redis

Karolína Burská
Pavel Šeda

Outline

- Intro to Redis
- Support and popularity
- Data Persistence
- Security
- Installation Steps + CLI
- Redis Data Types (+Working with it)

Intro to Redis

- Open source; in-memory key-value store
- Often ranked as one of the most popular key-value database (https://db-engines.com/en/ranking)
- Currently sponsored by a private software company **Redis Labs**
- Support of:
  - Different kinds of abstract data structures - strings, lists, sets, bitmaps, spatial indexes…
  - Optional durability
  - High-level, atomic operations (intersection, union)
  - Master-slave replication of data

Language support

- Many languages support Redis binding:
    - Java (Jedis, JDBC-Redis, RJC, RedisClient, ...),
    - C (credis, eredis, hiredis, libredis, ...),
    - C# (ServiceStack.Redis, StackExchange.Redis, Sider, csredis, ...),
    - Python (aredis, desir, brukva, Pottery, Pypredis, …),
    - Ruby (redic, redis-rb, oxblood, em-redis, em-hiredis, ...),
    - PHP (amphp/redis, Predis, phpredis, Credis, ...),
    - Matlab (redis-octave), etc.

Used by companies

- Used by companies like Twitter, GitHub, StackOveflow, Pinterest, Amazon Web services (Elasticache), Microsoft (offers Redis Cache in Azure)
- Common uses:
  - Caching
  - Publish-subscribe queues

Comparison with other Data Stores

- Memcached
  - Open-source, in-memory, multithreaded key-value store
  - Does not provide persistence
- MongoDB
  - Open-source, document DB (supports richer data types)
  - Slower than Redis and Memcached

Comparison with other Data Stores

	Redis	Memcached	MongoDB
in-memory	X	X	
persistent	X		X
key-value store	X	X	
support of more than strings	X		X
multithreaded		X	X
support of larger-than-memory datasets			X
speed	memory	memory	disk

In-memory: the speed of cache

The main feature of Redis - speed

- Memory access is faster than disk access (0.1 µs vs. 10 ms)
- Support of persistence - can persist its data to disk
- Two approaches to achieve durability

Redis Persistence

Data stored in-memory does not survive a server shutdown.

Two options which can be combined:

- **(RDB) Redis Database File**
  - Point-in-time snapshots of a dataset
  - Performed at specified intervals
- **(AOF) Append-Only File**
  - Logging of every received write operation
  - Commands are logged using Redis protocol format
  - Played again at server startup

Replication & Sharding

Redis Cluster - a distributed implementation of Redis

High performance and linear scalability

- Hashing - not consistent hashing, but a different form of sharding (partitioning). Every key is part of an **hash slot**
- **Master-slave** model replication (mirroring), every hash slot has from 1 to N replicas
- Uses asynchronous replication (thus does **not** guarantee **strong consistency**)

Security

- Redis is designed to be accessed by trusted clients inside trusted environments
  - Do not expose Redis directly to the internet
  - Web pages must mediate access between Redis and untrusted clients
- Redis port should be firewalled to prevent access from outside (network security)
- Protected mode (Redis replies queries only from the loopback interfaces)
- Redis does not support encryption
- Disable specific commands

What are the limitations?

- “in-memory” -> limited by the size of RAM. **But**, nowadays uses *swap file* for the unused values,
- Memory fragmentation - working with amounts of data may result in performance degradation
- master-slave architecture - when designed poorly (one master node, e.g.), there is more load on the master node

Practical Intro

Installing Steps (Linux - Tested on Ubuntu)

Installation commands:

- $ sudo apt-get update (after run cmd put your system pwd)
- $ sudo apt-get upgrade (confirm with Y)
- $ sudo apt-get install redis-server (confirm with Y)

Now it is prefer to copy configuration file to somewhere on the disk before we will do another steps

- $ sudo cp /etc/redis/redis.conf /etc/redis/redis/redis.conf.default

Run redis:

- $ redis-server (it is running on port 6379 - default port)

Check running Redis:

- $ redis-cli (running redis cmd client interface) -> $ ping (should return PONG)

Installing Steps (Windows)

Installation steps:

- Download from https://github.com/MSOpenTech/redis/releases (download last release .msi file)
- Install it, go through step by step windows (NOTE: when selecting **Destination Folder** check Add the Redis installation folder to the **PATH environment variable**)

Run cmd as administrator at Redis installation folder:

- $ redis-server redis.windows.conf (set configuration file)
- $ redis-cli
- $ ping (should return PONG)

CLI (Command Line Interface) - basic commands

- $ SET foo 100 (set for ‘foo’ key value ‘100’)
- $ MSET key1 “Hello” key2 “World” (set multiple keys)
- $ APPEND key1 “ World”
- $ GET foo (should return “100”)
- $ INCR foo (returns (integer) 101)   or $ DECR foo
- $ EXISTS foo (return (integer) 1 if exists or (integer) 0 if do not exists)
- $ DEL foo (deletes the key)

Running cli commands directly from standard cmd:

- $ redis-cli INCR foo > commands.txt (creates a file and saves the values here)

Monitoring: $redis-cli monitor (will monitor every action on Redis instance)

CLI (Command Line Interface) - basic commands

Key spaces:

- $ SET server:name myserver
- $ GET server:name
- $ SET server:port 6379
- $ GET server:port (give your values different name spaces)

Expiration cmds:

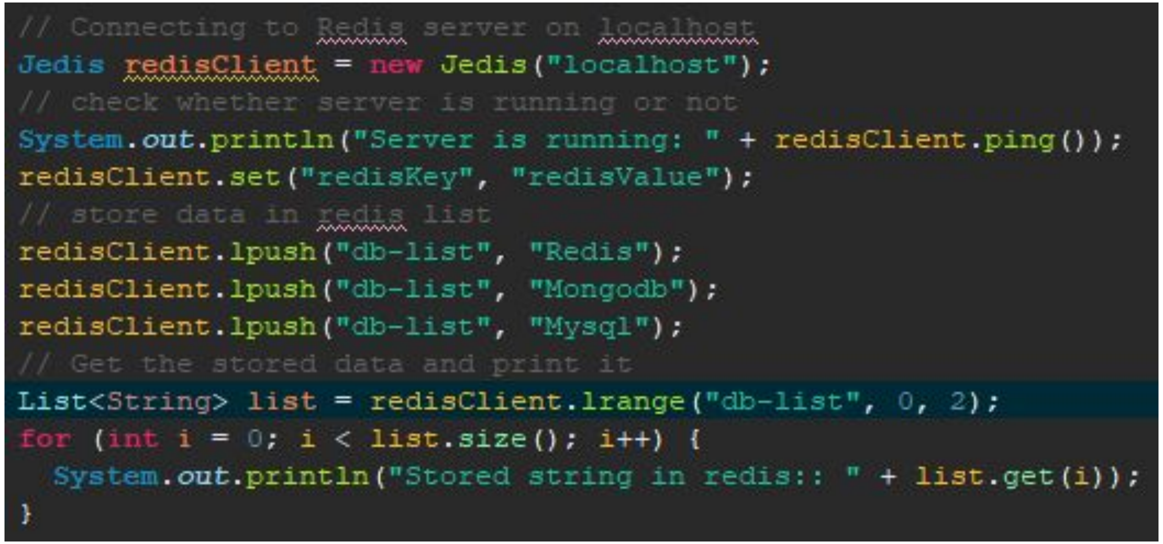
- $ SET resource:foo hello
- $ EXPIRE resource:foo 120
- $ TTL resource:foo (test time to expiration)

Delete everything:

- $ FLUSHALL (get rid of everything)

Java Clients

- https://redis.io/clients
- e.g., for Java Jedis:



```
// Connecting to Redis server on localhost
Jedis redisClient = new Jedis("localhost");
// check whether server is running or not
System.out.println("Server is running: " + redisClient.ping());
redisClient.set("redisKey", "redisValue");
// store data in redis list
redisClient.lpush("db-list", "Redis");
redisClient.lpush("db-list", "Mongodb");
redisClient.lpush("db-list", "Mysql");
// Get the stored data and print it
List<String> list = redisClient.lrange("db-list", 0, 2);
for (int i = 0; i < list.size(); i++) {
  System.out.println("Stored string in redis:: " + list.get(i));
}
```

Data Types

- Strings
- Lists
- Sets
- Sorted Sets
- Hashes

Working with Lists

- Sorted by insertion order
- Values could be pushed on the head or tail
- Basic commands:
    - $LPUSH mylist “a”; $LPUSH mylist “b”;, $RPUSH mylist “c”
    - LRANGE mylist 1 2; LRANGE mylist 0 -1 (returns all from mylist)
    - LLEN mylist (returns the length of the ‘mylist’ list)
    - LPOP (removes and returns the first element of a list)
    - LINSERT (insert on the exact place in the list)

Working with Sets

- Unordered collection of strings
- Can add, remove and test for existence
- Do NOT allow repeating members
- Basic commands:
    - SADD (Adds given values to a set - ignore existing values)
    - SREM (Removes values from a set)
    - SISMEMBER (Tests if the given value is in the set)
    - SMEMBERS (Returns a list of all of the members of a set)

Working with Sorted Sets

- Every member is associated with a “score”
- Score is required:
  - Float / Number, Score is NOT unique / Values are
- In case adding another same key, then the score is also overridden
- Basic Commands:
  - ZADD (Adds given values to a sorted set)
  - ZREM (Removes values from a sorted set)
  - ZRANGEBYSCORE people 1950 1990 (All people with score between ..)
  - ZRANK (Returns the rank of a member with scores ordered high to low)
  - ZINCRBY (Increments the score of member)

Working with Hashes

- What is a Hash?
  - Maps between string fields and string values
  - Perfect for representing objects
- Basic commands:
  - HSET (Sets a field in the Hash)
  - HMSET user2 name “Pavel” email “jill@gmail.com” age “26” (Sets a multiple fields to their respective values)
  - HGET user2 name (Returns name from user2)
  - HMGET user2 name age (Returns name and age from user2)
  - HGETALL user2 (Return all fields from user2)

Summary

- Key-Value DB
    - Free, Super fast,
    - In-memory cache, Open Source,
    - Stable, Ease to use, Performance
- Support many programming clients
- Support of advanced data structures (Strings, Lists, Sets, Sorted Sets, Hashes)

Thank you for your attention

Questions?

441048@mail.muni.cz (Pavel Šeda)

396296@mail.muni.cz (Karolína Burská)