## Netflix Data Benchmark (NDBench)

Ioannis Papapanagiotou Cloud Database Engineering

#### Databases and Caches

- >100M subscribers
- Running tens of thousands of nodes of
  - a. Cassandra
  - b. ElasticSearch
  - c. EVCache (Memcache)
  - d. Dynomite (Redis)





#### Requirements

- Single benchmark framework for all data stores
- Dynamically change the benchmark configuration while the test is running.
- Test a platform along with production microservices
- Be able to integrate with other platform cloud services
- Run for an infinite amount duration
  - Wanted to test failure scenarios
  - Test long running maintenance such as database repairs

#### What is NDBench?

Netflix Data Benchmark (NDBench) is a **Pluggable cloudenabled** benchmarking tool that can be used across **any data store system**.



#### **NDBench allows**

- Run infinite horizon tests to identify potential memory leaks from long running processes
- Test out different failure scenarios in TEST
- Performance test heavy processes like repairs, compaction, backup/restores



### **NDBench integration**

Netflix OSS integration:

- Archaius 2: Dynamic configurations
- Spectator: Metrics
- Eureka: Middle-tier discovery service

NDBench is designed so that libraries are injected:

• Cross-cloud support



#### Side-by-Side comparison





#### **Varying Data Models**





#### **Pluggable Patterns and Loads**





#### **Different Client APIs**



#### Architecture

- **Core**: Workload generator
- **API**: Adding plugins
- Web: UI and the servlet context listener



#### **Configuring a cluster**



1. Select a Cluster					Instance Statistics					
Cluster Jocalhost •					localhost:8080					
					- readLatAvg - readRPS - writeLatAvg - writeRPS					
2 Connect a Driver									4	
					40					
Jriver InMemoryTest										
inMemoryTestPlugin - Connectio	oninfo :: inMemor	yMap Key Count: 0			30					
		ER			22					
HIDE SETTINGS	UNINECT DRIVI				20					
					10					
Initial Settings		Runtime Settin	gs		100 5					
These settings can only be changed before connecting a Driver.		These settings can while a Load Test is	be changed at any time running.		0	0 15:54 15:54:30	15:55 15:55:30	15:56 15:5	56:30 15:57	
backfillStartKey	1	readRateLimit	25		cacheHitRa	atioInt 🗆 cacheHits	cacheMiss	readFailure		
dataSize	128	writeRateLimit	writeRateLimit 15		readLatAvg	readLatP50	readLatP95	readLatP99		
dataSizeLowerBound	1000				<ul> <li>writeFailure</li> </ul>	e vriteLatAvg	writeLatP50	writeLatP95	5	
dataSizeUpperBound	5000				<ul> <li>writeLatP9</li> <li>writeSucce</li> </ul>	9 owriteLatP995	writeLatP999	writeRPS		
numBackfill	1									
numKeys	10000				HIDE STAT	'S JSON				
numReaders	1									
numValues	100				{ "cache	HitRatioInt": 29.				
numWriters	1				"cache	Hits": 2791,				
readRateLimit	25				"readF	ailure": 0,				
statsResetFreqSeconds	200				"readL "readL	_atAvg": 2, _atP50": 1,				
statsUpdateFreqSeconds	5				"readL	atP95": 1,				
writeRateLimit	15				"readL	atP995": 1,				
readEnabled	1				"readL "readR	RPS*: 25,				
useStaticData					"readS "write	Success": 9450, Failure": 0,				
useVariableDataSize			-		"write	eLatAvg": 3,				
writeEnabled	1		SAVE SETTINGS		"write	LatP95": 5,				
					"write	eLatP99": 6, eLatP995": 7,				
					"write	at at 2000 * 14				

	Backfill	Writes	Reads	
Instance	(I)	● (II)	• •	
ocalhost:8080	0	0	DandomString	

#### **C\* AMI Certification Process**



#### NDBench exemplar uses





E Log of E Log



CASS | cass\_perftest\_trustyupg C Refresh All Auto Refresh Show Legend Logarithmic Start 2015-07-0512709 8 End Minus Smins 8 Shift None 8 Starp Auto 8 Time Zone US/Pacific 8 Cent OS Trusty Migration 80. SK ..... Trusty Trusty 1.00 ante nice adas entre entre entre atre atre atre adas atre entre 3.422 00.00 00.00 00.00 32.00 ik of II tog Cent OS adapation and a later of the second second 268.0 184 124.0 **Trusty Migration** .... Trusty **Termination/ Trusty Migration time** n'oo xin ofee eise eise uise uise 18 40 21 00 3.412 01 00 00 00 00 00 12 00 15 00 18 00 21 00 solar mite mine mine mine anime 00.00 00.00 12.00 15.00 C D Le C II Log Trusty ខ 🗈 🔟 🗗 👪 Log 이 타 너 바 H Log Cent OS Reads Count per CF Reads - ReadLate Loros (avg per note in micros) 1.0. 2.64 2.8-16NV 0.0 18:00 21:00 12.00 15.00 10.00 21.00 1.022 0 00.00 00.00 12.00 15.00 02 00 00 00 00 00 12 00 15 00 10 40 10 m 18 00 EL 00 M23 03 00 06 00 00 12 00 15 00 18 00 EL 00 M20 ation 22.00 3.4.25 00.00 00.00 09.00 32.00 -----00.00 12.00 15.00 10.00 21.00 82:00 06:00 #2 00 MG 00 00 00 12 00 15 00 10 00 72 00 1421 07 00 00 00 00 12 00 Trusty Writes - histo\_WriteLat\_99\_bucket (avg per CF in micros) Cent OS Writes - Writ writes - WriteLaten yMicros (avg per note in micros) 305.0 200.0 308.0-0.0 09 00 12 00 15 00 18 00 22 00 12 00 15 00 18 00 21 00 2.121 80.00 00.00 86 88 89 88 12 88 15 88 69 00 12 00 15 0 은 윤 년 **81 tog** C\* - ThreadPoolMonitor - Active 30.0 lehan hours have 11.00 x/23 03.00 04.00 05.00 12.00 13.00 16.00 11.00 x/20 03.00 04.00 05.00 200 25.0

: 🕞 🖉 🖬 🖬 Log

#### NETFLIX





#### **Does our systems scale linearly?**

# Generating Millions of Ops/sec





#### NDBench Generated Traffic (DC\_ONE mode)

### **Auto-tuning (Experimental)**

- Requirements
  - Find the correct to load that the cluster can sustain
  - Evaluate different data stores
    - How many RPS/WPS they can go up to
    - Number of documents indexed per second per node
- Gradually increase load until we get failures
- After X% of failures, NDBench stops stepping up the load
  - Performs exponential backup to figure out if cluster is oversaturated.





Github: <u>https://github.com/netflix/ndbench</u> Talk to us: <u>https://gitter.im/Netflix/ndbench</u>

#### **Q & A**

Ioannis Papapanagiotou https://twitter.com/ipapapa

