The Maritime Cloud Almanac Technical details

Kasper Nielsen (DMA) <kasperni@gmail.com>

Identifying other maritime actors and the information services provided in an area will facilitated by the digital publication 'The Almanac'.

This publication is an offline version of the public part of the Identity and Service Registry, acting as a 'white pages/yellow pages phone book'. The Almanac can be updated, whenever a suitably low cost or flat rate data connection is available. Using The Almanac, you can automatically lookup the MMSI number for a DSC call, VHF working channel or e-mail address, phone numbers or other contact information of a VTS center, Port, the nearest MRCC or another ship you may wish to contact. Or you may lookup which providers of any specific information service are available along a planned voyage.

For instance, the list of MRCC's are supposed to be listed in the IMO SAR plan, however experience documents that securing that this vital document is updated, is difficult. Enabling each MRCC to maintain their own contact information in the Maritime Identity Registry will facilitate more up-to-date information to be readily available.

This chapther describes how the almanac is implemented.

Requirements

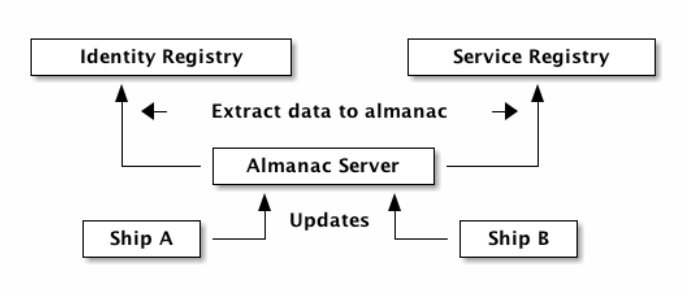
Before described the actual implementation. We just want to quickly run over the requirements that let to the current design.

- **Bandwifth,** The almanac and updates to it must be stored in a bandwidth saving format. And be as compressed as possible to save bandwidth when downloading it.
- **Delta updates,** Updates to the almanac must be able to be downloaded without downloading the complete copy of the new almanac.
- **Secure,** It must be possible to verify that updates to the almanac is valid update published by the central almanac server.
- **Distributed design,** While a central server is needed to make sure only one version of the almanac exists. Clients should be able to update the almanac from other clients that have already received the updates.
- **Flexible updating,** It must be possible to distribute updates to the almanac both on a physical medium such as an USB sticks. Or as a update over a computer network or radio based protocol.
- **Frequent Updates,** The almanac is not indented to be a real time indicator of exactly is in the Identity and Service Registry. On the other hand clients should be allowed to upgrade the almanac

multiple times a day if they want to.

Overall Design

Here is the basic design of the almanac.



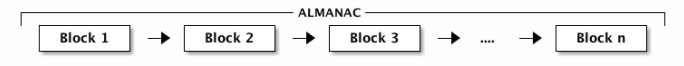
We have a single almanac server that gather updates from the service and identity registry. Each ship can then query the server for updates whenever a suitable connection is available.

The access to the almanac server will typically be done through the almanac client that is part of the reference implementation of the maritime cloud. This almanac client also takes of storing updates and provides an interface for querying the stored almanac. Equipment manufactures would not need to implement any details of the protocol. As all access to the actual data would go through high level interfaces available from the almanac client.

Blocks

Data is permanently recorded in the almanac through files called blocks. A block is a record of some or all of the most recent changes to the Service and Identity Registry that have not yet been recorded in any prior blocks. They could be thought of like the individual pages of a city recorder's recordbook (where changes to title to real estate are recorded) or a stock transaction ledger. Each block has a unique id that increments by 1 for every new block. The initial block starting with 1.

New blocks are added to the end of the block chain, and once written, are never changed or removed. So the almanac is essentially append-only. Each block memorializes what took place immediately before it was created.



Every 10-30 minute the almanac server queries the Identity and Service Registry for any updates that been performed since the last block was generated. It then takes all these changes and puts them into single binary block. Giving it the id: *lastId + 1*

Each block is then signed by the almanac server to make sure no one else but the almanac server publishes updates. Before a client adds a block to its blockchain the signature of the block is verified. To make sure it is an official update by the central almanac server. An added benifit it that blocks can

be distributed without requiring a central server.

Distributed design

There are no requirements for a ship to get the updates directly from the server.

If two ships meet somewhere. They can exchange their latest updated block id. For example, Ship A has got 23 as the latest received almanac block. And ship B has got 34 as the latest received block id. Ship A can then ask Ship B to send blocks 24-34 over, for example, VHF. Ship A verifies the signature of each block it receives from Ship B to make sure it is an official update by the central Almanac Server. If the signature of the block does not match the public key of the Almanac Server, Ship A will reject the block. And try to get the block at some other point.

Companies can also choose to use their own distribution mechanism to their ships. A leisure sailer might only use his boat in the summer. So a harbour might have an USB stick that is regular updated that he can borrow and plug into his equipment. The "almanac client" will then make sure to copy only those blocks that have been updated since last summer from the USB stick. Which is basically everyting from the last received block id to the maximum block id on the USB stick.

No matter what the basic almanac protocol does not care. Because each block is signed. And each block has a unique incrementing id. Any client can verify the integrity of an update no matter where it comes from. And people can choose to update the almanac in whenever way they want to.

Almanac Server interface

The Almanac server has 3 simple REST based methods that can be invoked by clients.

- ***getLatestBlock()***, returns the number (integer) of the latest available block.
- ***getBlock(int id)***, returns the contents (binary) of the specified block.
- ***getBlocks(int fromId, int maximumNumberOfBlocks)***, returns the contents (binary) of the specified blocks.

Block format

The following MSDL describes the format of each block as published by the central almanac server.

TODO

```
message Block {
    /** The id of the block. Increment by one every time. */
    1: int id;

    /** The time stamp of the block generation. */
    2: timestamp generationTime;

    /** A signature of the block (id^timestamp^contents). */
    3: Binary signature;

    /** The actual updates. */
    4: Binary blockUpdate;
}
```

```
message BlockUpdate {
    /** Updates to the identity. */
    list<IdentityUpdate> identityUpdates;
    list<ServiceUpdate> identityUpdates;
}
```

```
message IdentityUpdate {
   1: binary[] id;
   2: binary[] publicKey;
   3: text name;
   //identity type? mmsi number
}
```

```
message ServiceUpdate {
   ???
}
```