OpenDS
ZKI AK Verzeichnisdienste
Oct. 2009

Abdi Mohammadi
Principal Field Technologist
Sun Microsystems, Inc.

Agenda

- Introduction to OpenDS
- Features
- Replication Topology
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

Agenda

- **Introduction to OpenDS**
- Features
- Replication Topology
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

The OpenDS project

- Released in Open Source in July 2006
  - CDDL
  - Source code at https://opends.dev.java.net/
- Sponsored by Sun Microsystems
- A Java based Server supporting the LDAPv3 protocol based on Berkeley DB Java Edition
  - Not accessible from outside
- All security, access controls, password management features to safely store the information about Users

What is it for ?

- Generic object oriented data store
- White pages and Email Address Book
- Mostly the data store for Identities
  - For Authentication and Authorization
  - For profiles and personalization
- The underlying infrastructure in all Enterprises
  - Leveraged by Web and Mail infrastructure products
  - Cornerstone of Identity Management products:
    - Access Management and Federation
    - Provisioning and De-provisioning tools

OpenDS Goals

- A complete set of Directory Services
  - Directory Back-end database
  - Full LDAPv3 compliance and standard extensions
  - Multi-Master replication
  - Directory Proxy Services : load-balancing, data distribution, security services
  - Virtual Directory Capabilities
- Horizontal and Vertical Scalability
- Sun Directory Server Enterprise Edition 8.0 will be based on OpenDS code

Three Principles

- Ease of Use
  - Installation, Configuration, Management, Monitoring...
- Performance
  - Code with performance in mind:
    - Careful with memory
    - Threads and reduced contention
    - Monitors and Configurable queues
- Extensibility
  - Many interfaces defined
  - Default implementation provided

Agenda

- Introduction to OpenDS
- **Features**
- Replication Topology
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

Features

- LDAPv3 directory server fully standard compliant
  - Supports many LDAP standard and experimental extensions
  - Supports Multi Master Replication with 3 different levels of data consistency
  - Extensive security features
    - StartTLS, SASL, Password Policy, ACIs
- Virtual Attributes such as isMemberOf
- APIs to add components, plugins, controls, extensions and extended operations to the server

Features (cont.)

- DSML Gateway
- Graphical Control-Panel and Command Line Tools for configuration or monitoring
- Complete User Administration and Reference documentation
- Localized in 6 different languages
  - Community lead translation, You contribute
- Support available with Sun OpenDS Standard Edition 2.0.
- Installs in 6 clicks and less than 3 minutes

  https://www.opends.org/promoted-builds/latest/install/QuickSetup.jnlp

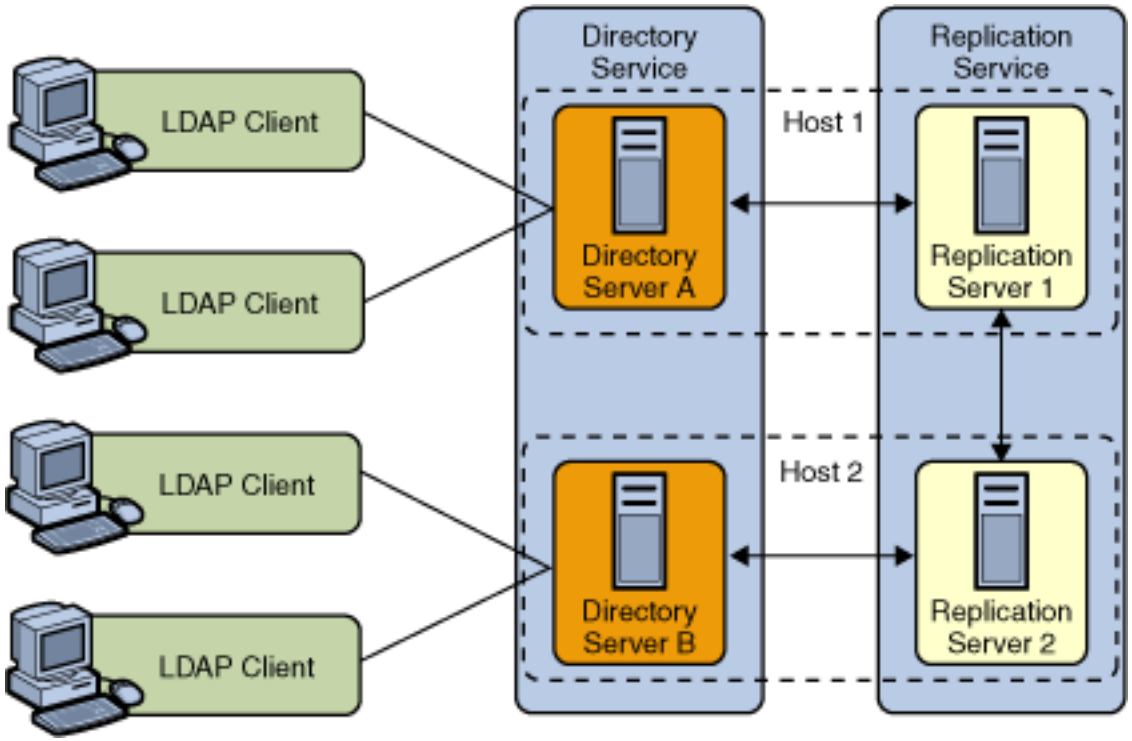
Agenda

- Introduction to OpenDS
- Features
- **Replication Topology**
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

Replication Features

- N-Way Multi Master
- Schema Replication
- Fractional Replication
- Strong Authentication
- WAN Support

Basic Replication Architecture



Replication Servers

A replication server performs the following tasks:

- Manages connections from directory servers
- Connects to other replication servers
- Listens for connections from other replication servers
- Receives changes from directory servers
- Forwards changes to directory servers and other replication servers
- Saves changes to stable storage, which includes trimming older operations

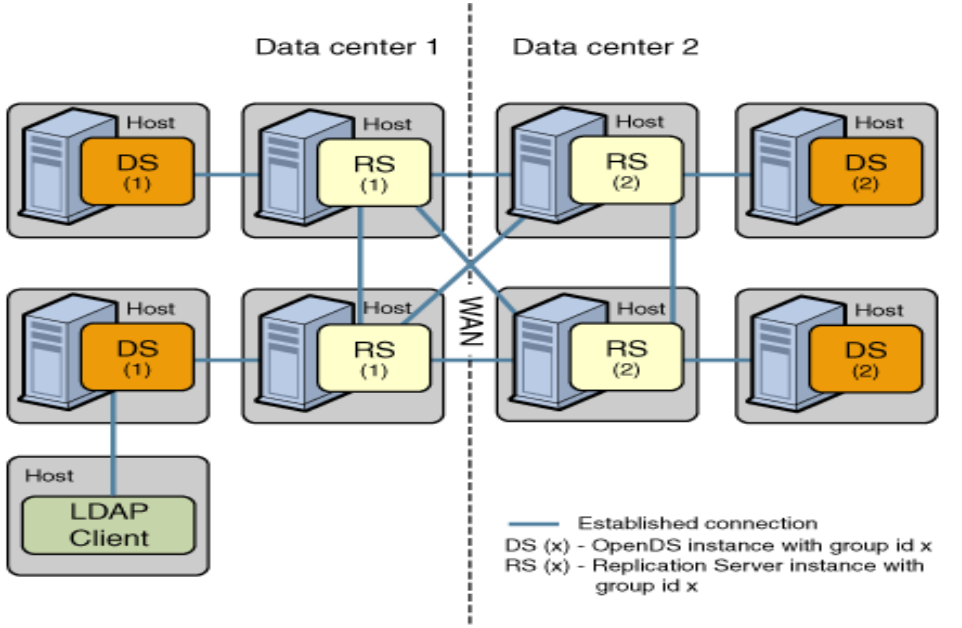
Centralized replication service

- Simpler topology
- Too many replication agreements hurt performances.
- Easier to implement assured, repair monitoring
- Easier to deploy large topologies
- All servers are writable.
- WAN Support

Replication Groups

Designed to support multi-data center deployments and disaster recovery scenarios

Group IDs determine how a directory server domain connects to an available replication server



Disaster Recovery deployment: 2 data centers with different group ids

Assured Replication

- Asynchronous Replication but wait for data to be replicated before sending ACK to application.
- Two modes :
  - Safe Data :

    Save data on Replication Servers before returning ACK to the LDAP application.
  - Safe Read :

    Propagate Change to selected Directory Servers before returning ACK to the LDAP application.
- https://opends.org/wiki/page/AssuredReplication

Agenda

- Introduction to OpenDS
- Features
- Replication Topology
- **Embedded OpenDS**
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

Embedding OpenDS in Apps

- OpenDS server runs in the same JVM as your application, standalone or web application
- OpenDS jars and files are becoming part of your application
- The applications controls when OpenDS starts and stops
- Application can access the server either
  - Using the LDAP protocol over the network
  - Or using Internal APIs
- Still can use Replication between instances

Why embedding ?

- To provide a better “Out of the Box” experience and secure access to the data
- To reduce administration and initial configration
- To setup an optimized service for your app's needs
- To reduce memory footprint by running in the same JVM
- To deploy in Web Application Container
- To test simple automatic testing of LDAP client code

Cost and Limitations

- OpenDS 2.0 Jars are less than 24 MB
- OpenDS configuration and schema is ~ 5MB
- Need to tune JVM Heap size accordingly for both application and OpenDS services
- Disable specific OpenDS services if unused
- Requires local disk and performances are mostly tied to the storage system
- Scalability: limited by the amount of memory available.

Example use

OpenSSO
Open Access. Open Federation

- OpenSSO
  - Embeds OpenDS as its configuration store
  - Replicates the configuration between instances for HA
  - Can be used as the User store (though not officially supported)
  - In a short future, will also store XACML policies for the Policy engine.
    - Goal is to scale up to 1 000 000 policies
    - Using Internal APIs for better performances

https://www.opends.org/wiki/page/UsingOpenDSAsAnEmbeddedApplication

https://www.opends.org/wiki/attach/OpenDSPresentations/OpenDS_Jazoon08.pdf

Agenda

- Introduction to OpenDS
- Features
- Replication Topology
- Embedded OpenDS
- **Performance**
  - **Scaling from Entreprise to Service Providers**
- MySQL NDB as Database Backend
- Roadmap & Integration in DSEE

Scalability

- A single server scales from few entries to several tens of millions of entries.
- No hard limit
  - Limit is based on performance requirements and Service Level agreements for specific hardware (i.e. cost)
    - How long to import
    - Response time, Throughput...
- Scale Read operations by Replicating the server
- Scale Write operations by distributing the data onto multiple servers

Performances

- LDAP servers performances are relative to:
  - The data
  - Access patterns (connections, reads, writes, ...)
  - CPUs, memory, filesystems and storage subsystem
  - Many more parameters...
- OpenDS baseline:
  - 10 Millions entries database (avg 1.5K/entry)
  - Can achieve up to 100K Search/Sec, 12K MODs /Sec in real deployment scenario
  - Response time is below 1ms for both reads and writes.

Performance (cont.)

- We managed to get more than 100 000 searches / seconds on a single box with 10 million entries
  - Intel Core i7 processors (8 core)
  - Machine over-clocked
  - Real numbers at : http://blogs.sun.com/ds/entry/opends_nehalem_benchmark_sunblade_6270
- Throughput is only one measure:
  - What is the average response time ?
  - What is the maximum response time ?

Performance (cont.)

- **The Context**
  - Sun X4150
  - 8 x Intel 3.2GHz
  - 64GB RAM
  - Internal disk
  - 10M 1.5K entries
  - Fully preloaded
- **Search rate**
  - 8 clients / CPU 35% idle
  - 15500 op/s
  - 10% = 0.193417
  - 50% = 0.223053
  - 90% = 0.278756
  - 99% = 0.362329
  - 99.9% = 0.422575
  - 99.99% = 35.5056
  - 99.999% = 41.8817
  - **Average = 0.237412 ms**
- **Modify rate**
  - 2 clients / CPU 75% idle
  - 4000 op/s
  - 10% = 0.237901
  - 50% = 0.288164
  - 90% = 0.36565
  - 99% = 0.486679
  - 99.9% = 0.706433
  - 99.99% = 11.1529
  - 99.999% = 65.5304
  - **Average = 0.303045 ms**

Who Needs this ?

- Telecom operators have used Directory services for Business side for years (Portal…)
  - Storing identities of their Customers and phones
  - Largest known deployment (with Sun Directory Server) is 120 M entries
- OpenDS for Naming Services
  - Solaris and Linux

    http://developers.sun.com/identity/reference/techart/opends-namesvcs.html
  - SAMBA https://www.opends.org/wiki/page/SambaCIFSServer
  - Kerberos https://www.opends.org/wiki/page/ConfiguringSASLGSSAPIAuthentication

Agenda

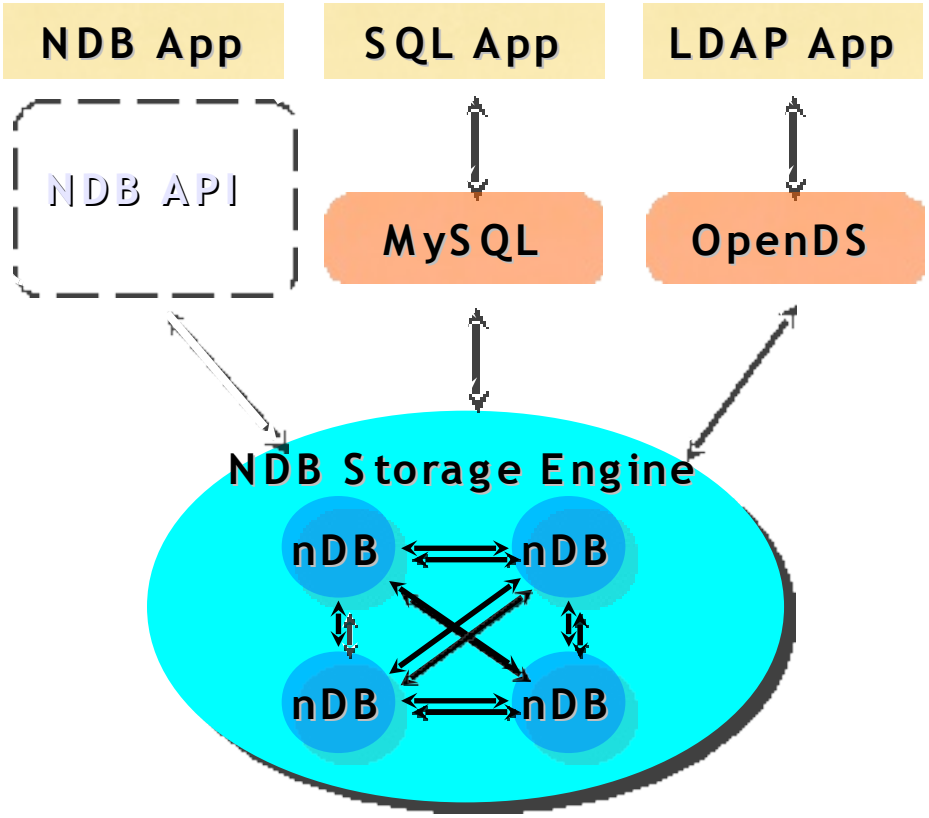
- Introduction to OpenDS
- Features
- Replication Topology
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- **MySQL NDB as Database Backend**
- Roadmap & Integration in DSEE

MySQL Cluster

- It's the Carrier Grade edition of MySQL database product
  - Provides 99,999% availability
  - Runs on commodity hardware
  - Performances are constant
- Uses Network DB (NDB):
  - Fully transactional
  - Partition the data across multiple nodes
  - Memory Based for Performances
  - Provides automatic replication and fail-over

OpenDS and MySQL Cluster

- Simultaneous access to the Data via LDAP, SQL or direct API
- Direct access to NDB
- Need LDAP Meta Data to be in NDB
- Implemented in OpenDS as an optional back-end



OpenDS NDB Back-end

- Uses NDB/J bindings to NDB API (JNI)
- Allows multiple OpenDS process to access the same NDB database concurrently
- Maps the SQL Tables schema to LDAP Schema
- Requires additional tables in NBD
  - For DNs, Objectclasses and other LDAP specific data

OpenDS NDB Back-end (cont.)

- OpenDS/NDB is feature-complete
- Finalizing perf and stress tests
- Code is available in OpenDS Trunk
- How To : https://www.opends.org/wiki/page/EnableNDBBackend

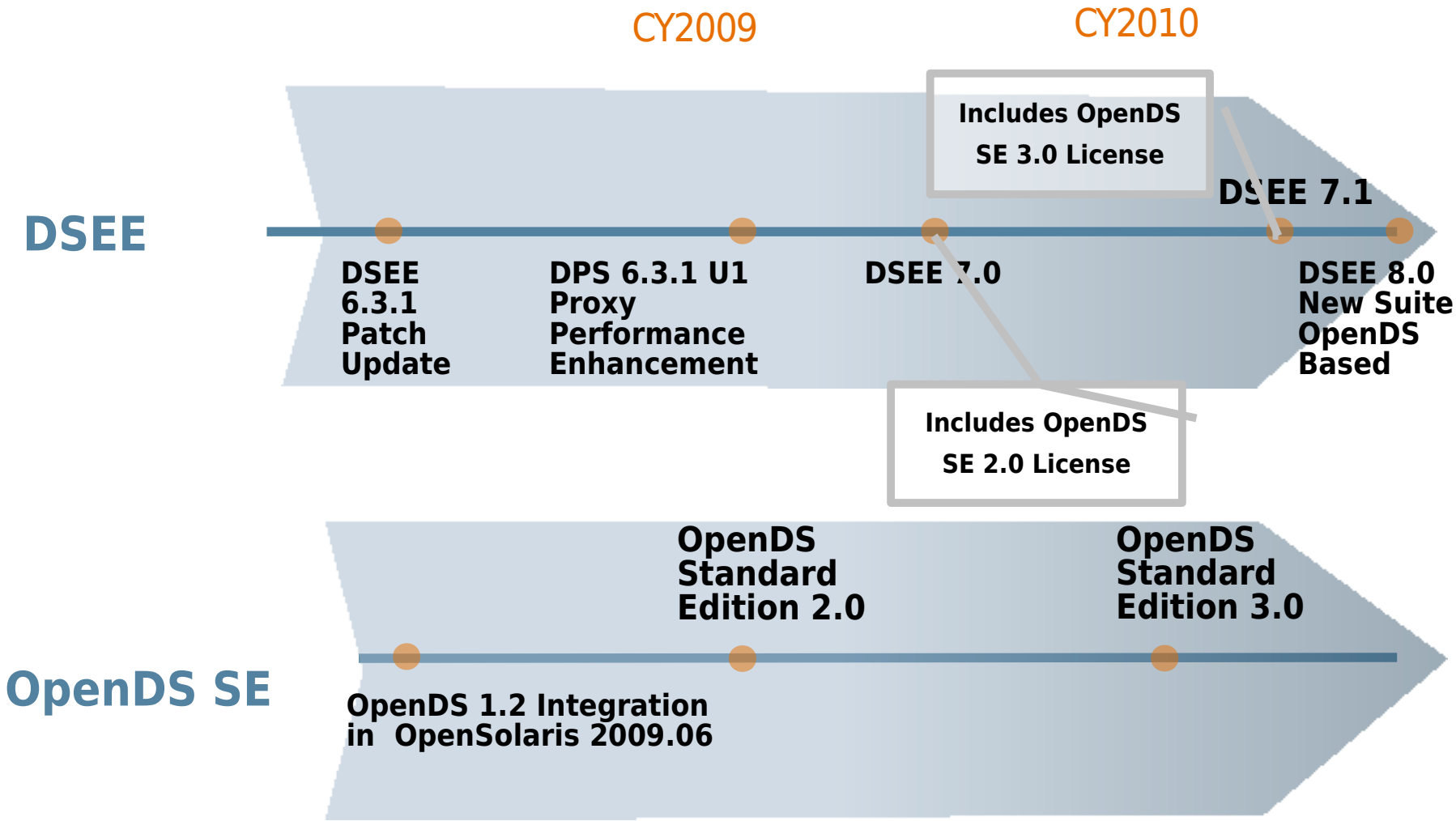
Benefits

- Lower cost
  - Runs on commodity hardware
  - Single database for multiple LDAP and SQL servers
- Performance scales linearly
  - Add more database nodes to scale capacity
  - Add more OpenDS server to scale throughput
- Response time is higher than a single node, but that's not the point

Agenda

- Introduction to OpenDS
- Features
- Replication Topology
- Embedded OpenDS
- Performance
  - Scaling from Entreprises to Service Providers
- MySQL NDB as Database Backend
- **Roadmap & Integration in DSEE**

Directory Services Roadmap



*** = All future dates subject to change, all calendar year**

Next Steps...

- Give OpenDS a try:
  - http://www.opends.org
- Participate in the community
  - Join/Login on Java.net
  - Request a Role in the OpenDS project
  - http://opends.dev.java.net
  - IRC: #opends on freenode.net
- Help with Translation to your preferred language

Thank You !

OpenDS
ZKI AK Verzeichnisdienste
Oct. 2009

Abdi.Mohammadi@sun.com