Glamour: An NFSv4-based File System Federation

Jon Haswell
Mgr NAS Systems
IBM Almaden Research Center
haswell@us.ibm.com

Based on work by Carl Burnett, Jim Myers, Manoj Naik, Steven Parkes, Renu Tewari, Andrew Tridgell

So what makes a protocol interesting ?

- Let’s look at HTTP/HTML
  - 300 Multiple Choices
  - 301 Moved permanently
  - 302 Moved temporarily
  - <A HREF=“foo.com/bar.html”>foo</A>
- The ability to have clients simply and transparently redirect between networks of servers

So what makes a protocol interesting ?

- Let's look at HTTP/HTML
  - 300 Multiple Choices

So let's go change the world

Welcome NFS V4

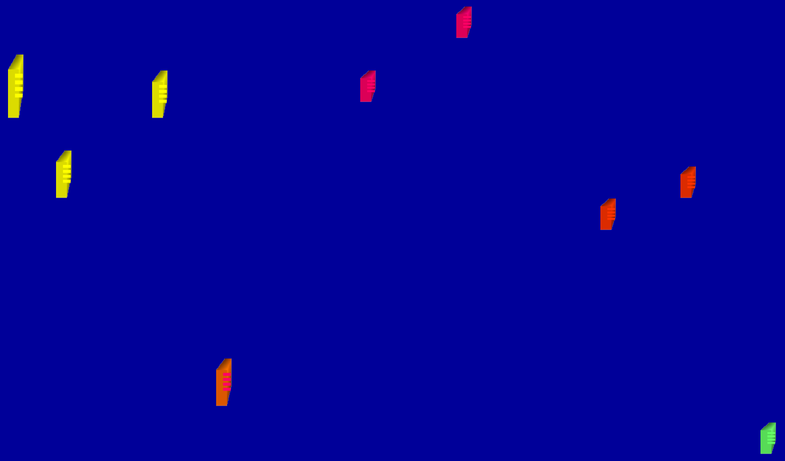
- ntly

So what should we get with NFS V4 leveraging such capabilities

- For the user/client
  - A unified enterprise wide namespace
  - Data always available with the desired performance
  - No broken links, missing data
  - Ability to work even in the presence of network partitions
- For the administrator
  - The ability to easily install and configure such a system, including existing NFS servers
  - The ability to manage such a federated system as a single system
  - The ability to add and remove servers/storage without disrupting clients
  - Automation to optimize system utilization to achieve high level business goals

Project Glamour

- A world where data replicates, is cached and migrates intelligently across networks of file servers, seamlessly, automatically and securely
- Enterprise-wide federation of islands of data
- Enables replication, migration and caching of data across geographically distributed physical file systems
- Implemented as 'middleware' for storage
  - Utilizing existing storage, filing systems and client access protocols

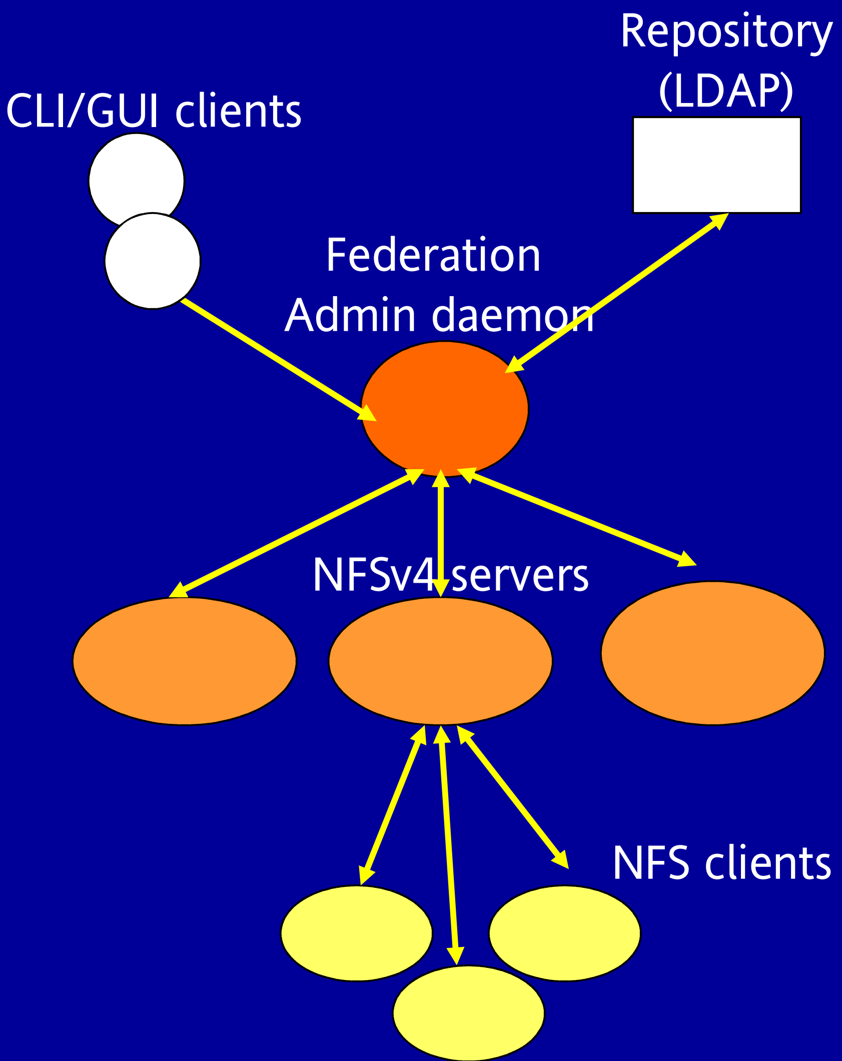


Given existing NFS V4 servers how should such a system be structured

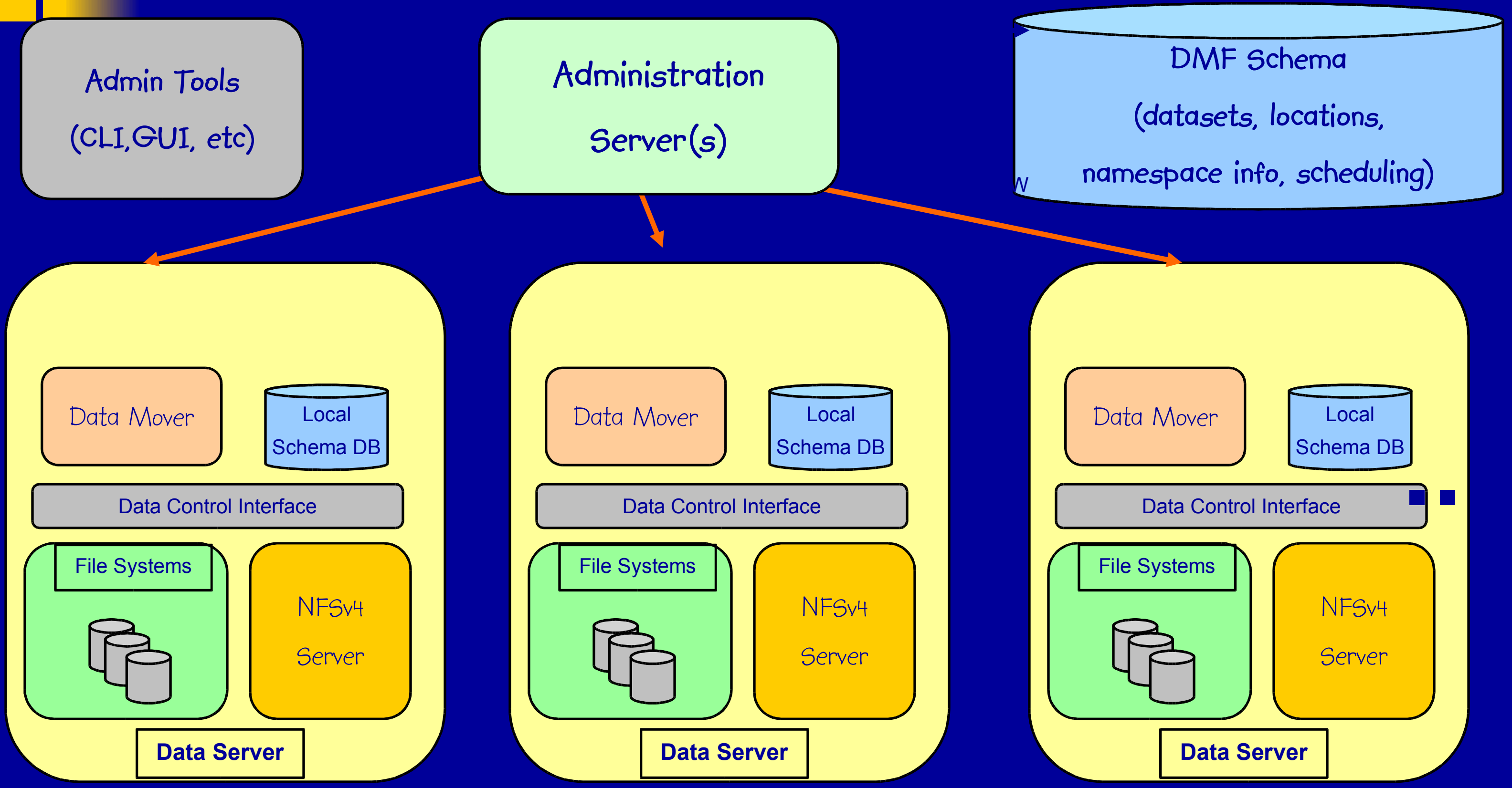
- Change as little as possible
  - Do not modify the underlying block storage or filing systems
  - Make no extensions to the NFS clients
  - Make the smallest modifications to existing NFS servers possible
- Reuse as much as possible
  - Reuse existing Kerberos and RPCGSS infrastructure
  - Reuse existing protocol where possible
- Implement as Middleware for Storage
  - Layer new functions into existing stacks
  - Provide new functionality in simple user space daemons

Glamour's Data Management Architecture

- Federation of NFS V4 servers
  - Centrally administered
  - Server to server movement of datasets
- Centralized administration
  - Can be externally administered as SMI-S style objects
- Persistent namespace and replication, migration and cache information
  - Optionally imported from a global namespace
- Delegation of responsibility
  - Designed to work with unplanned network partitions



Architecture



Admin Tools

(CLI,GUI, etc)

Administration

Server(s)

DMF Schema

(datasets, locations,

namespace info, scheduling)

Data Mover

Local Schema DB

Data Control Interface

File Systems

NFSv4 Server

Data Server

Data Mover

Local Schema DB

Data Control Interface

File Systems

NFSv4 Server

Data Server

Data Mover

Local Schema DB

Data Control Interface

File Systems

NFSv4 Server

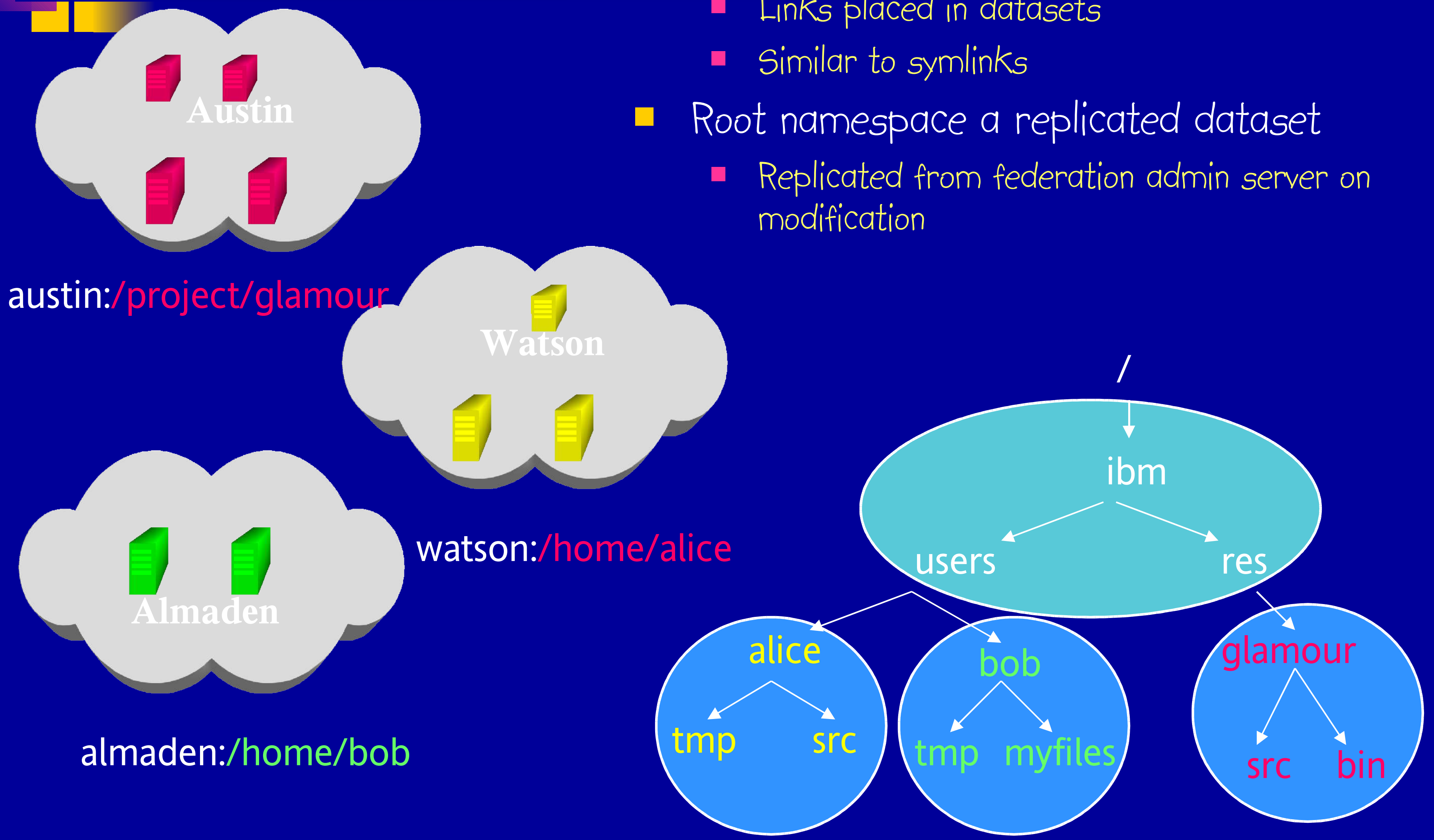
Data Server

Unit of Data Management

- Glamour provides fine grained data management
  - Existing frameworks work at LUN or FS level
    - Allocate a LUN, migrate a file system
  - Glamour works at the dataset level
    - Dataset is the basic unit of data administration
    - A directory or directory tree
    - May be a portion of a mounted filesystem instance
  - More flexible management
    - Replicate a directory
    - Migrate a directory tree
    - Cache a directory tree
    - Better load balancing

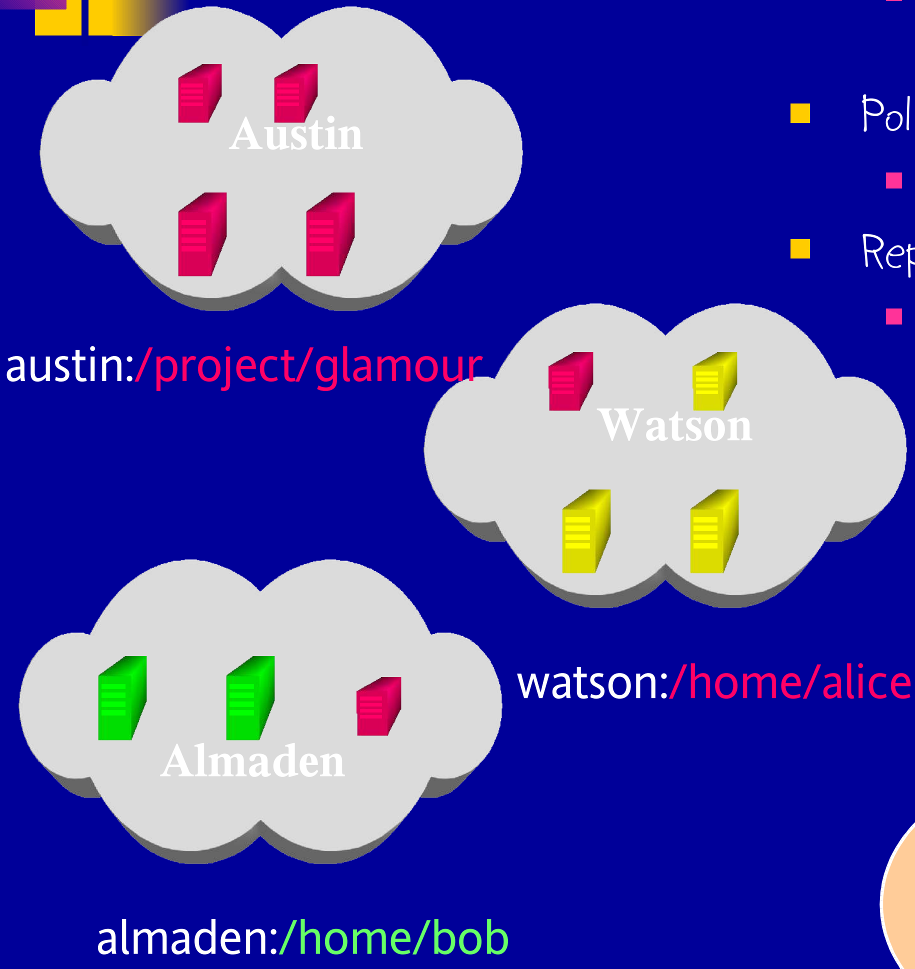
Glamour Namespace

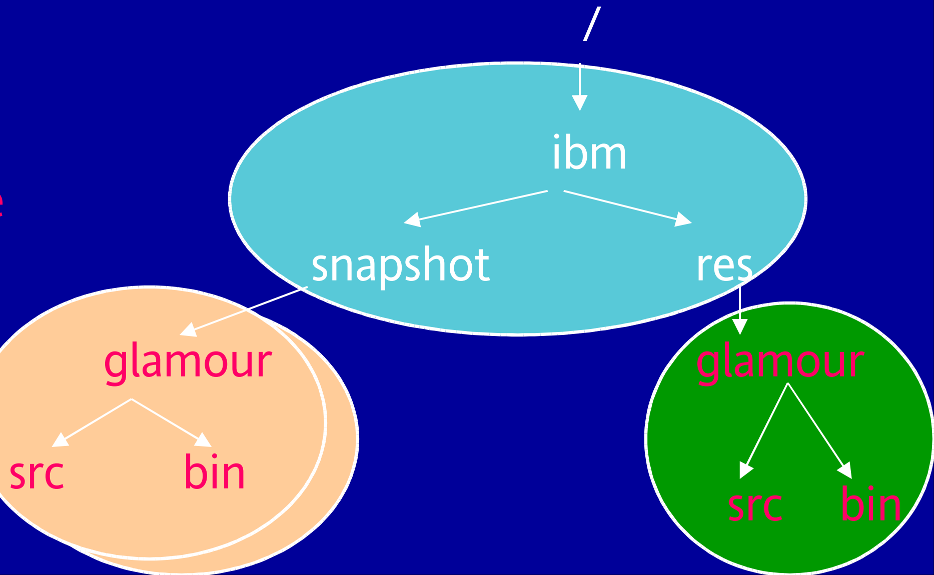
- Datasets mounted on datasets
  - Links placed in datasets
  - Similar to symlinks
- Root namespace a replicated dataset
  - Replicated from federation admin server on modification



Replication

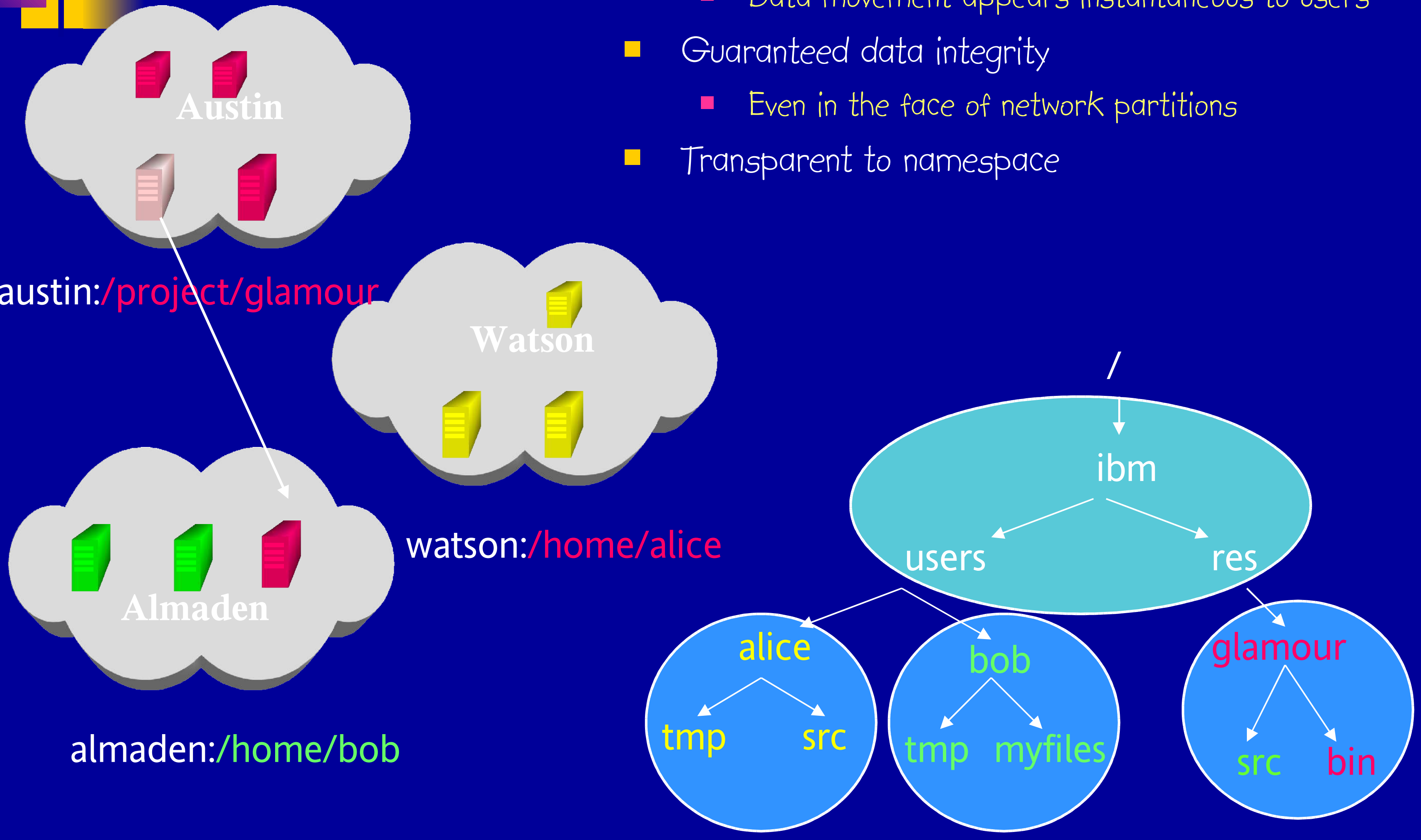
- Read-only replication across sites
  - Read-write support via clustered filesystems (e.g., GPFS)
- Policy-based consistency
  - e.g., hourly snapshots
- Replica attached in global namespace
  - In arbitrary configurations





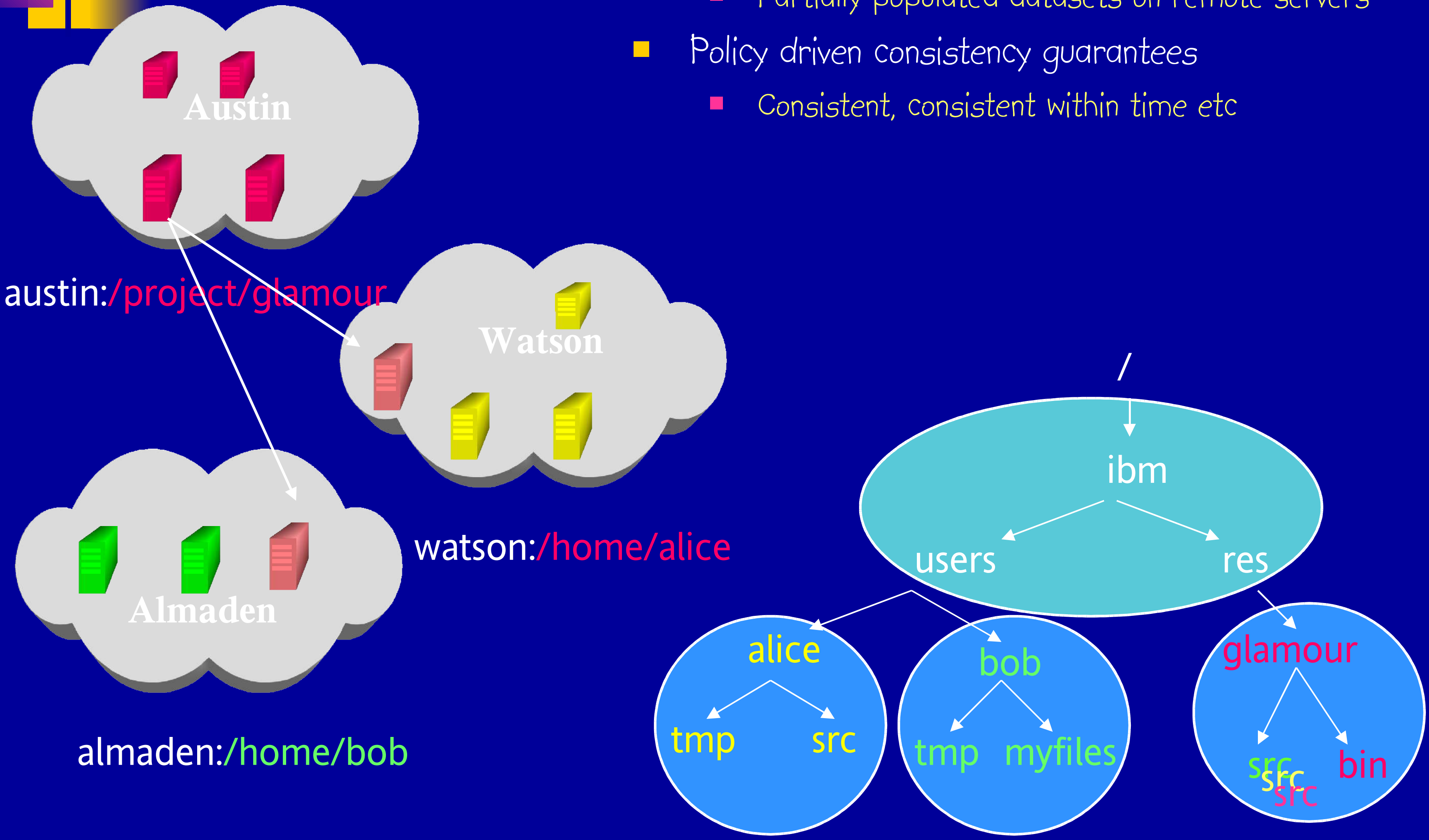
Migration

- Transparent movement of data
  - Data movement appears instantaneous to users
- Guaranteed data integrity
  - Even in the face of network partitions
- Transparent to namespace



Caching

- Persistent caching
  - Partially populated datasets on remote servers
- Policy driven consistency guarantees
  - Consistent, consistent within time etc



Data Movement

- Don't re-invent the wheel
  - Provides hooks to use existing transfer mechanisms
    - System level copy commands
    - Cluster file systems
    - Block based copy services
    - Sneaker-net
- Unless you can invent a better wheel
  - In-band transfer mechanisms
    - RPCGSS based copy
    - Advanced compression algorithms
      - Optimized for redundant block elimination
        - Regardless of namespace
        - Minimizing MIPS required

Client Steering

- A client connects to a random server
  - Starts to walk the namespace
  - Starts to cross dataset boundaries
- Servers detect
  - Client network location
  - Servers with available data
  - Servers with free bandwidth
- Client is sent subset of available locations
  - Builds upon previous workload balancing and prediction algorithms
  - Avoiding centralized single point of failure

Automated Data Placement

- Move the data to the client
  - As opposed to direct the client to the data
- System monitors workload and access patterns
  - Defines servers closer to clusters of clients
  - Monitors server workload and spare capacity
  - Based on high level policies will
    - Replicate on demand
    - Migrate on demand
    - Cache on demand
  - Based on distributed algorithms
    - No single point of failure

Status

- We currently have a working systems
  - fs__locations enabled AIX and Linux clients
  - A functioning federation administration server and management tools
  - Functioning AIX and Linux NFS server
- What we have demonstrated
  - A functioning namespace
  - Creation of datasets
  - Replication of datasets
  - High efficiency data movement protocols
  - Basic client steering
- Ongoing work
  - Advanced client steering and automated workload balancing
  - Migration and caching

What we will have achieved ?

- A storage System than
  - Is virtualized
  - Scales
  - Is secure
  - Is optimized and self-optimizing
  - Is self-managing
  - That only requires a NFS V4 infrastructure
    - No additional requirements beyond NFS V4

The future for storage

- NFS servers can be cheap and small (in addition to being large and expensive)
  - The 'cost' of the NFS functionality over an object store is negligible
  - The cost of an NFS server over a SAN based RAID controllers and adapters is small and diminishes with Moores Law
    - Consider the IBM ESS hardware also happens to be one of the worlds fastest NFS servers
      - What will be the difference in $/user IOP ?
- A federation of NFS servers can utilize existing commodity hardware and network infrastructure
  - Bandwidth is never free but this is about the most economical way to get it
- A federation of NFS servers can be flexible and provide high performance
  - Particularly when coupled to RDMA and pNFS
- Will be reliable and robust
  - Based on existing well understood security paradigms
  - Limits the 'trust' requirement placed on block access devices

The future for NFS

- NFS V4.0 Specification
  - Adequate but not ideal
    - Referral techniques need better documentation for consistency of implementations
    - Capabilities are limited
      - Controlling client steering
      - Describing consistency of file handles and state information
      - Ability to evolve filehandles on data movement
  - Incremental updates can and will improve
- Server side protocols
  - Significant value in defining open server and administration protocols
    - Always envisaged as an offshoot from V4
    - Time to re-energize this effort