# HTTP-*DTN*

delivery across ad-hoc networks

**draft-wood-dtnrg-http-dtn-delivery-01**
Lloyd Wood and Peter Holliday
Global Government Solutions Group,
Cisco Systems.

**Lloyd Wood**

IRTF Delay-Tolerant Networking session
IETF 71, Philadelphia, March 2008.

# HTTP-*DTN*

- MIME describes the things we move around the network. The most successful protocols support MIME.
- HTTP is the simplest MIME wrapper.
- HTTP provides infinitely-flexible text metadata.
- Use HTTP hop-by-hop between neighbouring DTN nodes.
- No proxying, no intercepting. Proxy cache model is not relevant here.
- Allow HTTP to be run over different transports: TCP, SCTP, *Saratoga*… HTTP can be separated from TCP's limitations.
- Divide HTTP from transport to make a true session layer. What HTTP requires from transport isn't that onerous.

# What makes HTTP*-DTN* special?

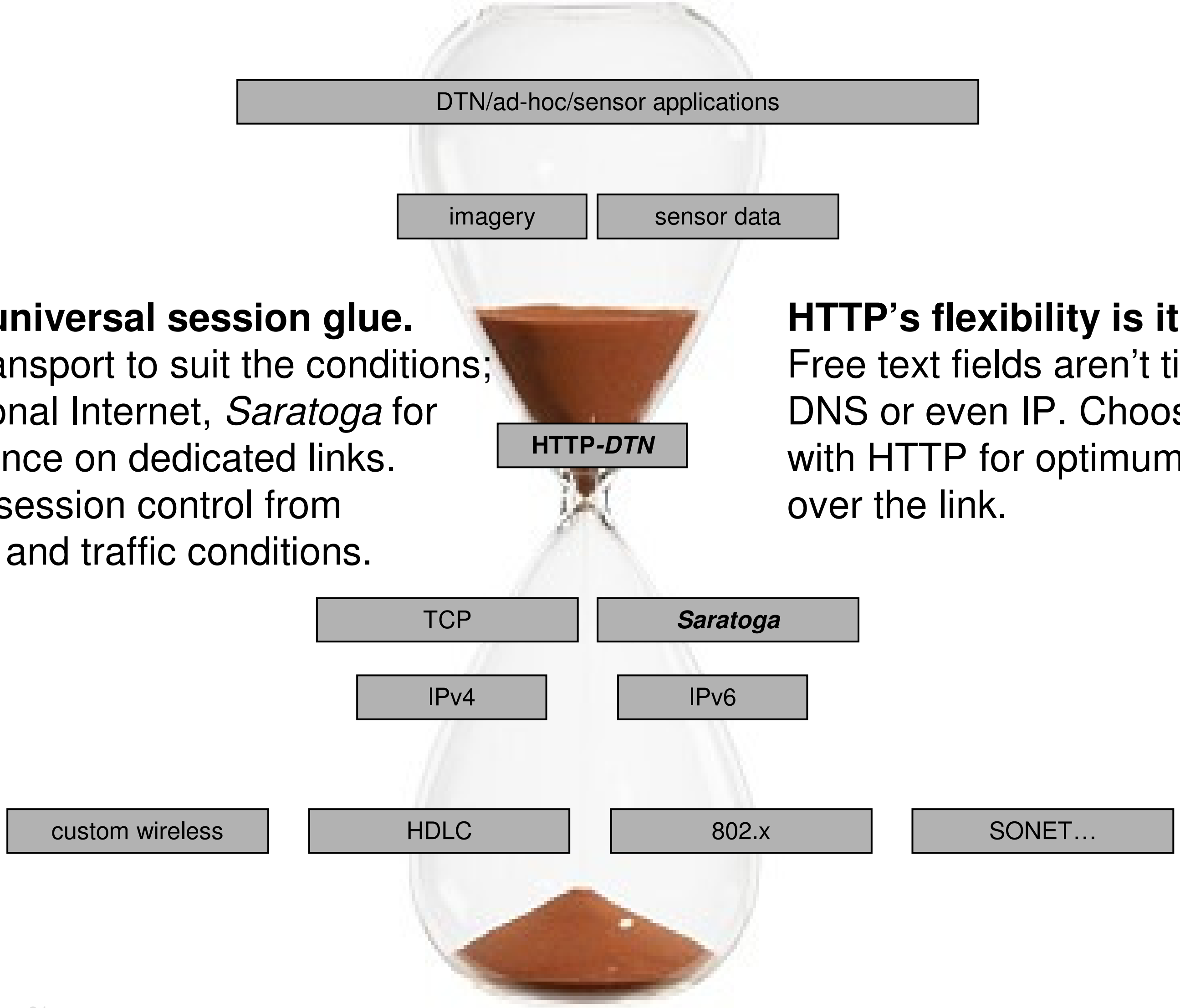
- Two new **Content-*** headers:

  **Content-Source:** where the object is originally from

  **Content-Destination:** final destination
- Basic HTTP rule: **Content-*** headers are special . If Content-blah is unfamiliar, reject the transfer.
- This makes HTTP*-DTN* separate from, and not polluting, existing web. Unlikely to alarm W3C.
- Optional e2e reliability over payloads by reusing existing **Content-MD5:** header or similar.
- Header/metadata reliability a bit trickier – may need new headers. HTTP already supports ‘per hop’ limited-scope headers.
- New Package- headers can *package* related objects together, track if they’ve all arrived or not.

# HTTP-DTN is the waist in *this* hourglass

**HTTP is the universal session glue.**
choose the transport to suit the conditions; TCP in traditional Internet, *Saratoga* for high performance on dedicated links. Separate the session control from transport, link and traffic conditions.



**HTTP's flexibility is its strength**
Free text fields aren't tied to TCP, DNS or even IP. Choose what to use with HTTP for optimum performance over the link.

# HTTP-*DTN* advantages

- Text fields aren’t tied to IP, TCP or to DNS. Could implement HTTP over own stack, with own routing namespace, etc.
- Doesn’t require a two-way session; HTTP PUT can be entirely unidirectional.
- Reuses large body of existing code and well-understood functionality. Only minor changes.
- Possible to build on top of HTTP-*DTN* base to reuse pieces of web infrastructure, e.g. SOAP.
- Conceptually very *very* simple.

# Issues

- Security

  Could use https: for hop-by-hop security.

  Could use S/MIME for end-to-end security – or applications could implement their own. Unsure. Early days yet.

- Timestamps

  pretty much the same timing/sync issues as the Bundle Protocol has come across.

- Header overhead

  may be significant for small transfers; it's the cost of flexibility. (Bit efficiency was *gopher*'s strong point.)

# What model do we use with HTTP DTN?

- We don't have to even use IP, but…
- **We still believe IP is useful for operational use of delay/disruption tolerant networks** – IP is not *just* convenient/cheap for prototyping DTN code.
- Make each transport layer work with HTTP and IP. The transport between HTTP and IP must support HTTP's simple session semantics.
- Pick the transport to match the local environment.
- How do we build these transfers into a bigger architecture that can make forwarding and routing decisions? Open – there's a lot of pieces of IP-based infrastructure that *may* be reusable, depending on the exact scenario.
- Early days, interesting adaptation questions to address.

## draft-wood-dtnrg-http-dtn-delivery-01

Questions?
thankyou