How to Utilize DxOdyssey Tunnels

Accessing MongoDB on an Ubuntu Node

Introduction

As a fully functioning, cloud-ready document database, MongoDB excels at the decentralized data applications of the modern enterprise. Accessing it on an Ubuntu node is a credible means of maximizing the sort of innovation both of these open source platforms encourage. With its intelligent data distributions successfully impacting aspects of regulatory compliance, governance protocols, hybrid clouds, integration efforts, high availability, and sharding capabilities, MongoDB scales horizontally to yield abundant business value in numerous ways.

The Challenge

Nevertheless, the heterogeneous application and computing demands MongoDB supports also present several unavoidable cyber security challenges that, if left unmet, quickly nullify the gains of today's distributed IT landscape. These challenges include:

Perimeter Security Limitations

» MongoDB allows organizations to globally position data as needed to adhere to regulations and data governance policies. Although the platform enables the rapid distribution of data resources for these reasons and others, organizations are still tasked with sufficiently securing them. Typical perimeter security methods are costly, impractical, and difficult to maintain. VPN's increase attack surfaces and leave organizations vulnerable to lateral movement attacks; updating ACLs becomes more time-consuming with each changing user, administrator, or business requirement.

Lack of Flexibility

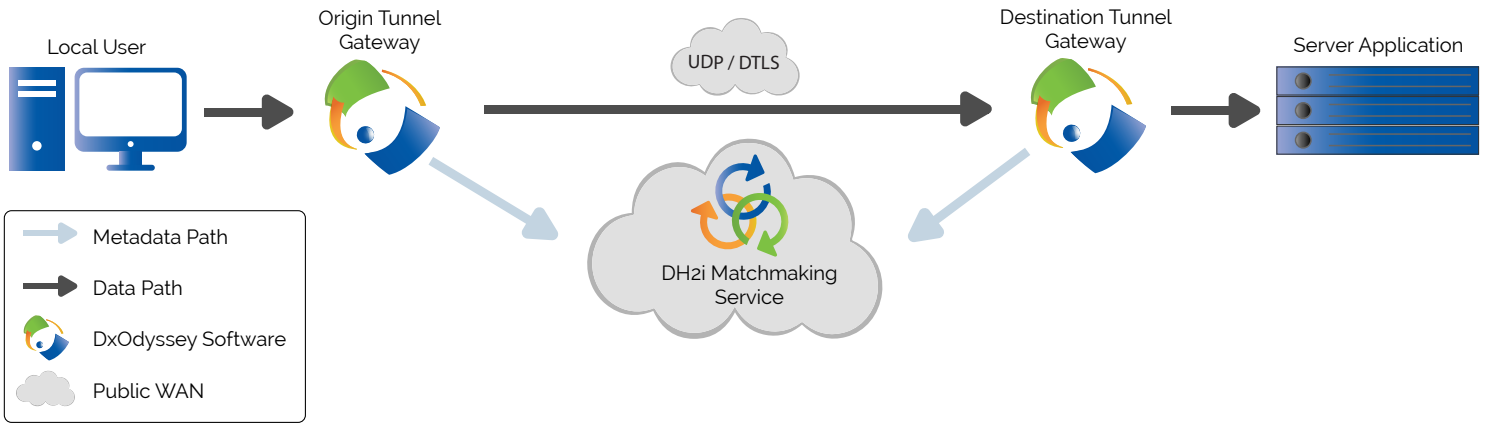
» MongoDB endows an enviable flexibility of runtime environments spanning on-premises locations, the cloud, containers, edge computing, and IoT settings. Without a security solution pliant enough to rapidly deploy to each of these environments as needed, organizations must either limit their use of MongoDB's vaunted flexibility, or risk exposing data and their transmissions to unknown, malicious actors. Traditional security methods were not designed to keep pace with the flexibility of hybrid clouds, multi-cloud deployments, and the swift changes in settings MongoDB accommodates.

Integration Vulnerability

» The cloud is regarded as an ideal means of federating and integrating data of all varieties. MongoDB supports all aspects of cloud use cases in this regard, especially because of its extremely adaptable document data format. Fortifying integration efforts between remote locations as quickly as MongoDB can distribute its database via sharding can prove difficult. That difficulty increases with the platform's mobile capabilities, which further add to the data quantities stemming from decentralized sources. With data coming from so many diverse places, simply assembling data in a central place for applications or analytics creates a huge opportunity for infiltrating data transmissions.

The Solution

The compartmentalized micro-tunnels of DxOdyssey offer invisible data transmissions between virtually any setting without exposing data to anyone other than desired target systems. Organizations can follow these simple steps for securing DxOdyssey's segmented tunnels between distributed locations:



» *Architectural diagram of a secure micro-tunnel created with DxOdyssey*

How to Access MongoDB on an Ubuntu Node Using DxOdyssey

1. Install DxOdyssey on the nodes you want connected.
2. Link them with a matchmaker tool assigning random port generation for heightened security.
3. Put the nodes for the gateway and database in each other's host files.

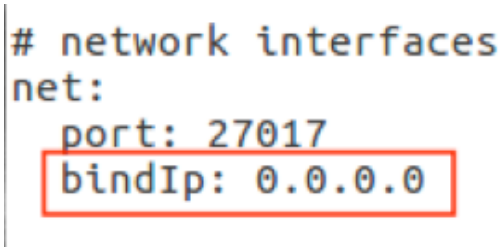
If MongoDB is already installed on an Ubuntu node:

4. Ensure a user has permissions to access the database.
5. Ensure the configuration file (/etc/mongod.conf) has the bindIP option set to 0.0.0.0.
6. Restart the service (sudo service mongod restart) if you changed the configuration file then skip to step 7.

If MongoDB is not previously installed:

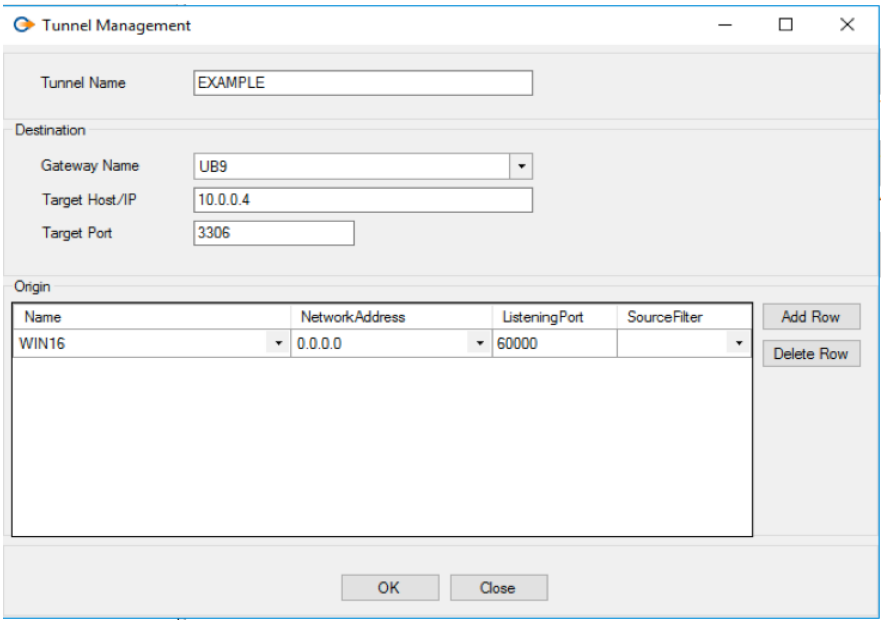
4. Install MongoDB.
5. Log in to the MongoDB shell: mongo and create a database: use dbname.
6. Go to step 5 of 'If MongoDB is already installed on an Ubuntu node' above.

STEP-BY-STEP INSTRUCTIONS CONTINUED ON PAGE 4



» *Make sure the configuration file (/etc/mongod.conf) has the bindIP option set to 0.0.0.0*

7. Ensure the Ubuntu firewall allows connections to the appropriate port by first: finding the port (you can check the port in service configuration files) the service is on in the firewall settings. Make sure incoming connections are allowed. MongoDB default: 27017.
8. If it's not allowed, allow it: sudo iptables -A IN_public_allow -p tcp -m tcp --dport PORT_NUMBER -m conntrack --ctstate NEW -j ACCEPT.
9. Create the tunnels in DxOdyssey by first: clicking on Tunnel Manager.
10. Click on Add Tunnel.
11. Name your tunnel.
12. Pick your gateway node from the dropdown menu.
13. Enter your database node's IP address in the Target Host/IP field.
14. Enter your database's port number in the Target Port field.
15. Add and configure your Origin node(s). Remember the listening port you choose for your origin nodes.
16. Click Ok to add the tunnel.



» *You should now be able to access your database through DxOdyssey tunnels by connecting to the origin node on the specified port number*

The Benefits

The benefits of deploying DxO tunnels for secure access to MongoDB on Ubuntu reinforce the business value of the distributed data storage and computations for which this database is lauded. They include:

Zero Trust Hybrid and Multi-Cloud Security

» Organizations can securely run different instances, nodes and clusters of MongoDB in hybrid and multi-cloud settings with cloaked transmissions hacker's can't detect. If intruders somehow are able to infiltrate these transmissions, DTLS encryption and Public Key Authentication will render the data meaningless to them. DxO tunnels enable organizations to realize all of the advantages of decentralization while decreasing, instead of increasing, cyber security risks.

Per-Application Connectivity

» Because DxO tunnels connect applications directly to each other (as opposed to connecting their larger networks), they provide one of the most granular forms of security for distributed data use cases—such as positioning data in remote locations for governance and compliance requirements. Application level connectivity allows for discreet transmissions for data integrations and communication between locations.

Spontaneous Port Selection

» The secure connections of DxO tunnels are issued via a cloud matchmaker service that randomly assigns ports to connect servers, then closes them once they're joined. No one, including the source system, target system, and especially hackers, knows which ports will be assigned until the designation is made. Thus, users can consolidate data from heterogeneous MongoDB applications for a single view of customer data, for instance, without compromising security.