Internet of Things Networks - Part 3

IoT and Cloud Services

Aveek Dutta
Assistant Professor
Electrical Engineering and Computer Science
University of Kansas
e-mail: aveekd@ku.edu
http://www.ittc.ku.edu/~aveekd
Overview

● Discussed so far
  ○ What is IoT and communication standards?
  ○ Mostly focused on wireless communication and networking of the network “Edge”

● In this part
  ○ Explore the other half - Content source, management, processing, etc.
  ○ IoT and cloud based services
  ○ New architecture: Edge Computing or Fog Computing

● In the end
  ○ We should have an end-to-end understanding of IoT
  ○ Algorithms, methods and tools for cloud based processing of IoT data
Global Network Architecture

Network Access Facilities

Application service provider
Content provider
Application provider

Network provider
Internet

Network provider

End user

Enterprise network (main campus)
WAN (e.g., MPLS)

IP backbone

Public cellular network

Ethernet LAN
Small-to-medium size business

Residential Wi-Fi network

Networking icons:
- Core router
- Edge/aggregate router
- Router
- Router with firewall
- Ethernet switch
- ATM switch
- Wi-Fi access point
Ethernet still rules the Internet

- Bulk of the Internet is a “Wired” network
- Ethernet is the predominant networking technology
  - Has evolved to 100Gbps (802.3bm)
  - Preserves the MAC. Dense WDM fibre PHY - [Link]
  - Good tutorial [Link]
- Ethernet scales better, relatively secure, low interference, highly interoperable
- Two recent extensions of Ethernet technology
  - Powerline carrier (PLC)
  - Power over Ethernet (PoE)
Enterprise networks and Datacenter

- Dominates the data center where very high data rates are needed to handle massive volumes of data among networked servers and storage units.

- For co-located servers and storage units, high-speed Ethernet fiber links and switches provided the needed networking infrastructure.

- Backplane Ethernet (802.3ap)
  - Runs over copper jumper cables that can provide up to 100 Gbps over very short distances.
  - Blade servers - Multiple server modules housed in a single chassis.
Cloud Computing Concepts

- NIST defines the essential characteristics of cloud computing as:
  - Broad network access
    - Thin and thick client
  - Rapid elasticity
    - On-demand resource
  - Measured service
    - Scheduling and optimization
  - On-demand self service
    - User provisioning
  - Resource pooling
    - Virtualization of resources
Network Convergence

- Convergence in terms of a three-layer model of enterprise communications:
  - Application convergence
    - These are seen by the end users of a business
    - Convergence integrates communications applications with business applications
    - Enterprise 2.0, SCM, BPM, CRM, etc
  - Enterprise services
    - Services to ensure that users can take full advantage of the applications
    - Privacy, printing, collaboration, storage, etc
  - Infrastructure
    - Communication links, LANs, WANs, and Internet connections available to the enterprise
    - Converged IP-based networks
Requirements - Traffic type

- Elastic Traffic
  - Can adjust, over wide ranges, to changes in delay and throughput across an internet and still meet the needs of its applications
  - Is the traditional type of traffic supported on TCP/IP based internets
  - Is the type of traffic for which Internet was designed
  - Applications that generate such traffic typically use Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) as a transport protocol
  - Smaller transfers are dominated by end-to-end propagation delay, while large data transfer is dominated by TCP congestion control
Requirements - Traffic type

- Inelastic Traffic
  - Does not easily adapt, if at all, to changes in delay and throughput across an internet
  - Examples of inelastic traffic include: multimedia transmission such as voice and video, and high-volume interactive traffic
  - Requirements may include:
    - Throughput
      - A minimum throughput value may be required
    - Delay
      - Also called latency
    - Delay jitter
      - The magnitude of delay variation is a critical factor in real-time applications
# Application Requirement

<table>
<thead>
<tr>
<th>Application Category</th>
<th>Service Class</th>
<th>Traffic Characteristics</th>
<th>Tolerance to Loss</th>
<th>Tolerance to Delay</th>
<th>Tolerance to Jitter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control</td>
<td>Network control</td>
<td>Variable-size packets, mostly inelastic short messages, but traffic can also burst (BGP)</td>
<td>Low</td>
<td>Low</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>OA&amp;M</td>
<td>Variable-size packets, elastic and inelastic flows</td>
<td>Low</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>Media-Oriented</td>
<td>Telephony</td>
<td>Fixed-size small packets, constant emission rate, inelastic and low-rate flows</td>
<td>Very low</td>
<td>Very low</td>
<td>Very low</td>
</tr>
<tr>
<td></td>
<td>Real-time interactive</td>
<td>RTP/UDP streams, inelastic, mostly variable rate</td>
<td>Low</td>
<td>Very low</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td>Multimedia conferencing</td>
<td>Variable-size packets, constant transmit interval, rate adaptive, reacts to loss</td>
<td>Low-medium</td>
<td>Very low</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td>Broadcast video</td>
<td>Constant and variable rate, inelastic, non-bursty flows</td>
<td>Very low</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td></td>
<td>Multimedia Streaming</td>
<td>Variable-size packets, elastic with variable rate</td>
<td>Low-medium</td>
<td>Medium</td>
<td>Yes</td>
</tr>
<tr>
<td>Data</td>
<td>Low-latency data</td>
<td>Variable rate, bursty short-lived elastic flows</td>
<td>Low</td>
<td>Low-medium</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>High-throughput data</td>
<td>Variable rate, bursty long-lived elastic flows</td>
<td>Low</td>
<td>Medium-high</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Low-priority data</td>
<td>Non-real-time and elastic</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
</tr>
<tr>
<td>Best effort</td>
<td>Standard</td>
<td>A bit of everything</td>
<td>Not specified</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Application based QoS

<table>
<thead>
<tr>
<th>Application Type</th>
<th>QoS Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Voice</strong></td>
<td>One-way latency $\leq 150$ ms.</td>
</tr>
<tr>
<td></td>
<td>One-way peak-to-peak jitter $\leq 30$ ms.</td>
</tr>
<tr>
<td></td>
<td>Per-hop peak-to-peak jitter $\leq 10$ ms.</td>
</tr>
<tr>
<td></td>
<td>Packet loss $\leq 1$ percent.</td>
</tr>
<tr>
<td><strong>Broadcast video</strong></td>
<td>Packet loss $\leq 0.1$ percent.</td>
</tr>
<tr>
<td><strong>Real-time interactive video</strong></td>
<td>One-way latency $\leq 200$ ms.</td>
</tr>
<tr>
<td></td>
<td>One-way peak-to-peak jitter $\leq 50$ ms.</td>
</tr>
<tr>
<td></td>
<td>Per-hop peak-to-peak jitter $\leq 10$ ms.</td>
</tr>
<tr>
<td></td>
<td>Packet loss $\leq 0.1$ percent.</td>
</tr>
<tr>
<td><strong>Multimedia conferencing</strong></td>
<td>One-way latency $\leq 200$ ms.</td>
</tr>
<tr>
<td></td>
<td>Packet loss $\leq 1$ percent.</td>
</tr>
<tr>
<td><strong>Multimedia streaming</strong></td>
<td>One-way latency $\leq 400$ ms.</td>
</tr>
<tr>
<td></td>
<td>Packet loss $\leq 1$ percent.</td>
</tr>
</tbody>
</table>

QoS Requirement for Application class
Real-time Traffic

(a) Continuous data source

(b) Voice source with silent intervals

(c) Compressed video source
Quality of Service (QoS)

- The measurable end-to-end performance properties of a network service, which can be guaranteed in advance by a service level agreement (SLA) between a user and a service provider, so as to satisfy specific customer application requirements. Commonly specified properties include:
  - Throughput
  - Delay
  - Packet jitter
  - Error rate
  - Packet loss
  - Priority
  - Availability
  - Security
Quality of Experience (QoE)

- A subjective measure of performance as reported by the user; relies on human opinion (makes things interesting)
  - Important with multimedia content delivery

- QoS processes by themselves are not sufficient in that they do not take into account the user’s perception of network performance and service quality

- Categories of factors and features that can be included in QoE are:
  - Perceptual - brightness, flicker, distortion, etc
  - Psychological - usefulness, satisfaction, etc.
  - Interactive - responsiveness, accessibility, etc
Evolving Network Requirements

- New trends, new challenges
  - Demand is increasing - Cloud computing, Big data, Mobile traffic, IoT
- QoS and QoE requirements are expanded, need agile networks fashion
- The traditional TCP/IP architecture is limited by
  - Static, complex architecture
  - Inconsistent policies
  - Inability to scale
  - Vendor dependence
## The Software Defined Networking (SDN) Approach

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adaptability</td>
<td>Networks must adjust and respond dynamically, based on application needs, business policy, and network conditions</td>
</tr>
<tr>
<td>Automation</td>
<td>Policy changes must be automatically propagated so that manual work and errors can be reduced</td>
</tr>
<tr>
<td>Maintainability</td>
<td>Introduction of new features and capabilities must be seamless with minimal disruption of operations</td>
</tr>
<tr>
<td>Model management</td>
<td>Network management software must allow management of the network at a model level, rather than implementing conceptual changes by reconfiguring individual network elements</td>
</tr>
<tr>
<td>Mobility</td>
<td>Control functionality must accommodate mobility, including mobile user devices and virtual servers</td>
</tr>
<tr>
<td>Integrated security</td>
<td>Network applications must integrate seamless security as a core service instead of as an add-on solution</td>
</tr>
<tr>
<td>On-demand scaling</td>
<td>Implementations must have the ability to scale up or scale down the network and its services to support on-demand requests</td>
</tr>
</tbody>
</table>
SDN

Control Plane

Data Plane

SDN control platform

Network services

Network infrastructure
Virtualization

- A driving factor for SDN is the widespread use of server virtualization
- Virtualization masks server resources from users
  - Makes it possible to partition a single machine into multiple, independent servers, conserving hardware resources
- Migrate a server from one machine to another for load balancing or for dynamic switchover in the case of machine failure (very important)
- Virtualization is central in dealing with big data applications

Image Source: http://www.camsbluewiretech.com/resources/vmware_virt.gif
Characteristics of SDN

- The control plane is separated from the data plane;
  - Data plane devices become simple packet-forwarding devices
- The controller is portable software that can run on commodity servers
- Open interfaces are defined between the controllers and those in the data plane
- The network is programmable by applications running on top of the SDN controllers
- REST APIs for SDN [Link1], [Link2]
SDN Control and Application Plane

- Controller operates like an Network Operating System (NOS)
  - APIs provide abstractions to applications and data plane hardware
  - Many open source platform available - OpenDaylight, ONOS, Floodlight, etc
- Application plane (northbound APIs are under development)
Network Function Virtualisation (NFV)

- SDN can be applied to network elements as well
- Decouples network functions, like routing, firewall, NATs, from proprietary HW
Virtualization of Cloud Resources

- OpenStack aims to produce an open source cloud operating system
- Provides multi-tenant Infrastructure as a Service (IaaS) and aims to meet the needs of public and private clouds regardless of size
  - Simple to implement and massively scalable
- SDN technology is expected to contribute to its networking part, and to make the cloud operating system more efficient, flexible, and reliable

Image Source: https://sathisharthars.files.wordpress.com/2014/03/openstack-sm.png
Cloud Computing Elements

Essential Characteristics:
- Broad Network Access
- Rapid Elasticity
- Measured Service
- On-Demand Self-Service

Service Models:
- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

Deployment Models:
- Public
- Private
- Hybrid
- Community

Resource Pooling
Responsibilities
XaaS

• XaaS is the latest development in the provisioning of cloud services

• XaaS is becoming increasingly attractive to customers because:
  ○ Total costs are controlled and lowered
  ○ Risks are lowered
  ○ Innovation is accelerated

• The acronym has three generally accepted interpretations:
IoT and Fog Computing

Any TIME connection
- night
- daytime
- on the move
- outdoor
- indoor (away from the computer)
- at the computer

Any PLACE connection
- between computers
- human to human, not using computer
- human to thing, using generic equipment
- thing to thing

Any THING connection
Data model in IoT
Modern Networking Schema

Quality of experience (QoE)

Quality of service (QoS)

Software defined networking (SDN)

Network functions virtualization (NFV)

Network devices
Curse of Dimensionality

- Old method to tackle massive scalability challenges
  - Parallel processing using supercomputers
  - Moving data can be costlier than processing
- Bring computation close to the data
- Solution is “cluster-computing” using commodity hardware
  - How do we handle a thousand machine failures each day?
  - How do we write a software that runs over a million machines?
  - How do we make it easy to write such a software?
- So what do we need?
  - A super-cool file system and a programming model
Why MapReduce?

“In Pioneer days, they used oxen for heavy pulling. When one ox couldn’t budge a log, they didn’t try to grow a larger ox…

We shouldn’t be trying to grow bigger computers, but to add more systems of computers.”

Grace Hopper

(Read This)

The MapReduce paradigm helps you add more oxen

By definition, big data is too large to handle by conventional means. Sooner or later, you just can’t scale up anymore
What is MapReduce?

- A parallel programming model suitable for big data processing
  - Split data into distributable chunks ("shards")
  - Define the steps to process those chunks
  - Run that process in parallel on the chunks
- Scalable by adding more machines to process chunks
  - Leverage commodity hardware to tackle big jobs
- The foundation for Hadoop
  - MapReduce is a parallel programming model
  - Hadoop is a concrete platform that implements MapReduce
When to use

- Problems that are “embarrassingly parallel”
- Examples
  - Word count
  - Reverse index
  - tf-idf
  - Distributed `grep` and distributed object recognition ("Where's Waldo?")
  - Distributed "associative" aggregation (marginalization, sum; mean if you track both numerator and denominator; min or max; count)
- Original Paper from Google on MapReduce
Map part of MapReduce

- **Transform**
  - (Map) input values to output values: \(<k_1,v_1> \rightarrow <k_2,v_2>\)

- **Input – Key/Value Pairs**
  - For instance, Key = line number, Value = text string

- **Map Function**
  - Steps to transform input pairs to output pairs
    - For example, count the different words in the input

- **Output – Key/Value Pairs**
  - For example, Key = <word>, Value = <count>

- **Map output is the input to Reduce**
Reduce part of MapReduce

- Merge (Reduce) Values from the Map phase
  - Reduce is optional. Sometimes all the work is done in the Mapper
- Input
  - Values for a given Key from all the Mappers
- Reduce Function
  - Steps to combine (Sum?, Count?, Print?,...) the values
- Output
  - Print values?, load into a DB? send to the next MapReduce job?
Motivating Example: Word Count

This is the “Hello World” of MapReduce

Distribute the text of millions of documents over hundreds of machines.

MAPPERS can be word-specific. They run through the stacks and shout “One!” every time they see the word “beach”

REDUCERS listen to all the Mappers and total the counts for each word.
### Example: Find Common Friends

<table>
<thead>
<tr>
<th>Friend List</th>
<th>MAP Step</th>
<th>REDUCE Step</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>A -&gt; B C D</td>
<td>For map(A -&gt; B C D): (A B) -&gt; B C D (A C) -&gt; B C D (A D) -&gt; B C D</td>
<td>Before we send these key-value pairs to the reducers, we group them by their keys and get: (A B) -&gt; (A C D E) (B C D) (A C) -&gt; (A B D E) (B C D) (A D) -&gt; (A B C E) (B C D)</td>
<td>The result after reduction is: (A B) -&gt; (C D) (A C) -&gt; (B D) (A D) -&gt; (B C)</td>
</tr>
<tr>
<td>B -&gt; A C D E</td>
<td>(B C) -&gt; A C D E (B D) -&gt; A C D E (B E) -&gt; A C D E</td>
<td>(B C) -&gt; (A B D E) (A C D E) (B D) -&gt; (A B C E) (A C D E) (B E) -&gt; (A C D E) (B C D)</td>
<td>(B C) -&gt; (A D E) (B D) -&gt; (A C E) (B E) -&gt; (C D)</td>
</tr>
<tr>
<td>C -&gt; A B D E</td>
<td>For map(B -&gt; A C D E):</td>
<td>(C D) -&gt; (A B C E) (A B D E)</td>
<td>(C D) -&gt; (A B E)</td>
</tr>
<tr>
<td>D -&gt; A B C E</td>
<td>(B C) -&gt; A C D E (B D) -&gt; A C D E (B E) -&gt; A C D E</td>
<td>(C E) -&gt; (A B D E) (B C D) (D E) -&gt; (A B C E) (B C D)</td>
<td>(C E) -&gt; (B D) (D E) -&gt; (B C)</td>
</tr>
<tr>
<td>E -&gt; B C D</td>
<td>(B C) -&gt; A C D E (B D) -&gt; A C D E (B E) -&gt; A C D E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Suppose you have 517,424 emails from an energy company under indictment.

The social network may be implied by all To:From pairings in the emails, with reflexivity accounted for.

Almost one month ago, Special Agent Carl Wake of the FBI called me about the Steve Todoroff investigation. He indicated that the FBI had recently learned of the article about EOTT’s NGL theft that appeared in the business section of the Houston Chronicle. Mr. Wake said it might be a matter the FBI would like to investigate. I told Mr. Wake that EOTT was currently working with the Harris County District Attorney on the prosecution of this matter, and I thanked him for the FBI’s interest. He told me that the FBI might want to work with the Harris County District Attorney in investigating this matter, and he stated that there may be investigative information that the FBI can obtain more quickly than the Harris County District Attorney. Mr. Wake requested a copy of the materials we had provided to the Harris County District Attorney.

In order to avoid damage to the good rapport we have established with Assistant District Attorney Bill Moore, I asked John DeGeeter to call Bill Moore and advise him of the contact that had been made by the FBI. Bill Moore agreed to call Carl Wake and work with Mr. Wake on his request for the materials provided by EOTT.

Carl Wake called me again yesterday. He has been working with Bill Moore. Mr. Wake stated it was too early to speculate as to what charges would be brought. He did say that our materials clearly indicated federal wire fraud and possibly mail fraud. He said that where there is wire fraud, there is usually money laundering.

The purpose of Mr. Wake’s call yesterday was to inquire about the status of some interview summaries that John DeGeeter and I have prepared and collected at the request of Bill Moore. Mr. Wake requested that EOTT send a copy of the summaries to him when we sent the summaries to Bill Moore. Those summaries were sent out today.

I gathered from my calls with Carl Wake that the FBI is very interested in taking an active part in this investigation. In order to build on the relationship we have established with Bill Moore, we will continue to direct our inquiries about the investigation to Mr. Moore until he tells us to do otherwise.
Social Triangle: First Directed Edge

Mapper1

Maps two regular expression searches:
  To: Michael, Dan, Lori, Susan
  From: Walt

Emits the outbound directed edge of the social graph:
  <Key, Value> = <Walt, [Michael, Dan, Lori, Susan]>

Reducer1

Gets the output from the mapper with different values
  <Key, Value> = <Walt, [Michael, Dan, Lori, Susan]>
  <Key, Value> = <Walt, [Lori, Susan, Jeff, Ken]>

Unions the values for the second directed edge:
  <Key, Value> = <Walt, [Dan, Jeff, Ken, Lori, Michael, Susan]>
Social Triangle: Second Directed Edge

Mapper2

Reverses the previous Map:
To: Michael, Dan, Lori, Susan
From: Walt
Emits the inbound directed edge of the social graph:
<Key, Value> = <Susan, Walt>; <Lori, Walt>; <Dan, Walt>; etc

Reducer2

Gets the output from the mapper with different values
<Key, Value> = <Susan, Walt>
<Key, Value> = <Susan, Jeff>

Unions the values for the third directed edge:
<Key, Value> = <Susan, [Jeff, Ken, Walt]>
Social Triangle: Third Directed Edge

Mapper3

Join [inbound] and [outbound] lists by Key
Walt, [Jeff, Ken, Lori, Susan], [Jeff, Lori, Stanley]

Emits <Person, Person> pair with level of association:
<Key, Value> = <Walt: Jeff reciprocal>; <Walt: Stanley directed>, etc

Reducer3

Reducer unions the output of the mappers and presents rules:
<Key, Value> = <Walt::Jeff, reciprocal>
<Key, Value> <Walt::Stanley, directed>

The third reducer can shape the data any way that serves the business objective.
What is Hadoop?

People use “Hadoop” to mean one of four things:

- MapReduce paradigm.
- Massive unstructured data storage on commodity hardware.
- Java Classes for HDFS types and MapReduce job management.
- HDFS: The Hadoop distributed file system.

With Hadoop, you can do MapReduce jobs quickly and efficiently.
What do we mean by Hadoop

- A framework for performing big data analytics
  - An implementation of the MapReduce paradigm
  - Hadoop glues the storage and analytics together and provides reliability, scalability, and management

Two Main Components

Storage (Big Data)
- HDFS – Hadoop Distributed File System
- Reliable, redundant, distributed file system optimized for large files

MapReduce (Analytics)
- Programming model for processing sets of data
- Mapping inputs to outputs and reducing the output of multiple Mappers to one (or a few) answer(s)
Hadoop and HDFS

Hadoop File System (HDFS) is a distributed file system designed for large datasets. It is part of the Hadoop ecosystem and is designed to scale easily to a large number of commodity servers. HDFS consists of a NameNode, which manages file system namespace and regulates file system access to clients, and DataNodes, which store the files and handle the data access requests. The data is replicated across DataNodes to ensure fault tolerance and high availability.
Hadoop Operational Modes

- **Java MapReduce Mode**
  - Write Mapper, Combiner, Reducer functions in Java using Hadoop Java APIs
  - Read records one at a time

- **Streaming Mode**
  - Uses *nix pipes and standard input and output streams
  - Any language (Python, Ruby, C, Perl, Tcl/Tk, etc.)
  - Input can be a line at a time, or a stream at a time
Hadoop defines a set of classes that extend the scalar classes in Java (examples: IntWritable, Text)

Hadoop offers a number of base classes to provide a framework for jobs

This Mapper incorporates the MapReduceBase, Reporter and OutputCollector classes explicitly

More tutorials and resources [Link]
Summary

- What matters for Cloud based resources and why are those necessary
- A paradigm shift from data-centric to user-centric networking
- We have learnt the language of Modern Networking
  - What, Why and Hows of Software Defined Networking, Virtualization of network resources
- Introduction to MapReduce, one of the key enablers in distributed high performance computing