ACK containing a CREATE ID. The higher layer may choose to use this CREATE ID together with the ID of the requesting node to associate OK messages it receives from the link layer to the correct request. Note that the ID of the requesting node is needed since the ACK is returned directly and the CREATE ID is thus not unique for requests from different nodes. If the link layer does not support the given request an error message is instead returned.

When a request is satisfied an OK message is sent to the higher layer. The OK message contains different fields depending on whether the request was of type K (keep) or M (measure directly). For K the
OK contains a logical qubit identifier (LQID) such that the higher layer can know which logical qubit holds the generated entanglement. For M the OK contains the basis which the qubit was measured and the measurement outcome.

Both during and after entanglement generation, the link layer can return error messages to the higher layers, as further described below. For example if something happens to the qubit or another error occurs such that the entanglement is not valid anymore, the link layer can issue an ERR_EXPIRE message.

4.2.1. Header specification

To distinguish the different types of messages that the link layer can return to the higher layer, the first part of the header is a 4 bit field which specifies the type of message using the following mapping:

- 0001: ACK
- 0010: Type K OK
- 0011: Type M OK
- 0100: ERR

The complete header specification for these four types of messages are shown below in Figure 2 to Figure 5.

The ACK message contains the following parameters:

- Create ID (16 bits): A Create ID that the higher layer can use to associate subsequent OK messages to the request.

```
+-----------------+-----------------+
| Type | Create ID       |
+-----------------+-----------------+
```

Figure 2: ACK message header format

The type K OK message contains the following parameters:

- Create ID (16 bits): Must be the same Create ID that was given in the ACK of the corresponding request.

Dahlberg, et al.  Expires October 17, 2019
- Logical Qubit ID (LQID) (4 bits): A logical ID of the qubit which is part of the entangled pair.

- Directionality flag (D) (1 bit): Specifies if the request came from this node (D=0) or from the remote node (D=1).

- Sequence number (16 bits): A sequence number for identifying the entangled pair. It is assumed to be unique for entangled pairs between the given nodes. Thus together with the IDs of the nodes between which the entanglement is produced, one can create an entanglement identifier which is unique in the network.

- Purpose ID (16 bits): The purpose ID of the request (only used by the node which did not initiate the request)

- Remote Node ID (32 bits): Used if the node is directly connected to multiple nodes.

- Goodness (16 bits): An estimate of the fidelity of the generated entangled pair. Should not be seen as a guarantee.

- Time of Goodness (ToG) (16 bits): The time of the goodness estimate. Not necessarily the time when the estimate is performed but rather the time for which the estimate is for. Can be used to make an updated estimate based on decoherence times of the qubits.

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Type  |          Create ID            | LQID  |D|   Unused    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Sequence Number        |          Purpose ID           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Remote Node ID                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Goodness            |      Time of Goodness         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Type K OK message header format

The type M OK message contains the following parameters:

- Create ID (16 bits): The same Create ID that was given in the ACK of the corresponding request.

- Measurement outcome (M) (1 bit): The outcome of the measurement performed on the entangled pair.
o Basis (3 bits): Which basis the entangled pair was measured in, used if the basis is random, i.e. if RBC is not 00 in the CREATE. The following representation is used:

* 000: Z-basis
* 001: X-basis
* 010: Y-basis
* 011: Z-basis rotated by angle \( \pi/4 \) around Y-axis
* 100: Z-basis rotated by angle \(-\pi/4\) around Y-axis
* 101: Unused
* 110: Unused
* 111: Unused

o Directionality flag (D) (1 bit): Specifies if the request came from this node (D=0) or from the remote node (D=1).

o Sequence number (16 bits): A sequence number for identifying the entangled pair. It is assumed to be unique for entangled pairs between the given nodes. Thus together with the IDs of the nodes, one can create an entanglement identifier which is unique in the network.

o Purpose ID (16 bits): The purpose ID of the request (only used by the node which did not initiate the request)

o Remote Node ID (32 bits): Used if the node is directly connected to multiple nodes.

o Goodness (16 bits): An estimate of the fidelity of the generated entangled pair. Should not be seen as a guarantee.

Note: Time of Goodness is not needed here since there is no decoherence on the measurement outcomes.
The ERR message contains the following parameters:

- **Create ID (16 bits):** The same Create ID that was given in the ACK of the corresponding request.

- **Error code (ERR) (4 bits):** Specifies what error occurred. See below what the error codes mean.

- **Expire by sequence numbers (S) (1 bit):** Used by ERR_EXPIRE, to specify whether a range of sequence numbers should be expired (S=1) or all sequence numbers associated with the given Create ID and Origin Node (S=0).

- **Sequence number low (16 bits):** Used by error code ERR_EXPIRE to identify a range of sequence numbers that needs to be expired. Numbers above Sequence number low (inclusive) and below Sequence number high (exclusive) should be expired.

- **Sequence number high (16 bits):** Used by error code ERR_EXPIRE to identify a range of sequence numbers that needs to be expired. Numbers above Sequence number low (inclusive) and below Sequence number high (exclusive) should be expired.

- **Origin Node (32 bits):** Used if the node is directly connected to multiple nodes. Needed here since Create IDs are not unique for request from different nodes.

---

Figure 4: Type M OK message header format
The different error codes used in an error message are the following:

- **Error returned directly when a CREATE message is received:**
  
  * **ERR_UNSUPP (0001):** The given request is not supported. For example if the minimum fidelity is not achievable or if the request is of type K and the hardware cannot store entanglement.
  
  * **ERR_CREATE (0010):** The create message could not be parsed.
  
  * **ERR_REJECTED (0011):** The request was rejected by this node based on for example the Purpose ID.
  
  * **ERR_OTHER (0100):** An unknown error occurred.

- **Error returned after a CREATE message is received, before or after an OK is returned:**

  * **ERR_EXPIRE (0101):** One or more already sent OK messages have expired and the entangled pair is not available anymore. Can either be specified as a range of sequence numbers or by a create ID by using the S flag.
  
  * **ERR_REJECTED (0011):** The request was rejected by the other node based on for example the Purpose ID.
  
  * **ERR_TIMEOUT (0110):** The request was not satisfied within the requested max waiting time.

5. IANA Considerations

This memo includes no request to IANA.
6. Acknowledgements

The authors would like to acknowledge funding received from the EU Flagship on Quantum Technologies, Quantum Internet Alliance.

The authors would further like to acknowledge Tim Coopmans, Leon Wubben, Filip Rozpedek, Matteo Pompili, Arian Stolk, Przemyslaw Pawelczak, Robert Knegjens, Julio de Oliveira Filho, Sidney Cadot, Joris van Rantwijk and Ronald Hanson for inputs and discussion and Wojciech Kozlowski for useful feedback on this draft.

7. Informative References


Authors’ Addresses

Axel Dahlberg
QuTech, Delft University of Technology
Lorentzweg 1
Delft 2628 CJ
Netherlands
Phone: +31 (0)65 8966821
Email: e.a.dahlberg@tudelft.nl

Matthew Skrzypczyk
QuTech, Delft University of Technology
Lorentzweg 1
Delft 2628 CJ
Netherlands
Email: m.d.skrzypczyk@student.tudelft.nl

Stephanie Wehner (editor)
QuTech, Delft University of Technology
Lorentzweg 1
Delft 2628 CJ
Netherlands

Abstract

The vision of a quantum internet is to fundamentally enhance Internet technology by enabling quantum communication between any two points on Earth. To achieve this goal, a quantum network stack must be built from the ground up as the physical nature of the communication is fundamentally different. The first realisations of quantum networks are imminent, but there is no practical proposal for how to organise, utilise, and manage such networks. In this memo, we attempt lay down the framework and introduce some basic architectural principles for a quantum internet. This is intended for general guidance and general interest, but also to provide a foundation for discussion between physicists and network specialists.
Quantum networks are distributed systems of quantum computers that utilise fundamental quantum mechanical phenomena such as superposition, entanglement, and quantum measurement to achieve capabilities beyond what is possible with classical networks. This new networking paradigm offers promise for a range of new applications such as tamper-proof communications [1], distributed
quantum computation [2], and quantum sensor networks [3]. The field
of quantum communication has been a subject of active research for
many years and the most well-known application of quantum computers
that has already been deployed, quantum key distribution (QKD), is a
protocol used for secure communications.

Fully quantum networks capable of transmitting and managing entangled
states in order to send, receive, and manipulate distributed quantum
states are now imminent [4] [5]. Whilst a lot of effort has gone
into physically connecting the devices and bringing down the error
rates there are no concrete proposals for how to run these networks.
To draw an analogy with a classical network, we are at a stage where
we can physically connect our devices and send data, but all sending,
receiving, buffer management, connection synchronisation, and so on,
must be managed by the application itself at what is essentially
assembly level. Furthermore, whilst physical mechanisms for
forwarding quantum states exist, there are no protocols for managing
it.

2. Model of computation

In order to understand the framework for quantum networking a basic
understanding of quantum information is necessary. The following
sections aim to introduce the bare minimum necessary to be understand
the principles of operation of a quantum network. This exposition
was written with a classical networking audience in mind. It is
assumed that the reader has never before been exposed to any quantum
physics.

2.1. Qubit

The differences between quantum computation and classical computation
begin at the bit-level. A classical computer operates on the binary
alphabet \( \{ 0, 1 \} \). A quantum bit, a qubit, exists over the same
binary space, but unlike the classical bit, it can exist in a so-
called superposition of the two possibilities:

\[ a \ket{0} + b \ket{1}, \]

where \( \ket{X} \) denotes a quantum state, here the binary 0 and 1, and the
coefficients \( a \) and \( b \) are complex numbers called probability
amplitudes. Physically, such a state can be realised using a variety
of different technologies such as electron spin, photon polarisation,
atomic energy levels, and so on.

Upon measurement, the qubit loses its superposition and irreversibly
collapses into one of the two basis states, either \( \ket{0} \) or \( \ket{1} \). Which
of the two states it ends up in is not deterministic. The
probability of measuring the state in the $|0>$ state is $|a|^2$ and similarly the probability of measuring the state in the $|1>$ state is $|b|^2$. This randomness is not due to our ignorance of the underlying mechanisms, but rather it is a fundamental feature of a quantum mechanical system [6].

The superposition property plays an important role in fundamental gate operations on qubits. Since a qubit can exist in a superposition of its basis states, the elementary quantum gates are able to act on all states of the superposition at the same time. For example, consider the NOT gate:

$$\text{NOT}(a\ |0> + b\ |1>) \rightarrow a\ |1> + b\ |0>.$$ 

2.2. Multiple qubits

When multiple qubits are combined in a single quantum state the space of possible states grows exponentially and all these states can coexist in a superposition. For example, the general form of a two qubit register is

$$a\ |00> + b\ |01> + c\ |10> + d\ |11>$$

where the coefficients have the same probability amplitude interpretation as for the single qubit state. Each state represents a possible outcome of a measurement of the two qubit register. For example, $|01>$, denotes a state in which the first qubit is in the state $|0>$ and the second is in the state $|1>$. Performing single qubit gates affects the relevant qubit in each of the superposition states. Similarly, two qubit gates also act on all the relevant superposition states, but their outcome is far more interesting.

Consider a two qubit register where the first qubit is in the superposed state $(|0> + |1>)/\sqrt{2}$ and the other is in the state $|0>$. This combined state can be written as:

$$(|0> + |1>)/\sqrt{2} \times |0> = (|00> + |10>)/\sqrt{2},$$

where $\times$ denotes a tensor product (the mathematical mechanism for combining quantum states together). Let us now consider the two-qubit CNOT gate. The CNOT gate takes as input two qubits, a control and target, and applies the NOT gate to the target if the control qubit is set. The truth table looks like
Now, consider performing a CNOT gate on the ensemble with the first qubit being the control. We apply a two qubit gate on all the superposition states:

\[
\text{CNOT } \left( |00\rangle + |10\rangle \right)/\sqrt{2} \rightarrow \left( |00\rangle + |11\rangle \right)/\sqrt{2}.
\]

What is so interesting about this two-qubit gate operation? The final state is *entangled*. There is no possible way of representing that quantum state as a product of two individual qubits, they are no longer independent and their behaviour cannot be fully described without accounting for the other qubit. The states of the two individual qubits are now correlated beyond what is possible to achieve classically. Neither qubit is in a definite $|0\rangle$ or $|1\rangle$ state, but if we perform a measurement on either one, the outcome of the partner qubit will *always* yield the exact same outcome. The final state, whether it’s $|00\rangle$ or $|11\rangle$, is fundamentally random as before, but the states of the two qubits following a measurement will always be identical.

Once a measurement is performed, the two qubits are once again independent. The final state is either $|00\rangle$ or $|11\rangle$ and both of these states can be trivially decomposed into a product of two individual qubits. The entanglement has been consumed and if the same measurement is to be repeated, the entangled state must be prepared again.

3. Entanglement as the fundamental service

Entanglement is the fundamental building block of quantum networks. To see this, consider the final state from the previous section:

\[
\left( |00\rangle + |11\rangle \right)/\sqrt{2}.
\]

Neither of the two qubits is in a definite $|0\rangle$ or $|1\rangle$ state and we need to know the state of the entire register to be able to fully describe the behaviour of the two qubits.

Now consider sending one of the qubits to another device. This device can be anywhere: on the other side of the room, in a different
country, or even on a different planet. Provided negligible noise has been introduced, the two qubits will forever remain in the entangled state until a measurement is performed. The physical distance does not matter at all for entanglement.

This lies at the heart of quantum networking, because it is possible to leverage these non-classical correlations in order to design completely new types of algorithms that are not possible to achieve with just classical communication. Examples of such applications are quantum cryptography, blind quantum computation, or distributed quantum computation.

As a trivial example consider the problem of reaching consensus between two nodes. The two nodes want to agree on the value of a single bit. In a quantum network they can simply request the network to generate the state \((|00\rangle + |11\rangle)/\sqrt{2}\) for them and that is essentially all that needs to be done. Once any of the two nodes performs a measurement the state of the two qubits collapses to either \(|00\rangle\) or \(|11\rangle\) so whilst the outcome is random, the two nodes will always measure the same value. We can also build the more general multi-qubit state \((|00...\rangle + |11...\rangle)/\sqrt{2}\) and perform the same algorithm between an arbitrary number of nodes.

However, it is impossible to entangle two qubits without ever having them directly interact with each other (e.g. by performing a local two-qubit gate, such as the CNOT). A local interaction is necessary to create entanglement and thus such states cannot be created between two quantum computers that cannot transmit quantum states to each other. Therefore, it is the entanglement property of multi-qubit states that draws the line between a genuine quantum network and a collection of quantum computers connected over a classical network.

A quantum network is defined as a collection of nodes that is able to distribute entangled states amongst themselves. A quantum computer that is able to communicate classically with another quantum computer is not a member of a quantum network.

This is a crucial difference between classical and quantum networks. Classical applications transmit data over the network to synchronise distributed state. Quantum network applications obtain distributed states, synchronised at the physical level via entanglement, from the network to perform quantum algorithms.

More complex services and applications can be built on top of entangled states distributed by the network.
4. Achieving quantum connectivity

4.1. No-cloning theorem

To build a network we must first physically connect all the nodes with quantum channels that enable them to distribute the entanglement. Unfortunately, our ability to transfer quantum states is complicated by the no-cloning theorem.

The no-cloning theorem states that it is impossible to create an identical copy of an arbitrary unknown quantum state. Since performing a measurement on a quantum state destroys its superposition, there is no practical way of learning the exact state of a qubit in an unknown state. Therefore, it is impossible to use the same mechanisms that worked for classical networks for error-correction, amplification, retransmission, and so on as they all rely on the ability to copy the underlying data. Since any physical channel will always be lossy, connecting a quantum network is a challenging endeavour and its architecture must at its core address this very issue.

4.2. Direct transmission

The most straightforward way to distribute an entangled state is to simply transmit one of the qubits directly to the other end across a series of nodes while performing sufficient error correction to bring losses down to an acceptable level. Despite the no-cloning theorem and the inability to directly measure a quantum state error-correcting mechanisms for quantum communication exist [7]. However, even in the most optimistic scenarios the hardware requirements to fault-tolerantly transmit a single qubit are beyond near-term capabilities. Nevertheless, due to the promise of fault-tolerance and its favourable poly-logarithmic scaling with distance, this may eventually become a desirable method for entanglement distribution.

4.3. Bell pairs and entanglement swapping

4.3.1. Bell Pairs

An alternative relies on the observation that we do not need to be able to distribute any arbitrary entangled quantum state. We only need to be able to distribute any one of what are known as the Bell Pair states. Bell Pair states are the entangled two-qubit states:

\[
\begin{align*}
|00\rangle + |11\rangle, \\
|00\rangle - |11\rangle, \\
|01\rangle + |10\rangle, \\
|01\rangle - |10\rangle,
\end{align*}
\]
where the constant $1/\sqrt{2}$ normalisation factor has been ignored for clarity. Any of the four Bell Pair state above will do as it is possible to transform any Bell Pair into another Bell Pair with local operations performed on only one of the qubits. That is, either of the nodes that hold the two qubits of the Bell Pair can apply a series of single qubit gates to just their qubit in order to transform the ensemble between the different variants.

Distributing a Bell Pair between two nodes is much easier than transmitting an arbitrary quantum state over a network. Since the state is known error-correction is easier and error-detection combined with reattempts becomes a valid strategy.

The reason for using Bell Pairs specifically as opposed to any other two-qubit state, is that they are the maximally entangled two-qubit set of basis states. Maximal entanglement means that these states have the strongest non-classical correlations of all possible two-qubit states. Furthermore, since single-qubit local operations can never increase entanglement, less entangled states would impose some constraints on distributed quantum algorithms. This makes Bell Pairs particularly useful as a generic building block for distributed quantum applications.

4.3.2. Teleportation

The observation that we only need to be able to distribute Bell Pairs relies on the fact that this enables the distribution of any other arbitrary entangled state. This can be achieved via quantum state teleportation. Quantum state teleportation consumes an unknown quantum state that we want to transmit and recreates it at the desired destination.

To achieve this, a Bell Pair needs to be distributed between the source and destination. The source then entangles the transmission qubit with one of the Bell Pair and performs a measurement. This consumes the Bell Pair’s entanglement turning the source and destination qubits into independent states. However, this process transforms the Bell Pair’s qubit at the destination into the transmission qubit’s original state. Note the process requires the source to also communicate its two-bit measurement result so that the destination can correct for the randomness of the outcome.

The unknown quantum state that was transmitted never entered the network itself. Therefore, the network needs to only be able to reliably produce Bell Pairs between any two nodes in the network.
4.3.3. Bell Pair links and entanglement swapping

Reducing the problem to one of generating a Bell Pair state has facilitated the problem, but it has not solved it.

The technology to generate a Bell Pair between two directly connected quantum nodes already exists and has been demonstrated in laboratory conditions [8]. Interestingly, neither of the two qubits of the pair need to be transmitted any further.

A Bell Pair between any two nodes in the network can be constructed from Bell Pairs generated along each individual link on the path between the two end-points. Each node along the path can consume the two Bell Pairs on the two links that it is connected to in order to produce a new Bell Pair between the two far ends. This process is known as entanglement swapping. Pictorially it can be represented as follows:

\[
\begin{array}{c}
\text{x---x} \\
\end{array}
\]

where x signifies a Bell Pair with individual qubits represented by x, -- denotes a quantum link, and [ ] denotes a node. The diagram above represents the situation after the middle node has generated a Bell Pair with two of its directly connected neighbours. Now, the middle node performs an entanglement swap operation (the exact details of the mechanism are beyond the scope of this memo). This operation consumes the two Bell Pairs and produces a new Bell Pair between the two far ends of this three-node network as follows:

\[
\begin{array}{c}
\text{x---x} \\
\end{array}
\]

The outcome is guaranteed to be a Bell Pair between the two end nodes, but which of the four possible Bell Pairs is produced is not deterministic. However, the middle node will know which one was produced as the entanglement swap is a measurement operation that yields two classical bits. The final state can be inferred from this two-bit readout. Therefore, the middle node needs only to communicate the outcome over a classical channel to one or both ends who can apply a correction to transform the pair into any of its other forms (if so desired).

4.3.4. Distillation

Neither the Bell Pair or the swapping operations are lossless operations. Therefore, with each link and each swap the quality of the state degrades. However, it is possible to create higher quality
Bell Pair states from two or more lower quality Bell Pair states. Therefore, once the quality loss over a given distance become prohibitive, additional redundancy may be used to restore the state quality.

4.4. Direct transmission vs. swapping

Direct state transmission whilst simpler conceptually is much more demanding to implement reliably in practice which means that any near-term practical realisation is more likely to succeed if it is based on the Bell Pair and entanglement swapping architecture. This is the architecture that we will focus on in the rest of this memo for practical reasons.

Nevertheless, we are not entirely discarding the direct transmission proposal. Whilst it does enable the fault-tolerant transmission of unknown quantum states, it might still be more beneficial to use it to distribute Bell Pairs instead. Distributing Bell Pairs via direct transmission means that one can leverage the advantages of entanglement swapping which allows for parallelisation as the Bell Pairs can be built up from both ends simultaneously. Furthermore, the generic nature of the Bell Pair means that a network may provision resources better before it receives any request.

5. Architecture of a quantum internet

5.1. Model of a quantum network

A generic quantum network of three nodes could be represented as

```
<table>
<thead>
<tr>
<th>App</th>
<th>-------------------CC-------------------</th>
<th>App</th>
</tr>
</thead>
<tbody>
<tr>
<td>QNet</td>
<td>-------CC-------</td>
<td>QNet</td>
</tr>
<tr>
<td>-----</td>
<td>-------</td>
<td>-----</td>
</tr>
</tbody>
</table>

\ Bell Pair Gen. / SWAP \ Bell Pair Gen. / 
```

Where "App" is some application running over a quantum network, --CC-- denote classical communication links (e.g. over the public Internet or a private LAN), and "QNet" is a generic network stack. Architectures for the network stack have been proposed already [9], but their discussion is beyond the scope of this memo. However, they all map onto this generic diagram. Nodes within a quantum network that are capable of performing the entanglement swap operation are often referred to as quantum repeaters and we shall adopt this terminology from this point on. End-hosts connecting at the edge of the network are not necessarily repeaters themselves.
The key message here is that a network stack relies on the hardware being able to provide two services: Bell Pair generation across a link, and swap operation. In any network model it is assumed that the physical device is capable of providing both of these services and offers a suitable interface for their usage.

Strictly speaking quantum memories are not needed for a functional quantum network as long as the network is able to generate the Bell Pairs, swap the entanglement, and deliver the final Bell Pair to the application in a usable form. However, in general, to be able to provide the two services above, the hardware will also need to be able to store the qubits in memory which is highly non-trivial.

Furthermore, it is also assumed that the applications are able to communicate classically, and that the nodes themselves are also connected over some classical channel. The classical links between the nodes need not always have an associated quantum link, but it is assumed that any quantum link has a classical link running in parallel.

5.2. Physical constraints

The model above has effectively abstracted away the particulars of the hardware implementation. However, certain physical constraints need to be considered in order to build a practical network. Some of these are fundamental constraints and no matter how much the technology improves, they will always need to be addressed. Others are artefacts of the early stages of a new technology.

5.2.1. Fidelity

The quality of a quantum state is described by a physical quantity called fidelity. Fidelity is the measure of how close a quantum state is to the quantum state we desire it to be in. It expresses the probability that one state will pass a test to identify as the other.

Fidelity is an important property of a quantum system that stems from the fact that no physical operation is perfect. Furthermore, applications will in general require the fidelity of a quantum state to be above some minimum threshold in order to guarantee the correctness of their algorithm and it is the responsibility of the network to provide such a state.

Additionally, entanglement swap operations, even if perfect, lead to a further reduction in the fidelity of the final state. Two imperfect Bell Pairs when combined will produce a slightly worse Bell Pair. Whilst distillation is one of the available mechanisms to
correct for these errors it requires additional Bell Pairs to be produced. There will be a trade-off between how much distillation is to be done versus what fidelity is acceptable.

This is a fundamental constraint as perfect noiseless operations and lossless communication channels are unachievable. Therefore, no Bell Pair will be generated with perfect fidelity and the network must account for this.

5.2.2. Memory lifetimes

In addition to discrete operations being imperfect, storing a qubit in memory is also highly non-trivial. The main difficulty in achieving persistent storage is that it’s extremely challenging to isolate a quantum system from the environment. The environment introduces an uncontrollable source of noise into the system which affects the fidelity of the state. This process is known as decoherence. Eventually, the state has to be discarded once its fidelity degrades too much.

The memory lifetime depends on the particular physical setup, but the highest achievable values currently are on the order of hundreds of milliseconds. These values have increased tremendously over the lifetime of the different technologies and are bound to keep increasing. However, if quantum networks are to be realised in the near future, they need to be able to handle short memory lifetimes. An architecture that handles short lifetimes may also be more cost-efficient in the future.

5.2.3. Rates

Entanglement generation on a link between two connected nodes is not a very efficient process and it requires many attempts to succeed. A fast repetition rate for Bell Pair generation is achievable, but only one in a few thousands will succeed. Currently, the highest achievable rates of success are of the order of 10 Hz. Combined with short memory lifetimes this leads to very tight timing windows to build up network-wide connectivity. Achievable rates are likely to increase with time, but just like with quantum memories, it may be more cost-efficient in the future to provide low-rate links in some parts of the network.

5.2.4. Communication qubit

Some physical architectures are not able to generate entanglement using any memory qubit that they have access to. In these systems, entanglement is generated using a communication qubit and once a Bell Pair has been generated, the qubit state is transferred into memory.
This may impose additional limitations on the network. In particular if a given node has only one communication qubit it cannot simultaneously generate Bell Pairs over two links. It must generate entanglement over the links one at a time.

5.2.5. Homogeneity

Currently all hardware implementations are homogeneous and they do not interface with each other. In general, it is very challenging to combine different quantum information processing technologies due to their sensitivity to losses. Coupling different technologies with each other is of great interest as it may help overcome the weaknesses of the different implementations, but this is not a near-term goal.

5.3. Architectural principles

Given that the most practical way of realising quantum network connectivity is using Bell Pair and entanglement swapping repeater technology what sort of principles should guide us in assembling such networks such that they are functional, robust, efficient, and most importantly: they work. Furthermore, how do we design networks so that they work under the constraints imposed by the hardware available today, but do not impose unnecessary burden on future technology. Redeploying network technology is a non-trivial process.

As this is a completely new technology that is likely to see many iterations over its lifetime, this memo must not serve as a definitive set of rules, but merely as a general guide based on principles and observations made by the community. The benefit of having a community built document at this early stage is that expertise in both quantum information and network architecture is needed in order to successfully build a quantum internet.

5.3.1. Goals of a quantum internet

When outlining any set of principles we must ask ourselves what goals do we want to achieve as inevitably trade-offs must be made. So what sort of goals should drive a quantum network architecture? The following list has been inspired by the history of the classical Internet, but it will inevitably evolve with time and the needs of its users. The goals are listed in order of priority which in itself may also evolve as the community learns more about the technology.

1. Support distributed quantum applications

The primary purpose of a quantum internet is to run distributed quantum algorithms and it is of utmost importance that they can
run well and efficiently. Therefore, the needs of quantum applications should always be considered first.

If a network is able to distribute entanglement it is officially quantum. However, if it is unable to distribute these states with a sufficiently high fidelity at a reasonable rate for a majority of potential applications it is not practical.

2. Support tomorrow’s distributed quantum applications

There are many applications already proposed to run over a quantum internet. However, more algorithms will be invented as the community grows as well as the robustness and the reliability of the technology. Any proposed architecture should not constrain the capabilities of the network for short-term benefit.

3. Hardware heterogeneity

There are multiple proposals for realising practical quantum repeaters and they all have their advantages and disadvantages. It is also very likely that the most optimal technologies in the future will be hybrid combinations of the many different solutions currently under development. It should be an explicit goal of the architecture to allow for a large variety of hardware implementations.

4. Be flexible with regards to hardware capabilities and limitations

This goal encompasses two important points. First, the architecture should be able to function under the physical constraints imposed by the current generation hardware. Second, it should not make it difficult to run the network over any hardware that may come along in the future. The physical capabilities of repeaters will improve and redeploying a technology is extremely challenging.

5. Security, availability, and resilience

Whilst the priority for the first quantum networks should be to simply work, we cannot forget that ultimately they have to also be secure. There are three key security considerations at the network level, confidentiality, integrity, and authenticity.

Confidentiality and integrity – it is vital that the network can provide a reasonable guarantee of the minimum fidelity of a delivered Bell Pair as the application’s own security mechanisms rely on this. Uncertainty about the fidelity of a Bell Pair may potentially expose its data to an eavesdropper.
Authenticity – it is important that any application can have confidence that the other end of the Bell Pair has been delivered to the desired partner.

Additionally a practical and usable network is able to continue to operate despite losses and failures, and will be robust to malicious actors trying to disable connectivity. These may be simply considered different aspects of security, but it is worthwhile to address them explicitly at the architectural level already.

6. Easy to manage and monitor

Quantum networks rely on complex physical phenomena and require hardware that is challenging to build. Furthermore, the quantum resources will at first be very scarce and potentially very expensive. This entails a need for a robust management solution. It is important that a good management solution needs to come with adequate monitoring capabilities.

Good management solutions may also be key to optimising the networks which in turn may be crucial in making them economically feasible. Unlike user data that is transmitted over classical networks, quantum networks only need to generate generic Bell Pairs. This leaves a lot of room for pre-allocating resources in an efficient manner.

5.3.2. The principles of a quantum internet

The principles support the goals, but are not goals themselves. The goals define what we want to build and the principles provide a guideline in how we might achieve this. The goals will also be the foundation for defining any metric of success for a network architecture, whereas the principles in themselves do not distinguish between success and failure.

1. Bell Pairs are the fundamental building block

The key service that a quantum network provides is the distribution of entanglement between the nodes in a network. This point additionally specifies that the entanglement is primarily distributed in the form of the entangled Bell Pair states which should be used as a building block in providing other services, including more complex entangled states.

2. Fidelity is part of the service
In addition to being able to deliver Bell Pairs to the communication end-points, the Bell Pairs must be of sufficient fidelity. However, different applications will have different requirements for what fidelity they can work with. It is the network’s responsibility to balance the resource usage with respect to the application’s requirements. It may be that it is cheaper for the network to provide lower fidelity pairs that are just above the threshold required by the application than it is to guarantee high fidelity pairs to all applications regardless of their requirements.

3. Bell Pairs are indistinguishable

Any two Bell Pairs between the same two nodes are indistinguishable for the purposes of an application provided they both satisfy its required fidelity threshold. This point is crucial in enabling the reuse of resources of a network and for the purposes of provisioning resources to meet application demand.

4. Time as an expensive resource

With the current technology, time is the most expensive resource. It is not the only resource that is in short supply (memory, and communication qubits are as well), but ultimately it is the lifetime of quantum memories that imposes the most difficult conditions for operating an extended network of quantum nodes. Current hardware has low rates of Bell Pair generation, short memory lifetimes, and access to a limited number of communication qubits. All these factors combined mean that even a short waiting queue at some node could be enough for the Bell Pairs to decohere.

However, time is only expensive once quantum operations are underway. If no quantum operations are currently being processed then the network can use this time to prepare and provision resources.

As hardware improves, the need for carefully timing quantum operations may become smaller. It is currently unknown what the cost of these improvements will be, but it is conceivable that there is value in having relatively cheap and undemanding links connected at the edges of a network which will have very short memory lifetimes and low rates of Bell Pair generation.

5. Limit classical communication
This point offers a practical guideline to the issue of timing. A bottleneck in many quantum networked algorithms is the classical communication needed between quantum operations to synchronise state.

For example, some quantum protocols may need to perform a correct for the random outcome of a quantum measurement. For this, they will block the state from further operations until a classical message is received with the information necessary to perform the correction. The time during which the quantum state is blocked is effectively wasted. It reduces the time available for subsequent operations possibly rendering the state useless for an application.

Trade-offs that allow a protocol to limit the number of blocking classical communication rounds once quantum operations have commenced will in general be worth considering.

6. Parallelise quantum operations

A further point to address the issue of timing constraints in the network. The Bell Pairs on the individual links need not be generated one after another along the path between the communication end-points. The order does not matter at all. Furthermore, the order of the swap operations is flexible as long as they don’t reduce the fidelity too much. Parallelising these operations is key to optimising quantum protocols.

7. Avoid time-based coordination when possible

A solution to timing constraints is to synchronise clocks and agree on the timing of events. However, such solutions have several downsides. Whilst network clock synchronisation may be accurate enough for certain purposes it introduces an additional element of complexity, especially when multiple nodes in different networks must be synchronised. Furthermore, clock synchronisation will never be perfect and it is conceivable that hardware capabilities advance so much that time-based mechanisms under-utilise resources in the more efficient parts of the network.

Nevertheless, it may not be possible to avoid clocks, but such solutions should be adequately justified.

8. Pre-allocate resources

Regardless of what application is running over the network it will have the same needs as any other application: a number of
Bell Pairs of sufficient fidelity. Whilst the fidelity is a variable number, the indistinguishability of Bell Pairs means that there is lots of flexibility in how a network may provision resources to meet demand. The additional timing constraints mean that pre-allocation of resources will be central to a usable quantum network.

6. Security Considerations

Even though no user data enters a quantum network security is explicitly listed as a goal in this memo. However, as this is an informational memo it does not propose any concrete mechanisms to achieve these goals.

In summary:

- Confidentiality and integrity in the quantum context is the network’s guarantee on the minimum fidelity of the delivered Bell Pair states. Uncertainty about the fidelity of a Bell Pair may potentially expose an application to an eavesdropper.

- Authenticity in a quantum network is the guarantee that the other end of the Bell Pair is with the requested partner and not any other third party.

7. IANA Considerations

This memo includes no request to IANA.

8. Acknowledgements

The author would like to acknowledge funding received the Quantum Internet Alliance.

The author would further like to acknowledge Stephanie Wehner, Carlo Delle Donne, Matthew Skrzypczyk, and Axel Dahlberg for useful discussions on this topic prior to the submission of this memo.

9. Informative References


Author’s Address
Abstract

This document describes the use of link-state routing protocols on classical links in Quantum Networks. It contains proposals for additions to the IS-IS and OSPF protocols in order for them to transport relevant information for a Quantum Network, specifically, for the creation and manipulation of entangled pairs. The document will describe some of the necessary attributes and some suggestions of how this information may be used.

No Schrodinger’s cats were harmed in the creation of this document.
1. Introduction

Quantum networking is an emerging field using the strange (even counterintuitive) properties of quantum mechanics to bring new, useful capabilities to computing and networking. One of these is "entanglement" [8], where the state of a group of particles must be described as a unit -- it cannot be decomposed to the state of each particle independently. Entangled pairs (often called EPR pairs,
abbreviated here as EP) of particles can be used for quantum teleportation [10] and for quantum key distribution (QKD) [14].

A Quantum Network consists of quantum nodes and links. Here, we will be concerned with controllable quantum nodes (CQN) that allow control decisions. We posit a classical network parallel to the quantum network, with classical nodes (CN) and links. A classical node is colocated with a quantum node; a classical link may be a fiber or wavelength parallel to the corresponding quantum link. The existence of such a classical link is required by most quantum methods to create EPs deterministically or in a heralded fashion, where the creation of EPs is conditioned on a specific signal. To make useful decisions, it is desirable to augment this data to describe the capabilities and states of quantum nodes and links. At current time there is a need for classical links besides quantum links. In the future this might change into a situation where classical links will perhaps become obsolete.

This document proposes to carry entanglement capability data as Type Length Values (TLVs) over IS-IS or OSPF link-state advertisements over the corresponding classical network. A subset of the CQNs may run quantum applications such as QKD; these nodes may want to initiate multihop EPs.

Once an EP is created, the state of one particle ("quantum bit" or qubit) of an EP can be transferred to another qubit within the same QN by a process known as swapping or a SWAP gate ([12]). Also, several pairs of imperfectly entangled qubits can be "distilled" ([13]) to fewer but "better entangled" qubits.

Long distance entanglement can be produced from piecewise short distance entanglement: Given an EP between CQN A and CQN B, and another EP between CQN B and CQN C, one can create an EP between CQN A and CQN C by a process known as an "entanglement swap". These operations can be used to manipulate EPs to improve their lifetimes or their quality, or to create multihop EPs. Physically, qubits can be realized in many ways. For example, they can be represented by the energy levels of Nitrogen Vacancy (NV) Centers in diamond ([16], [17]). Logically, a qubit can be classified as a "communication qubit", a "traveling qubit" or a "storage qubit".

This document primarily discusses the exchange of quantum capabilities over a classical network. Some illustrative examples of how these capabilities can be used in a quantum network may be given, but this document should not be considered authoritative on these procedures.
1.1. Definitions and Notation

The following terms are used in this document:

Quantum link: A quantum link is a connection transporting traveling qubits, typically photons. This could be a physical link, or by means of teleportation over pre-established entanglement amongst distant network nodes. Such pre-shared entanglement effectively forms a shortcut - a virtual quantum link - which can be used exactly once. This document does not describe the usage of these links itself.

Classical link: A classical link is a connection transporting packets. This could be a physical or virtual link carried over a (MPLS) network. The proposed extensions in this document use these links to exchange capabilities.

Controllable Quantum Node (CQN): A controllable quantum node is a quantum device consisting of at least one qubit, capable of performing (a subset of) the following operations described in detail below: storing qubits for some amount of time, performing quantum operations such as entanglement distillation and entanglement swapping, and producing entanglement between the nodes and traveling qubits. The latter are generally realized using photons over fibers or through free space.

The term controllable refers to the fact that external control in software is capable of selecting the desired operations and qubits to use. Such nodes can be quantum repeaters that allow choices of operations to be made, as well as quantum end nodes capable of executing complex application protocols [14]. Quantum repeaters that merely allow timing control, such as automatic entanglement swapping whenever qubits arrive in a specific timing interval, will not be referred to as CQN. Such automated repeaters can be seen as lying at the quantum physical layer and do not enter routing or other decision making, apart from being switched on or off, and hence are not relevant to advertisement protocols like the ones considered here.

Quantum end node (QEN): In this document, a quantum end node [14] is one of a pair of quantum nodes forming an entanglement via a sequence of zero or more CQNs. Quantum end nodes typically run a higher-layer quantum application such as QKD.

Entangled Pair (EP): An entangled pair is a special state of two qubits, known as an EPR pair [8]. An entangled pair of qubits c@A and c@B is denoted [{c@A, c@B}].
The process of entangling two particles \(c@A\) and \(c@B\) is denoted as follows:

\[
\text{ent}(c@A, c@B) \rightarrow [[c@A, c@B]]
\]

\(\text{ent}(c@A, c@B)\) may take time \(T\) and succeed with probability \(P\), and yield an entangled pair \([[c@A, c@B]]\) of fidelity \(F\).

**Fidelity:** A measure of the quality of the entanglement of an EP \(QFid\) \([11]\)). Fidelity lies in the interval \([0, 1]\) where a higher value indicates better quality; usable fidelity values lie in the half-open interval \((0.5, 1]\).

**Communication qubit:** A qubit is called a communication qubit if it is possible to produce entanglement between this qubit and a traveling photon. This can be done by emission from the quantum node, that is, entanglement is produced between the qubit and the photon which is emitted from the quantum node. This process has been demonstrated in a number of physical systems that can be used as quantum nodes such as NV in diamond \([16], [17]\), Ion Traps \([18]\) and Neutral Atoms \([19]\). An example of a communication qubit is the electron spin of the NV in diamond system \([15]\). Entanglement between a communication qubit and traveling photons can also be produced by absorption. Examples include atomic ensemble memories \([20]\).

A communication qubit \(c\) at CQN \(A\) is denoted by \(c@A\), or simply \(c\) (if the node \(A\) is understood).

**Storage qubit:** A qubit is called a storage qubit if the node has the capability to use this qubit as a (temporary) quantum memory, but the qubit cannot serve as a communication qubit. To make storage qubits useful a node is required to possess the ability to transfer the state of a communication qubit to a storage qubit. An example of a storage qubit is the nuclear spin in the NV in diamond system \([16]\).

A storage qubit \(s\) at node \(B\) is denoted \(s@B\).

**Swap:** Two qubits located in the same CQN can interchange states \([13]\). For example, the states of a communication qubit and a storage qubit at \(A\) can be swapped as follows:

\[
\text{swap}(c@A, s@A)
\]

If \(c@A\) was entangled with \(c@B\), the result is that \(s@A\) is now entangled with \(c@B\).
Distillation: Distillation is the process of turning a large number of weakly entangled states into a smaller number of highly entangled states ([13]).

For example, EPs \([c_1@A, c_1@B]\) and \([c_2@A, c_2@B]\) of fidelities \(F_1\) and \(F_2\) respectively may be distilled as follows:

\[
\text{dist}([c_1@A, c_1@B]), ([c_2@A, c_2@B]) \rightarrow [c_3@A, c_3@B]
\]

If distillation is successful, the fidelity \(F_3\) of \([c_3@A, c_3@B]\) will be higher than \(F_1\) and \(F_2\).

Entanglement Swap: Given two EPs \([c@A, c_1@B]\) and \([c_2@B, c@C]\), one can perform an entanglement swap:

\[
\text{entSwap}([c@A, c_1@B]), ([c_2@B, c@C]) \rightarrow [c@A, c@C]
\]

to create a new EP between \(c@A\) and \(c@C\). This is how "multihop" EPs are created from a sequence of "single-hop" EPs.

The swap operation can also be used within a CQN. A possible use case is when there aren’t enough communication qubits to create the needed EPs. If, in the above example, \(B\) doesn’t have two communication qubits \(c_1\) and \(c_2\), the following can be done:

\[
\text{ent}(c@A, c@B) \rightarrow [c@A, c@B] \quad \# \text{entangle}
\]
\[
\text{swap}(c@B, s@B) \rightarrow [c@A, s@B] \quad \# \text{swap EP to storage qubit}
\]
\[
\text{ent}(c@B, c@C) \rightarrow [c@B, c@C] \quad \# \text{use freed up qubit } c@B
\]
\[
\text{entSwap}(c@A, c@B) \rightarrow [s@B, c@C] \quad \# \text{create multihop EP}
\]

2. Motivation

Consider the following (very simple) quantum network consisting of QENs \(A\) and \(B\), and CQNs \(X, Y, Z, U, V\). The goal is to create an EP between qubits at \(A\) and at \(B\), perhaps for the high-level task of QKD between \(A\) and \(B\).

\[
\begin{align*}
X & - Y - Z \\
A & - B \\
U & \cdash\cdash V \\
\end{align*}
\]

\(A, B: \text{QEN}\)

\(X, Y, Z, U, V: \text{CQNs}\)

From \(A\)’s point of view, here are a number of questions:

1. Is \(B\) reachable from \(A\) via quantum links that allow EP creation?

2. If so, along what sequence(s) of quantum nodes?
3. Can each pair of adjacent CQNs in this sequence form EPs? If so, how long will it take, and what fidelity can be expected?

4. If each pair of adjacent CQNs successfully forms EPs of sufficient fidelity, can these be swapped to form a multihop EP between A and B?

5. If a multihop EP between A and B were to be formed, would it be of good enough fidelity, or should a second multihop EP be formed and the two EPs distilled into one high fidelity EP? How many times should this process be repeated?

6. If the overall answer is Yes, should A proceed via sequence A, X, Y, Z, B, or sequence A, U, V, B?

This document aims to provide all CQNs in a quantum network with the information they need to answer such questions, and to create EPs at their desired fidelity and speed.

3. Theory of Operation

A CQN contains one or more communication qubits and one or more storage qubits. Many proposals exist for producing EPs between remote quantum nodes (see for example [16], [17], [18], [20]). Abstractly, these result in the generation of EPs with fidelity F after an expected time t. To give an example, we describe the generation of EPs that has been implemented in NV in diamond ([16]), and Ion Traps ([18]). The largest distance for producing long-lived entanglement is presently 1.3kms ([17]). To entangle a pair of communication qubits, the QNs send carefully timed photons towards the HS. If the process is successful, HS sends an OK message to both QNs.

The classical network control plane is of particular interest here as it would be used by the proposed protocol to advertise and exchange information about the capabilities of the CQNs to generate entanglement. This classical channel exists between all CQNs and is
shared with other application specific control and data plane traffic.

3.1. Multihop Entanglement

```
resulting entanglement
+---+  +---+  +---+
|A+-----------------+B+-----------------+C|
```

Node B properties:
- Number of Communication Qubits
- Number of Storage Qubits

Node capabilities (operations):
- Swap comm <-> storage
- Entanglement swap
- [(Distillation scheme, time)...]

In the figure above, an example request for an entangled pair between nodes A and B will be affected by the following properties:

- A chosen combination of F(idelity) and t(ime) duration to produce an entanglement at the respective Fidelity. These parameters roughly equate to the quality of the link, the accuracy with which the nodes can use the link, and the delay in classical networking.

- The actual capability of nodes A and B to make use of the communication qubits.

A new EP creation between CQNs B and C will similarly be affected by the same parameters as above.

```
resulting entanglement
+---+  +---+  +---+
|A+-----------------+B+-----------------+C|
```

```
|A+-----------------+B+-----------------+C|
```

Node B entanglement swap operation

And finally, with an entanglement swap operation at node B (which is a node specific capability and has a specific duration) we end up with an A-C EP:

```
resulting entanglement
+---+  +---+  +---+
|A+-----------------+B+-----------------+C|
```

Node B entanglement swap operation

3.2. Distillation

If a pair of CQNs A and B share a number of EPs of insufficient quality, they may be combined into a single EP of higher quality by distillation. To do so, these CQNs need to agree on which distillation scheme to use before distillation can proceed. This does not necessarily need to be via communication between A and B, if one agrees upon a deterministic procedure of selecting one. This document suggests the following procedure:

1. A and B look at the distillation schemes that both advertise in common.
2. If there is none in common, stop. Distillation is not possible.
3. If there is a non-trivial subset in common, the first scheme in the node with the lower router ID is to be used by A and B.

Given a chosen distillation scheme \((S, t, p)\), an additional time delay will be added for the actual operation: For a 2:1 distillation scheme between nodes A and B, 2 EPs need to be produced followed by an operation on A and B that produces 1 EP. This operation will take time some expected time \(t\), and succeed with probability \(p\).

\[
\text{2:1 distillation } (S, t, p) \\
\text{+---------------------------+} \\
\text{|A+-------------------------+B+------------------------+C|} \\
\text{+-+ A-B Link properties ++} \\
\text{[(F1,t1), (F2,t2)]} \\
\]

3.3. Node Properties

We are interested in exposing the properties of CQNs (including QENs) to allow sophisticated decision making, for example in the creation of entanglement. These properties include:

1. Number of communication qubits. The number of communication qubits determines the number of entangled pairs that the node can produce simultaneously.
2. Number of storage qubits
3. Possible operations, along with their execution time and probability of success:
   1. Swap between communication and storage qubits
2. Entanglement swap

3. List of supported distillation schemes (in order of preference).

Note that several other parameters can be advertised, such as the T1 and T2 times for a qubit’s decoherence. These are omitted for now, instead just giving the decay of the fidelity of an EP. If deemed useful, T1 and T2 times can additionally be advertised.

3.4. Link Properties

A list of (Fn, tn) pairs describing the tradeoffs of a possible entanglement produced by two nodes (the ends of said link): tn is the time to produce an entangled pair with fidelity Fn.

4. The (Ab)use of Protocols

The routing protocols IS-IS and/or OSPF could be used in order to advertise entanglement capabilities. This section describes the additional data fields needed in order to facilitate the objective.

4.1. A Brief Primer on Link-state Protocols

This document suggests the use of a link-state protocol to distribute the capabilities of CQNs to create entanglement. This section offers a short introduction to link-state protocols for those not familiar with them.

Consider a directed graph $G=(V, E)$ with vertices (nodes) V and edges (links) E. Consider also $G'=(V', E')$; there is a 1-1 mapping from V’ to V and from E’ to E such that $e_1' = (v_1', v_2')$ is in E’ iff $e_1 = (v_1, v_2)$ is in E and $v_1'$ maps to $v_1$ and $v_2'$ maps to $v_2$. $G'$ represents the quantum network; V’ represents the set of CQNs, and E’ the set of quantum links between pairs of CQNs; G represents a classical network parallel to G’; that is, each CQN v’ has a corresponding classical node v. v plays a dual role: it is the control node for v’, and proxies on behalf of v’ in the link-state protocol.

The basic objective of a link-state protocol is to "flood" properties of nodes and (directed) links to all nodes in the network. This is accomplished by means of "link-state advertisements" (LSAs) that each node originates and sends to its immediate neighbors. The neighbors in turn send received LSAs to their own neighbors; this process repeats until every node receives every LSA (hence the term "flooding"). The focus of LSAs is the link properties (hence _link-state_ advertisements), although node properties are also advertised.
There are mechanisms to prevent looping of LSAs, and for reliable flooding. There is also a sequence number by which a more recent update of an LSA can be identified as such, and a mechanism for "aging out" LSAs belonging to nodes no longer in the network. In what follows, quantum node and link properties are added to the link-state advertisements of the corresponding classical node. Note that link properties need not be symmetric; that is, the link properties of \((v, w)\) need not be the same as those of \((w, v)\).

The net result of flooding is that every node has the same picture of the network (modulo LSAs in flight); in particular, each node knows the overall topology and connectivity of the network, and can use this information to make decisions. In a classical network, such a decision could be to compute a shortest path; for the quantum network, it could be choosing a feasible path (i.e., sequence of CQNs) for a multihop entanglement. Note that a node doesn't really know when it has complete and up-to-date information about the network; LSA updates may be originated at any time. Usually, this is okay: for example, if a node \(v\) learns enough of the network to have a path to another node \(w\), it can compute a multihop entanglement to \(w\). Subsequent updates may provide a more optimal (or higher probability) entanglement path. There are heuristics one can apply to guess that the link-state database (LSDB) (i.e., the union of all LSAs) is complete-ish; however, as nodes (and links) can fail or disconnect, there really is no such thing as "the full LSDB".

Each node \(v\) is identified by a "router ID" (an IP address uniquely allocated to \(v\)), denoted by \(\text{rid}(v)\). A link \(L = (v, w)\) is identified by \((\text{rid}(v), i)\) where \(i\) is an index allocated by \(v\) for \(L\) unique for each link emanating from \(v\). \((L\) may also be identified by IP addresses, but we'll ignore that for now.) It is generally expected that a directed link \((v, w)\) is matched by a link \((w, v)\); if not, \((v, w)\) is ignored from subsequent consideration; in particular, no link properties are advertised for this link by \(v\). Note that a pair of nodes may have multiple links between them; for simplicity, the notation will not be extended to indicate this. We'll assume \(\text{rid}(v') = \text{rid}(v)\) and the index allocated to a quantum link \(e'\) is the same as that of the corresponding classical link \(e\).

Let \(v, w\) be a pair of neighboring nodes, and let \(L_1 = (v, w)\) and \(L_2 = (w, v)\) in \(E\) be directed links in opposite directions between \(v\) and \(w\) with identifiers \((\text{rid}(v), i_1)\) and \((\text{rid}(w), i_2)\) respectively (where \(i_1\) is the index allocated for \(L_1\) by \(v\), and similarly for \(i_2\)). As a first step in running a link-state protocol, \(v\) runs a "hello protocol" all its links; in particular, over \(L_1\). Similarly, \(w\) will run the hello protocol over \(L_2\). The hello protocol serves to exchange the indices \(i_1\) and \(i_2\), and thus identify \((\text{rid}(v), i_1)\) as the reverse link of \((\text{rid}(w), i_2)\). This allows both \(v\) and \(w\) to correlate
the link properties of L1 and L2. If the hello protocol fails
between v and w, neither node includes link properties for the link
in their LSAs.

Once the hello protocol has been run on all links, v starts the
process of generating and sending its own LSA over all its links, and
of receiving the current LSDB from its neighbors. Note that an LSA
originated by v must propagate unchanged across the network; only v
is allowed to change it (and such a change must be accompanied by
updating the LSA’s sequence number). Such an update is triggered by
a new link coming up, an existing link going down, or a node or link
property changing.

IS-IS and OSPF are in principle similar, although the details of the
protocol mechanisms and encodings vary. In both protocols, a Type-
Length-Value (TLV) is used to encode most node and link properties.
In IS-IS, TLVs are used for all properties, and a single type of LSA
is used; in OSPF, there are several types of LSAs, and many (but not
all) properties are encoded as TLVs.

[1] has examples of "standard" LSAs for routing; [4] has the so-
called Traffic Engineering LSAs.

4.2. Node Properties

Here, we give a protocol-independent description of quantum node
properties; later documents will specify the encoding specifically
for IS-IS and OSPF.

Note that the following list of node properties is a strawman; all
details are subject to change, and other properties may be added as
needed.

The following node properties are added to the appropriate LSA:

<Qubit-TLV><NCQ><NSQ>
<CS-Swap><Prob><ExecTime>
<Ent-Swap><Prob><ExecTime>
<Measure><Prob><ExecTime>
<NDistSch><DistScheme1><DistScheme2>

4.3. Link Properties

Only one link property is listed. It gives the time-fidelity
tradeoffs of an entanglement operation as a list:

<N-Ent-TO><time1><fid1><time2><fid2>...
This is interpreted as follows: an entanglement operation may be initiated between nodes v and w over link (v, w). Depending on how fast one wants to complete (time-i), the list gives the corresponding fidelity of the resulting entanglement (fid-i). time-i is given in nanoseconds; fid-i as a number between 0 and 999999. The denominator is 1000000.

Note that this link property is symmetric, as entanglement is initiated simultaneously at v and w.

5. Security Considerations

It is not anticipated that adding these extensions to IS-IS and OSPF will present new security hazards to those protocols. Since however a common application of entangled pairs is for security purposes (such as QKD), it is worth investigating whether this application places a higher burden of security on the underlying protocols.

6. Acknowledgments

The authors would like to thank the following people for their contributions and support to this document: Vesna Manojlovic, Marcello Caleffi and Rodney Van Meter. Kompella would also like to thank Bruno Rijksman for encouraging him to learn about Quantum Computing and Networking.

Also:

7. Editorial Comments

It could be worth investigating the use of a distance-vector routing protocol to limit flooding.

8. IANA Considerations

There are no requests as yet to IANA for this document.
9. References

9.1. Normative References


9.2. Informative References


Authors’ Addresses

Kireeti Kompella
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA  94089
USA

Email: kireeti.kompella@gmail.com
Melchior Aelmans
Juniper Networks, Inc.
Boeing Avenue 240
Schipol-Rijk, PZ 1119
The Netherlands

Email: maelmans@juniper.net

Stephanie Wehner
QuTech
Van der Waalsweg 122
Delft, LC 2611
The Netherlands

Email: s.d.c.wehner@tudelft.nl

Cristian Sirbu
Redbit Networks
Dublin
Republic of Ireland

Email: cristian@redbit.network

Axel Dahlberg
QuTech
Van der Waalsweg 122
Delft, LC 2611
The Netherlands

Email: E.A.Dahlberg@tudelft.nl
Abstract

Near-term quantum networks will grow to form a Noisy, Intermediate-Scale Quantum Internet (NISQI). Connection setup will require adapting behavior along the path to the noise levels of individual elements. In this proposal, path creation is triggered by an application at the Initiator, information is accumulated node-by-node on an outbound pass in a series of QCaps (quantum capability) blocks, then the RuleSets are created at the Responder. RuleSets are installed at the individual nodes on the return pass. This document describes the architecture of connection setup in a network. Details of the RuleSets and QCaps, addressing architecture, link protocols, routing, resource allocation (multiplexing), extension of this setup procedure to an internetwork, and extension to multiparty communications are beyond the scope of this document.
1. Introduction

Building a connection across a quantum network [theqi] is a classical task. Because of the low success probability of quantum communication due to photon loss and the extremely high error rates due to the fragile nature of quantum information, quantum communication between two nodes more closely resembles a coordinated computation distributed among the set of nodes forming the path.
between the two nodes than a store-and-forward network session [qnetworking].

Use of the quantum network is driven by applications running at two (or more) classical nodes. Overall behavior is similar to client-server computing. The connection is initiated from a node similar to client and responded to by a node similar to a server. The details of the sending and receiving of the classical messages are not specified in this document, but can be modeled as if being sent over a TCP socket. Messages are assumed to be reliable and delivered in order. These messages have no hard real time requirement, though the subsequent data phase of the operation may.

This connection setup process must collect information about the hardware (channels and buffer memories) to be used, because of the heterogeneity of the underlying hardware. Loss in optical channels naturally varies with channel length and other factors, and has a large impact on quantum communication performance. Individual quantum buffers holding quantum bits (qubits) will vary in quality, as well.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Concepts and Glossary

The following terms will be used:

Bell pair a common form of entangled quantum state useful in communications.

End node a quantum network node with a single interface.

Entanglement the condition of a group of qubits (typically two qubits in this document) in a shared state that cannot be described using only real, non-negative, classical probabilities.

Entanglement Swapping executed at node B splices an entangled state shared with node A to an entangled state shared with node C, creating A-C entanglement and disentangling B from both nodes.

Fidelity a measure of the quality of a quantum state; roughly, the probability that the system holds the desired state.
Initiator  the initiator of the classical process of establishing the
connection by sending a message toward the Responder.

Purification  an error detection mechanism on quantum states.
Typically, one quantum state is used to test the condition of
a second state; the first state is destroyed in the process.
If the purification fails, it is unknown whether the first or
second state was in error, and the second state is discarded
as well. If purification succeeds, our confidence in the
state is improved.

QCap    an information block describing the quantum capabilities of a
particular node and link.

Qubit   a quantum system with two states that can be stored in memory
or transmitted through a channel, manipulated in a
constrained set of operations, entangled with other qubits,
and measured.

Repeater a quantum network node with a two interfaces, typically
sitting in the middle of a chain.

Responder is the endpoint of the connection setup process, where the
message sent by the Initiator terminates. The Responder
creates the RuleSets for all nodes in the path, and commonly
will be the smarter node.

Router   a quantum network node with a more than two interfaces,
requiring routing capability.

RuleSet describes the actions that a nodes should take when certain
conditions occur. The contents of RuleSets are beyond the
scope of this document.

The terms "source" and "destination" are not appropriate at the
connection level in a quantum network, because distributed quantum
states are not necessarily used for the unidirectional transfer of
information. Therefore, we use Initiator and Responder to designate
roles in the connection setup process, but those roles do not not
necessarily correspond to any asymmetry during the connection
lifetime. "Source" and "destination" may be used to describe the
movement of an individual classical message.

Links are assumed to be point-to-point. Multidrop physical layers
are possible, but quantum broadcast or multicast are not directly
possible at the physical level, and would have to be emulated.
3. Connection Setup Phases

3.1. Short Description of Phases

The single-network, two-node connection setup procedure consists of three basic phases:

1. The outbound request is routed from Initiator to Responder using a standard NextHop-based forwarding table, accumulating information about the path along the way in a stack of QCaps.

2. When the request arrives at the Responder, the Responder uses that information to create a complete RuleSet for every node. The RuleSets are assembled into a stack with the nearest node at the top.

3. The RuleSets are sent back along the original path, with each node removing its RuleSet from the message (popping the stack), then forwarding the remaining QCaps on until it returns to the Initiator.

3.2. Rationale for this Architecture

The outbound pass collects information about the nodes and links, to be used by the Responder to formulate the RuleSets. Why is the information collected in this fashion rather than shared more broadly across the network, e.g. as part of a modified routing protocol such as OSPF [RFC2328]? Why does a single node create the RuleSets for all nodes, rather than allowing individual nodes to create their own RuleSets when they see the PathSetupRequest message?

1. Because Repeaters may be spaced as closely as every 10km, a full topology for a network listing every Repeater may be excessively large for routing purposes, but such information is needed for building RuleSets.

2. The information collected may be substantially larger in volume than simple link costs.

3. The information collected and used may be too dynamic for a routing protocol.

4. Sharing of this information can be unnecessary when routing is driven by policy decisions rather than technical capabilities.

5. Centralization of the RuleSet creation is necessary because all RuleSets must cooperate toward a single goal, and the correct
breakdown of responsibility cannot be determined from partial information.

6. Centralization of RuleSet creation allows a Responder to upgrade its policies independently and to improve the process if its developers have found better tuning mechanisms. A distributed mechanism would require that all nodes in the path upgrade at the same time to avoid the creation of inconsistent policies, and limit the ability of Responders (often service providers of some sort) to innovate.

4. Message Contents and Elements

This section outlines the principal information to be carried in the messages. Detailed packet formats are beyond the scope of this document, and may vary from network to network.

4.1. PathSetupRequest

At minimum, the PathSetupRequest message must contain:

1. node addresses for the Initiator and Responder
2. the class of service requested [qiroadmap]
3. minimum performance parameters (fidelity and throughput)

4.2. Quantum Capabilities

A QuantumCapabilities block to be added to the stack in the PathSetupRequest message describes the functions, performance and quality of the node and link. This may include:

1. the fidelity of Bell pairs created by the quantum channel
2. the fidelity of local operations performed by the node for purification or entanglement swapping
3. the rate at which entanglement can be created (Bell pairs per second)

The details of the required information may differ between networks. A standardized form of this information for sharing between networks will be used for internetworking operation.
4.3. RuleSets

A RuleSet block in the stack in the PathSetupResponse message describes the rules to be executed at each node. A rule consists of a Condition clause and an Action clause. A Condition clause lists the existence of particular entangled states, or the reception of particular messages. The Action clause describes the actions of purification, entanglement swapping, or even discarding an entangled state, as appropriate. The details are beyond the scope of this document.

5. Processing the SetupRequest

5.1. Initiating a Connection Setup Request

An Initiator, driven by an application request for quantum network services between itself and the Responder, builds the PathSetupRequest, populates the first QCap block, selects the next hop, and sends the request. Note that there is no need for either the Initiator or the Responder to know the entire network topology, only be able to select a next hop appropriately. The details of the routing are beyond the scope of this document.

5.2. Outbound Processing

Creation of the RuleSets requires knowledge of the number of nodes involved. A quantum node adds its own address when receiving the request packet, before sending to the next node. The stack size indicates how many nodes are involved. Additionally, the RuleSet creator may require information regarding links between nodes along the path - e.g. to be used when optimizing the order of entanglement swapping.
The pseudocode below outlines the processing on receipt of the PathSetupRequest message.

procedure ProcessFlatPathSetupRequest(Msg)
  Msg.HopStack.Push(MyHopInfo)
  if (MyAddr != Msg.ConnSpec.Responder)
    // Process and forward
    NextQuantumHop = GetNextQuantumHop(Msg.ConnSpec.Responder)
    LinkInfo = GetLinkInfo(NextQuantumHop)
    Msg.HopStack.Push(LinkInfo)
    Forward(NextQuantumHop,Msg)
  else
    // have reached the far end, need to build RuleSets
    // for everybody, then return
    ReturnMsg = ProcessFlatPath(Msg)
    MyRuleSet = ReturnMsg.RuleSetStack.Pop()
    InstallRuleSet(MyRuleSet)
    NextQuantumHop = ReturnMsg.RuleSetStack.Top.Addr
    Forward(NextQuantumHop,Msg)
  endif
endprocedure

Note that although we use the term "NextQuantumHop" here, that refers to a neighboring quantum node, and does not imply that the classical node’s neighbor is necessarily the same; it could, in theory, pass through multiple nodes to get there.

5.3. Responder Processing

The Responder accepts the final PathSetupRequest message with the complete stack of information about node capabilities and links, and builds a corresponding stack of RuleSets, one per node in the path. The details of this creation process are beyond the scope of this document, and may be kept secret from other nodes in the path.

5.4. Return Processing
The pseudocode below outlines the processing on receipt of the PathSetupReturn message.

```
procedure ProcessFlatPathSetupReturn(Msg)
    MyRuleSet = ReturnMsg.RuleSetStack.Pop()
    InstallRuleSet(MyRuleSet)
    If (ReturnMsg.RuleSetStack.Size != 0)
        NextQuantumHop = ReturnMsg.RuleSetStack.Top.Addr
        Forward(NextQuantumHop,Msg)
    endif
endprocedure
```

The RuleSetStack should only be empty after the "Initiator" node of the original request removes its RuleSet, so this should be followed by initiating the connection.

6. Rejection and Robustness of the Setup Process

6.1. Rejection by a Repeater or Router

A repeater or router that receives a PathSetupRequest may reject the request if it has no quantum communication resources available. It should not reject the request simply because it believes the requirements of the request (fidelity or rate) to be difficult to fulfill; that responsibility lies with the Responder.

When a node rejects the PathSetupRequest, it shall inform the other nodes along the portion of the path that have already received the PathSetupRequest by creating a PathSetupResponse message with an error code that indicates failure and sending that message to the node on the top of the stack. As with a successful PathSetupResponse, the list of nodes to which the message must be sent is created as a stack. Other than the addresses and the error code, the message may be empty; no RuleSets are required. The message is then iteratively returned, with each node popping its own address and forwarding to the next.

6.2. Rejection by a Responder

A Responder may reject a PathSetupRequest for any reason:

1. As with any classical system, it may simply choose to reject the request for any service-related reason, such as security, licensing, etc.

2. It may determine that the request cannot be fulfilled with the resources offered by nodes in the path.
When a node rejects the PathSetupRequest, it shall inform the other nodes along the path by creating a PathSetupReturn message with an error code that indicates failure and sending that message to the node on the top of the stack. As with a successful PathSetupResponse, the list of nodes to which the message must be sent is created as a stack. Other than the addresses and the error code, the message may be empty; no RuleSets are required. The message is then iteratively returned, with each node popping its own address and forwarding to the next.

6.3. Robustness

As the rate of connection initiation increases, competition for resources will also increase. A soft reservation mechanism that temporarily allocates resources in the anticipation of reception of a RuleSet may be used, with the reservation timing out and resources being released if no RuleSet arrives within a certain period. Specification of this mechanism is beyond the scope of this document.

Deeper integration of routing with real-time availability of resources is beyond the scope of this document.

7. Contributors

Besides the authors, Luciano Aparicio, Clement Durand, Dominic Horsman, Shota Nagayama, Takahiko Satoh, Shigeya Suzuki, Amin Taherkhani, and Joe Touch have made substantial contributions to the network architecture and the concepts described here.

We also thank Chia-Hung Chien, Kaori Ishizaki, Bill Munro, Kae Nemoto, Takafumi Oka, Shinnosuke Ozawa, and Thaddeus Ladd.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

Security implications of this entire process are extensive.

To minimize the probability of tampering, each information block added to the request on the outbound leg should be signed by the node adding the block.

Each information block describes hardware configuration, and therefore inherently leaks information about the network topology and condition. This document addresses only connection setup within a single network. Internetwork connection setup will require
mechanisms to limit the leaking of sensitive network information across organizational boundaries.

Likewise, each RuleSet should be signed to prevent tampering during the PathSetupResponse phase.

Both the Request and Response phase may be encrypted using appropriate public key mechanisms.

It is also known that quantum networks may be vulnerable to attacks not possible in classical networks. These concerns are beyond the scope of this document.

10. References

10.1. Normative References


10.2. Informative References


Authors' Addresses

Rodney Van Meter
Keio University
5322 Endo
Fujisawa, Kanagawa 252-0882
JP

Phone: +81-46-649-3529
Email: rdv@sfc.wide.ad.jp
Takaaki Matsuo
Keio University
5322 Endo
Fujisawa, Kanagawa 252-0882
JP

Phone: +81-46-649-3529
Email: kaaki@sfc.wide.ad.jp