The Topology Discovery Service

Richard Earnshaw

February 18, 1993

In a network consisting of many point-to-point links it becomes extremely difficult to generate broadcast messages for the purposes of address resolution so a different approach is needed. This consists of an independent service which monitors the topology of a subnetwork for changes and informs interested parties (such as routers) of any changes that it discovers.

1 Current Protocol

The current protocol is defined only for FDL networks. Machines running a routeing dæmon[1] broadcast a beacon at regular intervals to each of their neighbours indicating that they have routeing capability and give a SAP for obtaining further information; the response given depends on the class of machine receiving the beacon. A machine that is a trivial host should simply respond with a "trivial topology" message containing its own address; it plays no further part in any topology discovery. Machines that can gateway connexions respond by connecting to the published SAP and then conduct subsequent communications using the connexion. The beacon messages are transmitted on the promiscuous VCI.

1.1 Format of Fields in messages

Most of the fields used in the following messages are MSNL addresses and subnet masks; these fields have no 'endian-ness' and are transmitted as a block of four octets. Fields that are numbers, or that contain flags are transmitted in 'network endian' format.

1.2 Format of beacon messages

The official definition of the beacon messages is given in the section on common MSDL features; the message format given below gives the *current* usage of the reserved fields: *software should not be written that makes use of these fields without first consulting an authority, it is quite possible that they may yet change.*

The topology beacon is defined in Table 1: the interface MSNL number and subnet mask are those of the interface on which the beacon is being transmitted.[2] When a machine boots the router beacon messages are transmitted once every two seconds, backing off exponentially with each transmission until the interval is greater than a minute.

The "Trivial Topology" message format is shown in Table 2 and is currently restricted to advertising the primary MSNL number of the machine. It must only be transmitted in direct response to receiving a topology beacon, and must not be broadcast in any manner.

[1] Due to limitations in the current Wanda kernel the current version is part of the RARP dæmon.

[2] The Wanda kernel does not maintain MSNL numbers on a per-interface basis but as a table of alias numbers, so this binding has to be cludged; subnet numbers are not stored at all except in the routeing dæmon. The current solution is to define an environment string of the form
`TOPMSNL-<ifname> = <address>-<subnet_mask>`.

Octet	Field	Description
3	MSCommand	MSCMD_I_AM_ROUTER (12)
12–15	OrigAddr	Topology dæmon Primary MSNL address
16–19	OrigPort	Topology dæmon Port number
28–31	Argument	Interface MSNL address
32–35		Interface MSNL Subnet mask

Table 1: Topology beacon fields: fields not shown are the same as the standard FDL metasignalling format.

Octet	Field	Description
3	MSCommand	MSCMD_I_AM_HOST (13)
12–15	OrigAddr	Primary MSNL number of machine

Table 2: Trival topology response message: fields not shown are the same as the standard FDL metasignalling format.

1.3 Interconnecting between topology dæmons

When a topology dæmon (TD) receives a beacon on an interface it must connect back to the remote dæmon using that interface regardless of what the routeing table entries contain. This implies that the dæmon must be capable of circumventing the standard connexion routeing procedure and specify a route directly to the kernel; multiple connexions may exist between two machines if they are connected by different interfaces, indeed if there is an external loop-back then a machine may be connected to itself.[3]

Not more than one verified connexion may exist between any pair of interfaces at any one time. To ensure that this remains the case each new connexion undergoes a verification procedure to ensure that a simultaneous connexion is not being established in the reverse direction at the same time; only verified connexions may be used for topology discovery communications. The procedure used to collapse multiple connexions is that the machine emitting a connexion request (not the beacon) with the largest[4] primary MSNL number should keep its connexion. Loop-back connexions to the same machine should not be used for topology change discussions and their management is up to the implementation. The information exchanged with the beacon and connexion messages is not sufficient to resolve multiple connexions so the verification procedure is as follows.

[3]Such a configuration might be useful if, for example, a port controller were fitted with a multicast board, the routeing software would need to know about the loop-back connexion to take advantage of this.

[4]MSNL numbers are not naturally ordered. For the purposes of this definition the MSNL number $a0.a1.a2.a3$ is defined to be *less than* the MSNL number $b0.b1.b2.b3$ if either

$$a0 < b0$$

or

$$a0 = b0 \text{ and } a1 < b1$$

or

$$a0 = b0 \text{ and } a1 = b1 \text{ and } a2 < b2$$

or

$$a0 = b0 \text{ and } a1 = b1 \text{ and } a2 = b2 \text{ and } a3 < b3,$$

equal if all four octets compare equally and *greater than* otherwise.

If a TD receives a topology beacon and no connexion via that interface to the remote machine exists then a new connexion may be initiated. If a TD receives a connexion request and no verified connexion via that interface exists to the remote machine then the connexion shall be accepted and marked unverified, otherwise it shall be rejected. If the connexion is accepted then the TD that generated the connexion request transmits a parameters message as described in Table 3. The TD that accepted the connexion uses

Octet	Field	Description
0–3	remPrimary	Remote's Primary MSNL Address
4–7	remIface	MSNL address of remote's interface
8–11	remMask	Subnet mask of remote's interface
12–15	locPrimary	Local Primary MSNL Address
16–19	locIface	MSNL Address of local interface
20–23	locMask	Subnet mask of local interface

Table 3: Initial parameter message on Topology connexion.

the parameters message to complete fields that it may have missing and tests for multiple connexions according to the rule given above. If it rejects the connexion in preference of another it simply closes the rejected one, it does not mark the other as verified at this stage; otherwise it marks the connexion as verified and sends the message back as an acknowledgement. Once a TD has verified a connexion no further connexion attempts must be made on that interface pair until the existing connexion is deemed to have died (topology beacons may still be transmitted, however).

1.4 Messages on verified connexions

Once a connexion has been established and verified it may be used for topology change negotiations. In addition, "stay-alive" messages are also transmitted to detect any disconnexion that might occur. The format of the stay-alive message is given in Table 4. A

Octet	Field	Description
0–3	MsgClass	Message Class — TOP_PING (0)
4–7	locIface	MSNL Address of local interface

Table 4: Format of stay-alive (ping) message.

connexion may be deemed to have died if no message of any class has been received in two and a half stay-alive intervals: the current stay-alive interval is fifteen seconds.[5]

A TD that believes that the topology of a subnetwork has changed may initiate a reconfiguration on that subnetwork. The topology gathering method is almost identical to that used in Autonet [1]. To do this the TD transmits the message described in Table 5 on each of the connexions that it has in the changed subnet. The *epoch* is a stateful number that persists between reconfigurations; the epoch transmitted is one greater than the last known epoch that was used in configuration negotiations (it does not matter whether or not that renegotiation successfully completed). The epoch is used in the situation where multiple reconfiguration requests are active within the subnetwork at the same time: messages and state generated by messages of an epoch earlier than that believed

[5]All timeout intervals are still subject to (perhaps radical) tuning.

Octet	Field	Description
0–3	MsgClass	Message Class — TOP_RECONFIG (1)
4–7	rcfgType	Reconfiguration type — RCFG_INIT (0)
8–11	epoch	Reconfiguration epoch (see text)
12–15	rootMSNL	MSNL Address of root (see text)

Table 5: Reconfiguration initiation message.

to be current are ignored; messages with different roots (see below) are selected on the basis that the highest[4] root MSNL address wins.[6] The root MSNL address is the primary MSNL address of a TD initiating a reconfiguration sequence.

The action of a TD receiving a reconfiguration initiation message depends on its current state. If it has an epoch higher than that of the message then it ignores the message entirely. If the epochs are the same then if it is not currently a member of a reconfiguration tree or it is, but the root of the tree of which it is already a member has a higher MSNL number, then it also ignores the message; if the epoch number and root are the same as a tree to which it already belongs then it returns a Nack message (Table 6); otherwise it joins the new tree and sends an acknowledgement message (Table 7) back to the sender. Upon

Octet	Field	Description
0–3	MsgClass	Message Class — TOP_RECONFIG (1)
4–7	rcfgType	Reconfiguration type — RCFG_NAK (3)
8–11	epoch	Reconfiguration epoch
12–15	rootMSNL	MSNL Address of root

Table 6: Reconfiguration Nack message.

Octet	Field	Description
0–3	MsgClass	Message Class — TOP_RECONFIG (1)
4–7	rcfgType	Reconfiguration type — RCFG_ACK (1)
8–11	epoch	Reconfiguration epoch
12–15	rootMSNL	MSNL Address of root

Table 7: Reconfiguration Acknowledgement message.

joining a tree a TD sends the reconfiguration initiation message to all of the remaining neighbours in the same subnetwork. If a TD has no neighbours other than the one from which it received the initiation message, or it receives a nack from all such neighbours, then it is a 'leaf' node and it transmits one or more partial-data message back to its parent in the tree. The partial-data message can grow quite large, so the message may be split into multiple blocks for transmission, Table 8; the size of the message is dependent on a

[6] In a large subnetwork it is quite possible that some elements or changing combination of elements will remain up for an almost indefinite length of time. The issue therefore arises of how wrapping of the epoch number might be handled. One alternative might be to define a circular ordering on the set of epochs. The chances of a disjoint subnet being reconnected when the epochs have drifted by exactly 2^{31} is sufficiently slim that the probability of inconsistencies occurring is practically insignificant.

Octet	Field	Description
0–3	MsgClass	Message Class — TOP_RECONFIG (1)
4–7	rcfgType	Reconfiguration type — RCFG_MY_INFO (4)
8–11	epoch	Reconfiguration epoch
12–15	rootMSNL	MSNL Address of root
16–19	blkNum	Sequence number of message
20–23	blkSize	Size of following data (octets)
24–27	size	Total size of all my info (in all blocks)
28–31	lines	Total number of 'lines' in my info
32–35	lnHostId0	Primary MSNL address of line 0
36–39	lnFlags0	Flags for line 0
40–43	nCons0	Number of connexions in line 0
44–47	hostId0,0	Remote host interface address
48–51	Flags0,0	Link flags
52–55	hostId0,1	Remote host interface address
...		
	hostId0,(nCons0 - 1)	Remote host interface address
	Flags0,(nCons0 - 1)	Link flags
	lnHostId1	...

Table 8: Reconfiguration partial-data message; octets beyond the double line constitue the data part of a message, octets above the double line are repeated for each message.

machine's capability, but should not be longer than 1920 octets in total.[7] The lnFlags$x$ and Flagsy,z fields are place-holders; their contents is not yet defined. If multiple partial-data messages are required then they are sent in the order of descending blkNum, with block zero last. Correct receipt of block zero should be acknowledged with the acknowledgement message. Otherwise a Nack may be generated if the total number of bytes received in all blocks is not equal to the total size of the data. When the data is split into multiple blocks it need not be split at any particular boundary. TD's in the tree with dependent subtrees wait until they have received partial data from each such subtree and concatenate the messages together; each TD adds one line to the partial-data message and forwards the resultant data up the tree towards the root; there is no requirement to sort the contents of partial-data messages. When the root of the tree has received responses from each of its subtrees it formats lines, and entries within lines in ascending MSNL address order and floods the result back through the tree with messages identical to partial-data messages except that the rcfgType field is set to RCFG_GLOBAL_DATA (5). Once a suitable acknowledgement is received from each subtree the discovery process is complete for that TD.

1.5 Use of skeptics

Skeptics may be used to determine whether or not a TD should respond to a topology beacon with a connexion request. They must not be used to determine whether or not to accept such a connexion request. This arrangement permits TDs that have been down for a long time to force themselves back into the subnetwork, but if both ends of a faultly

[7] The largest IObuf in Wanda is 2048 bytes, of which 128 are reserved for kernel use.

link choose to ignore occasional beacons then no harm is done. Determining unreliable nodes within the subnetwork is best left to the routeing dæmon, which can take a global view of the subnetwork and any alternative routes that might exist. No skeptics have been employed at present.

1.6 Timeouts

Timeouts may be employed between any message/ack or message/nack pair. In such circumstances one retransmission may occur after a timeout. If a second timeout occurs then the connexion should be closed and a reconfiguration in the next epoch initiated. Timeouts cannot be employed between other message combinations since there is a potential non-neighbour interdependence. Failures of this type should be detected with the stay-alive message timeout and recovered from in the normal manner.

2 Topology Discovery and the New MSNL Connexion Management

This section is purely a random sequence of ideas on how the topology discovery service might be integrated with the new MSNL connexion management system currently under development; none of these ideas has been implemented.

The new system is fully described in the section on Connexion Management. A network consists of a series of management domains interconnected by various types physical media; the network is subdivided into subnetworks according to the type of media and other administrative criteria; a management domain consists of one or more processing entities that have their communications controlled by a single set of management programs. Routeing of connexions is managed by routeing dæmons; there may be multiple dæmons for each subnetwork, but each subnetwork should have at least one.

2.1 Distribution of the topology information

It is the responsibility of the topology dæmon to keep the routeing dæmon and any other interested parties[8] aware of the current topology of the network. Each management domain should have a topology dæmon. Since the list of processes requiring such information is, potentially, unbounded, the best approach to this problem would seem to be that the topology dæmon should provide a registration service; each process would then register its requirements when first started and would be added to the distribution list. Different classes of registration might be possible, since the routeing dæmon clearly requires more information than the connexion manager.

2.2 Non-broadcast networks

The behaviour of the topology dæmon in point-to-point type networks (Fairisle, for example) would be broadly the same as the current implementation. More information, however, would probably be required than is currently collected. For example, information about link speed, and error rate would probably be useful to the routeing dæmon, particularly since the fpc3 will be able to reduce the effective link speed; the error rate might be useful in avoiding links prone to error, except when other alternatives are overloaded; it would also be of use to network maintenance programs. Because management domains may have several "internal" addresses these will also need to be included in the topology information. A complete description, therefore, of a node's capabilities might consist of the Primary address, whether the management domain contains a router, a table of point-to-point links described by the addresses of the local and remote interfaces

[8] The connexion manager in a management domain without a router needs to know the route to a router.

and link status information, a table of addresses and subnet masks for interfaces that are connected to other subnetworks, and a table of addresses for internal interfaces or external interfaces that are deemed to be unconnected.[9] It would also be desirable to be able to list alternative addresses for multi-homed leaf domains (i.e. domains that have multiple interface addresses, or are connected to more than one network, but are not running any gatewaying software).

The forced routeing required by the protocol might be more easily managed by the assignment of permanent virtual circuits; the stay-alive messages would then be replaced by "I can/can't hear you" messages.

2.3 Broadcast networks

Broadcast networks can be assumed, for the purposes required here, to include both bus based networks (such as Ethernet), and ring based networks, since in both a single message can be transmitted to all hosts without duplication being required by the network. All entities connected to the network are effectively at a distance of a single hop from each other. It is distinctly undesirable, and not necessary, to have all the TDs in such networks connected to each other, since the only purpose would be for detecting the reachability of the domain, and resolving its MAC level address (ARPing); the traditional approach of ARPing for a remote machine's address is a far less wasteful approach.[10]

However, it is desirable that management domains with routers should know about each other and about domains with gateways, and that management domains without routers should know where to find a router. The anticipated number of such domains, even on a large broadcast network, is expected to be fairly small; so it would seem desirable that domains of this class should allow their TDs to connect to each other; therefore the trivial response message should not be used since these would be likely to cause "message storms". The conventional approach of ARPing for an address consists of broadcasting to every host on the subnetwork an ARP request, to which only the desired machine replies; this can have a significant impact on the performance of the network when there are many machines ARPing at regular intervals. A possible way to reduce the loading caused by this is for TDs connected to management domains with a router to act as an ARP server; a machine requiring a MAC level address contacts the ARP server, which probably has the address cached already, but if it hasn't it can ARP in the traditional manner and cache the address for future use; the MAC level address of the ARP server is known from its beacons. Proxy ARPing (ARPing for an address on a subnet different to the one a domain is connected to) should not be needed, since the connexion manager in the domain will connect to the router, indeed there are reasons (see below) why it might be desirable for intra-subnet connexions to be directed via the router as well.

2.4 Bridging a subnet

Because of the structure and management techniques used within a domain it should be possible to bridge two or more segments of broadcast subnetwork providing all are connected to different interfaces of a single domain. For example a Fairisle switch, managed as a single domain with a gateway, could be used (in theory) to bridge up to 16 Ethernet segments, which would still act as a single subnet; or two CFR's could be bridged together

[9] A point-to-point connexion is unconnected if there are no stay-alive messages, a broadcast network connexion is unconnected if no router can be seen for that network

[10] The current MSNL code does not explicitly ARP for an address; the ARP is an implicit part of the connexion request, which is broadcast to all machines.

via the CBN provided the two interface nodes on the CBN were managed as a single domain with a gateway. Such bridged networks would still need to support ARPing and other types of broadcast message, so a spanning tree, like that of Autonet would need to be created by any gateways involved. Each segment of the subnetwork would probably require a router in order to work efficiently; and using an ARP server would help to reduce the broadcast load in such a configuration.

References

[1] M. D. Schroeder, A. D. Birrel, M. Burrows, H. Murray, R. M. Needham, T. L. Rodeheffer, E. H. Satterthwaite, and C. P. Thacker. Autonet: a high-speed self-configuring local area network using point-to-point links. Technical report, Systems Research Center, 130 Lytton Avenue, Palo Alto, Califorina 94301, April 1990. Techreport 59.