Cloud Computing – Where ISR Data Will Go for Exploitation

22 September 2009

Albert Reuther, Jeremy Kepner, Peter Michaleas, William Smith

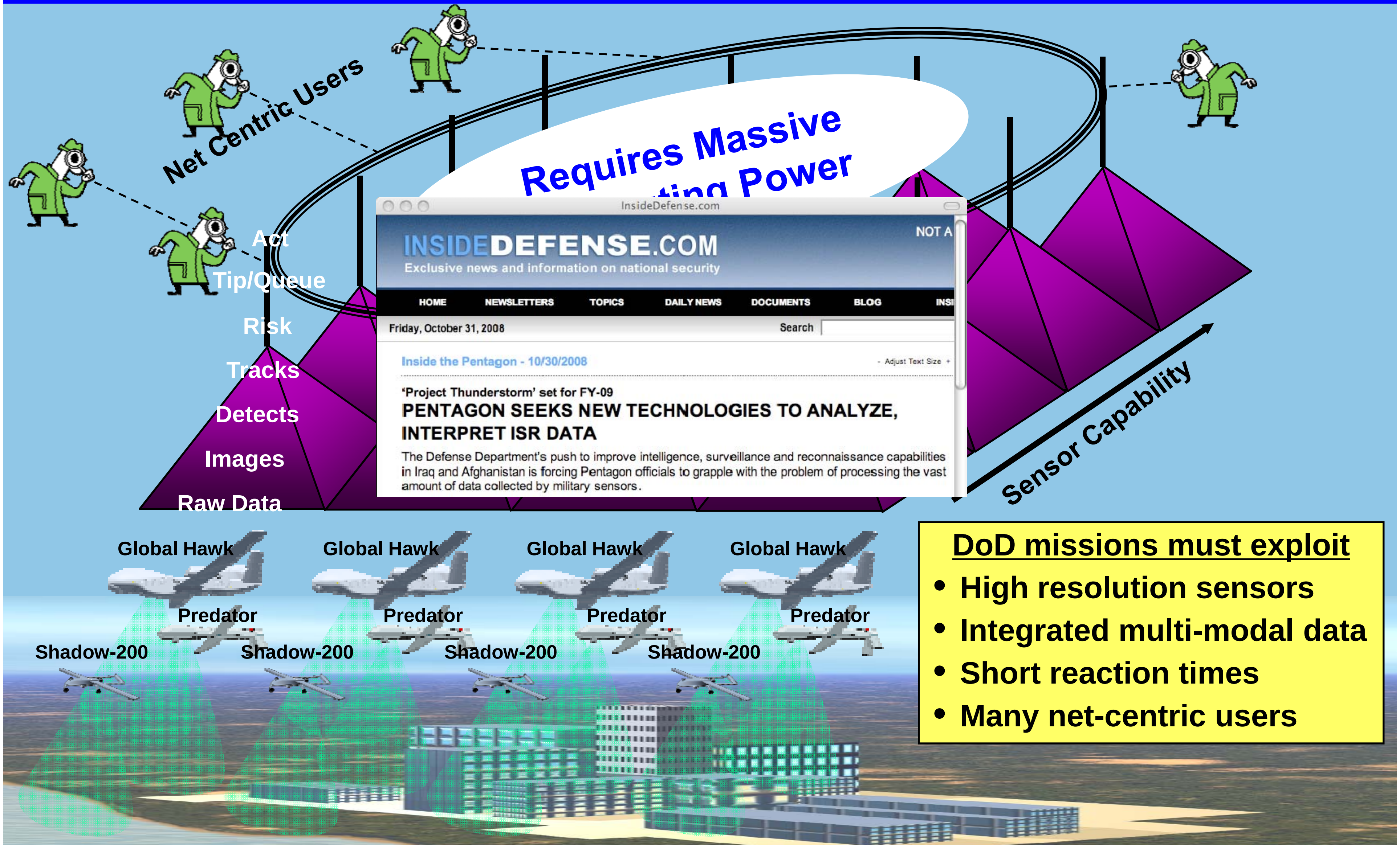
This work is sponsored by the Department of the Air Force under Air Force contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

Outline

- **Introduction**
- **Cloud Supercomputing**
- **Integration with Supercomputing System**
- **Preliminary Results**
- **Summary**

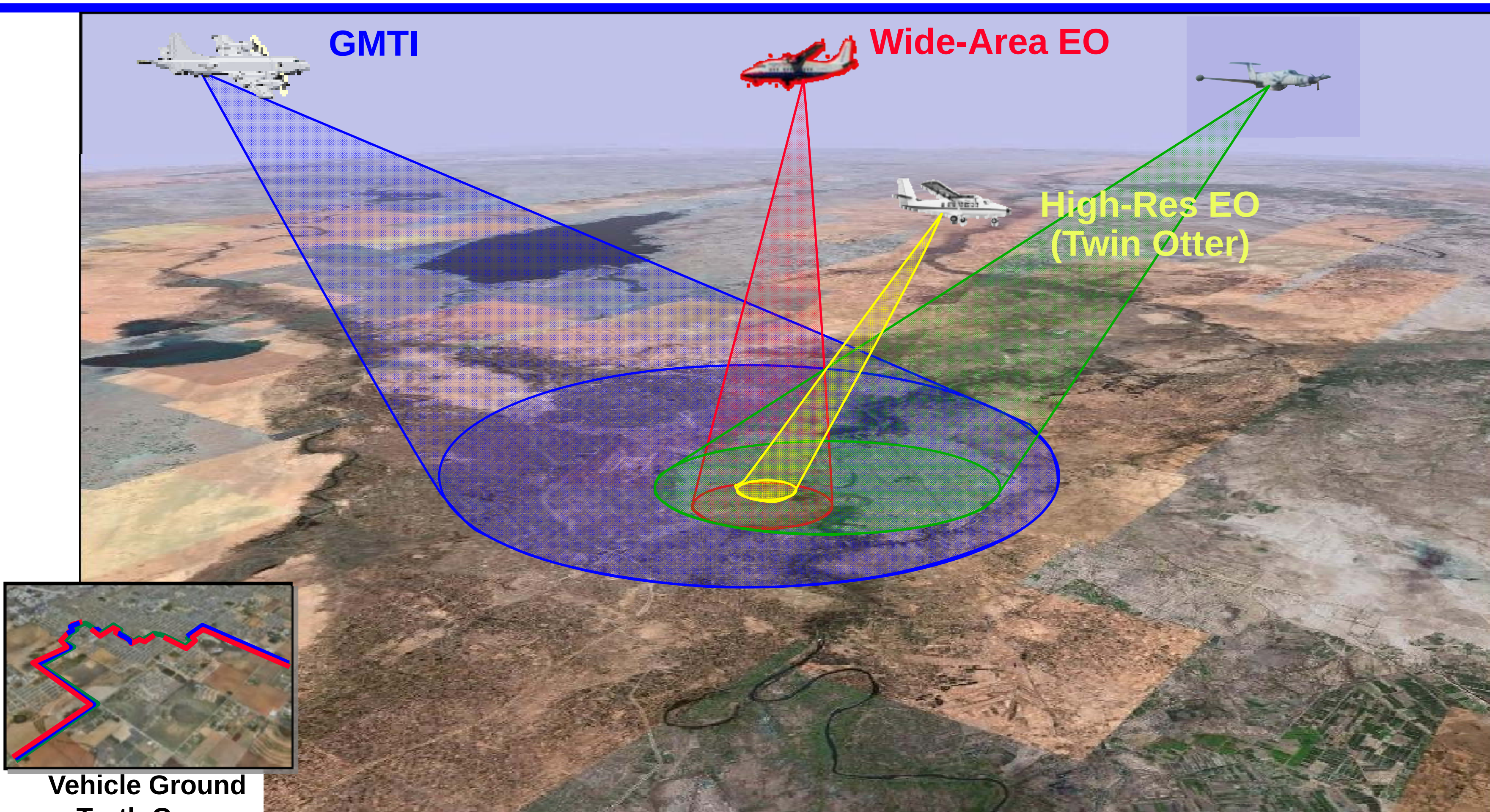
- ***Persistent surveillance requirements***
- ***Data Intensive cloud computing***

Persistent Surveillance: The "New" Knowledge Pyramid



DoD missions must exploit
- High resolution sensors
- Integrated multi-modal data
- Short reaction times
- Many net-centric users

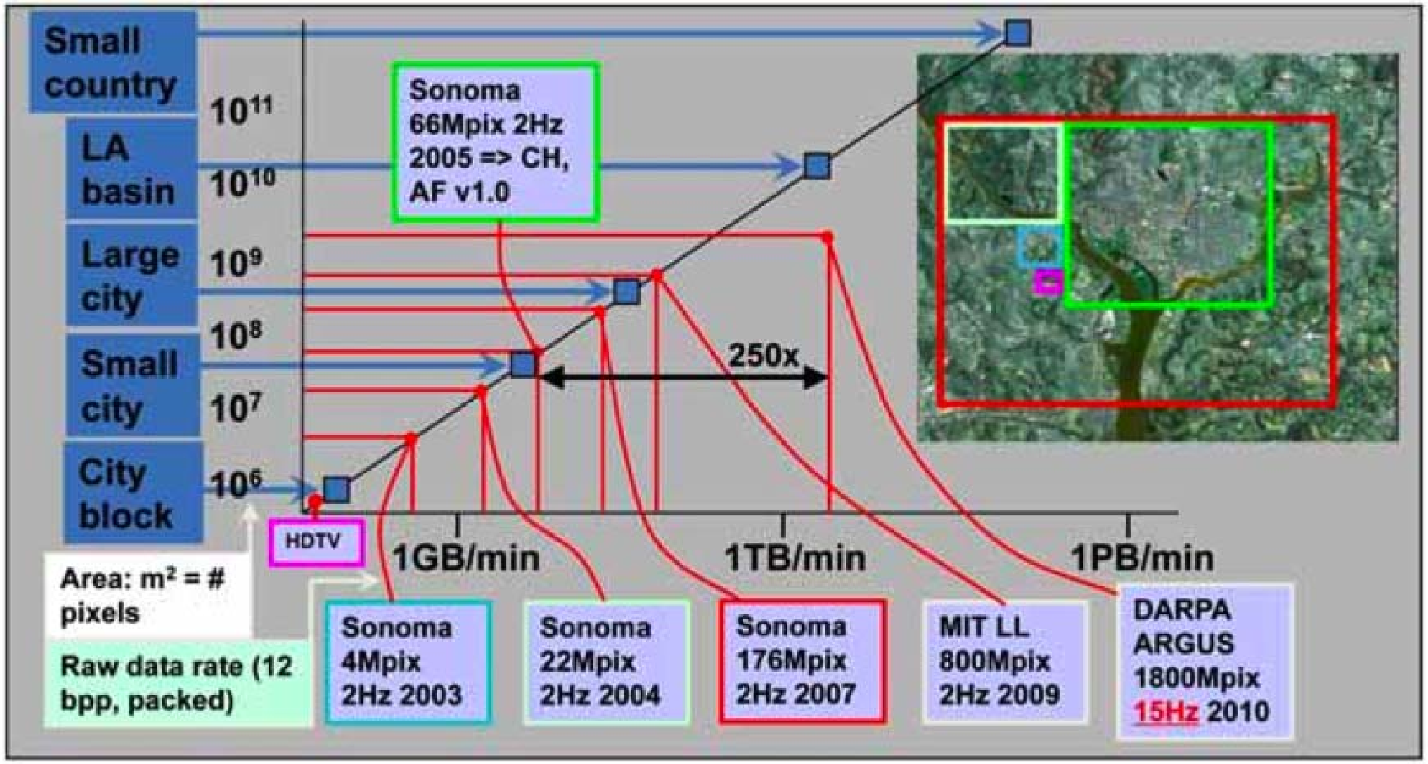
Bluegrass Dataset (detection/tracking)



Vehicle Ground Truth Cues

- Terabytes of data; multiple classification levels; multiple teams
- Enormous computation to test new detection and tracking algorithms

Persistent Surveillance Data Rates



- Persistent Surveillance requires watching large areas to be most effective
- Surveilling large areas produces enormous data streams
- Must use distributed storage and exploitation

Cloud Computing Concepts

Data Intensive Computing

- **Compute architecture for large scale data analysis**
  - Billions of records/day, trillions of stored records, petabytes of storage
    - Google File System 2003
    - Google MapReduce 2004
    - Google BigTable 2006
- **Design Parameters**
  - Performance and scale
  - Optimized for ingest, query and analysis
  - Co-mingled data
  - Relaxed data model
  - Simplified programming
- **Community:**

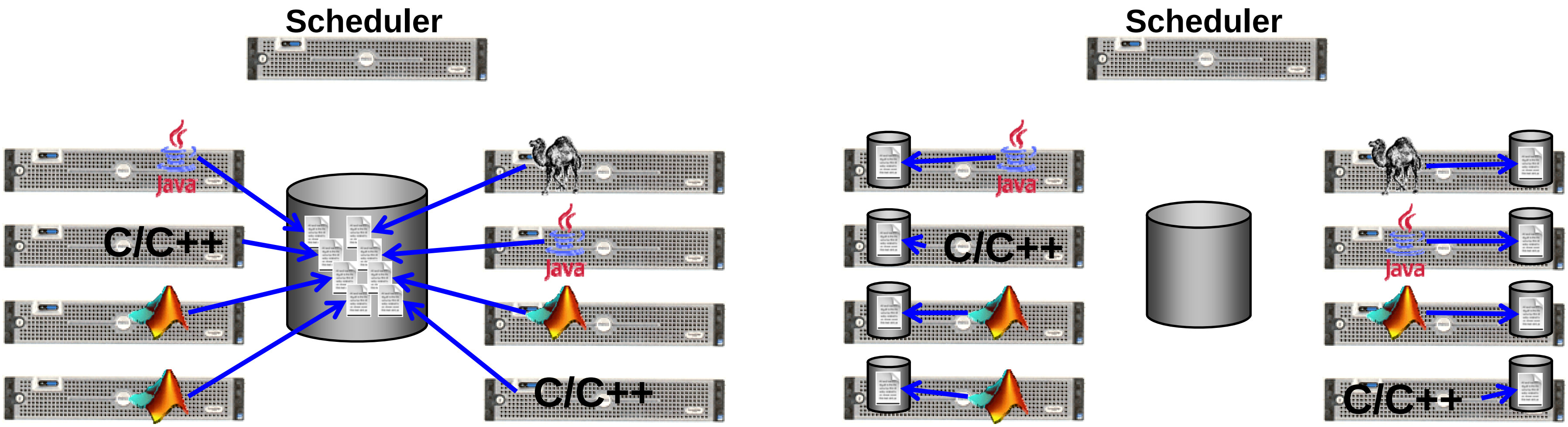
Utility Computing

- **Compute services for outsourcing IT**
  - Concurrent, independent users operating across millions of records and terabytes of data
    - IT as a Service
    - Infrastructure as a Service (IaaS)
    - Platform as a Service (PaaS)
    - Software as a Service (SaaS)
- **Design Parameters**
  - Isolation of user data and computation
  - Portability of data with applications
  - Hosting traditional applications
  - Lower cost of ownership
  - Capacity on demand
- **Community:**

Advantages of Data Intensive Cloud: Disk Bandwidth

Traditional:
Data from central store to compute nodes

Cloud:
Data replicated on nodes, computation sent to nodes



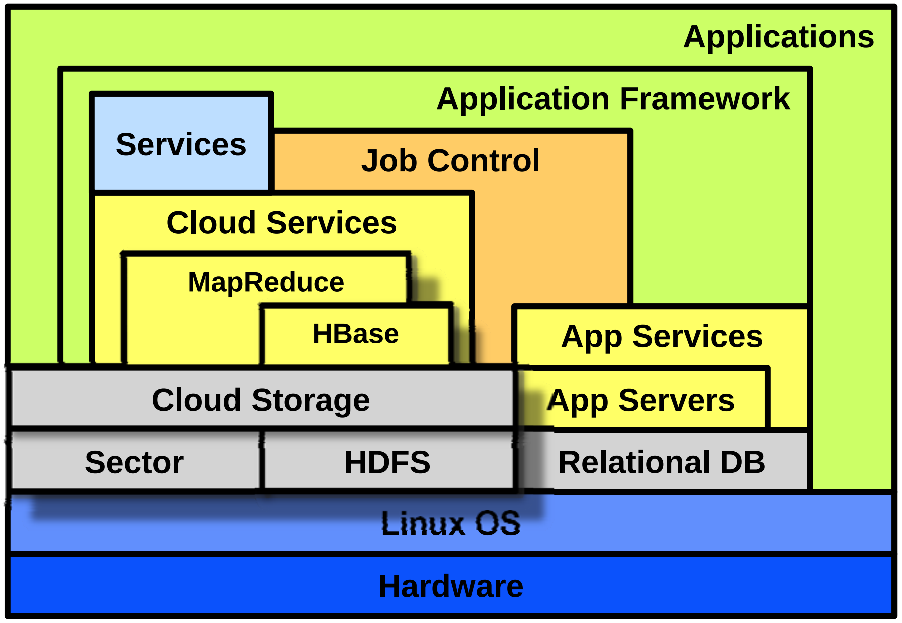
- **Cloud computing moves computation to data**
  - **Good for applications where time is dominated by reading from disk**
- **Replaces expensive shared memory hardware and proprietary database software with cheap clusters and open source**
  - **Scalable to hundreds of nodes**

Outline

- Introduction
- **Cloud Supercomputing**
  - ***Cloud stack***
  - ***Distributed file systems***
  - ***Distributed database***
  - ***Distributed execution***
- Integration with Supercomputing System
- Preliminary Results
- Summary

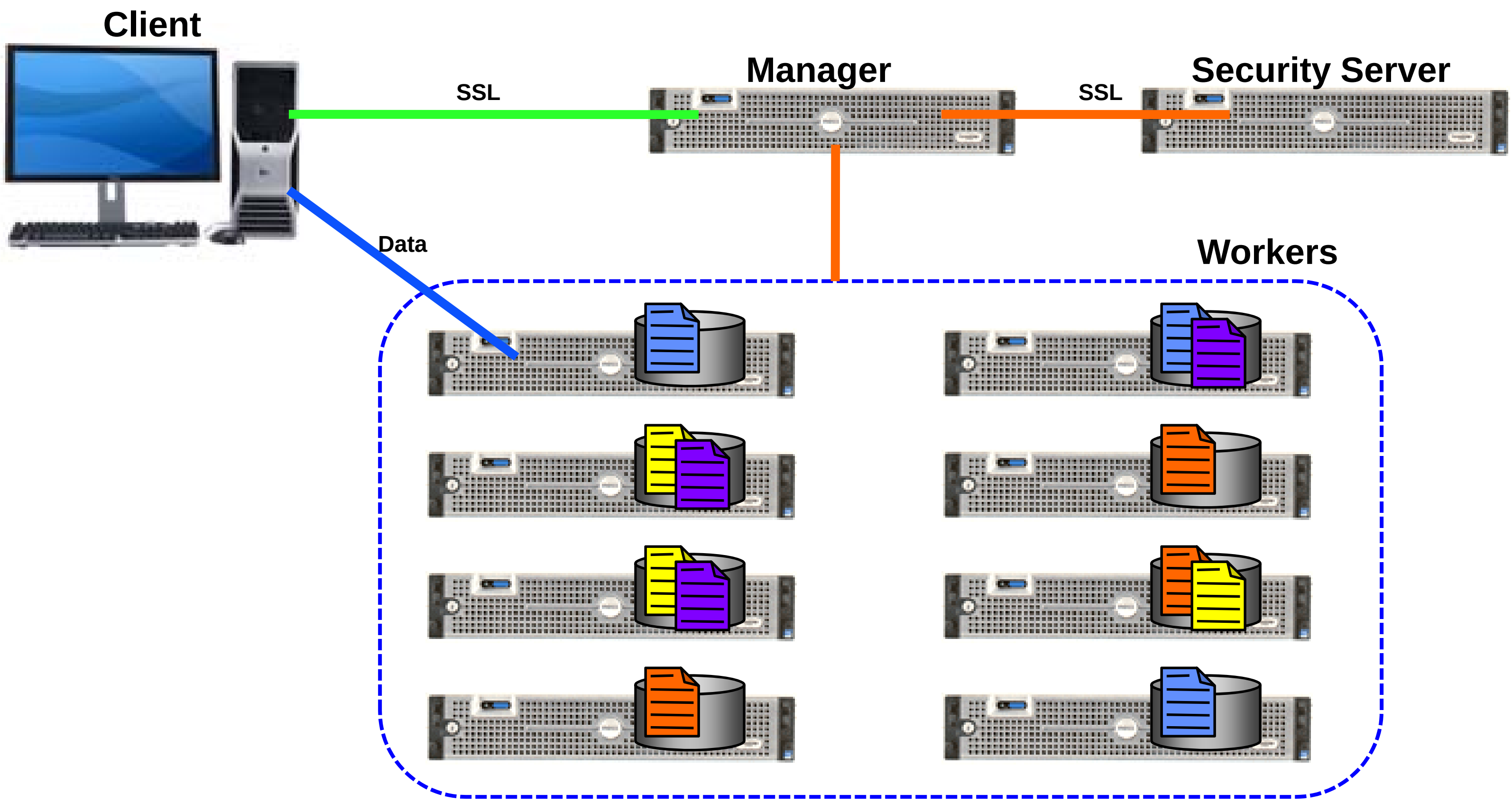
Cloud Software: Hybrid Software Stacks

- **Cloud implementations can be developed from a large variety of software components**
  - Many packages provide overlapping functionality
- **Effective migration of DoD to a cloud architecture will require mapping core functions to the cloud software stack**
  - Most likely a hybrid stack with many component packages
- **MIT-LL has developed a dynamic cloud deployment architecture on its computing infrastructure**
  - Examining performance trades across software components



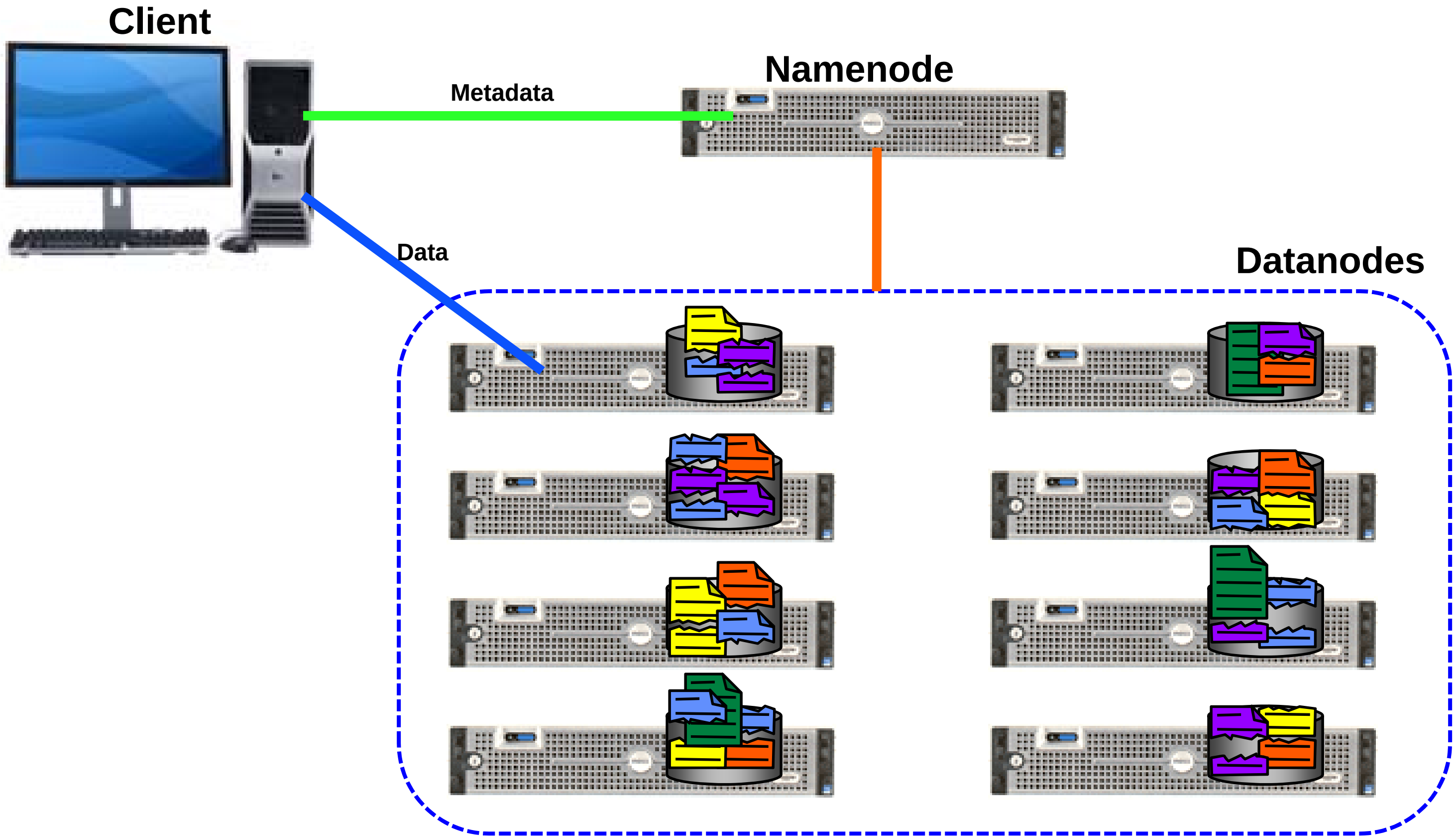
- **Distributed file systems**
  - File-based: Sector
  - Block-based: Hadoop DFS
- **Distributed database: HBase**
- **Compute environment: Hadoop MapReduce**

P2P File system (e.g., Sector)



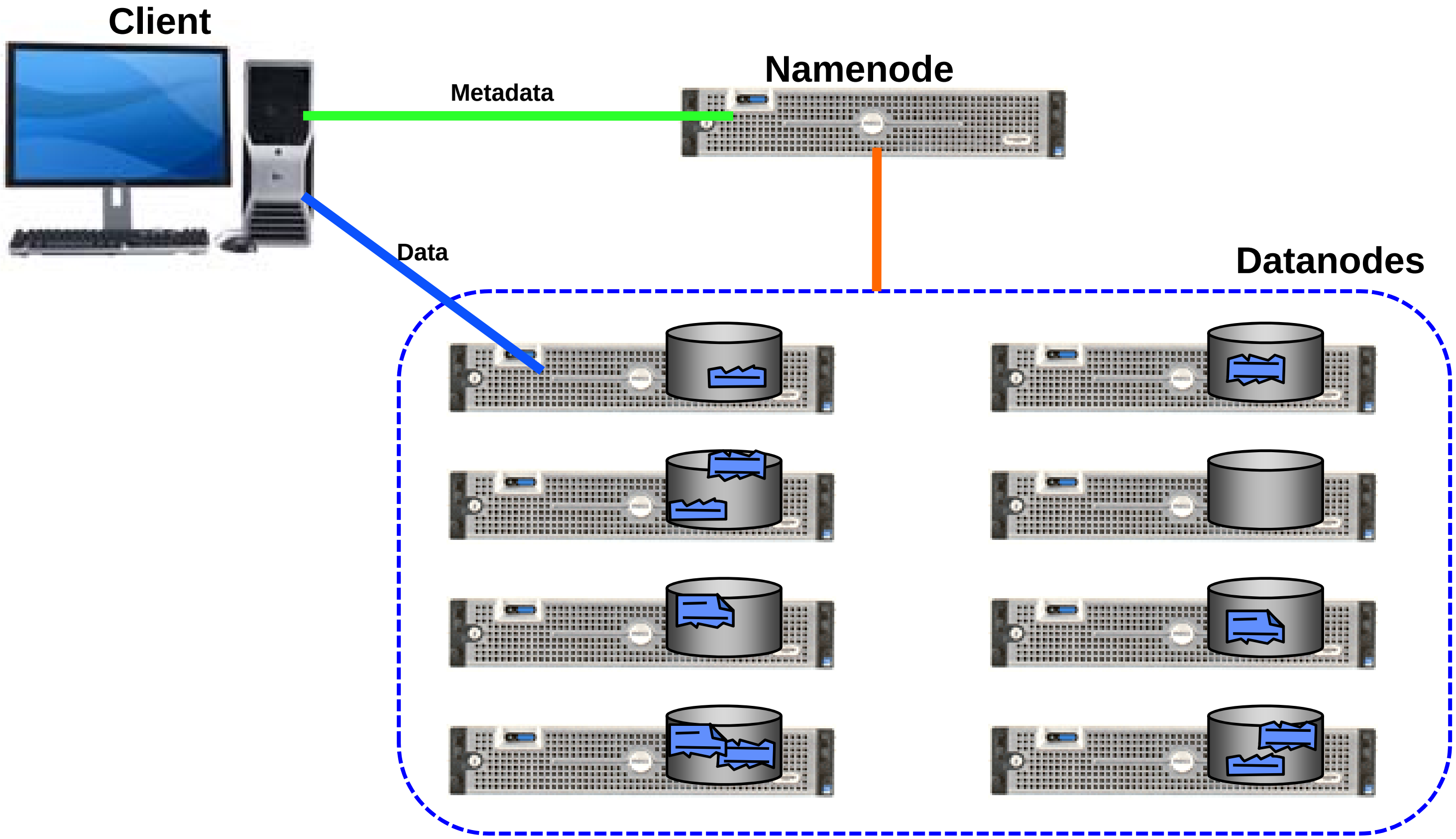
- Low-cost, file-based, “read-only”, replicating, distributed file system
- Manager maintains metadata of distributed file system
- Security Server maintains permissions of file system
- Good for mid sized files (Megabytes)
  - Holds data files from sensors

Parallel File System (e.g., Hadoop DFS)



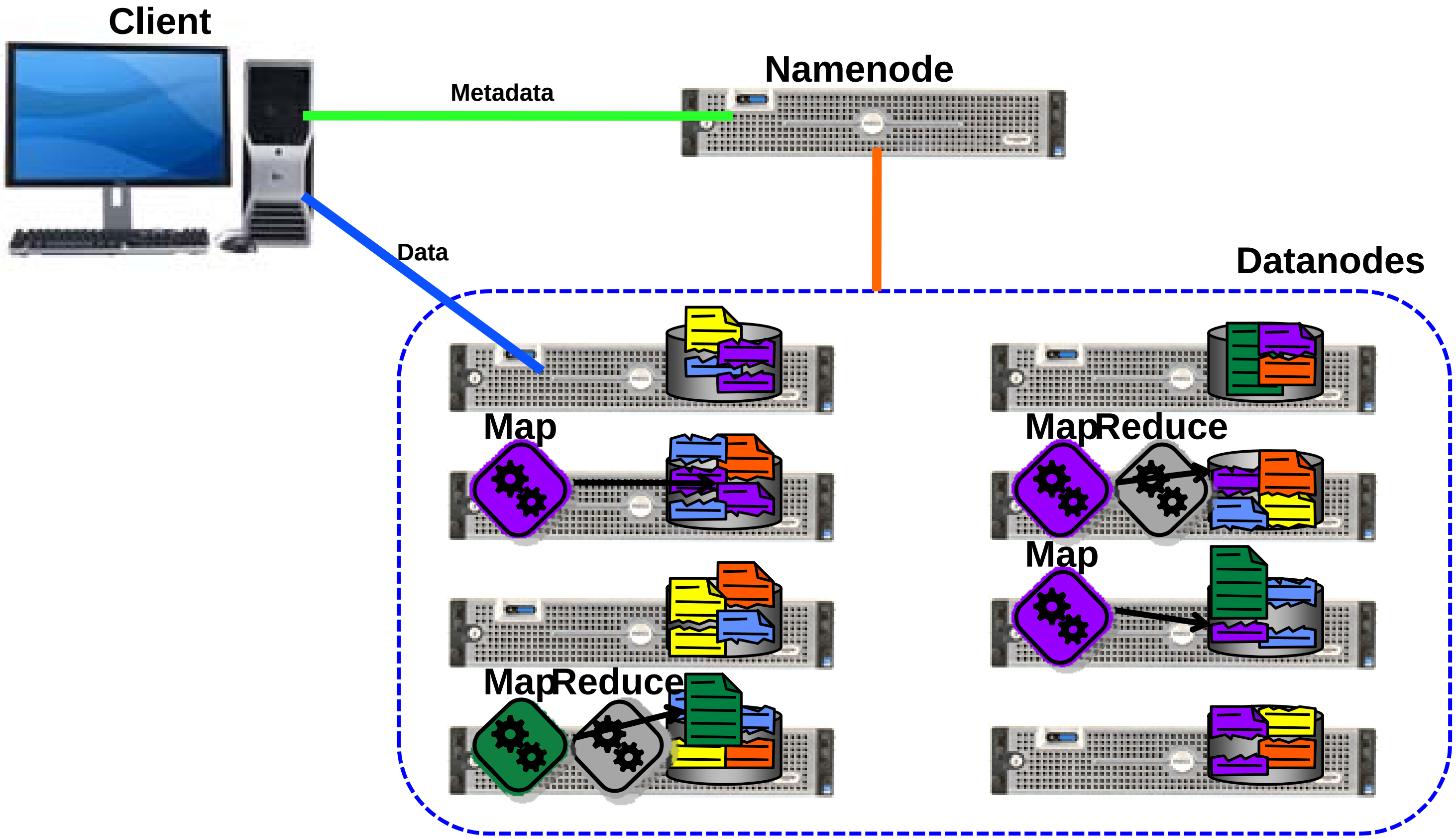
- Low-cost, block-based, “read-only”, replicating, distributed file system
- Namenode maintains metadata of distributed file system
- Good for very large files (Gigabyte)
  - Tar balls of lots of small files (e.g., html)
  - Distributed databases (e.g. HBase)

Distributed Database (e.g., HBase)



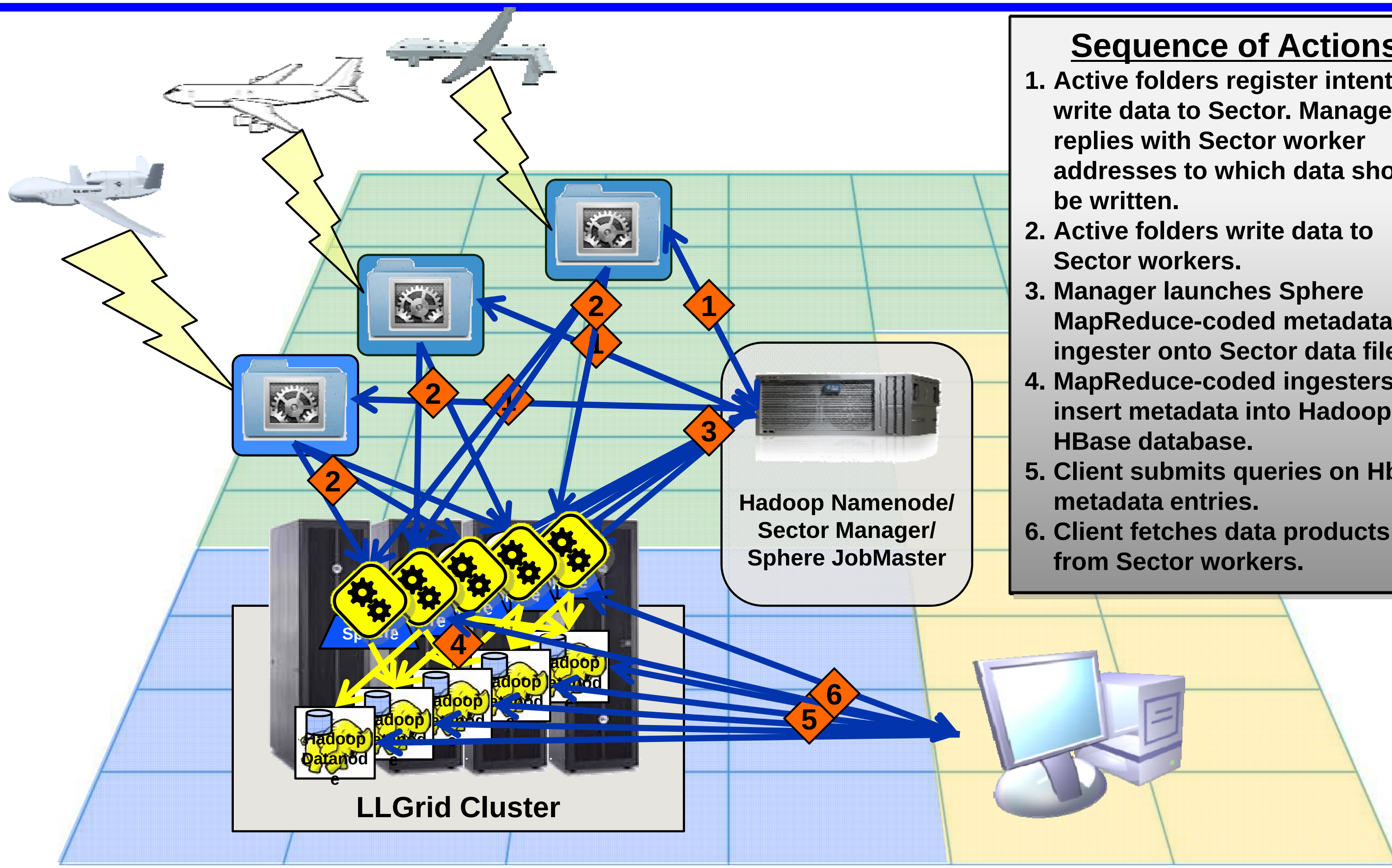
- Database tablet components spread over distributed block-based file system
- Optimized for insertions and queries
- Stores metadata harvested from sensor data (e.g., keywords, locations, file handle, …)

Distributed Execution (e.g., Hadoop MapReduce, Sphere)



- Each Map instance executes locally on a block of the specified files
- Each Reduce instance collects and combines results from Map instances
- No communication between Map instances
- All intermediate results are passed through Hadoop DFS
- Used to process ingested data (metadata extraction, etc.)

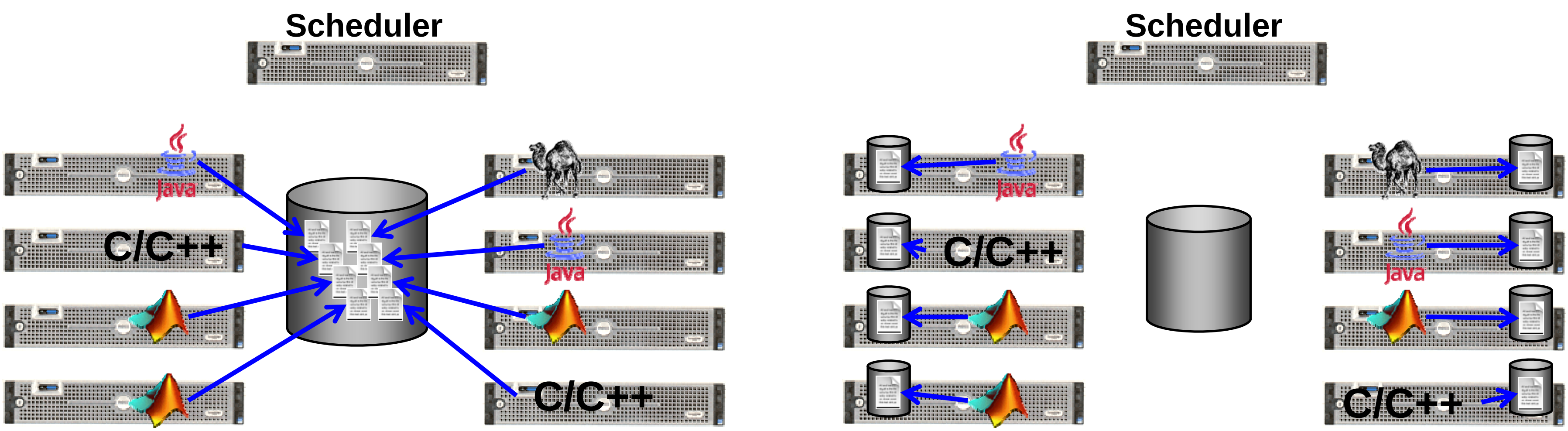
Hadoop Cloud Computing Architecture



Sequence of Actions

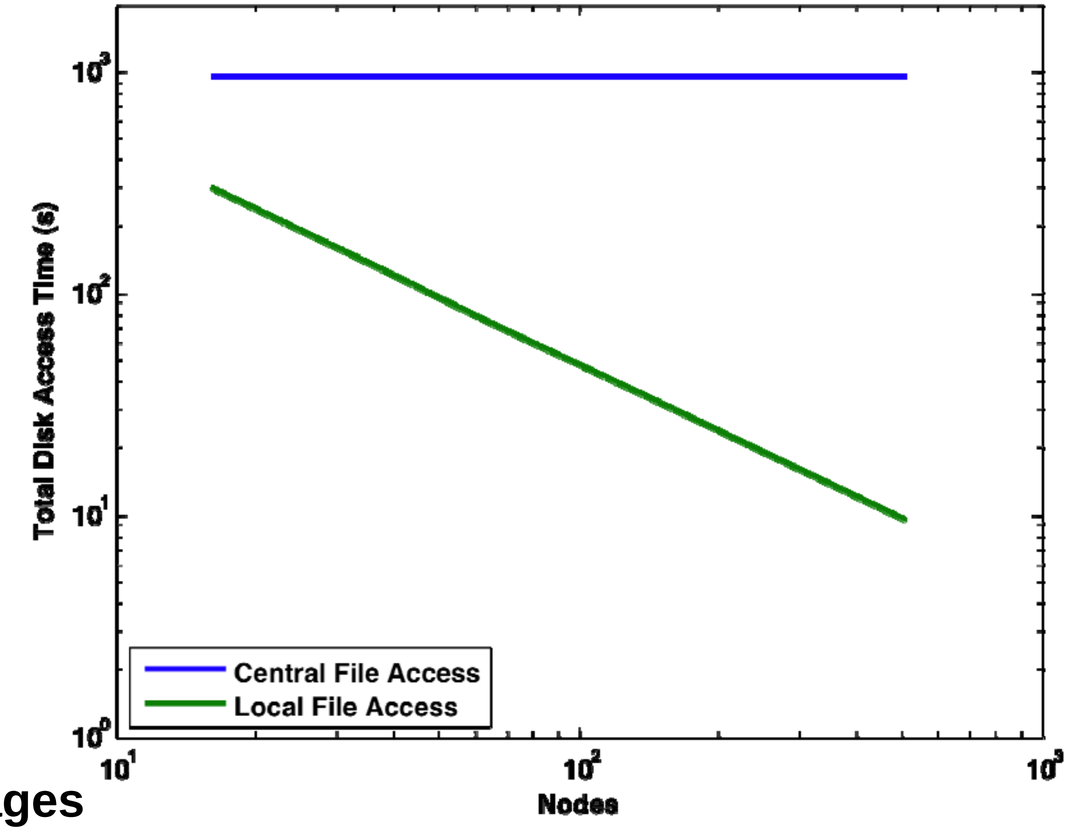
1. Active folders register intent to write data to Sector. Manager replies with Sector worker addresses to which data should be written.
2. Active folders write data to Sector workers.
3. Manager launches Sphere MapReduce-coded metadata ingester onto Sector data files.
4. MapReduce-coded ingesters insert metadata into Hadoop HBase database.
5. Client submits queries on Hbase metadata entries.
6. Client fetches data products from Sector workers.

Examples



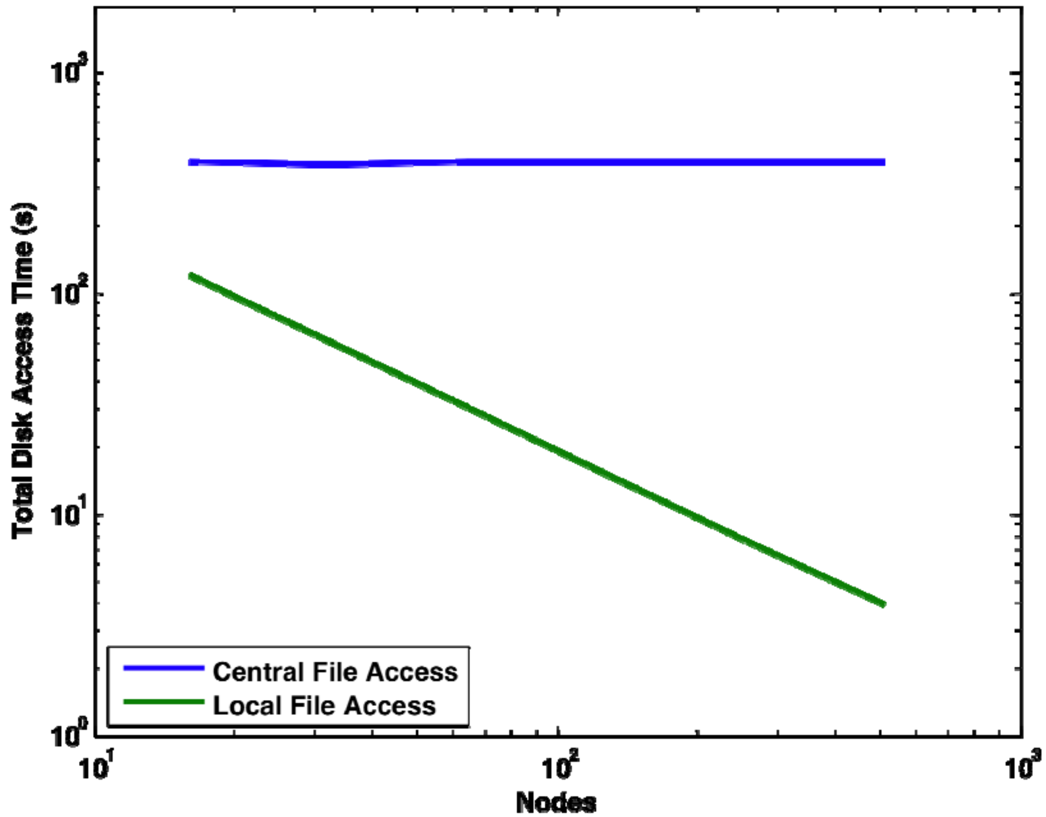
- Compare accessing data
  - Central parallel file system (500 MB/s effective bandwidth)
  - Local RAID file system (100 MB/s effective bandwidth)
- In data intensive case, each data file is stored on local disk in its entirety
- Only considering disk access time
- Assume no network bottlenecks
- Assume simple file system accesses

E/O Photo Processing App Model



- **Two stages**
  - **Determine features in each photo**
  - **Correlate features between current photo and every other photo**
- **Photo size: 4.0 MB each**
- **Feature results file size: 4.0 MB each**
- **Total photos: 30,000**

Persistent Surveillance Tracking App Model



- Each processor tracks region of ground in series of images
- Results are saved in distributed file system
- Image size: 16 MB
- Track results: 100 kB
- Number of images: 12,000

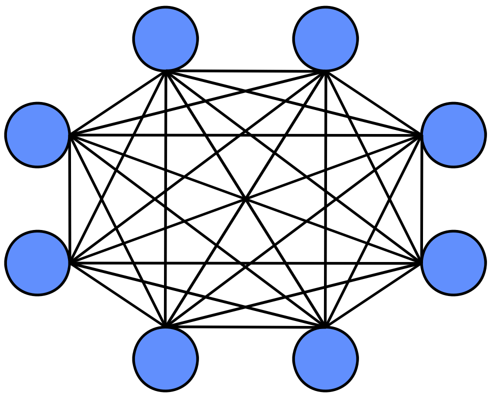
Outline

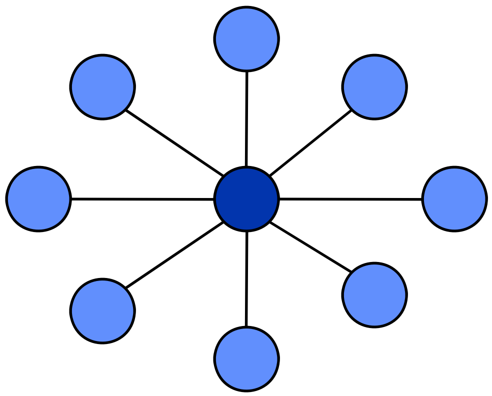
- Introduction
- Cloud Supercomputing
- **Integration with Supercomputing System**
  - ***Cloud scheduling environment***
  - ***Dynamic Distributed Dimensional Data Model (D4M)***
- Preliminary Results
- Summary

Cloud Scheduling

- Two layers of Cloud scheduling
  - Scheduling the entire Cloud environment onto compute nodes
    - Cloud environment on single node as single process
    - Cloud environment on single node as multiple processes
    - Cloud environment on multiple nodes (static node list)
    - Cloud environment instantiated through scheduler, including Torque/PBS/Maui, SGE, LSF (dynamic node list)
  - Scheduling MapReduce jobs onto nodes in Cloud environment
    - First come, first served
    - Priority scheduling
- No scheduling for non-MapReduce clients
- No scheduling of parallel jobs

Cloud vs Parallel Computing



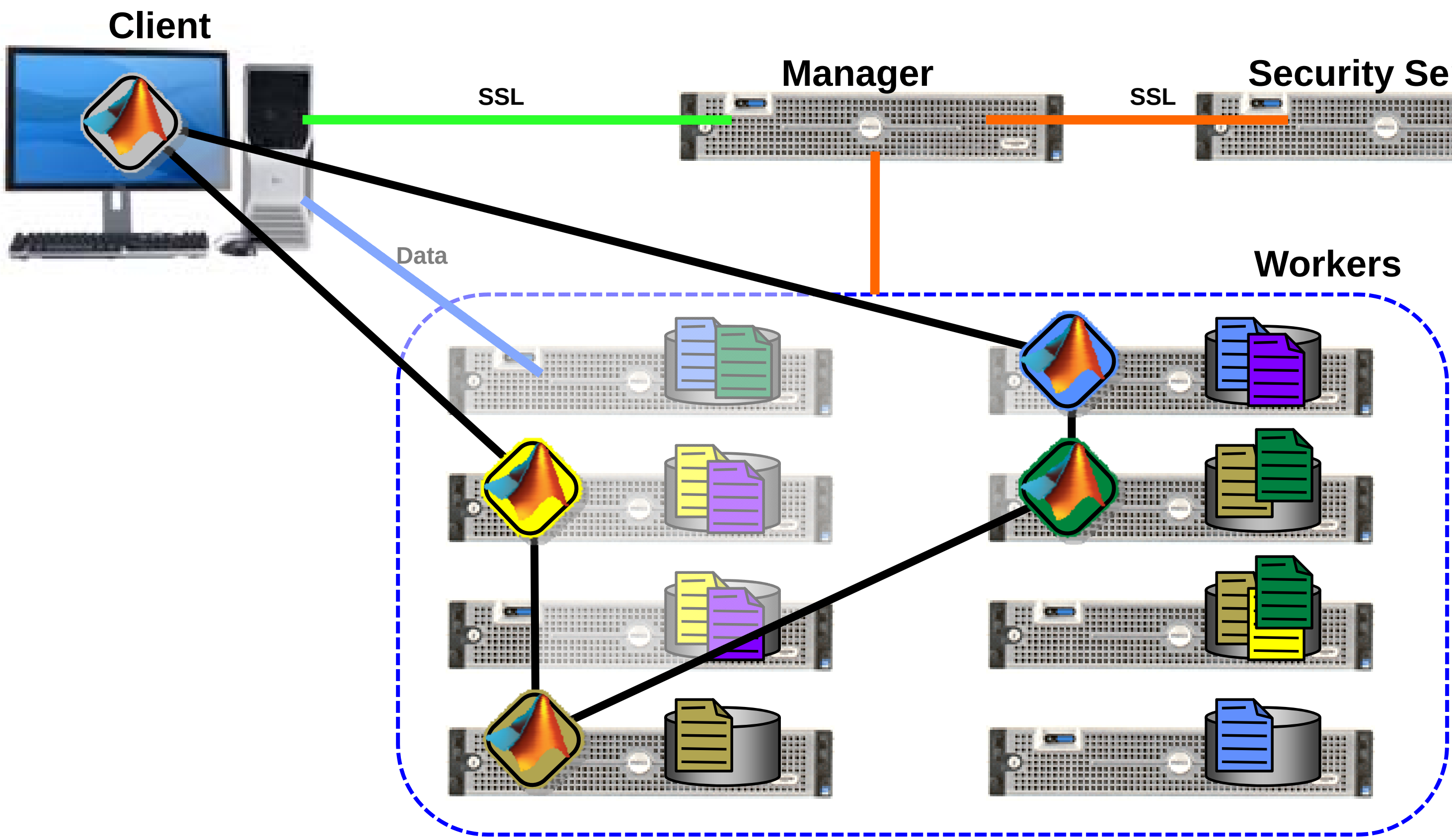


- **Parallel computing APIs assume all compute nodes are aware of each other (e.g., MPI, PGAS, …)**
- **Cloud computing API assumes a distributed computing programming model (computed nodes only know about manager)**

However, cloud infrastructure assumes parallel computing hardware (e.g., Hadoop DFS allows for direct comm between nodes for file block replication)

Challenge: how to get best of both worlds?

D4M: Parallel Computing on the Cloud

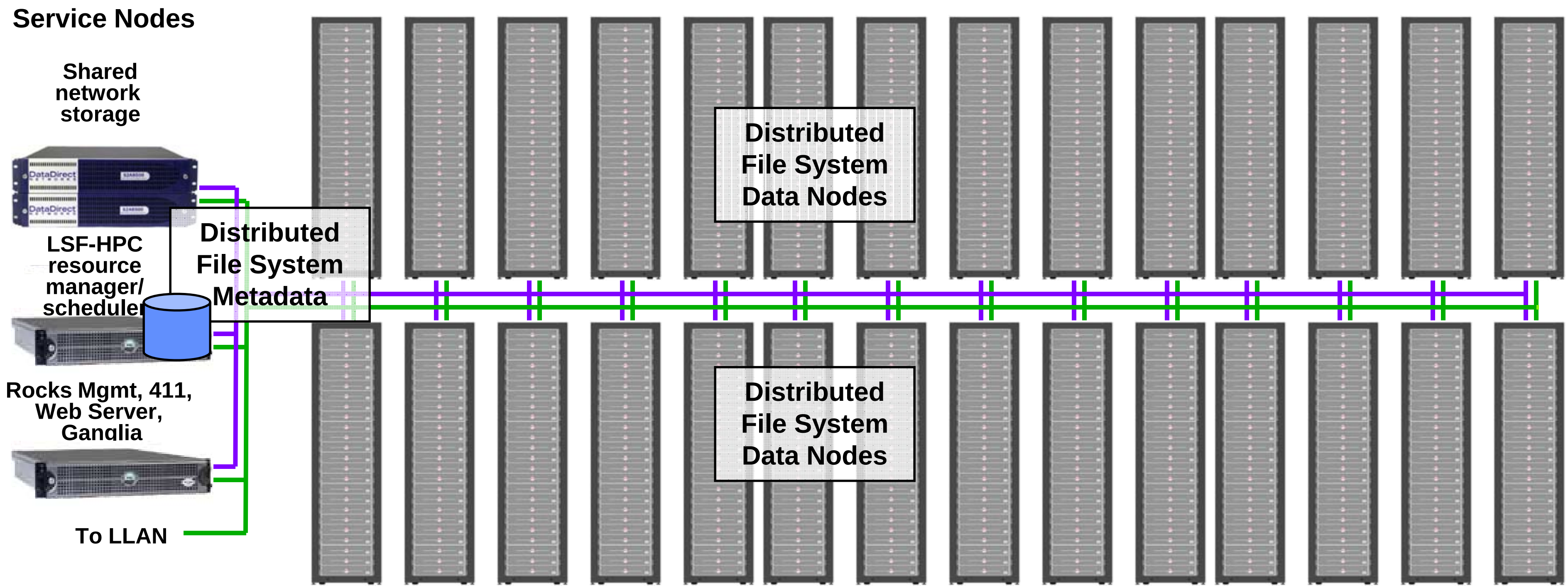


- D4M launches traditional parallel jobs (e.g., pMatlab) onto Cloud environment
- Each process of parallel job launched to process one or more documents in DFS
- Launches jobs through scheduler like LSF, PBS/Maui, SGE
- Enables more tightly-coupled analytics

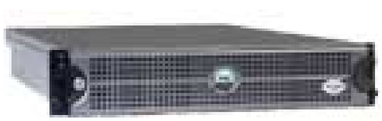
Outline

- Introduction
- Cloud Supercomputing
- Integration with Supercomputing System
- **Preliminary Results**
  - *Distributed file systems*
  - *D4M progress*
- Summary

Distributed Cloud File Systems on TX-2500 Cluster



432 DELL PowerEdge 2850

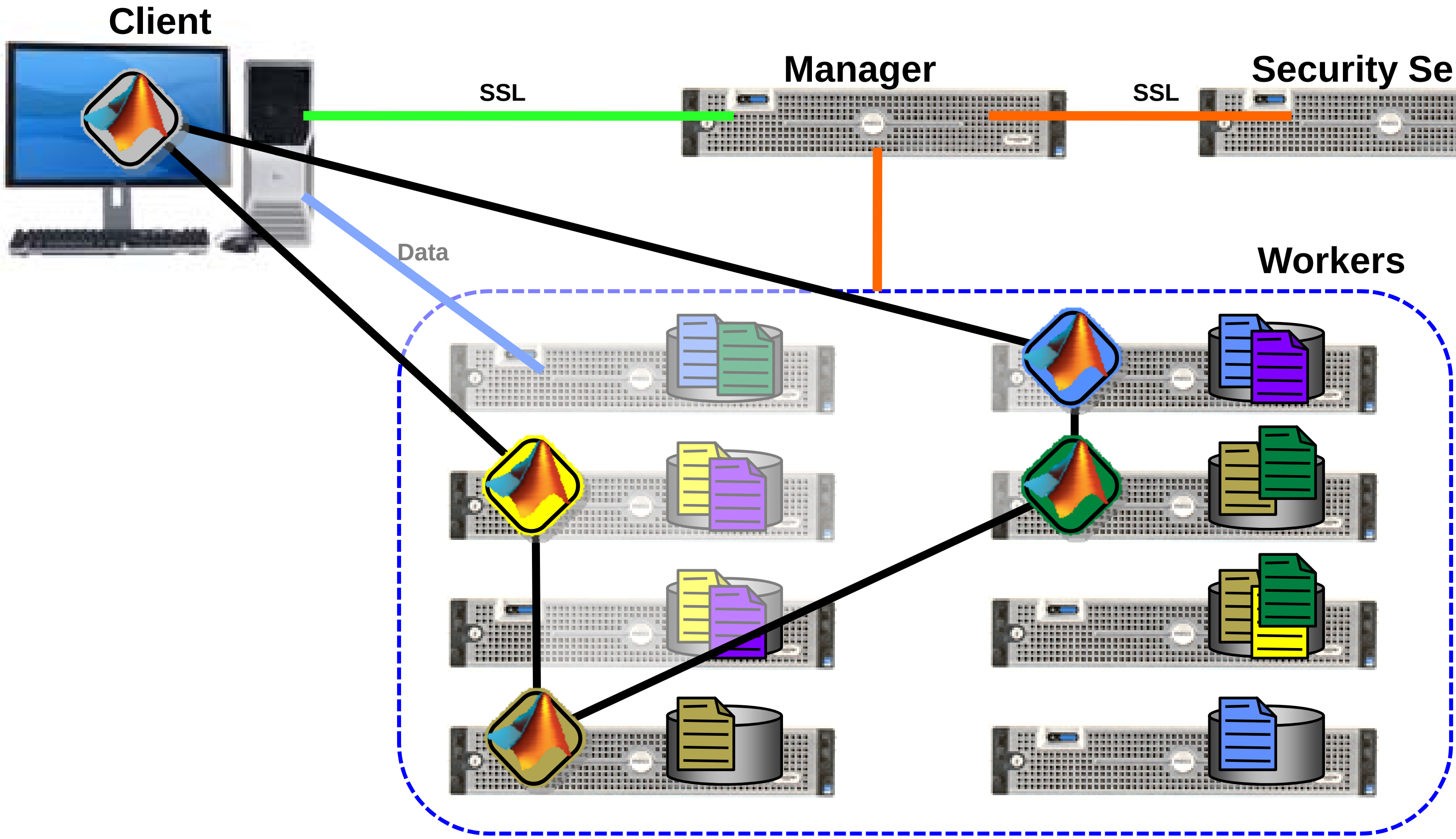


Dual 3.2 GHz EM64-T Xeon (P4)
8 GB RAM memory
Two Gig-E Intel interfaces
Infiniband interface
Six 300-GB disk drives

- **432+5 Nodes**
- **864+10 CPUs**
- **3.4 TB RAM**
- **0.78 PB of Disk**
- **28 Racks**

MIT-LL Cloud	Hadoop DFS	Sector
Number of nodes used	350	350
File system size	298.9 TB	452.7 TB
Replication factor	3	2

D4M on LLGrid



- Demonstrated D4M on Hadoop DFS
- Demonstrated D4M on Sector DFS
- D4M on HBase (in progress)

Summary

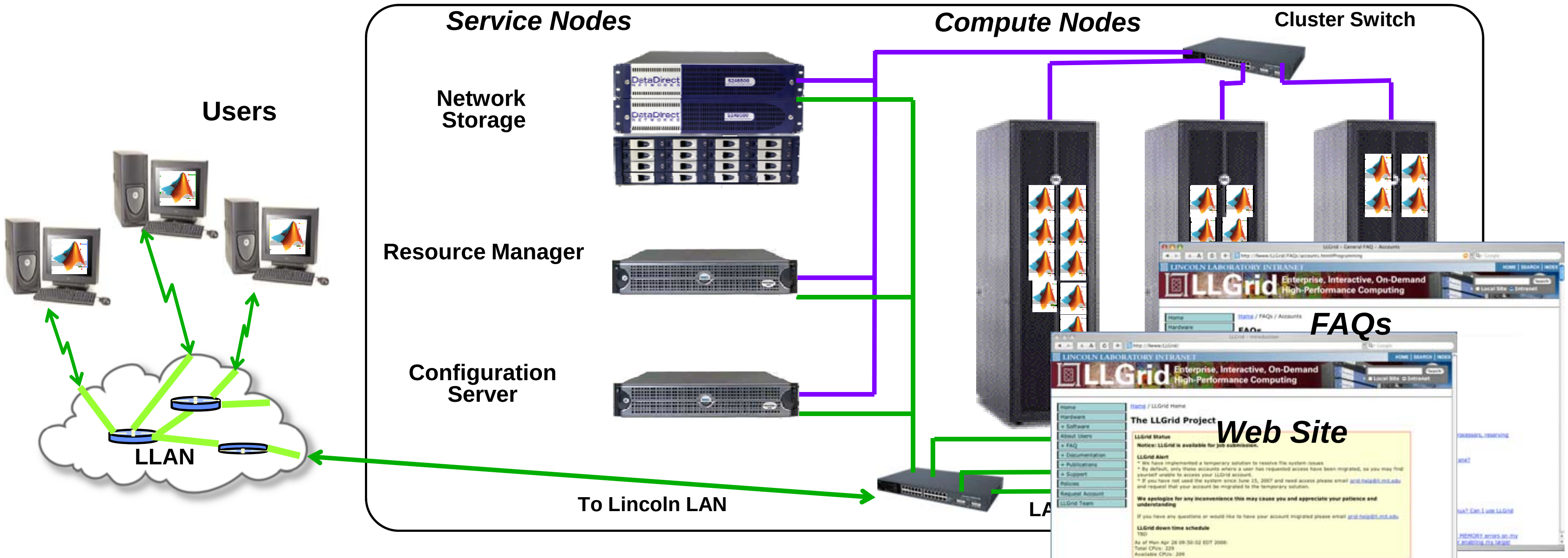
- **Persistent Surveillance applications will over-burden our current computing architectures**
  - **Very high data rates**
  - **Highly parallel, disk-intensive analytics**
- **Good candidate for Data Intensive Cloud Computing**
- **Components of Data Intensive Cloud Computing**
  - **File- and block-based distributed file systems**
  - **Distributed databases**
  - **Distributed execution**
- **Lincoln has Cloud experimentation infrastructure**
  - **Created >400 TB DFS**
  - **Developing D4M to launch traditional parallel jobs on Cloud environment**

Backups

Outline

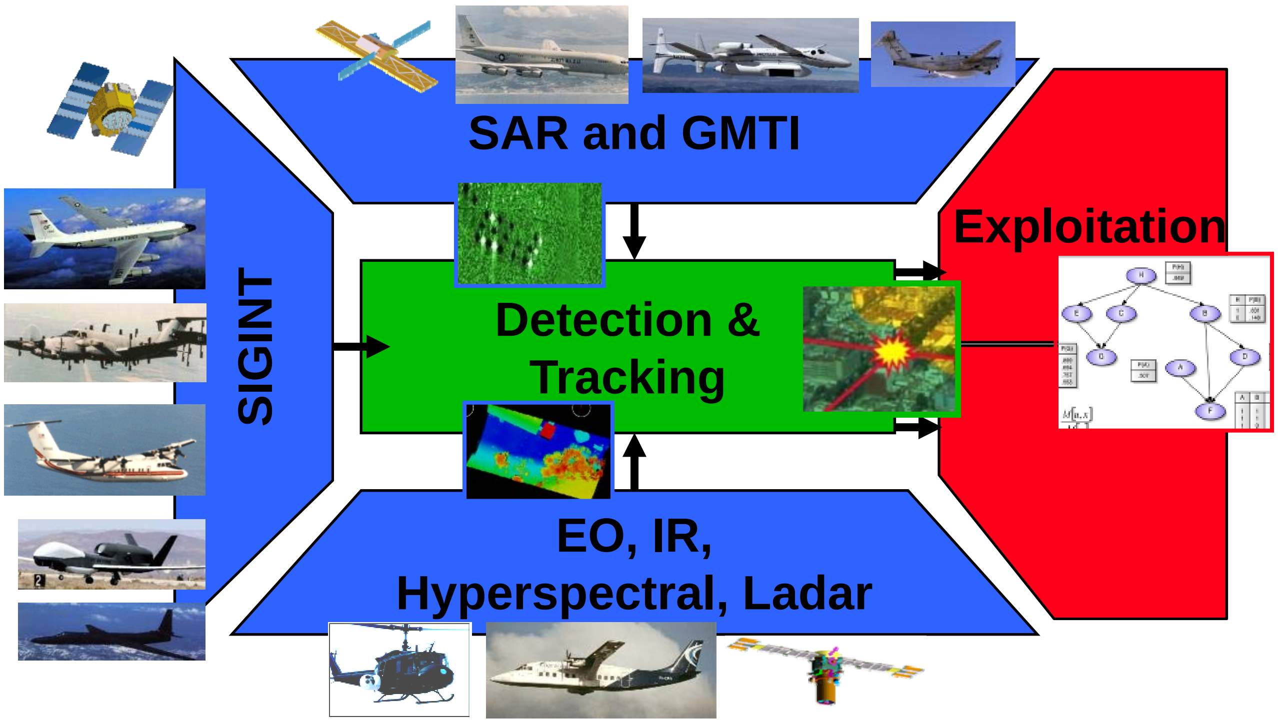
- Introduction
  - Persistent surveillance requirements
  - Data Intensive cloud computing
- Cloud Supercomputing
  - Cloud stack
  - Distributed file systems
  - Computational paradigms
  - Distributed database-like hash stores
- Integration with supercomputing system
  - Scheduling cloud environment
  - Dynamic Distributed Dimensional Data Model (D4M)
- Preliminary results
- Summary

What is LLGrid?



- **LLGrid is a ~300 user ~1700 processor system**
- **World's only desktop interactive supercomputer**
  - **Dramatically easier to use than any other supercomputer**
  - **Highest fraction of staff using (20%) supercomputing of any organization on the planet**
- **Foundation of Lincoln and MIT Campus joint vision for "Engaging Supercomputing"**

Decision Support
Diverse Computing Requirements



Algorithm prototyping
- Front end
- Back end
- Exploitation

Processor prototyping
- Embedded
- Cloud / Grid
- Graph

Stage	Signal & Image Processing / Calibration & registration	Detection & tracking	Exploitation
Algorithms	Front end signal & image processing	Back end signal & image processing	Graph analysis / data mining / knowledge extraction
Data	Sensor inputs	Dense Arrays	Graphs
Kernels	FFT, FIR, SVD, …	Kalman, MHT, …	BFS, DFS, SSSP, …
Architecture	Embedded	Cloud/Grid	Cloud/Grid/Graph
Efficiency	25% - 100%	10% - 25%	< 0.1%

Elements of Data Intensive Computing

- **Distributed File System**
  - Hadoop HDFS: Block-based data storage
  - Sector FS: File-based data storage
- **Distributed Execution**
  - Hadoop MapReduce: Independently parallel compute model
  - Sphere: MapReduce for Sector FS
  - D4M: Dynamic Distributed Dimensional Data Model
- **Lightly-Structured Data Store**
  - Hadoop HBase: Distributed (hashed) data tables