

Integrating MongoDB Atlas with Heroku Private Spaces

[Try MongoDB Atlas](#)

Raphael Londner

January 16, 2018

Technical, Cloud



Introduction

Heroku and MongoDB Atlas are the perfect fit for modern, cloud-based app development and deployment. Since its inception in 2007, Heroku has been a PaaS (Platform-as-a-Service) favorite of developer and operations teams thanks to its tight integration to CI tools and ease of app deployment. MongoDB is also a long-time favorite of developers who value increasing their productivity and decreasing application development cycles. MongoDB's fully managed DBaaS (Database-as-a-Service), [Atlas](#), is also popular among cloud DevOps teams, who are naturally demanding a strong integration between Heroku and MongoDB Atlas.

Today, we are happy to present a tutorial showcasing how to securely integrate Heroku with MongoDB Atlas.

Protecting your cloud data assets with MongoDB Atlas

MongoDB Atlas provides industry-grade, out-of-the-box security controls: encrypted data in-flight and at-rest, encrypted backups, authentication enabled by default, IP whitelisting and

Companies hosting their MongoDB Atlas-backed applications on Heroku typically require that their data be only accessed by their applications. This has proved to be challenging in most Heroku deployments, which typically don't offer guarantees that requests performed by their hosted applications originate from fixed IPs or a fixed range of IPs (defined as CIDR blocks).

With [Heroku Private Spaces](#) however, companies can combine Heroku powerful developer experience with enterprise-grade secure network topologies. More specifically, [peering a Heroku Private Space](#) with a MongoDB Atlas cluster running in AWS is a straightforward option to secure the communication between a Heroku-deployed application and a MongoDB Atlas database, by using [MongoDB Atlas VPC Peering capabilities](#).

The tutorial below goes through the specific steps required to link a Heroku Private Space with a MongoDB Atlas project.

Initiating the VPC Peering request

The first step is to initiate the VPC Peering request on the Atlas side. To do so, it's necessary to retrieve a few parameters from the Heroku Private Space, by using the [Heroku CLI](#). After logging in with an account having access to a Private Space, use the `spaces:peering:info` command to retrieve the AWS information required by MongoDB Atlas:

```
heroku spaces:peering:info <your_private_space_name>
```

```
$ heroku login
Enter your Heroku credentials:
Email: raphael@mongodb.com
Password: *****
Logged in as raphael@mongodb.com
$ heroku spaces:peering:info oregon-ecosystem-partners
=== oregon-ecosystem-partners Peering Info
AWS Account ID:      645490581190
AWS Region:          us-west-2
AWS VPC ID:           vpc-f8e7b09c
AWS VPC CIDR:         10.0.0.0/16
Dyno CIDRs:          10.0.128.0/20, 10.0.144.0/20
Unavailable CIDRs:
$ █
```

that region.

Copy the **AWS Account ID**, **AWS Region**, **AWS VPC ID** and **AWS VPC CIDR** values from the Heroku console above.

Now, head over the MongoDB Atlas website and navigate to the **Security** tab of your cluster (M10 or above and in the same region as your Heroku Private Space). Select the **+New Peering Connection** button and fill out the form with the values you previously copied:

Peering Connection

Describe Your Existing VPC

In order to initiate a peered connection, we need some information about your existing AWS VPC.

Note: VPC Peering is only available for AWS clusters that are M10 and above.

AWS Account ID

You can find this in *My Account* in your AWS console.

[Show me how to find this](#) ✓

645490581190

VPC ID

You can find this in your list of VPCs in the VPC dashboard in your AWS account. It usually starts with 'vpc-'.

[Show me how to find this](#) ✓

vpc-f8e7b09c

Region

If you have an existing cluster, this cannot be changed.

us-west-2

VPC CIDR

This CIDR block cannot overlap with your Atlas VPC CIDR block (shown below) or the peering will fail.

10.0.0.0/16

☒ Add this CIDR block to my IP whitelist

Press the **Initiate Peering** button, and verify that the VPC Peering request appears in Atlas' VPC Peering list (with a "Waiting for Approval" status):

VPC ID	Status	Peering ID ⓘ	Atlas CIDR	VPC CIDR ⓘ	
vpc-9738a7f0	Available	pcx-abb835c2	172.16.0.0/21	172.31.0.0/16	<button>TERMINATE</button>
vpc-f8e7b09c	Waiting for Approval How do I approve this connection?	pcx-c8e2c1a1	172.16.0.0/21	10.0.0.0/16	<button>TERMINATE</button>

Approving the VPC Peering request

Now that the VPC Peering request has been initiated on the MongoDB Atlas side, let's approve it on the Heroku side. In the Heroku console, the following command should display the request we just created in MongoDB Atlas:

```
heroku spaces:peerings <your_private_space_name>
```

```
$ heroku spaces:peerings oregon-ecosystem-partners
== oregon-ecosystem-partners Peerings
```

PCX ID	Type	CIDR Block	Status	VPC ID	AWS Account ID	Expires
pcx-c8e2c1a1	unknown	172.16.0.0/21	pending-acceptance	vpc-dd0171ba	625844033181	2018-01-15T22:45:12Z

Take note of the PCX ID value of your VPC Peering ID and pass it to Heroku space:peerings:accept command:

```
heroku spaces:peerings:accept <your_PCX_ID> --space <your_private_space_name>
```

```
$ heroku spaces:peerings:accept pcx-c8e2c1a1 --space oregon-ecosystem-partners
Accepting and configuring peering connection pcx-c8e2c1a1
```

Verifying that VPC Peering works

The first step to verify that VPC Peering has been properly set up between your Heroku Private Space and MongoDB Atlas is by running the following Heroku command again:

```
heroku spaces:peerings <your_private_space_name>
```

```
$ heroku spaces:peerings oregon-ecosystem-partners
== oregon-ecosystem-partners Peerings
```

PCX ID	Type	CIDR Block	Status	VPC ID	AWS Account ID	Expires
pcx-c8e2c1a1	customer-managed	172.16.0.0/21	active	vpc-dd0171ba	625844033181	

The peering connection should now appear as active.

In MongoDB Atlas, the peering connection should now also appear as available:

MongoDB Users					IP Whitelist					Peering				
VPC ID					Status					Peering ID ⓘ				
[REDACTED]					[REDACTED]					[REDACTED]				
vpc-f8e7b09c					● Available					pcx-c8e2c1a1				
										172.16.0.0/21				
										10.0.0.0/16				
										[REDACTED]				

[+ NEW PEERING CONNECTION](#)

TERMINATE

TERMINATE

The next verification step would be to run an Heroku-deployed app connected to your MongoDB Atlas cluster and verify that you can read from or write to it.

For instance, you could clone [this GitHub repository](#), customize its config.js file with your MongoDB Atlas connection string, and deploy its atlas-test branch to your Heroku Private Space using [Heroku GitHub Deploys](#). Since Heroku automatically runs npm start for each Node-detected app, it will keep calling the [produce.js](#) script. As a result, documents should be created in the devices collection of a demo database in your Atlas cluster (if it doesn't, I recommend that you first verify that the CIDR block of your Heroku Private Space is present in the IP Whitelist of your MongoDB Atlas cluster).

Next steps

I hope that you found this Heroku-MongoDB Atlas integration tutorial useful. As next steps, I recommend the following:

[Sign up for MongoDB Atlas](#) if you don't already use it.

- Watch a [VPC Peering video tutorial](#) with MongoDB Atlas.
- Get more familiar with [MongoDB Atlas documentation](#)
- Contact Heroku if you don't already have access to a Private Space.
- Explore the [Heroku Private Spaces documentation](#).

*About the Author - **Raphael Londner***

Raphael Londner is a Principal Developer Advocate at MongoDB, focused on cloud technologies such as Amazon Web Services, Microsoft Azure and Google Cloud Engine. Previously he was a developer advocate at Okta as well as a startup entrepreneur in the identity management space. You can follow him on Twitter at [@rlondner](#)

 **Recommend** **Tweet** **Share****Sort by Newest** ▼**LOG IN WITH****OR SIGN UP WITH DISQUS** 

Be the first to comment.

ALSO ON MONGODB.COM BLOG**The Top 12 BSON Data Types you won't find in JSON**

1 comment • 4 months ago

**rporrwal** — Nice and informative article defining BSON data types and also please provide some examples ...**MongoDB Schema Design Patterns - Bucket | MongoDB**

1 comment • 2 months ago

**rporrwal** — Bucket Pattern facilitates organizing storage of data into specific groups leveraging support of ...**MongoDB Stitch/Mobile Mars Rover Lands at AWS re:Invent**

1 comment • 4 months ago

**Shrey Batra** — Amazing post... Wished I could see it live..**A Proposal to End-of-Life Our Generic Linux Tar Packages**

5 comments • 4 months ago

**Nima Parsa** — Please supporting Arch/Manjaroor add snapcraft or flatpak to use any distro . **Subscribe**  **Add Disqus to your site** **Add Disqus** **Add** **Disqus** **Disqus** **Disqus** **Disqus** **Disqus** **Disqus****Resources**[NoSQL Database Explained](#)[MongoDB Architecture Guide](#)[MongoDB Enterprise Advanced](#)[MongoDB Atlas](#)

[Products](#)[Solutions](#)[Customers](#)[Resources](#)[Learn](#)[W](#)
[Mon](#)[Referral Program](#)

Education & Support

[View Course Catalog](#)[Certification](#)[MongoDB Manual](#)[Installation](#)[FAQ](#)

Popular Topics

[Migrate to MongoDB Atlas](#)[Building a REST API with MongoDB Stitch](#)[Ingesting and Visualizing API Data with Stitch and Charts](#)

About

[MongoDB, Inc.](#)[Leadership](#)[Press Room](#)[Investors](#)[Careers](#)[Contact Us](#)[Legal Notices](#)[Security Information](#)[Office Locations](#)[Code of Conduct](#)

Follow Us

[Facebook](#)

[Products](#)[Solutions](#)[Customers](#)[Resources](#)[Learn](#)[W](#)
[Mon](#)[Twitter](#)[LinkedIn](#)[Slack](#)[StackOverflow](#)

Get MongoDB Email Updates



© 2019 MongoDB, Inc.

Mongo, MongoDB, and the MongoDB leaf logo are registered trademarks of MongoDB, Inc.