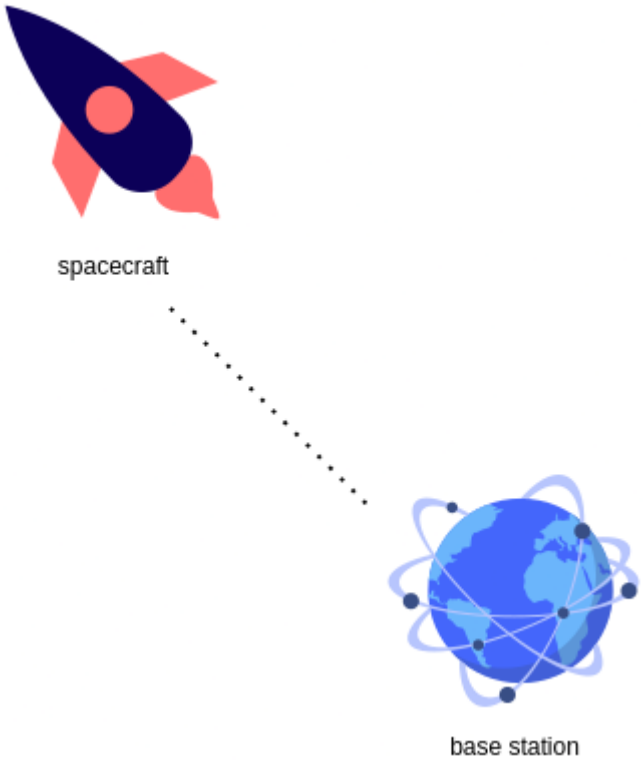
Spaceship communication system

Part 2

The communication system between a base station and a spaceship allows the base station to issue commands to spaceships, while the spaceships report back their position to a monitor so the base station operator can check it.



Each message sent is a bytes array. There are two types of messages

- command messages
- reporting messages

The first byte of the array is the type of message. Its value is 0 for reporting messages and 1 for command messages.

Command messages $b_0b_1b_2b_3$:

b_0 - message type: for command messages, this is 1
b_1 - spacecraft id: it is the id of the spacecraft that the command is issued to
b_2 - move direction: It takes one of the values 1,2,3,4 where 1 corresponds to upwards, 2 to right, 3 to downwards and 4 to left
b_3 - move distance: it takes one of the values 1,2,3,4 and it corresponds to the number of places that a spaceship shall move

For example command message 1123 is a command message for spacecraft with id 1 to move to the right for 3 places.

Reporting messages $b_0b_1b_2b_3$:

b_0 - message type: for reporting messages this is 0
b_1 - spacecraft id: it is the id of the spacecraft that sends the message
b_2 - position x: is the position of the spacecraft on the x-axis
b_3 - position y: is the position of the spacecraft on the y-axis

For example, reporting message 0245 is a reporting message from spacecraft with id 2 which is at position (4,5).

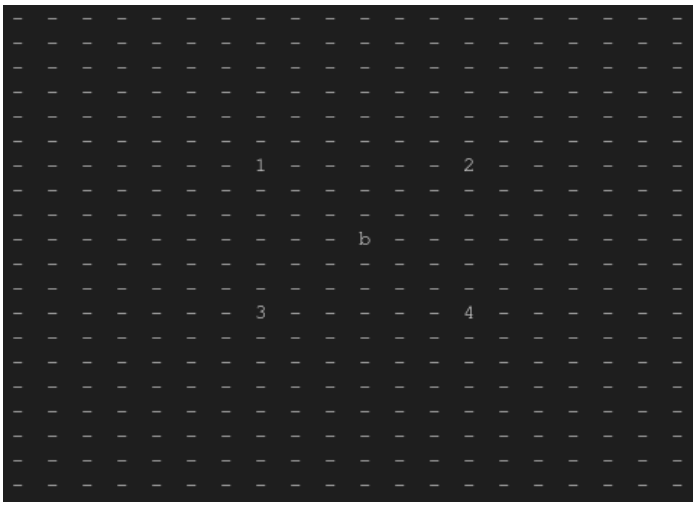
Task 1.1

You shall get yourself familiar with the system by operating it according to the following instructions:

a. Execute environment.sh and leave it running on a terminal (you have to check this only for debugging purposes)
   ./environment.sh 1.1
b. Execute monitor.py and let it run in a terminal (this will create a space map and show the spacecrafts)
   python3 monitor.py
c. Launch 4 spaceships by executing
   python3 launch_spacecrafts.py 4

d. Send commands to the spaceships by executing
   python3 base_station.py

You should be able to arrange the 4 spaceships in the following way:



Submit: A screenshot with the required arrangement of the spaceship

Task 1.2

You have to modify the code in the file spaceship_inst.py, which corresponds to the code that each one of the spaceships runs. You shall add logging functionality to the spaceships as follows.

- When the spaceship launches it shall create a file under the name id.log (where id is the id of the spaceship).
- Every time the spaceship accepts a command message that is destined for its id, it shall append to the log file a line of the following format:

timestamp,direction, distance

In order to achieve that you can modify the function **handle_msg** in the file spacecraft_inst.py (you can also add imports of any required libraries in the same file).

Submit: The modified spacecraft_inst.py file

Task 1.3

Given the fact that the space environment is a difficult environment to communicate through it is common to have errors in the communication. Setup the system in a more realistic way that multiple errors occur in the communication between the base station and the spacecrafts by running:

./environment.sh 1.3
python3 monitor.py
python3 launch_spacecrafts.py 4
python3 base_station.py

In this case, if you try to play around with the base station and issue commands to the spaceships you will see that many times these commands are accidentally changed during communication and thus spaceships either ignore commands or misinterpret those.

You need to modify the file spacecraft_inst.py and the base_station.py files, in order to add an integrity checking mechanism to the communication. Specifically:

- The base station shall add a signature (of 4 bytes) to the message through the use of the sha256 algorithm.
- The spacecraft shall check that signature and if it is valid it shall forward the message, either wise it shall reject it.

In order to achieve that you can modify the function **handle_msg** in the file spacecraft_inst.py and the **prepare_msg** function in the base station.py file (you can also add imports of any required libraries in the same files).

Submit: The modified spacecraft_inst.py and base_station.py files

Task 1.4

When a message is rejected by the spaceship, the base station never gets to know that. You are required to build an acknowledgement mechanism through which the spaceship is going to respond to the base station about what has happened with the messages received. Define a new type of message that is intended for acknowledgement. It shall include only the id of the spaceship and a 0 or 1 value according to whether the command message has been accepted or rejected.

Each time a spacecraft receives a command message it sends out an acknowledgement message to inform the base station of what has happened with the command message. You have to modify the function **handle_msg** in the file spacecraft_inst.py, so it sends out such messages. You can find the transmission functionality in the base_station.py file.

You have also to create a new script **listen_ack.py** that listens for such messages in the base station and prints the corresponding details. You can find the functionality of receiving messages in the spacecraft_inst.py file.

Submit: The modified spacecraft_inst.py file and the new listen_ack.py file.

Task 2.1

Given the fact that the space environment is a difficult environment to communicate through, it is common to have counterparts trying to interfere with communication and change messages. On top of the approach you have followed in task 1.3, you have to make the communication robust against intentional manipulation on top of accidental errors. Setup the system in a more realistic way that multiple errors (intentional/accidental) occur in the communication between the base station and the spacecrafts by running:

./environment.sh 1.3
python3 monitor.py
python3 launch_spacecrafts.py 4
python3 base_station.py

You need to modify the file spacecraft_inst.py and the base_station.py files, in order to add an authenticated integrity checking mechanism to the communication.
Specifically:

- The base station and the spaceships all share a common secret of 16 bytes.
- The base station shall add a signature (of 4 bytes) to the message through the use of the HMAC algorithm (using the pre-shared secret).
- The spacecraft shall check that signature and if it is valid it shall forward the message, either wise it shall reject it.

In order to achieve that you can modify the function **handle_msg** in the file spacecraft_inst.py and the **prepare_msg** function in the base station.py file (you can also add imports of any required libraries in the same files). You have to also add a hard coded secret in the code of both sides.

Submit: The modified spacecraft_inst.py and base_station.py files

Task 2.2

You shall add encryption to the communication between the two sides. Assuming that the two sides (the base station and the spacecrafts) share a common AES key you have to make the base station -> spacecraft communication encrypted through an AES PKCS#7 scheme.

Submit: The modified spacecraft_inst.py and base_station.py files

Task 2.3

You shall add hybrid encryption to the communication between the two sides. When a new spacecraft is launched a pair of RSA keys is created and the public key is stored in a text file that can then be read by the base station. The base station reads the proper file before issuing a command to a spacecraft and uses RSA-AES hybrid encryption to send the command. The spacecraft decrypts the command upon receiving it.

Submit: The modified spacecraft_inst.py and base_station.py files

Task 2.4

You shall add public key signatures to the command. You have to create a pair of RSA keys for the base station. The spacecrafts carry the public key of the base station. The base station signs the messages with the private key of the pair and the spacecrafts validate the signature before executing the commands.

Submit: The modified spacecraft_inst.py and base_station.py files