NDN AT INTEL
Andrew Brown
September 29th, 2015
Background

We build integrated, secure, end-to-end telemetry and analytics solutions for the Internet of Things.

We need a way to flexibly move analytics/computation to the most efficient location in the network.
Need: A Common Platform

Flexibility wanted:
- Multiple transports
- Publish/subscribe
- Code on demand
- App-controlled routing
- Intermittent connectivity

Traditional IoT Data Flow

Future IoT Data Flow
Contributions

- Bug fixes, minor features for client libraries (jndn, ndn-js)
- Open sourced several utility libraries (Java 8):
  - jndn-utils: e.g. client to stream data segments, with retries ([http://github.com/01org/jndn-utils](http://github.com/01org/jndn-utils))
  - jndn-management: tool for NFD configuration, i.e. programmatic nfdc ([http://github.com/01org/jndn-management](http://github.com/01org/jndn-management))
  - jndn-mock: for unit testing, to trap and simulate NDN exchanges ([http://github.com/01org/jndn-mock](http://github.com/01org/jndn-mock))
- Working on jndn-forwarder; minimalist NDN forwarder for embedding in Java applications
Challenges

- Best confidentiality mechanism?
  - App-level encryption and key management in progress
  - possibly group-based encryption
  - need to integrate with hardware-level trust anchors
- Random publishers—best approach data randomly generated over time? Sync ✅
- Avoiding network flooding—NFD strategy approach ✅
- Portability—see work on Java NDN forwarder; need memory/processor measurements for current NFD, CCLs
- Implications to enterprise security—e.g., open ports through firewalls
Questions or Comments

Andrew Brown, andrew.brown@intel.com
Eve Schooler, eve.m.schooler@intel.com