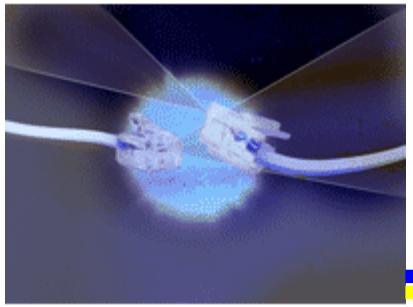
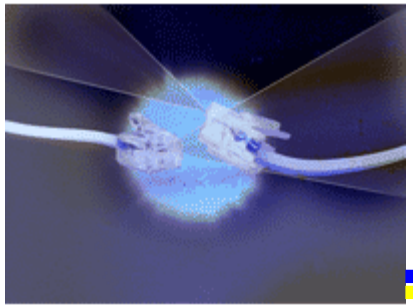


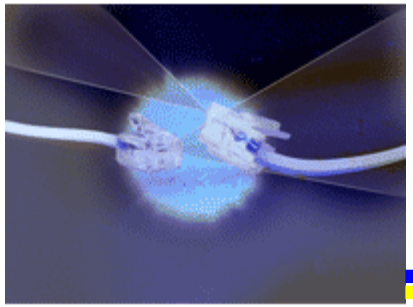
Link-State Routing

- Same assumptions as distance vector routing
  - Each node can determine its neighbors
  - Each node can assign a cost to the link
- Basic algorithm: if every node knows how to reach its neighbors, and this information is distributed to entire network, every node can figure out shortest paths



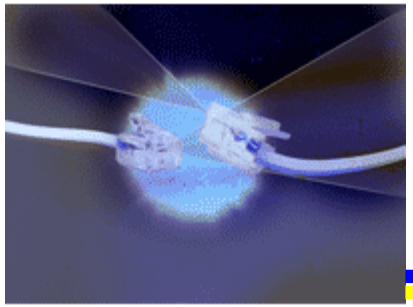
Link-State Routing (cont.)

- Requires two mechanism
  - Reliably distribute link-state information
  - Compute routes from this information
- First mechanism uses *reliable flooding*
  - Each node sends information out on all links
  - Each node that receives forwards information on all other links



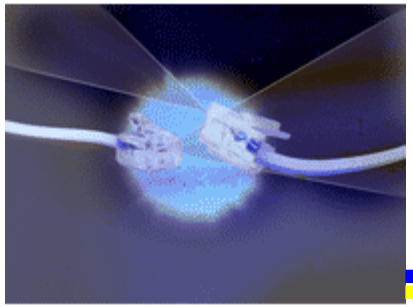
Link State Packets

- LSPs contain the following:
  - ID of node that created LSP
  - List of directly connected nodes, with costs
  - Sequence number
  - TTL
- Last two fields added to improve reliability of flooding



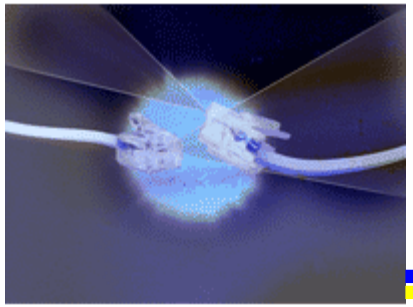
Reliability

- Exchange of LSP with neighbors uses reliable protocol
- When node receives LSP
  - Stores if it doesn't already have on from that source
  - If it already has the LSP and new seq # greater than stored value, updates entry and forwards to all other neighbors



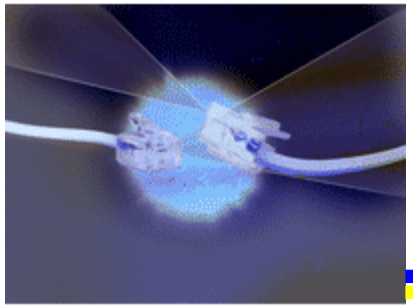
Generating LSPs

- Done in two circumstances:
  - Periodically
  - When topology changes
    - Same techniques to detect as used in distance vector routing
- To reduce LSP traffic, use long timers for periodic updates; since LSPs are reliably distributed, don't need to be updated very often



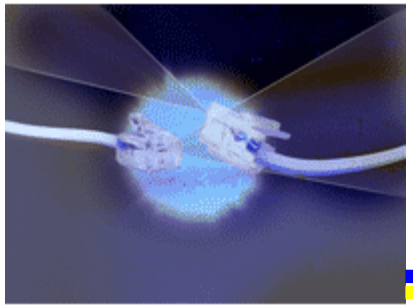
Generating LSPs (cont.)

- Every time LSP is updated, seq. # incremented
  - Starts at 0 when node reboots
  - Node may receive its own updated LSP from another node – just starts using that seq #
- LSPs discarded when TTL expires
  - TTL decremented before node floods to neighbors, ages when stored in node. When TTL is 0, node discards LSP



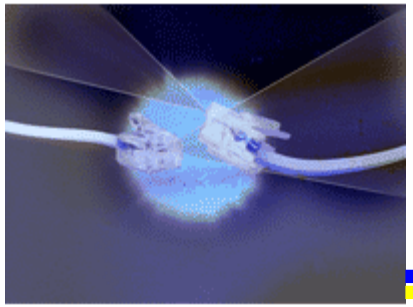
Route Calculation

- Entire network topology known from LSPs
- Node uses Dijkstra's shortest-path algorithm to compute routes
  - Constructs graph of network from LSPs
  - Algorithm uses *adjacency matrix* to represent graph: *N* is set of nodes, *l(i,j)* is cost of edge from *i* to *j*, *i*, *j* in *N*, cost = ∞ if no connection
  - Also keeps *M*, set of nodes incorporated so far, and *C(n)*, cost of path from start node *s* to node *n*



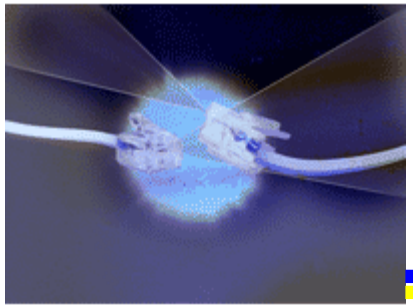
Dijkstra's Algorithm

- $M = [s]$
  for each n in $N - s$
    $C(n) = l(s, n)$
  while ($N \neq M$)
    $M = M \cup \{w\}$, where $C(w)$ minimum
      for all w in $(N - M)$
    for each n in $(N - M)$
      $C(n) = \min(C(n), C(w) + l(w, n))$



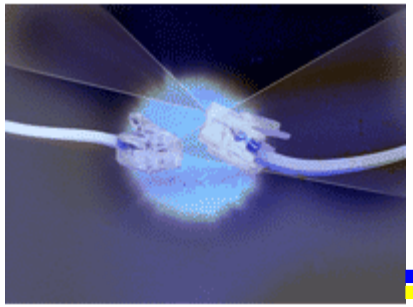
Forward Search Algorithm

- Modified version of Dijkstra's algorithm that builds routes directly from LSPs – *forward search* algorithm
  - Doesn't require creation of graph from LSPs
  - Maintains two lists, *Tentative* and *Confirmed*. Each list has entries (Dest, Cost, NextHop)



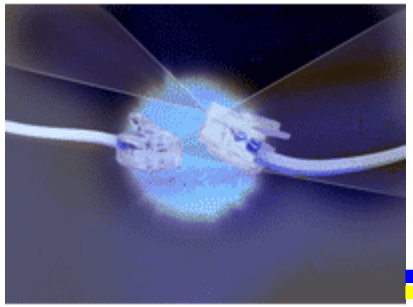
Forward Search (cont.)

1. Initialize *Confirmed* with entry for current node (S, 0, -)
2. For node just added, *Next*, select its LSP
3. For each neighbor *N* of *Next*, *cost*(*s*,*N*) = *cost*(*s*,*Next)* + *cost*(*Next,N*)
   1. If *N* not in *Confirmed* or *Tentative*, add (*N*, *cost*, *NextHop*) to *Tentative*
   2. If *N* in *Tentative* and *cost*<current cost, replace with new entry
4. If *Tentative* empty, stop. Otherwise, pick *Tentative* entry with least cost, move to *Confirmed*, select as *Next*, and go to 2



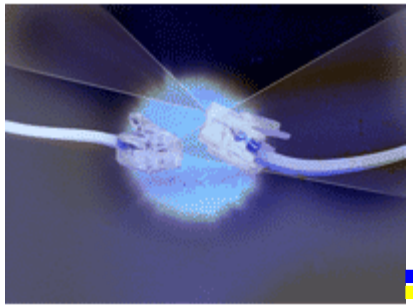
Link-State Tradeoffs

- Advantages:
  - Does not generate much routing table traffic
  - Responds quickly to topology changes
- Disadvantages
  - Requires a lot of storage in each router
- Link-state vs. distance vector:
  - Link-state talks to everyone, only tells them what it knows definitely
  - Distance vector talks only to neighbors, tells them everything it thinks it has learned



Open Shortest Path First Protocol (OSPF)

- Most common link-state protocol
- Adds the following to basic link-state:
  - Authenticate routing messages
  - Additional hierarchy – partition routing domain into *areas*. For other nodes outside a router's area, router only needs to know how to get to area
  - Load balancing –multiple routes can have same cost and both be used by router



OSPF Messages

- Five types: “Hello” to let neighbors know router is alive, plus messages to request, send, and acknowledge data
- Based on Link-State Advertisement messages (LSAs)
  - One to send cost of links between routers
  - One to send networks to which router is directly connected
  - One to send area information