Big Thing Network™

Patent Pending: 63/494,792

Robert Carl Rice

First Preliminary Draft

What is this App?

This app is a Demonstration / Test Environment / Proof of Concept for a new type of computer network service named the *Big Thing Network™* . What does the app do? Not much, just a cute animation, unless you have an interest in computer network services like the Internet.

The app opens with a scene containing a preset configuration of four network nodes, dispersed randomly, each having one transmit tower and one receive antenna. The transmitters are not broadcast and are linked to the receive antenna of one other node forming a unidirectional ring topology network.

The ring topology is a good test for the adaptation algorithms as all logical network connections will require intermediate relays in one or both directions. Unlike most existing network services, this new network service can adapt to efficiently use unidirectional communication links.

The animation begins with each node periodically sending a set of “Discovery” packets first identifying itself and then forwarding a limited number of Discovery packets that it has received and buffered from nearby nodes. When a node receives one of its own Discovery packets, a transmission loop exists such that a connection attempt to each other node in the loop is likely to succeed resulting in a bi-directional logical connection.

As local connections are established through discovery, application servers will use them to establish longer connections. Activity increases exponentially until all desired connections are established. The network then returns to an idle state sending only periodic Discovery packets and periodic Connect packets used to map shorter paths if possible.

Basic network algorithms are coded in Apple’s new language “Swift”, with animation rendered in a SceneView. The user interface enables the scene to be edited at any time. The app implements only the basic connection and adaptation logic required for the demo / proof-of-concept.

Abstract:

A new packet switched Mobile Ad-hoc Network (MANET) service combines the adaptability of Internet protocols with the higher efficiency and security enabled by packet switched network technology. Adaptation is more deterministic than Internet adaptation.

With network technology transmission efficiency can be improved by maintaining state information for logical connections at intermediate network relay nodes. Older network protocols are relatively unsophisticated and make inefficient usage of transmission bandwidth to minimize memory requirements of network nodes. However, with the current generation of inexpensive microcontrollers having huge memory addressing capability, the ideal tradeoff swings to support the development of a more efficient network protocol. Further, Internet Protocols were developed without the requirement for network security, especially from "Denial of Service" attacks. A newer network protocol can incorporate the Public Key Infrastructure (PKI) to build security into the network rather than depending upon only end-to-end encryption technology.

Self-configuration and adaptation are required for any new wide area network service. Even networks using planted nodes and fixed network topology becomes unmanageable beyond a limited number of nodes. Older packet switched network technology, e.g., CCITT X.25, had efficiency and security advantages as compared to IP but also had little capacity for adaptation. The difficulty in manually maintaining routing tables was a major failing that enabled IP technology to overtake packet switched network technology.

The obvious "Holy Grail" solution is to design a switched network service that can self-configure and adapt even better than IP. Lower packet overhead is achieved by replacing the long globally unique network routing portion of IP address with a short alias identifier unique only relative to each transmission link. This identifier corresponds to a destination network node and is translated to a new identifier at each intermediate network hop.

By definition, MANET networks must map alternate paths to reach a destination node in order to be able to adapt quickly to the failure of an intermediate node or communication link. Without good design memory requirements for connection mapping could grow exponentially. It is believed that the current invention manages connections in an efficient manner to limit memory requirements.

The new network service will further implement several innovative features such as supporting intermittent and transmit only links. Connection routing that can use any existing namespace, e.g., IPv6, IPv4, phone number, personal and business names, etc.

Statement Of Need:

The Internet was developed at a time when network security could be deemed as relatively unimportant, i.e., intended primarily for communications between academic and government research institutions and other collaborative efforts. A low level of security was also necessary to limit complexity and enable adoption using machines much less powerful than modern machines. However, current global usage of the Internet requires a much higher level of security. In particular, “Denial-Of-Service” attacks are difficult to counter while using Internet Protocol (IP). A new service can integrate the Public Key Infrastructure (PKI) throughout to counter many of the vulnerabilities of IP technology.

Adaptive and self-configuring networks are an intensely competitive area of computer science due to rapidly growing demand. The industry has named these networks “Mobile Ad-hoc Networks” (MANET), sometimes also referred to as “on-the-fly” or “spontaneous” networks, continuously self-configuring, self-organizing, infrastructure-less networks of mobile devices sometimes augmented by fixed static networks, i.e. a general mesh network with frequent dynamically varying topology.

The need to support highly mobile environments for robots, vehicles, sensors, the Internet-Of-Things, etc. has fueled the development of many competing MANET protocols. Most are based upon adaptations of Internet protocols with routing algorithms optimized to meet particular application requirements. For example, remote sensors have a limited battery lifetime. Sensor network routing must find divergent paths to share the power usage among nodes to extend the lifetime of the sensor network.

The Internet Engineering Task Force (IETF) maintains a number of these Internet based MANET protocols. Routing adaptation protocols are tweaked to best meet particular application requirements. But Internet Protocol (IP) may not be the best foundation to build a new MANET service on. IP was originally designed to be adaptive, but with the adaptation primarily for self-configuration of the network, not for fast and robust mobile adaptation. As a service expands to a large number of nodes, the overhead for both addressing and adaptation becomes very significant.

Information privacy can be added by encryption/decryption at endpoints but the network itself is not well protected from attack especially "denial of service" attacks. Packet switched networks did better in this regard by identifying users albeit this was primarily to support usage based billing.

The current invention will use Public Key Infrastructure (PKI) certificates throughout to control access, levels of service, usage based billing and provide other security features.

Overview of the Invention:

The *Big Thing Network™* service is designed to "behave" much like the Internet but with higher performance, reliability and security, especially from "denial of service" attacks. Although the behavior is similar to that of the Internet, the network service has an entirely different architecture that is open to multiple administrations.

A core packet switch defers to "Directory Service Agents" (DSAs) for routing of all data and usage based meta information connections beyond initial connections established with local nodes through a “Discover” process. DSA routing can support many existing namespaces. Examples include IPv6, IPv4, X.121, X.500, phone numbers, twitter hashtags, .com, .net, etc.

Each supported namespace is expected to have separate administration and these separate administrations will form a consortium to manage the core packet switch. This is roughly equivalent to the role of the Internet Engineering Task Force (IETF) to manage the Internet Protocol. Namespace administration is described in detail below.

Even without support for mobile network nodes, any new network service must be able to adapt efficiently to be manageable for a large number of users and over a wide area. This was a “failing” of prior packet switched networks which required manual updates to routing tables. Algorithms used in this invention to achieve this efficient adaptation are described in considerable detail within this document.

Similar to IP protocol, the *Big Thing Network™* uses a routing identifier to deliver to a receiving host computer and a logical channel identifier having significance only to the two connected host computers. The primary difference is a connection establishment protocol that maps a relatively small routing identifier, relative to each transmission link, for each destination host computer.

Unlike X.25, routing identifiers are assigned uniquely in each direction of transmission. This enables the use of unidirectional, i.e., transmit only, links and it enables the return path to be entirely different from the send path between two endpoint systems.

To avoid collisions in the assignment of alias identifiers, broadcast/multicast transmission links are supported as multiple point-to-point links with each transmitter/receiver pair being identified as if it were a separate transmission link. This is done in the "Discovery" process as described in more detail further on in this document.

By definition, a Mobile Ad-hoc Network service maps multiple paths to each destination host computer in order to adapt paths quickly in response to nodal or transmission link failure. The service must also be able to find new paths and adapt existing paths quickly as links are reestablished and as new transmission links are discovered.

For each transmission link, a routing identifier maps to a particular destination host computer. During connection establishment, intermediate nodes check if they already have one or more paths to the desired destination and, if so, then the new connection path is mapped to converge with the existing path or paths to that destination. The assignment and mapping of alias identifiers is ended but the packet continues to be forwarded to the destination node. This helps to prevent maps from growing exponentially but it also makes adaptation efficient. As local paths are adapted, longer paths that converge onto them are also automatically adapted.

The *Big Thing Network™* service uses sliding window link level acknowledgement and retransmission to avoid slower end-to-end transmission error recovery as is done with Internet protocol. The link level acknowledgement will often detect the imminent failure of a transmission link and reroute traffic before the link is actually lost.

Due to the high overhead for related hosts to establish their logical connection mappings, hosts will likely store mappings to a persistent memory device when network connectivity is lost. When network connectivity is restored, the hosts will verify the integrity match of their stored mappings before resuming from where they left off.

As stated earlier, the *Big Thing Network™* service can route connections to remote systems using a variety of existing namespaces. To accomplish this, the gateway server refers connection requests to the local DSA server that is managing the namespace used for routing.

For large namespaces, the local namespace server may again refers the connection to another remote system specializing in the portion of the namespace being referenced. Thus, connection requests may be forwarded by the referral process several times in rapid succession before reaching the desired destination. The path for newly established connection must be immediately optimized in both directions before endpoint servers are notified of the connection otherwise established paths would be excessively long and could pick up undeliverable loops. Endpoint nodes may be many hops away but can also end up as a local connection on the originating node.

DSAs run as independent processes on every network for every namespace used for connection routing. DSAs establish new connections using a mechanism supported by the switch which I have named “Referral”. Referral operations work in any namespace and are described in detail in following sections.

Local Discovery and Initial Connection Establishment:

Connections to nearby nodes, i.e. within a small number of transmitter hops, are established and optimized through a “Discovery” and “Connection” process.

At regular intervals, nodes transmit a Discovery or “Hello” packet through each of its transmit links to inform receiving nodes of its presence and it also forwards Discovery packets that it has received from other local nodes, i.e., within a limited number of transmitter hops.

For security, Discovery packets contain a randomly generated node identifier having a high probability of being unique within a neighborhood of two or three link level hops. A 32 or 64 bit random number is preferable to any information that could be used to identify the transmitter or its location. Each node will assign itself an identifier upon initialization and keep it until it is reset.

A link identifier assigned at the first receiving node and added to the packet. This identifier is a 16 bit hash of the originating node identifier, the receiving node identifier, the originating node transmitter identifier and the receiving node antenna identifier. The two nodes will then use this link identifier to treat the transmitter / receiver pair as if it were a point-to-point link in subsequent transmissions. This link identifier will then avoid identifier assignment collisions for broadcast transmission links having multiple receivers.

A transmission loop is detected when a node receives its own Discovery packet returned from another node. Once detected, it is likely that a bi-directional connection can be established with at least one and possibly all of the remote nodes it has learned about from received Discovery packets.

The switch will generate a “Connect” packet and send it on the successful transmit link for each newly discovered node and for each already connected remote node which may have a new path to reach it. Connect packets assign and map a unique destination alias identifier, relative to the link identifier, at the originating node. The packets contain:

the link identifier, assigned alias identifier, its own node ID and identifying certificate as the source, the prior hop node identifier if there is one, and the desired destination node ID.

For any connection attempt to succeed, any intermediate node or nodes must have already established a connection to the destination otherwise the connection packet is dropped. This is because the Connect packet does not carry the link identifier information needed to assign an alias identifier for the next hop. This information is provided only by the Discovery packets. The originating node will periodically resend the Connect packet, keeping the same assigned alias identifier, a limited number of times before giving up on the connection.

Intermediate nodes having a connection to the requested destination node will map the converging path and forward the Connect packet toward the destination.

A Connect packet successfully received at a the requested destination node is changed to a ConnectReturn packet and the packet is returned to the originating node. The return routing uses the same alias and mapping logic described for the Connect packet.

A ConnectReturn packet successfully received at the originating node is changed to a ConnectConfirm packet and sent back to the destination node. As with the Connect packets, ConnectReturn packets will be dropped if an intermediate relay node has not established a path back to the originating node. However, ConnectReturn packets are not periodically retransmitted, rather the process will wait for a retransmitted Connect packet.

Borrowing a term from Open Systems Interconnect (OSI), I have named the interface objects between the packet switch and the application level servers “Service Access Points (SAPs)”. Each SAP object contain state information for a logical connections identified by the combination of the remote host ID, a receive Logical Channel Identifier (LCI) and a send LCI. The objects provide a buffer re-sequencing packets arriving from multiple paths, adding end-to-end sliding window acknowledgement and retransmission.

Each node is initialized with the packet switch, a gateway server to manage initial logical connections both on the local node and with remote nodes. Logical Channel Identifier (LCI) 0 is reserved for the switch to automatically establish a connection between the local gateway server and the remote gateway server on a newly discovered node. A ConnectConfirm packet received at the destination node is assigned receive LCI 0. The switch creates a new SAP and notifies the gateway server with the SAP.

Node Initiation:

Each node is initialized with the packet switch, a gateway server to manage initial logical connections both on the local node and with remote nodes. The node will also start up servers provided by supported namespace administrators to provide connection routing, network monitoring and usage-based billing for each namespace.

The servers will each request the switch to establish a local connection with the gateway. The switch will return a SAP object representing the local connection to both the gateway server and the requesting server. As remote nodes are discovered, the gateway will use its SAPs to refer local servers to their peer servers on the remote node. The referral process is described in a following section of this document.

All other logical connections are established through a “Referral” process. Upon notification of a new host connection, the gateway server proceeds to inform the new host of the other hosts that it has a relationship with. It does this by requesting the switch to refer its new SAP to the gateway server SAP (LCI 0) for each of it’s other established relationships. It will also automatically generate referrals for many of its other local server connections including administration and Directory Service Agents (DSAs) such that these servers will be able to communicate directly with their peers.

The referral process is described in detail further on in this document.

Referral Operation:

All logical connections except for initial local area connections are established by a referral operation. A referring server, often a DSA, will request the switch to perform a referral operation referencing two or three SAP objects for existing connections.

Two of the SAPs provide logical connections to reach the two destination endpoints that are to be connected directly through a new logical connection. The operation is symmetrical and either or both existing connections may terminate at the local node. Typically one endpoint will be requesting a new connection and the other SAP connection has been established to receive referrals, i.e., a registration connection, however the switch does impose this constraint. The switch supports concurrent referral operations.

The optional third SAP provides a connection to a server to receive meta data to support usage based billing and network performance monitoring. After establishing the new end-to-end connection, the switch will establish a meta connection at each endpoint of the new connection before providing the endpoint SAP to the receiving server.

The new SAPs provide a reference to the SAP for the connection that was followed from the referring node to help in validating the other endpoint node. No SAP is provided to the endpoint servers for the meta connections. Each endpoint switch will automatically terminate the meta connection to generate the metadata. The meta connections will periodically report statistics for packet counts sent and received, error rates, connect time, etc. Metadata will not copy data sent by either endpoint server.

The logic to establish the two meta connections is virtually identical to the logic required to establish the requested data connection. Thus each referral operation requesting meta connections will perform three referrals before the operation is complete.

The operation begins with the referring node sending a notification of the requested referral to both endpoint nodes. The notification includes an identifying certificate provided by the namespace administrator for the opposite node to verify before accepting the operation. Note that servers are not given a chance to change their identity. The operation will then proceed only after both endpoint nodes have accepted it.

The acceptance packets will also contain a locally assigned receive LCI for the switch to use in creating and mapping a new SAP. This LCI is passed to other endpoint such that the new SAPs will have both the send and receive LCI assignments uniquely identifying the logical connection for the relationship. The send LCI will differ from the receive LCI as connections are established with different remote nodes.

The operation proceeds with the referral node sending a request to each endpoint node to establish the new logical connection to the other endpoint by creating and sending a "Referral" packet to follow the path that is a concatenation of the paths of both connections being referred. However, at the originating and each intermediate node, the switch will check if it already has a relationship with the destination host and, if so, the new path is mapped to converge with the existing path or paths and the packet is forwarded using the existing relationship.

Otherwise, the switch will begin a relationship with the newly discovered destination node, assign a new alias identifier for the hop and forward the referral packet following the reference path. The switch will further generate a Connect packet to send along the same path to continue to establish its own relationship with the newly discovered node. Other than locally discovered nodes, this is the only mechanism for nodes to learn of other network nodes.

Note that the path for the new data connection is immediately optimized in both directions. This immediate optimization is critical to the functioning of DSA routing where several referrals may be chained in rapid succession before finally reaching the desired destination. Otherwise paths could accumulate loops and fail to be deliverable.

A node receiving a Referral packet will change the packet to a "ReferralAcknowledge" packet and send it back to the originating endpoint.

A node receiving a ReferralAcknowledge packet will change the packet to a “ReferralConfirmation” packet and send it back to the destination endpoint.

A node receiving a ReferralConfirmation packet will create a SAP and notify the receiving server that a new connection has been established unless meta connections have been requested in which case the server would be notified after the meta connection is established. The new SAP can be immediately used for data transmission and other referral requests.

Link Level Acknowledgement:

As stated earlier, the *Big Thing Network™* uses sliding window link level acknowledgement to provide faster error recovery network path optimization. Link level acknowledgement is supported only for data packets. As described earlier, packets that establish new connections are acknowledged end-to-end. Data packets are also sequence numbered end-to-end for resequencing from multi-path delivery and sliding window acknowledged.

Because the service supports unidirectional transmitter links, with bidirectional physical link hardware treated merely as a special case of two unidirectional links, you may be wondering how link level acknowledgement packets are delivered back to the transmitting node.

During the routing of referral multi-hop connections, intermediate nodes my learn of the existence of other nodes and the switch will automatically generate Connect packets to establish a relationship with the newly discovered node. Multi-hop connections are enabled for data traffic only after all intermediate nodes have established their relationship with the prior transmitting node.

The switch generates and sends link level acknowledgement packets unsequenced using the established gateway connection to the transmitting node. Link level acknowledgement packets include the link identifier common to both the transmitter and the receiver. The receiving switch intercepts link level acknowledgement packets, rather than delivering to the gateway server, then searches for the object transmitting to that link identifier to complete the delivery.

Connection Clearing:

Packet switching trades off memory usage to reduce packet overhead. During connection routing, the new service will map many temporary connections. As with X.25, the protocol must be able to quickly recover alias identifier assignments and memory used for mapping the identifiers.

Because connections are mapped unidirectionally, clearing acknowledgment and confirmation would be more difficult to achieve. Therefore, it is assumed that clearing acknowledgment and confirmation will not be required. Further, it is assumed that memory resources will need to be recovered using idle timeouts both for links and end-to-end.

Link logic will automatically transmit a periodic idle packet barring no data transmission. If several idle packets go unacknowledged, then the link is deallocated. Otherwise, the idle packet acknowledgement is useful for monitoring the performance of the link. Mobile links can gain signal strength to become the preferred link for the transmitting node.

The packet switch will also automatically generate end-to-end idle packets for otherwise idle SAPs. Because some servers will maintain a large number of SAPs, idle packets would add unnecessary complication to server design.

Namespace Management:

As stated previously, the network can support many existing namespaces for connection routing. Examples include IPv6, IPv4, X.121, X.500, phone numbers, twitter hashtags, .com, .net, etc. Each supported namespace is expected to have separate administration. A administration may also support a “store and forward” message service for a namespace address when a real-time connection cannot be established.

At a minimum the administration must provide a Directory Server Agent (DSA) server for startup at each network node to aid the packet switch in connection routing for the namespace. On startup the DSA server will request the switch to establish an initial registration connection to its local gateway server identifying the namespace that it is supporting via certificate. The gateway will later use the SAP from this initial connection to refer connection requests providing an address within the namespace to the DSA. The gateway will also use this SAP to refer the DSA to the peer DSA of newly discovered nodes.

DSAs for each namespace must be able to ultimately resolve an address in the namespace to a particular destination server currently reachable in the network or a store and forward message service. To accomplish this the DSAs must accept a registration connections from a target server for every reachable address in the namespace.

For secure use of the namespace, administrations will act as a certification authority for the namespace. The administration will provide each registered user with a certificate containing the address within the namespace. DSAs for the namespace can then verify the certificate before allowing any connections to the address.

If every target server were to connect to the corresponding namespace DSA on every node, the number of connections would quickly become unmanageably large. Therefore, most namespaces will divide the address space into manageable portions of responsibility and distribute the portions to peer DSAs on remote nodes. For robustness, the DSA will maintain redundant servers distributed throughout the network for each portion in the event of node failure or loss of connectivity. Both registration connections will be referred to the appropriate redundant nodes.

The administration will likely want to support usage based billing and network performance monitoring for the namespace. To support this the administration will provide a startup server to run on each node to collect meta information from successful connections. Peer meta servers could be managed similar to the distribution of address space for DSAs.

Multicast Connections:

Internet routers support multicast connections as this is considered an important feature to reduce network traffic. The packet switch does not support multicast connections directly and adding this feature at the switch level would make securing the service more difficult.

However, the Internet reserves a portion of the IP address space for multicast connection support. This implies that multicast connections should be a responsibility of the namespace administrations.

Within the current architecture, multicast connections can be supported using a multicast server provided by the administration. The multicast server would buffer data received through a provider connection and relay the data to multiple consumer connections.

The DSAs responsible for the multicast portion of the namespace would establish optimal multicast relay connections at intermediate network nodes. Optimization of the multicast routing would also be a responsibility of the DSA servers. How an administration could achieve this path optimization is beyond the scope of this document.

Brief Background of the Invention:

Internet Protocol networks, also referred to as Connectionless (CL) networks, use a globally unique routing address to deliver each packet to a specific end computer system. As the Internet has grown exponentially, this routing address has become expanded to 48 bits or more out of a possible 128 bits for IPv6. Thus the routing address adds significant overhead to the length of each data packet.

Prior packet switched networks, e.g., X.25, also referred to as Connection-Oriented (CO) networks, significantly reduce packet overhead by using a connection establishment protocol to assign a Logical Channel Number (LCN) relative to each transmission link. Because network nodes supported a limited number of connections at any given time, this resolves to a small number that is translated at each relay node relative to each outgoing transmission link. However LCN mapping was single level. Applications on two remote nodes could establish multiple logical connections through the network consuming memory for each LCN mapping at intermediate network nodes.

To limit the complexity and memory requirements of this LCN mapping process, traditional packet switched protocols placed constraints on the network topology. For example, CCITT X.25 networks could use only bidirectional physical communication links. More significantly, the protocol had no capability for adapting network paths other than to clear and reestablish connections. Internet protocols support adaptation and place fewer restrictions on the network topology.

The current invention restricts logical channel mapping to remote peer pairs. For network packet routing, a single alias identifier, unique only to the outgoing transmit link, is assigned to represent the remote destination node at the originating and intermediate network nodes. This mapping mimics CL routing while minimizing the memory require for alias mapping.

The X.25 network protocol was popular at a time when the cost of CPU power and high speed memory was very high. Limitations to the complexity of the protocol was necessary for adoption. However, given the current generation of low cost, high performance and high memory capacity microprocessor based systems, the time has come to revisit packet switched network technology. With a significant upgrade to protocol sophistication, packet switched networks can add support for unidirectional transmission links, multi-path alias mapping for adaptability while reducing packet transmission overhead. In fact, network adaptation can occur even faster and more reliably as compared to Internet adaptation. It will even enable integration of intermittent very high speed transmission links, e.g., radar.

While older packet switched network services were limited to using bi-directional point-to-point physical links, this new service will accommodate broadcast / multicast links wherein network node antennas can receive transmissions from multiple transmitters. The network will treat each transmitter/receiver pair as if it were a separate unidirectional physical link.

This project will have to overcome a political barrier as CO packet switched protocols were originally developed in Europe. i.e., the Consultive Committee for International Telephone and Telegraph (CCITT), X.25 & X.75 while CL Internet protocols were developed by the Defense Advanced Research Projects Agency (DARPA). An intense rivalry has existed between CO and CL network services.

Lacking self configuration capability, CO X.25 based network services, e.g., GTE Telenet, became unwieldy and difficult to manage beyond a certain number of users. But perhaps the major reason for the failure of packet switching in the U.S. was the decision by DARPA to grant free and open access to the Internet. It’s hard to compete with free taxpayer money!

Serious attempts have been made to adapt IP technology to mobile environments. Generally they have failed, e.g., the Aeronautical Telecommunication Network (ATN), as evidence that more sophisticated protocol suites are required to support mobile environments, especially wide area mobile environments, e.g., autonomous vehicles. It is anticipated that the current invention has the potential to serve these environments including both mobile and statically located network nodes.

Most MANET services, including the new Apple Multi-peer Connectivity Framework, are based on protocols maintained by the Internet Engineering Task Force (IETF) and inherit the basic technology of connection-less networking. The packet switched MANET service is fundamentally different in that routing uses no network addresses not even temporarily assigned addresses. In fact, for security, network nodes are generally denied any information about remote nodes or the current network topology other than the number of hops or delivery time. Directory services may use this information to delegate the responsibility for portions of the database to nearby nodes while replicating as necessary for redundancy.

The most notable CO network protocol X.25 was specified by the Consultive Committee for International Telephone and Telegraph (CCITT) but is now maintained by the International Telecommunications Union Telecommunication standardization division (ITU-T). X-25 uses a connection establishment procedure to map long network addresses to locally significant Logical Channel Numbers (LCN). LCNs are assigned relative to each physical link and mapped to change with each network hop. The small LCN significantly reduces the packet size overhead of the network service.

However, the single layer mapping of X.25 LCNs could not be adapted to perform satisfactorily in a mobile environment due to the number of LCNs that would need to be remapped for changing physical links. X.25 has other limitations unsuitable for the mobile environment such as the requirement for all physical links to be bidirectional. Broadcast radio links would be utilized very inefficiently.

The most notable CL network is, of course, Internet Protocol (IP). IP has the ability to adapt network paths but generally too slowly for highly mobile environments. Using IP, data packets carry full network addresses which have been extended in length (128 bit IPv6) due to the growth of the Internet. Thus IP packets must be relatively long for the packet overhead to not consume a significant portion of the physical link bandwidth. The larger packet size is a problem for potential mobile applications that are limited to small packets of data.

CL IP technology has virtually eliminated X.25 network service in the US. This is partly due to the manual administration of X.25 routing tables which became problematic for the GTE Telenet service as the network grew. It became clear that adaptability was a requirement even for large static networks. However, the failure of the Telenet service more likely resulted from the government's decision to open Internet access to the public.

Initially, Internet access was limited to government and universities. But as other network services overtook Internet usage, particularly by the phone company sponsored Fidonet modem network, the government didn't like being number 2. So Internet access was made free because IP lacked the capability to identify and track user usage - an advantage of the other service providers at the time. But it was hard for a private companies to compete with a free service.

With the ability of the Telenet service to identify and track user usage, the service had a security advantage over the Internet. At the time, the Internet Engineering Task Force (IETF) was claiming that necessary network security could be added at the endpoints using encryption but the threat of "denial-of-service" attacks to the service was underestimated. The Telenet service was much more resistant to such attacks.

Clearly the "Holy Grail" of computer networking was to marry the desirable features of CO and CL network while overcoming the limitations of both types of networks. This Big Thing Network technology has the potential to achieve this marriage.

Patent Claims:

For provisional patent application I claim:

1) A network service qualifying as a Mobile Ad-hoc Network Service (MANET) including self configuration and rapid adaptation to changing network topology and further comprised of:
2) Connection establishment using a variety of common namespaces including Internet, phone number, name and address, etc.
3) Support for a separate administration of each supported namespace.
4) Secure access via Public Key Infrastructure certificates to verify user identity.
5) Support for usage based billing by namespace administrations.
6) A lowered transmission overhead by mapping longer namespace addresses to short alias identifiers. Given the memory capacity of current microprocessors and the still increasing use of limited bandwidth links, e.g., satellite, the tradeoff is favorable for using additional memory to shorten transmissions.
7) A lowered transmission overhead by grouping information packets destined for the same remote node into transmission frames.
8) Support for one-way, i.e., transmit only physical links
9) Support for intermittent physical links, i.e., scanning radar and microwave.
10) A mechanism to efficiently adapt logical channel transmission paths without requiring end-to-end packet delivery.

Phase 1 Implementation:

A phase 1 “Proof Of Concept” should focus on an application requiring fast and efficient adaptation for a limited number of network nodes in a small geographic area. This could possibly be a sensor network.

Given that the initial development and test environment is the Swift programming language running on Mac OS, perhaps the most convenient phase 1 platform will be otherwise decommissioned iPhones. Apple provides an excellent Xcode development environment for ios devices and used iPhones readily support a wide variety of communication applications and interfaces to other devices.

The phase 1 “Proof Of Concept” should demonstrate a high efficiency and reliability before proceeding with another phase of development. A possible strategy for proceeding is to add an Internet “Bridge” to the service enabling the technology to be quickly deployed in environments where Internet protocols have been demonstrated to be insufficient, e.g., aero-mobile and satellite. Other bridges would be useful for various environments and applications.

The strategy to build bridges to existing network services follows the original rollout strategy for Internet Protocol. At the time many companies were competing in the development of private network services but then the need arose for these private networks to interoperate. The Defense Advance Research Projects Agency (DARPA) stepped in with the Inter-Network Protocol and built bridges to various private network services. Eventually companies, including Apple, phased out their private network protocols in favor of extending Internet routers and switches direct to devices.

Phase 1 development could be accomplished by a small but efficient team representing a small risk to investors, but with the possibility of a high return on investment.

Frame Header Formats

Packets sent to the switch on a local host connection are delivered immediately to a destination server receive queue without the addition of a frame header, i.e., the Null frame header.

Many physical transmission links have higher efficiency with large frame sizes. The *Big Thing Network™* optimizes use of these transmission links by batching packets destined to the same node into a transmission frame prefixed with an appropriate frame header.

Packets sent to the switch are sorted into separate queues for batching into frames. A single Discovery packet queue exists for each node. A Link queue exists for each established logical link. Unsequenced and sequenced packet queues exist for each relationship the node has with a remote node.

Discovery packets have a unique Discovery frame that do not designate a logical link to enable multiple receivers in a broadcast environment.

A link frame contains packets that must be decoded and processed by the receiving node, i.e., the frame is not automatically relayed by a Relationship LCN. These packets are for link control and connection establishment.

Unsequenced frames typically contain connection establishment and periodic time sensitive messages.

Sequenced frames add sequence numbering for reliable delivery. This is the majority of network data packets.

Frame headers also vary by transmission link type. Broadcast and multi-drop links add a Link Identifier to uniquely identify a transmitter/receiver pair. The Link Identifier is generated during the Discover process by the node receiving the transmission. It is a hash of information from both the transmitting and receiving nodes such that it is highly likely to uniquely identify the transmitter / receiver pair. Each pair is then treated as if it were a separate physical link.

Frame contents vary depending upon the frame type

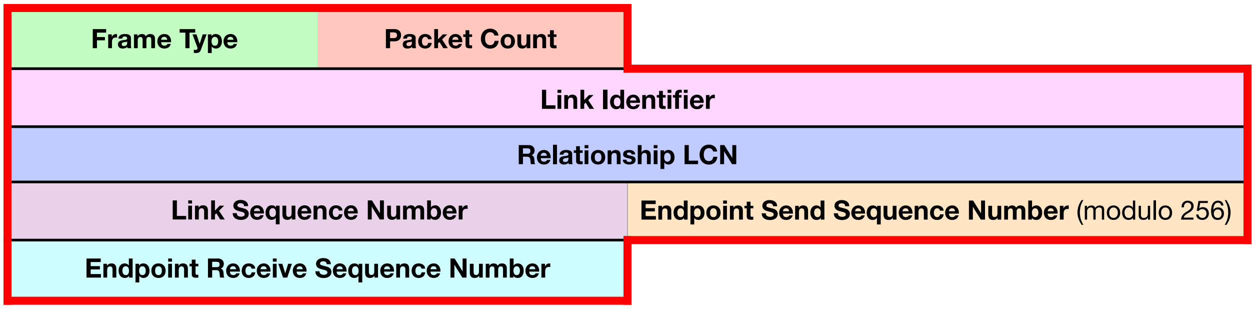
Frame Type	Link Identifier	RelationshipLCN	Sequence Numbers
Discovery	No	No	No
Point-To-Point Link	No	No	No
Point-To-Point Connect	No	Yes	No
Point-To-Point Unsequenced	No	Yes	No
Point-To-Point Sequenced	No	Yes	Yes
Multi-Drop Link	Yes	No	No
Multi-Drop Connect	Yes	Yes	No
Multi-Drop Unsequenced	Yes	Yes	No
Multi-Drop Sequenced	Yes	Yes	Yes

Link Identifier is a 16 bit hash of transmitter and receiver node information to uniquely identify a transmitter receiver pair in a broadcast or multicast link. It is omitted on a point-to-point link and also omitted for a Discovery broadcast transmission.

RelationshipLCN is a 16 bit Logical Channel Number that the switch maps to route peer-to-peer packet frames between two network nodes. It is assigned relative to each link and independent of direction.

Sequence numbering is modulo 256 sliding window. Sequence numbering includes both a link level, i.e., local one way, and end-to-end send/receive sequence numbering.

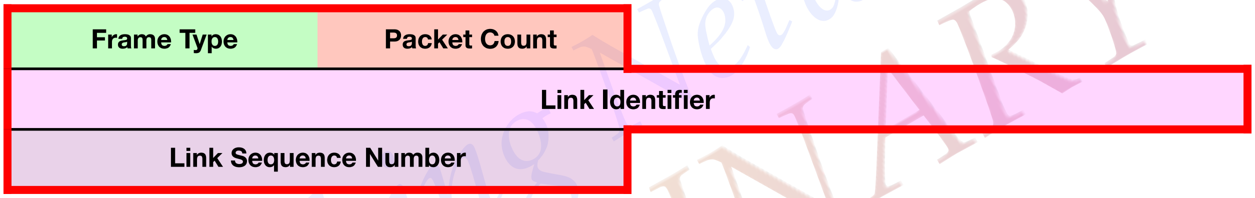
Sample Frame Formats: Frame headers are bordered in red.



Frame Type	**Packet Count**
Link Identifier	
Relationship LCN	
Link Sequence Number	**Endpoint Send Sequence Number** (modulo 256)
Endpoint Receive Sequence Number	

MultiDrop / Multicast Sequenced Frame Header

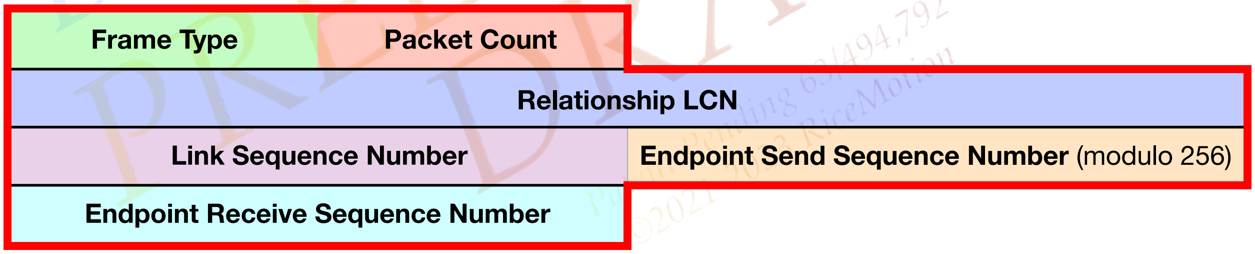
Relationship LCN is mapped to relay at intermediate nodes to reach the destination node. This frame header is used for the majority of data transmission.



Frame Type	**Packet Count**
Link Identifier	
Link Sequence Number	

MultiDrop / Multicast Link Frame Header

A Link Frame Header carries packets that must be processed by the receiving node, i.e., the frame cannot be automatically forwarded by a Relationship LCN mapping.



Frame Type	**Packet Count**
Relationship LCN	
Link Sequence Number	**Endpoint Send Sequence Number** (modulo 256)
Endpoint Receive Sequence Number	

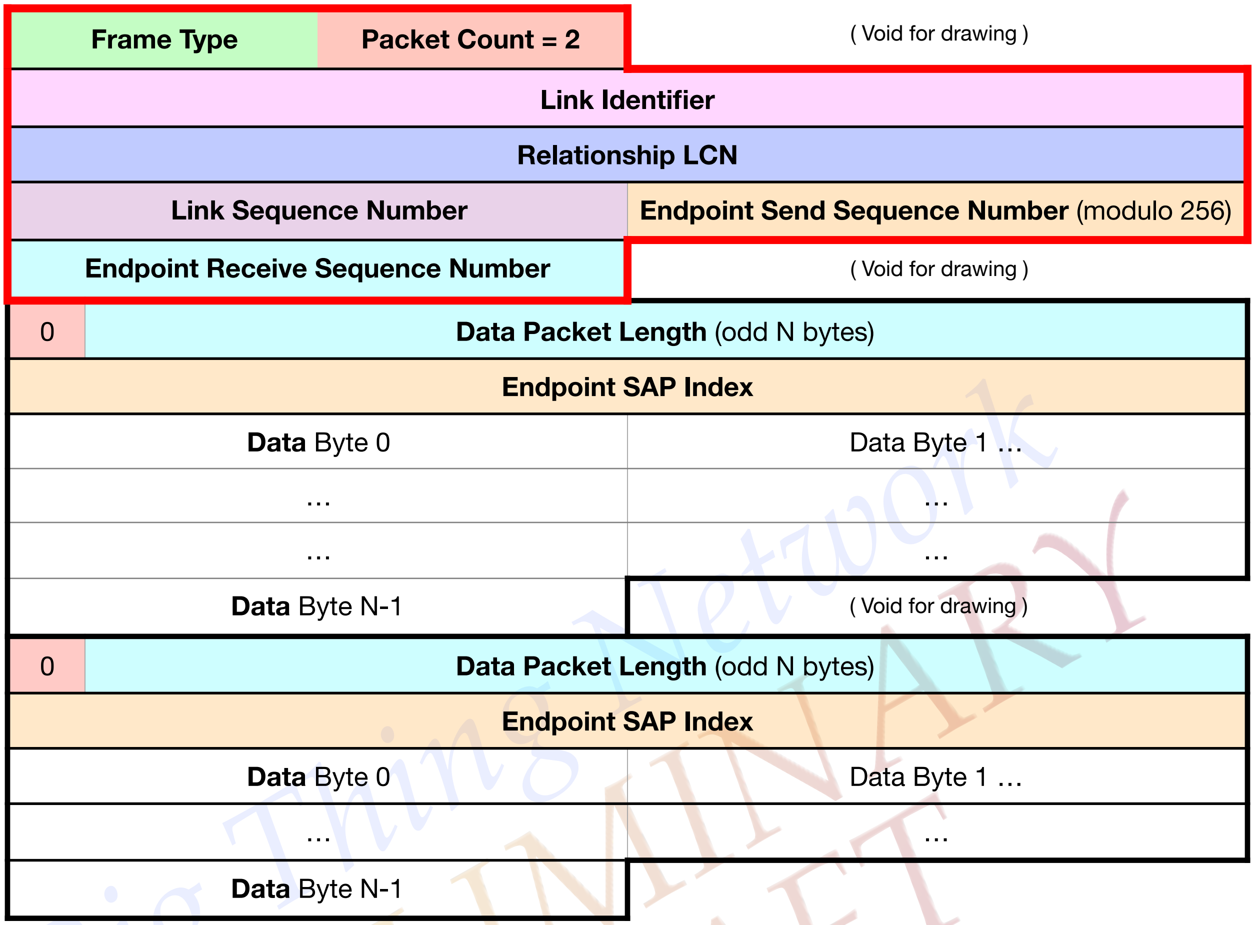
Point-To-Point Sequenced Transmission Frame Header

The Link Identifier can be eliminated or set to zero for a point-to-point physical link.

Frame Type	Packet Count

Discovery Frame Header

Point-To-Point Link Frame Header



Frame Type | **Packet Count = 2**	(Void for drawing)
Link Identifier	
Relationship LCN	
Link Sequence Number	**Endpoint Send Sequence Number** (modulo 256)
Endpoint Receive Sequence Number	(Void for drawing)
0 | **Data Packet Length** (odd N bytes)	
Endpoint SAP Index	
Data Byte 0	Data Byte 1 …
…	…
…	…
Data Byte N-1	(Void for drawing)
0 | **Data Packet Length** (odd N bytes)	
Endpoint SAP Index	
Data Byte 0	Data Byte 1 …
…	…
Data Byte N-1	

Transmission Frame - Non-Broadcast / Non-Multicast Link

(Sequenced frame with 2 data packets destined to same node)

Packet Formats

1*	**Control Packet Type**	**Control Packet Length**
Data Byte 0		Data Byte 1 …
…		…
…		…
Data Byte N-1		

General Control Packet

* A leading 1 indicates a Control packet type.

0*	**Data Packet Length** (even N bytes)	
Endpoint SAP Index *2		
Data Byte 0		Data Byte 1 …
…		…
…		…
Data Byte N-2		**Data** Byte N-1

Data Packet

* A leading 0 indicates a Data packet

*2 At remotes nodes of a network connection, the switch maintains a table of Service Access Point object which are used to interface with application servers.

Discovery Frame

<table>
<tr><td colspan="2"><b>Frame Type</b></td><td><b>Packet Count</b> = 3</td><td>(Void for drawing)</td></tr>
<tr><td>1</td><td colspan="2"><b>Discovery Packet Type</b></td><td><b>8</b></td></tr>
<tr><td colspan="4"><b>Source Node ID</b> (randomly assigned) *1</td></tr>
<tr><td colspan="3"><b>0</b> (Hops)</td><td><b>Source Transmitter ID</b> *2</td></tr>
<tr><td>1</td><td colspan="2"><b>Discovery Packet Type</b></td><td><b>16</b></td></tr>
<tr><td colspan="4"><b>Source Node ID</b> (forwarded packet)</td></tr>
<tr><td colspan="3"><b>Hops</b> (incremented from forwarded packet)</td><td><b>Source Transmitter ID</b> *2</td></tr>
<tr><td colspan="4"><b>First Hop Node ID</b> (Added at first hop - Hops = 1)</td></tr>
<tr><td colspan="4"><b>Link Identifier*</b> (first hop)</td></tr>
<tr><td>1</td><td colspan="2"><b>Discovery Packet Type</b></td><td><b>16</b></td></tr>
<tr><td colspan="4"><b>Source Node ID</b> (forwarded packet)</td></tr>
<tr><td colspan="3"><b>Hops</b> (incremented from forwarded packet)</td><td><b>Source Transmitter ID</b> *2</td></tr>
<tr><td colspan="4"><b>First Hop Node ID</b> (Added if Hops = 1)</td></tr>
<tr><td colspan="4"><b>Link Identifier*</b> (first hop)</td></tr>
</table>

Discovery Frame (Unsequenced with 3 packets)

(Generated and transmitted periodically)

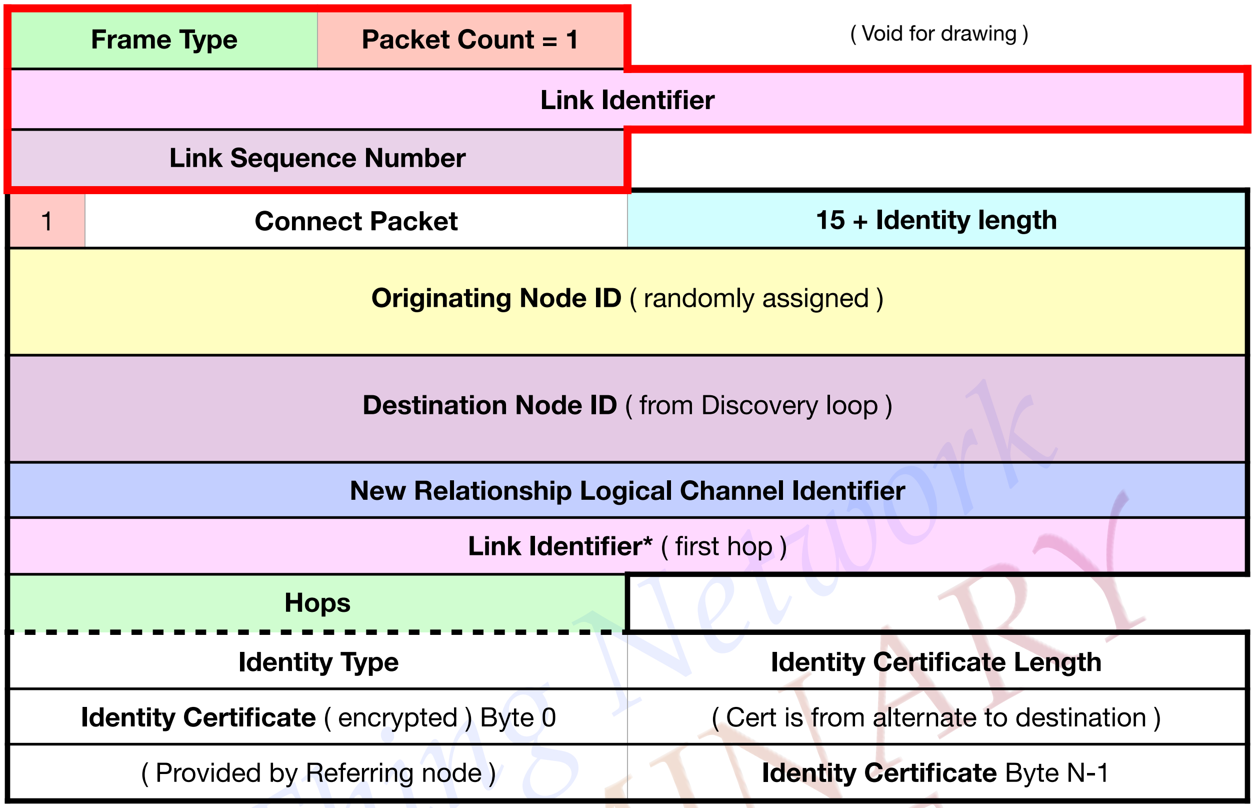
Forwarded Discovery Packets are added up to max frame size sorted by Hops

*1 Node IDs may be expanded to 64 bits to increase the probability of being globally unique

*2 A transmitter identifier is added by the originating node of a Discovery packet and used to match with the successful transmit link upon return.

* A Link Identifier is assigned at the first hop of the Discovery packet and used in succeeding transmissions to uniquely identify a transmitter/receiver pair for a broadcast or Multi-drop physical link.

Retransmitted periodically until acknowledged



Frame Type	Packet Count = 1	(Void for drawing)
Link Identifier		
Link Sequence Number		
1	**Connect Packet**	**15 + Identity length**
Originating Node ID (randomly assigned)		
Destination Node ID (from Discovery loop)		
New Relationship Logical Channel Identifier		
Link Identifier* (first hop)		
Hops		
Identity Type		**Identity Certificate Length**
Identity Certificate (encrypted) Byte 0		(Cert is from alternate to destination)
(Provided by Referring node)		**Identity Certificate** Byte N-1

Connect Packet (Unsequenced)

(In link frame possibly with other link control packets)

Connect Packets build the frame relay maps that enable the Relationship Identifier to efficiently forward succeeding frames at intermediate network nodes without decoding packets.

* The first hop Link Identifier is used by the originating node to match a Connect Acknowledge packet to the successful originating transmit path.

Identity certificates are encrypted such that they can only be decrypted by a peer remote entity. The identity certificate on a Connect packet is used by a receiving Gateway server to verify that a remote node is an authorized network node.

Response from remote host to Connect

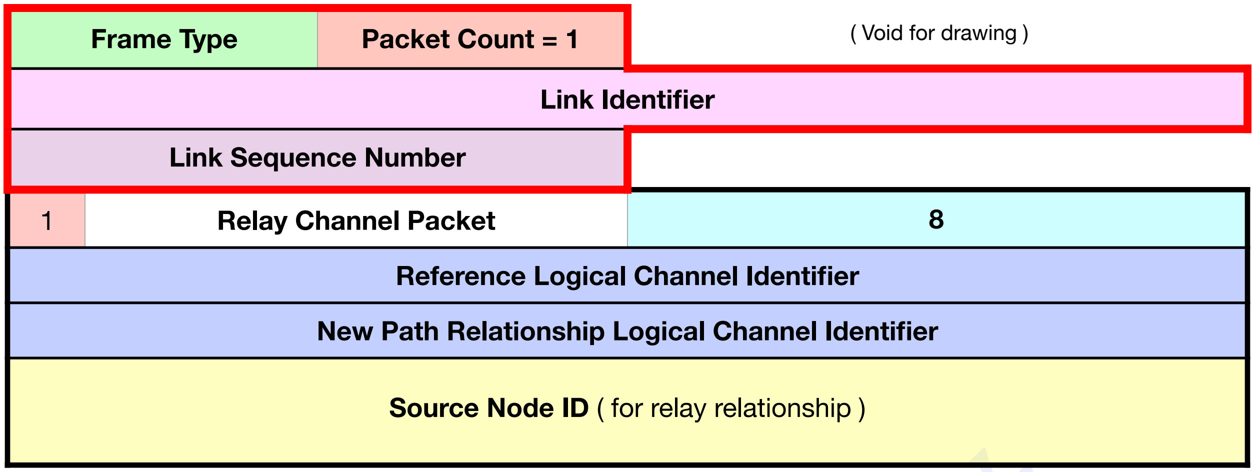
Frame Type	**Packet Count = 1**	(Void for drawing)
Link Identifier		
Link Sequence Number		
1	**Connect Acknowledge Packet**	**16 + Identity length**
Source Node ID (from Connect Destination, i.e., self)		
Destination Node ID (from Connect Source)		
New Relationship Logical Channel Identifier		
Link Identifier* (from Connect Packet)		
Delivery Hops		**Return Hops**
Identity Type		**Identity Certificate Length**
Identity Certificate (encrypted) Byte 0		(Cert is from alternate to destination)
(Provided by Referring node)		**Identity Certificate** Byte N-1

Connect Acknowledge Packet (Unsequenced)

(In link frame possibly with other link packets

* A Connect Acknowledge packet acknowledges a particular transmission link for the Connect packet originator. The packet helps to build the frame relay map for the Relationship Identifier in the return direction at each intermediate network node.

Identity certificates are encrypted such that they can only be decrypted by a peer remote entity. The identity certificate on a Connect Acknowledge packet is used by a originating Gateway server to verify that the accepting remote node is an authorized network node.



Frame Type

Packet Count = 1

(Void for drawing)

Link Identifier

Link Sequence Number

1

Relay Channel Packet

8

Reference Logical Channel Identifier

New Path Relationship Logical Channel Identifier

Source Node ID (for relay relationship)

Relay Path Packet (Unsequenced)

(In link frame possibly with other link packets

The Relay Path Packet enables local area network adaptation without generating the end-to-end network traffic that would result using the Connect/Connect Acknowledgement process only. As new transmission paths are discovered and acknowledged for a node to reach a remote destination, either by discovery or by referral operations, traffic being relayed from other source nodes to the same destination also have a possible new path. Because the new path is confirmed, new relay paths can be constructed without requiring end-to-end confirmation.

A Relay Path Packet is generated for each relay relationship referencing an already confirmed path to the same destination - somewhat more efficient than using the destination node ID. Upon receipt, the Relay Path Packet the receiving node maps the new path and acknowledges by returning a Quality of Service packet to the transmitting node.

Acknowledge for Frame Sequencing

Frame Type	**Packet Count = 1**	(Void for drawing)
Link Identifier (if multi-drop link)		
1	**Frame Acknowledge Packet**	**6**
Source Node Transmitter ID		(Void for drawing)
Acknowledged Link Identifier (≠ 0)		
Ack Sequence Number		**Nak Count**

Link Acknowledge Packet (Unsequenced)

(In unsequenced frame possibly with other unsequenced packets)

* Because the Link Identifier is assigned by the receiving node of the link, the ID is not guaranteed to be unique at the transmitting node. Adding the transmitter ID from the source increases the probability of uniqueness.

Quality Of Service Packet

Frame Type	**Packet Count = 1**	
Link Identifier (if multi-drop link)		
Relationship LCN (to transmitting node)		
Link Sequence Number		**Endpoint Send Sequence Number** (modulo 256)
Endpoint Receive Sequence Number		(Void for drawing)
1	**Quality Of Service Packet**	**6**
Source Node Transmitter ID		
Acknowledged Link Identifier (≠ 0)		
Relationship LCN (to receiving node)		
Delivery Time		**Send Hops**

Quality Of Service Packet (Unsequenced)

(In unsequenced frame possibly with other unsequenced packets)

Similar to the link acknowledge packet, this packet is sent from a link receiver to the transmitting node. Note, however, that it addresses a particular relationship relay logical channel rather than all transmissions on the link. It is likely that additional information, useful for network routing, will be added, e.g., surcharged satellite links.

Referral Node to Endpoint Node

Frame Type	**Packet Count = 1**
Link Identifier	
Relationship Logical Channel Identifier	
Link Sequence Number	**Endpoint Send Sequence Number** (modulo 256)
Endpoint Receive Sequence Number	(Void for drawing)
1 **Referral Request Packet**	**11 + N** (Identity length)
Meta **Operation ID**	
Endpoint SAP Index	
Alternate Endpoint Node ID	
Identity Type	**Identity Certificate Length**
Identity Certificate (encrypted) Byte 0	(Cert is from alternate to destination)
(Provided by Referring node)	**Identity Certificate** Byte N-1

Referral Request Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

The operation starts with a Request sent to both endpoints, i.e., fully symmetrical.

Endpoint Response to Referral Request

<table>
<tr><td colspan="3"><b>Frame Type</b></td><td><b>Packet Count = 1</b></td><td></td></tr>
<tr><td colspan="5"><b>Link Identifier</b></td></tr>
<tr><td colspan="5"><b>Relationship Logical Channel Identifier</b> (index to SAP)</td></tr>
<tr><td colspan="4"><b>Link Sequence Number</b></td><td><b>Endpoint Send Sequence Number</b> (modulo 256)</td></tr>
<tr><td colspan="4"><b>Endpoint Receive Sequence Number</b></td><td>(Void for drawing)</td></tr>
<tr><td><b>1</b></td><td colspan="3"><b>Referral Reject Packet</b></td><td><b>6 + N</b> (Datagram length)</td></tr>
<tr><td colspan="2"><b>Meta</b></td><td colspan="3"><b>Operation ID</b></td></tr>
<tr><td colspan="5"><b>Originator SAP Index</b></td></tr>
<tr><td colspan="4"><b>Datagram Length</b></td><td><b>Datagram</b> (Optional)</td></tr>
<tr><td colspan="4">(Void or datagram replaces the ID Cert)</td><td><b>Datagram</b> Byte N-1</td></tr>
</table>

Referral Reject Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

Datagram, when provided, improves response time by delivering a small packet of information to the alternate endpoint to process while connection establishment continues.

For Reject, Datagram may contain error information.

Endpoint Response to Referral Request

Frame Type	**Packet Count = 1**	
Link Identifier		
Relationship Logical Channel Identifier		
Link Sequence Number	**Endpoint Send Sequence Number** (modulo 256)	
Endpoint Receive Sequence Number	(Void for drawing)	
1	**Referral Accept Packet**	**6 + N** (Datagram length)
Meta	**Operation ID**	
Originator SAP Index		
Datagram Length	**Datagram** (Optional)	
(Void or datagram replaces the ID Cert)	**Datagram** Byte N-1	

Referral Accept Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

Datagram, when provided, improves response time by delivering a small packet of information to the alternate endpoint to process while connection establishment continues.

For Reject, Datagram may contain error information.

Referral Node to Alternate Endpoint Node

<table>
<tr><td colspan="2"><b>Frame Type</b></td><td><b>Packet Count = 1</b></td></tr>
<tr><td colspan="3"><b>Link Identifier</b></td></tr>
<tr><td colspan="3"><b>Relationship Logical Channel Identifier</b></td></tr>
<tr><td colspan="2"><b>Link Sequence Number</b></td><td><b>Endpoint Send Sequence Number</b> (modulo 256)</td></tr>
<tr><td colspan="2"><b>Endpoint Receive Sequence Number</b></td><td>(Void for drawing)</td></tr>
<tr><td><b>1</b></td><td><b>Referral Accepted Packet</b></td><td><b>7 + N</b> (Datagram length)</td></tr>
<tr><td><b>Meta</b></td><td colspan="2"><b>Operation ID</b></td></tr>
<tr><td colspan="3"><b>Endpoint SAP Index</b></td></tr>
<tr><td colspan="2"><b>Datagram Length</b></td><td><b>Datagram</b> (Optional carried from Accept)</td></tr>
<tr><td colspan="2">(Void or datagram replaces the ID Cert)</td><td><b>Datagram</b> Byte N-1</td></tr>
</table>

Referral Accepted Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

The Referring node waits for all Request responses before sending Accepted or Rejected packets.

For Rejected, Datagram may have error information.

Alternate Endpoint Response to Referral Node

<table>
<tr><td colspan="3"><b>Frame Type</b></td><td><b>Packet Count = 1</b></td><td></td></tr>
<tr><td colspan="5"><b>Link Identifier</b></td></tr>
<tr><td colspan="4"><b>Link Sequence Number</b></td><td></td></tr>
<tr><td><b>1</b></td><td colspan="3"><b>Referral Follow Packet</b></td><td><b>14</b></td></tr>
<tr><td colspan="2"><b>Meta</b></td><td colspan="3"><b>Operation ID</b></td></tr>
<tr><td colspan="5"><b>Followed Relationship Logical Channel Identifier</b></td></tr>
<tr><td colspan="5"><b>Originator SAP Index</b></td></tr>
<tr><td colspan="5"><b>Destination Node ID</b></td></tr>
<tr><td colspan="5"><b>New Relationship LCI</b> (set to 0xFFFF to indicate bypass mapping)</td></tr>
<tr><td colspan="5"><b>New Receiving SAP Index</b> (Becomes send SAP index at FollowThrough destination)</td></tr>
</table>

Referral Follow Packet (Link)

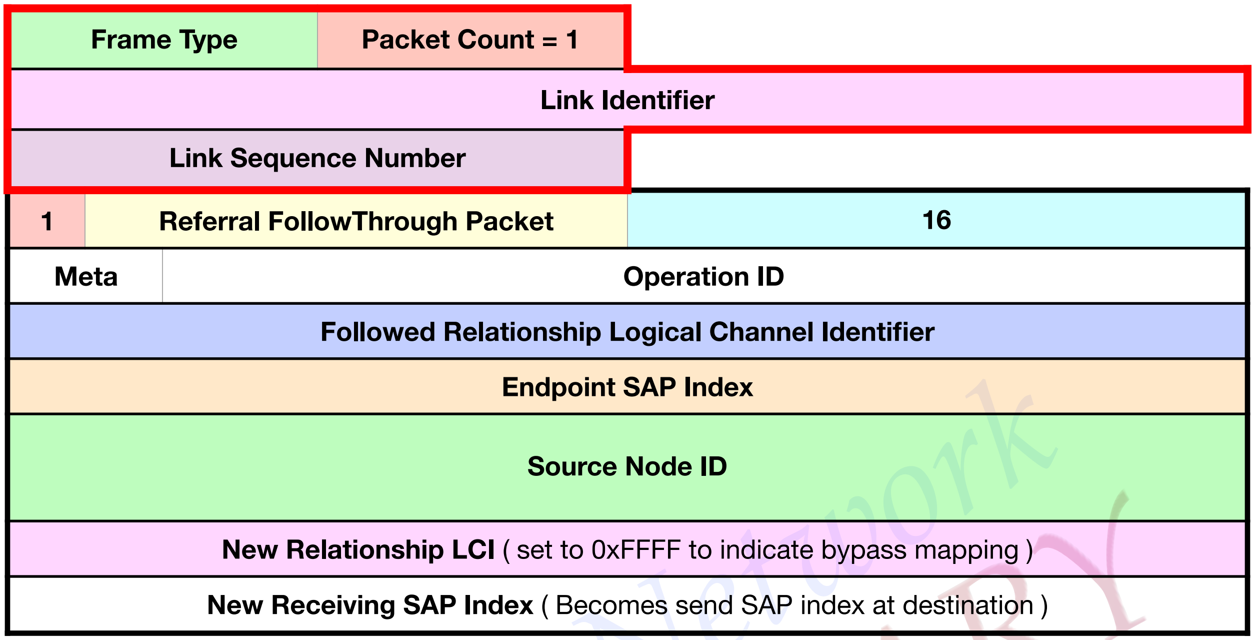
(in link frame possibly with other link packets)

A Follow packet is generated and sent upon receipt of an Accepted packet. It contains all information necessary to map the new connection through the network. Because the packet is mapping a new logical network connection, it is sent in a link frame to be processed at intermediate nodes.

* Bypass mapping flag. Set when Relationship with Endpoint is already established and LCI has been mapped to converge with destination relationship.

Note that the source node ID is known from the connection being followed, but the remote destination may be newly discovered by intermediate nodes.

Referral Node to Endpoint Node



Frame Type | Packet Count = 1

Link Identifier

Link Sequence Number

1 | Referral FollowThrough Packet | 16

Meta | Operation ID

Followed Relationship Logical Channel Identifier

Endpoint SAP Index

Source Node ID

New Relationship LCI (set to 0xFFFF to indicate bypass mapping)

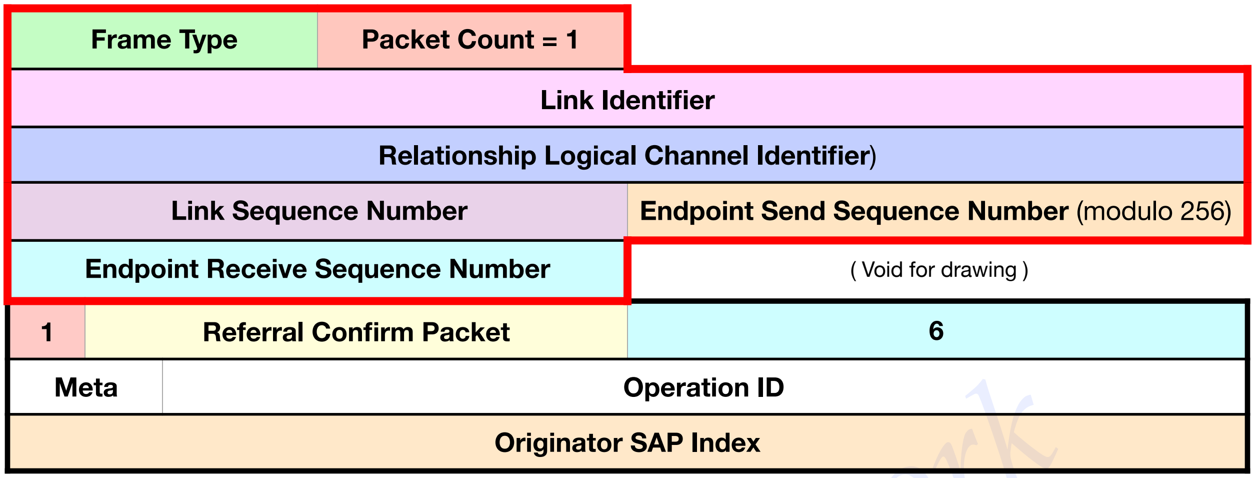
New Receiving SAP Index (Becomes send SAP index at destination)

Referral FollowThrough Packet (Link)

(in Link Frame possibly with other link packets)

FollowThrough continues to deliver the Follow packet to its destination endpoint while mapping the new connection unless bypass mode is set indicating that a relationship is already established for the remote relay. Note that the destination node ID is known from the connection being followed, but the remote source may be newly discovered by intermediate nodes.

Endpoint Response to Referral Node



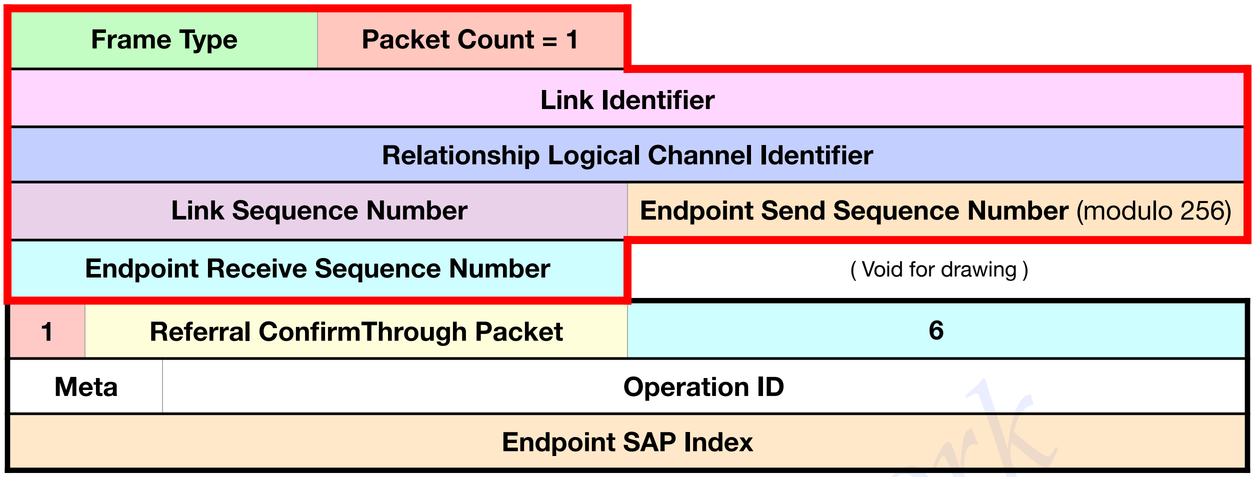
Frame Type	Packet Count = 1	
Link Identifier		
Relationship Logical Channel Identifier)		
Link Sequence Number	Endpoint Send Sequence Number (modulo 256)	
Endpoint Receive Sequence Number	(Void for drawing)	
1	Referral Confirm Packet	6
Meta	Operation ID	
Originator SAP Index		

Referral Confirm Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

A Confirm packet informs the referring node that the new connection is mapped toward the confirming endpoint.

Referral Node to Alternate Endpoint Node



<table>
<tr><td colspan="3">Frame Type</td><td>Packet Count = 1</td><td></td></tr>
<tr><td colspan="5">Link Identifier</td></tr>
<tr><td colspan="5">Relationship Logical Channel Identifier</td></tr>
<tr><td colspan="4">Link Sequence Number</td><td>Endpoint Send Sequence Number (modulo 256)</td></tr>
<tr><td colspan="4">Endpoint Receive Sequence Number</td><td>(Void for drawing)</td></tr>
<tr><td>1</td><td colspan="3">Referral ConfirmThrough Packet</td><td>6</td></tr>
<tr><td colspan="2">Meta</td><td colspan="3">Operation ID</td></tr>
<tr><td colspan="5">Endpoint SAP Index</td></tr>
</table>

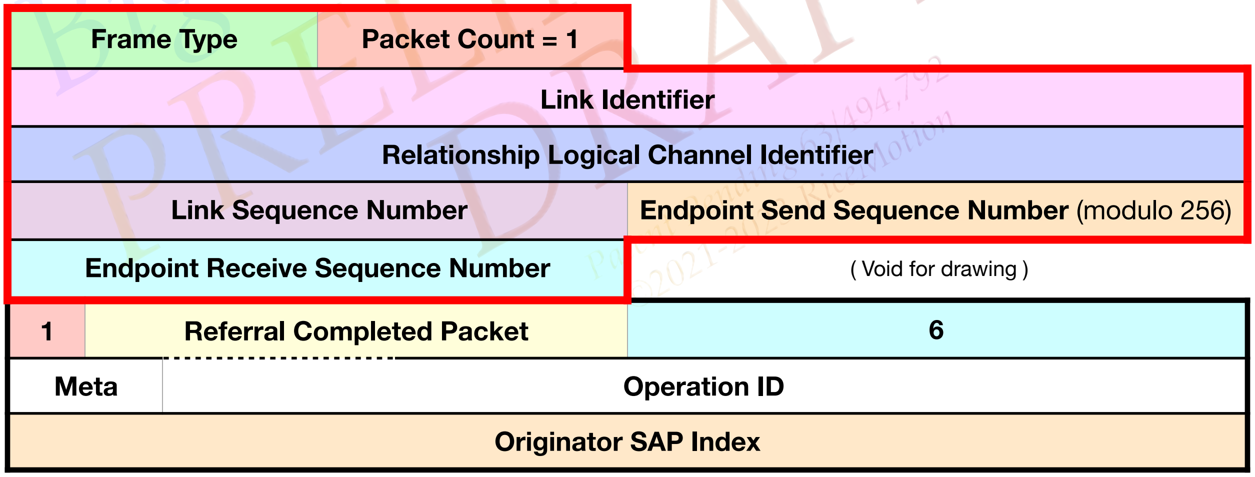
Referral ConfirmThrough Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

The referring node will wait for a Confirm from all endpoint before sending a ConfirmThrough to all endpoints.

A ConfirmThrough packet informs the endpoint nodes that the new connection is mapped in both directions, immediately path optimized and usable.

Alternate Endpoint Response to Referral Node



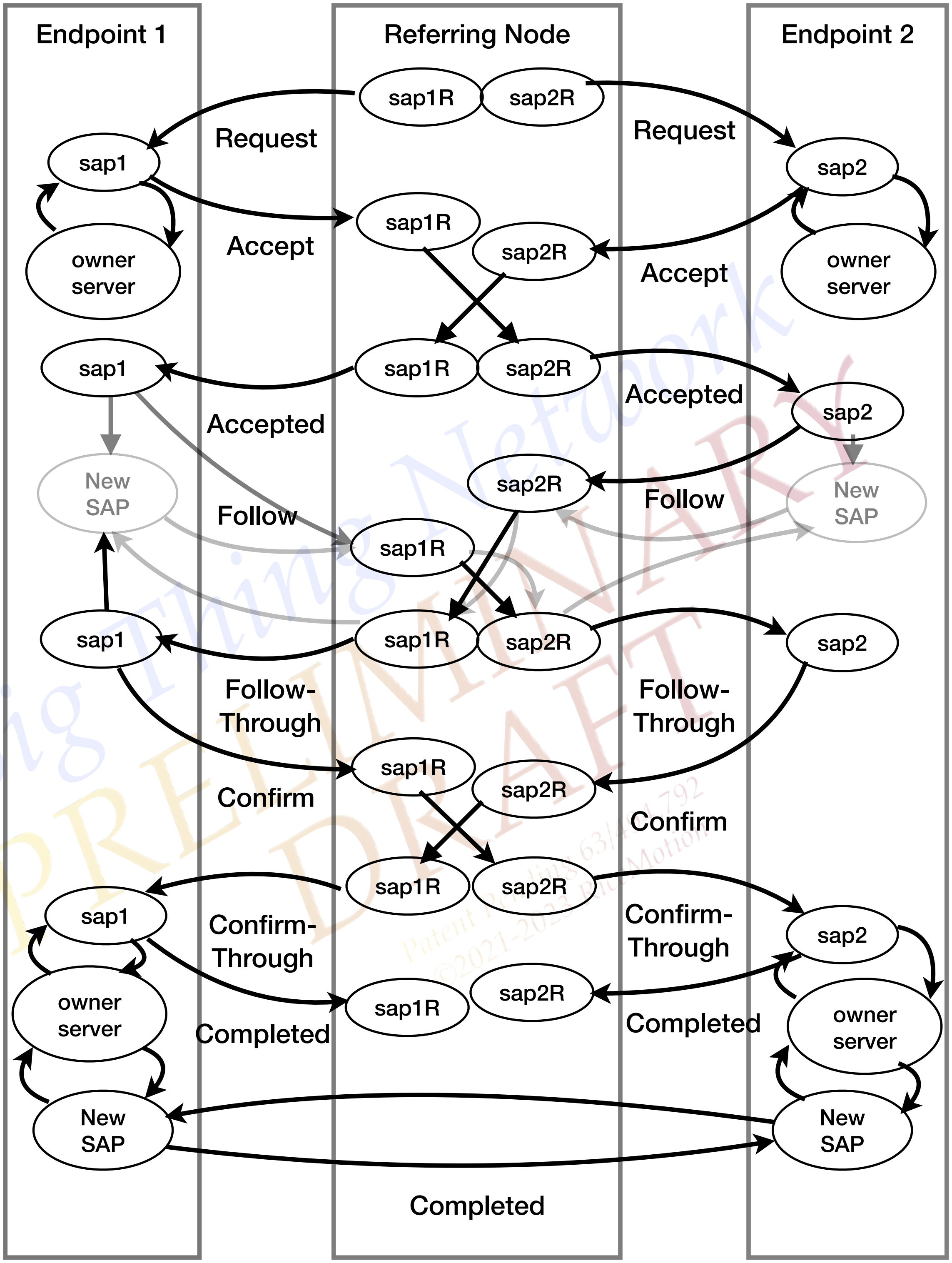
<table>
<tr><td colspan="3">Frame Type</td><td>Packet Count = 1</td><td></td></tr>
<tr><td colspan="5">Link Identifier</td></tr>
<tr><td colspan="5">Relationship Logical Channel Identifier</td></tr>
<tr><td colspan="4">Link Sequence Number</td><td>Endpoint Send Sequence Number (modulo 256)</td></tr>
<tr><td colspan="4">Endpoint Receive Sequence Number</td><td>(Void for drawing)</td></tr>
<tr><td>1</td><td colspan="3">Referral Completed Packet</td><td>6</td></tr>
<tr><td colspan="2">Meta</td><td colspan="3">Operation ID</td></tr>
<tr><td colspan="5">Originator SAP Index</td></tr>
</table>

Referral Completed Packet (Sequenced)

(in Sequenced Frame possibly with other sequenced packets)

The referring node will wait for a Completed packet from all endpoints before notifying the server requesting the referral of successful completion and freeing the operation ID for reuse.

Referral Timeline


Endpoint 1
Referring Node
Endpoint 2
sap1R
sap2R
Request
Request
sap1
sap2
owner
server
owner
server
sap1R
sap2R
Accept
Accept
sap1
sap1R
sap2R
sap2
Accepted
Accepted
New
SAP
New
SAP
sap2R
Follow
Follow
sap1R
sap1
sap1R
sap2R
sap2
Follow-
Through
Follow-
Through
sap1R
sap2R
Confirm
Confirm
sap1R
sap2R
sap1
sap2
Confirm-
Through
Confirm-
Through
owner
server
owner
server
sap1R
sap2R
Completed
Completed
New
SAP
New
SAP
Completed