# Data Storage and Retrieval using Multichain Streams

In this Hands-On document you learn how to use the concept of streams in Multichain to store and retrieve data on the blockchain network using SAP Cloud Platform Blockchain – Multichain.

## Preparation

# You have already created your Multichain Instance (refer to “Create your first Multichain Node”) or you have even created your own network of multichain nodes (refer to “Build up your own Multichain network”). You have noted down your api key(s).

## Execution

Blockchain are focusing on creation and tracking of assets (refer to Hands-On document on Asset transfer). However, sometimes you have a requirement to simply store and retrieve data on the blockchain like storing and retrieving data in a classical database.

For this, Multichain introduces the concept of streams. You may want to read the official Multichain introduction to streams: <https://www.multichain.com/blog/2016/09/introducing-multichain-streams/>

Basically, a stream is a bucket to store key/values with basic authorization concept. In this document, we show how to store values per key and read them later. Our sample scenario is a sales order tracking scenario where we write status information for a sales order ID by one node and read it from a second one.

In this hands-on document we will call json-rpc to the node. For this you might use any http client software or library of your programming language of choice. For demos, we use Postman or Postman for Chrome. The reference guide for json-rpc commands in Multichain is found under:

<https://www.multichain.com/developers/json-rpc-api/>

### Create a stream

In your Multichain setup you should have two nodes connected to one Multichain network (refer to Hands-On document “Build up your own Multichain network”). Note down the api key and rpc URL from the service key of your first Multichain node.

Execute the following json-rpc call to your first node (host and api key is given in the service key of the Multichain service):

POST HTTP/1.1

Host: maas-proxy.cfapps.sap.hana.ondemand.com/<instance ID>/rpc

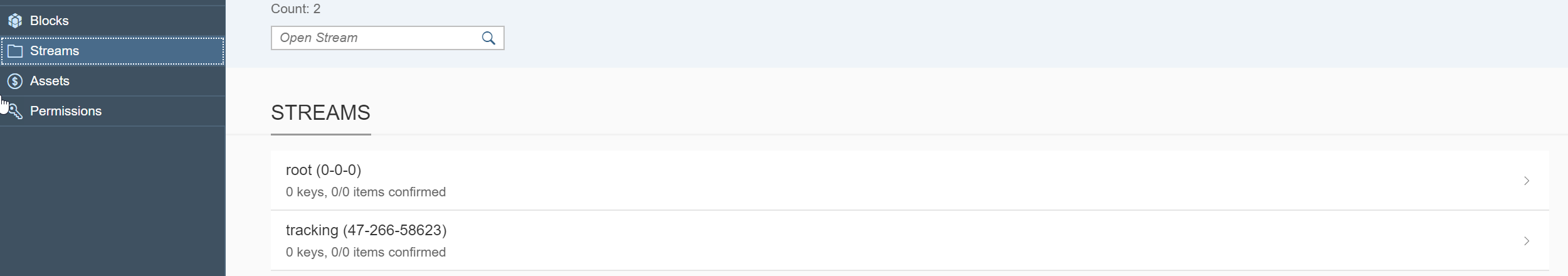
(HEADER) apikey: <your API Key>

(BODY) {"method": "create", "params": ["stream", "tracking", true]}

This call will create a new stream with the name “tracking”. The Boolean true indicates that anyone on the network can write data into the stream. If the value is false, you explicitly need to assign write permissions on address level and makes it a private bucket.

In the Multichain nodes created by SAP Cloud Platform Multichain all streams that will be created are subscribed automatically by other nodes (autosubscribe=true). This will create an index table on each node to optimize the read performance. The drawback is using disk space for the index table. If you do not want to have the autosubscribe feature you may disable it per stream (rpc command unsubscribe) or generally disable the autosubsribe feature (rpc command setruntimeparam for parameter unsubscribe).

After creating the stream you may check it on the Multichain Dashboard.



Click on the tracking item to see more details. Mind that the stream is open and subscribed.

### Create a stream item

Execute the following json-rpc call to your first node:

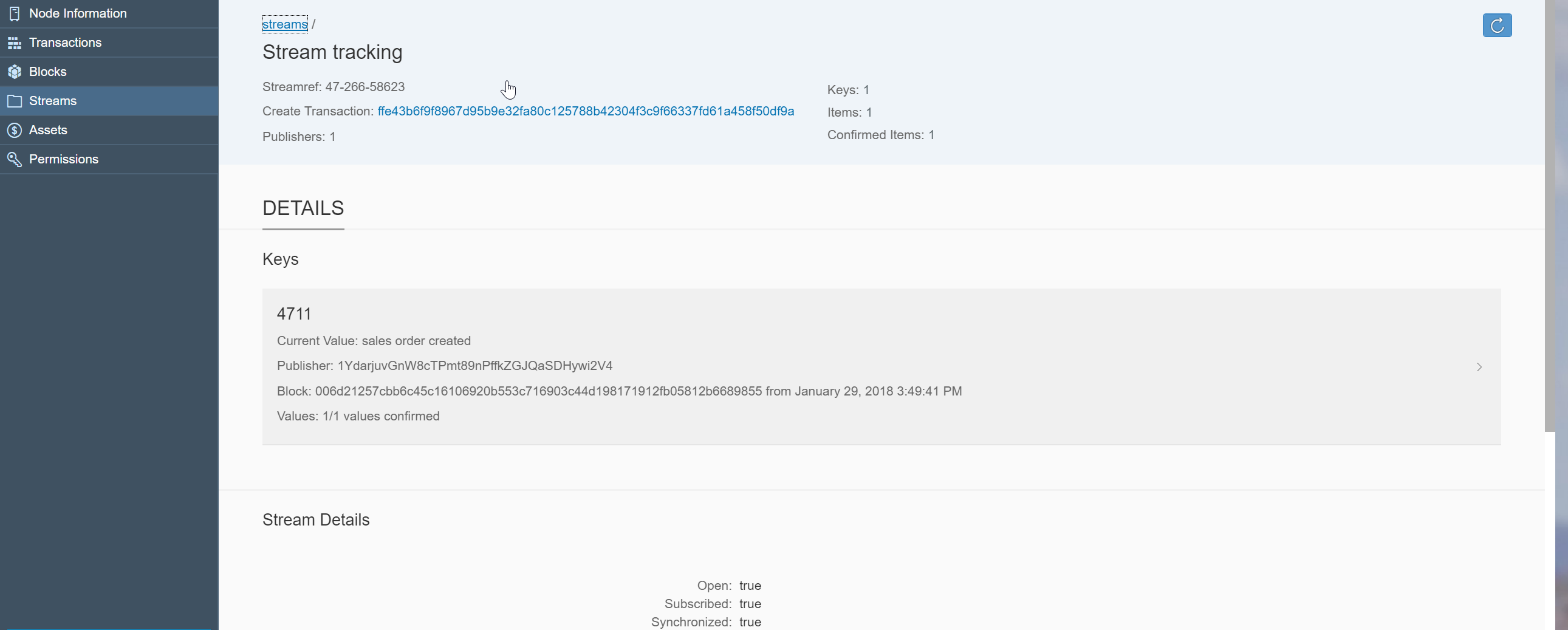
POST HTTP/1.1

Host: maas-proxy.cfapps.sap.hana.ondemand.com/<instance ID>/rpc

(HEADER) apikey: <your API Key>

(BODY) {"method": "publish", "params": ["tracking", "4711", "73616c6573206f726465722063726561746564"]}

The json-rpc command “create” requires three parameters, the stream name, the key and the value in hex format. So this example will create an entry in the stream tracking with the key 4711 and the value “sales order created” ("73616c6573206f726465722063726561746564" is the hex encoded value of “sales order created”). Check it in the Multichain Dashboard:



The dashboard automatically convert the hex value to a string, so that you can see that the value was created for the key 4711. Of course, you see who has created (signed) the entry and you find the reference to the raw blockchain transaction and the block. The data is therefore stored as part of the chain of blocks (once the referred block is mined) and the entry is persisted immutable. On detail level you find the timestamp created by the blockchain for this entry.

### Create another stream item

Let’s create a second value for the key: “sales order released”, but from another node (or account using createfrom) To do so, execute the following command:

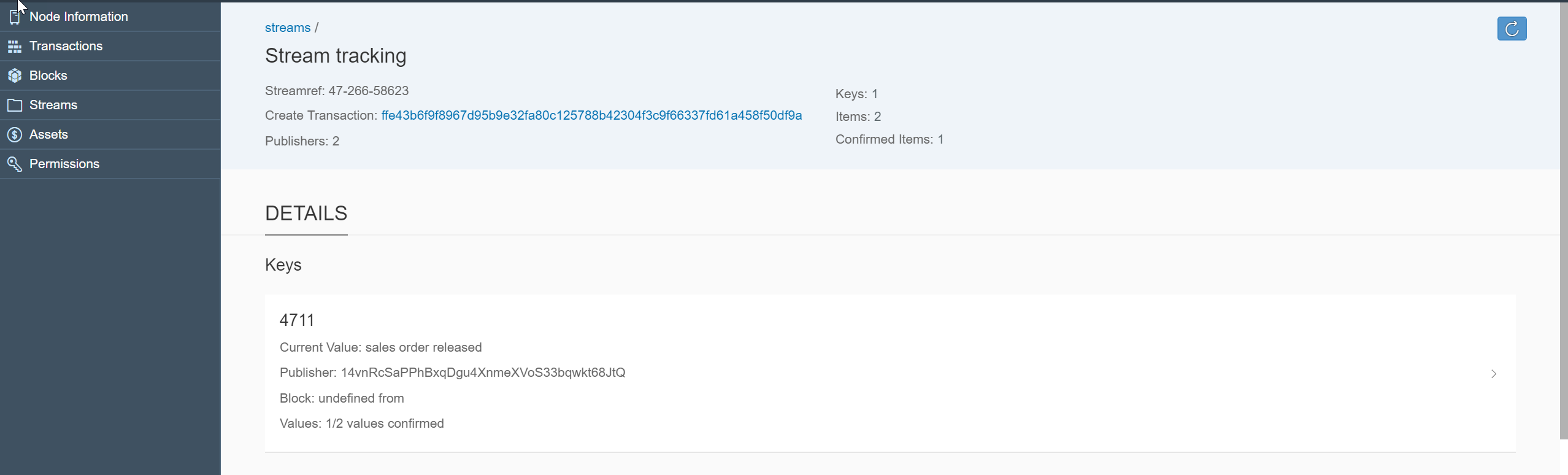
POST HTTP/1.1

Host: maas-proxy.cfapps.sap.hana.ondemand.com/<instance ID>/rpc

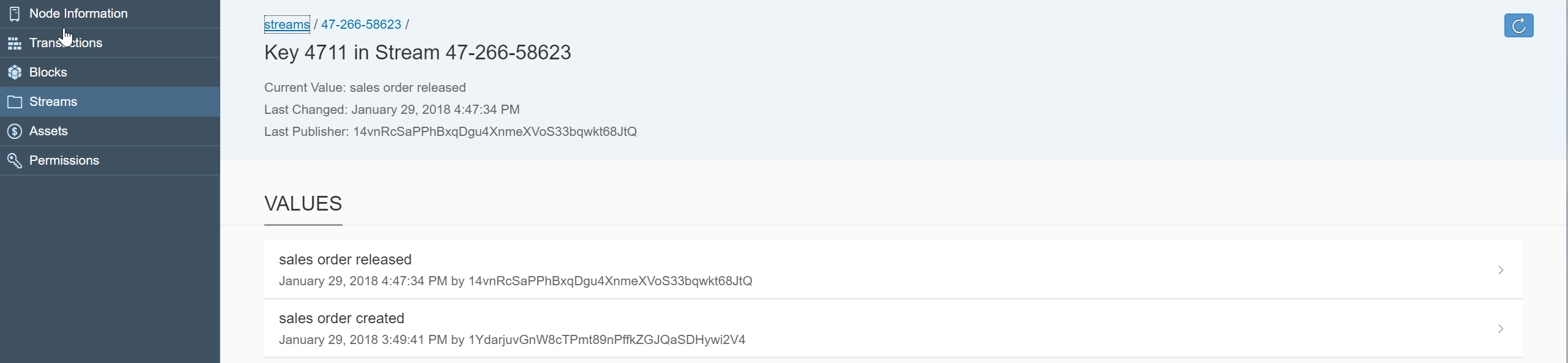
(HEADER) apikey: <your API Key>

(BODY) {"method": "create", "params": ["tracking", "4711", "73616c6573206f726465722072656c6561736564"]}

In the dashboard you will now see new entry:



On detail level you will see both entries:



### Read the data

Let’s now read the data from the blockchain. Mind, that you must be subscribed to the stream to read the data, which is the default for the multichain nodes on SAP Cloud Platform. In json-rpc you have several options to read data from a stream. We use liststreamkeyitems, that gives us all items for a given stream and a given key.

Execute the following command:

POST HTTP/1.1

Host: maas-proxy.cfapps.sap.hana.ondemand.com/<instance ID>/rpc

(HEADER) apikey: <your API Key>

(BODY) {"method": "liststreamkeyitems", "params": ["tracking", "4711"]}

and you get the response:

{

"result": [

{

"publishers": [

"1YdarjuvGnW8cTPmt89nPffkZGJQaSDHywi2V4"

],

"key": "4711",

"data": "73616c6573206f726465722063726561746564",

"confirmations": 22,

"blocktime": 1517237381,

"txid": "01da8a082f8679b6d0fc93c2334532c040940addb0ce9dc41c954415dc7b0894"

},

{

"publishers": [

"14vnRcSaPPhBxqDgu4XnmeXVoS33bqwkt68JtQ"

],

"key": "4711",

"data": "73616c6573206f726465722072656c6561736564",

"confirmations": 11,

"blocktime": 1517240854,

"txid": "14369f0967293cb68222170e672bd30af485eb5c1c9224602c641d9a8bdd1af5"

}

],

"error": null,

"id": null

}

You get all historic entries for the key. The most recent entry is the current value.

You may convert the data field from hex to string and the blocktime to the YYYY/MM/DD format to retrieve the both original values with creation date displayed also in the dashboard.

In this document you have successfully implemented your first stream scenario.