



Overview of JPL in Disruption-Tolerant Networking

Scott Burleigh

Jet Propulsion Laboratory

California Institute of Technology

26 June 2012



From the Beginning

- Convened the first meeting on the Interplanetary Internet – later, Delay-Tolerant Networking (DTN) – concepts, at MCI WorldCom in January 1998.
- Co-founded the DTN Research Group (Internet Research Task Force) in 2002.
- Co-authored the original DTN architecture paper in IEEE Communications Magazine, June 2003: “Delay-Tolerant Networking: An Approach to Interplanetary Internet.”



2003 to 2007

- First tested the Interplanetary Overlay Network (ION) implementation of DTN Bundle Protocol in November 2003.
- Developed deep space networking roadmap for NASA in 2006, resulting in NASA DTN project in 2008.
- Co-authored RFC 4838 (DTN Architecture) published in April 2007.
- Co-authored RFC 5050 (Bundle Protocol) published in November 2007.



2008 to date

- Co-authored RFCs 5325-5327 (Licklider Transmission Protocol) published in September 2008.
- Exercised ION on a spacecraft in interplanetary space (EPOXI) for four weeks in October 2008.
- ION deployed aboard the International Space Station in May 2009.
- JPL's SharedNet system tested over DTN (BBN implementation) for SPAWAR in Jan.-Feb. 2010.
- ION released as open source, through Open Channel, in April 2010 and through SourceForge in June 2011.



Constraints on DTN in Space

	<u>Terrestrial DTN</u>	<u>DTN for Space Flight</u>
Links	Ethernet or WiFi Fast, cheap, symmetrical	Directed, highly attenuated Relatively slow, very expensive, asymmetrical <u>Must use reception/transmission contacts efficiently.</u>
CPU, memory	Commodity generic chips Fast, cheap	Limited-production radiation-hardened chips Relatively slow, very expensive <u>Must use processing resources efficiently.</u>
Resource management	Reboots are easy. Dynamic management of memory is routine.	Hands-on repair is impossible; must minimize risk. Dynamic memory management is unpredictable. <u>Fixed memory allocation is provided at startup.</u>
Operating System	Commercial O/S with memory protection; tasks run in user space.	Real-time O/S, normally no memory protection – all tasks run in kernel space. <u>Must be RTOS-compatible.</u>

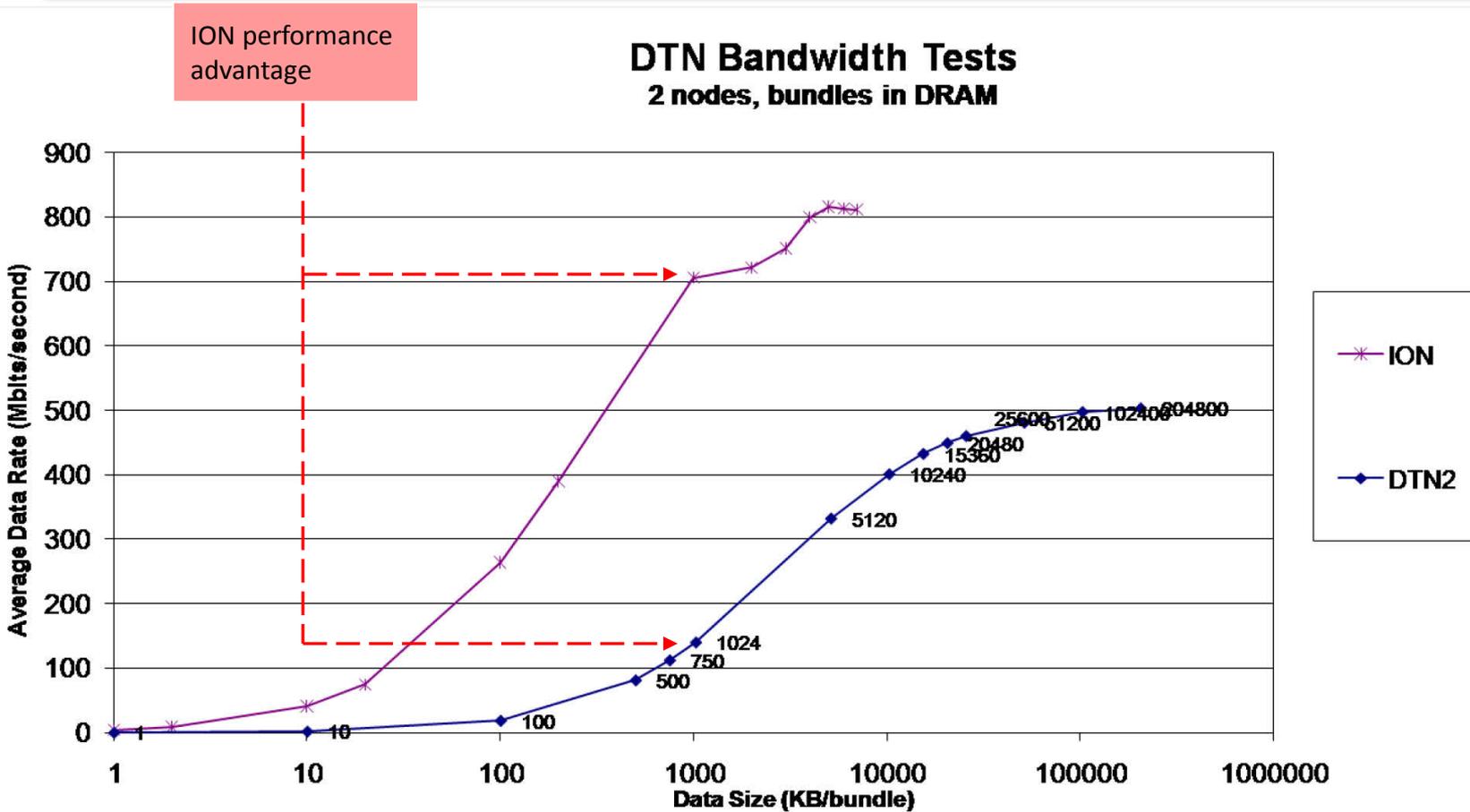


ION's Design

- The design of ION uniquely addresses these constraints.
 - **Built-in private dynamic management of memory** allocated at startup.
 - High-speed shared direct access to **built-in object database**.
 - **System-wide transaction mechanism**, for safety:
 - Ensures mutual exclusion, preventing lockouts and race conditions.
 - Enables reversal of all database updates made within the current transaction in case of software failure.
 - **Compressed bundle headers**, for transmission economy.
 - **Zero-copy objects**, for processing and storage economy.
 - Written in C, for processing economy and small footprint.
 - About 60,000 physical lines of code.
 - About 35,000 logical lines of code (omitting comments and whitespace).
 - Portable among POSIX operating systems, including RTOS.
 - Currently available for Linux, Solaris, OS/X, FreeBSD, VxWorks, RTEMS.
 - Runs natively on Windows – no Cygwin – with MinGW libraries.
 - Ported to “bionic”, first step toward Android port.



Benchmarking results*



ION flight software footprint: about 750 kilobytes including database management system.
(These tests were run on a gigabit Ethernet – don't expect this kind of performance in flight!)

*Courtesy of The Mitre Corporation, September 2006



Current Work

- Supporting the ION open source community via SourceForge.
- NASA Space Internetworking
 - DTN applications
 - Bundle Streaming Service
 - Delay-Tolerant Payload Conditioning
 - Bundle Delivery Time Estimation
 - Congestion forecasting
 - Contact Graph Routing
 - Advanced DTN Quality of Service markings
 - Network management



On the Horizon

- Defining the architecture for the international Solar System Internet.
- Bundle multicast.
- Topology discovery and opportunistic forwarding for ION.