

CDMI™ Conformance and Performance Testing

**David Slik
NetApp, Inc.**

Session Agenda

- A Brief Overview of CDMI
- CDMI Conformance Model
- CDMI Profiles
- CDMI Performance Model
- Suggested Performance Tests
- Example of a Performance Test
- Next Steps

Cloud Data Management Interface



Cloud Storage TWG

The CDMI standard has been developed over the last two years by leading storage vendors, users and researchers of cloud technology

This session assumes a basic understanding of CDMI concepts and terminology

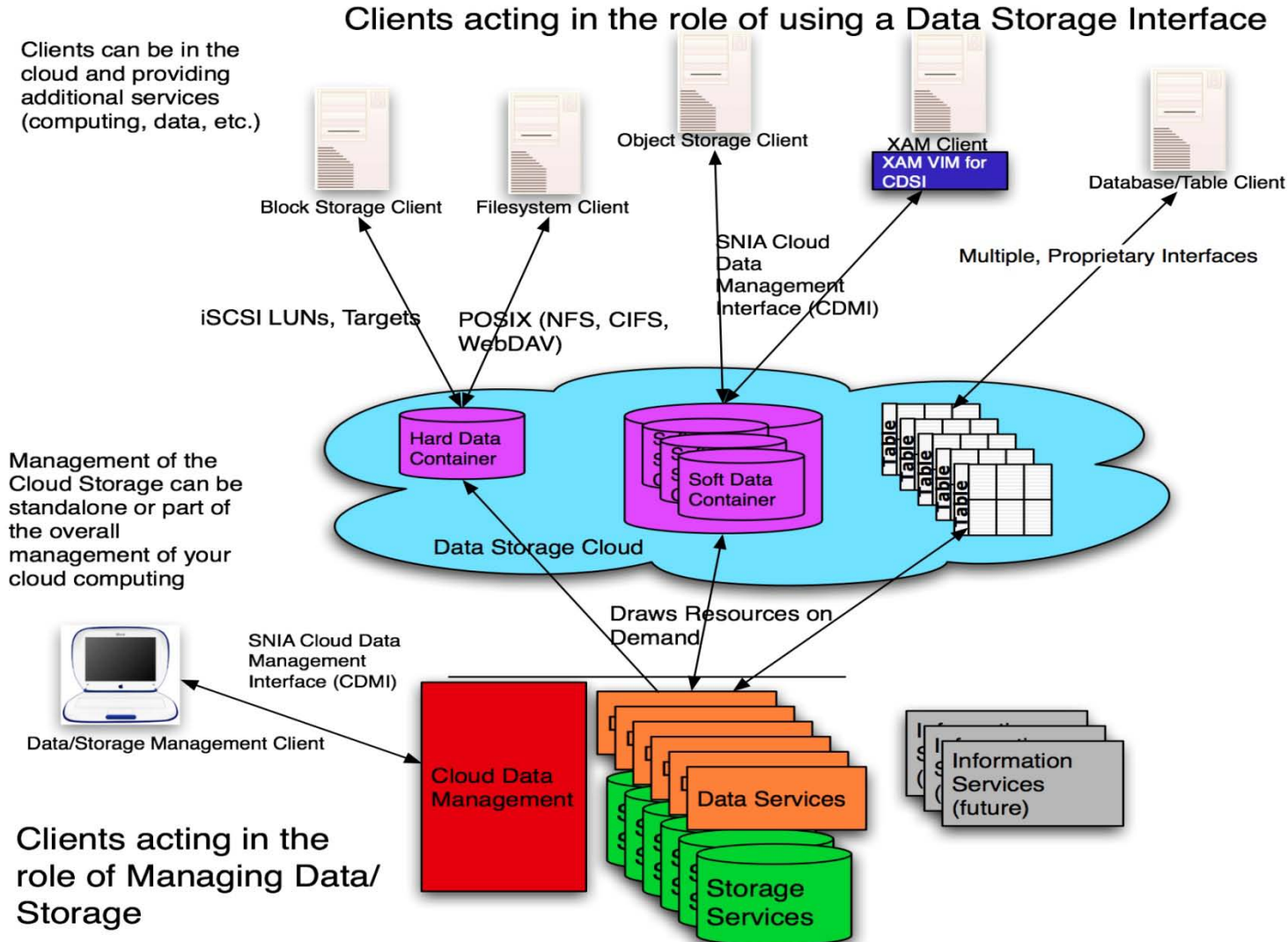
A Brief Overview of CDMI

- ❑ CDMI has the following goals:
 - ❑ To provide a standard interface for clients to communicate with storage clouds
 - ❑ To provide a standard approach for adding vendor-specific functionality without breaking client compatibility
 - ❑ To enable standardized Cloud-to-Cloud use cases

For more details on use cases, see:

http://www.snia.org/tech_activities/publicreview/CloudStorageUseCasesv0.5.pdf

A Brief Overview of CDMI



A Brief Overview of CDMI

- CDMI provides:
 - A standardized API for client interactions built on top of JSON and RESTful HTTP
 - A standardized object and metadata model for data storage and management
 - A standardized query and notifications model
 - A standardized foundation for multi-tenancy, ownership and federation

For more details on the CDMI standard, see:
<http://www.snia.org/cloud/> and <http://cdmi.sniacloud.com/>

- The CDMI Specification specifies expected behaviors for both CDMI clients and servers by stipulating valid over-the-wire exchanges
 - Client Conformance can be tested by running the client under test against a test server, or by capturing and validating over-the-wire messages
 - Server Conformance can be tested by running a test client against a server under test, or by capturing and validating over-the-wire messages

□ Example from Section 8.2.5 of CDMI 1.0.1:

Field Name	Type	Description	Requirement
mimetype	JSON String	<p>MIME type of the data contained within the value field of the data object</p> <ul style="list-style-type: none">• This field may be included when creating by value or when deserializing, serializing, copying, and moving a data object.• This field shall be stored as part of the object.• If this field is not specified, the value of "text/plain" shall be assigned as the field value.• This field shall not be included when creating a reference.	Optional

□ Distilled Test Requirements

□ If a Client specifies a JSON “mimetype” field:

- 1. The JSON field “mimetype” shall be present only once or a 400 result code will be returned
- 2. The JSON field “mimetype” shall be a string or a 400 result code will be returned
- 3. The JSON field “reference” shall not be present or a 400 result code will be returned
- 4. When the object is retrieved, the “metadata” field shall be present, and shall be the same value as specified here

□ If a Client does not specify a JSON “mimetype” field

- 5. When the object is retrieved, the “metadata” field shall be present with the value “text/plain”

□ Distilled Test Cases

- 1. For each creation style except for “reference”, create a new object without a mimetype being specified. Retrieve it and verify the mimetype
- 2. For each creation style except for “reference”, create a new object with a mimetype being specified. Retrieve it and verify the mimetype
- 3. Create a new object by “reference” and specify a mimetype. Verify that a 400 Bad Request status code is returned
- 4. Create a new object with invalid fields. Verify that a 400 Bad Request status code is returned

□ Approaches to Test Implementation

- Creation of a test client that validates the server results
 - Can verify all test requirements (#1 - #5)
 - Can perform all test cases (#1 - #4)
- Creation of a over-the-wire transaction validator
 - Can verify all test requirements (#1 - #5)
 - Test cases limited to the traffic intercepted
- Creation of a test server that validates client operations
 - Can verify only some test requirements (#1 - #3)
 - Test cases limited to client transactions received

- ❑ The SNIA is considering options around programs for conformance testing
- ❑ Tools such as WSTF may be of assistance to enable vendors to perform interoperability testing
 - ❑ <http://wstf.org/>
- ❑ Opportunities for third-party conformance testing tools

CDMI Profiles for Test Coverage

- ❑ Almost every part of the CDMI protocol is optional, so clients must discover what subsets of the protocol a given server implements in order to determine functionality and compatibility
- ❑ In CDMI, this is enabled by a mechanism called “Capabilities”
- ❑ There are over a hundred capabilities defined in CDMI 1.0.1
- ❑ Conformance testing is keyed to capabilities

- ❑ To simplify keeping track of capabilities, CDMI 1.1 proposes “Profiles”, which are well-defined sets of capabilities corresponding to typical use cases
 - ❑ For example, a “Basic Object Storage by ID” could be used to define the ability to perform CRUD operations by Object ID
 - ❑ Test coverage would be based on a combination of capabilities and profiles, with a server implementation only being considered to implement a profile if all corresponding capability tests pass

CDMI Performance Model

- ❑ CDMI (and Object Storage in general) does not have a well-defined performance benchmarking model
- ❑ Performance testing is very important for both vendors and purchasers
- ❑ Performance testing should reflect real-world workloads

Suggested Performance Tests

- ❑ Performance testing must take into account:
 - ❑ Mixture of operations
 - ❑ Concurrency of operations
 - ❑ Size of object Value
 - ❑ Size/quantity of object Metadata
 - ❑ Partial vs. Complete Object Retrieval/Update
- ❑ Performance testing must measure:
 - ❑ Latency Distribution to First and Last Byte
 - ❑ Transactions per Second
 - ❑ Bytes per Second

Suggested Performance Tests

Requires multi-dimensional testing to cover different workloads

Op/Size I Thread	0 Byte	10 KiB	1 MiB	1 GiB
Create	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Retrieve	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Update	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Delete	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency

Suggested Performance Tests

Op/Size 10 Threads	0 Byte	10 KiB	1 MiB	1 GiB
Create	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Retrieve	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Update	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Delete	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
	Latency	Latency	Latency	Latency

Suggested Performance Tests

Op/Size 100 Threads	0 Byte	10 KiB	1 MiB	1 GiB
Create	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Retrieve	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Update	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
Delete	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency	Ops/s Bytes/s Latency
		Latency	Latency	Latency
		Latency	Latency	Latency

Suggested Performance Tests

- ❑ For each performance test, the system is run in two modes:
Warm-up and Steady-State
 - ❑ This allows caching effects to be qualified and separated from sustained performance

- ❑ Each test is run for a specified period of time. Tests involving smaller objects are run for a shorter period of time, and tests involving larger objects are run for a longer period of time
 - ❑ E.G. 0 byte object tests may be run for tens of minutes, where 1 GiB object tests may run for hours

Suggested Performance Tests

- ❑ Additional test dimensions include:
 - ❑ Mixtures of operations
 - ❑ Mixtures of object sizes
 - ❑ Size and number of metadata items attached to objects
 - ❑ Concurrency scalability beyond 100 clients
 - ❑ Queue and Dequeue operations
 - ❑ Notification Queue latency and throughput
 - ❑ Query Queue latency and throughput
 - ❑ Requires standardized corpus and queries

Suggested Performance Tests

- ❑ Captured traces of operations can also provide value for performance testing. By replaying a trace on a system under test, a comparable benchmark of time to complete and client latency can be calculated.
- ❑ Methods for capturing and playing back traces should be developed
- ❑ Traces for common workloads should be developed



Ruby Performance Test Demonstration

- ❑ Vendors are invited to become involved with the SNIA Cloud Storage Technical Working Group to:
 - ❑ Contribute towards the creation of a standard conformance test suite
 - ❑ Contribute towards the definition of standardized profiles of capabilities
 - ❑ Contribute towards the creation standardized performance testing tools

Thank you!

Questions and Answers

Contact Info:
dslik@netapp.com