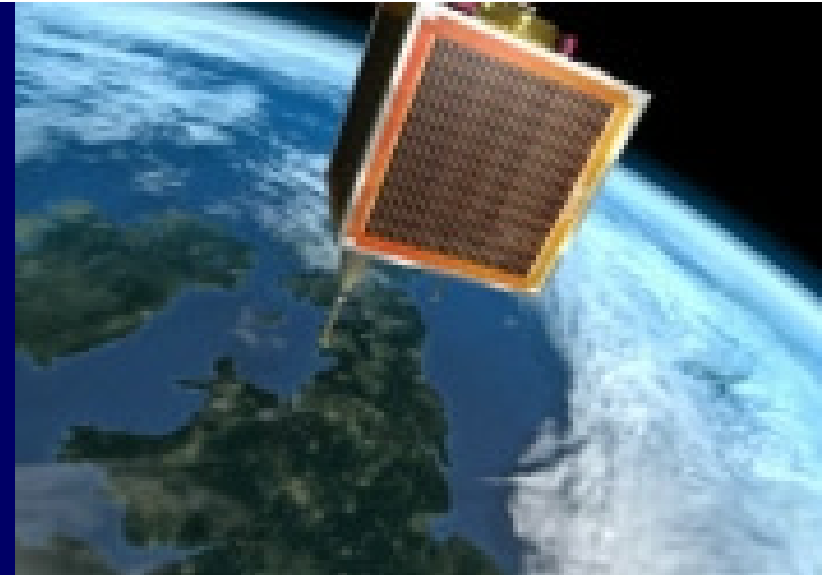


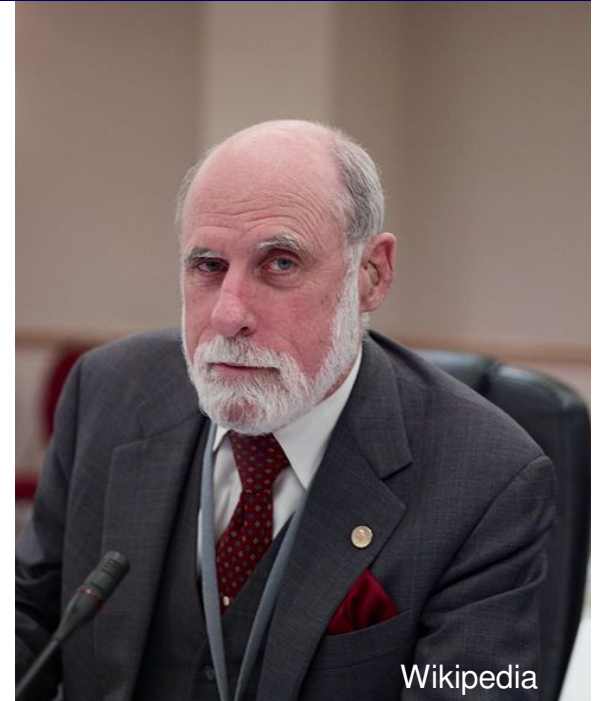
# A Bundle of Problems



**Lloyd Wood**  
IEEE Aerospace conference  
Big Sky, Montana.  
March 2009.

## How did it all begin?

- Vint Cerf announces start of effort over ten years ago, in July 1998.
- Collaborates with Adrian Hooke of **NASA Jet Propulsion Lab (JPL)** – who leads CCSDS (Consultative Committee for Space Data Systems), an ISO subgroup that sets *standards for space*.
- Space probes predate computing; tape recorder bitstream mindset. Want to move them towards packets and networking.
- Long propagation delays difficult; can't work with protocol timers.



Wikipedia



Associated Press

## Vint sets up an Internet Society SIG...

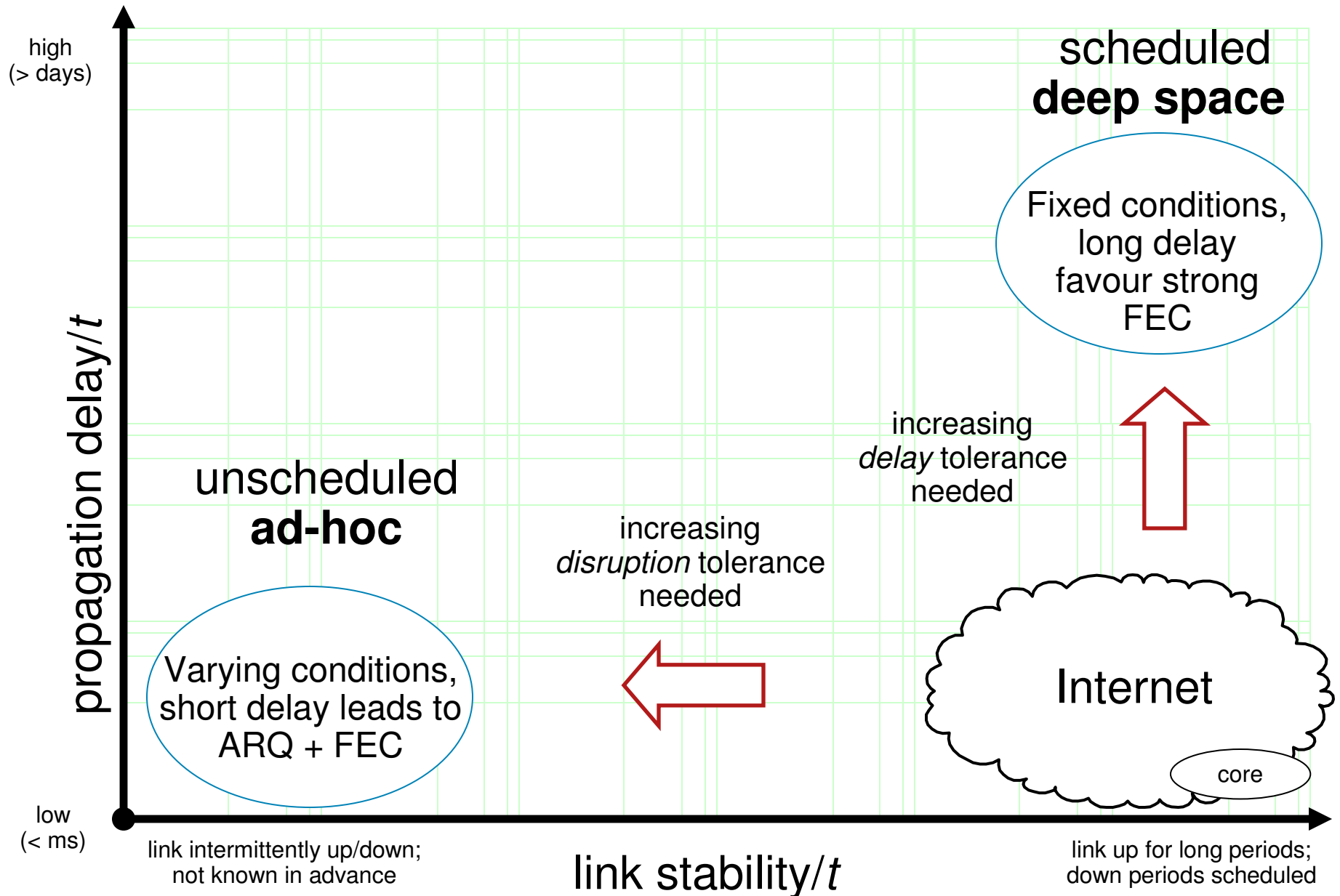
- IPN Special Interest Group (IPNSIG).
- Then a short-lived IRTF 'Interplanetary Internet' group (IPNRG) and a couple of internet-drafts. 2000/2001.
- Problem scope widens to 'Delay Tolerant Networking' (Kevin Fall) and bundles are created, 2002/2003.
- IRTF DTN research group set up. (Kevin introduces DTNRG at IETF 56, March 2003.)
- DARPA *Disruption-Tolerant Networking* proposers' day, January 2004. (Lots of funding.)



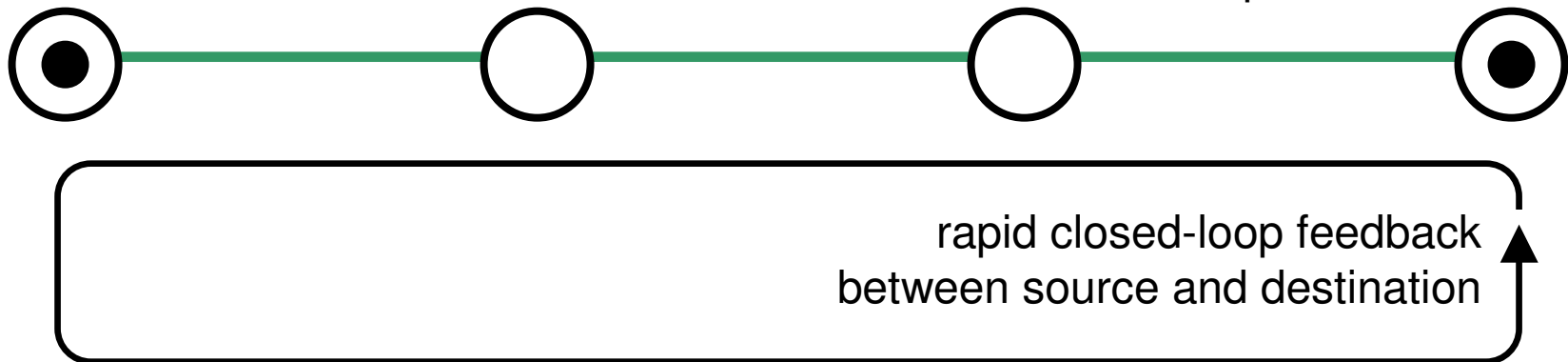
# Problem scope was consistently widened

- First, let's solve *interplanetary* networking for the long delays of deep space.
- Then, let's solve *delay-tolerant* networking for intermittently-connected ad-hoc networks.
- Then, let's solve *disrupted* ad-hoc military networks under battlefield conditions.
- Increased the interest/attention/funding.
- R&D efforts and costs are now spread over many groups and budgets outside NASA ... but will results still solve the original problem?

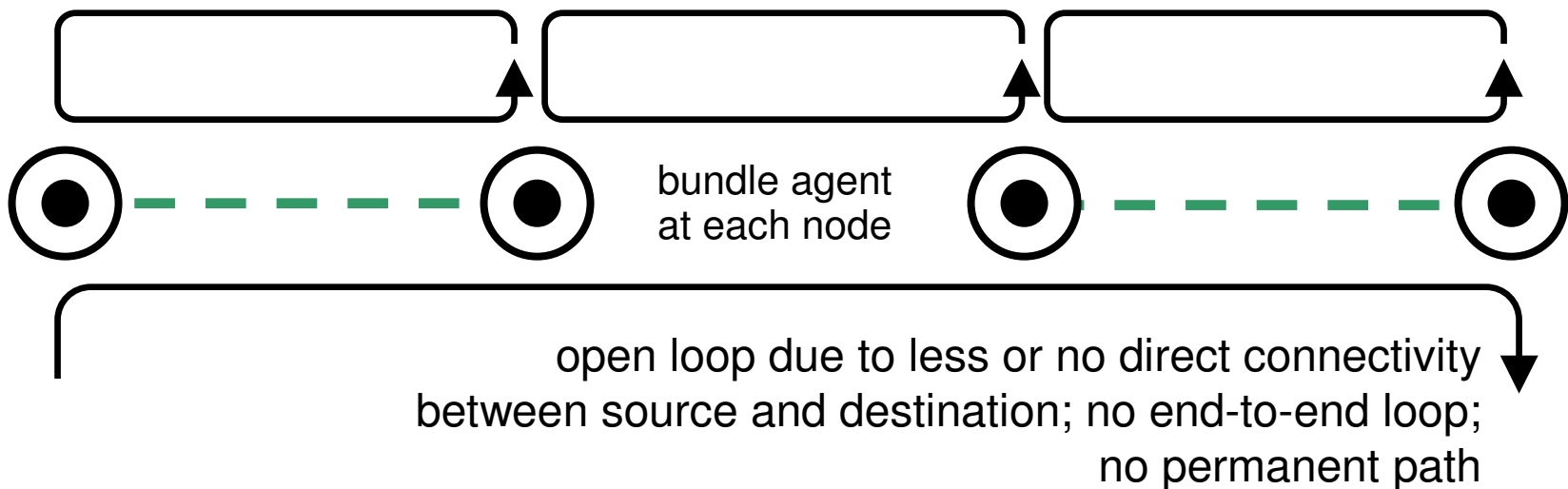
# Two different problem spaces



**Terrestrial fixed Internet** little need for resends between or checking at nodes when resends can easily and quickly be done end-to-end over the whole path instead



**Delay-tolerant network** more reliance on separate closed loops between each pair of nodes with local checking for *e.g. custody transfer* and to increase throughput



# What is the Bundle Protocol?

- Basically layer over different *internets*, just as the Internet Protocol layered over different *networks*.
- late binding of Bundle *endpoint identifiers* to a local network address.

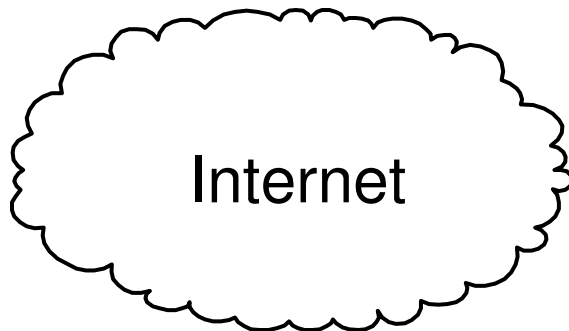
## Bundle Protocol

convergence  
layer adapter  
suited to local  
conditions

TCP

Licklider (LTP)

custom



something  
else

# Basic Bundle structure – blocks.

Primary Bundle Block

version	flags
Block length	
Offsets into Dictionary identifying source, destination, custodians etc.	
Timestamps and lifetime	
Dictionary information listing Endpoint Identifiers (EIDs)	
Any fragmentation and length info	



First Payload Block

type	flags	length
Any references to Dictionary EIDs		
payload		



*n*th Payload Block

type	flags	length
Any references to Dictionary EIDs		
payload		

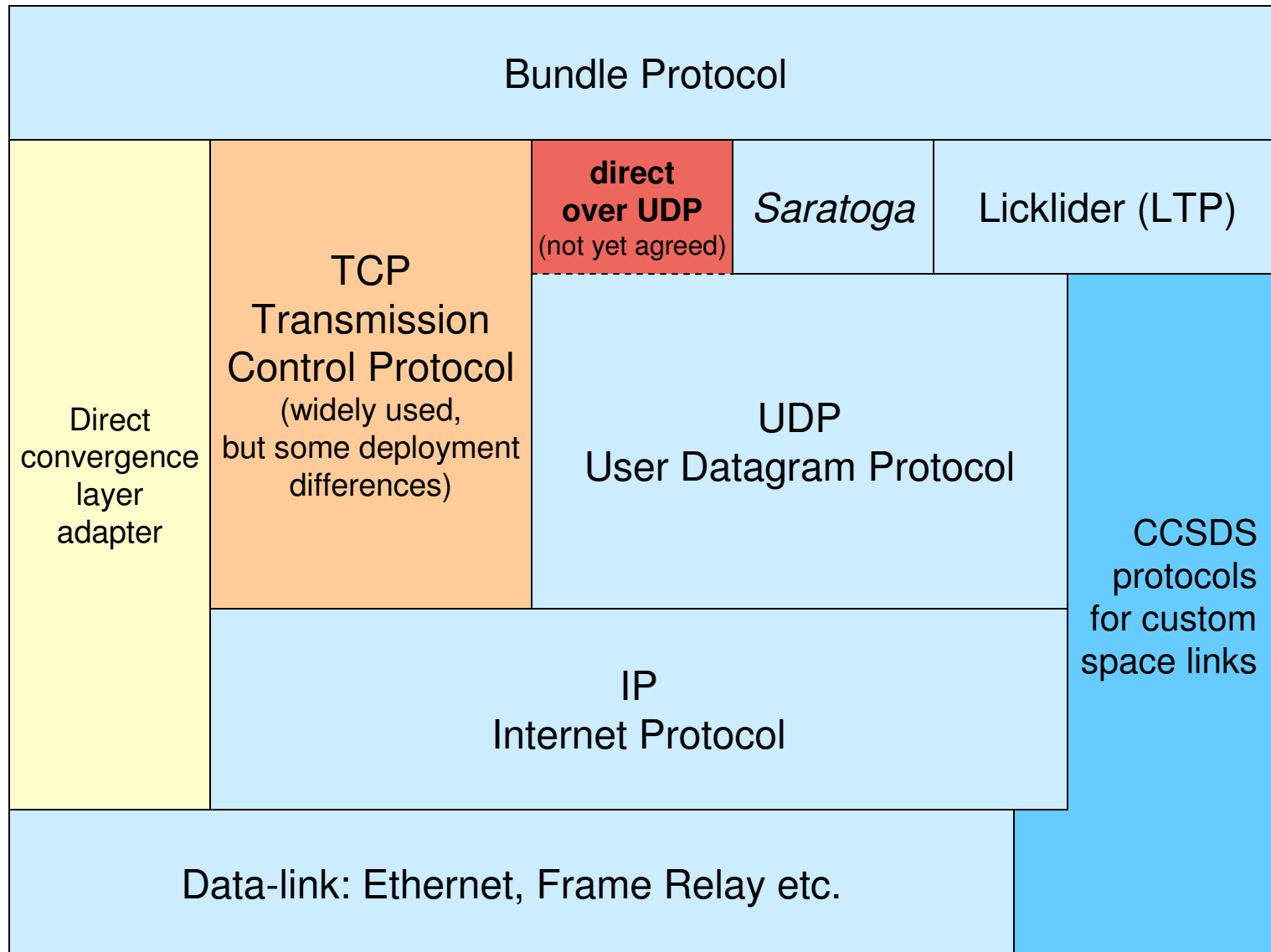
Most fields use SDNVs (Self-Delimiting Numeric Values, like ASN.1) and are not fixed-length.  
No checksums.



## Bundle Protocol really a container format.

- Multiple blocks, following a primary block with a dictionary. Blocks can be encrypted.
- *Mutable canonicalisation* – idea that block ciphers can cover and protect some different metadata (header) primary fields, similar to IP pseudo-header. Other fields are unprotected.
- Custody transfer allows handing over responsibility of delivery.
- **But no end-to-end reliability.** Custody transfer doesn't check bundle has been copied correctly!
- Variable-length SDNVs are like ASN.1 – last bit indicates continuation. If that bit gets corrupted...

# Existing convergence layers for the Bundle Protocol



Most Bundle Protocol use is over IP. Except for the CCSDS world, of course.

# Our approach to DTN networking

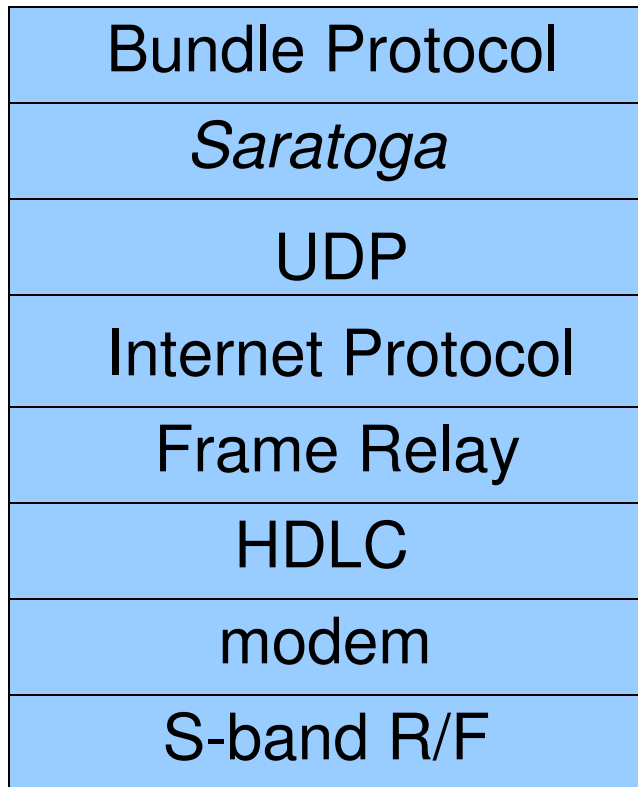
- **We believe that the Internet Protocol (IP) is useful for operational use in delay or disruption-tolerant networks.** Being convenient and cheap are compelling reasons to use IP for DTN. IP runs over many links already. Implementing support for custom “DTN bundle” convergence layers directly over all these links simply isn’t scalable or cost-effective. Many IP-based protocols can be reused for DTN.
- The Disaster Monitoring Constellation (DMC) uses IP both on the ground and in space, with the ground station acting as a gateway between different types of network links.
- How IP is used differs between ground and space (link use, shared contention *vs* dedicated scheduling models – this discourages TCP reuse) but the base IP protocol remains the same. DMC satellites provide a *real*/DTN scenario, with long disruptions between passes over ground stations.

## Bundle Protocol tests in space

- On Surrey Satellite Technology's UK-DMC satellite, in January and September 2008. Used Bundle Protocol over *Saratoga*.
  - downloaded real operational sensor data, transferred fragments across Internet from Surrey to NASA Glenn.
- On NASA JPL EPOXI (Extrasolar Planet Observation and *Deep Impact* Extended Investigation) comet probe, October 2008. Used Bundle Protocol over LTP over CFDP over lots of stuff. DINET – Deep Impact DTN Experiment
  - uploaded pictures to probe, got them back again.
  - Implemented ground network simulating other probes.

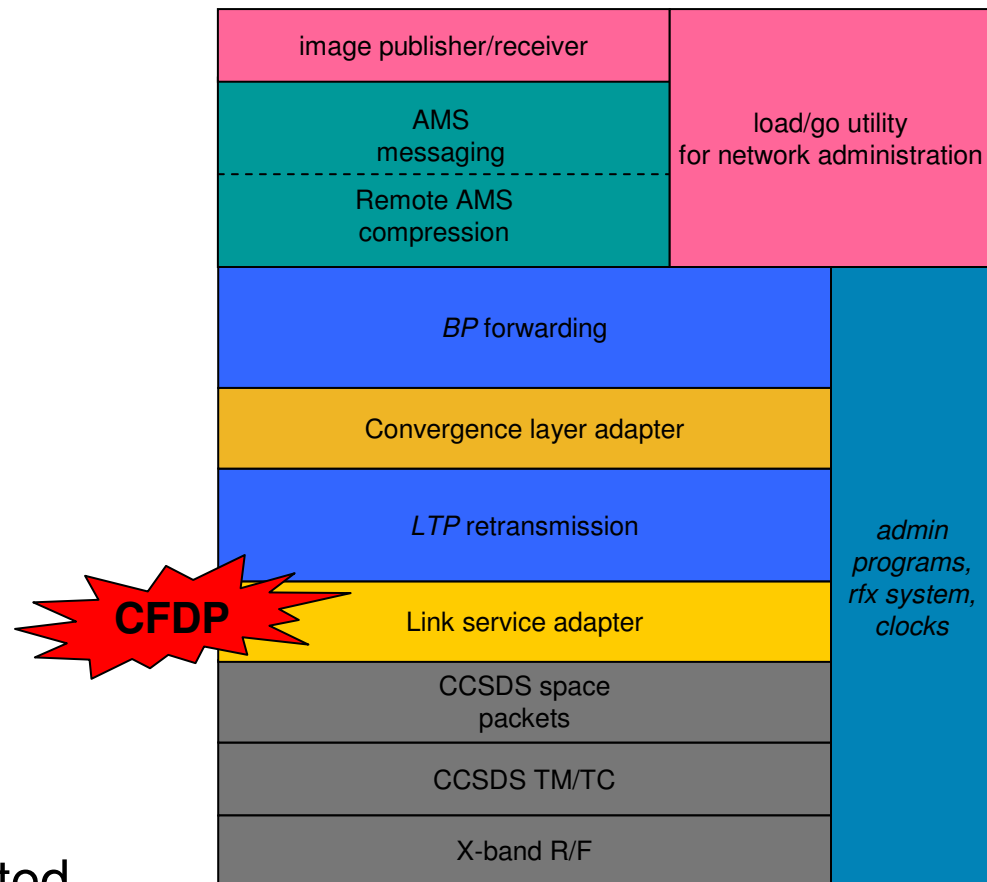
# Networking stacks used in these experiments

UK-DMC tests, Jan/Sep 2008  
after Hogie. Max possible bundle size: 4GB



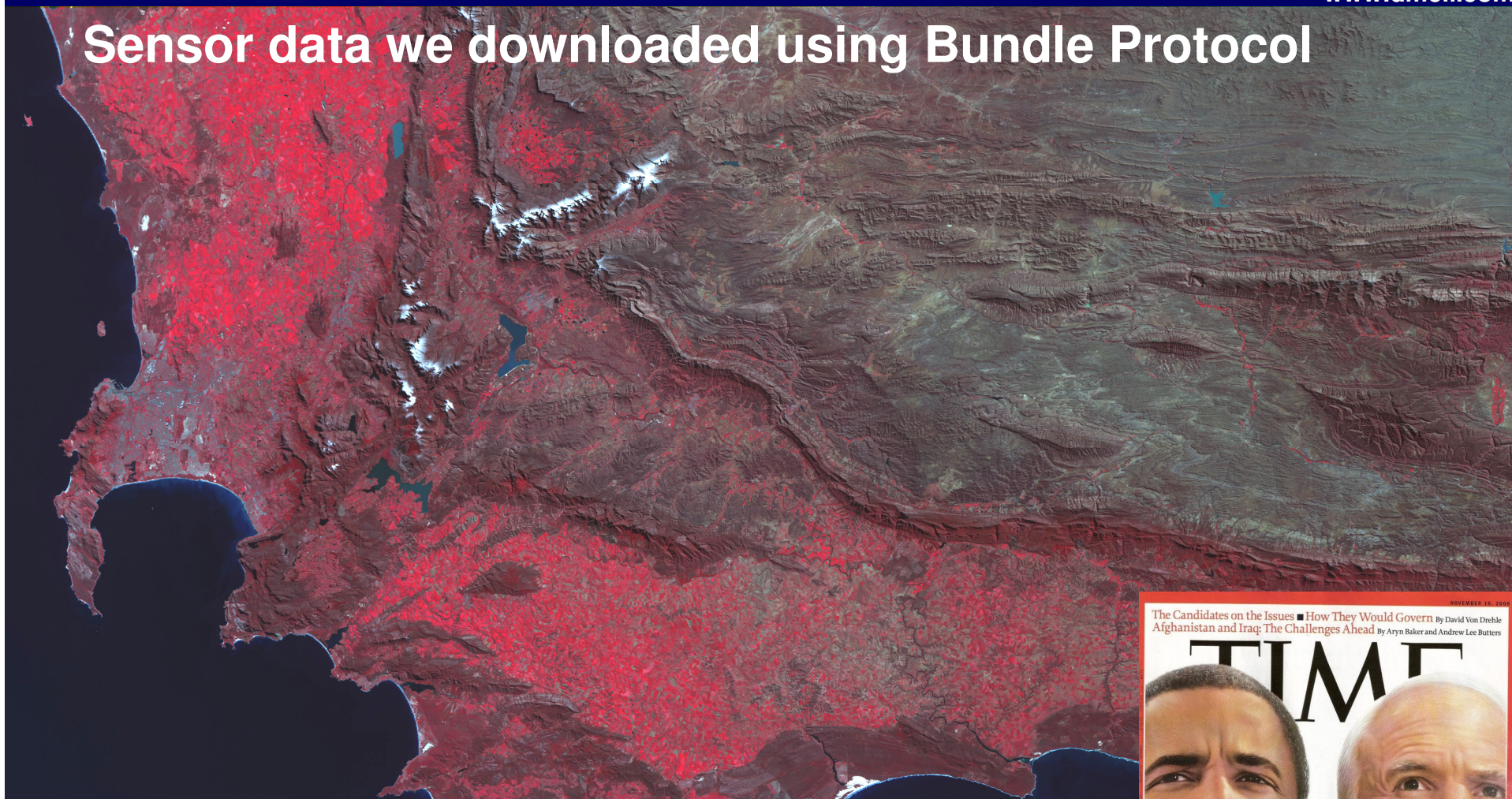
Bundle security not implemented onboard either spacecraft.

Deep Impact, Oct 2008  
after Burleigh. Max possible bundle: 64K



Scott Burleigh, IETF 73 DTNRG meet, 20 Nov 2008

# Sensor data we downloaded using Bundle Protocol



150MB image transferred from UK-DMC satellite using Bundle Protocol over *Saratoga* with proactive fragmentation, 25 August 2008.

*TIME Magazine* best inventions of the year **#9 Orbital Internet**, 10 November 2008 issue – before EPOXI tests announced.



# We have discovered problems with bundling

- **Reliability.** No error detection, and reusing security to give reliability is not ideal. Errors seen Jan 2008.
- **Timing.** Every bundle agent is expected to know current UTC time. This has limits in space (relativity, eventually). Leap seconds must be communicated. Synchronization is a problem; bundles can get dropped as expired (Jan 2008).
- **Convergence layer adapters.** Pretty much all use and deployment is over IP – *except* for CCSDS.
- naming schemes/routing/QoS/management.
- No content identification *a la* MIME and HTTP...

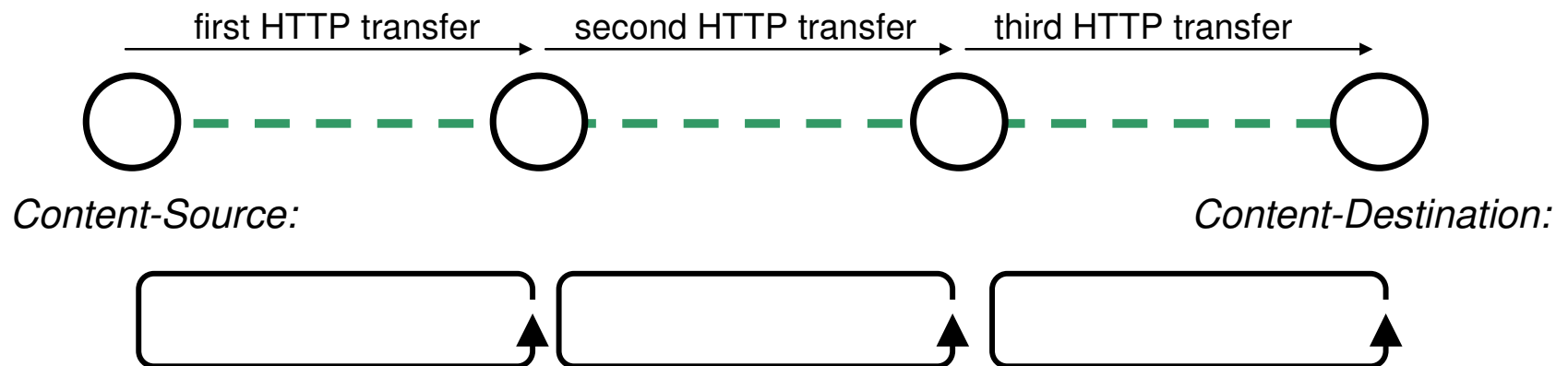
# The Bundle Protocol...

- ... ignores its environment; relies on the convergence layer for reliable transport and other services for things the bundle protocol cannot do itself.
- ... ignores the effects of control loops.
- ... ignores real-world errors – read **Jonathan Stone's papers** on the occurrence and unavoidability of errors in everything.
- ... is an example of **stone soup**; it has created a happy research community building support infrastructure, yet supplies very little itself.



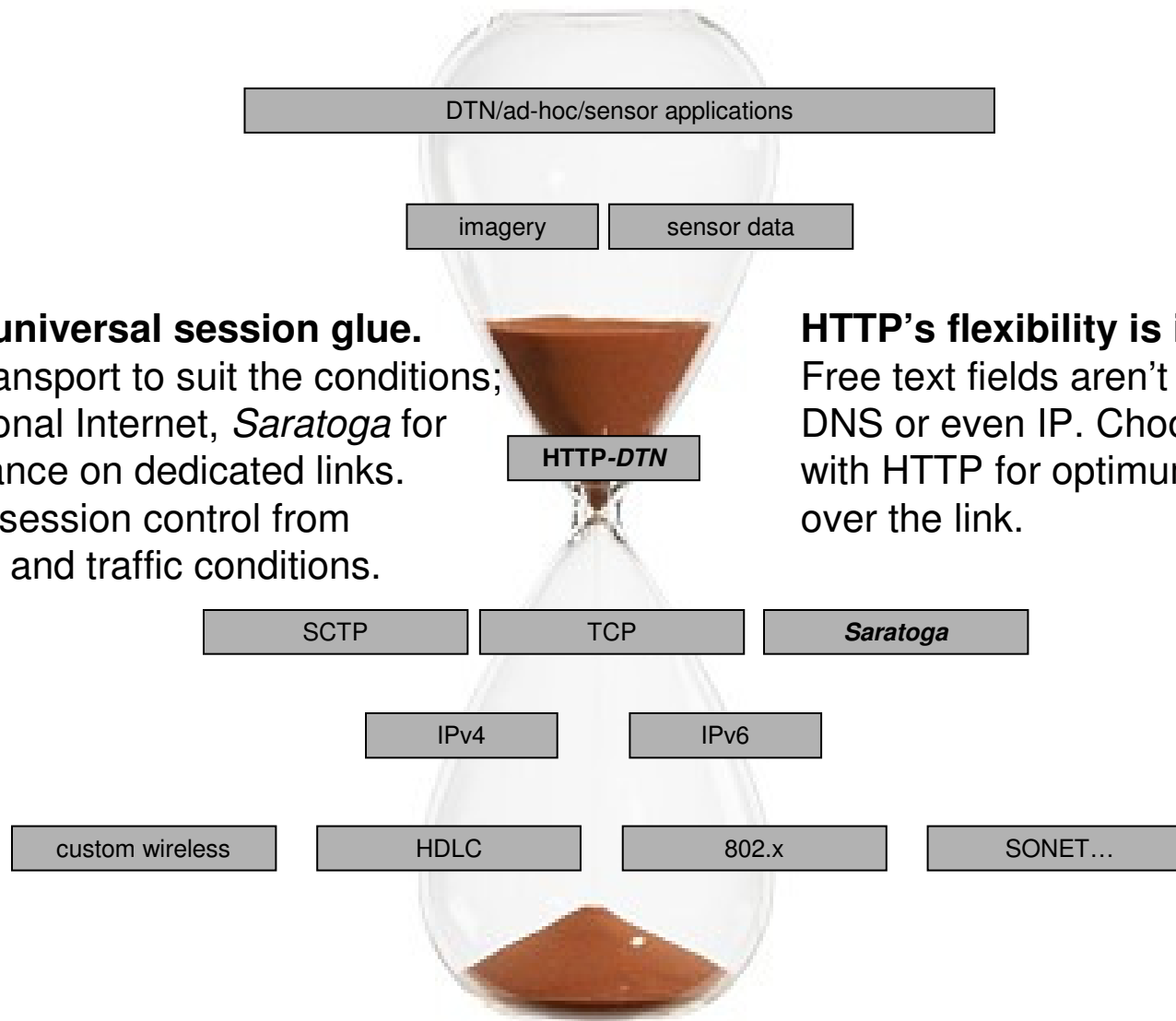
# An alternative to bundling: HTTP-*DTN*

- 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.



- 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.

# 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.
- Shares some of the bundle protocol's problems, e.g. shared timebase, but gets there with far less development work. *Very very* simple.

# Some thoughts

- Does the Bundle Protocol really meet the needs of its various problem spaces?
- Is such a complex and fragile bundle format suited to harsh errored ad-hoc conditions?
- Given the problems that we have identified, is this protocol really ready for real use? Or is more operational experience and development required? Or a different approach?
- **Consider the implications of the end-to-end principle and control loops.**



For more information  
**google UK-DMC bundle**

papers on Lloyd's Surrey webpages:  
**<http://info.ee.surrey.ac.uk/Personal/L.Wood/>**

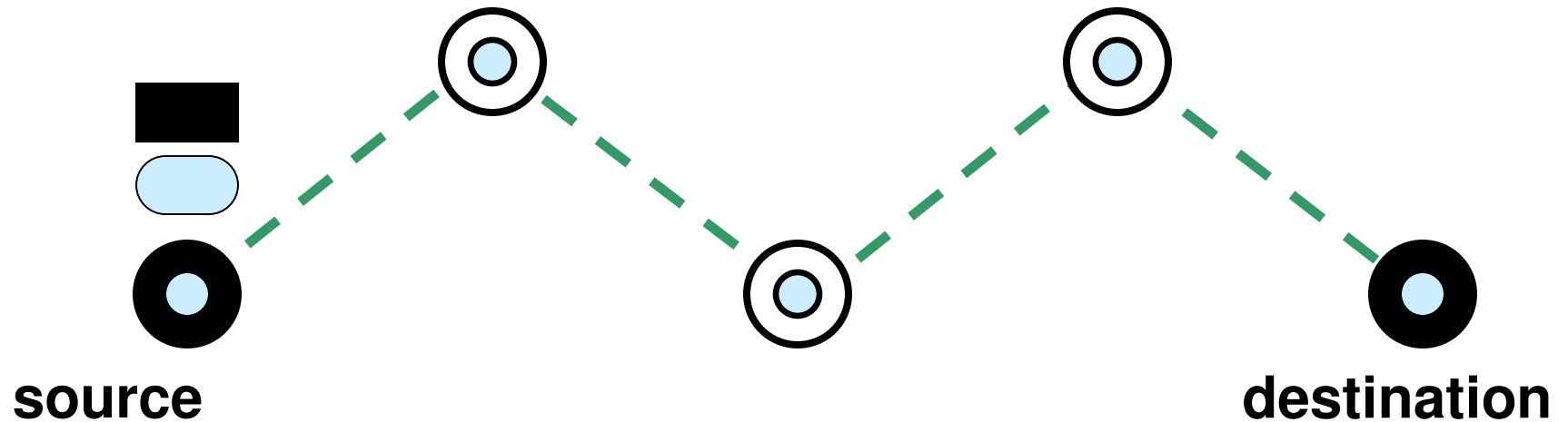


extra slides  
**added detail**

# Bundling compared to IPv6

- IPv6 packets don't get fragmented and reassembled in the network. **Bundles do.**
- IPv6 runs in a tight, closed, end-to-end control loop. **Bundles don't. Open loop between applications.**
- IPv6 can leave all its checking to the endhosts and applications, thanks to closed control loops and fast resends. **Bundling can't.**
- **DTN networks must take a different approach to reliability.**

# Control loops: security and custody transfer #1




**secure PIB**

payload integrity block,  
as in bundle security drafts

shared or private keys 

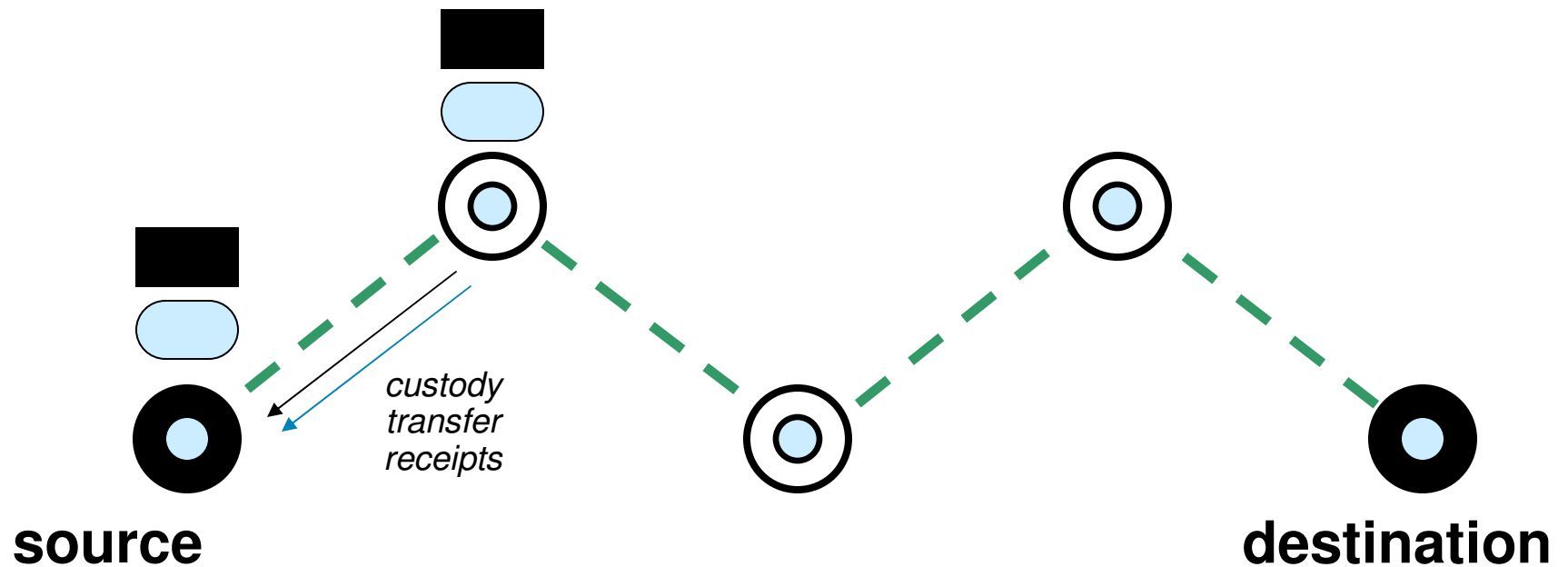
**insecure  
ciphersuite**

as in draft-irtf-dtnrg-bundle-checksum

shared keys only 





# Control loops: security and custody transfer #2



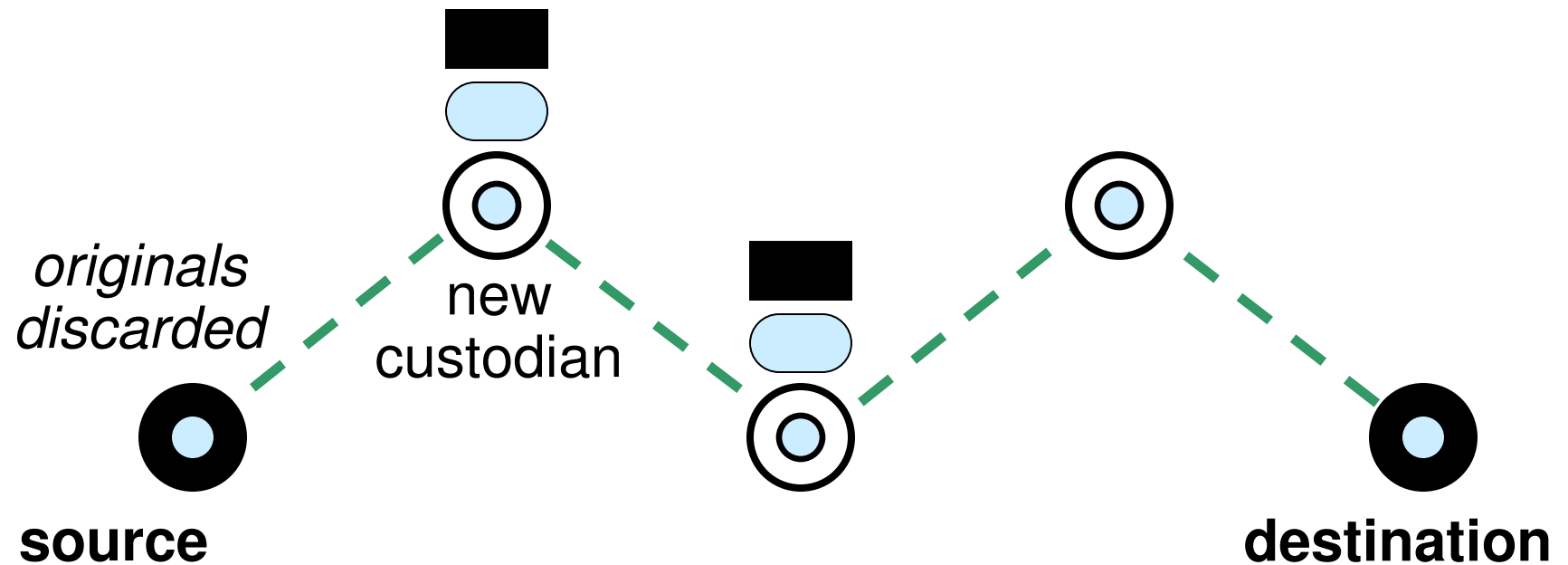
**secure PIB**

**insecure  
ciphersuite**

shared or private keys 


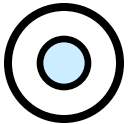
shared keys only 

# Control loops: security and custody transfer #3

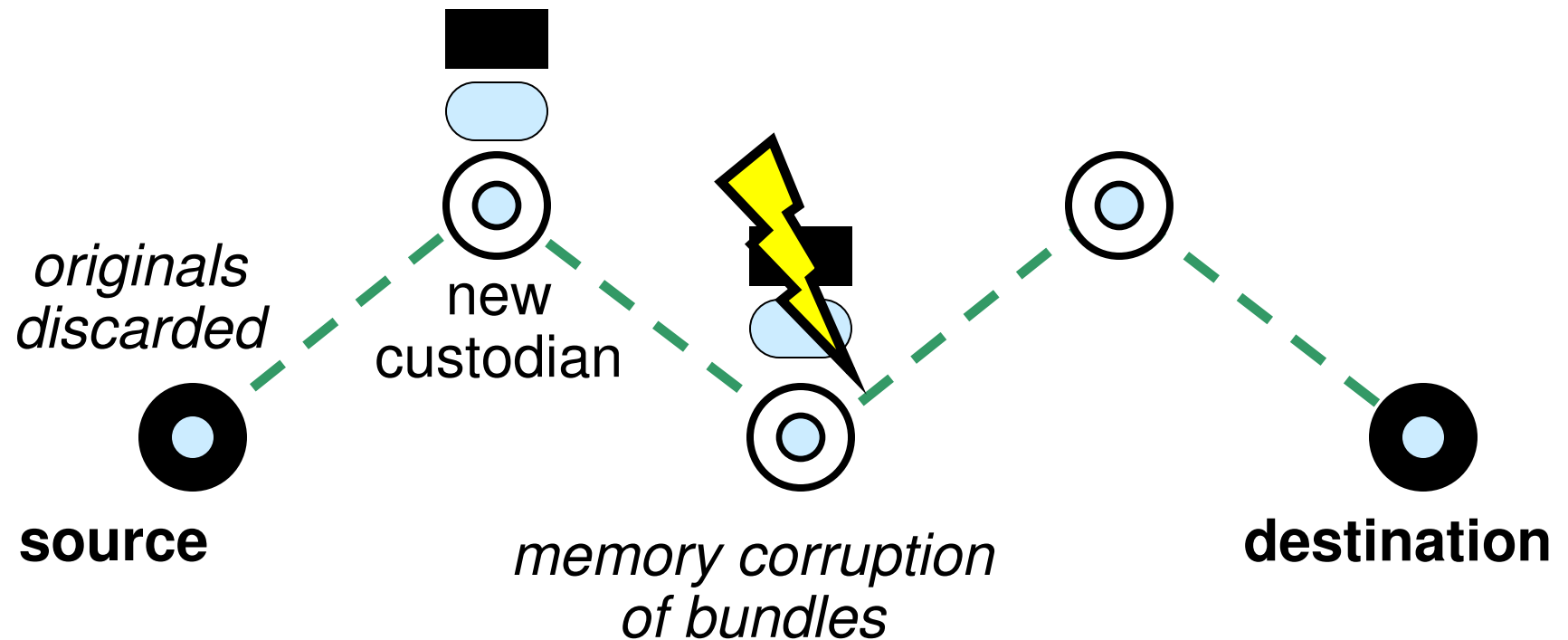


**secure PIB**

**insecure ciphersuite**


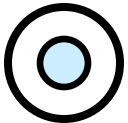
shared or private keys   
shared keys only 

# Control loops: security and custody transfer #4

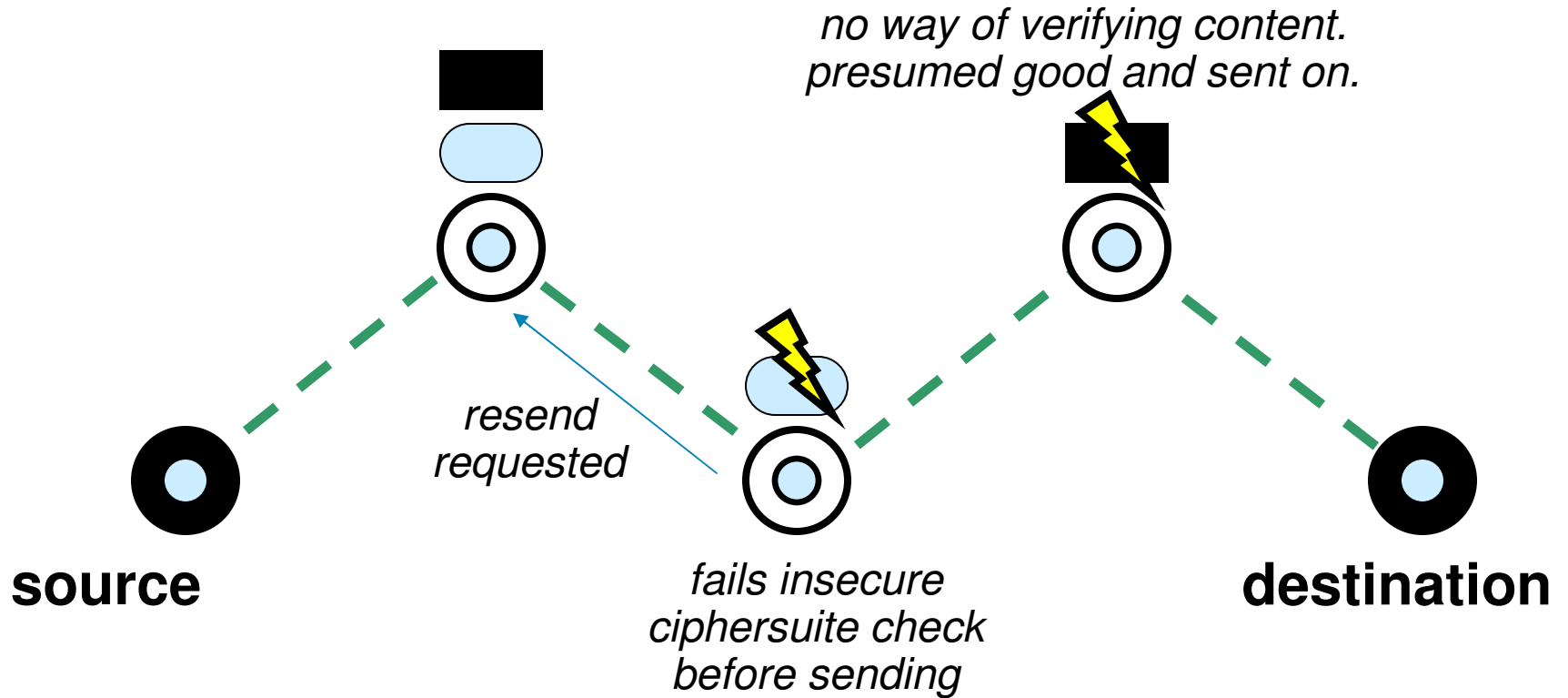


**secure PIB**

**insecure ciphersuite**



shared or private keys   
shared keys only 

# Control loops: security and custody transfer #5

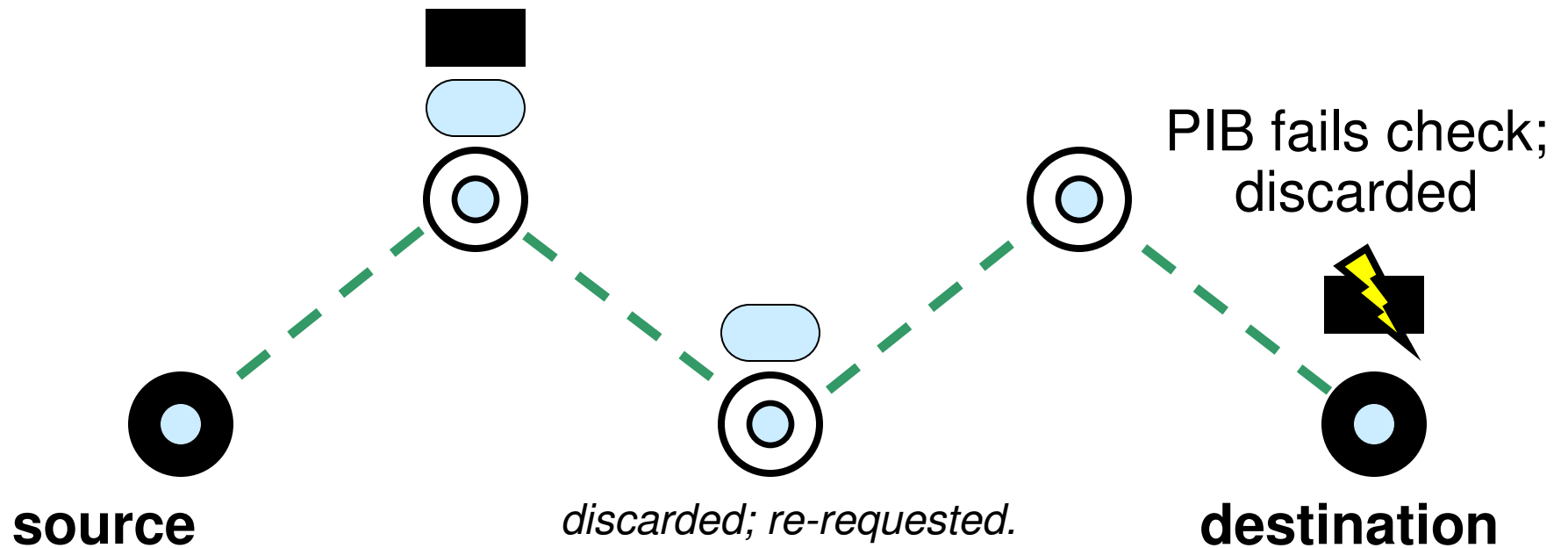


**secure PIB**

**insecure ciphersuite**


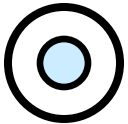
shared or private keys   
shared keys only 

# Control loops: security and custody transfer #6

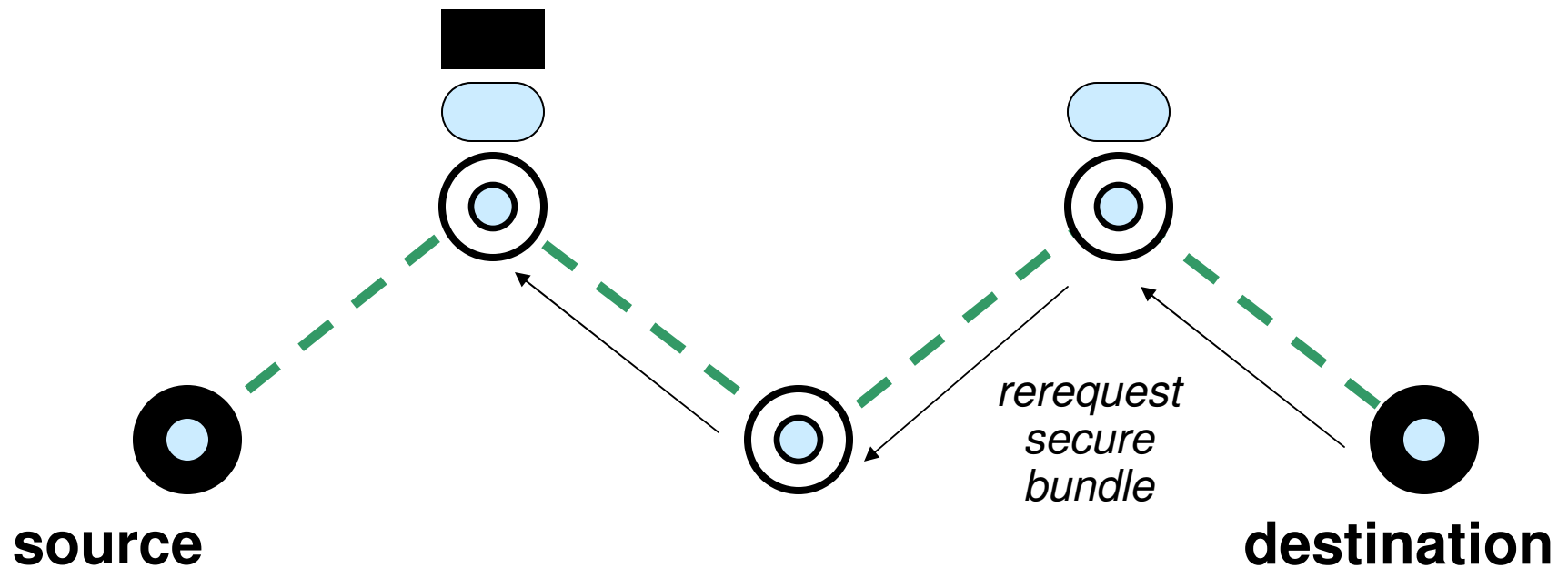


**secure PIB**

**insecure  
ciphersuite**



shared or private keys   
shared keys only 

# Control loops: security and custody transfer #7

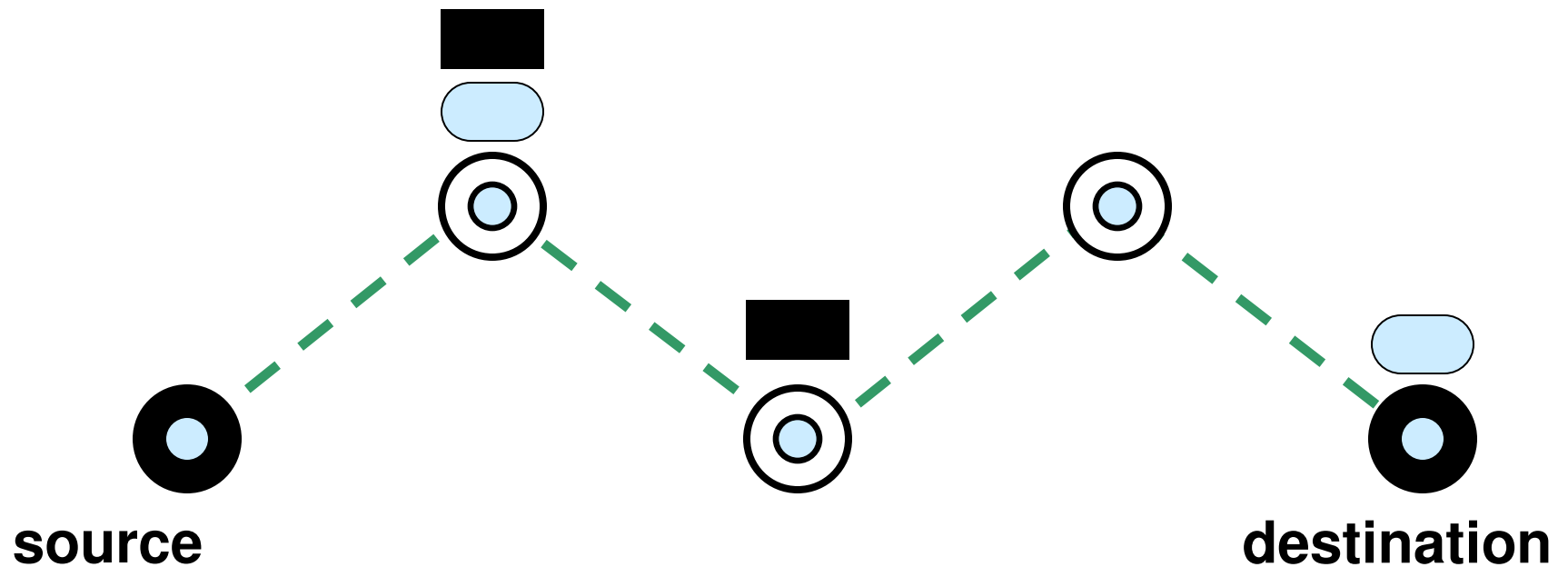


**secure PIB**

**insecure  
ciphersuite**

shared or private keys   
shared keys only 


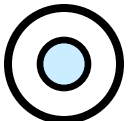
# Control loops: security and custody transfer #8



*Insecure bundle that can be checked in-transit has arrived faster.*

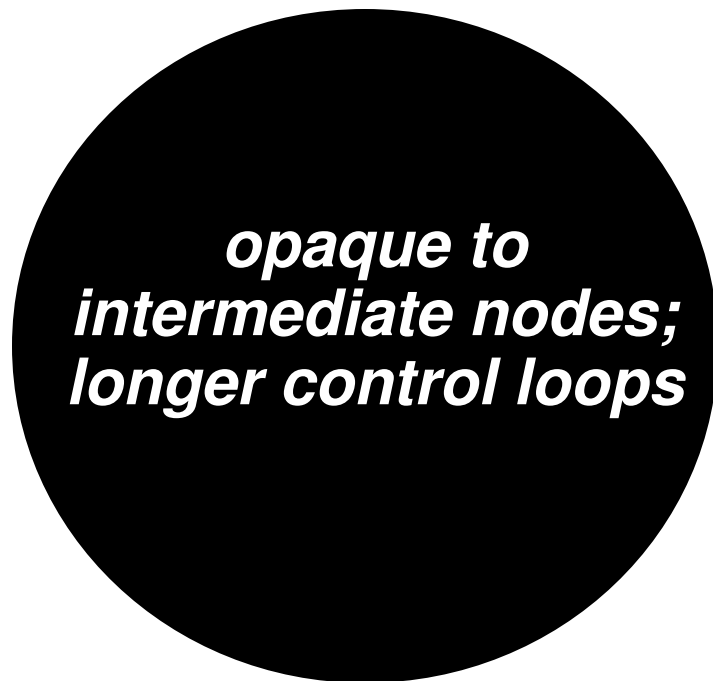
**secure PIB**

**insecure ciphersuite**

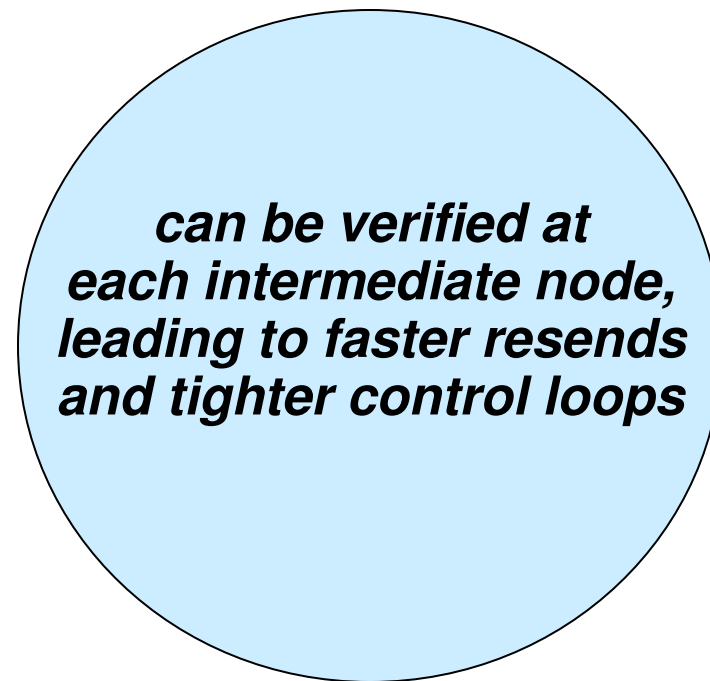
shared or private keys   
shared keys only 

# Tradeoffs

**PIB secure bundle**



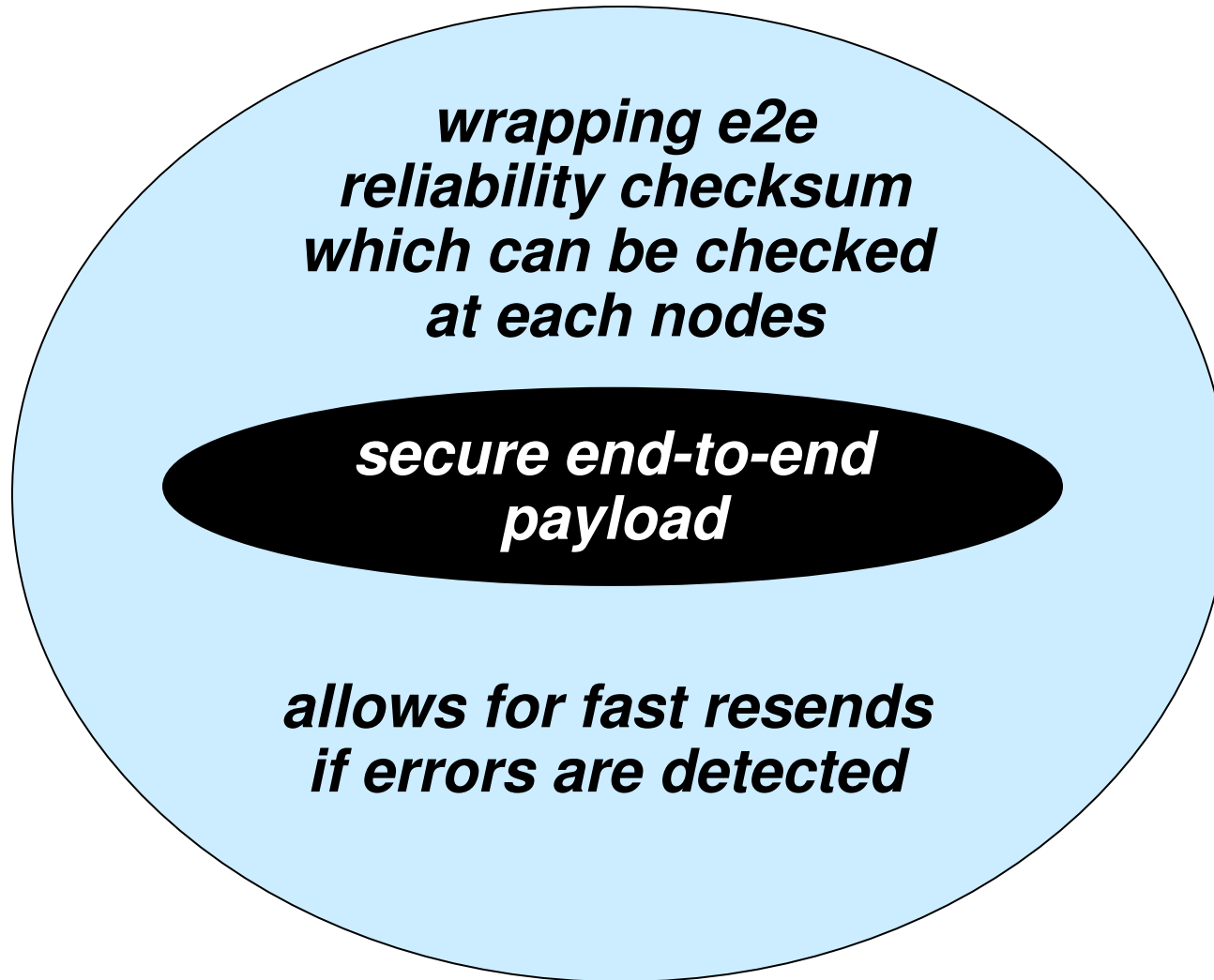
**insecure payload  
using INSECURE ciphersuite**



can also be used by applications  
implementing their own e2esecurity



# Best of both worlds – end-to-end



push an e2e reliability checksum on after the secure PIB is used.