3/13/2017
DTN -- Project Proposals Due Weds

Store & forward / data muling / delay tolerant networks

Topic: routing in a "challenged internet"

<u>What is that? What are some examples of challenged internets we have seen?</u>

Sensor networks like TinyDB, but also interplanetary networks, intermittently connected "data muling" networks, etc.

<u>Simple scenario -- suppose route is known / unchanging, but frequency of connectivity may be low / unpredictable.</u>

Why can't we just use an existing transport protocol, e.g., TCP or UDP?

No -- because these generally assume some predictable end-to-end latency.

Better choice might be a "store and forward design", i.e., mail routing, early email / usenet

<u>How does mail routing work?</u>

(FedEx -- all airmail Memphis, then on to region airports, then by truck...)

<u>How about email?</u>

(DNS to look up mail server for a given domain)

Used to be "bang paths"

MIT!ATT!Berkeley!BerkeleyEECS!joe

MIT!!{ATT,ComCast}!Berkeley...

Even in a network with fixed nodes, there are some questions:

<u>Who is responsible for delivering message?</u>
- origin
- "custody transfer"

<u>When to re-transmit/retry?</u> This boils down to how to estimate end-to-end (E2E) latency. (What's the problem with retrying too frequently? We will end up resending packets we've already sent!)

Hop-by-hop acks vs e2e acks?

In any case, don't delete until received. How are e2e acks delivered?

How long do I need to keep around the fact that i've already ack'd a packet?

Ok, so where does this paper fit in?

Note that this paper doesn't actually tell us how to route packets in a network like this but presents a general set of principles that are useful in this setting. It's really a vision / architecture paper, not a paper about a specific system.

Envisioned Architecture:

Overlay network -- a "network of networks" -- show diagram

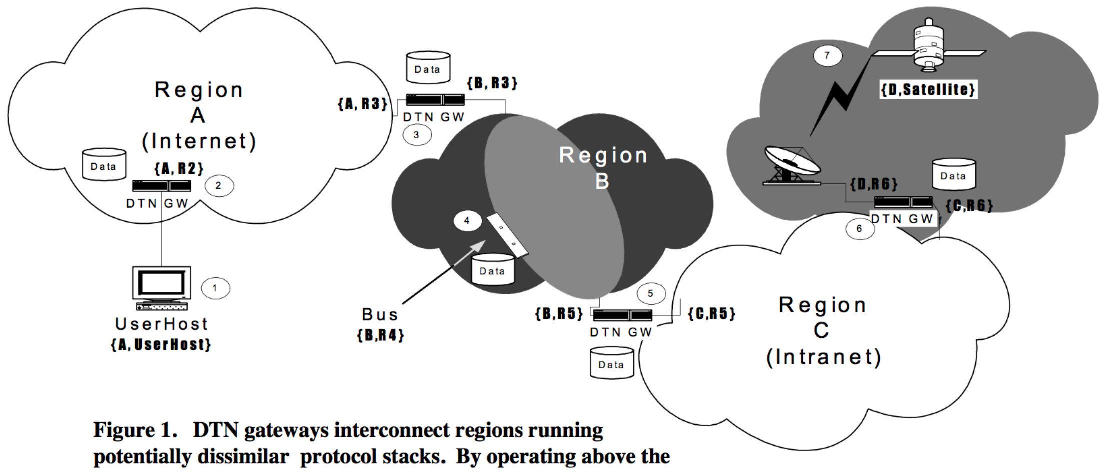


Figure 1. DTN gateways interconnect regions running potentially dissimilar protocol stacks. By operating above the transport protocols in use on the incident networks, they provide virtual message switching, in-network retransmission, and name mapping, allowing globally-interoperable names to be mapped

Networks are called regions

"DTN gateways" are responsible for receiving data from one or more regions and forwarding it to one or more other regions. Basically a proxy.

Author rejects proxy design in S3. Why? (Show text)

"The disadvantage of proxies are in their specificity. Proxies usually use one of two approaches: they respond to a specialized set of commands specific to the special network, or act as raw data conduits. The first approach limits the ability to re-use the proxies for different applications; the second method fails to take advantage of any special resources the proxy node may have to offer (such as storage or processing capabilities), and requires applications communicating with the proxy to employ specialized protocols that are compatible with those of the special networks'. Finally, no general inter- proxy routing capability is currently used, meaning that if any dynamic routing is performed between proxies it is specific to the types of proxies in question. It would be generally more attractive to standardize on a set of proxy-based services which provide I/O to and through the challenged network, if possible, using a common set of methods. "

Essentially he is advocating for proxies but saying we should have a generalized architecture for routing across these challenged networks, rather than specialized proxy protocols developed by each network designer.

What is the basic interface he envisions for DTNs?

Asynchronous messaging, much like email. Groups of messages called "bundles" are forwarded hop by hop through the network.

This is sometimes called a "store and forward" architecture -- messages move from one node to the next, and are buffered and forwarded as connections come and go from one network to the next.

How are messages addressed in a DTN?

Some hierarchical naming system for regions, akin to DNS. Within regions, abstract *names*, not addresses. Names are a way of referring to a destination that may not identify a single machine / process, but rather a service which may be transient, change over time, or be distributed.

Why not just use an IP address?

Because not everything in the universe has an IP address, and IP addresses are generally fixed to a single machine / endpoint.

What delivery features does he envision a DTN should provide?

Motivated by postal "classes of service". E.g., different priority levels, "delivery receipt" (ack), reliable (guaranteed) delivery.

How are bundles routed in a DTN?

Idea is that packets will be delivered by internal networking protocol to gateway nodes on edges of each DTN. Between DTN gateways, some kind of routing protocol is needed.

He basically punts on this, saying:

"The particular details of path selection and message scheduling are expected to be heavily influenced by region-specific routing protocols and algorithms. "

<u>How do you ensure reliable delivery in a DTN?</u>

Challenge -- E2E transfer times may be days or weeks, often longer than lifetime of process sending the message, or in extreme cases **even longer than the lifetime of thephysical node sending the message!**

<u>How does TCP ensure reliability?</u>

"Exactly once" delivery -- retransmissions to ensure message delivered at least once and a list of previously delivered messages to ensure at most once.

<u>Same idea can be applied in a DTN, but when to retransmit?</u> He doesn't really tell us, because the problem is hard if link times are unpredictable (punt's again!)

"In particular, some rough expectation of the round-trip time is extremely useful to trigger attempted repair actions. "

Fact that he doesn't say how to solve the two most interesting problems -- routing & retransmission -- is understandable in a generic architecture sense but also frustrating because as an implementer these are the exact things I would want a network protocol to handle. I.e., the reason I use TCP/IP instead of rolling my own network protocol is because a) it interoperates with everything and b) it handles retransmission and routing!

He does introduce on important idea here -- "<u>custody transfer</u>". What is that?

Because an endpoint can be transient and fail. it may want to ask some other, more persistent node, to take over reliable delivery of a bundle for it. For example, a sensornet node could transfer custody of some readings to a gateway, or a martian rover could transfer custody to a nearby satellite. Once a custody transfer is completed, the new custodian will do its best to ensure reliable delivery, presumably with a higher likelihood of success than the original endpoint because it has more regular connectivity, a longer expected lifetime, or both.

So how might we solve the routing problem.   <u>How does internet routing work?</u>

BGP: basically each network advertises addresses it can route to, and nodes advertise address they can reach, and path / distance to each address it knows about. Nodes learn about paths by listening to neighbor advertisements.

Generally can apply this idea in any network; could extend this with frequency / duration of connectivity of each link or other information, as with neighbor tables in TinyDB>

Can also do this on demand -- which may make sense if links are very unpredictable but last for while. E.g., suppose I want to discover a route from some node A to some node F:

"AODV" -- ad-hoc, on-demand, distance vector

See: http://www.antd.nist.gov/wctg/aodv_kernel/aodv_guide.pdf

Basically flood a request asking for path to a given destination

<u>Now suppose nodes can move or links are very transient?</u> Hard because distance vector returned route may be totally invalid by the time we discover it!

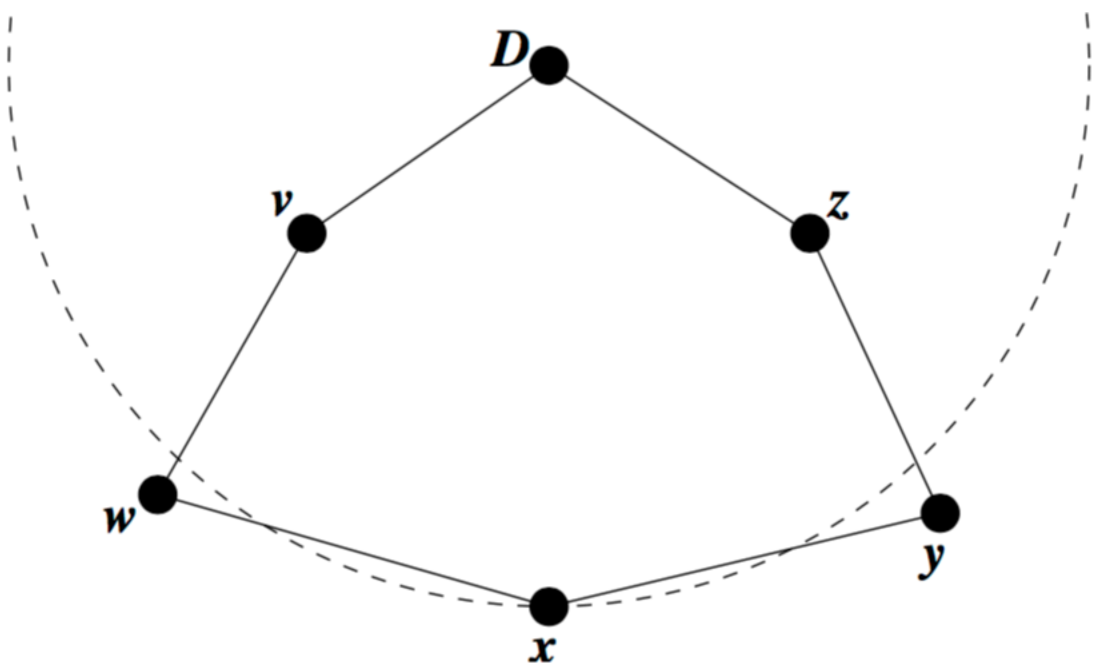
Some solutions:

1. Flooding
    Problems: flood storms
                        state on nodes (have to keep track of a lot of messages)
    When to retry/retransmit, and whom?
    When can you forget about a message?

2. Geo-routing

If nodes know their location, and the approximate location of their neighbors, maybe we can use geographic "routes'

Basic idea: look at all your neighbors, send message to one that is closer to the destination than you are; Problem: "holes". --Soln: GPSR



(source: Karp & Kung: GPSR: Greedy Perimeter Stateless Routing for Wireless Networks, Mobicom 2000)
Basic solution: "right hand rule" -- send to first neighbor counter-clockwise to yourself, relative to the path from yourself to the destination, along with information about your distance to destination.

If someone on path is closer to you, they can exit this "perimeter mode". Otherwise they forward according to the same rule.

If packet comes back to a node, there is no path.

This is provably true in "planar" graphs.
Planar = A graph with no edges that intersect each other -- e.g., a graph embedded in a Euclidian coordinate system has this property.

That's all: How are projects? Project partners