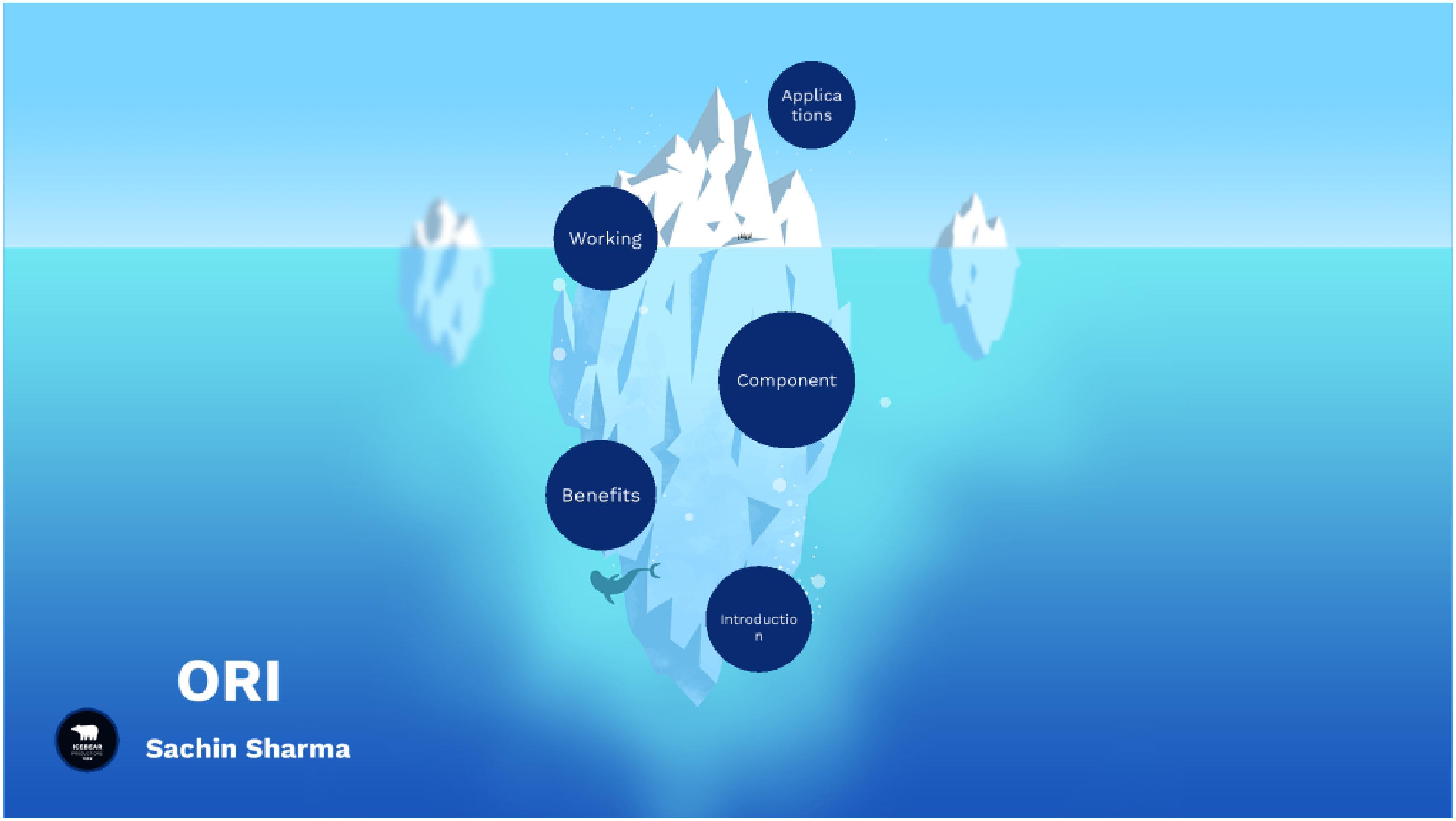


Applications

Working

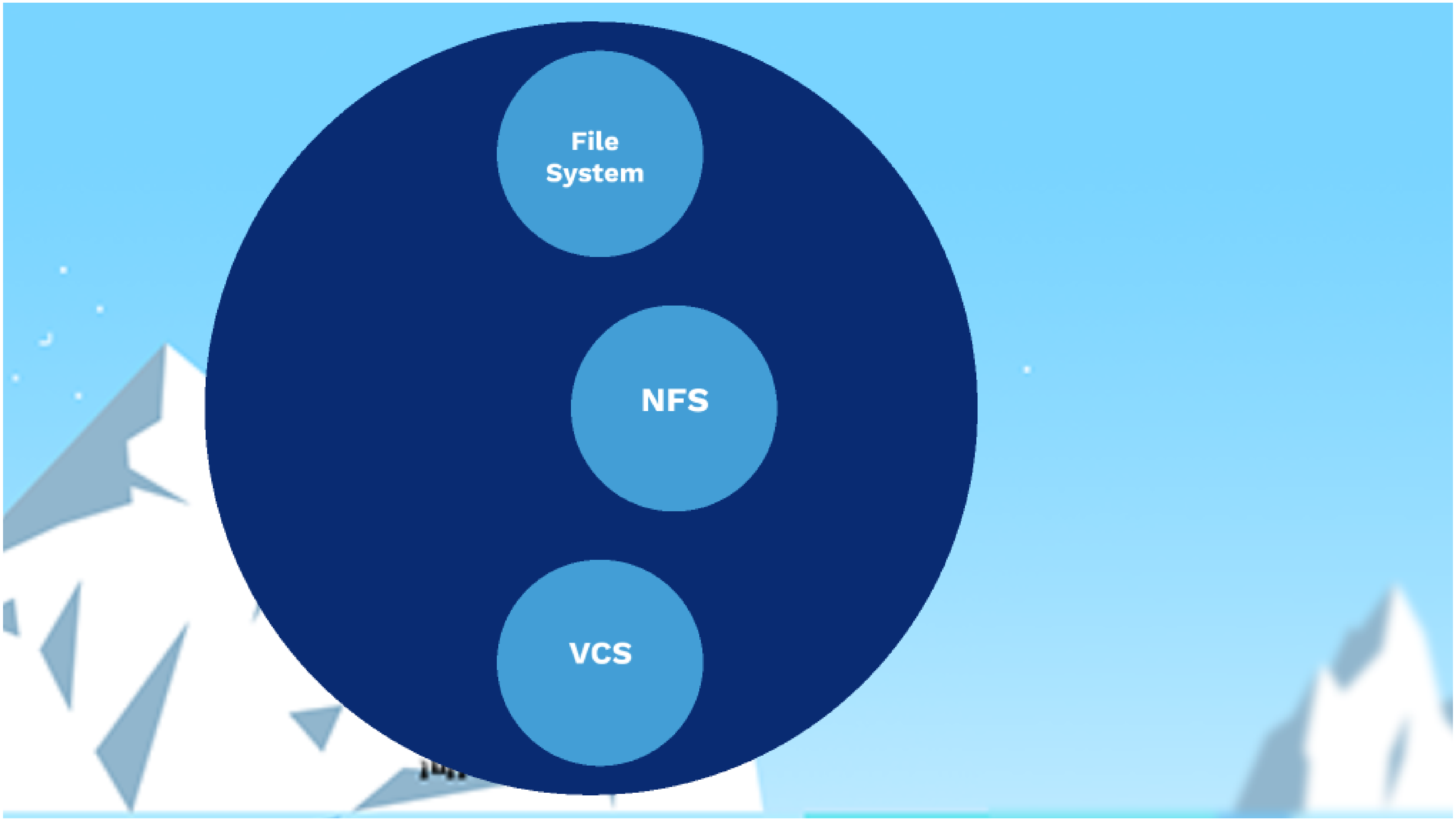
Component

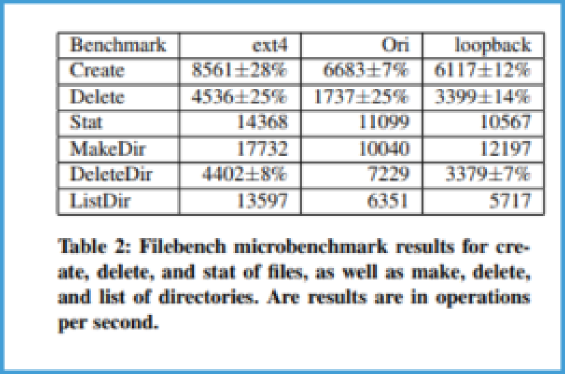
Benefits

Introduction

ORI

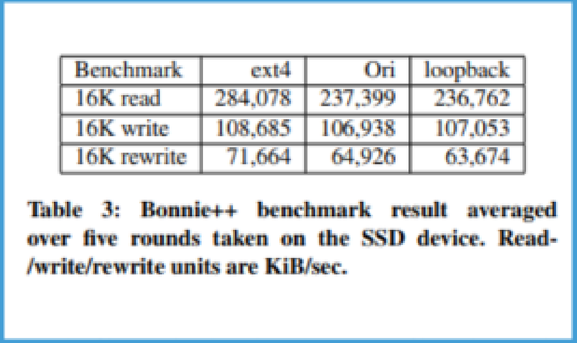
Sachin Sharma


File System
NFS
VCS



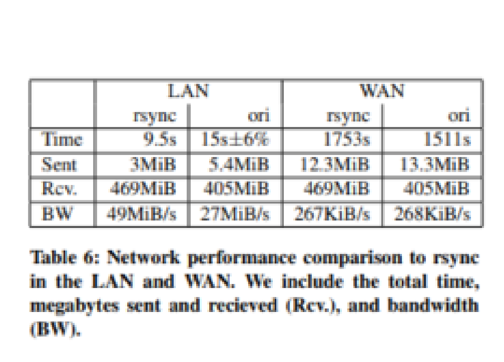
Benchmark	ext4	Ori	loopback
Create	8561±28%	6683±7%	6117±12%
Delete	4536±25%	1737±25%	3399±14%
Stat	14368	11099	10567
MakeDir	17732	10040	12197
DeleteDir	4402±8%	7229	3379±7%
ListDir	13597	6351	5717

Table 2: Filebench microbenchmark results for create, delete, and stat of files, as well as make, delete, and list of directories. Are results are in operations per second.



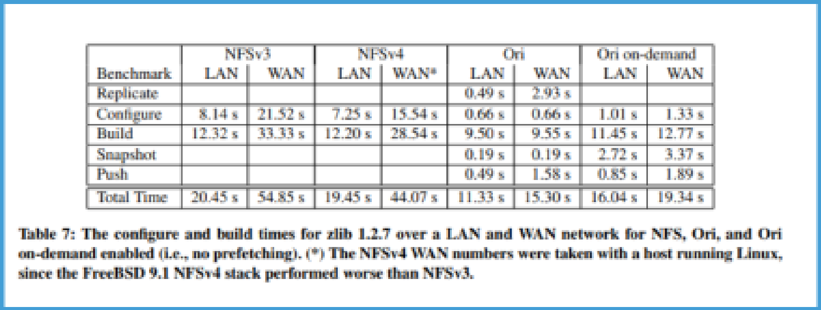
Benchmark	ext4	Ori	loopback
16K read	284,078	237,399	236,762
16K write	108,685	106,938	107,053
16K rewrite	71,664	64,926	63,674

Table 3: Bonnie++ benchmark result averaged over five rounds taken on the SSD device. Read-/write/rewrite units are KiB/sec.



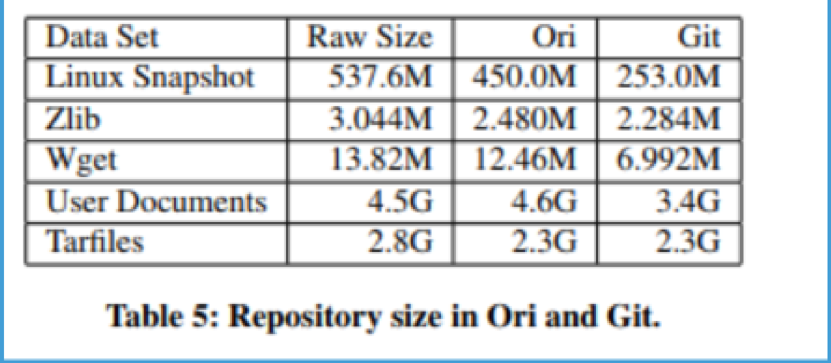
	LAN		WAN	
	rsync	ori	rsync	ori
Time	9.5s	15s±6%	1753s	1511s
Sent	3MiB	5.4MiB	12.3MiB	13.3MiB
Rcv.	469MiB	405MiB	469MiB	405MiB
BW	49MiB/s	27MiB/s	267KiB/s	268KiB/s

Table 6: Network performance comparison to rsync in the LAN and WAN. We include the total time, megabytes sent and recieved (Rcv.), and bandwidth (BW).



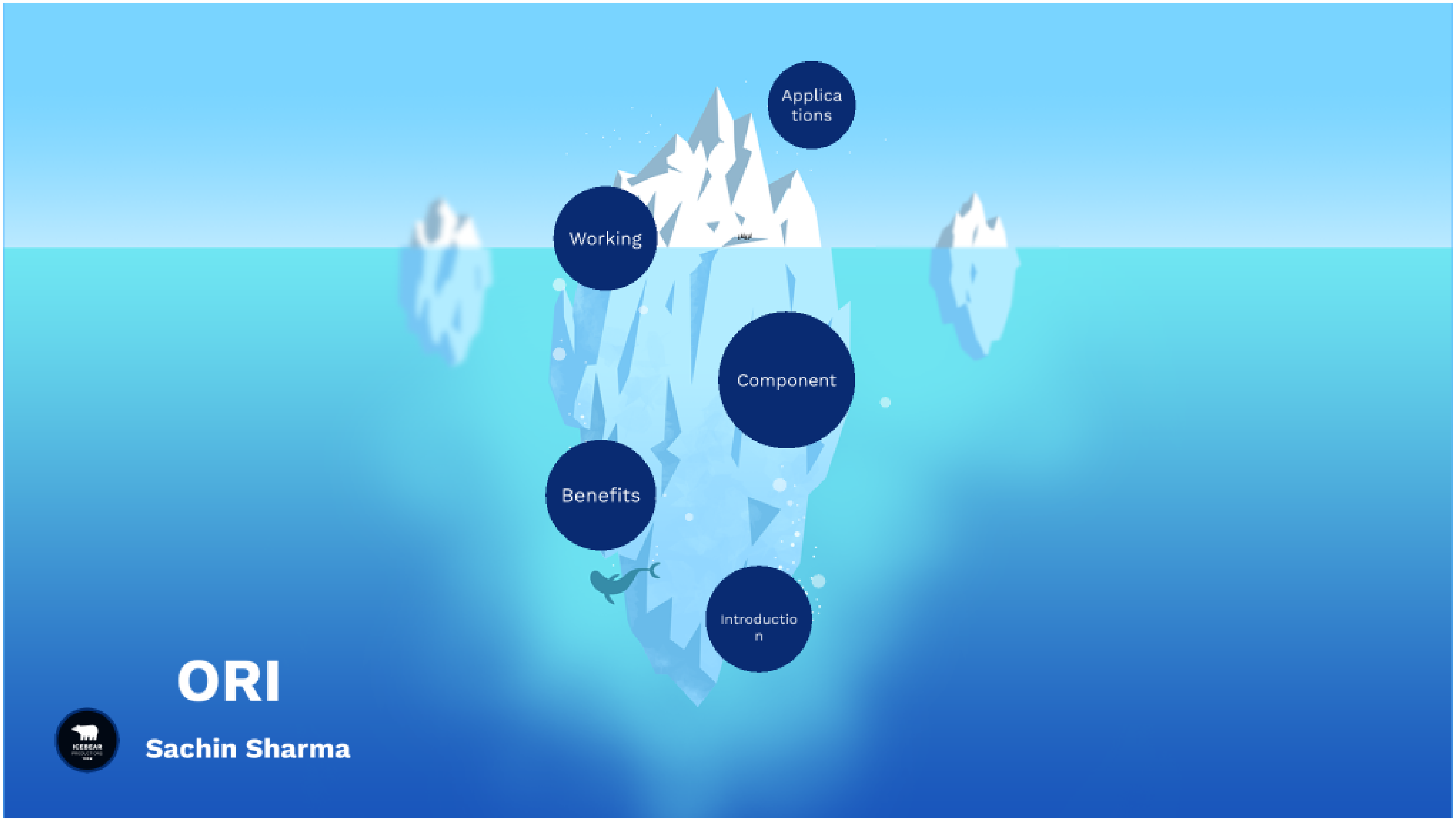
	NFSv3		NFSv4		Ori		Ori on-demand	
Benchmark	LAN	WAN	LAN	WAN*	LAN	WAN	LAN	WAN
Replicate					0.49 s	2.93 s		
Configure	8.14 s	21.52 s	7.25 s	15.54 s	0.66 s	0.66 s	1.01 s	1.33 s
Build	12.32 s	33.33 s	12.20 s	28.54 s	9.50 s	9.55 s	11.45 s	12.77 s
Snapshot					0.19 s	0.19 s	2.72 s	3.37 s
Push					0.49 s	1.58 s	0.85 s	1.89 s
Total Time	20.45 s	54.85 s	19.45 s	44.07 s	11.33 s	15.30 s	16.04 s	19.34 s

Table 7: The configure and build times for zlib 1.2.7 over a LAN and WAN network for NFS, Ori, and Ori on-demand enabled (i.e., no prefetching). (*) The NFSv4 WAN numbers were taken with a host running Linux, since the FreeBSD 9.1 NFSv4 stack performed worse than NFSv3.



Data Set	Raw Size	Ori	Git
Linux Snapshot	537.6M	450.0M	253.0M
Zlib	3.044M	2.480M	2.284M
Wget	13.82M	12.46M	6.992M
User Documents	4.5G	4.6G	3.4G
Tarfiles	2.8G	2.3G	2.3G

Table 5: Repository size in Ori and Git.



Applications

Working

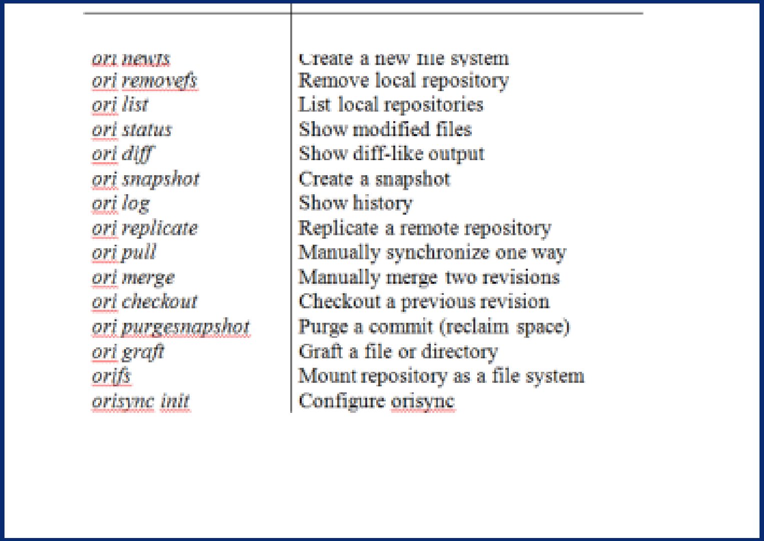
Component

Benefits

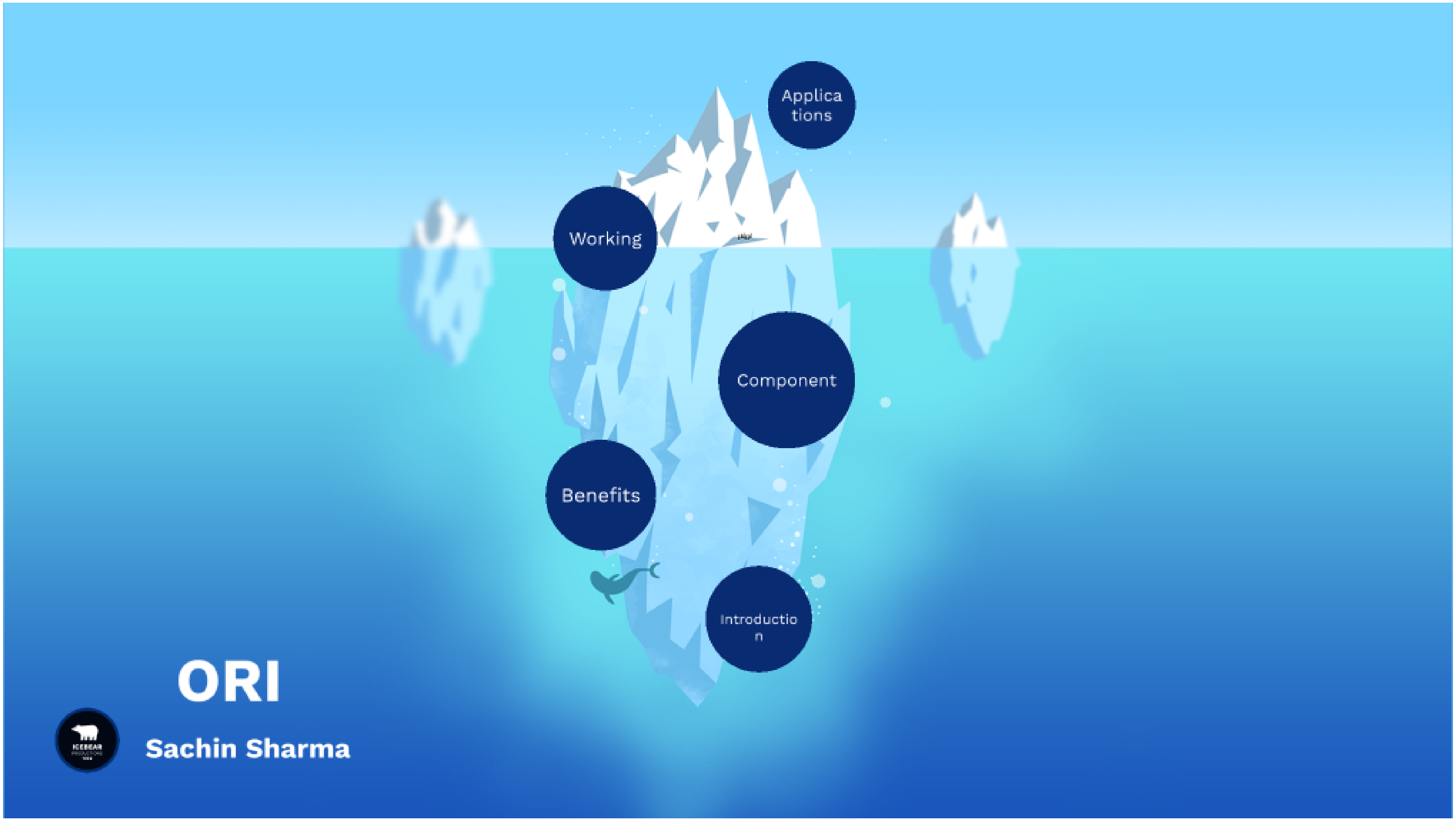
Introduction

ORI

Sachin Sharma



ori newfs	Create a new file system
ori removefs	Remove local repository
ori list	List local repositories
ori status	Show modified files
ori diff	Show diff-like output
ori snapshot	Create a snapshot
ori log	Show history
ori replicate	Replicate a remote repository
ori pull	Manually synchronize one way
ori merge	Manually merge two revisions
ori checkout	Checkout a previous revision
ori purgesnapshot	Purge a commit (reclaim space)
ori graft	Graft a file or directory
orifs	Mount repository as a file system
orisync init	Configure orisync



Applications

Working

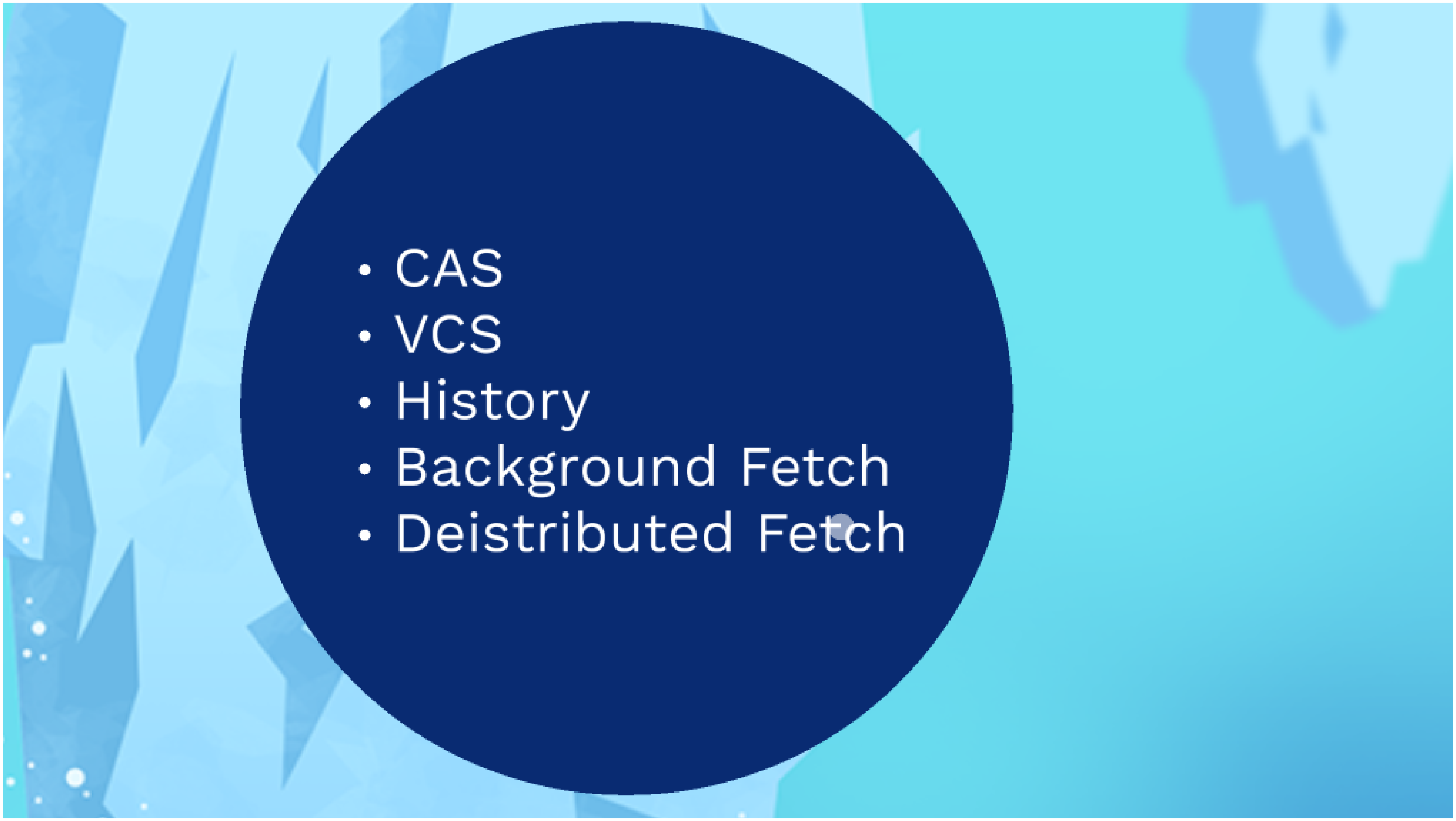
Component

Benefits

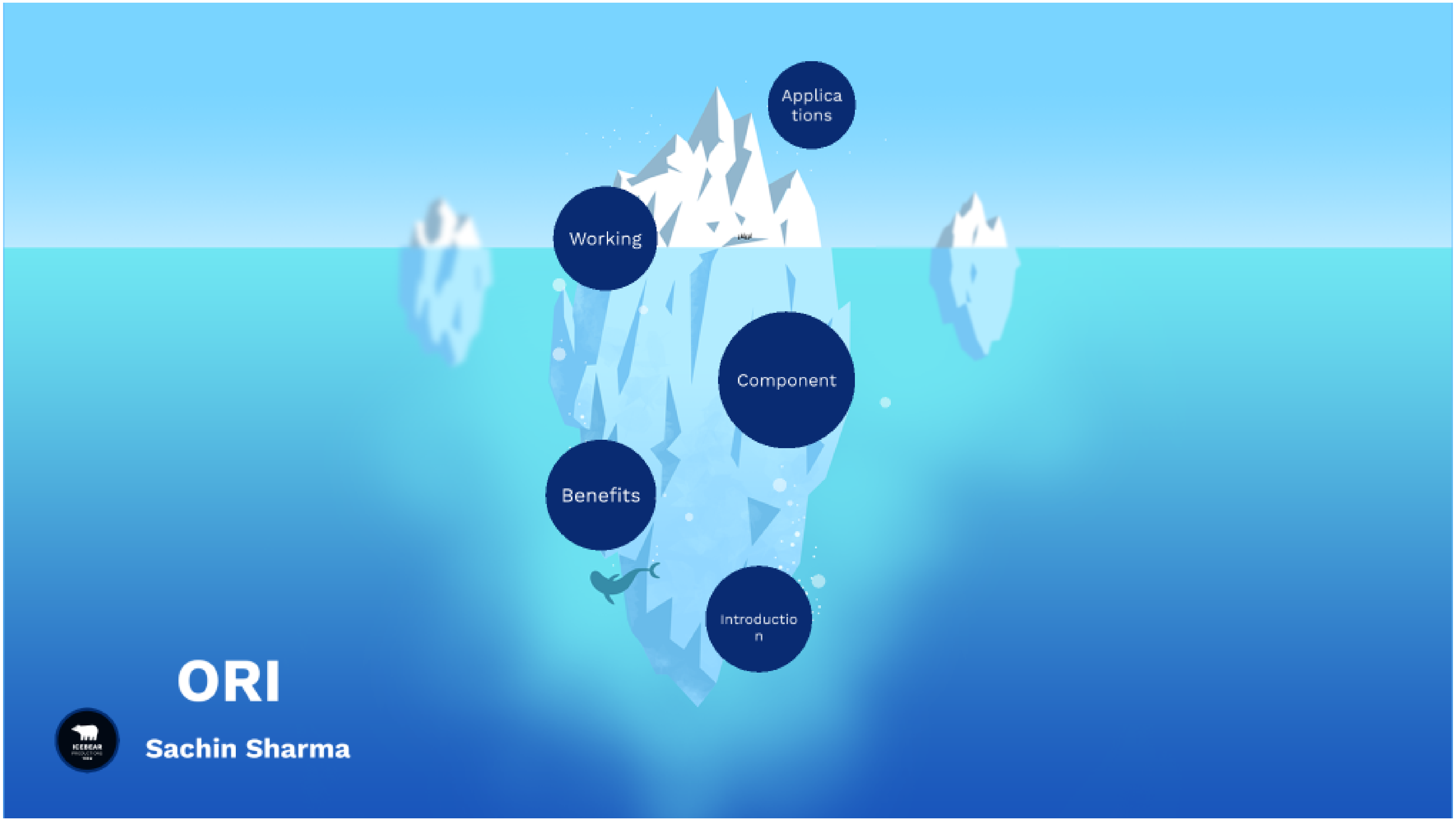
Introduction

ORI

Sachin Sharma



- CAS
- VCS
- History
- Background Fetch
- Deistributed Fetch



Applica
tions

Working

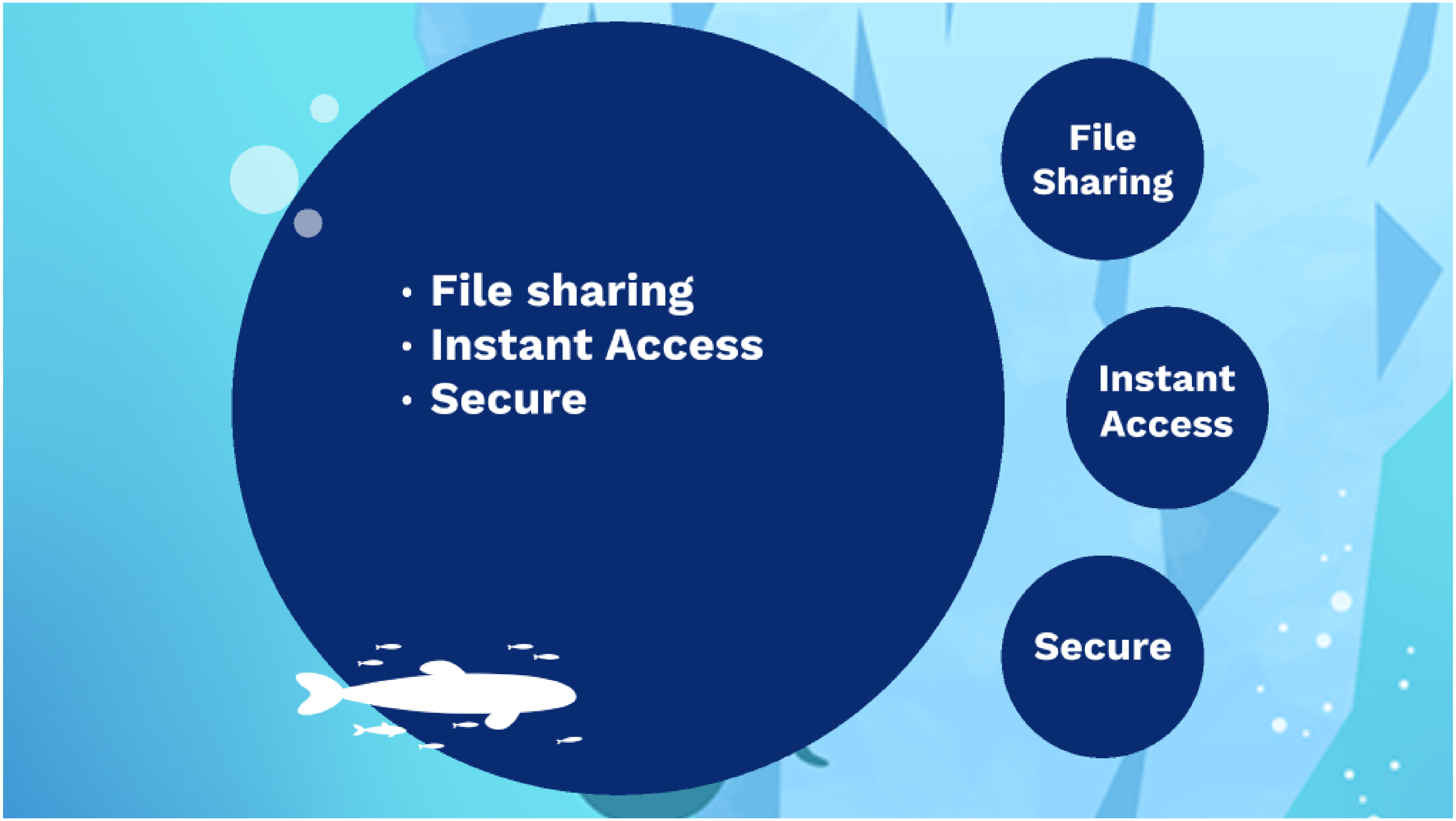
Component

Benefits

Introductio
n

ORI

Sachin Sharma

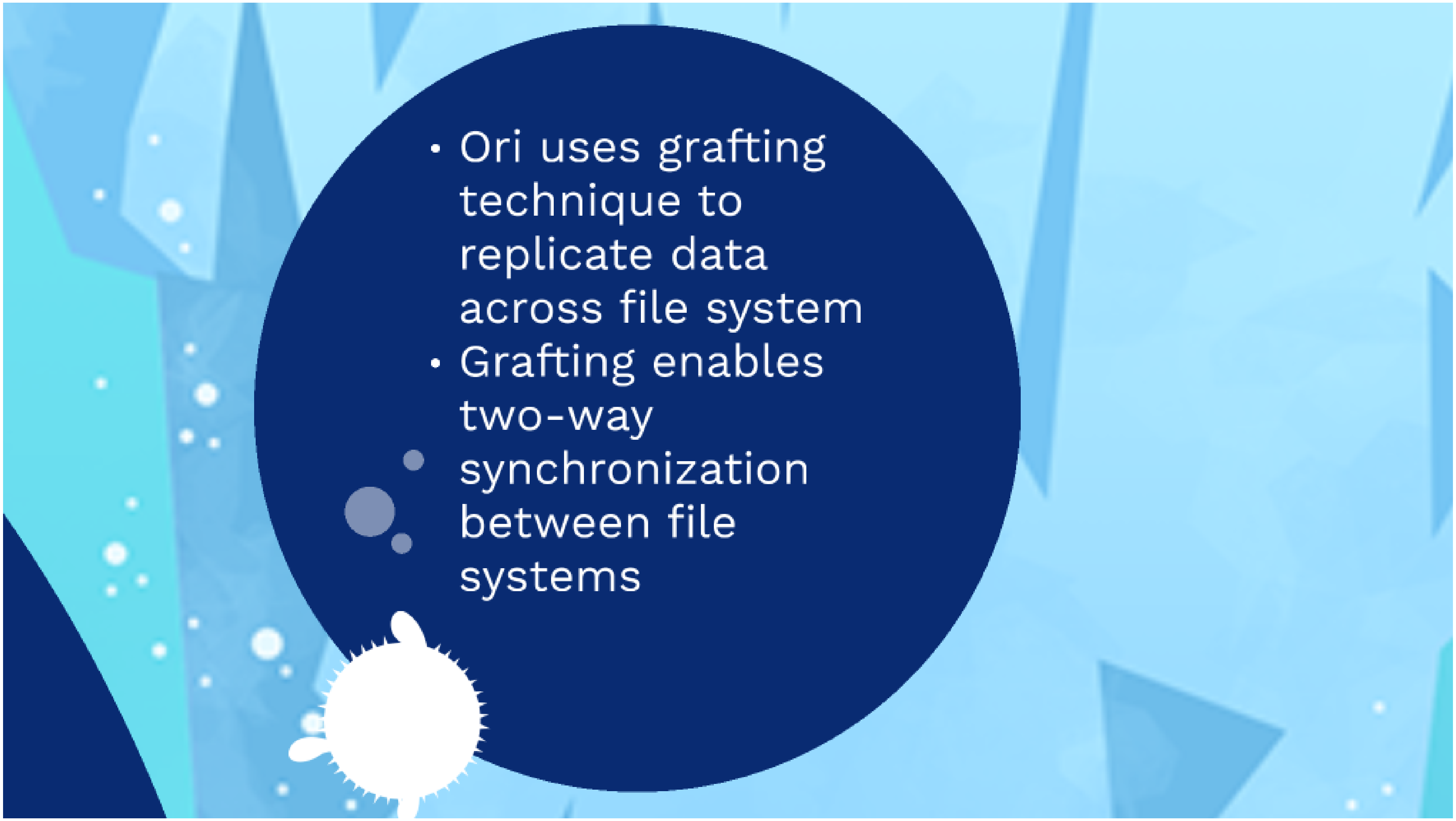


- File sharing
- Instant Access
- Secure

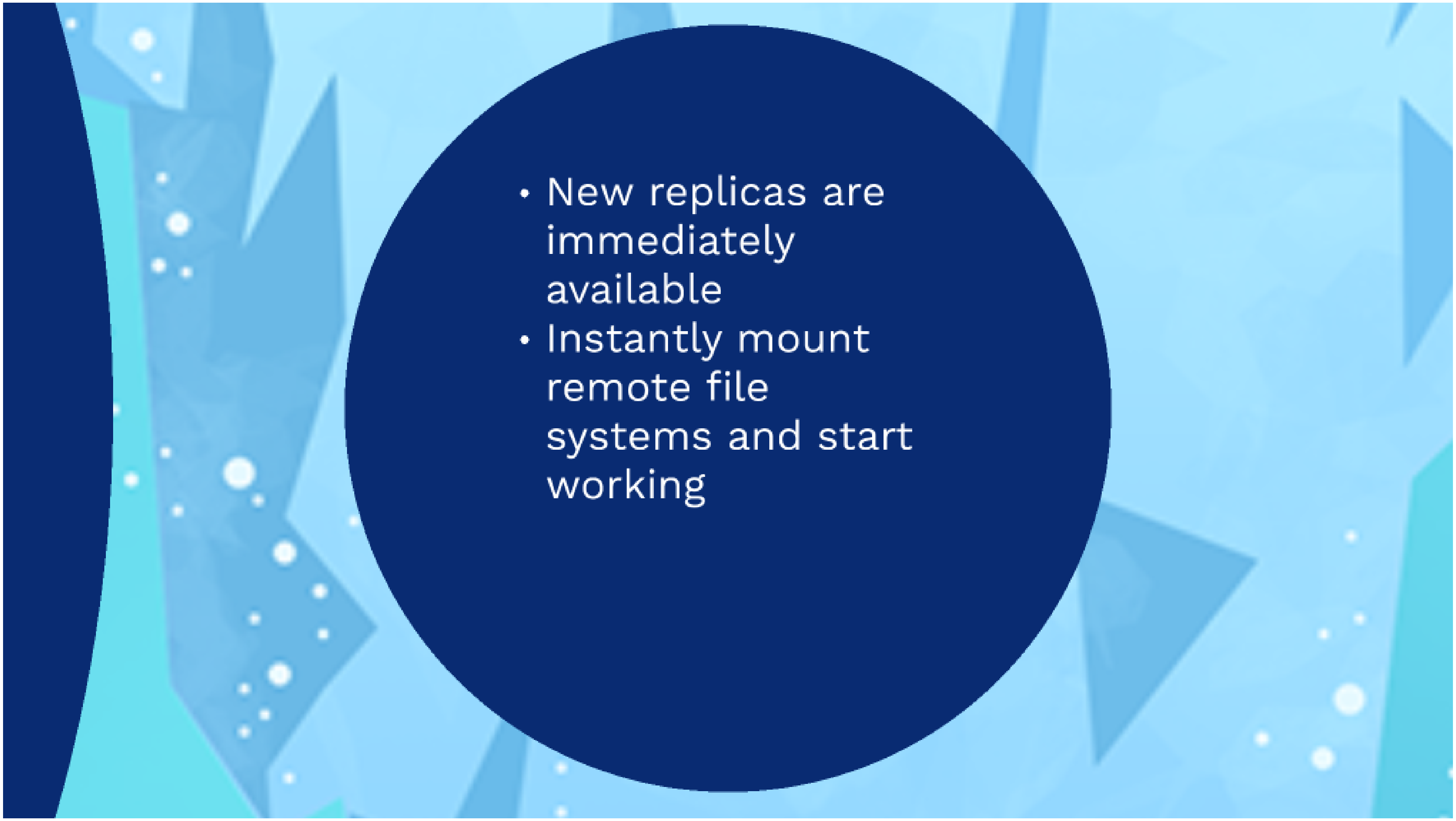
File Sharing

Instant Access

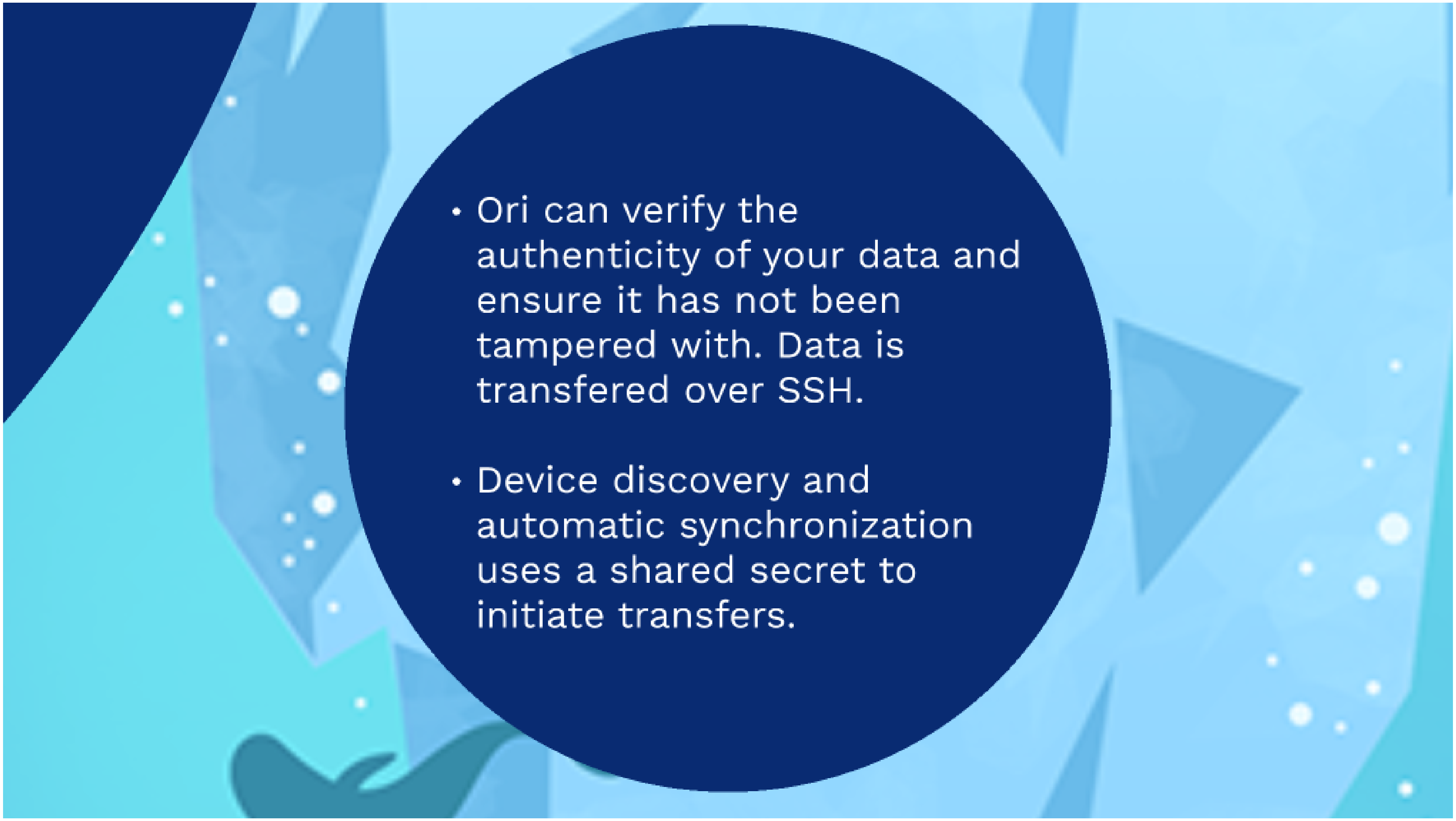
Secure



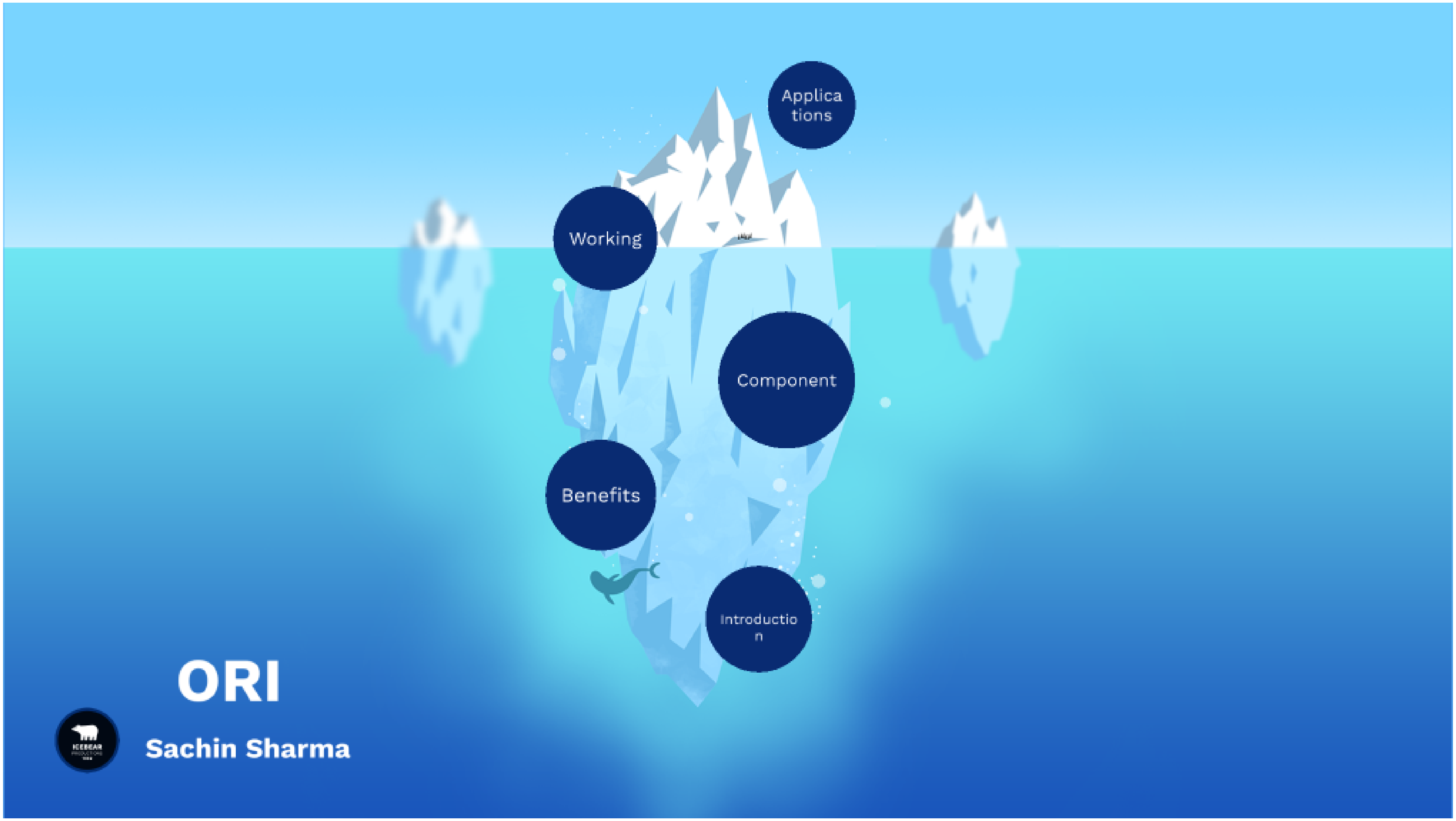
- Ori uses grafting technique to replicate data across file system
- Grafting enables two-way synchronization between file systems



- New replicas are immediately available
- Instantly mount remote file systems and start working



- Ori can verify the authenticity of your data and ensure it has not been tampered with. Data is transfered over SSH.
- Device discovery and automatic synchronization uses a shared secret to initiate transfers.



Applica tions

Working

Component

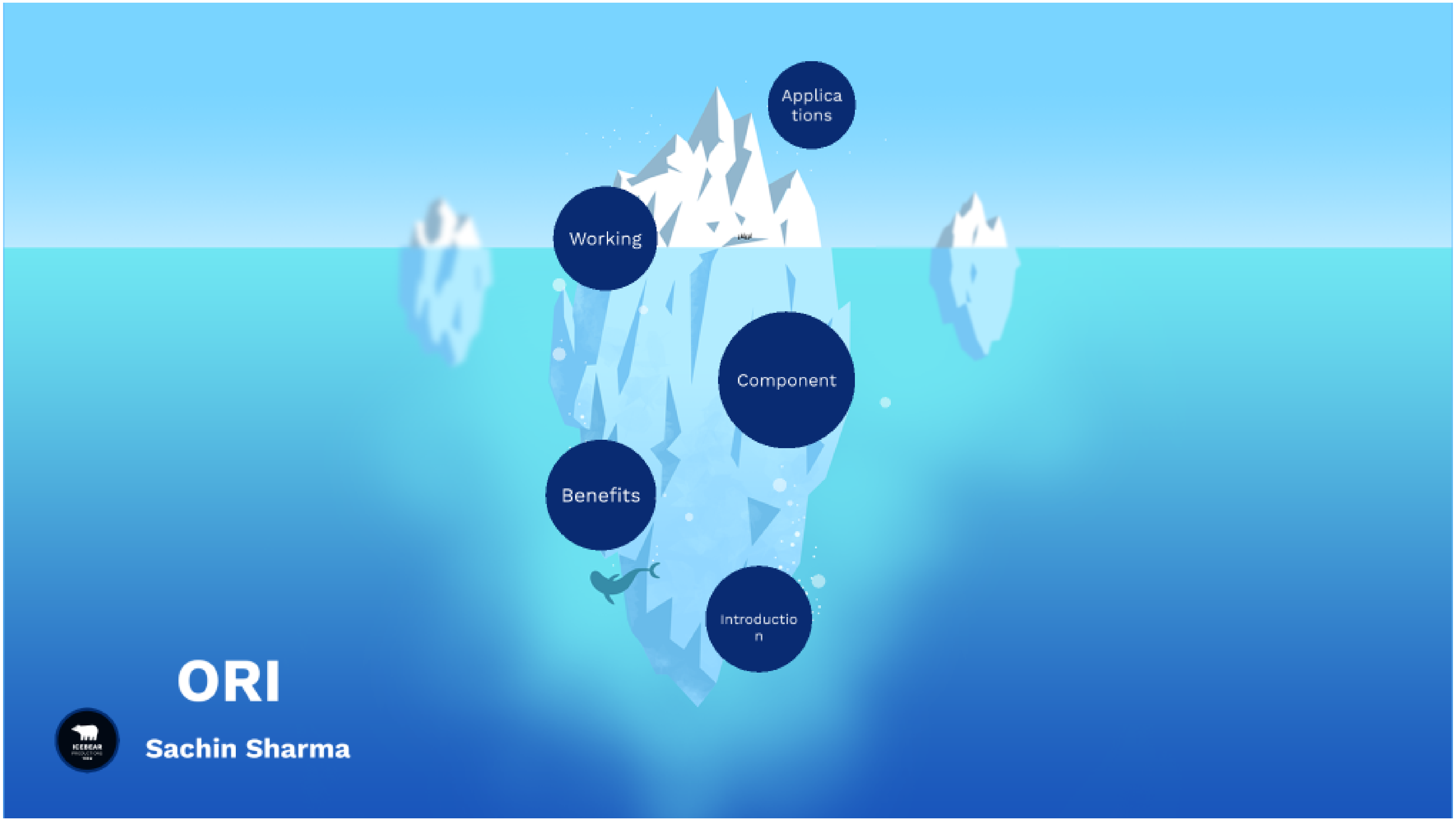
Benefits

Introductio n

ORI

Sachin Sharma

- Started by Standford University in 2013
- Currently under research
- Ori is a distributed file system built for offline operation.
- It offers control over synchronization operations and conflict resolution.
- It provides history through light weight snapshots



Applica
tions

Working

Component

Benefits

Introductio
n

ORI

Sachin Sharma