Are Mongodb Databases Secure?

Martin Rupp
SCIENTIFIC AND COMPUTER DEVELOPMENT SCD LTD

In that article, we aim to look at the data about the security of MongoDB database - a very popular no-SQL database - which is widely used all over the world.

MongoDB has suffered from many bad stories where the databases of many mongoDB users were hacked because of what is known as a ‘no-security configuration’ by default. Here we wish to look at the facts and investigate if Mongo Database is secure or not.

Some background about MongoDB and MongoDB Inc

MongoDB started to be developed in 2007 by a company named 10gen Software. Shortly after 10gen Software changed its name to MongoDb Inc.

MongoDB falls into the category of no-SQL, non-transactional databases. It uses JSON to create the database structure and records. The records are collections of documents and their format is left to the responsibility of the developers so it differs quite from the SQL model where records follow a strict and well-defined structure,

MongoDB is considered traditionally "simpler" to use than SQL databases such as MySQL or Microsoft SQL. Therefore it has attracted a large audience of a "new generation" of developers not wishing to learn or use the SQL syntax.

From the corporate website of MongoDB, at the time of writing the article[1]: “*MongoDB has more than 13,000 customers in more than 100 countries. The MongoDB database platform has been downloaded over 60 million times and there have been more than 1 million MongoDB University registrations.*”

[1] https://www.mongodb.com/company

MongoDB Inc. has more than 1,000 employees in around 20 countries. This means that we are dealing with an established and serious company. The business model of MongoDB is mass market and based on paid hosted services but the database itself is open source and free to download and use.

MongoDB which is listed on the NASDAQ stock market as MDB posted revenues of $99.37 million for the quarter ended July 2019.

Some of the hacks involved with Mongo database

One of the most recent hacks discovered involved a man named Bob Diachenko, a security researcher, whose passion is to hunt for misconfigured Mongo Databases.

One of the main tools used by Diachenko to search vulnerable Mongo databases is Shodan.

Shodan is a paid search engine that can access information from the "Internet of Things", routers, smart TVs, webcam, or other such devices. It monitors usually metadata or service banners. Users who have bought a subscription to Shodan can access data from traffic lights, as well as industrial water control systems, nuclear plant control systems, synchrotrons, etc.

On 23 April 2019, Diachenko discovered a database, using Shodan, which had more than 275 million records containing private details about Indian citizens!

Records included the individual's name, gender, email address and also employment history, employer, salary, and mobile phone numbers.

```
"_id" : ObjectId("5cbf0fd076da82177d173910"),
"Course(2nd Highest Education)" : NaN,
"Name" : "[redacted]",
"Current Location" : [redacted],
"Industry" : "Catering/Food Services/Restaurant, Hotel/Travel/Tourism/Airlines/Hospitality",
"Institute(Highest Education)" : "Others",
"Specialization(2nd Highest Education)" : NaN,
"Resume Id" : "[redacted]",
"Specialization(Highest Education)" : "Other B.A.",
"Current Employer" : "[redacted]",
"Mobile No" : "9[redacted]",
"Preferred Location" : "Anywhere in India",
"Course(Highest Education)" : "B.A.",
"Key Skills" : "GPs, PNL, stocks, guest care, staff developement and training, IT skills, gener
"Previous Employer" : "[redacted]",
"Date of Birth" : "1986-04-14 00:00:00",
"Address" : "[redacted]",
"Area of Specialization" : "Food & Beverage, Guest Relation, Restaurant",
"Institute(2nd Highest Education)" : NaN,
"Resume Title" : "f&b operational expert and has got graduation from london",
"Current Salary" : "6,00,000 annually",
"Email Id" : "[redacted]@yahoo.com",
"Gender" : "Male",
"Level" : "Others",
"Functional Area" : "Hotel/Restaurant",
"Alternate Number" : "[redacted]"
```


Diachenko reported this to the Indian authorities but shortly after, the database was erased and replaced with coordinates and instructions on how to pay a ransom to get the data back … so others than Diachenko were monitoring the vulnerable databases in Shodan - of more realistically, they knew about that database by reading Diachenko's blog.

As Diachenko explained: "*I have previously reported that the lack of authentication allowed the installation of malware or ransomware on the MongoDB servers.*"

Previously the same Diachenko discovered another unprotected database.

The database was owned by Verifications IO enterprise email validation service. That company was involved in a bulk email list. A total of 808,539,939 records - email addresses with details - had been exposed.

A security researcher from Microsoft, Niall Merrigan claimed that - in 2017- more than 27,000 Mongo databases had already been seized by ransomware. The attack came as a sudden wave, targeting unsecured Mongo databases all over the world.

The scheme of a typical MongoDB attack consists of identifying unsecured databases, copying or ciphering the data then replacing them by ciphered or voiding data or simply

deleting the databases. Finally, the attacker asks for money - usually via Bitcoins - to restore the data.

The official position of MongoDB Inc

Following the waves of attacks, MongoDB communicated about the fact that their databases are 'opened' by default. This means these databases - by default - can be reached via the internet without no passwords! Here is the integral statement from MongoDB Inc.:

"*We respect that our innovative users ask for freedom to set their course and we do what we can to keep that possible, while at the same time answering to the standards of care expected in safety-conscious measured operations. That balance has meant offering both a frictionless experience for developers and a thorough configuration guide to complex controls like authentication. We believe setting localhost by default puts users in a mode where they have to make a conscious decision about their appropriate path to network safety.*"

As we can see, The official position of MongoDB Inc., at least at the time of the attacks and discoveries of leaked databases by Diachenko, consisted in assuming the politics of releasing the databases with "zero security" by default.

The security mechanisms of the Mongo database

Now let us look at the core of the problem… Are Mongo Databases insecure or not? Mongo Databases are provided with a great range of features:

- Authentication via LDAP, the Lightweight Directory Access Protocol, x509 or scram ;
- Authorization mechanisms;
- Encryption of the data;
- Auditing;
- Governance.

Note that MongoDB is not common criterion evaluated unlike, for instance, Microsoft SQL Server 2016

A security “checklist” has been published by MongoDB and displays everything that a MongoDB administrator must do before releasing the database. We list here these requirements exactly as they are displayed on the mongoDB website:

- Enable Access Control and Enforce Authentication;
- Configure Role-Based Access Control;
- Encrypt Communication;
- Encrypt and Protect Data;
- Limit Network Exposure;
- Audit System Activity;
- Run MongoDB with a Dedicated User;
- Run MongoDB with Secure Configuration Options;
- Request a Security Technical Implementation Guide (where applicable);
- Consider Security Standards Compliance.

It seems therefore that MongoDB has strong security in place …

BUT ...

MySQL, Microsoft SQL Server, PostgreSQL, and other equivalent relational databases almost always default to local installation (connections are only possible from 127.0.0.1) and to some form of authorization (users and passwords are needed).

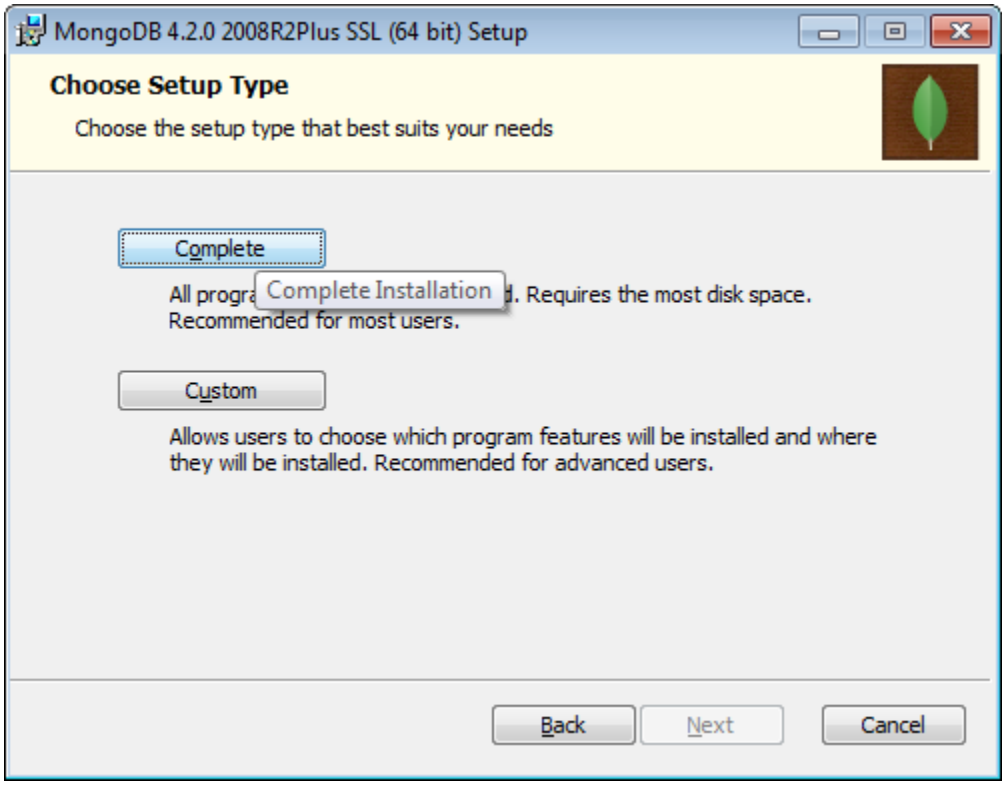
On the contrary, many MongoDB databases (all the versions except the newest ones) are exposed to the internet and don't require credentials by default.

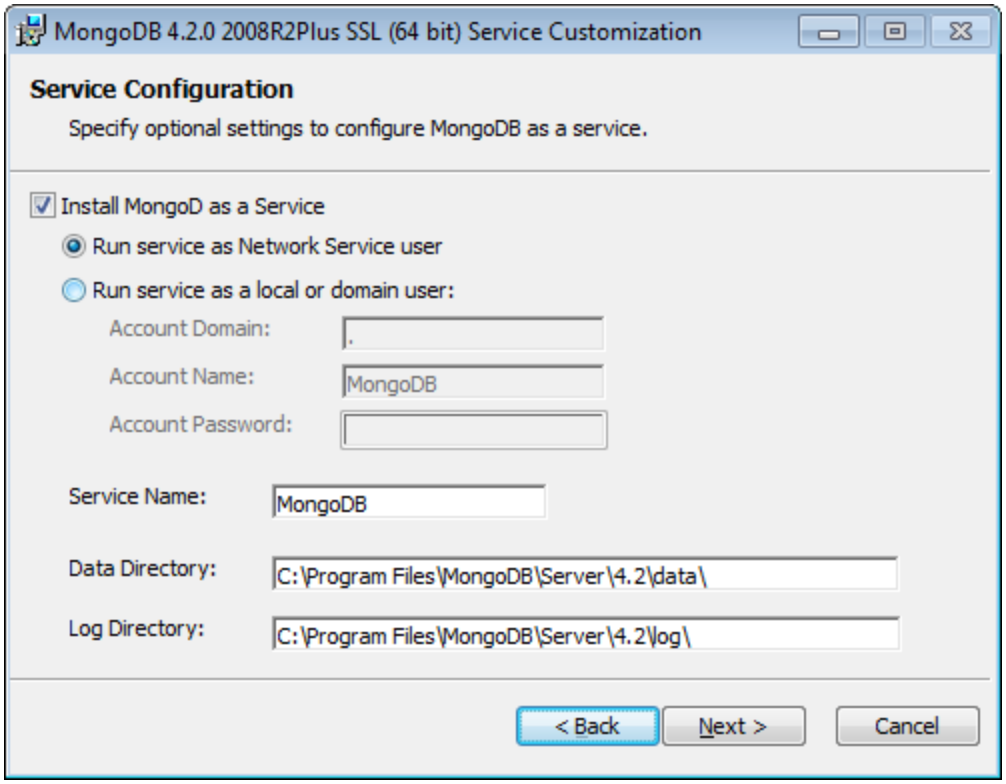
Security researchers advised that the configuration file of a new MongoDB database is immediately changed to restrict connections to localhost or a private secure. subnetwork.

Some Tests

We test the security of the database on Windows by installing the latest release of the Mongo Database for Windows x64:
`mongodb-win32-x86_64-2012plus-4.2.0-signed.msi`.

We also chose to download the MongoDB Compass tool.



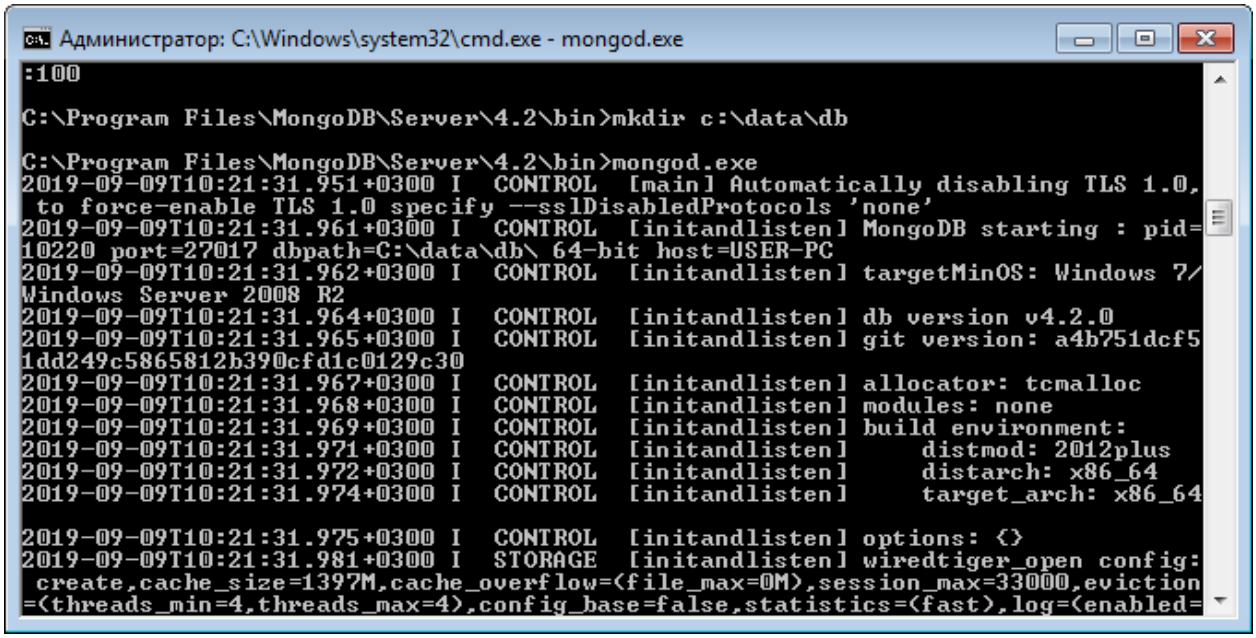


Looking at the configuration file, `mongod.cfg`, we see that Mongo databases are restricted to the localhost connections by default.

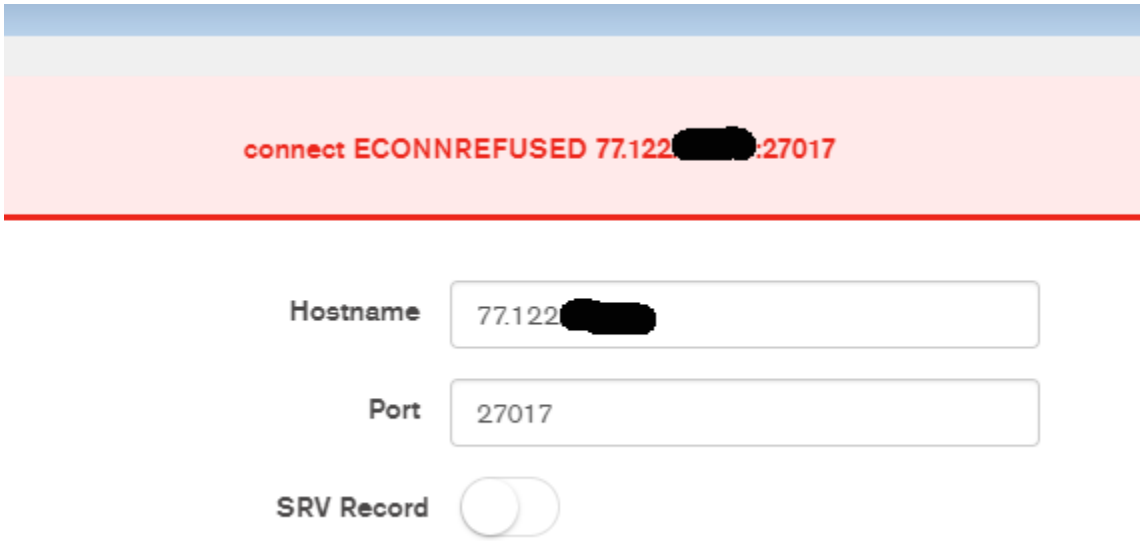
```
# where to write logging data.
systemLog:
  destination: file
  logAppend: true
  path:  %MONGO_LOG_PATH%\mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 127.0.0.1
```

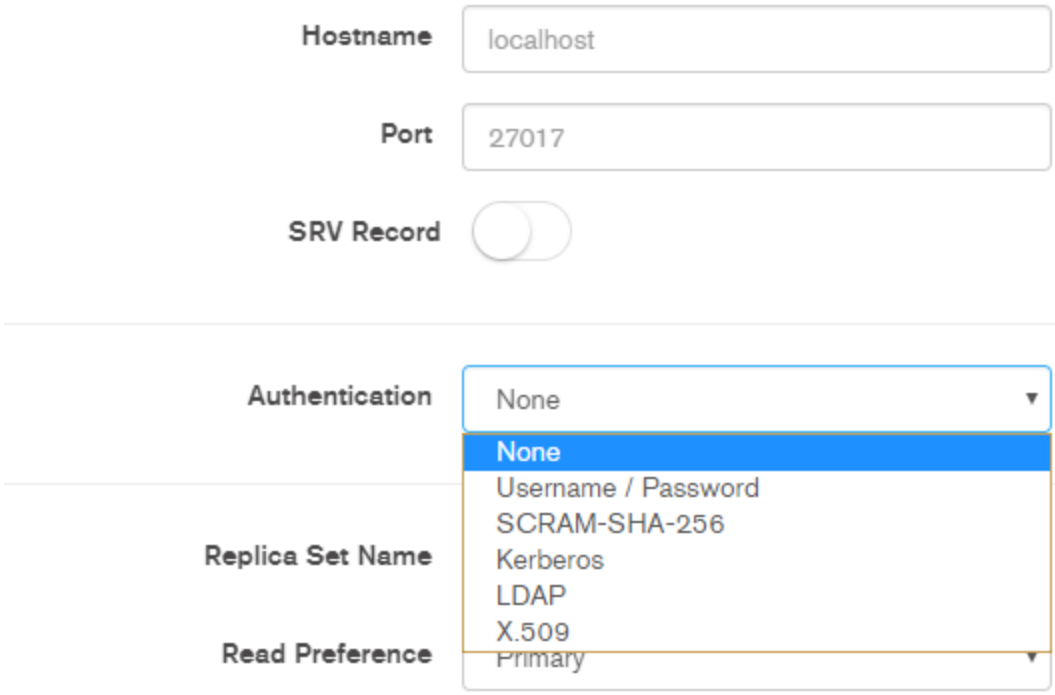
We start the server manually



We test that we can connect to 127.0.0.1 but that other interfaces are blocked.



Indeed we check that an external connection is refused.
The newest versions of MongoDB restrict access to localhost by default.
Anyway, we were able to connect without a password.



We can choose different mechanisms for access control but by default, the method is set to "none".

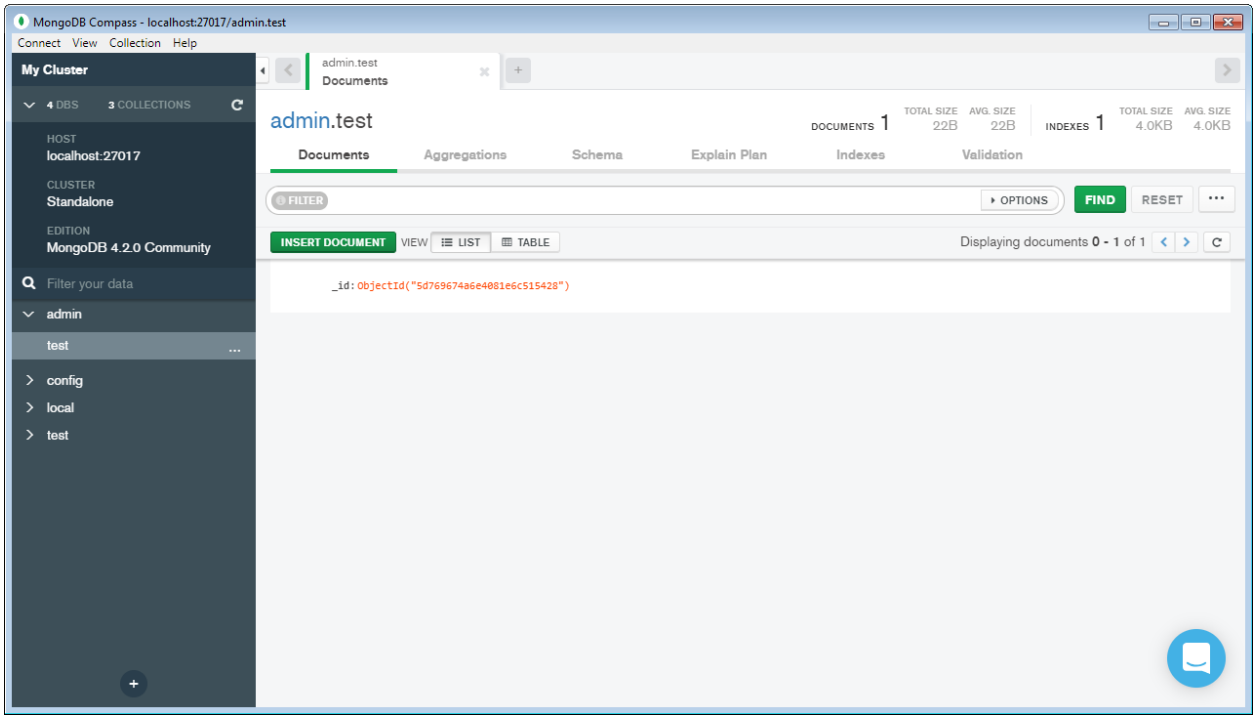
We test user creation. We see - using the Mongo Shell *Mongo.exe* - that we can specify the role of the new users and specify their level of access and rights.

```
> db.createUser(
... {user: "Guru99",
... pwd: "password",
...
... roles:[{role: "userAdminAnyDatabase" , db:"admin"}]})
Successfully added user: {
        "user" : "Guru99",
        "roles" : [
                {
                        "role" : "userAdminAnyDatabase",
                        "db" : "admin"
                }
        ]
}
>
```

We create a user with read-only access in the Admin database

```
> use  admin
switched to db admin
>
>
>
>
> db.createUser(
... {user: "test2",
... pwd: "password",
...
... roles:[{role: "read" , db:"admin"}]})
Successfully added user: {
        "user" : "test2",
        "roles" : [
                {
                        "role" : "read",
                        "db" : "admin"
                }
        ]
}
>
```

But the user has - by default - write access in the database “test” …



This means there is a problem with the role-based access system of MongoDB.

Additionally, data is not ciphered by TLS by default so data will transit in clear without explicit configuration. This is also clearly a security issue.

Some conclusions

While the most recent versions of MongoDB restrict the connection by default to localhost, there are still no passwords, role-based access issues, and no data ciphering by default. Certainly MongoDB databases -being no-sql- are not vulnerable to SQL injections but recent history shows that they lack strong security and they must be used only by skilled administrators who will respect scrupulously all the conditions to make them secure. By default a MongoDB database isn't secure and this seems to be clearly a strategic choice from MongoDB Inc.