Security

Open windows prompt and connect to primary node

mongo --port 36000 -u user1 -p pass1

```
test-replicas:PRIMARY>
db.createUser({user:'u1',pwd:'pwd123',customData:{msg:'u1'},roles:['read']})
Successfully added user: {
	"user" : "u1",
	"customData" : {
		"msg" : "u1"
	},
	"roles" : [
		"read"
	]
}

test-replicas:PRIMARY> db.getUsers()
[
	{
		"_id" : "cars.u1",
		"userId" : UUID("23c39b0e-5351-41a7-95e7-195de1dcf0dc"),
		"user" : "u1",
		"db" : "cars",
		"customData" : {
			"msg" : "u1"
		},
		"roles" : [
			{
				"role" : "read",
				"db" : "cars"
			}
		],
		"mechanisms" : [
			"SCRAM-SHA-1",
			"SCRAM-SHA-256"
		]
	}
]
```

```
test-replicas:PRIMARY> db.getUser('u1')
{
	"_id" : "cars.u1",
	"userId" : UUID("23c39b0e-5351-41a7-95e7-195de1dcf0dc"),
	"user" : "u1",
	"db" : "cars",
	"customData" : {
		"msg" : "u1"
	},
	"roles" : [
		{
			"role" : "read",
			"db" : "cars"
		}
	],
	"mechanisms" : [
		"SCRAM-SHA-1",
		"SCRAM-SHA-256"
	]
}

test-replicas:PRIMARY> use owners
switched to db owners
test-replicas:PRIMARY>
db.createUser({user:'u2',pwd:'pwd123',roles:['readWrite',{db:'cars',role:'read'}]})
Successfully added user: {
	"user" : "u2",
	"roles" : [
		"readWrite",
		{
			"db" : "cars",
			"role" : "read"
		}
	]
}


test-replicas:PRIMARY> db.getUsers()
[
	{
		"_id" : "owners.u2",
		"userId" : UUID("197403bf-2167-4823-aade-68d388e1ce0a"),
		"user" : "u2",
		"db" : "owners",
```

```
        "roles" : [
                {
                        "role" : "readWrite",
                        "db" : "owners"
                },
                {
                        "role" : "read",
                        "db" : "cars"
                }
        ],
        "mechanisms" : [
                "SCRAM-SHA-1",
                "SCRAM-SHA-256"
        ]
    }
]


test-replicas:PRIMARY> use admin
switched to db admin
test-replicas:PRIMARY> show collections
system.users
system.version
```

```
test-replicas:PRIMARY> db.system.users.find().pretty()
{
	"_id" : "admin.user1",
	"userId" : UUID("5b05dad8-2bc1-4f4c-9f20-60946e57c269"),
	"user" : "user1",
	"db" : "admin",
	"credentials" : {
		"SCRAM-SHA-1" : {
			"iterationCount" : 10000,
			"salt" : "IfRU7HyDd2dVtpOaY4lqnw==",
			"storedKey" : "7rbM5vYKALKqQv/fEB59d/Xj5tM=",
			"serverKey" : "H9guWwz1JO6e0jOWfzU5ZEsSqh8="
		},
		"SCRAM-SHA-256" : {
			"iterationCount" : 15000,
			"salt" : "isVdMbmkTaAinGrrwO/943WV8+f/LasqFPDNkA==",
			"storedKey" : "2j8n4HEsJODV+HxoYx5RJCNS++0k0HtwS0SeSpwLhdE=",
			"serverKey" : "aAd9SL5AOSYMPMhcyj6j+8ovHd0/mRub2N5/dprXKDg="
		}
	},
	"roles" : [
		{
			"role" : "userAdminAnyDatabase",
			"db" : "admin"
		},
		{
			"role" : "readWriteAnyDatabase",
			"db" : "admin"
		}
	]
}
{
	"_id" : "cars.u1",
	"userId" : UUID("23c39b0e-5351-41a7-95e7-195de1dcf0dc"),
	"user" : "u1",
	"db" : "cars",
	"credentials" : {
		"SCRAM-SHA-1" : {
			"iterationCount" : 10000,
			"salt" : "M/w15YP7QfpyCjYwToyL5g==",
			"storedKey" : "uZIsCeHa+vyMZMC2H3BKYiU1pwI=",
			"serverKey" : "WuOe5SBKxq7dtIps/LjiK/x/QnQ="
		},
		"SCRAM-SHA-256" : {
```

```
                "iterationCount" : 15000,
                "salt" : "nEW4HwiO7Z31qo+nXLrnNLY16NH2lVNmMS3jRw==",
                "storedKey" : "AstEsFP+2q4jM+G2jUSh800NcJd6GTPjiAyuZHiK7Aw=",
                "serverKey" : "mV8BbLyU3o+xX+2dYDOfAaOWXSfwZVlitK0eHTP7IYU="
            }
        },
        "customData" : {
            "msg" : "u1"
        },
        "roles" : [
            {
                "role" : "read",
                "db" : "cars"
            }
        ]
}
{
        "_id" : "owners.u2",
        "userId" : UUID("197403bf-2167-4823-aade-68d388e1ce0a"),
        "user" : "u2",
        "db" : "owners",
        "credentials" : {
            "SCRAM-SHA-1" : {
                "iterationCount" : 10000,
                "salt" : "0orDnO5Qk4/qS1mUd0kzmQ==",
                "storedKey" : "EVpg4fnNcZ/rkBMBxj/pewEQuo0=",
                "serverKey" : "rG5E5pgg+Y3Uu2IY6vtpgdqI0uc="
            },
            "SCRAM-SHA-256" : {
                "iterationCount" : 15000,
                "salt" : "5CHCRs5Es0zMDuXkREqNLweDF8nYbUWHJj4mXQ==",
                "storedKey" : "uj7LuGybJv8NEzzdTwBfhfu3uqAX9TrVSbnO3esh/RQ=",
                "serverKey" : "kivaisBwpRqij54M+yFGDhYlh5mrsUnVDNaj1ZOkCWg="
            }
        },
        "roles" : [
            {
                "role" : "readWrite",
                "db" : "owners"
            },
            {
                "role" : "read",
                "db" : "cars"
            }
```

```
    ]
}
```

Connect by user u1 and validate authentication:

```
test-replicas:PRIMARY> db.auth({user:'u1',pwd:passwordPrompt()})
Enter password:
1

C:\mongo --port 36000 --authenticationDatabase cars -u u1 -p
MongoDB shell version v4.4.6
Enter password:
connecting to:
mongodb://127.0.0.1:36000/?authSource=cars&compressors=disabled&gssapiServiceName=m
ongodb
Implicit session: session { "id" : UUID("00b87b6a-3f27-4816-83dc-0b0a9a906d63") }
MongoDB server version: 4.4.6
test-replicas:PRIMARY> show dbs
cars  0.000GB


test-replicas:PRIMARY> use cars
switched to db cars

test-replicas:PRIMARY> show collections
list

test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Toyota" }


test-replicas:PRIMARY> db.list.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]


test-replicas:PRIMARY> db.stats()
{
    "db" : "cars",
    "collections" : 1,
    "views" : 0,
    "objects" : 1,
    "avgObjSize" : 35,
    "dataSize" : 35,
    "storageSize" : 20480,
```

```
        "indexes" : 1,
        "indexSize" : 20480,
        "totalSize" : 40960,
        "scaleFactor" : 1,
        "fsUsedSize" : 543914672128,
        "fsTotalSize" : 717687353344,
        "ok" : 1,
        "$clusterTime" : {
                "clusterTime" : Timestamp(1622818115, 1),
                "signature" : {
                        "hash" : BinData(0,"Kdh1WlAUDevRbbEBN8ZD3ZgwiY4="),
                        "keyId" : NumberLong("6969704999022493699")
                }
        },
        "operationTime" : Timestamp(1622818115, 1)
}
test-replicas:PRIMARY> db.list.stats()
{
        "ns" : "cars.list",
        "size" : 35,
        "count" : 1,
        "avgObjSize" : 35,
        "storageSize" : 20480,
        "freeStorageSize" : 0,
        "capped" : false,
        "wiredTiger" : {
                "metadata" : {
                        "formatVersion" : 1
                },
                "creationString" :
"access_pattern_hint=none,allocation_size=4KB,app_metadata=(formatVersion=1),assert=(com
mit_timestamp=none,durable_timestamp=none,read_timestamp=none,write_timestamp=off),
block_allocation=best,block_compressor=snappy,cache_resident=false,checksum=on,colgroups
=,collator=,columns=,dictionary=0,encryption=(keyid=,name=),exclusive=false,extractor=,forma
t=btree,huffman_key=,huffman_value=,ignore_in_memory_cache_size=false,immutable=false,i
mport=(enabled=false,file_metadata=,repair=false),internal_item_max=0,internal_key_max=0,i
nternal_key_truncate=true,internal_page_max=4KB,key_format=q,key_gap=10,leaf_item_max
=0,leaf_key_max=0,leaf_page_max=32KB,leaf_value_max=64MB,log=(enabled=false),lsm=(aut
o_throttle=true,bloom=true,bloom_bit_count=16,bloom_config=,bloom_hash_count=8,bloom
_oldest=false,chunk_count_limit=0,chunk_max=5GB,chunk_size=10MB,merge_custom=(prefix=
,start_generation=0,suffix=),merge_max=15,merge_min=0),memory_page_image_max=0,mem
ory_page_max=10m,os_cache_dirty_max=0,os_cache_max=0,prefix_compression=false,prefix_
compression_min=4,readonly=false,source=,split_deepen_min_child=0,split_deepen_per_child
=0,split_pct=90,tiered=(chunk_size=1GB,tiers=),tiered_storage=(auth_token=,bucket=,local_ret
```

ention=300,name=,object_target_size=10M),type=file,value_format=u,verbose=[],write_timestamp_usage=none",
"type" : "file",
"uri" : "statistics:table:collection-0--9043518699591075507",
"LSM" : {
"bloom filter false positives" : 0,
"bloom filter hits" : 0,
"bloom filter misses" : 0,
"bloom filter pages evicted from cache" : 0,
"bloom filter pages read into cache" : 0,
"bloom filters in the LSM tree" : 0,
"chunks in the LSM tree" : 0,
"highest merge generation in the LSM tree" : 0,
"queries that could have benefited from a Bloom filter that did not exist" : 0,
"total size of bloom filters" : 0,
"sleep for LSM checkpoint throttle" : 0,
"sleep for LSM merge throttle" : 0
},
"block-manager" : {
"allocations requiring file extension" : 0,
"blocks allocated" : 0,
"blocks freed" : 0,
"checkpoint size" : 4096,
"file allocation unit size" : 4096,
"file bytes available for reuse" : 0,
"file magic number" : 120897,
"file major version number" : 1,
"file size in bytes" : 20480,
"minor version number" : 0
},
"btree" : {
"btree checkpoint generation" : 39,
"btree clean tree checkpoint expiration time" : NumberLong("9223372036854775807"),
"column-store fixed-size leaf pages" : 0,
"column-store internal pages" : 0,
"column-store variable-size RLE encoded values" : 0,
"column-store variable-size deleted values" : 0,
"column-store variable-size leaf pages" : 0,
"fixed-record size" : 0,
"maximum internal page key size" : 368,
"maximum internal page size" : 4096,
"maximum leaf page key size" : 2867,
"maximum leaf page size" : 32768,

```
                "maximum leaf page value size" : 67108864,
                "maximum tree depth" : 0,
                "number of key/value pairs" : 0,
                "overflow pages" : 0,
                "pages rewritten by compaction" : 0,
                "row-store empty values" : 0,
                "row-store internal pages" : 0,
                "row-store leaf pages" : 0
            },
            "cache" : {
                "data source pages selected for eviction unable to be evicted" : 0,
                "eviction walk passes of a file" : 0,
                "bytes currently in the cache" : 423,
                "bytes dirty in the cache cumulative" : 0,
                "bytes read into cache" : 152,
                "bytes written from cache" : 0,
                "checkpoint blocked page eviction" : 0,
                "eviction walk target pages histogram - 0-9" : 0,
                "eviction walk target pages histogram - 10-31" : 0,
                "eviction walk target pages histogram - 128 and higher" : 0,
                "eviction walk target pages histogram - 32-63" : 0,
                "eviction walk target pages histogram - 64-128" : 0,
                "eviction walk target pages reduced due to history store cache pressure" : 0,
                "eviction walks abandoned" : 0,
                "eviction walks gave up because they restarted their walk twice" : 0,
                "eviction walks gave up because they saw too many pages and found no
candidates" : 0,
                "eviction walks gave up because they saw too many pages and found too few
candidates" : 0,
                "eviction walks reached end of tree" : 0,
                "eviction walks restarted" : 0,
                "eviction walks started from root of tree" : 0,
                "eviction walks started from saved location in tree" : 0,
                "hazard pointer blocked page eviction" : 0,
                "history store table insert calls" : 0,
                "history store table insert calls that returned restart" : 0,
                "history store table out-of-order resolved updates that lose their durable
timestamp" : 0,
                "history store table out-of-order updates that were fixed up by moving existing
records" : 0,
                "history store table out-of-order updates that were fixed up during insertion" : 0,
                "history store table reads" : 0,
                "history store table reads missed" : 0,
                "history store table reads requiring squashed modifies" : 0,
```

```
                "history store table truncation by rollback to stable to remove an unstable update" : 0,
                "history store table truncation by rollback to stable to remove an update" : 0,
                "history store table truncation to remove an update" : 0,
                "history store table truncation to remove range of updates due to key being removed from the data page during reconciliation" : 0,
                "history store table truncation to remove range of updates due to non timestamped update on data page" : 0,
                "history store table writes requiring squashed modifies" : 0,
                "in-memory page passed criteria to be split" : 0,
                "in-memory page splits" : 0,
                "internal pages evicted" : 0,
                "internal pages split during eviction" : 0,
                "leaf pages split during eviction" : 0,
                "modified pages evicted" : 0,
                "overflow pages read into cache" : 0,
                "page split during eviction deepened the tree" : 0,
                "page written requiring history store records" : 0,
                "pages read into cache" : 2,
                "pages read into cache after truncate" : 0,
                "pages read into cache after truncate in prepare state" : 0,
                "pages requested from the cache" : 1,
                "pages seen by eviction walk" : 0,
                "pages written from cache" : 0,
                "pages written requiring in-memory restoration" : 0,
                "tracked dirty bytes in the cache" : 0,
                "unmodified pages evicted" : 0
            },
            "cache_walk" : {
                "Average difference between current eviction generation when the page was last considered" : 0,
                "Average on-disk page image size seen" : 0,
                "Average time in cache for pages that have been visited by the eviction server" : 0,
                "Average time in cache for pages that have not been visited by the eviction server" : 0,
                "Clean pages currently in cache" : 0,
                "Current eviction generation" : 0,
                "Dirty pages currently in cache" : 0,
                "Entries in the root page" : 0,
                "Internal pages currently in cache" : 0,
                "Leaf pages currently in cache" : 0,
                "Maximum difference between current eviction generation when the page was last considered" : 0,
                "Maximum page size seen" : 0,
```

```
        "Minimum on-disk page image size seen" : 0,
        "Number of pages never visited by eviction server" : 0,
        "On-disk page image sizes smaller than a single allocation unit" : 0,
        "Pages created in memory and never written" : 0,
        "Pages currently queued for eviction" : 0,
        "Pages that could not be queued for eviction" : 0,
        "Refs skipped during cache traversal" : 0,
        "Size of the root page" : 0,
        "Total number of pages currently in cache" : 0
    },
    "checkpoint-cleanup" : {
        "pages added for eviction" : 0,
        "pages removed" : 0,
        "pages skipped during tree walk" : 0,
        "pages visited" : 0
    },
    "compression" : {
        "compressed page maximum internal page size prior to compression" : 4096,
        "compressed page maximum leaf page size prior to compression " : 131072,
        "compressed pages read" : 0,
        "compressed pages written" : 0,
        "page written failed to compress" : 0,
        "page written was too small to compress" : 0
    },
    "cursor" : {
        "bulk loaded cursor insert calls" : 0,
        "cache cursors reuse count" : 0,
        "close calls that result in cache" : 1,
        "create calls" : 1,
        "insert calls" : 0,
        "insert key and value bytes" : 0,
        "modify" : 0,
        "modify key and value bytes affected" : 0,
        "modify value bytes modified" : 0,
        "next calls" : 2,
        "operation restarted" : 0,
        "prev calls" : 0,
        "remove calls" : 0,
        "remove key bytes removed" : 0,
        "reserve calls" : 0,
        "reset calls" : 2,
        "search calls" : 0,
        "search history store calls" : 0,
        "search near calls" : 0,
```

```
        "truncate calls" : 0,
        "update calls" : 0,
        "update key and value bytes" : 0,
        "update value size change" : 0,
        "Total number of entries skipped by cursor next calls" : 0,
        "Total number of entries skipped by cursor prev calls" : 0,
        "Total number of entries skipped to position the history store cursor" : 0,
        "cursor next calls that skip due to a globally visible history store tombstone" : 0,
        "cursor next calls that skip greater than or equal to 100 entries" : 0,
        "cursor next calls that skip less than 100 entries" : 2,
        "cursor prev calls that skip due to a globally visible history store tombstone" : 0,
        "cursor prev calls that skip greater than or equal to 100 entries" : 0,
        "cursor prev calls that skip less than 100 entries" : 0,
        "open cursor count" : 0
    },
    "reconciliation" : {
        "dictionary matches" : 0,
        "internal page key bytes discarded using suffix compression" : 0,
        "internal page multi-block writes" : 0,
        "internal-page overflow keys" : 0,
        "leaf page key bytes discarded using prefix compression" : 0,
        "leaf page multi-block writes" : 0,
        "leaf-page overflow keys" : 0,
        "maximum blocks required for a page" : 0,
        "overflow values written" : 0,
        "page checksum matches" : 0,
        "pages written including at least one prepare" : 0,
        "pages written including at least one start timestamp" : 0,
        "records written including a prepare" : 0,
        "approximate byte size of timestamps in pages written" : 0,
        "approximate byte size of transaction IDs in pages written" : 0,
        "fast-path pages deleted" : 0,
        "page reconciliation calls" : 0,
        "page reconciliation calls for eviction" : 0,
        "pages deleted" : 0,
        "pages written including an aggregated newest start durable timestamp " : 0,
        "pages written including an aggregated newest stop durable timestamp " : 0,
        "pages written including an aggregated newest stop timestamp " : 0,
        "pages written including an aggregated newest stop transaction ID" : 0,
        "pages written including an aggregated newest transaction ID " : 0,
        "pages written including an aggregated oldest start timestamp " : 0,
        "pages written including an aggregated prepare" : 0,
        "pages written including at least one start durable timestamp" : 0,
        "pages written including at least one start transaction ID" : 0,
```

```
                "pages written including at least one stop durable timestamp" : 0,
                "pages written including at least one stop timestamp" : 0,
                "pages written including at least one stop transaction ID" : 0,
                "records written including a start durable timestamp" : 0,
                "records written including a start timestamp" : 0,
                "records written including a start transaction ID" : 0,
                "records written including a stop durable timestamp" : 0,
                "records written including a stop timestamp" : 0,
                "records written including a stop transaction ID" : 0
            },
            "session" : {
                "object compaction" : 0,
                "tiered storage local retention time (secs)" : 0,
                "tiered storage object size" : 0
            },
            "transaction" : {
                "race to read prepared update retry" : 0,
                "rollback to stable history store records with stop timestamps older than newer
records" : 0,
                "rollback to stable inconsistent checkpoint" : 0,
                "rollback to stable keys removed" : 0,
                "rollback to stable keys restored" : 0,
                "rollback to stable restored tombstones from history store" : 0,
                "rollback to stable restored updates from history store" : 0,
                "rollback to stable sweeping history store keys" : 0,
                "rollback to stable updates removed from history store" : 0,
                "transaction checkpoints due to obsolete pages" : 0,
                "update conflicts" : 0
            }
        },
        "nindexes" : 1,
        "indexBuilds" : [ ],
        "totalIndexSize" : 20480,
        "totalSize" : 40960,
        "indexSizes" : {
            "_id_" : 20480
        },
        "scaleFactor" : 1,
        "ok" : 1,
        "$clusterTime" : {
            "clusterTime" : Timestamp(1622818075, 1),
            "signature" : {
                "hash" : BinData(0,"5YmaL31TNsee1yotwxYHxNVKU7w="),
                "keyId" : NumberLong("6969704999022493699")
```

```
            }
        },
        "operationTime" : Timestamp(1622818075, 1)


test-replicas:PRIMARY> db.list.insert({_id:1,brand:'Lexus'})
WriteCommandError({
        "operationTime" : Timestamp(1622818185, 1),
        "ok" : 0,
        "errmsg" : "not authorized on cars to execute command { insert: \"list\", ordered: true, lsid:
{ id: UUID(\"00b87b6a-3f27-4816-83dc-0b0a9a906d63\") }, $clusterTime: { clusterTime:
Timestamp(1622818115, 1), signature: { hash: BinData(0,
29D8755A50140DEBD16DB10137C643DD9830898E), keyId: 6969704999022493699 } }, $db:
\"cars\" }",
        "code" : 13,
        "codeName" : "Unauthorized",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1622818185, 1),
                "signature" : {
                        "hash" : BinData(0,"Q8ZKk+fhz0NDLfaAkKrbz1CE01E="),
                        "keyId" : NumberLong("6969704999022493699")
                }
        }
})


test-replicas:PRIMARY> use owners
switched to db owners
test-replicas:PRIMARY> show collections
Warning: unable to run listCollections, attempting to approximate collection names by parsing
connectionStatus   (Note: because u1 is authorized to read cars database only)

test-replicas:PRIMARY> db.list.find()
Error: error: {
        "operationTime" : Timestamp(1622818305, 1),
        "ok" : 0,
        "errmsg" : "not authorized on owners to execute command { find: \"list\", filter: {}, lsid: { id:
UUID(\"00b87b6a-3f27-4816-83dc-0b0a9a906d63\") }, $clusterTime: { clusterTime:
Timestamp(1622818235, 1), signature: { hash: BinData(0,
7CD9D34A42BA0C4DB81F8710292DB79880CFD42E), keyId: 6969704999022493699 } }, $db:
\"owners\" }",
        "code" : 13,
        "codeName" : "Unauthorized",
        "$clusterTime" : {
```

```
        "clusterTime" : Timestamp(1622818305, 1),
        "signature" : {
            "hash" : BinData(0,"u0pb1Ozvrz2suyGVL/GdVIldYNU="),
            "keyId" : NumberLong("6969704999022493699")
        }
    }
}

test-replicas:PRIMARY> show collections
Warning: unable to run listCollections, attempting to approximate collection names by parsing
connectionStatus
test-replicas:PRIMARY> db.system.version.find()
Error: error: {
    "operationTime" : Timestamp(1622818375, 1),
    "ok" : 0,
    "errmsg" : "not authorized on admin to execute command { find: \"system.version\", filter:
{}, lsid: { id: UUID(\"00b87b6a-3f27-4816-83dc-0b0a9a906d63\") }, $clusterTime: { clusterTime:
Timestamp(1622818365, 1), signature: { hash: BinData(0,
93236C8F3586B92AF49BBE7926CDD50372FE9DFE), keyId: 6969704999022493699 } }, $db:
\"admin\" }",
    "code" : 13,
    "codeName" : "Unauthorized",
    "$clusterTime" : {
        "clusterTime" : Timestamp(1622818375, 1),
        "signature" : {
            "hash" : BinData(0,"kulYGEIZqYCNvUf6BJ9YzvVlAio="),
            "keyId" : NumberLong("6969704999022493699")
        }
    }
}


test-replicas:PRIMARY> quit()
C:\
```

C:**mongo --port 36000 --authenticationDatabase owners -u u2 -p pwd123**
MongoDB shell version v4.4.6
connecting to:
mongodb://127.0.0.1:36000/?authSource=owners&compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("4115c6f4-8f3c-4405-9c98-bf2504342843") }
MongoDB server version: 4.4.6
test-replicas:PRIMARY>

```
test-replicas:PRIMARY> show dbs
cars    0.000GB
owners  0.000GB


test-replicas:PRIMARY> use owners
switched to db owners

test-replicas:PRIMARY> show collections
list

test-replicas:PRIMARY> db.list.insert({_id:1,name:'Jason'})
WriteResult({ "nInserted" : 1 })


test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Timmy" }
{ "_id" : 1, "name" : "Jason" }


test-replicas:PRIMARY> db.list.createIndex({name:1})
{
      "createdCollectionAutomatically" : false,
      "numIndexesBefore" : 1,
      "numIndexesAfter" : 2,
      "commitQuorum" : "votingMembers",
      "ok" : 1,
      "$clusterTime" : {
            "clusterTime" : Timestamp(1622822219, 7),
            "signature" : {
                  "hash" : BinData(0,"rPs/+qKgddj3zpFX5g9in8DVVJo="),
                  "keyId" : NumberLong("6969704999022493699")
            }
      },
      "operationTime" : Timestamp(1622822219, 7)
```

```
}


test-replicas:PRIMARY> db.list.getIndexes()
[
	{
		"v" : 2,
		"key" : {
			"_id" : 1
		},
		"name" : "_id_"
	},
	{
		"v" : 2,
		"key" : {
			"name" : 1
		},
		"name" : "name_1"
	}
]

test-replicas:PRIMARY> db.list.dropIndex('name_1')
{
	"nIndexesWas" : 2,
	"ok" : 1,
	"$clusterTime" : {
		"clusterTime" : Timestamp(1622822376, 2),
		"signature" : {
			"hash" : BinData(0,"M1vimbGj6jJYggBHEEcsLGfWp/k="),
			"keyId" : NumberLong("6969704999022493699")
		}
	},
	"operationTime" : Timestamp(1622822376, 2)
}

test-replicas:PRIMARY> db.list.getIndexes()
[ { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_" } ]


test-replicas:PRIMARY> db.list.update({_id:1},{$set:{name:'Jill'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Timmy" }
```

```
{ "_id" : 1, "name" : "Jill" }
test-replicas:PRIMARY> db.list.deleteOne({_id:1})
{ "acknowledged" : true, "deletedCount" : 1 }

test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Timmy" }

test-replicas:PRIMARY> db.list.drop()
true

test-replicas:PRIMARY> show collections

test-replicas:PRIMARY> db.list.insert({_id:0,name:'Tom'})
WriteResult({ "nInserted" : 1 })


test-replicas:PRIMARY> use admin
switched to db admin

test-replicas:PRIMARY> db.shutdownServer()
Error: shutdownServer failed: {
      "operationTime" : Timestamp(1622822816, 1),
      "ok" : 0,
      "errmsg" : "not authorized on admin to execute command { shutdown: 1.0, lsid: { id:
UUID(\"4115c6f4-8f3c-4405-9c98-bf2504342843\") }, $clusterTime: { clusterTime:
Timestamp(1622822806, 1), signature: { hash: BinData(0,
B303D89658D595872C025B4A7E77B7FFB77CC47D), keyId: 6969704999022493699 } }, $db:
\"admin\" }",
      "code" : 13,
      "codeName" : "Unauthorized",
      "$clusterTime" : {
            "clusterTime" : Timestamp(1622822816, 1),
            "signature" : {
                  "hash" : BinData(0,"FuyEFFw2fEgFx23202r0qSg9it8="),
                  "keyId" : NumberLong("6969704999022493699")
            }
      }
} :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DB.prototype.shutdownServer@src/mongo/shell/db.js:426:19
@(shell):1:1

test-replicas:PRIMARY> quit()
```

```
C:\ >mongo -port 36002 -u user1 -p
MongoDB shell version v4.4.6
Enter password:
connecting to:
mongodb://127.0.0.1:36002/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("c39a1622-08eb-4795-a742-a9f3420989c6") }
MongoDB server version: 4.4.6
test-replicas:PRIMARY> show dbs
admin   0.000GB
cars    0.000GB
config  0.000GB
local   0.000GB
owners  0.000GB

test-replicas:PRIMARY> use admin
switched to db admin

test-replicas:PRIMARY> db.createUser({user:'indirect_root',pwd:'pwd123',roles:['userAdmin']})
Successfully added user: { "user" : "indirect_root", "roles" : [ "userAdmin" ] }

test-replicas:PRIMARY> db.getUsers()


test-replicas:PRIMARY> use admin
switched to db admin
test-replicas:PRIMARY> db.auth({user:'indirect_root',pwd:'pwd123'})
1
test-replicas:PRIMARY> show dbs
admin  0.000GB

test-replicas:PRIMARY>

test-replicas:PRIMARY> db.getUsers()

test-replicas:PRIMARY>
db.grantRolesToUser('indirect_root',['readAnyDatabase',{db:'cars',role:'readWrite'}])
uncaught exception: Error: not authorized on admin to execute command { grantRolesToUser:
"indirect_root", roles: [ "readAnyDatabase", { db: "cars", role: "readWrite" } ], writeConcern: {
w: "majority", wtimeout: 600000.0 }, lsid: { id: UUID("c39a1622-08eb-4795-a742-
a9f3420989c6") }, $clusterTime: { clusterTime: Timestamp(1622834491, 1), signature: { hash:
BinData(0, E1CB75D055E00CC16467B5C1FB7B07E7C46F547A), keyId: 6969704999022493699 }
}, $db: "admin" } :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DB.prototype.grantRolesToUser@src/mongo/shell/db.js:1613:15
```

@(shell):1:1

test-replicas:PRIMARY>
db.grantRolesToUser('indirect_root',['readAnyDatabase','userAdminAnyDatabase',{db:'cars',role:'readWrite'}])
uncaught exception: Error: **not authorized on admin to execute command** { grantRolesToUser: "indirect_root", roles: ["readAnyDatabase", "userAdminAnyDatabase", { db: "cars", role: "readWrite" }], writeConcern: { w: "majority", wtimeout: 600000.0 }, lsid: { id: UUID("c39a1622-08eb-4795-a742-a9f3420989c6") }, $clusterTime: { clusterTime: Timestamp(1622834621, 1), signature: { hash: BinData(0, 6FAC5E8CCB449D454F799A31FDFD7A0EA9583CD8), keyId: 6969704999022493699 } }, $db: "admin" } :
_getErrorWithCode@src/mongo/shell/utils.js:25:13
DB.prototype.grantRolesToUser@src/mongo/shell/db.js:1613:15
@(shell):1:1

test-replicas:PRIMARY> **db.grantRolesToUser('indirect_root',['readAnyDatabase'])**

test-replicas:PRIMARY> **db.grantRolesToUser('indirect_root',[{db:'cars',role:'readWrite'}])**

test-replicas:PRIMARY> db.auth({user:'indirect_root',pwd:'pwd123'})
1
test-replicas:PRIMARY> use cars
switched to db cars
test-replicas:PRIMARY> show collections
list
test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Toyota" }

test-replicas:PRIMARY> db.list.insert({_id:1,brand:'Nissan'})
WriteResult({ "nInserted" : 1 })
test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Toyota" }
{ "_id" : 1, "brand" : "Nissan" }

test-replicas:PRIMARY> use owners
switched to db owners
test-replicas:PRIMARY> show collections
list
test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Tom" }

test-replicas:PRIMARY> use owners
switched to db owners

```
test-replicas:PRIMARY> db.getUsers()

test-replicas:PRIMARY> db.updateUser('u2',{customData: {msg:'updated U2'},
roles:['dbOwner']})

test-replicas:PRIMARY> db.getUsers()
[
	{
		"_id" : "owners.u2",
		"userId" : UUID("197403bf-2167-4823-aade-68d388e1ce0a"),
		"user" : "u2",
		"db" : "owners",
		"roles" : [
			{
				"role" : "dbOwner",
				"db" : "owners"
			}
		],
		"customData" : {
			"msg" : "updated U2"
		},
		"mechanisms" : [
			"SCRAM-SHA-1",
			"SCRAM-SHA-256"
		]
	}
]


test-replicas:PRIMARY> db.createUser({user:'u3',pwd:'pwd123',roles:['dbAdmin']})
Successfully added user: { "user" : "u3", "roles" : [ "dbAdmin" ] }

test-replicas:PRIMARY> db.getUsers()
[
	{
		"_id" : "owners.u2",
		"userId" : UUID("197403bf-2167-4823-aade-68d388e1ce0a"),
		"user" : "u2",
		"db" : "owners",
		"roles" : [
			{
				"role" : "dbOwner",
				"db" : "owners"
			}
```

```
        ],
        "customData" : {
            "msg" : "updated U2"
        },
        "mechanisms" : [
            "SCRAM-SHA-1",
            "SCRAM-SHA-256"
        ]
    },
    {
        "_id" : "owners.u3",
        "userId" : UUID("30b29394-b0bf-4b75-8432-1719e6c2fe4a"),
        "user" : "u3",
        "db" : "owners",
        "roles" : [
            {
                "role" : "dbAdmin",
                "db" : "owners"
            }
        ],
        "mechanisms" : [
            "SCRAM-SHA-1",
            "SCRAM-SHA-256"
        ]
    }
]
```

To create custom role to grant read only permission on a collection :

test-replicas:PRIMARY>

db.createRole({role:'read_list_only',privileges:[{resource:{db:'cars',collection:'list'},actions:['find']}],roles:[]})

```
{
    "role" : "read_list_only",
    "privileges" : [
        {
            "resource" : {
                "db" : "cars",
                "collection" : "list"
            },
            "actions" : [
                "find"
            ]
        }
    ],
```

```
    "roles" : [ ]
}


test-replicas:PRIMARY> db.getRoles()
[
    {
        "role" : "read_list_only",
        "db" : "cars",
        "isBuiltin" : false,
        "roles" : [ ],
        "inheritedRoles" : [ ]
    }
]
```

Create new user and grant newly create role:

```
test-replicas:PRIMARY>
db.createUser({user:'read_list_user',pwd:'pwd123',roles:['read_list_only']})
Successfully added user: { "user" : "read_list_user", "roles" : [ "read_list_only" ] }


test-replicas:PRIMARY> db.getUser('read_list_user')
{
    "_id" : "cars.read_list_user",
    "userId" : UUID("88834688-ec92-4211-9557-b0b9c7337816"),
    "user" : "read_list_user",
    "db" : "cars",
    "roles" : [
        {
            "role" : "read_list_only",
            "db" : "cars"
        }
    ],
    "mechanisms" : [
        "SCRAM-SHA-1",
        "SCRAM-SHA-256"
    ]
}



test-replicas:PRIMARY> db.auth({user:'read_list_user',pwd:'pwd123'})
1
```

```
test-replicas:PRIMARY> show collections
list
test-replicas:PRIMARY> db.list.find()
{ "_id" : 0, "name" : "Toyota" }
{ "_id" : 1, "brand" : "Nissan" }
test-replicas:PRIMARY> db.list.insert({})
WriteCommandError({
	"operationTime" : Timestamp(1622839832, 1),
	"ok" : 0,
	"errmsg" : "not authorized on cars to execute command { insert: \"list\", ordered: true, lsid:
{ id: UUID(\"b37f9f52-90a5-46a3-9b6e-87d7794c24f4\") }, $clusterTime: { clusterTime:
Timestamp(1622839822, 1), signature: { hash: BinData(0,
D36C452FF1F16BBBEC8F5E70350E2A2B9541C2A3), keyId: 6969704999022493699 } }, $db:
\"cars\" }",
	"code" : 13,
	"codeName" : "Unauthorized",
	"$clusterTime" : {
		"clusterTime" : Timestamp(1622839832, 1),
		"signature" : {
			"hash" : BinData(0,"4JVjnJe3sbvrFbngZFh5W2bPuec="),
			"keyId" : NumberLong("6969704999022493699")
		}
	}
})
```

```
mongo -port 36000 -u user1 -p
MongoDB shell version v4.4.6
Enter password:
connecting to:
mongodb://127.0.0.1:36000/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("7c0a1ad4-d272-48d5-97a3-1783498d96aa") }
MongoDB server version: 4.4.6
test-replicas:PRIMARY> use admin
switched to db admin
test-replicas:PRIMARY>
db.createRole({role:'changePwdForAll',roles:[],privileges:[{resource:{db:'',collection:''},actions:['
changePassword']}]})
{
	"role" : "changePwdForAll",
	"roles" : [ ],
	"privileges" : [
		{
			"resource" : {
				"db" : "",
				"collection" : ""
			},
			"actions" : [
				"changePassword"
			]
		}
	]
}


test-replicas:PRIMARY> show collections
system.roles
system.users
system.version
test-replicas:PRIMARY> db.system.roles.find()
{ "_id" : "cars.read_list_only", "role" : "read_list_only", "db" : "cars", "privileges" : [ { "resource"
: { "db" : "cars", "collection" : "list" }, "actions" : [ "find" ] } ], "roles" : [ ] }
{ "_id" : "admin.changePwdForAll", "role" : "changePwdForAll", "db" : "admin", "privileges" : [ {
"resource" : { "db" : "", "collection" : "" }, "actions" : [ "changePassword" ] } ], "roles" : [ ] }
```

```
test-replicas:PRIMARY> db
admin

test-replicas:PRIMARY>
db.createUser({user:'pwdManager',pwd:'pwd123',roles:['changePwdForAll']})
Successfully added user: { "user" : "pwdManager", "roles" : [ "changePwdForAll" ] }
```