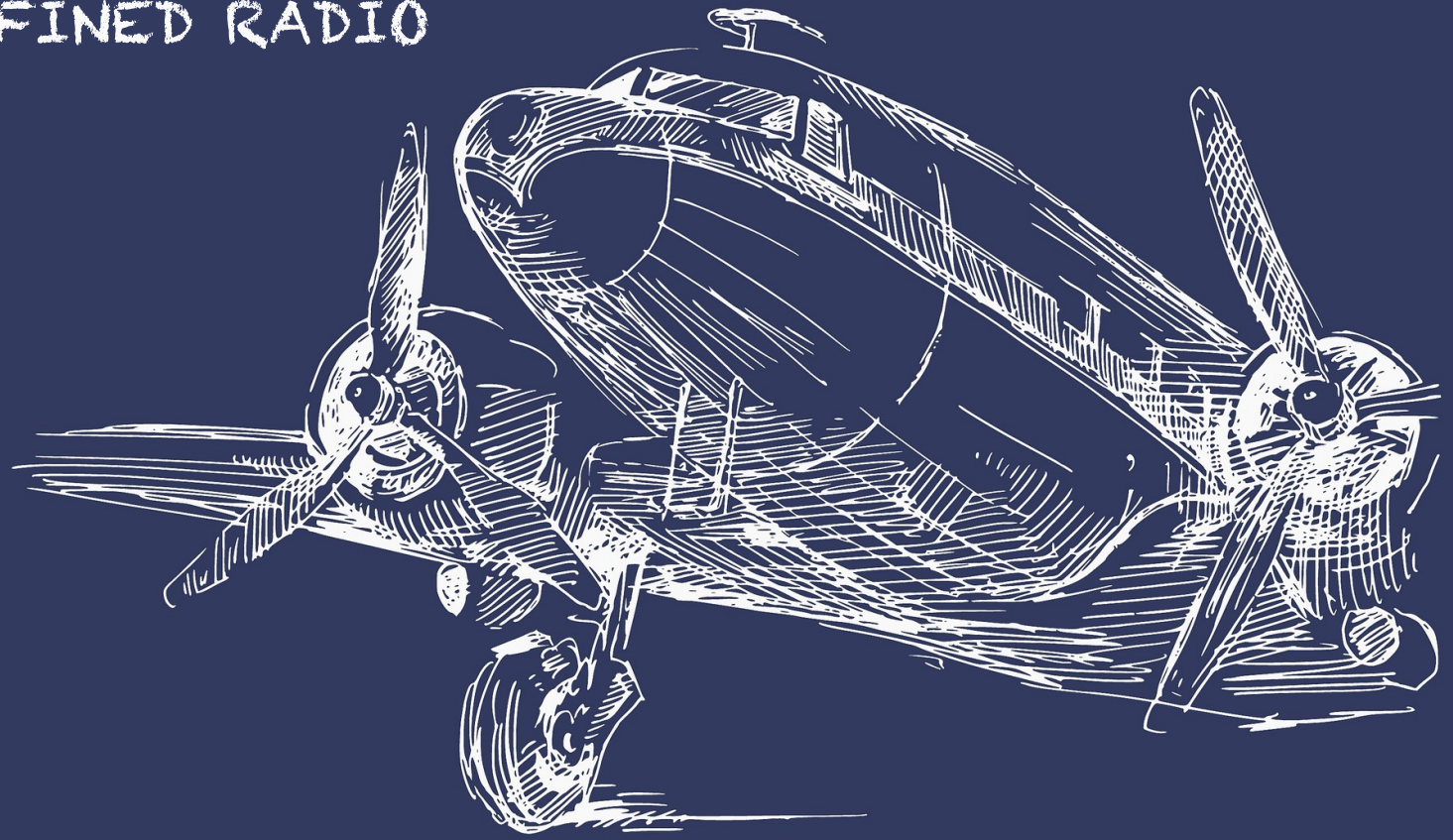
TRACKING AIRCRAFT

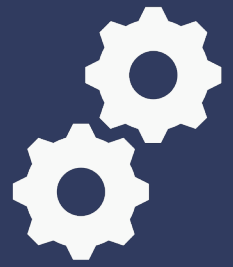
WITH REDIS + SOFTWARE-DEFINED RADIO



redis

AIRCRAFT

RADIO



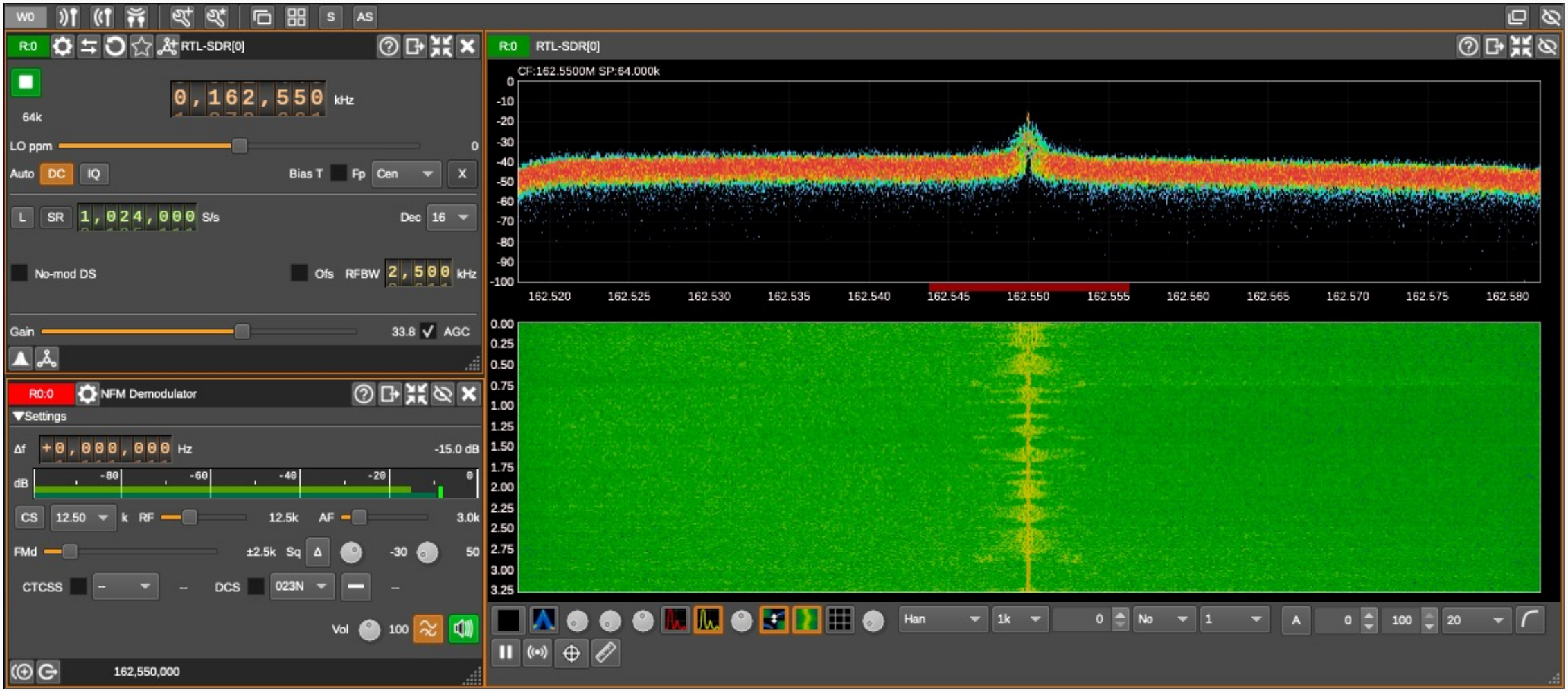
MICROSERVICES

STREAMS

REDISEARCH


WHAT IS
SOFTWARE-DEFINED
RADIO?
redis
transistor

SOFTWARE DOES THE REST



GETTTING THE HARDWARE

RTL-SDR

Upconverter

HF Antenna

SOFTWARE OPTIONS

Gqrx SDR

Open source software defined radio by Alexandru Csete OZ9AEC

CubicSDR

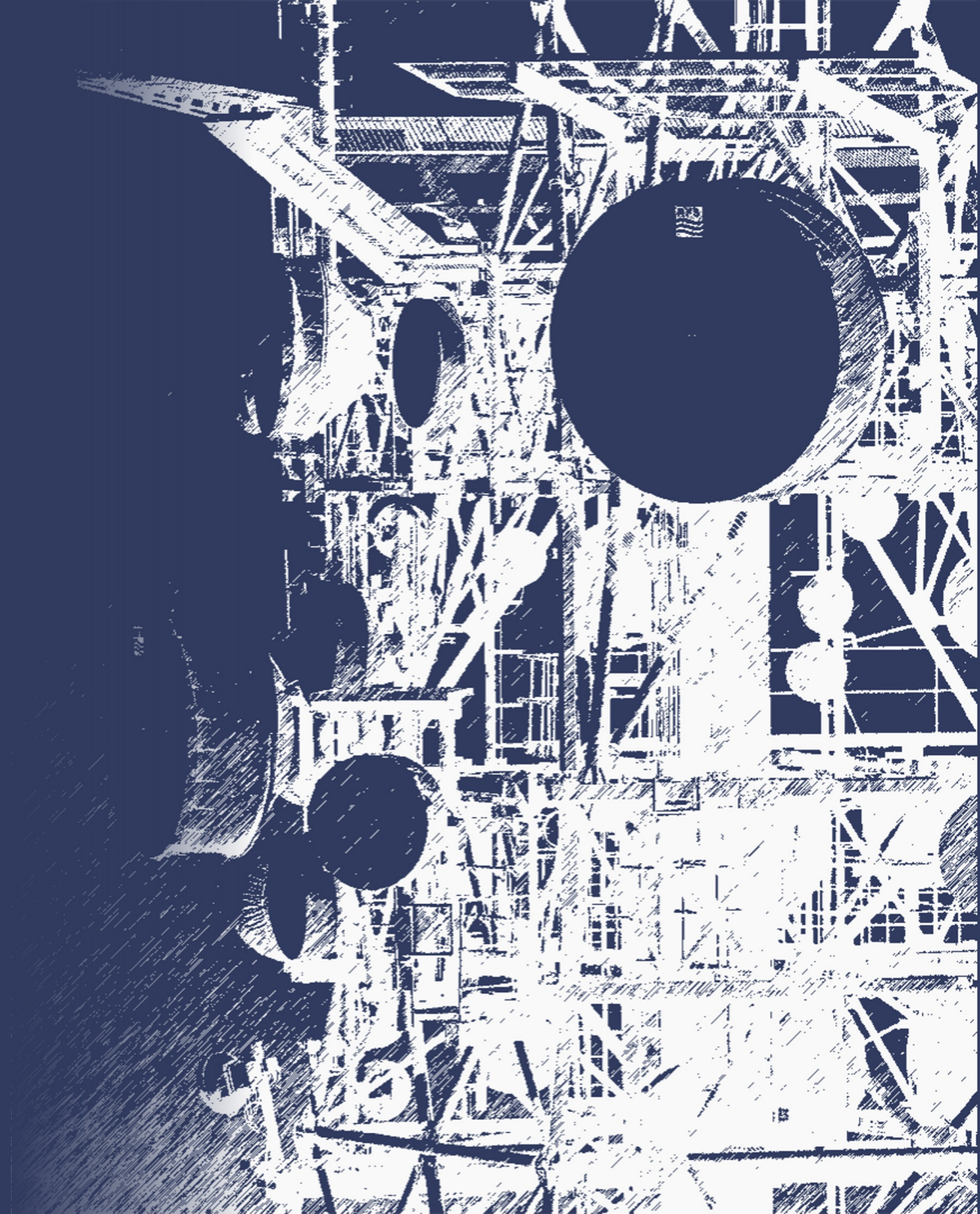
Cross-Platform and Open-Source Software-Defined Radio Application

DATA FORMATS

MORSE CODE

RTTY
RADIO TELETYPE

APRS
AUTOMATIC PACKET REPORTING SYSTEM



AUTOMATIC PACKET REPORTING SYSTEM

VEHICLES

WEATHER STATIONS

INDIVIDUALS

INDIVIDUALS

REPEATERS

INTERNET GATEWAYS

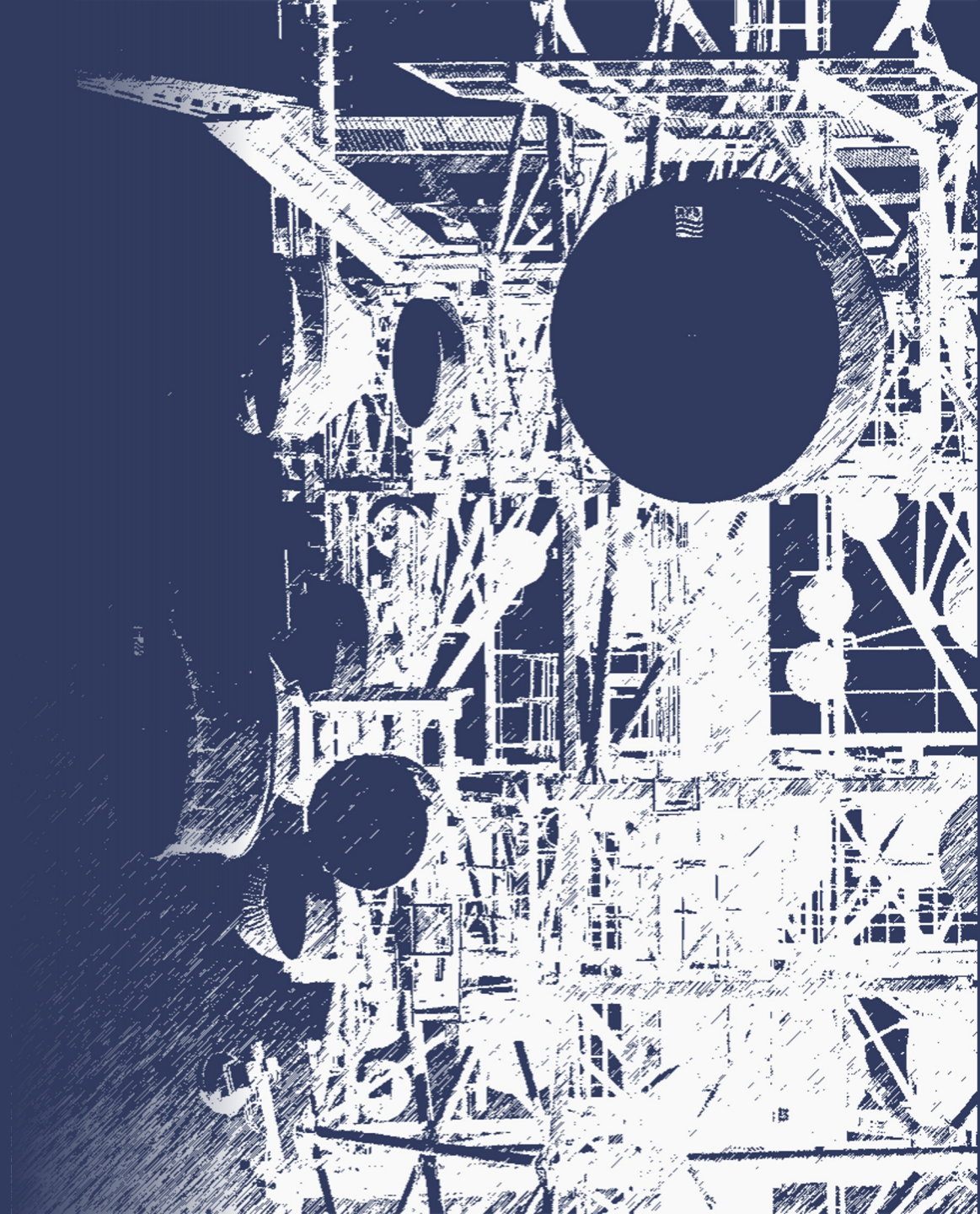
DATA FORMATS

MORSE CODE

RTTY
RADIO TELETYPE

APRS
AUTOMATIC PACKET REPORTING SYSTEM

AIS
AUTOMATIC IDENTIFICATION SYSTEM



DATA FORMATS

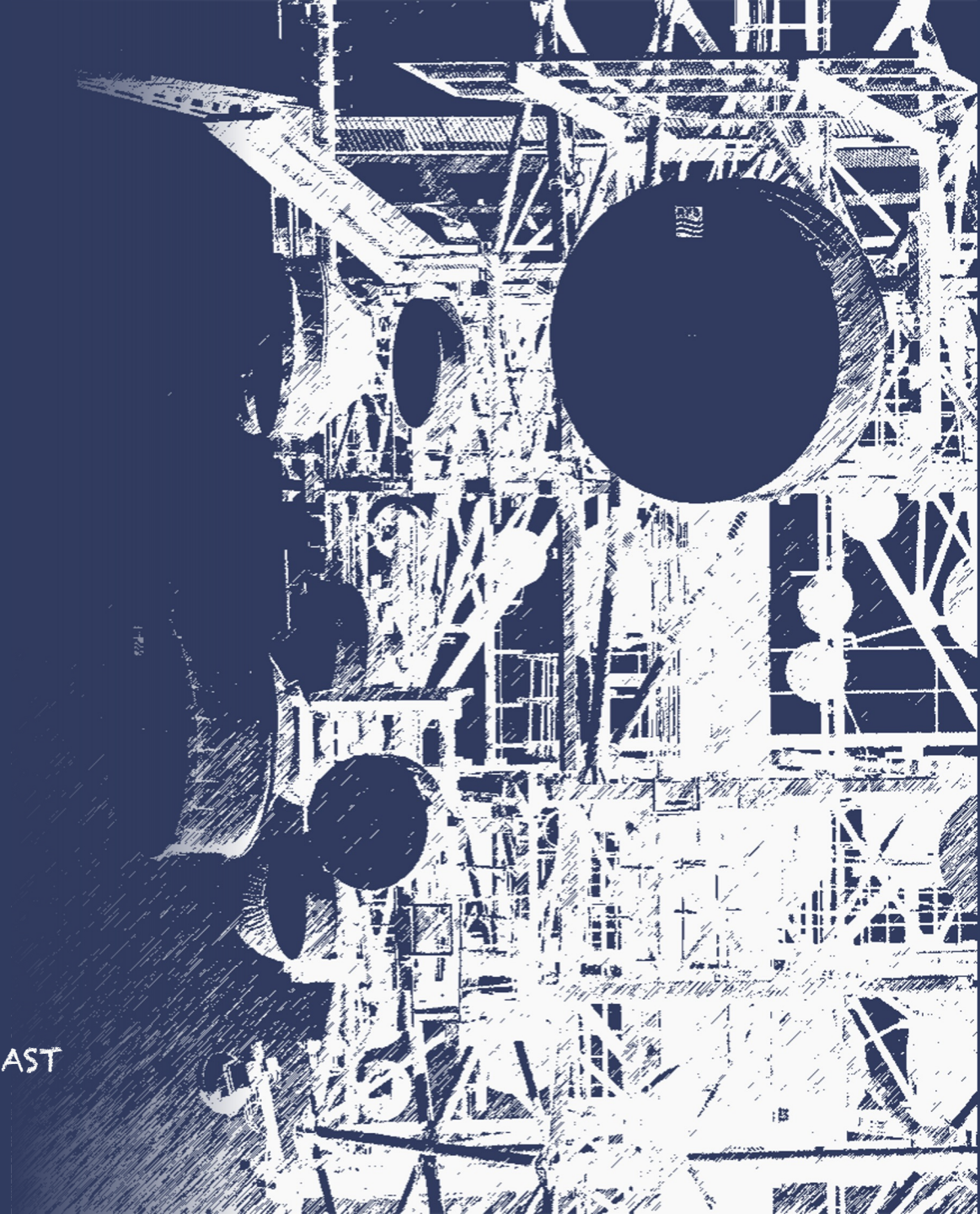
MORSE CODE

RTTY
RADIO TELETYPE

APRS
AUTOMATIC PACKET REPORTING SYSTEM

AIS
AUTOMATIC IDENTIFICATION SYSTEM

ADS-B
AUTOMATIC DEPENDENT SURVEILLANCE BROADCAST



1090 MHz

ADS-B

ICAO ID
A835AF

CALLSIGN
N628TS

ALTITUDE
22,100 ft.

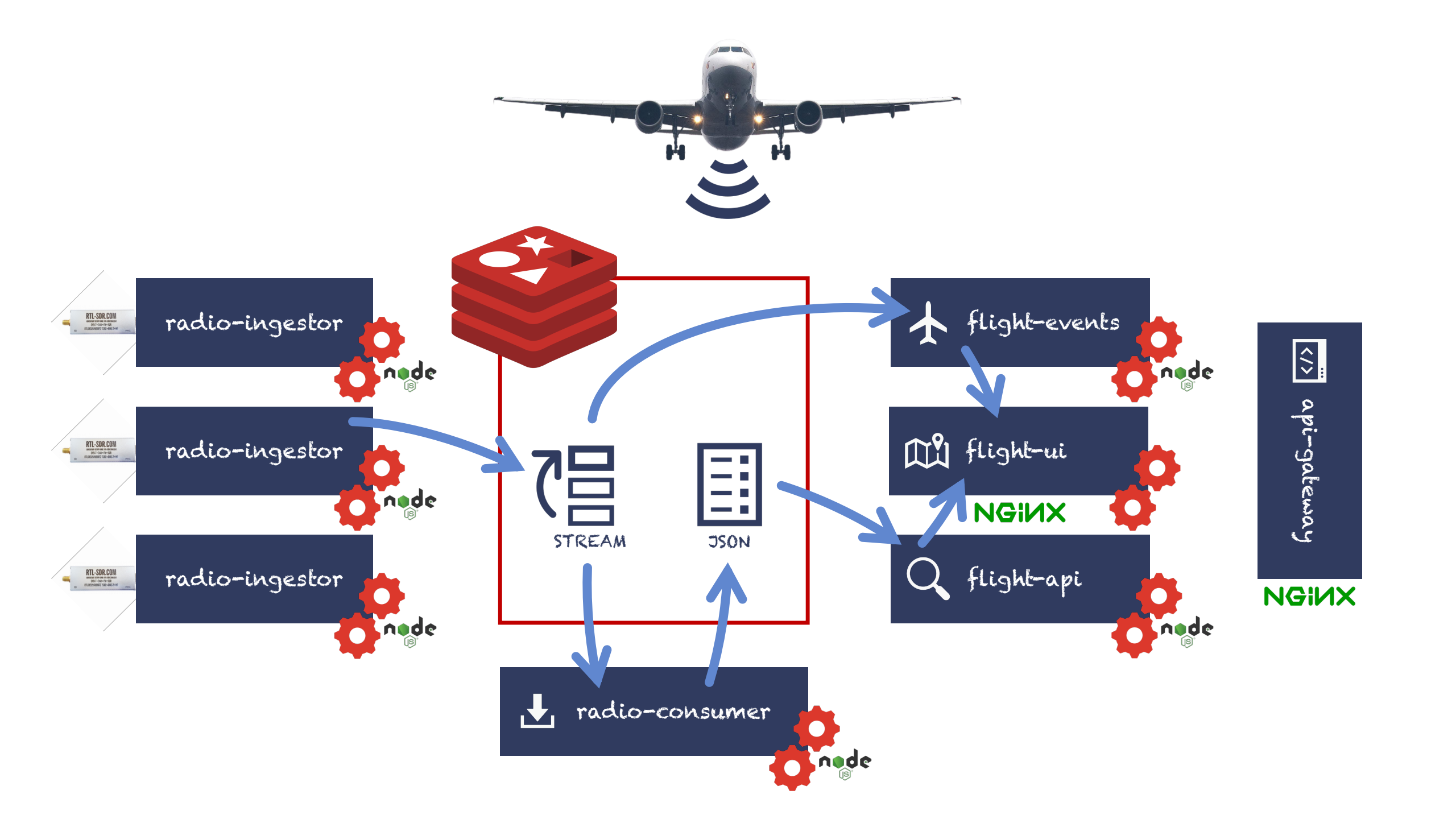
SPEED
395 kn.

HEADING
344°

LOCATION
39.963°N
83.205°W

Tot: 15 Vis: 15 RSSI: Max -25.3+ Mean -32.4 Min -34.7- MaxD: 0.0nm+   /

Hex	Mode	Sqwk	Flight	Alt	Spd	Hdg	Lat	Long	RSSI	Msgs	Ti
A446F1	A2	1640	LXJ375	22100	395	344	39.963	-83.205	-31.4	347	04
A296F4	A2	6746	N266TD	5575	189	273	39.994	-82.956	-30.5	123	30
A87108	A2	6702	PAC369	30650	523	051	40.384	-83.412	-33.3	286	00
A04E60	A2		RPA4747	30000	394	271	40.285	-82.991	-32.4	265	29
A655F1	A2	5612	RVJ55160	6075	296	098	40.097	-82.960	-25.3+	454	08
AAA244	A2	1040	UAL296	31000	494	101	40.369	-83.046	-31.5	1026	12
AC0417	A2	4063	ASH6330	9725	268	272	40.009	-83.261	-34.0	410	22
A2587B	A2	1535	XSR250	39000	534	059	40.560	-82.938	-33.6	688	52
AB2AE1	A2	2231		35025	575	059	40.676	-83.134	-34.7-	346	20
A8A70F	A0		DABJET55	36025	347	264	39.756	-83.036	-32.9	51	31
AA1A1F	A2		ABX3185	31000					-33.5	571	43
ABBBC7	A2		ASA1108	31025	517	098	40.238	-82.689	-32.0	885	44
ADA94D	A2	4164	N98FE	40000	348	239	39.852	-83.509	-34.1	2350	00
A575D3	A2		JRT51	43000	399	209	39.947	-83.783	-33.1	852	15
ACBCFC	A2		N92MZ	20000	210	281	40.257	-83.377	-33.7	1632	26


radio-ingestor
radio-ingestor
radio-ingestor
RTL-SDR.COM
node
STREAM
JSON
radio-consumer
flight-events
flight-ui
NGINX
flight-api
api-gateway
NGINX

USING JSON

```
redis.cloud:6379> JSON.SET aircraft:A835AF $ '{"icaoId": "A835AF",
"callsign": "N628TS"}' (OK)

redis.cloud:6379> JSON.GET aircraft:A835AF $.icaoId
"A835AF"

redis.cloud:6379> JSON.GET aircraft:A835AF
"{\"icaoId\":\"A835AF\",\"callsign\":\"N628TS\"}"
```

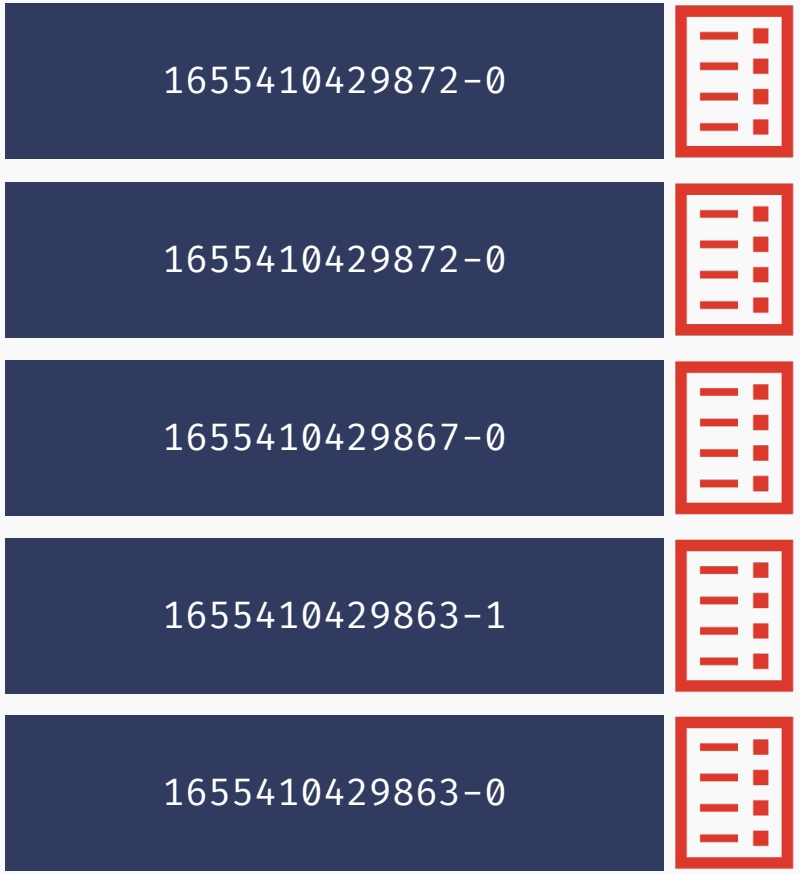
JSON

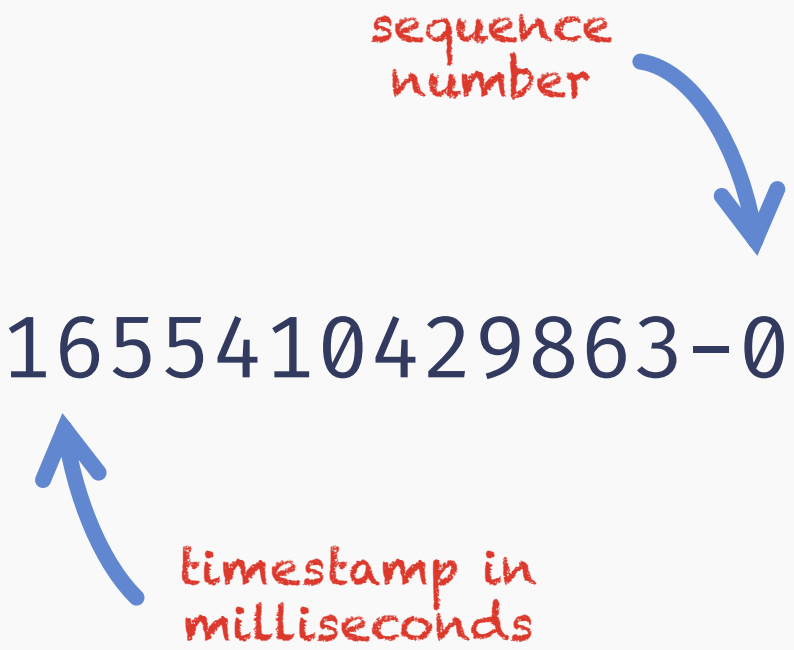
- Multi-level key-value data structure
- Keys are Strings
- Values are Strings, Lists, Numbers, or Objects
- Included since Redis Stack

STREAMS

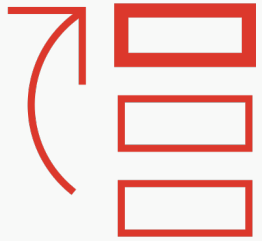
- An event stream contained in a key
- Events have ID in format 12345-0
- Events have key-value data
- Included since Redis 5.0

UNDERSTANDING STREAMS





USING STREAMS



```
redis.cloud:6379> XADD radio:events * icaoId A835AF callsign N628TS
"1664985043318-0"

redis.cloud:6379> XADD radio:events * icaoId A835AF callsign N628TS
"1664985260457-0"

redis.cloud:6379> XREAD STREAMS radio:events 0-0
1) 1) "radio:events"
   2) 1) 1) "1664985043318-0"
         2) 1) "icaoId"
            2) "A835AF"
            3) "callsign"
            4) "N628TS"

redis.cloud:6379> XREAD COUNT 10 STREAMS radio:events 1664985043318-0
1) 1) "radio:events"
   2) 1) 1) "1664985260457-0"
         2) 1) "icaoId"
            2) "A835AF"
            3) "callsign"
            4) "N628TS"
```

WAITING FOR EVENTS

```
redis.cloud:6379> XREAD BLOCK 5000 STREAMS radio:events $
1) 1) "radio:events"
   2) 1) 1) "1664985367948-0"
         2) 1) "icaoId"
            2) "A835AF"
            3) "callsign"
            4) "N628TS"
(0.50s)
```

```
redis.cloud:6379> XADD radio:events * icaoId A835AF callsign N628TS
"1664985367948-0"
```

JSON

- Multi-level key-value data structure
- Keys are Strings
- Values are Strings, Lists, Numbers, or Objects
- Included with Redis Stack

STREAMS

- An event stream contained in a key
- Events have ID in format 12345-0
- Events have key-value data
- Included since Redis 5.0

REDISEARCH

- Indexing & full-text search for Redis
- Searches & indexes Hashes
- Searches & indexes JSON
- Included with Redis Stack

UNDERSTANING REDISEARCH

Text

Tag

Numeric

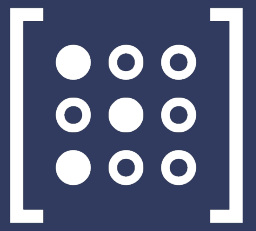
Geo

USING REDISEARCH

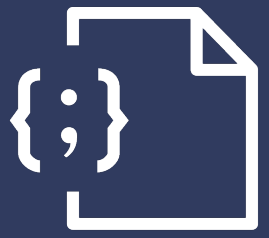
```
redis.cloud:6379> FT.CREATE aircraft:index ON HASH
  PREFIX 1 aircraft:
  SCHEMA icaoId TAG altitude NUMERIC location GEO

redis.cloud:6379> FT.SEARCH aircraft:index
  "@icaoId:{A835AF} @altitude:[0 10000] @location:[0.0 0.0 50 mi]"

1) (integer) 1
2) "aircraft:A835AF"
3) 1) "icaoId"
   2) "A835AF"
   3) "callsign"
   4) "N628TS"
   5) "altitude"
   6) "5000"
   7) "location"
   8) "0.0,0.0"
```

redis stack

REDIS BLOOM



REDIS GRAPH



REDIS JSON

REDIS SEARCH

REDIS TIMESERIES

RTL-SDR.COM

radio-ingestor

node

RTL-SDR.COM

radio-ingestor

node

RTL-SDR.COM

radio-ingestor

node

STREAM

JSON

flight-events

node

flight-ui

NGINX

flight-api

node

api-gateway

NGINX

radio-consumer

node

radio-ingestor

radio-ingestor

radio-ingestor

STREAM

HASH

flight-events

flight-ui

NGINX

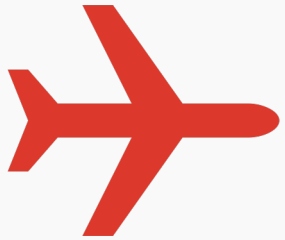
flight-api

api-gateway

NGINX

radio-consumer

CONNECTING TO THINGS



```
import * as sbs1 from 'sbs1'
import * as redis from 'redis'

// connect to SBS1 source
const sbs1Client = sbs1.createClient({
  host: SBS1_HOST, port: SBS1_PORT })

// create a Redis client
const redisClient = redis.createClient({
  socket: { host: REDIS_HOST, port: REDIS_PORT },
  password: REDIS_PASSWORD
})

// catch and log any Redis errors
redisClient.on('error', (err) => console.log('Redis Client Error', err))

// connect to Redis
await redisClient.connect()
```

WAIT FOR MESSAGES

```
// await messages
sbs1Client.on('message', msg => {

  ...process the message...

})
```

PROCESSING THE MESSAGE

```
// add fields that are always in the message
const event = {
  radio: radioId,
  icaoId: msg.hex_ident,
  type: toTransmissionType(msg.transmission_type),
  generatedDateTime: toEpochMilliseconds(msg.generated_date, msg.generated_time).toString(),
  loggedDateTime: toEpochMilliseconds(msg.logged_date, msg.logged_time).toString()
}

// add fields that might be in the message
if (msg.callsign !== null) event.callsign = msg.callsign.trim()
if (msg.altitude !== null) event.altitude = msg.altitude.toString()
if (msg.lat !== null) event.latitude = msg.lat.toString()
if (msg.lon !== null) event.longitude = msg.lon.toString()
if (msg.ground_speed !== null) event.velocity = msg.ground_speed.toString()
if (msg.track !== null) event.heading = msg.track.toString()
if (msg.vertical_rate !== null) event.climb = msg.vertical_rate.toString()
if (msg.is_on_ground !== null) event.onGround = msg.is_on_ground.toString()
```

ADDING EVENTS TO A STREAM

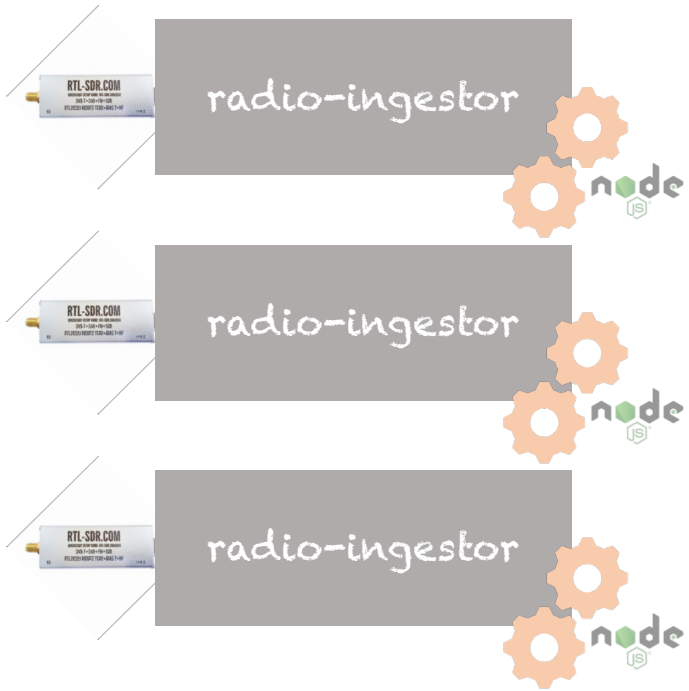
```
// add the event to the stream
redisClient.xAdd(STREAM_KEY, '*', event)
```

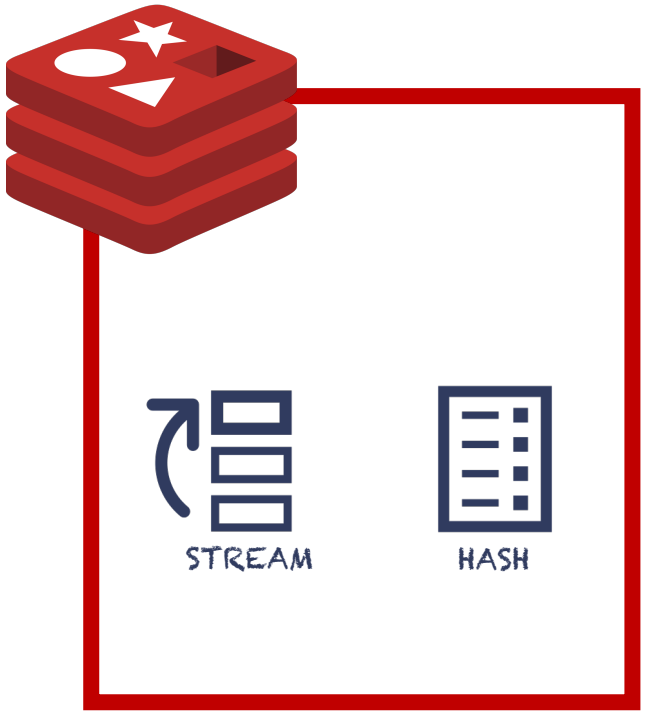
```
redis.cloud:6379> XADD radio:events * icaoId A835AF callsign N628TS ...
"1655410429863-0"
```

TRIMMING OLD EVENTS

```
// find oldest event id to keep
const oldestEventId = new Date().getTime() - STREAM_LIFETIME * 1000


// add the event to the stream and expire old events
redisClient.xAdd(STREAM_KEY, '*', event, {
  TRIM: {
    strategy: 'MINID',
    threshold: oldestEventId
  }
})
```

```
redis.cloud:6379> XADD radio:events * icaoId A835AF ... MINID 1655410429000-0
"1655411024042-0"
```


RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node


STREAM
HASH


flight-events
node
flight-ui
NGINX
flight-api
node


api-gateway
NGINX


radio-consumer
node

STARTING NOW

```
// just start with recent events
let currentId = '$'

// read stream forever
while (true) {

  // wait at most a second for a result
  const result = await redisClient.xRead({
    key: STREAM_KEY, id: currentId
  }, {
    BLOCK: 1000,
    COUNT: 1
  })

  // loop if we have no results
  if (result === null) continue

  ...process the events...

}
```

```
redis.cloud:6379> XREAD COUNT 10 BLOCK 1000
  STREAM radio:events $
```

PROCESSING THE EVENT

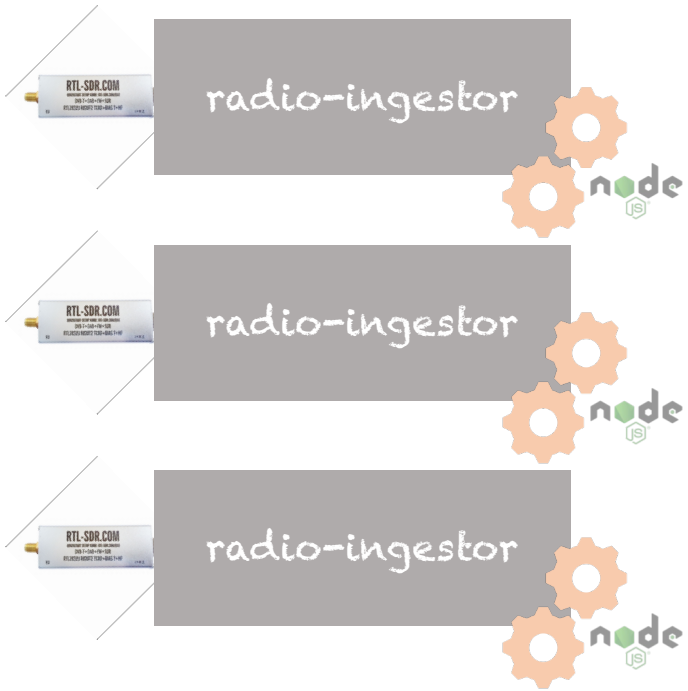
```
// create the object to set
const aircraft = {}

// set the always stuff
aircraft.icaoId = event.icaoId,
aircraft.dateTime = event.loggedDateTime,
aircraft.radio = event.radio

// set the sometimes stuff
if (event.callsign !== undefined) aircraft.callsign = event.callsign
if (event.altitude !== undefined) aircraft.altitude = event.altitude
if (event.latitude !== undefined) aircraft.latitude = event.latitude
if (event.longitude !== undefined) aircraft.longitude = event.longitude
if (event.velocity !== undefined) aircraft.velocity = event.velocity
if (event.heading !== undefined) aircraft.heading = event.heading
if (event.climb !== undefined) aircraft.climb = event.climb
if (event.onGround !== undefined) aircraft.onGround = event.onGround

// set the location for geo searches
if (event.latitude !== undefined && event.longitude !== undefined) {
  aircraft.location = `${event.longitude},${event.latitude}`
}
```

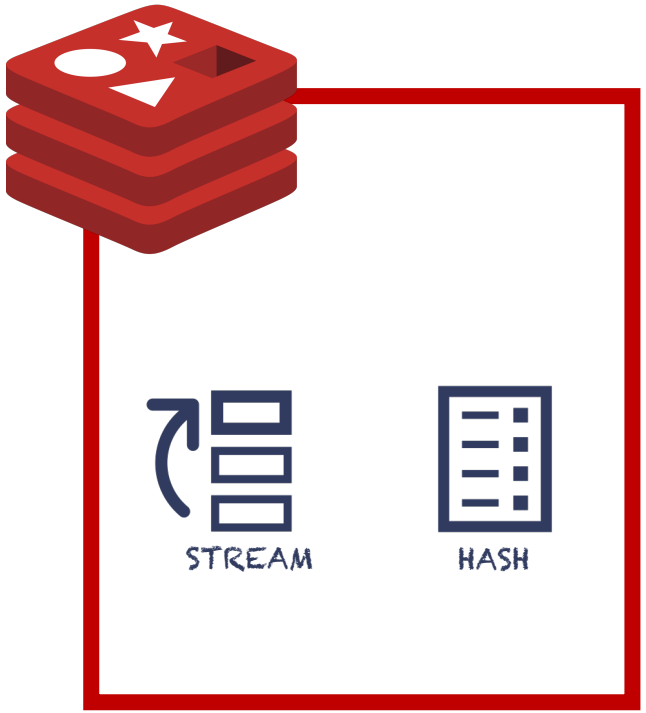
UPDATING REDIS

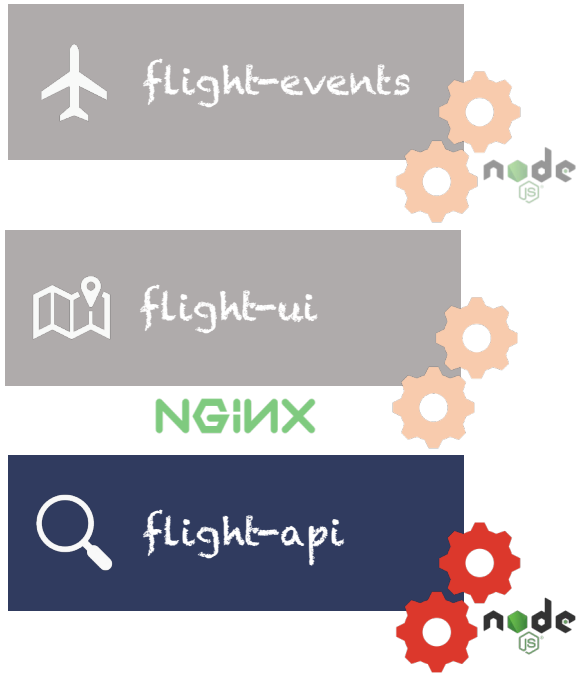
```
// set the data in Redis with an expiration
const key = `aircraft:${event.icaoId}`
redisClient.sendCommand([ 'JSON.MERGE', key, '$',
JSON.stringify(aircraft) ])
redisClient.expire(key, AIRCRAFT_DATA_LIFETIME)
```

```
redis.cloud:6379> JSON.MERGE aircraft:A835AF $ '{"icaoId": "A835AF",
"callsign": "N628TS"}'
redis.cloud:6379> EXPIRE aircraft:A835AF 3600
```

```
// update the current id so we get the next event next time
currentId = id
```


RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node

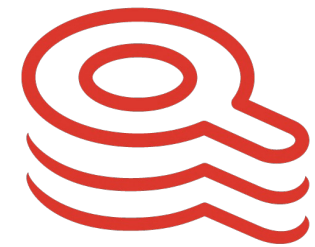

STREAM
HASH


flight-events
node
flight-ui
NGINX
flight-api
node


api-gateway
NGINX


radio-consumer
node

express



CREATING THE INDEX

```
// drop the index if it exists
try {
  await redisClient.ft.dropIndex('aircraft:index')
} catch (error) {
  if (error.message !== 'Unknown Index name') throw error
}

// create the RediSearch index
await redisClient.ft.create('aircraft:index', {
  'radio': SchemaFieldTypes.TAG,
  'icaoId': SchemaFieldTypes.TAG,
  'dateTime': SchemaFieldTypes.NUMERIC,
  'callsign': SchemaFieldTypes.TAG,
  'altitude': SchemaFieldTypes.NUMERIC,
  'latitude': SchemaFieldTypes.NUMERIC,
  'longitude': SchemaFieldTypes.NUMERIC,
  'location': SchemaFieldTypes.GEO,
  'velocity': SchemaFieldTypes.NUMERIC,
  'heading': SchemaFieldTypes.NUMERIC,
  'climb': SchemaFieldTypes.NUMERIC,
  'onGround': SchemaFieldTypes.TAG
}, { ON: 'HASH', PREFIX: 'aircraft:' })
```

EXPRESS YOURSELF

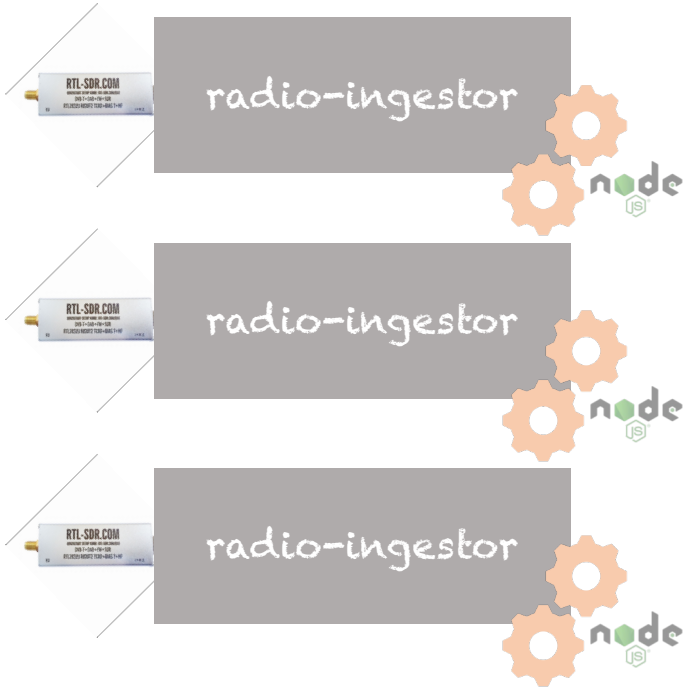
express

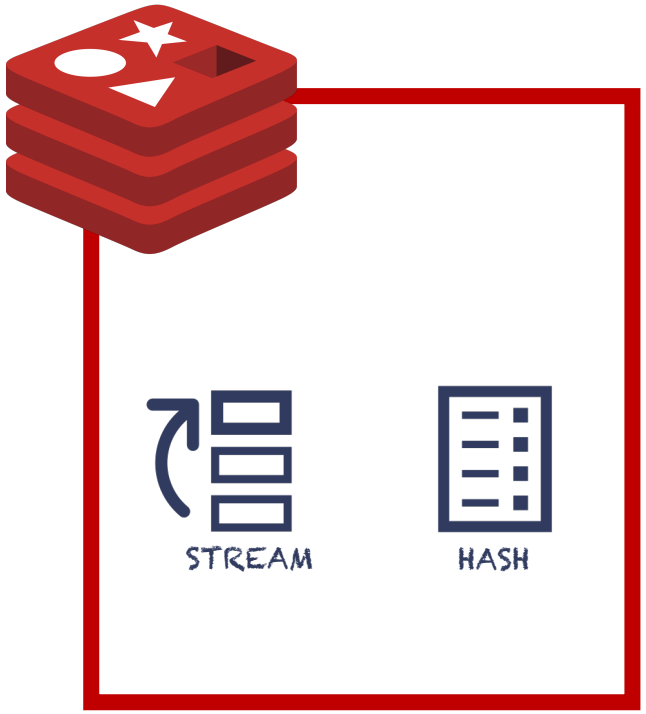
```
router.get('/', async (req, res) => {
  const result = await redisClient.ft.search(aircraftIndex, '*')
  const aircraft = convertMultipleResult(result)
  res.send(aircraft)
})

router.get('/icao/:id', async (req, res) => {
  const { id } = req.params
  const result = await redisClient.ft.search(aircraftIndex,
    `@icaoId:{${id}}`)
  const aircraft = convertSingleResult(result)
  res.send(aircraft)
})

router.get('/above/:altitude', async (req, res) => {
  const { altitude } = req.params
  const result = await redisClient.ft.search(aircraftIndex,
    `@altitude:[${altitude} +inf]`)
  const aircraft = convertMultipleResult(result)
  res.send(aircraft)
})
```

MORE INTERESTING QUERIES

express

```
router.get('/heading/:direction', async (req, res) => {
  const { direction } = req.params

  let query = '*'
  if (direction === 'north') query = ` @heading:[315 360] @heading:[0 45]`
  if (direction === 'east') query = `@heading:[45 135]`
  if (direction === 'south') query = `@heading:[135 225]`
  if (direction === 'west') query = `@heading:[225 315]`


  const result = await redisClient.ft.search(aircraftIndex, query)
  const aircraft = convertMultipleResult(result)
  res.send(aircraft)
})

router.get('/within/:radius/:units/of/:longitude/:latitude', async (req, res) => {
  const { longitude, latitude, radius, units } = req.params
  const result = await redisClient.ft.search(aircraftIndex,
    `@location:[${longitude} ${latitude} ${radius} ${units}]`)
  const aircraft = convertMultipleResult(result)
  res.send(aircraft)
})
```


RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node


STREAM
HASH

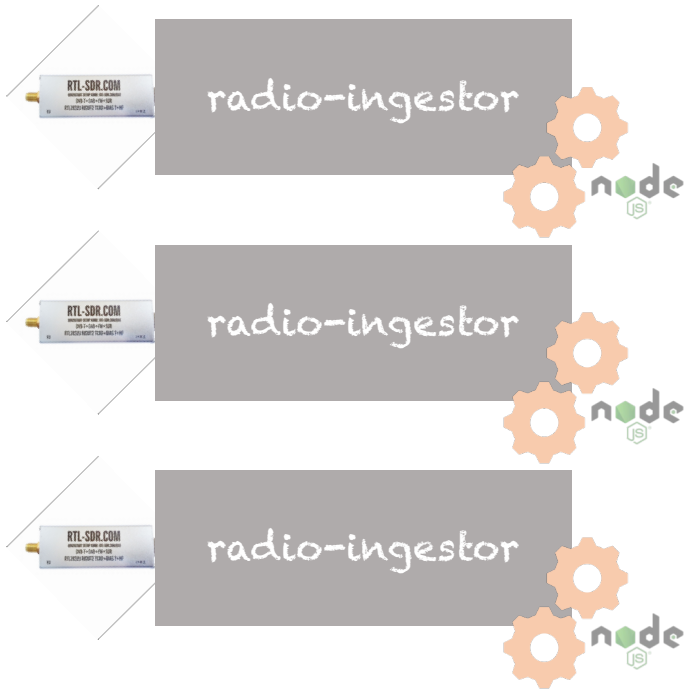

flight-events
node
flight-ui
NGINX
flight-api
node

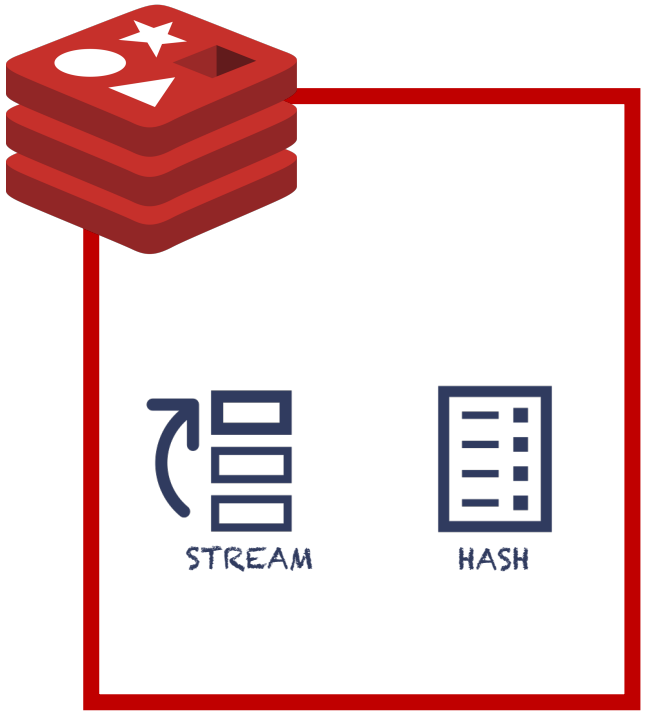

api-gateway
NGINX


radio-consumer
node

PUBLISHING EVENTS

```
// set up a basic web sockect server and a set to hold all the sockets
const wss = new WebSocketServer({ port: 80 })
const sockets = new Set()

// when someone connects, add their socket to the set of all sockets
// and remove them if they disconnect
wss.on('connection', socket => {
  sockets.add(socket)
  socket.on('close', () => sockets.delete(socket))
})

// read stream forever
while (true) {

  ... read stream and convert...

  // stringify and send to the all the listeners
  const json = JSON.stringify(event)
  sockets.forEach(s => s.send(json))
}
```


RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node

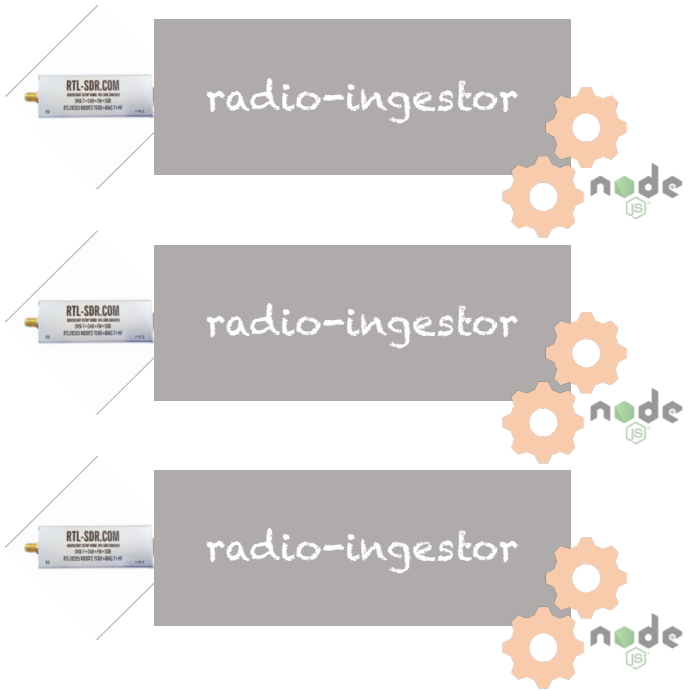

STREAM
HASH

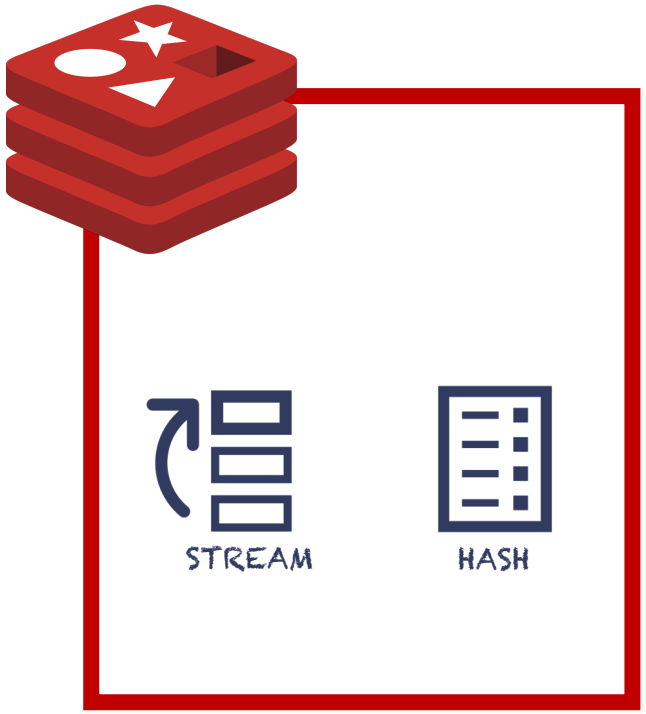

flight-events
node
flight-ui
NGINX
flight-api
node


api-gateway
NGINX


radio-consumer
node

MAKING A MAP


RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node
RTL-SDR.COM
radio-ingestor
node


STREAM
HASH


flight-events
node
flight-ui
NGINX
flight-api
node

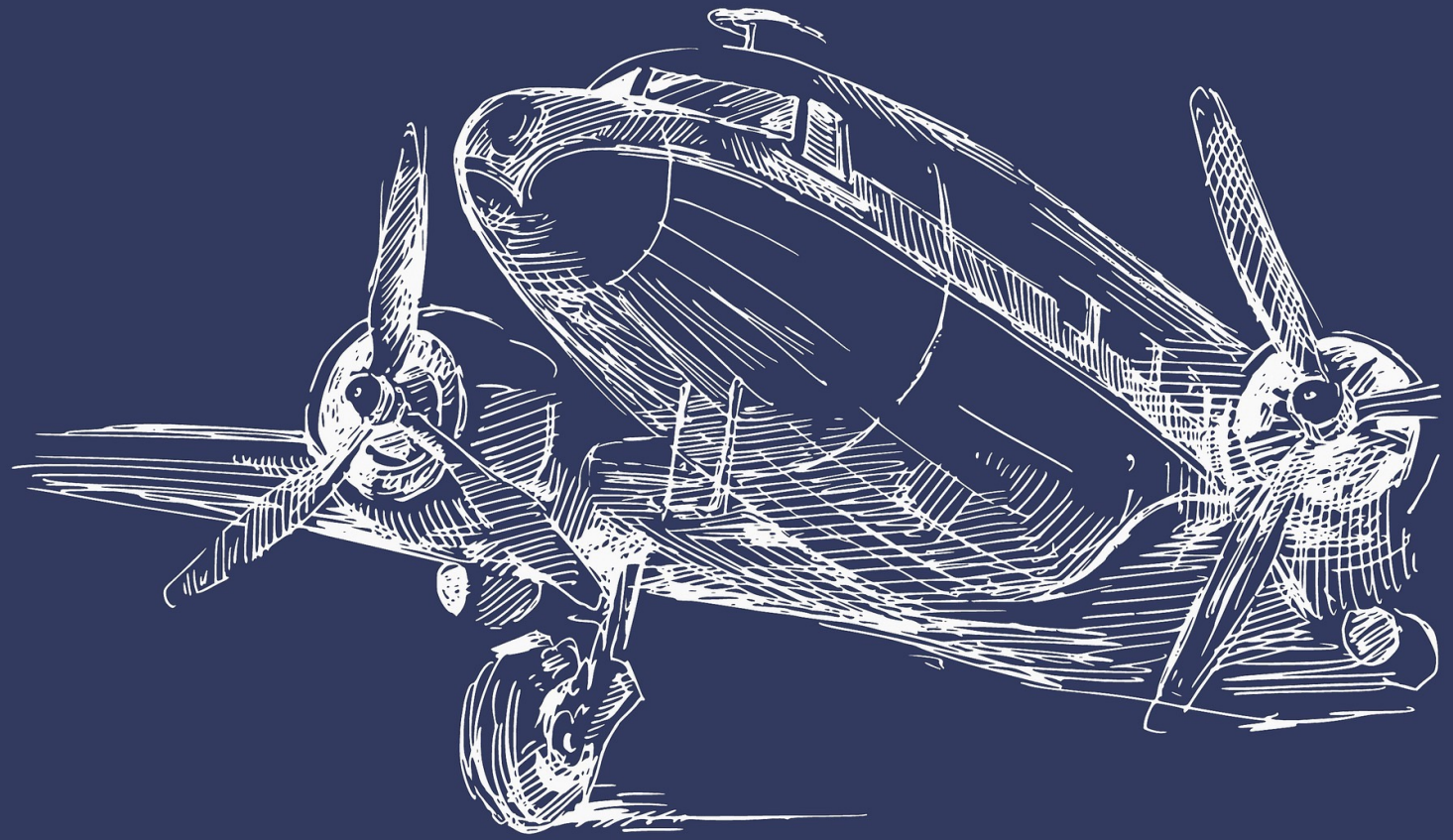

api-gateway
NGINX


radio-consumer
node

nginx.conf

```
events {
  worker_connections  4096;  ## Default: 1024
}
http {
  server {
    listen 80;
    location /aircraft {
      proxy_pass http://flight-api:80/aircraft;
    }
    location /events {
      proxy_pass http://flight-events:80;
      proxy_http_version 1.1;
      proxy_set_header Upgrade $http_upgrade;
      proxy_set_header Connection "Upgrade";
      proxy_set_header Host $host;
    }
    location / {
      proxy_pass http://flight-ui:80;
    }
  }
}
```

DEMO



redis

RESOURCES

Redis Stack
https://redis.io/docs/stack/

SDR Software
https://airspy.com/download/
https://gqrx.dk/
https://cubicsdr.com/
https://www.sdrangel.org/

AIS Trackers
https://www.vesselfinder.com/
https://www.marinetraffic.com/

ADS-B Trackers
https://flightaware.com/
https://globe.adsbexchange.com/

dump1090
https://github.com/antirez/dump1090

REDIS DISCORD SERVER

https://discord.gg/redis

REDIS UNIVERSITY

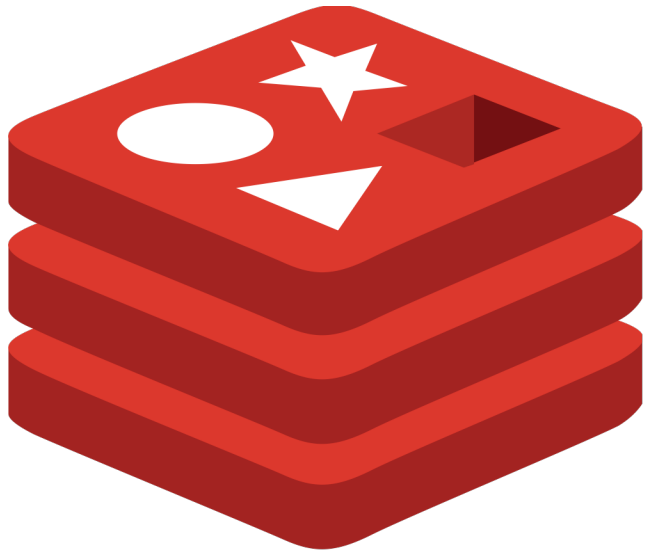
https://university.redis.com/

REDIS CLOUD

https://redis.com/try-free

SLIDES, CODE, AND WHATNOT

https://github.com/redislabs-training/tracking-aircraft-edu



redis