Java Track

Merits of Directories

Alex Karasulu

What's your experience with Directories?

- Have you used a directory before?
- Which directories have you used?
- What do you want to gain from this session?

ØREDEV

What is a Directory?

- data store
- heirarchical namespace
- access model based on entries with searchable attributes

- It's an indivisible cornerstone technology!

ØREDEV

Directory Server Products

- X.500
  - ISODE M-Vault
  - Nexor Directory
- LDAP
  - ApacheDS
  - OpenLDAP
  - SUN DS
  - Novel eDirectory
  - CA eTrust
  - Active Directory
  - Oracle Internet Directory

ØREDEV

Pros

- Fast read optimized store
- Easily replicated
- Centralized service with distributed storage
- Highly available, fault tolerant and load balanced
- Interoperable and based on standards
  - IETF
  - ITU
- Federated system

ØREDEV

Cons

- Slow writes or updates
- Not good at storing large blobs
- Hard to grasp (not intuitive) nor easy to design with
- Non-transactional
- ǂ Missing rich integration tier constructs
  - Triggers
  - Views
  - Stored Procedures

LDAP verses X.500

- In the beginning there was X.500
- LDAP created as a simple X.500 gateway protocol
- LDAP lacks features beyond simple access model
- LDAP adoption increases
  - easier to build clients
  - TCP/IP prevales over OSI
- LDAP starts to store entries
- X.500 adoption decreases
- X.500 adopts TCP/IP but it's too late
- LDAP borrows from X.500 administrative model

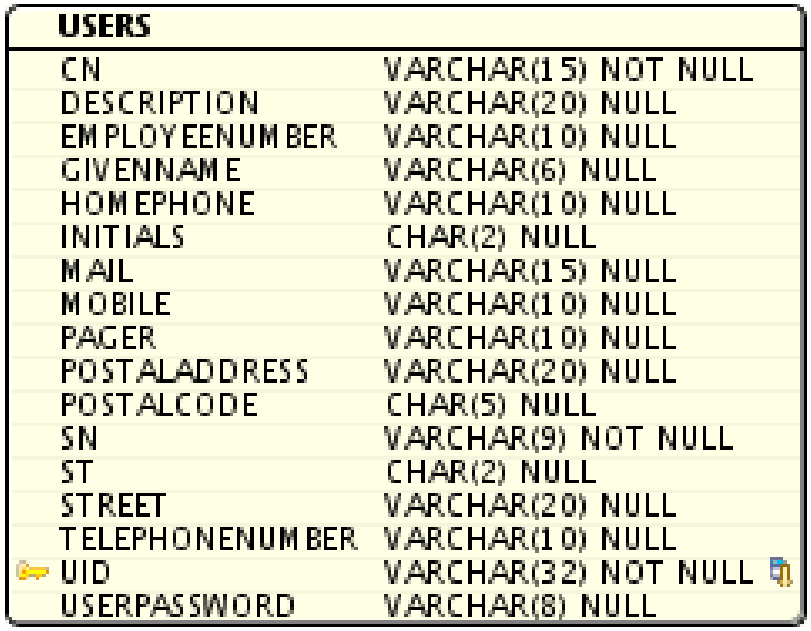
RDBMS verses LDAP

- # of clients (hundreds verses thousands)
- Connections costs and simplicity
  - Rapid connect read disconnect
  - Sporadic read activity verses steady writes
  - Connection State/Transaction overhead
- LDAP is fast
  - binary network access protocol
- RDBMS network layers use incompatible protocols
- Data location independent
- Degree of indexing
- Both can be normalized

Let's do an experiment!

- Create a user database (Apache Derby)
- Create a user directory (Apache Directory Server)
- Populate it with 10K records
- Perform lookups
  - from multiple computers
  - with many clients
  - with many threads per client
- We'll use the SLAMD load generator

A Simple User Database

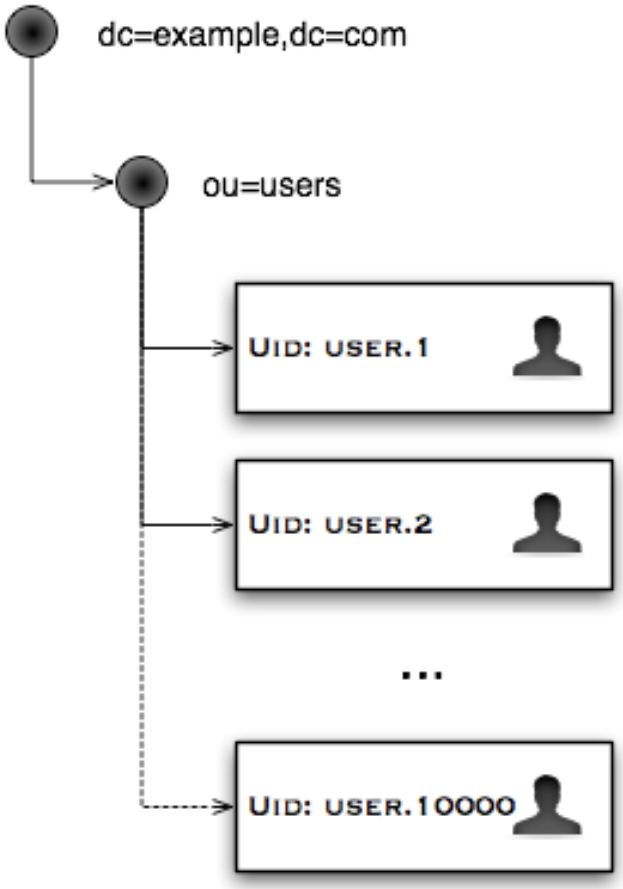


USERS	
CN	VARCHAR(15) NOT NULL
DESCRIPTION	VARCHAR(20) NULL
EMPLOYEENUMBER	VARCHAR(10) NULL
GIVENNAME	VARCHAR(6) NULL
HOMEPHONE	VARCHAR(10) NULL
INITIALS	CHAR(2) NULL
MAIL	VARCHAR(15) NULL
MOBILE	VARCHAR(10) NULL
PAGER	VARCHAR(10) NULL
POSTALADDRESS	VARCHAR(20) NULL
POSTALCODE	CHAR(5) NULL
SN	VARCHAR(9) NOT NULL
ST	CHAR(2) NULL
STREET	VARCHAR(20) NULL
TELEPHONENUMBER	VARCHAR(10) NULL
UID	VARCHAR(32) NOT NULL
USERPASSWORD	VARCHAR(8) NULL

- Simple
- Single table
- inetOrgPerson
- Unique key (uid)
- CN NOT NULL
- SN NOT NULL

ØREDEV

A Simple User Directory



- Suffix: dc=example,dc=com
- ou=User container
- inetOrgPerson entries for users

ØREDEV

Hardware Configuration

- 4 client machines (load injectors)
  - curie: Dual Intel 800 Mhz w/ 768 Mb Memory
  - hertz: Intel 2.4 Ghz w/ 1 Gb Memory
  - dirac: Athlon 2400 w/ 1 Gb Memory
  - tesla: Ultra Sparc 5 400Mhz w/ 512 Mb Memory

- 1 server machine
  - pauli: Dual Athlon 1900 MP w/ 2Gb Memory

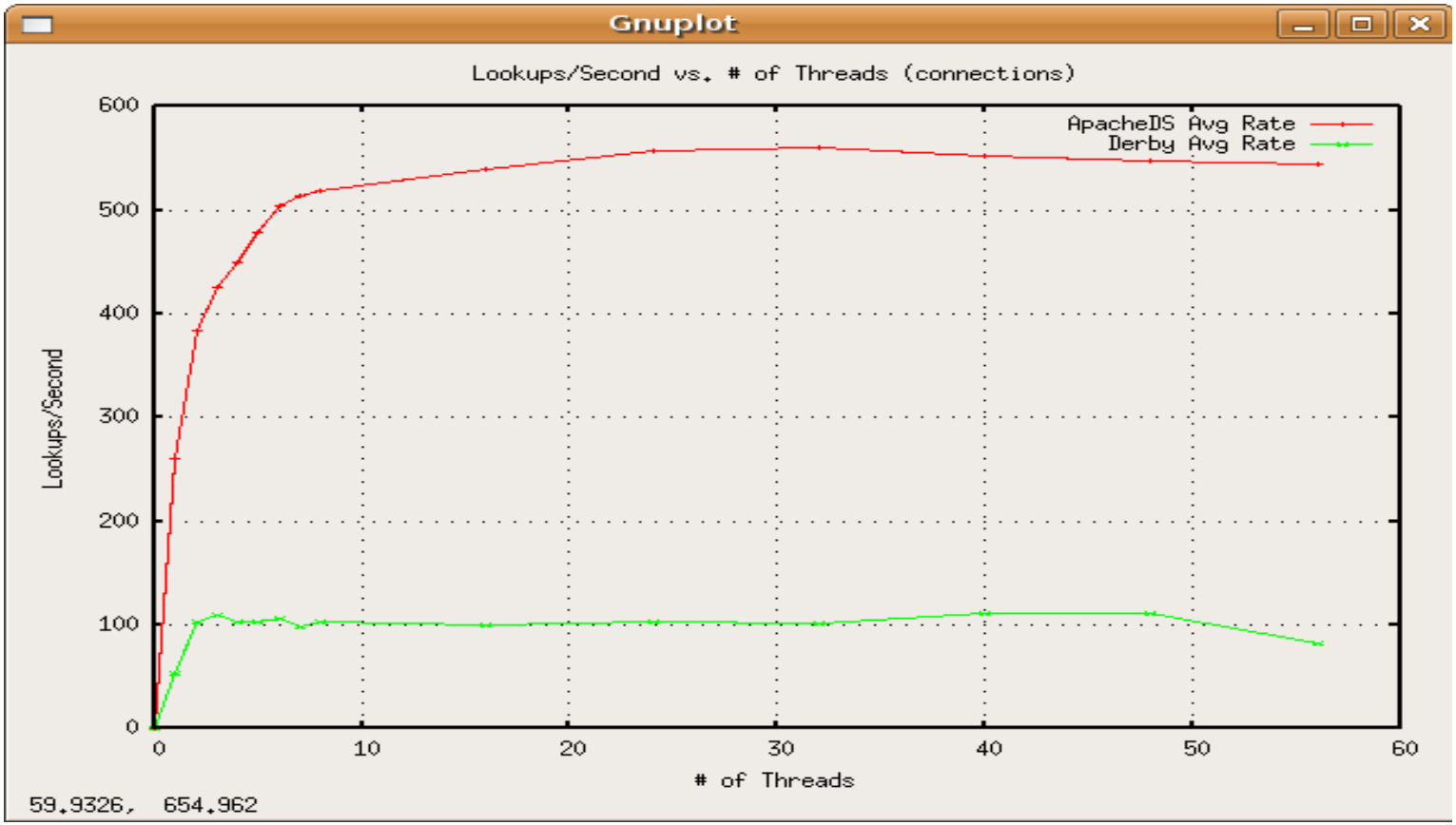
Software Configuration

- Apache Directory Server
  - 384 Mb heap
  - 16 Mb stack
  - uid attribute index
  - objectClass attribute index
  - other default system indices
  - 8 worker threads
- Apache Derby (with Network Server)
  - 384 Mb heap
  - 16 Mb stack
  - uid column index
  - 56 worker threads

Lookups Per Second (LPS)

# of Threads	ApacheDS LPS	Derby LPS
1	261	54
2	384	104
3	425	109
4	450	103
5	478	103
6	505	107
7	514	98
8	519	103
16	540	101
24	557	104
32	560	102
40	553	111
48	548	112
56	545	82

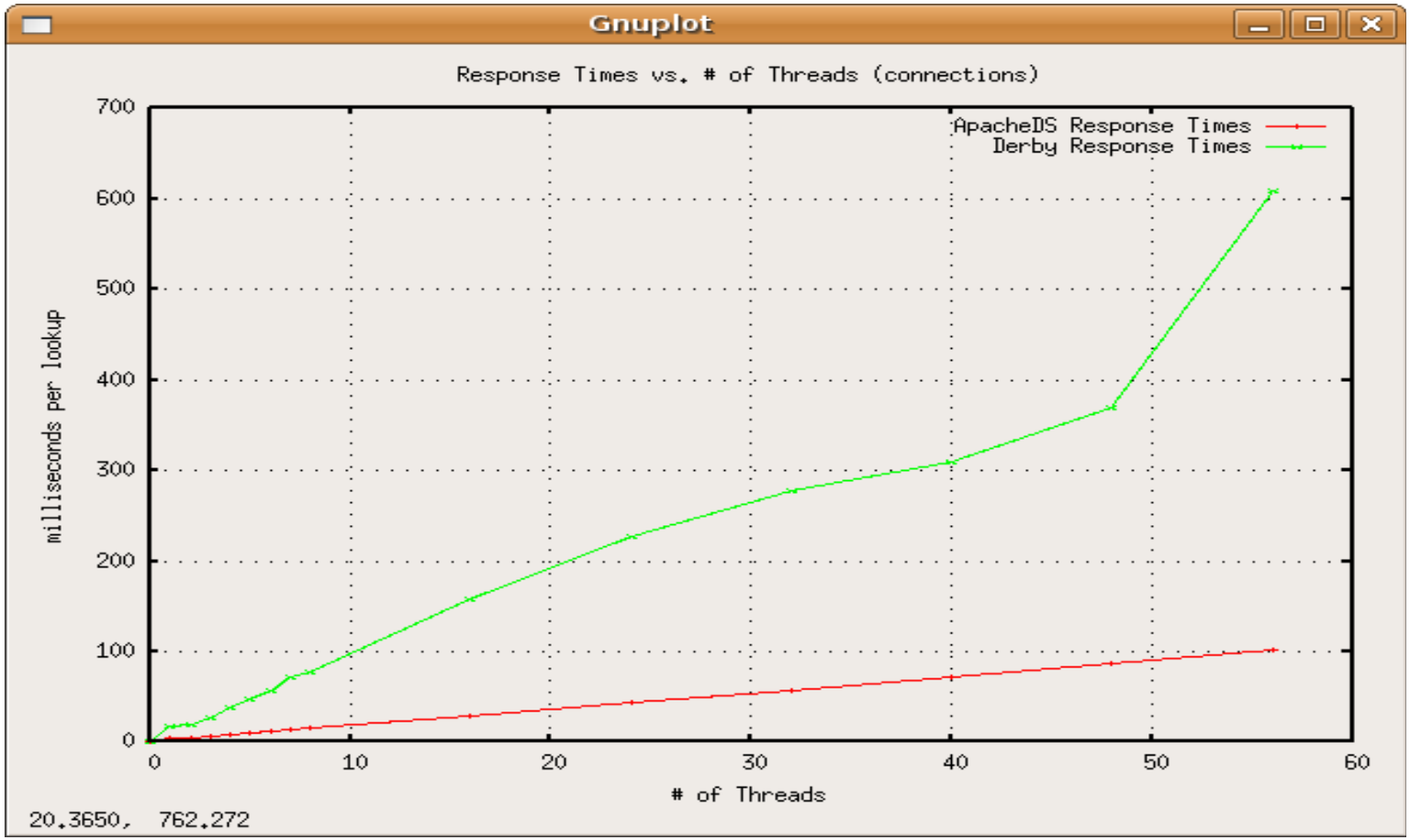
Lookups Per Second (LPS)



Response Time (RT)

# of Threads	ApacheDS RT	Derby RT
1	3.82	18.34
2	5.2	19.21
3	7.03	27.43
4	8.88	38.65
5	10.45	48.48
6	11.87	56.28
7	13.61	71.19
8	15.39	77.34
16	29.6	158.25
24	43.11	226.61
32	57.18	277.39
40	72.29	310.31
48	87.52	369.5
56	102.66	608.12

Response Time (RT)



ØREDEV

Conclusions

- ApacheDS lookups scale ~5x better than Derby with increasing load and concurrency.
- ApacheDS lookup response times are ~6x faster than Derby with increasing load and concurrency.
- No replication yet!

- We can say directories are faster at lookups and outperform databases by several factors?

- More experiments of value
  - minimal load but scale clients, & measure RT
  - same experiments with many LDAP replicas

Questions?

ØREDEV

Remember!

Enter the evaluation form and be a part of making Øredev even better.

You will automatically be part of the evening lottery