Redis is 10!

Dr. Christoph Zimmermann

redislabs, 6. 7. 2019

stat /proc/self

- PhD in reflective operating system architectures
- First crush on Linux: kernel 0.95
- Tech support + more @ FraLUG
- Arch package maintainer
- Hobbies include:
  - SDLC
  - IT security and other forms of black art
  - Community liaison / solution architect @ redislabs

cat /etc/motd

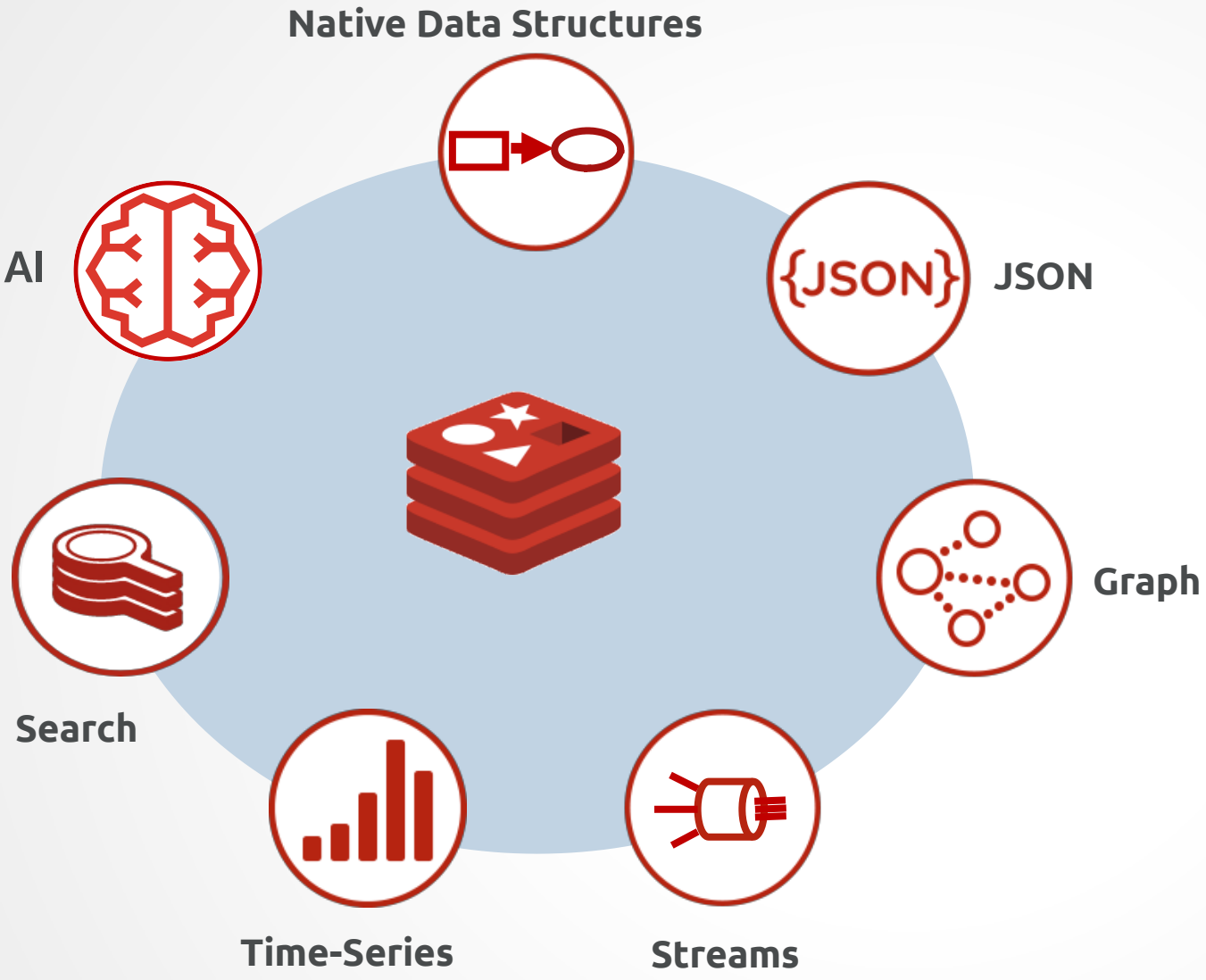
- What is Redis
- Architecture
- Features
- The application perspective
- Summary / outlook

Redis

*An **in-memory multi-model database***

cat /etc/group

ls /opt/redis

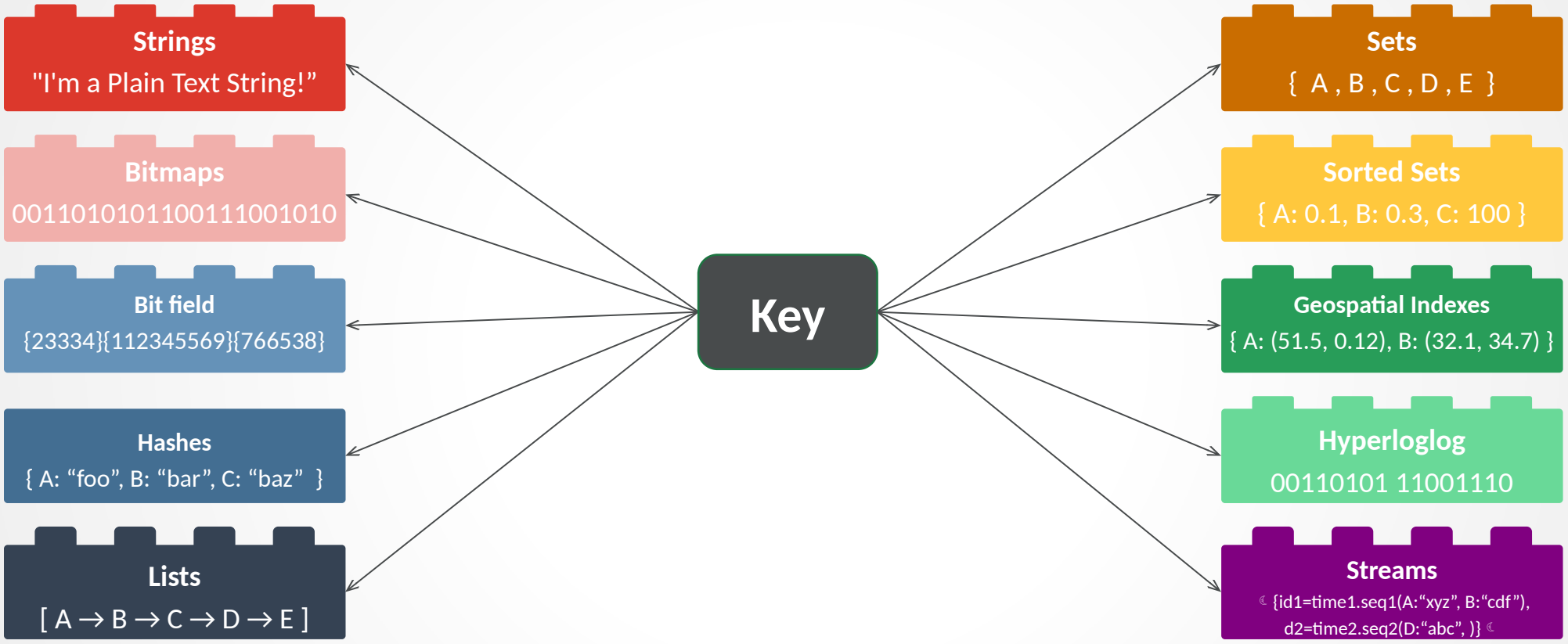


- Dedicated engine for each data model (vs. API only)
- Models engines can be selectively loaded, according to use case
- All model engines access the same data, eliminating the need for transferring data between them

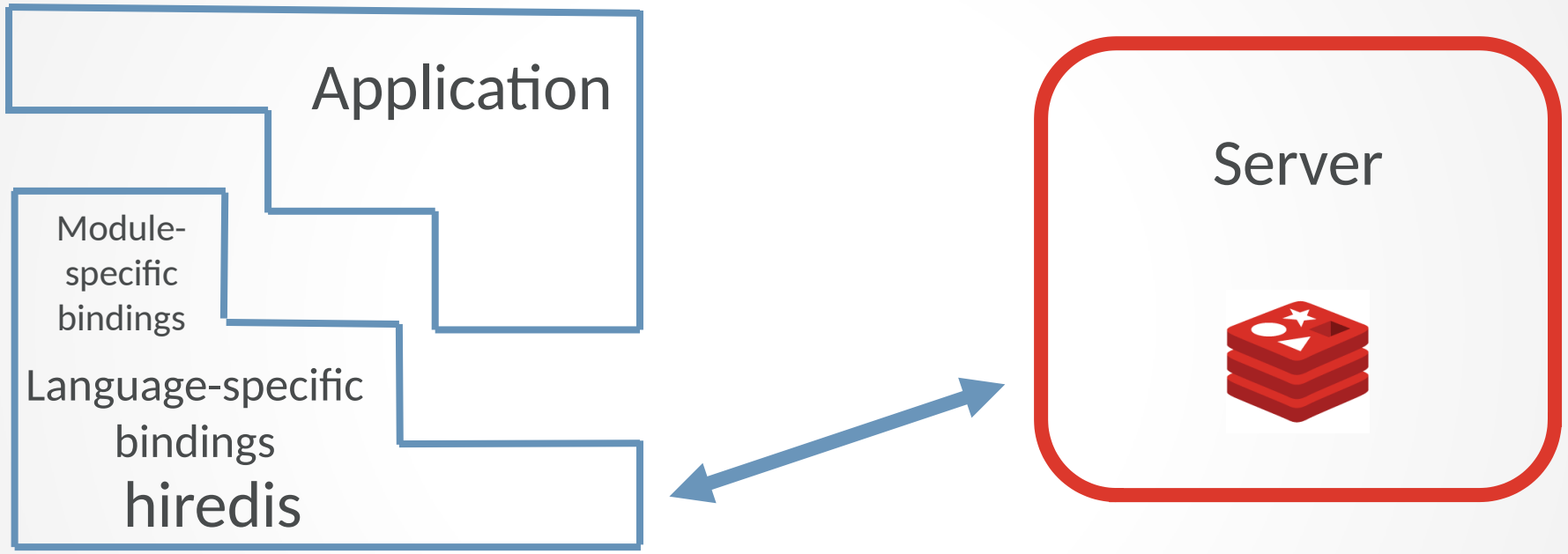
man redis-server

- Different persistence modes:
  - Append-only files (AOF)
  - Snapshots (RDB)
- Cluster: `utils/create-cluster/create-cluster`
- High availability:
  - Via sentinels (`src/redis-sentinel`)
- Full CAP coverage:
  - HA → ACID compliance

type which


Strings
"I'm a Plain Text String!"
Bitmaps
0011010101100111001010
Bit field
{23334}{112345569}{766538}
Hashes
{ A: “foo”, B: “bar”, C: “baz” }
Lists
[A → B → C → D → E]
Key
Sets
{ A , B , C , D , E }
Sorted Sets
{ A: 0.1, B: 0.3, C: 100 }
Geospatial Indexes
{ A: (51.5, 0.12), B: (32.1, 34.7) }
Hyperloglog
00110101 11001110
Streams
{id1=time1.seq1(A:“xyz”, B:“cdf”),
d2=time2.seq2(D:“abc”,)}

redis-server



cat hello_world.py

```
#
# Hello world for redis
#
import redis

def main():
    # decode_responses: depending on the Python version
    r = redis.Redis(decode_responses=True)
    r.set('Hello', 'world')
    val = r.get('Hello')
    print('Hello: ', val)

if __name__ == '__main__':
    exit(main())
```

shutdown -H +5

- Outlook (redis 6):
  - ACLs / user administration
  - Improved wire protocol
  - Cluster proxy
- Popular in-memory NoSQL
- Key/value store → multi-modal DB
- Extensibility: modules + more

apropos redis

- redis.io: fulls OSS documentation
- antirez.com: Salvatore‘s blog
- github.com/antirez/redis: Redis source code
- redislabs.com/community: community documentation
- redislabs.com/redis-for-dummies: Redis quickstart
- university.redislabs.com: redislabs university

Discussion

Thank you!

Dr. Christoph Zimmermann

monochromec at <ignore>space</ignore>gmail<dot></dot>com

Backup

less ~home/dev/redis_graph.py

```
import redis
from redisgraph import Node, Edge, Graph

def print_res(result):
# Iterate through resultset, skip header row at position 0
    for record in result.result_set:
         person_name = record[0]
         person_age = record[1]
         visit_purpose = record[2]
         country_name = record[3]
         print('Name: {:>12}\tAge: {:>12}\tVisit: {:>12}\tCountry
{}'.format(person_name,                          person_age, visit_purpose,
country_name))

def main():
    # Depending on Python version
    r = redis.Redis(decode_responses=True)
    redis_graph = Graph('social', r)

    john = Node(label='person', properties={'name': 'John Doe', 'age': 33, 'gender':
'male',                 'status': 'single'})
          redis_graph.add_node(john)
```

less ~home/dev/redis_graph.py

```
japan = Node(label='country', properties={'name': 'Japan'})
redis_graph.add_node(japan)

edge_john = Edge(john, 'visited', japan, properties={'purpose': 'pleasure'})
redis_graph.add_edge(edge_john)

pearl = Node(label='person', properties={'name': 'Pearl White', 'age': 25, 'gender':
            'female', 'status': 'married'})
redis_graph.add_node(pearl)

australia = Node(label='country', properties={'name': 'Australia'})
redis_graph.add_node(australia)

edge_pearl = Edge(pearl, 'visited', australia, properties={'purpose': 'business'})
redis_graph.add_edge(edge_pearl)

redis_graph.commit()

print('')
```

less ~home/dev/redis_graph.py

```
for i in ['pleasure', 'business']:

print('==== Purpose: {} ===='.format(i))

query = '''MATCH (p:person)-[v:visited {purpose:"{}"}]-
>(c:country)

                    RETURN p.name, p.age, v.purpose,
c.name'''.format(i)

result = redis_graph.query(query)

print_res(result)

print('')


if __name__ == '__main__':

    exit(main())
```