COMMAND

PROFESSIONAL EDITION

USER MANUAL

Command Professional Edition

User Manual

Version 2.4

Last Updated October 13, 2024

Table of Contents

INTRODUCTION

Command Professional Edition v2 (CPE 2) is the professional-oriented superset of **Command: Modern Operations (CMO),** MatrixGames' flagship commercial wargame of modern cross-domain military operations. CMO is the successor to Command: Modern Air/Naval Operations, the wargame on which CPE 1.x is based. General information about CMO can be obtained on its product page at MatrixGames: https://www.matrixgames.com/game/command-modern-operations

This manual acts as an addition to the existing documentation for CMO. The commercial CMO user manual is bundled together with the software and is also accompanying this CPE manual.

The CMO manual can also be located here:
https://cdn.akamai.steamstatic.com/steam/apps/1076160/manuals/CMO_manual_EBOOK.pdf

As a rule, this manual avoids referencing features that are already present on the commercial version of CMO, as it is assumed the reader is already familiar with them. There are exceptions to this rule, for example when addressing a feature that exists in the commercial version but is further enhanced on the professional and/or academic edition.

Throughout this manual, the terms "Command Professional Edition", "Command PE" and "CPE" are used interchangeably to refer to the pro-oriented superset of CMO.

Users are highly encouraged to read the release notes of past, present and future CPE releases in order to become familiar with changes and additions to the software.

Recommended hardware & OS specs

- **Operating System:** Windows 10+
- **Processor:** Quad-core, 4th-generation Intel or equivalent
- **Memory:** 8+ GB RAM
- **Graphics:** Direct-X 11 compatible GeForce GT1030+ or equivalent *(less powerful GPUs may also work but with inconsistent performance characteristics)*
- **Direct-X:** Version 11
- **Disk Storage:** (Core install) 40 GB available space initially; likely to grow as map tiles are downloaded on-demand and cached locally.

Checking hardware

It is possible to check if a machine's processor and GPU support the minimum specifications by typing "dxdiag" into the Start Menu search bar and launching the dxdiag program.

The System tab of the dxdiag (DirectX Diagnostic Tool) program should indicate in parentheses next to "Processor:" that the processor has at least 4 logical CPUs and indicate next to "Memory:" that the system has at least 8192MB of RAM. The Display tab's Device section should indicate next to

"Display Memory (VRAM):" that the GPU has at least 2048MB of VRAM and the Driver section should indicate next to "Feature Levels:" that 11_0 is one of the feature levels supported.

Launching File Explorer and clicking on This PC will show available space remaining on all connected drives. The installation drive should have at least 80GB available prior to installation. The optional HD Pack requires an additional 160GB of disk space.

Installation & Hardware Notes

Command PE v2.3+ uses a new renderer based on DirectX 11. DirectX 11 is included in the base install of Windows 10 and 11, so it is not necessary to install any version of DirectX separately from Command PE.

Command PE v2.3+ requires .NET Framework 4.8 to be present. If the machine does not have .NET 4.8 or higher, it will be installed automatically during installation.

Command PE quickly loads cached map tiles from local disk and network if they are available. We recommend using fast SSDs to take advantage of this. Likewise, if CPE is installed on a connected computer and access to streaming maps is desired, we recommend a fast Internet connection.

Machines with multiple GPUs (e.g. laptops with discrete GPUs) that encounter slow performance or UI issues should explicitly force their system to use the discrete GPU for running CPE (Command will try to do this automatically but is not always guaranteed to succeed).

If no hardware GPU is found at startup, CPE will fall back to using a software renderer. However, GPU rendering is much faster than software rendering. Using the software renderer should be avoided if at all possible as the software renderer is an emergency backstop only.

Command PE can use as much RAM as a machine has available (Pre-v2.3 versions of CPE were limited to using approximately 3GB of RAM). 8GB is sufficient for most scenarios, but more is recommended for very large scenarios.

Note on VM/Cloud deployments

Command PE can be deployed on any virtual machine or cloud operations platform (such as Azure, AWS etc.) that offers a Windows (desktop or server) OS environment as a running host. However, care must be taken to ensure that the hardware facilities exposed to CPE within such an environment match or exceed the minimum hardware requirements of an equivalent physical host. This is particularly true with regards to the GPU, as some VM & cloud solutions expose a software-mode or otherwise underwhelming GPU. If running inside a virtual machine, enabling access to GPU-assisted rendering at the virtual machine level can be very useful for better map/UI performance.

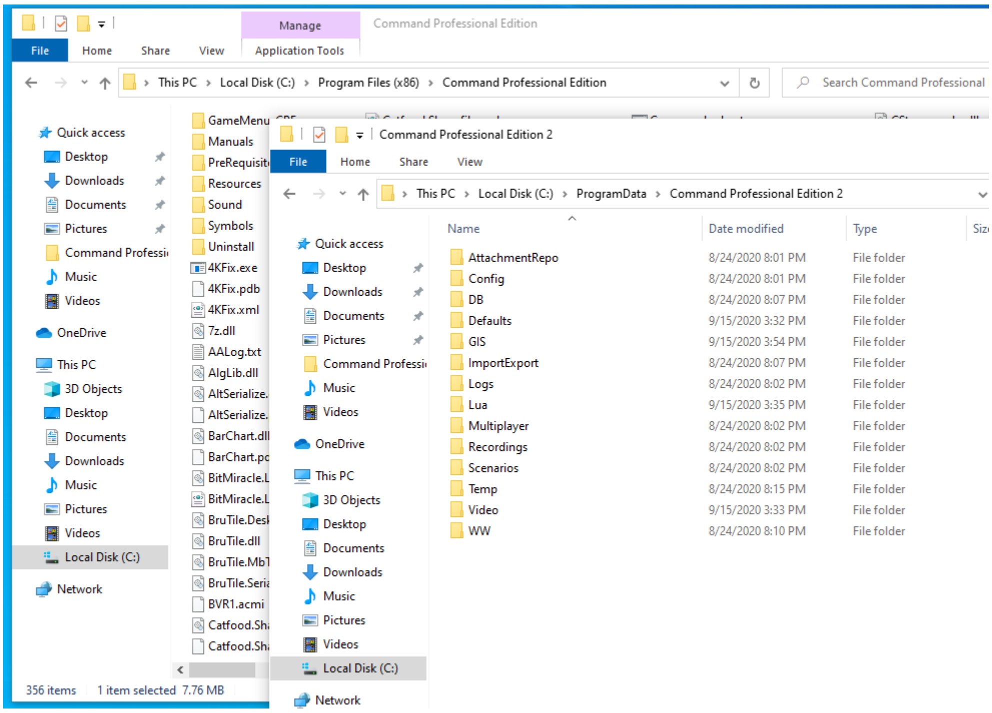
INSTALLATION & FILE/FOLDER PATHS

By default, CPE is installed in the folder path “**[Program Files x86]\Command Professional Edition**”, however you can change this path during installation.

In some Windows OS installations, particularly in high-security systems where the user account is severely restricted, executed programs are prohibited from writing on the Program Files folder and any subfolders within it. This is problematic in the case of CPE, as the program would not be able to wriute any information to disk (e.g. export any analysis data).

Windows provides a dedicated folder for non-user-specific writable application data, the “Program Data” system folder (see here for a brief description: https://www.howtogeek.com/278562/what-is-the-programdata-folder-in-windows/)

To make CPE suitable for installation and use in such environments, all the program folders in which data is written are located on a dedicated “top-level writable folder” stored within the Program Data tree. This folder’s path is: **“[Program Data]\Command Professional Edition\Command Professional Edition 2”**. This is an example of this folder, alongside the main installation folder, after a default installation on system drive C:



Name	Date modified	Type	Siz
AttachmentRepo	8/24/2020 8:01 PM	File folder	
Config	8/24/2020 8:01 PM	File folder	
DB	8/24/2020 8:07 PM	File folder	
Defaults	9/15/2020 3:32 PM	File folder	
GIS	9/15/2020 3:54 PM	File folder	
ImportExport	8/24/2020 8:07 PM	File folder	
Logs	8/24/2020 8:02 PM	File folder	
Lua	9/15/2020 3:35 PM	File folder	
Multiplayer	8/24/2020 8:02 PM	File folder	
Recordings	8/24/2020 8:02 PM	File folder	
Scenarios	8/24/2020 8:02 PM	File folder	
Temp	8/24/2020 8:15 PM	File folder	
Video	9/15/2020 3:33 PM	File folder	
WW	8/24/2020 8:10 PM	File folder	

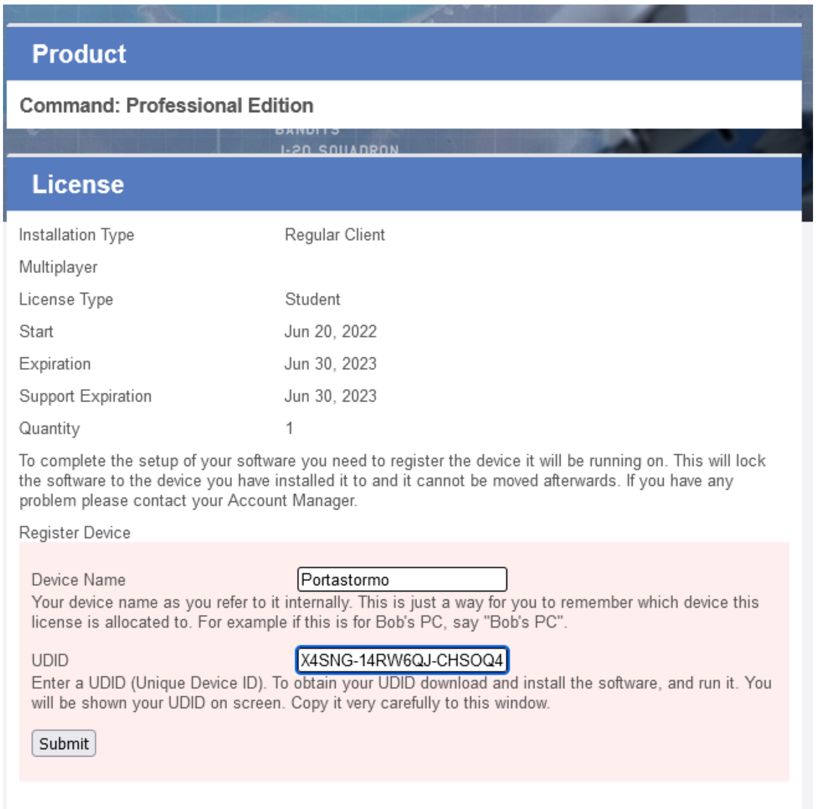
As demonstrated, while read-only data (incl. the application binaries) still reside on the [Program Files] folder, the writable folders have instead been moved under the [Program Data] tree. This is where you may locate generated logs, analysis-borne files, recordings etc.

LICENSE REVOKE

If desired, you can revoke an existing per-machine license and have it transferred to a different machine. To do so, contact the support team through the MPS helpdesk system.

(OPTIONAL) Requesting a new license from the website

To request a new license just fill the text boxes and click “Submit”:



Product

Command: Professional Edition

License

Installation Type	Regular Client
Multiplayer	
License Type	Student
Start	Jun 20, 2022
Expiration	Jun 30, 2023
Support Expiration	Jun 30, 2023
Quantity	1

To complete the setup of your software you need to register the device it will be running on. This will lock the software to the device you have installed it to and it cannot be moved afterwards. If you have any problem please contact your Account Manager.

Register Device

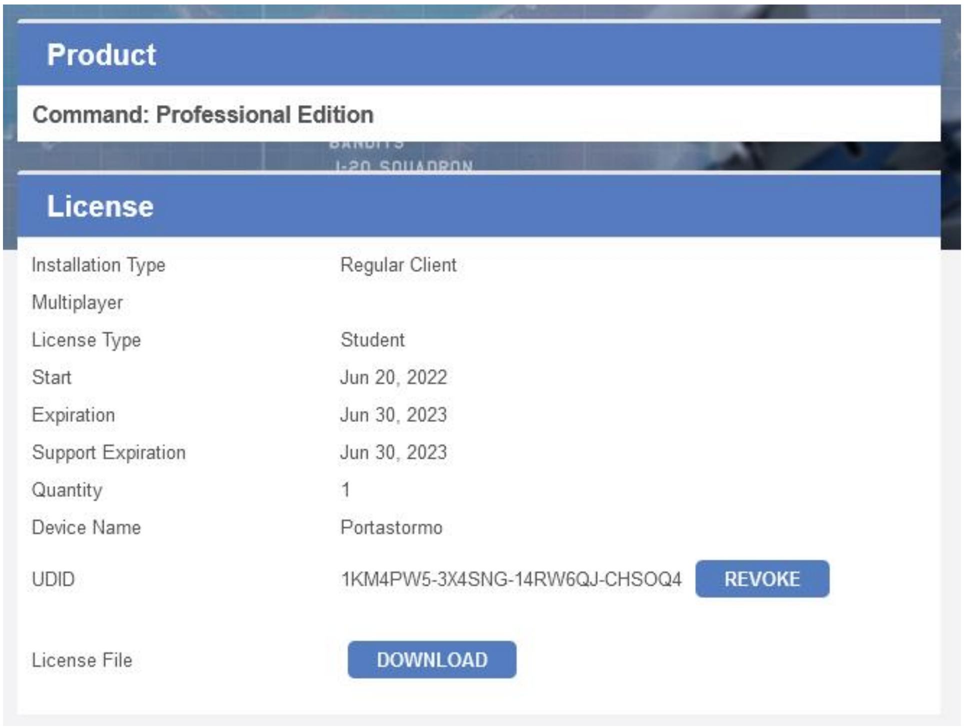
Device Name: Portastormo

Your device name as you refer to it internally. This is just a way for you to remember which device this license is allocated to. For example if this is for Bob's PC, say "Bob's PC".

UDID: X4SNG-14RW6QJ-CHSOQ4

Enter a UDID (Unique Device ID). To obtain your UDID download and install the software, and run it. You will be shown your UDID on screen. Copy it very carefully to this window.

A new license will be issued for the chosen machine:



Product

Command: Professional Edition

License

Installation Type	Regular Client
Multiplayer	
License Type	Student
Start	Jun 20, 2022
Expiration	Jun 30, 2023
Support Expiration	Jun 30, 2023
Quantity	1
Device Name	Portastormo
UDID	1KM4PW5-3X4SNG-14RW6QJ-CHSOQ4
License File	

WEGO MULTIPLAYER

(Available in PE)

CPE offers the ability to host and run a WEGO-type multiplayer session. This section describes the steps to accomplish this.

Basic Setup

To successfully set up and play Command Multiplayer (hence described as CMP), at least two computers are required. The computers must be connected over a TCP/IP network. All computers must be capable of running CPE.

CMP uses a client/server architecture. One computer will act as a server, while the rest of the computers will act as clients. The server computer can also be running a client instance at the same time.

Scenario Selection

CMP can be used with any scenario made for command, as long as the scenario has more than one side.

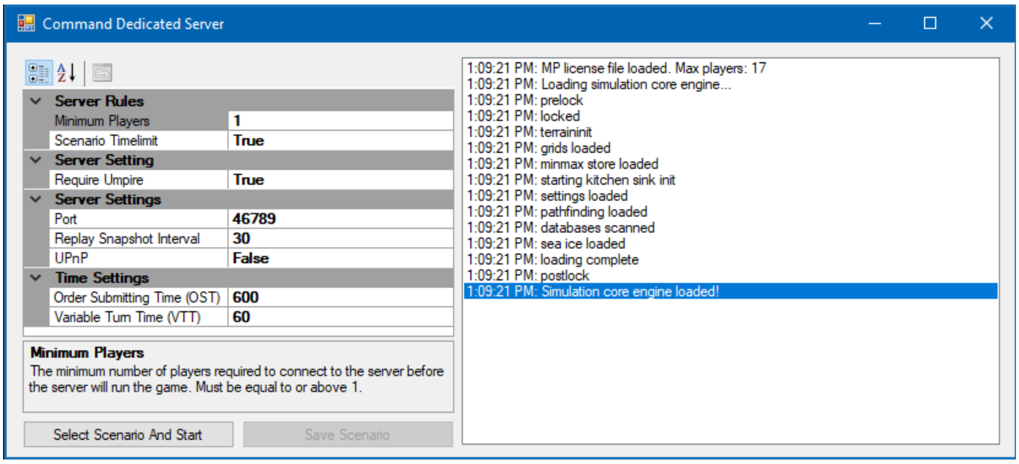
Network Setup

CMP uses a client/server architecture, this means the clients only need to connect to the server. For most LAN applications of CMP, putting all the computers on the same network and also making sure that no computers firewall is preventing CMP from being able to connect is sufficient to run CMP successfully.

Server Setup

Command Multiplayer has a dedicated server application that is distinct from Command Professional Edition's usual executable. The dedicated server application is started by executing the file CommandWEGOServer.exe. The server requires that a dedicated multiplayer license file (License_MP.dat) be present when starting up, otherwise it will display a licensing error message and exit.

Launching the dedicated server presents the user with the following window:



Server Rules	
Minimum Players	1
Scenario Timelimit	True
Server Setting	
Require Umpire	True
Server Settings	
Port	46789
Replay Snapshot Interval	30
UPnP	False
Time Settings	
Order Submitting Time (OST)	600
Variable Turn Time (VTT)	60

Minimum Players
The minimum number of players required to connect to the server before the server will run the game. Must be equal to or above 1.

```
1:09:21 PM: MP license file loaded. Max players: 17
1:09:21 PM: Loading simulation core engine...
1:09:21 PM: prelock
1:09:21 PM: locked
1:09:21 PM: terraininit
1:09:21 PM: grids loaded
1:09:21 PM: minmax store loaded
1:09:21 PM: starting kitchen sink init
1:09:21 PM: settings loaded
1:09:21 PM: pathfinding loaded
1:09:21 PM: databases scanned
1:09:21 PM: sea ice loaded
1:09:21 PM: loading complete
1:09:21 PM: postlock
1:09:21 PM: Simulation core engine loaded!
```

The left-hand side of this window is dedicated to server settings:

- **Minimum players:** This sets the minimum number of players that have to join a session before the scenario can start.

- **Scenario time limit:** If set to “True”, the server will respect the end of the scenario time limit defined in the scenario file. Therefore, the server will not permit gameplay past the end of the scenario. If set to “False”, the server will allow the gameplay to pass the end of the preset time limit in the scenario. It will ignore any set time limit and just continue to play; for some scenarios this is not desirable.

- **Port:** This is the TCP/IP port that is used to listen to new clients, the setting needs to be set globally across all computers, server and client alike.

- **Require umpire:** If set to “True”, the Umpire role must be taken by a client before the game session can begin. Before and after the execution phase of a turn, the Umpire has the ability to adjudicate and alter the state of the scenario with the full power of the scenario-editor tools. If set to “False”, once all players commit their orders, the game immediately executes the turn and returns control to the players without any manual adjudication.

- **Replay snapshot interval**: This is the interval in seconds in which the server will display a visual snapshot of what is happening in-turn for all the clients. For example, if set to 5, then then server will present to the clients successive snapshots of “what is happening” mid-turn every 5 seconds (of scenario time, not actual time). For large-duration turns (e.g. 10 minutes) a small second snapshot interval is not suitable, as this will result in too much processor time and bandwidth taken up with sending snapshots from the server to the clients. A value of 0 will completely disable the “replay snapshot” feature (this speeds up play speed significantly).

- **UPnP:** This tells the server to attempt to get a external port using a compatible router. This is an advanced feature and not necessary to be used unless the network geometry and administrator requires the use of UPnP.

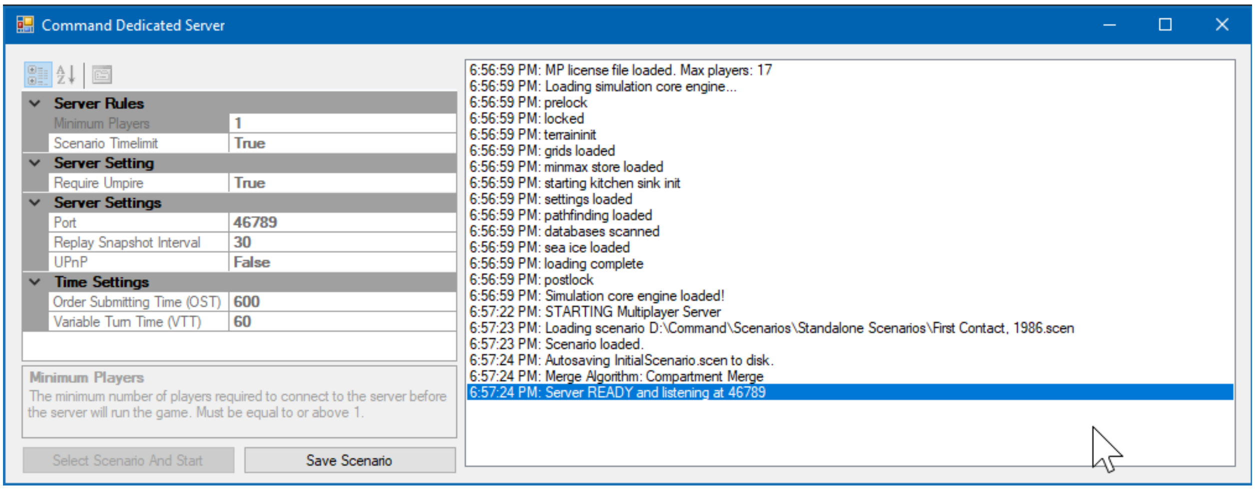
- **Order Submitting Time (OST):** This is the real time (in seconds) available to the player to evaluate the situation and issue orders. This can be set to a very large value to allow the player to carefully consider different strategies (e.g. in an MBX context), or a very short value to force the player to make split-second decisions, or anything in between.

- **Variable Turn Time (VTT):** This is the simulated time (in seconds) that each turn takes.

If the game session is controlled by an umpire, the OST and VTT settings can be changed during the pre- and post-turn adjudication steps. This allows great flexibility in the playout of the game session, as both the order-phase and turn-phase times can be adjusted for large-timespan events (e.g. non-eventful transit), or short-span action (weapon engagements etc.) or anything in between. If no umpire is set, these values are set only prior to loading the desired scenario.

On the right-hand side of this window, a log of events that occur on the server is listed. In the screenshot above, there are a number of diagnostics statements that show the server starting up.

To start the server one must select a scenario; the **"Select Scenario And Start"** button allows one to select a scenario. After the scenario is selected, it is loaded and the server starts to accept connections. **The server does NOT accept connections until a scenario is loaded.**



Server Rules	
Minimum Players	1
Scenario Timelimit	True
Server Setting	
Require Umpire	True
Server Settings	
Port	46789
Replay Snapshot Interval	30
UPnP	False
Time Settings	
Order Submitting Time (OST)	600
Variable Turn Time (VTT)	60

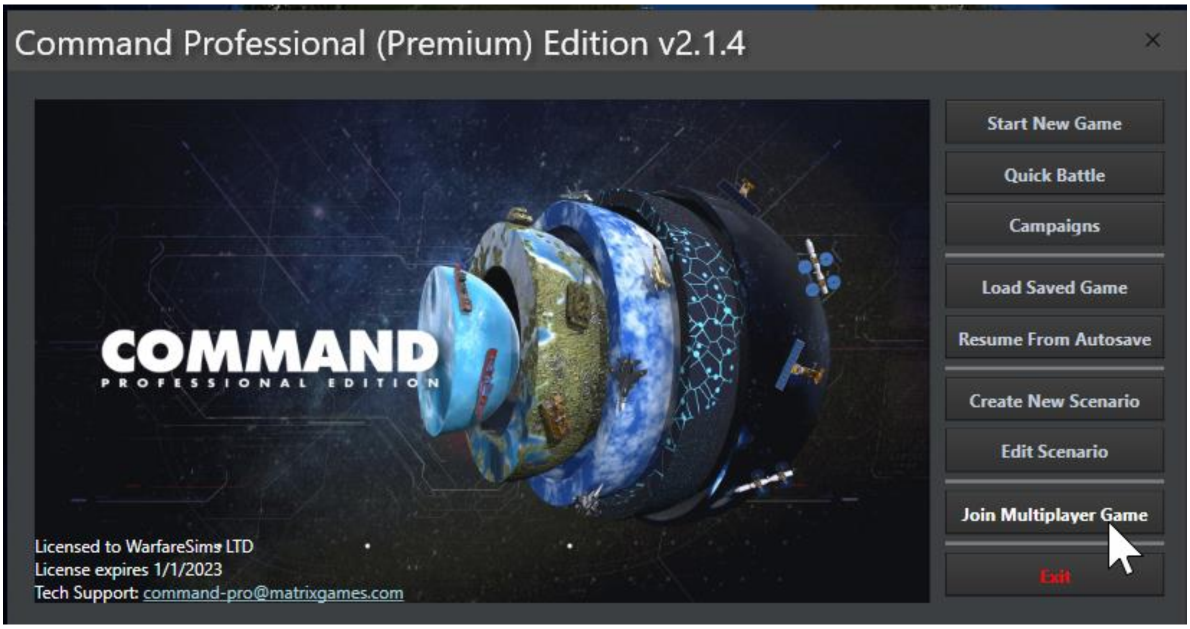
Minimum Players
The minimum number of players required to connect to the server before the server will run the game. Must be equal to or above 1.

```
6:56:59 PM: MP license file loaded. Max players: 17
6:56:59 PM: Loading simulation core engine...
6:56:59 PM: prelock
6:56:59 PM: locked
6:56:59 PM: terraininit
6:56:59 PM: grids loaded
6:56:59 PM: minmax store loaded
6:56:59 PM: starting kitchen sink init
6:56:59 PM: settings loaded
6:56:59 PM: pathfinding loaded
6:56:59 PM: databases scanned
6:56:59 PM: sea ice loaded
6:56:59 PM: loading complete
6:56:59 PM: postlock
6:56:59 PM: Simulation core engine loaded!
6:57:22 PM: STARTING Multiplayer Server
6:57:23 PM: Loading scenario D:\Command\Scenarios\Standalone Scenarios\First Contact, 1986.scen
6:57:23 PM: Scenario loaded.
6:57:24 PM: Autosaving InitialScenario.scen to disk.
6:57:24 PM: Merge Algorithm: Compartment Merge
6:57:24 PM: Server READY and listening at 46789
```

The screenshot above shows what the server would look like when it is ready to accept connections. The server is not ready until you get that server ready log entry.

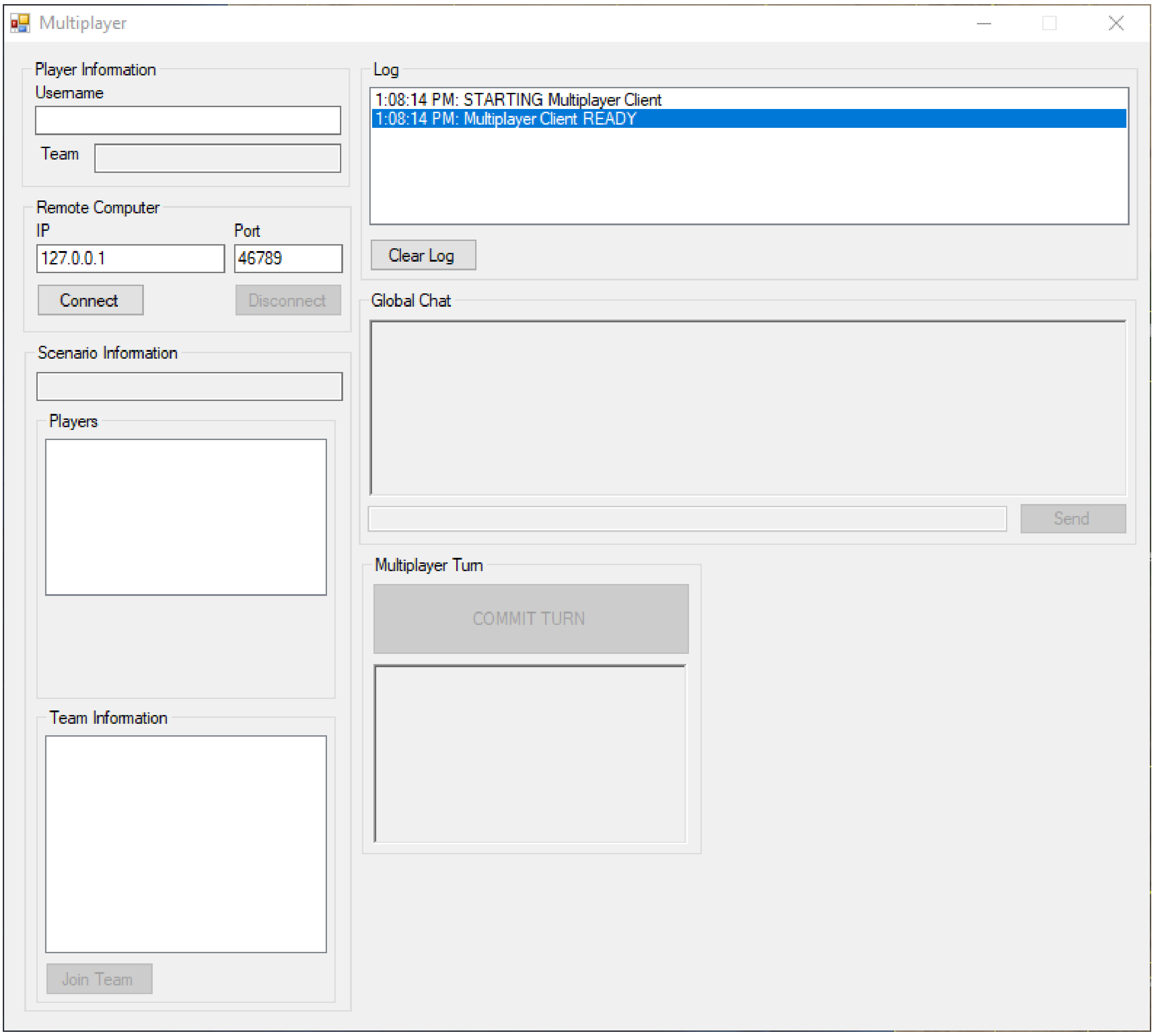
At this point clients are able to connect to the server.

Client Setup and Gameplay



Licensed to WarfareSims LTD
License expires 1/1/2023
Tech Support: command-pro@matrixgames.com

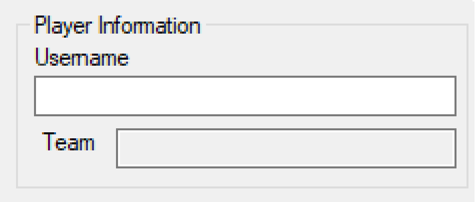
On the CPE start menu, select **"Join Multiplayer Game"**.



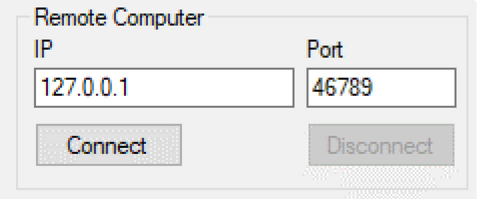
```
1:08:14 PM: STARTING Multiplayer Client
1:08:14 PM: Multiplayer Client READY
```

IP: 127.0.0.1
Port: 46789

This brings up the multiplayer window.



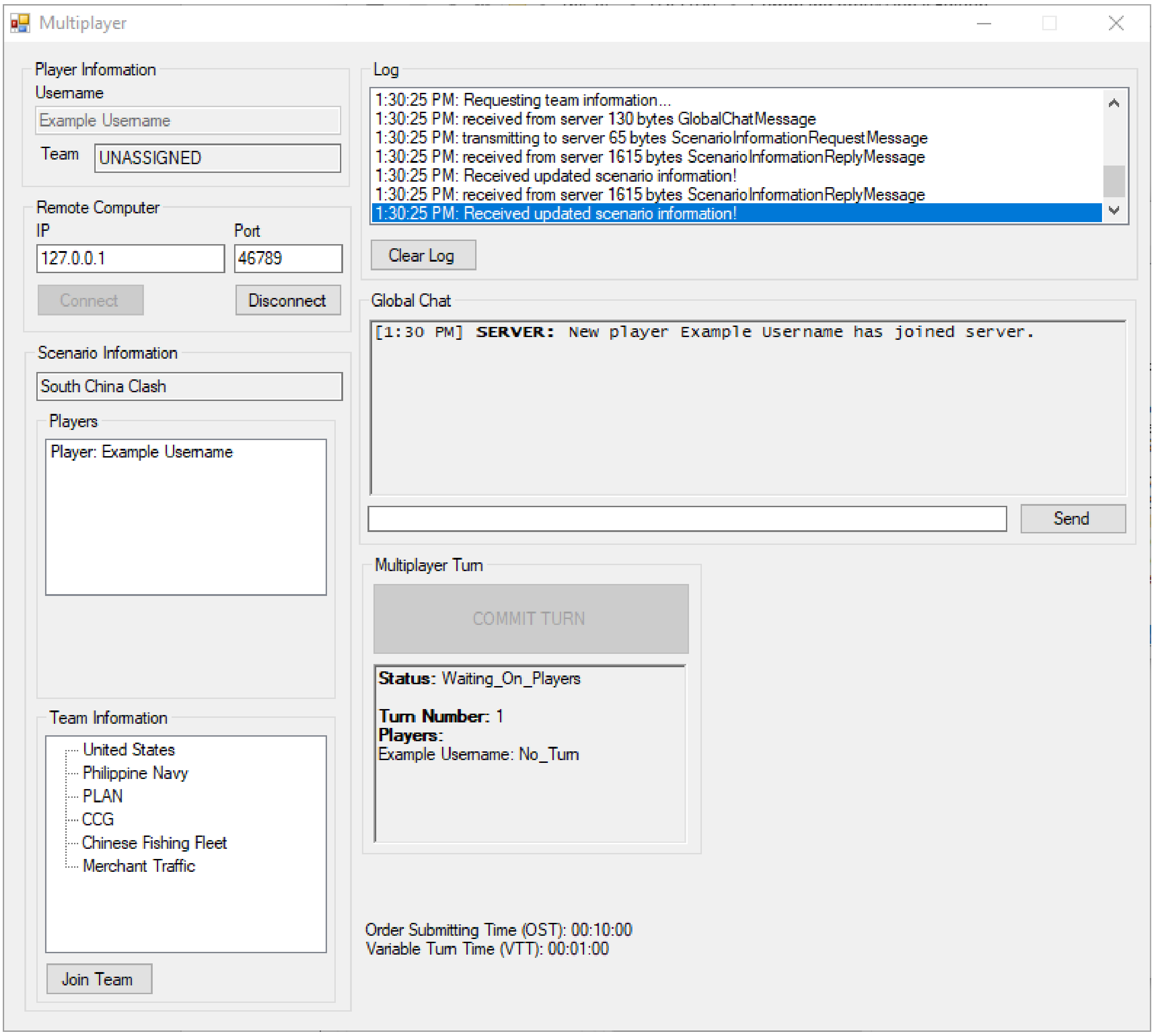
First off, input a username.



IP: 127.0.0.1
Port: 46789

Then input the IP address of the dedicated server along with any custom port selection.

Then click on **"Connect"**.



Username: Example Username

Team: UNASSIGNED

IP: 127.0.0.1

Port: 46789

South China Clash

Player: Example Username

- United States
- Philippine Navy
- PLAN
- CCG
- Chinese Fishing Fleet
- Merchant Traffic

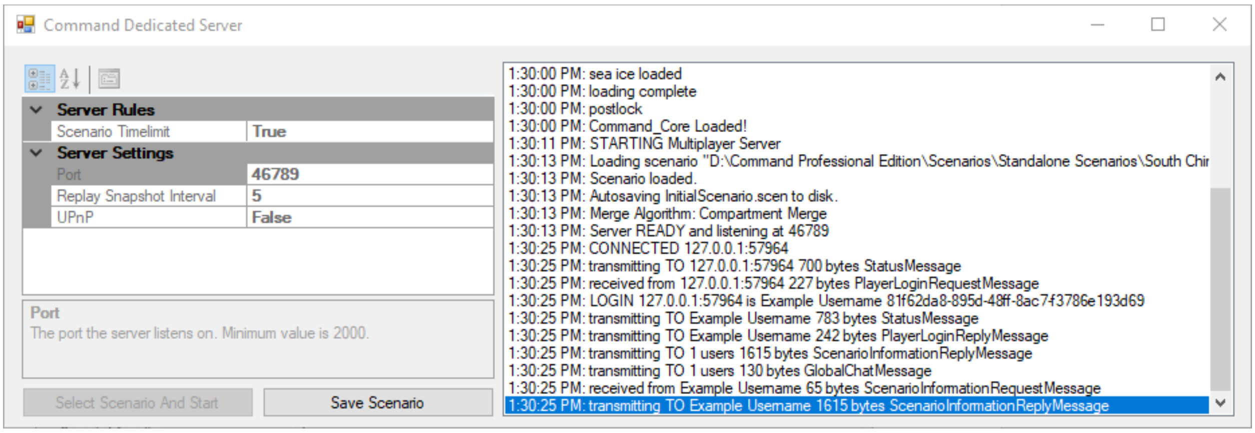
```
1:30:25 PM: Requesting team information...
1:30:25 PM: received from server 130 bytes GlobalChatMessage
1:30:25 PM: transmitting to server 65 bytes ScenarioInformationRequestMessage
1:30:25 PM: received from server 1615 bytes ScenarioInformationReplyMessage
1:30:25 PM: Received updated scenario information!
1:30:25 PM: received from server 1615 bytes ScenarioInformationReplyMessage
1:30:25 PM: Received updated scenario information!
```

```
[1:30 PM] SERVER: New player Example Username has joined server.
```

Status: Waiting_On_Players

Turn Number: 1
Players:
Example Username: No_Turn

Order Submitting Time (OST): 00:10:00
Variable Turn Time (VTT): 00:01:00



Server Rules

Scenario Timelimit	True

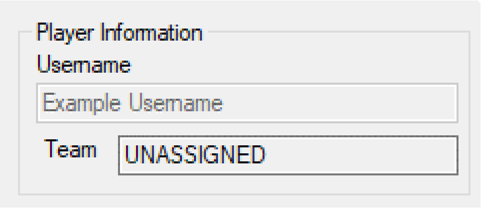
Server Settings

Port	46789
Replay Snapshot Interval	5
UPnP	False

Port
The port the server listens on. Minimum value is 2000.

```
1:30:00 PM: sea ice loaded
1:30:00 PM: loading complete
1:30:00 PM: postlock
1:30:00 PM: Command_Core Loaded!
1:30:11 PM: STARTING Multiplayer Server
1:30:13 PM: Loading scenario "D:\Command Professional Edition\Scenarios\Standalone Scenarios\South Chir
1:30:13 PM: Scenario loaded.
1:30:13 PM: Autosaving InitialScenario.scen to disk.
1:30:13 PM: Merge Algorithm: Compartment Merge
1:30:13 PM: Server READY and listening at 46789
1:30:25 PM: CONNECTED 127.0.0.1:57964
1:30:25 PM: transmitting TO 127.0.0.1:57964 700 bytes StatusMessage
1:30:25 PM: received from 127.0.0.1:57964 227 bytes PlayerLoginRequestMessage
1:30:25 PM: LOGIN 127.0.0.1:57964 is Example Username 81f62da8-895d-48ff-8ac7f3786e193d69
1:30:25 PM: transmitting TO Example Username 783 bytes StatusMessage
1:30:25 PM: transmitting TO Example Username 242 bytes PlayerLoginReplyMessage
1:30:25 PM: transmitting TO 1 users 1615 bytes ScenarioInformationReplyMessage
1:30:25 PM: transmitting TO 1 users 130 bytes GlobalChatMessage
1:30:25 PM: received from Example Username 65 bytes ScenarioInformationRequestMessage
1:30:25 PM: transmitting TO Example Username 1615 bytes ScenarioInformationReplyMessage
```

Once connected to the MP server, the client will automatically receive the scenario and database file (if it is not already present on the client system) from the server. As database files may be large, and thus the transfer may take some time, it is recommended to have the required database file pre-installed on the client system whenever possible.

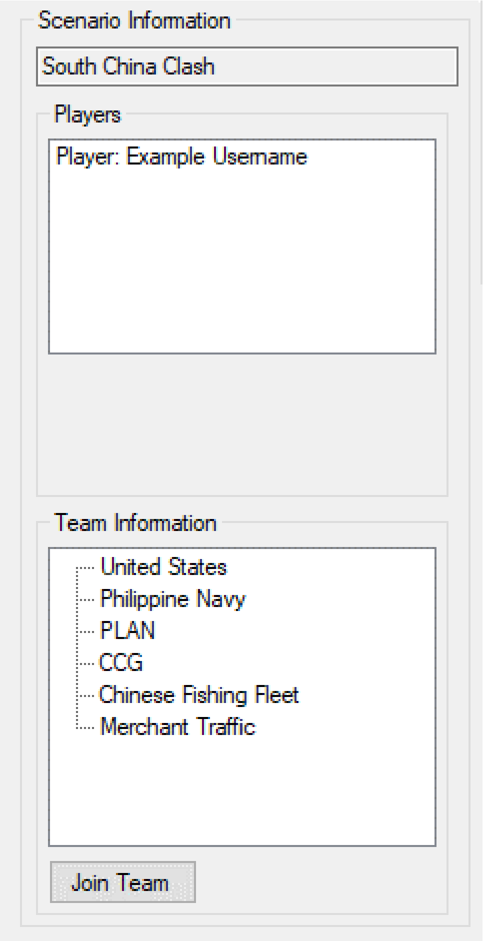


Username

Example Username

Team UNASSIGNED

The player information box now has locked in our username, and also indicates that we are not assigned to a team:

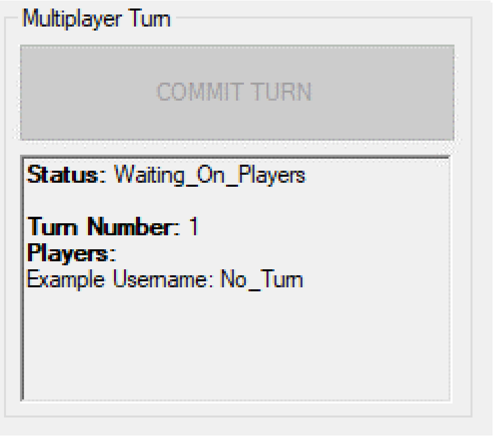


South China Clash

Player: Example Username

- United States
- Philippine Navy
- PLAN
- CCG
- Chinese Fishing Fleet
- Merchant Traffic

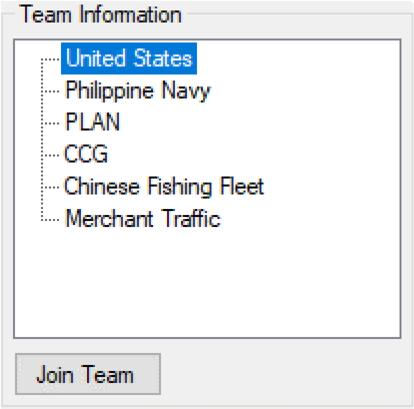
The scenario information box indicates the name of the scenario, in this case *South China Clash*, as well as the connected players. The team information box shows all the available sides in the scenario.



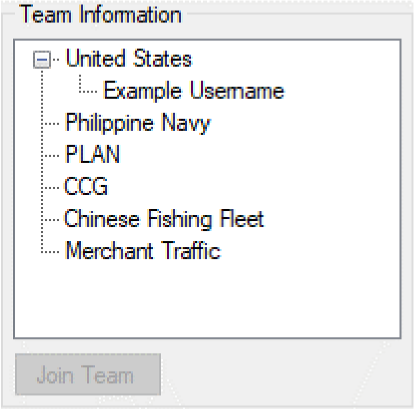
Status: Waiting_On_Players

Turn Number: 1
Players:
Example Username: No_Turn

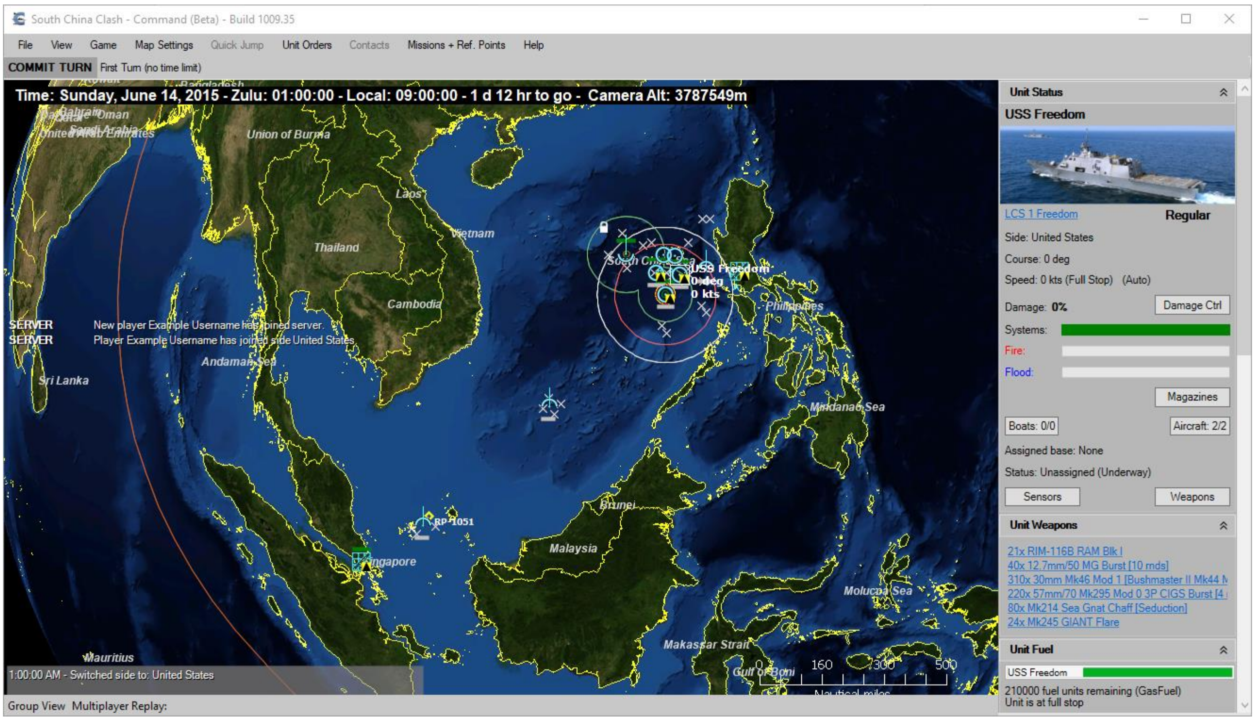
And the multiplayer turn box indicates that the server is currently waiting on the connected players, this is the first turn, and that there is one player connected.



Selecting the side *United States* in the team information box, we can then click on **"Join Team"**.



After joining the specified team, the interface updates and shows that we have control over the United States side. The server has sent us our picture of the battlefield on the main CPE map window:



First Turn (no time limit)

Time: Sunday, June 14, 2015 - Zulu: 01:00:00 - Local: 09:00:00 - 1 d 12 hr to go - Camera Alt: 3787549m

SERVER New player Example Username has joined server.
SERVER Player Example Username has joined side United States

1:00:00 AM - Switched side to: United States

USS Freedom

LCS 1 Freedom Regular

Side: United States

Course: 0 deg

Speed: 0 kts (Full Stop) (Auto)

Damage: 0%

Systems:

Fire:

Flood:

Boats: 0/0 Aircraft: 2/2

Assigned base: None

Status: Unassigned (Underway)

21x RIM-116B RAM Blk I
40x 12.7mm/50 MG Burst [10 rnds]
310x 30mm Mk46 Mod 1 [Bushmaster II Mk44 M
220x 57mm/70 Mk295 Mod 0 3P CIGS Burst [4
80x Mk214 Sea Gnat Chaff [Seduction]
24x Mk245 GIANT Flare

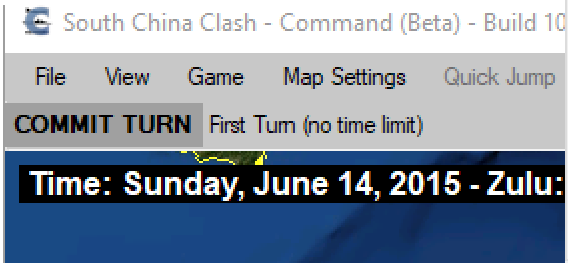
USS Freedom

210000 fuel units remaining (GasFuel)
Unit is at full stop

We can now issue orders to units on our side, just like in single-player mode. The game is paused while we issue orders.

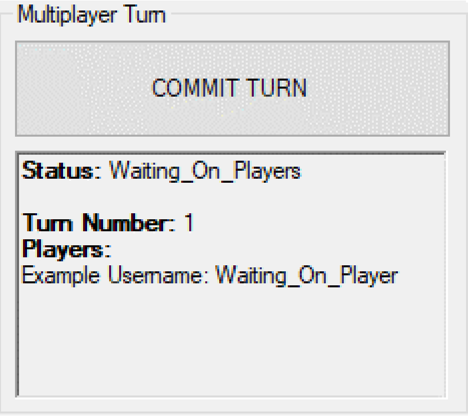
We can use any part of Command's UI to control our forces, including assigning missions and doctrines to units. Since we do not have control over the units while the scenario is executed on the server, it is important to "look ahead" and set up desired unit behaviors through missions and doctrines, and have a clear set of ROEs for our units to reliably control their behavior while they are out of direct control during turn execution.

When finished issuing orders, press **"COMMIT TURN"** on either the main map window, or in the multiplayer window (or press spacebar).


South China Clash - Command (Beta) - Build 10
File View Game Map Settings Quick Jump
COMMIT TURN

First Turn (no time limit)

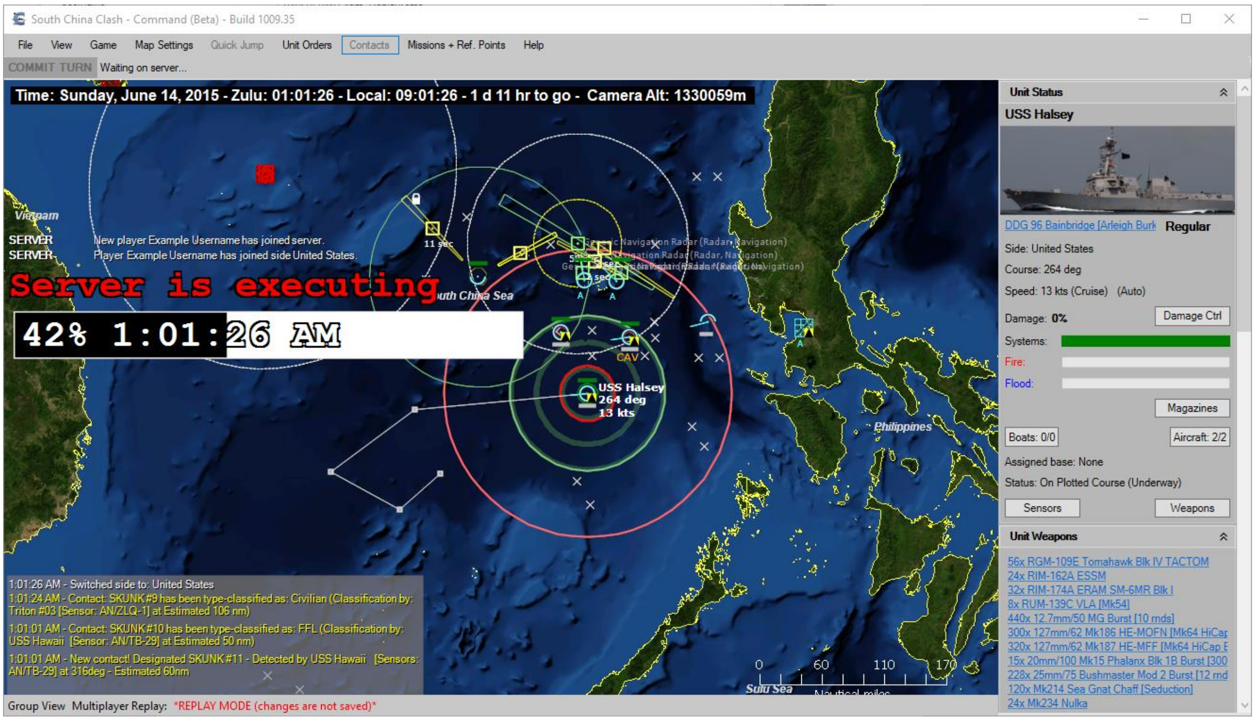
Time: Sunday, June 14, 2015 - Zulu:



Status: Waiting_On_Players

Turn Number: 1
Players:
Example Username: Waiting_On_Player

After all the players have committed their orders, then the server will execute and progress the scenario for the allotted Variable Turn Time (VTT) duration.



South China Clash - Command (Beta) - Build 1009.35

COMMIT TURN Waiting on server...

Time: Sunday, June 14, 2015 - Zulu: 01:01:26 - Local: 09:01:26 - 1 d 11 hr to go - Camera Alt: 1330059m

SERVER New player Example Username has joined server.
SERVER Player Example Username has joined side United States.

Server is executing

42% 1:01:26 AM

Unit Status

USS Halsey

DDG 96 Bainbridge [Arleigh Burk Regular

Side: United States

Course: 264 deg

Speed: 13 kts (Cruise) (Auto)

Damage: 0%

Boats: 0/0

Aircraft: 2/2

Assigned base: None

Status: On Plotted Course (Underway)

Unit Weapons

56x RGM-109E Tomahawk Blk IV TACTOM
24x RIM-162A ESSM
32x RIM-174A ERAM SM-6MR Blk I
8x RUM-139C VLA [Mk54]
440x 12.7mm/50 MG Burst [10 rnds]
300x 127mm/62 Mk186 HE-MOFN [Mk64 HiCap
320x 127mm/62 Mk187 HE-MFF [Mk64 HiCap E
15x 20mm/100 Mk15 Phalanx Blk 1B Burst [300
228x 25mm/75 Bushmaster Mod 2 Burst [12 rnd
120x Mk214 Sea Gnat Chaff [Seduction]
24x Mk234 Nulka

1:01:26 AM - Switched side to: United States
1:01:24 AM - Contact: SKUNK #9 has been type-classified as: Civilian (Classification by: Triton #03 [Sensor: AN/ZLQ-1] at Estimated 106 nm)
1:01:01 AM - Contact: SKUNK #10 has been type-classified as: FFL (Classification by: USS Hawaii [Sensor: AN/TB-29] at Estimated 50 nm)
1:01:01 AM - New contact! Designated SKUNK #11 - Detected by USS Hawaii [Sensors: AN/TB-29] at 316deg - Estimated 60nm

Group View Multiplayer Replay: *REPLAY MODE (changes are not saved)*

While the server is executing, if the "Replay Snapshot Interval" setting has been set to a non-zero value, players are provided with visual snapshots of what is occurring during turn execution. In this example screenshot, the server has processed 42% of the turn, and the current scenario time is 01:01:26 Zulu (09:01:26 local).

"MP-Only" mode: Connecting non-Premium instances to MP on a secure network

One of the unique features of the Premium license tier is the ability to run in an isolated (ie. non-Internet connected) machine. All other license tiers require a present Internet connection in order to function, and will refuse to start without it.

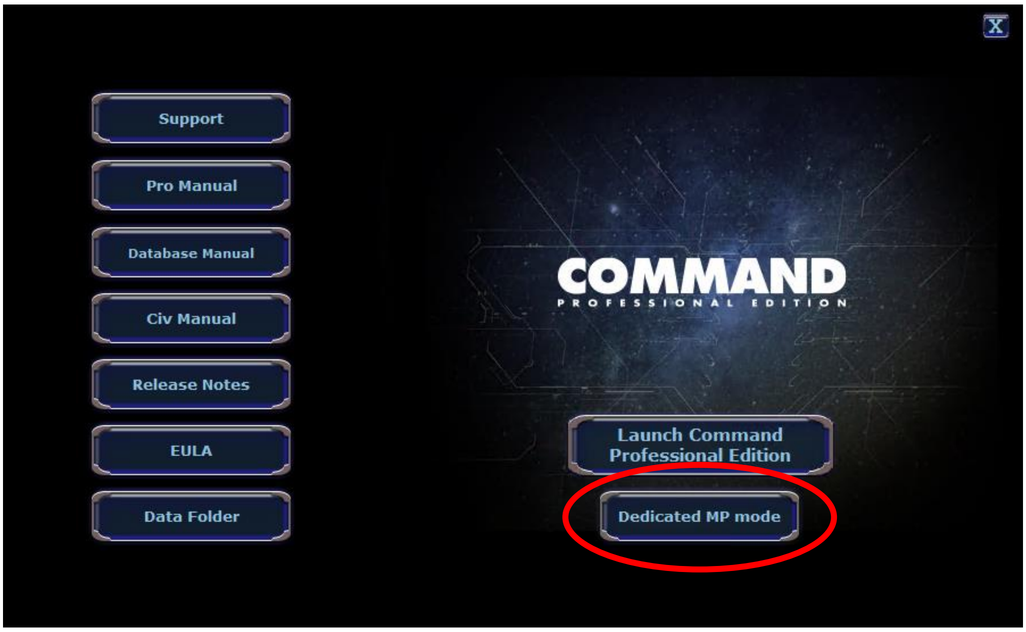
So, what happens if we have a set of non-Premium CPE instances and want to make them clients in a WEG-MP session in a secure (no Internet) network?

Beginning in CPE v2.1.4, we can use a special mode for this predicament called "MP-Only". Using this mode has two practical consequences:

1) The CPE instance that runs in this mode can start up even without an active Internet connection, even if it has a non-Premium license. It can then join a new or existing WEGO-MP session as usual.
2) At the same time, running in this mode allows the CPE instance to only function as an MP client. It cannot function as a normal single-user workstation for analysis, scenario editing etc. In order to function as a normal standalone instance, CPE needs to be shut down and restarted in normal mode.

There are two ways to start a CPE instance in MP-Only mode:

1) In the CPE v2.1.4+ application launcher window, click on the button "Dedicated MP mode":



2) You can also trigger MP-Only mode from the command line, by launching CPE with the –MPOnly switch. For example:

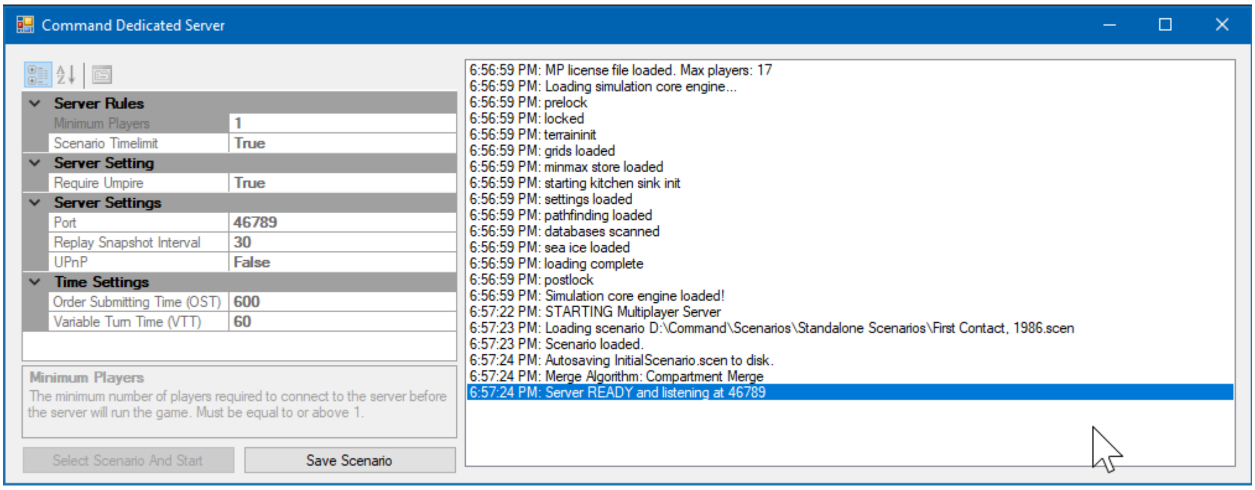
```
[CPE2 binaries folder]\Command.exe -MPOnly
```

The Umpire role

CMP’s umpire capability is an optional game mode which involves the addition of a special player: the Umpire.

The Umpire has the ability to edit the scenario before and after server-side turn execution occurs. The umpire can add units, remove units, change the damage on units, and perform any task associated with scenario editing in Command. The Umpire can also change the VTT and OST values at any time during the scenario playout, to speed up or slow down (or focus) the game tempo as desired.

Furthermore, the Umpire can force all of the connected players to submit their turns at the press a button.



Server Rules

Minimum Players	1
Scenario Timelimit	True

Server Setting

Require Umpire	True

Server Settings

Port	46789
Replay Snapshot Interval	30
UPnP	False

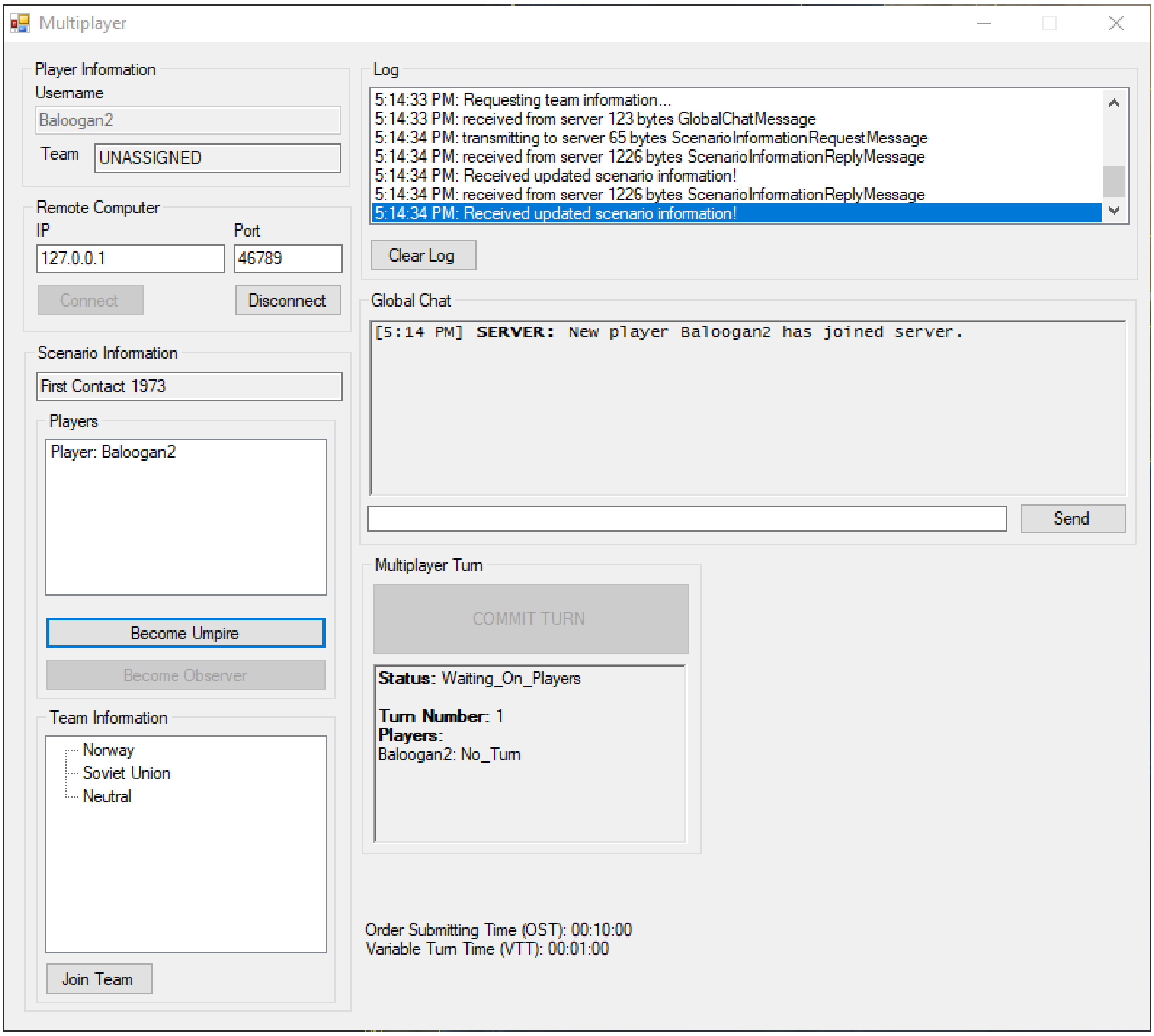
Time Settings

Order Submitting Time (OST)	600
Variable Turn Time (VTT)	60

Minimum Players
The minimum number of players required to connect to the server before the server will run the game. Must be equal to or above 1.

```
6:56:59 PM: MP license file loaded. Max players: 17
6:56:59 PM: Loading simulation core engine...
6:56:59 PM: prelock
6:56:59 PM: locked
6:56:59 PM: terraininit
6:56:59 PM: grids loaded
6:56:59 PM: minmax store loaded
6:56:59 PM: starting kitchen sink init
6:56:59 PM: settings loaded
6:56:59 PM: pathfinding loaded
6:56:59 PM: databases scanned
6:56:59 PM: sea ice loaded
6:56:59 PM: loading complete
6:56:59 PM: postlock
6:56:59 PM: Simulation core engine loaded!
6:57:22 PM: STARTING Multiplayer Server
6:57:23 PM: Loading scenario D:\Command\Scenarios\Standalone Scenarios\First Contact, 1986.scen
6:57:23 PM: Scenario loaded.
6:57:24 PM: Autosaving InitialScenario.scen to disk.
6:57:24 PM: Merge Algorithm: Compartment Merge
6:57:24 PM: Server READY and listening at 46789
```

To enable the Umpire role, the **"Require Umpire"** option must be enabled prior to loading the scenario. When this server setting is set to "True", the server will not allow turn execution to progress without a connected umpire.



Player Information
Username: Baloogan2
Team: UNASSIGNED

Remote Computer
IP: 127.0.0.1
Port: 46789

Scenario Information
First Contact 1973

Players
Player: Baloogan2

Team Information
- Norway
- Soviet Union
- Neutral

Log

```
5:14:33 PM: Requesting team information...
5:14:33 PM: received from server 123 bytes GlobalChatMessage
5:14:34 PM: transmitting to server 65 bytes ScenarioInformationRequestMessage
5:14:34 PM: received from server 1226 bytes ScenarioInformationReplyMessage
5:14:34 PM: Received updated scenario information!
5:14:34 PM: received from server 1226 bytes ScenarioInformationReplyMessage
5:14:34 PM: Received updated scenario information!
```

Global Chat

```
[5:14 PM] SERVER: New player Baloogan2 has joined server.
```

Multiplayer Turn

Status: Waiting_On_Players

Turn Number: 1
Players:
Baloogan2: No_Turn

Order Submitting Time (OST): 00:10:00
Variable Turn Time (VTT): 00:01:00

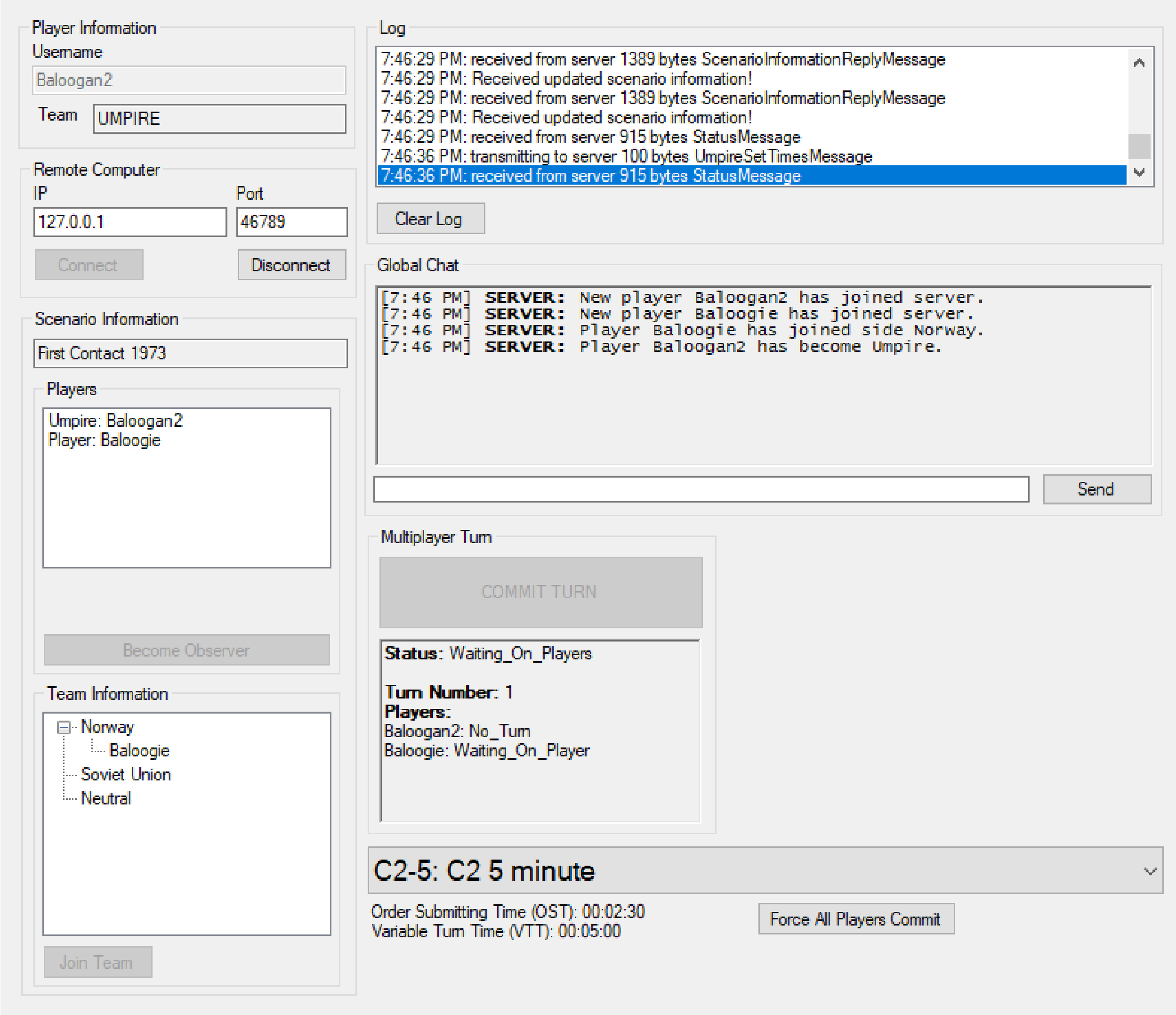
Upon connecting to a server, if the Umpire role is enabled, an additional button called **"Become Umpire"** appears on the scenario information panel.

Upon selecting **“Become Umpire”,** the Umpire player does not have the ability to edit the scenario immediately (it is assumed that initial scenario configuration will occur before the MP game starts). Once all of the players are connected and have issued their initial orders, then the umpire gets the chance to edit the scenario; this is the pre-turn adjudication phase. This happens before any time passes on the server, and the Umpire is able to change the orders issued by the players if he so desires to. After the Umpire commits his turn, the scenario is then executed for the Variable Turn Time selected.

After server execution has ended and the entire VTT execution time has elapsed, the Umpire is once again able to edit the scenario; this is the post-turn adjudication phase. This gives the umpire a second chance to make any alterations to the scenario, this time after the turn has executed. Once the Umpire ends this phase, all the players get fresh copies of the scenario state and are now able to issue new orders.

It is important to note that while the Umpire is adjudicating, all the players are waiting on server execution to complete. Likewise, until all players commits their orders, the Umpire is unable to edit the scenario state.

Let us walk through a representative Umpire-adjudicated turn sequence.



Username: Baloogan2
Team: UMPIRE

IP: 127.0.0.1
Port: 46789

First Contact 1973

Umpire: Baloogan2
Player: Baloogie

Norway
- Baloogie

Soviet Union
Neutral

```
7:46:29 PM: received from server 1389 bytes ScenarioInformationReplyMessage
7:46:29 PM: Received updated scenario information!
7:46:29 PM: received from server 1389 bytes ScenarioInformationReplyMessage
7:46:29 PM: Received updated scenario information!
7:46:29 PM: received from server 915 bytes StatusMessage
7:46:36 PM: transmitting to server 100 bytes UmpireSetTimesMessage
7:46:36 PM: received from server 915 bytes StatusMessage
```

```
[7:46 PM] SERVER: New player Baloogan2 has joined server.
[7:46 PM] SERVER: New player Baloogie has joined server.
[7:46 PM] SERVER: Player Baloogie has joined side Norway.
[7:46 PM] SERVER: Player Baloogan2 has become Umpire.
```

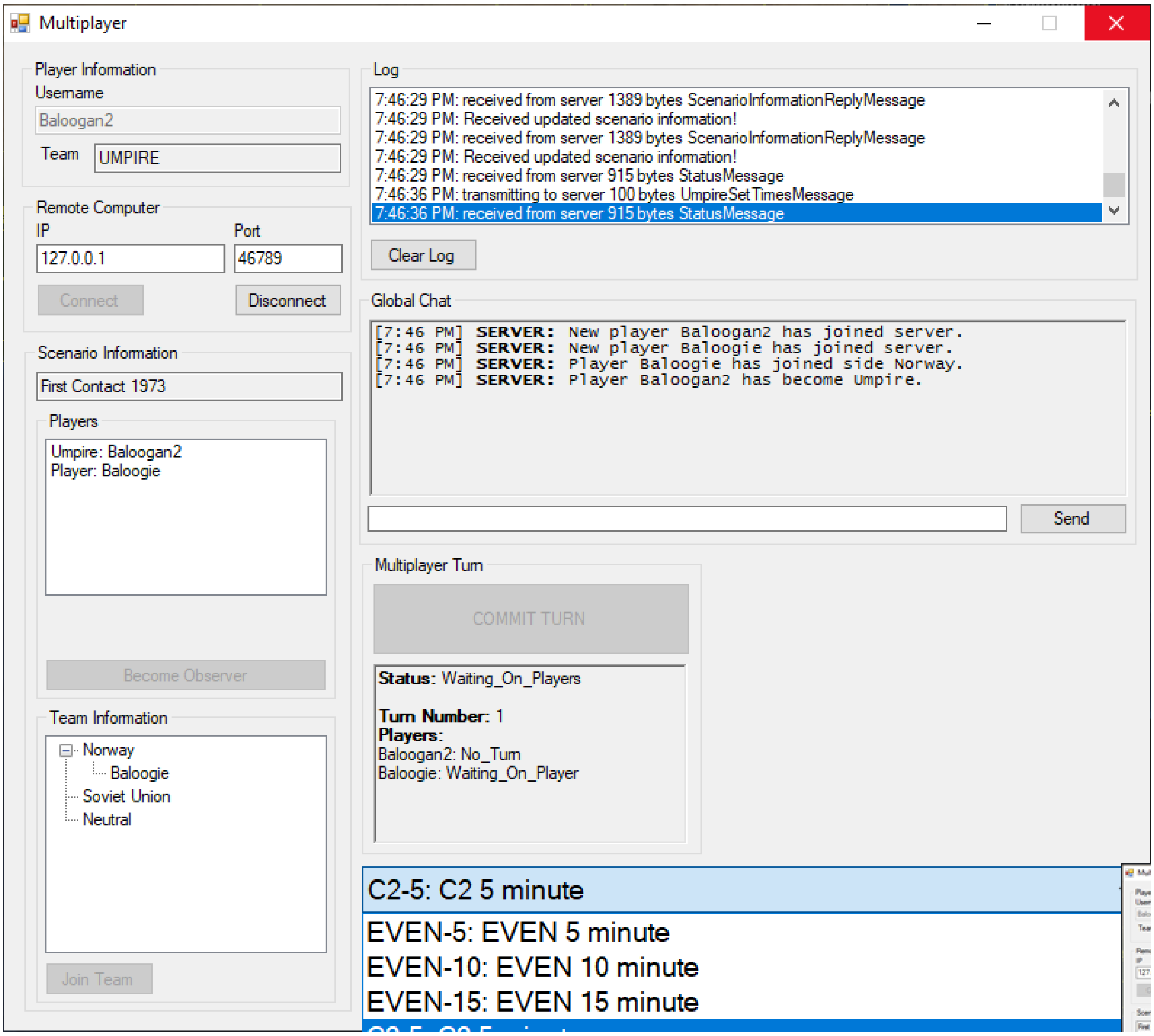
Status: Waiting_On_Players

Turn Number: 1
Players:
Baloogan2: No_Turn
Baloogie: Waiting_On_Player

C2-5: C2 5 minute

Order Submitting Time (OST): 00:02:30
Variable Turn Time (VTT): 00:05:00

This is what the Umpire sees in the MP window; the distinct Umpire-specific elements are visible on the bottom right-hand side.



Username: Baloogan2

Team: UMPIRE

Remote Computer

IP: 127.0.0.1

Port: 46789

Scenario Information: First Contact 1973

Players:
Umpire: Baloogan2
Player: Baloogie

Team Information:
- Norway
  - Baloogie
- Soviet Union
- Neutral

```
7:46:29 PM: received from server 1389 bytes ScenarioInformationReplyMessage
7:46:29 PM: Received updated scenario information!
7:46:29 PM: received from server 1389 bytes ScenarioInformationReplyMessage
7:46:29 PM: Received updated scenario information!
7:46:29 PM: received from server 915 bytes StatusMessage
7:46:36 PM: transmitting to server 100 bytes UmpireSetTimesMessage
7:46:36 PM: received from server 915 bytes StatusMessage
```

```
[7:46 PM] SERVER: New player Baloogan2 has joined server.
[7:46 PM] SERVER: New player Baloogie has joined server.
[7:46 PM] SERVER: Player Baloogie has joined side Norway.
[7:46 PM] SERVER: Player Baloogan2 has become Umpire.
```

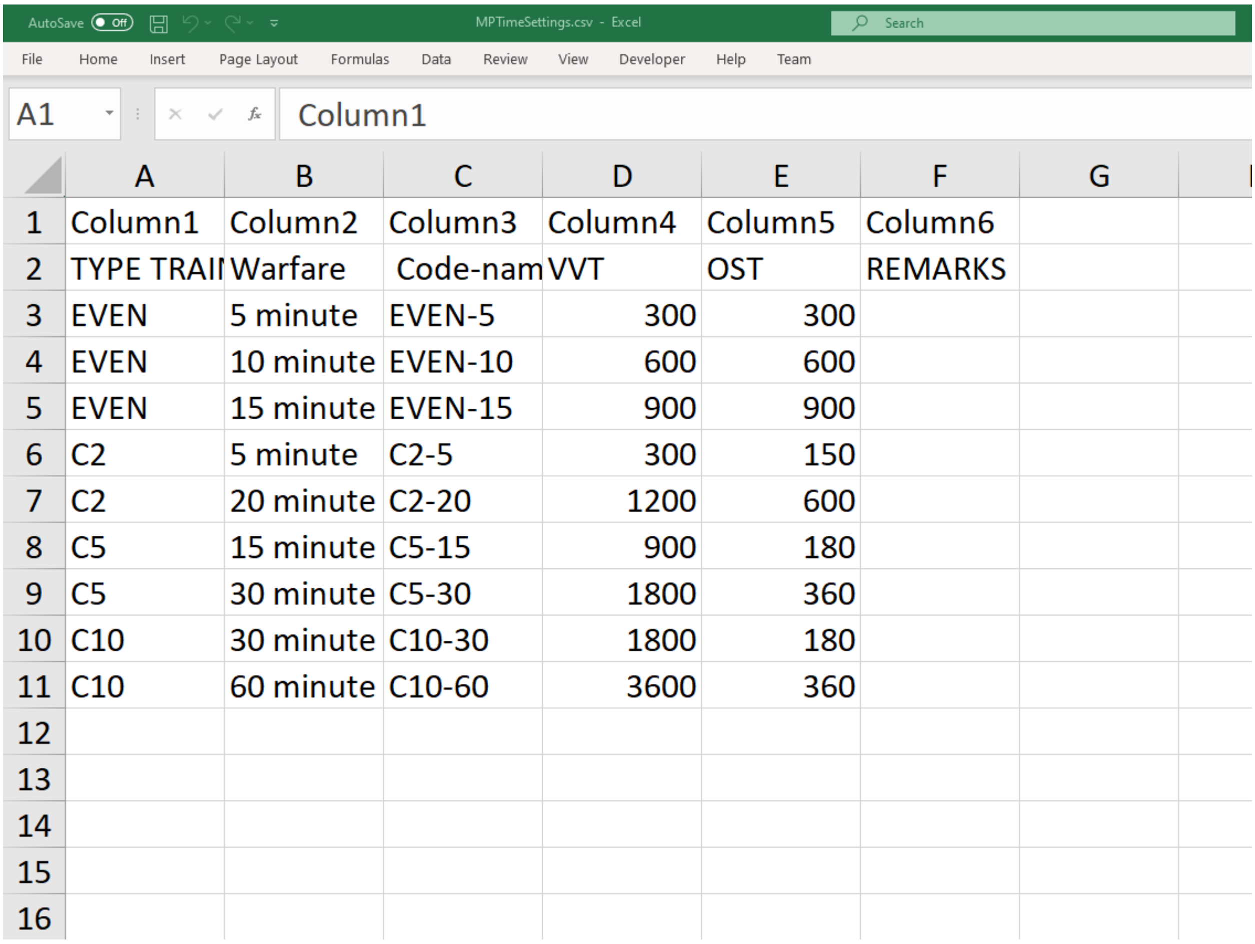
Status: Waiting_On_Players

Turn Number: 1
Players:
Baloogan2: No_Turn
Baloogie: Waiting_On_Player

C2-5: C2 5 minute
EVEN-5: EVEN 5 minute
EVEN-10: EVEN 10 minute
EVEN-15: EVEN 15 minute

As mentioned, the Umpire can alter the OST and VTT duration values. This is performed through this drop-down menu.

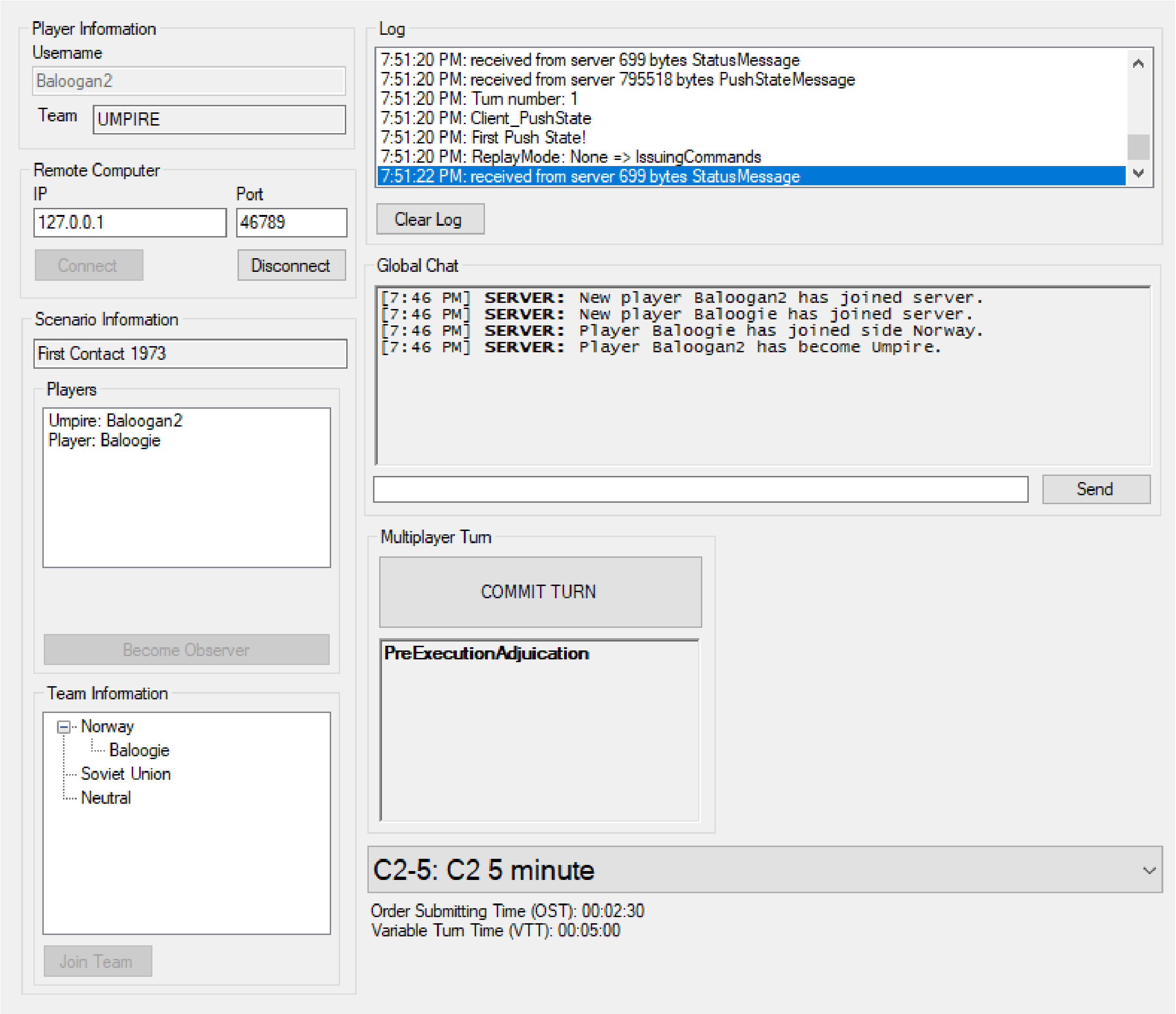
Several pre-set OST and VTT combinations are offered, covering different situations. These, however, can also be manually changed, by editing the "[top-level writable folder]\Config\MPTimeSettings.csv" file:



	A	B	C	D	E	F	G
1	Column1	Column2	Column3	Column4	Column5	Column6	
2	TYPE TRAII	Warfare	Code-nam	VVT	OST	REMARKS	
3	EVEN	5 minute	EVEN-5	300	300		
4	EVEN	10 minute	EVEN-10	600	600		
5	EVEN	15 minute	EVEN-15	900	900		
6	C2	5 minute	C2-5	300	150		
7	C2	20 minute	C2-20	1200	600		
8	C5	15 minute	C5-15	900	180		
9	C5	30 minute	C5-30	1800	360		
10	C10	30 minute	C10-30	1800	180		
11	C10	60 minute	C10-60	3600	360		
12							
13							
14							
15							
16							

The D and E column correspond to the Variable Turn Time (VTT) in seconds, and the Order Submitting Time (OST) in seconds respectively.

The computer that acts as the server does not need to be the same computer that connects to the server as Umpire; the CSV file with the OST/VTT settings is always read from the Umpire’s computer.



Username: Baloogan2
Team: UMPIRE
IP: 127.0.0.1
Port: 46789

First Contact 1973

Umpire: Baloogan2
Player: Baloogie

Norway
Baloogie
Soviet Union
Neutral

```
7:51:20 PM: received from server 699 bytes StatusMessage
7:51:20 PM: received from server 795518 bytes PushStateMessage
7:51:20 PM: Turn number: 1
7:51:20 PM: Client_PushState
7:51:20 PM: First Push State!
7:51:20 PM: ReplayMode: None => IssuingCommands
7:51:22 PM: received from server 699 bytes StatusMessage
```

```
[7:46 PM] SERVER: New player Baloogan2 has joined server.
[7:46 PM] SERVER: New player Baloogie has joined server.
[7:46 PM] SERVER: Player Baloogie has joined side Norway.
[7:46 PM] SERVER: Player Baloogan2 has become Umpire.
```

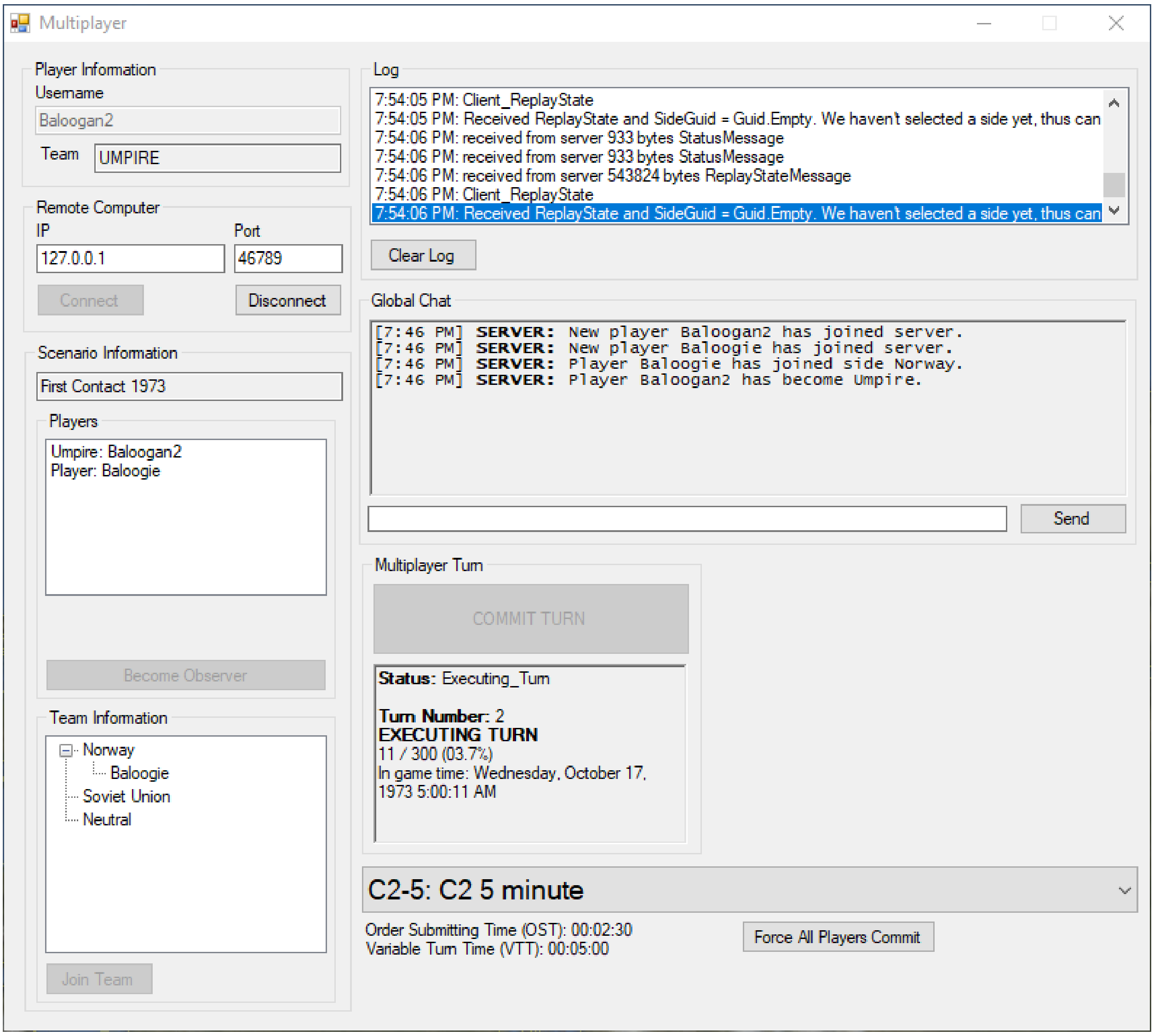
PreExecutionAdjuication

C2-5: C2 5 minute

Order Submitting Time (OST): 00:02:30
Variable Turn Time (VTT): 00:05:00

After all the players have submitted their turns, the Umpire is presented with the current scenario state with all of the orders issued by the players brought in. The multiplayer panel displays PreExecutionAdjudication. This indicates that the umpire is modifying the scenario state. When satisfied, the Umpire must click **"COMMIT TURN"** (or press spacebar) to send the scenario state from the Umpire's computer to the server. After the Umpire has committed his changes, the turn will execute on the server.

While the Umpire is editing the game scenario he can switch from side to side using the "Editor" menu. **IMPORTANT: Do not add or remove sides in the middle of the game as the Umpire.**



Username: Baloogan2
Team: UMPIRE

IP: 127.0.0.1
Port: 46789

First Contact 1973

Umpire: Baloogan2
Player: Baloogie

Norway
Baloogie
Soviet Union
Neutral

```
7:54:05 PM: Client_ReplayState
7:54:05 PM: Received ReplayState and SideGuid = Guid.Empty. We haven't selected a side yet, thus can
7:54:06 PM: received from server 933 bytes StatusMessage
7:54:06 PM: received from server 933 bytes StatusMessage
7:54:06 PM: received from server 543824 bytes ReplayStateMessage
7:54:06 PM: Client_ReplayState
7:54:06 PM: Received ReplayState and SideGuid = Guid.Empty. We haven't selected a side yet, thus can
```

```
[7:46 PM] SERVER: New player Baloogan2 has joined server.
[7:46 PM] SERVER: New player Baloogie has joined server.
[7:46 PM] SERVER: Player Baloogie has joined side Norway.
[7:46 PM] SERVER: Player Baloogan2 has become Umpire.
```

Status: Executing_Turn

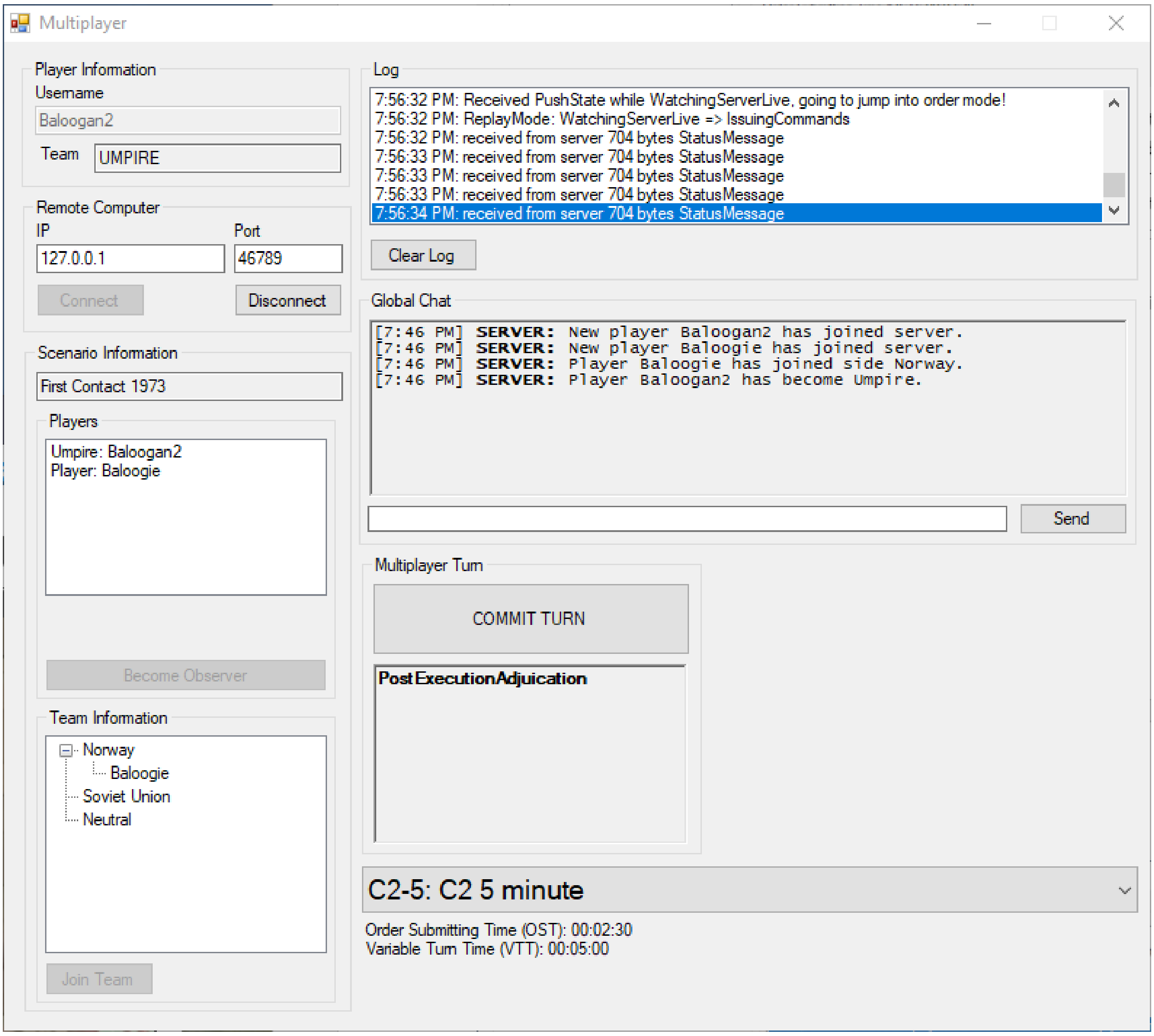
Turn Number: 2
EXECUTING TURN
11 / 300 (03.7%)
In game time: Wednesday, October 17, 1973 5:00:11 AM

C2-5: C2 5 minute

Order Submitting Time (OST): 00:02:30
Variable Turn Time (VTT): 00:05:00

After the Umpire commits his changes (if any), the server executes the length of time defined by the Variable Turn Time. This means the server will move the scenario state forward of time equal to the Variable Turn Time value, and will use the CPE simulation engine to adjudicate any game events.

After the server has completed the turn, the post-turn scenario state is presented to the Umpire first (the normal players are still waiting at this step); the Umpire can make any changes to the result of the adjudication as he wishes.



Username: Baloogan2
Team: UMPIRE
IP: 127.0.0.1
Port: 46789
Scenario Information: First Contact 1973

Players:
Umpire: Baloogan2
Player: Baloogie

Team Information:
- Norway
  - Baloogie
- Soviet Union
- Neutral

```
7:56:32 PM: Received PushState while WatchingServerLive, going to jump into order mode!
7:56:32 PM: ReplayMode: WatchingServerLive => IssuingCommands
7:56:32 PM: received from server 704 bytes StatusMessage
7:56:33 PM: received from server 704 bytes StatusMessage
7:56:33 PM: received from server 704 bytes StatusMessage
7:56:33 PM: received from server 704 bytes StatusMessage
7:56:34 PM: received from server 704 bytes StatusMessage
```

```
[7:46 PM] SERVER: New player Baloogan2 has joined server.
[7:46 PM] SERVER: New player Baloogie has joined server.
[7:46 PM] SERVER: Player Baloogie has joined side Norway.
[7:46 PM] SERVER: Player Baloogan2 has become Umpire.
```

PostExecutionAdjuication

C2-5: C2 5 minute

Order Submitting Time (OST): 00:02:30
Variable Turn Time (VTT): 00:05:00

After the server has completed the turn execution, the Umpire is again presented with this multiplayer window. Please note the status: PostExecutionAdjuication in the multiplayer turn status panel. Upon pressing “**COMMIT TURN**”, the players are then presented with the result of what the simulation engine has progressed plus the additions/changes that the Umpire makes during this post-execution adjudication phase. The Umpire can also modify the current Order Submitting Time (OST) and Variable Turn Time (VTT) at this point.

The play sequence cycle continues in this fashion until the scenario is concluded either by its own built-in criteria or any arbitrary decision by the Umpire.

The Observer role

The Observer role is similar to the Umpire role with regards to its omniscience, but critically lacks its omnipotence. Here are the main differences between these roles:

- There is no setting for specifying whether the Observer role will be required/available or not; players can always opt to become Observers at the beginning of a session.
- Observers, like Umpires, can monitor the flow of scenario execution and switch between sides to observe anything going on in the virtual battlefield. However, they CANNOT make any alterations to the scenario state. They also cannot change the OST/VTT settings.

REAL-TIME MULTIPLAYER (RTMP)

Introduction

Command RTMP is the realtime multiplayer capability of Command PE. As the name implies, this allows for multiple players to play together in the same scenario, the same simulation, in real time.

The purpose of this manual is to guide users through setting up and running a Command RTMP session, addressing common challenges, and maximizing the potential of its features. This document provides comprehensive details on installation, configuration, and gameplay mechanics to ensure a smooth experience in conducting multiplayer wargames.

Key Features and Benefits of RTMP

◦ Support for multiple players on the same or opposing sides.

◦ Umpires and observers for enhanced game control and oversight.

◦ Seamless "join in progress" functionality.

◦ Ability to accelerate time during gameplay.

◦ Revert the game to any previous state.

◦ Change scenarios mid-game without disrupting player connections.

◦ Compatibility with any scenario from Command PE, without the need for modifications.

◦ Event export for deeper analysis and review.

With Command RTMP, Command PE’s proven simulation engine is elevated to a multiplayer environment, creating new opportunities for collaborative and competitive play in real-time strategic scenarios.

Concepts

Command RTMP splits Command PE into two parts:

- **CommandHost.exe**, which runs the simulation, and permits clients to connect to the simulation.
- **Command.exe**, which connects to CommandHost.exe and allows users to play the scenario running on CommandHost.exe.

Command RTMP is still normal Command PE under the hood. The same Command PE simulation is running, in essence the simulation core is ‘unaware’ that it is being played multiplayer.

Definitions

Side: When this document talks about a 'side' it means a side in a Command scenario. E.g. you might have a scenario with two sides, USA and PRC.

Player: This means a computer running Command PE connected to a RTMP server, which usually will have a human sitting in front of it.

Server: This means the computer running CommandHost.exe.

Client: This means the computer running Command.exe, which is connected to a computer running CommandHost.exe.

Umpire: This is a 'superuser' player. Can modify the running scenario, and switch sides, view the viewports of other connected players.

Observer: This is another type of 'superuser' player, but cannot modify the running scenario. All attempts to modify the scenario state by an observer are ignored by the server.

Server and Client Roles

With one exception (Absolute Control mode, more on that later), the server will always have the 'definitive' version of the scenario. All simulation occurs on the server, the clients connected to the server are akin to 'dumb terminals' where they display and input information but run zero simulation.

Client Roles

There are 3 roles of connected client in Command RTMP: Player, Umpire, and Observer.

The best way to understand Command RTMP's client roles is to start with Umpires. Umpires are a connected client that have every ability including the ability to change sides, see all viewports, make edits to the game state etc.

An observer is an umpire that is unable to change the game state. They can change sides, see all viewports, but are unable to make any change to the game state.

A player is a connected client that is unable to change sides, and can only see the viewports of forces on his side.

Network Architecture

The best networking setup would be an unmanaged switch connecting all of the computers, clients and servers together on one network. However, we've had luck working over wifi many times.

Only the server's IP should remain static, the clients' IPs aren't important as long as they don't overlap.

System Requirements

Normal Command PE system requirements are required for Command RTMP, however obviously computers do need a network capability.

Hardware Requirements

A simple setup for Command RTMP is as follows:

- 1x Server Computer
- 4x Client Computers
- 1x 8-port Unmanaged Switch
- 5x CAT-5e cables of suitable length.

Server Recommendations

The minimum requirements for the server are the system requirements for CPE which can be found on the Matrix Pro Sims portal website:

https://www.matrixprosims.com/game/command-professional-edition

For the server, a faster gaming CPU is ideal, along with high quality (fast) RAM. Potentially using Windows Server might improve networking performance, but we haven't tested this internally.

For optimal server performance, do not use the server computer for anything other than a server. For optimal server performance, do not have the server computer perform double duty as server and umpire computer.

Client Recommendations

As zero simulation happens on the client computers, normal Command PE system requirements will work fine.

The system requirements for CPE can be found on the Matrix Pro Sims portal website:

https://www.matrixprosims.com/game/command-professional-edition

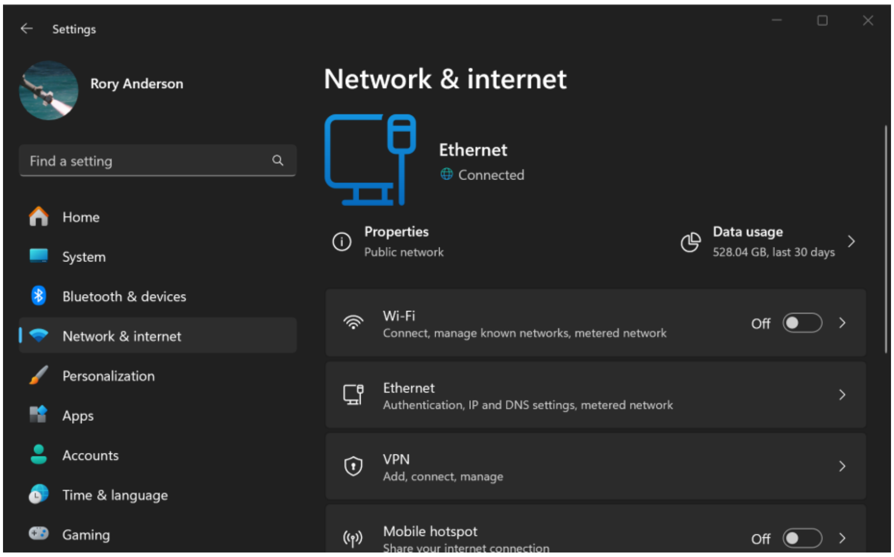
Network Architecture

Client-Server Model

The network architecture used by RTMP is a client-server model. The clients don't need to communicate among themselves, but every computer involved with an RTMP wargame must be able to communicate with the server.

How to find the Server's IP (Using Windows UI)

To find the IP address of a Windows computer using the Windows UI is as follows.



Network & internet

Ethernet
Connected

Properties
Public network

Data usage
528.04 GB, last 30 days

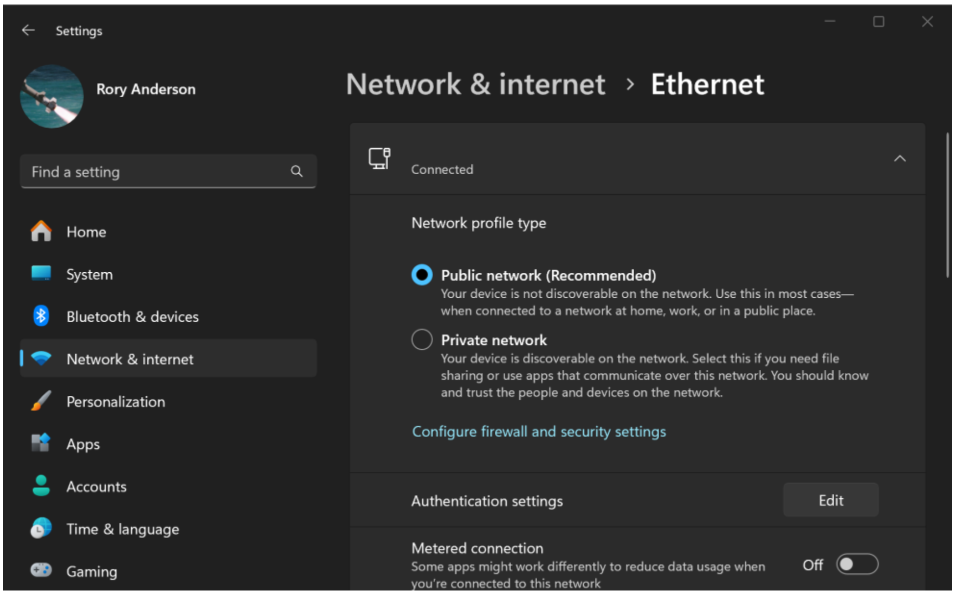
Wi-Fi
Connect, manage known networks, metered network
Off

Ethernet
Authentication, IP and DNS settings, metered network

VPN
Add, connect, manage

Mobile hotspot
Share your internet connection
Off

Open Network and Internet settings



Network & internet › Ethernet

Connected

Network profile type

● Public network (Recommended)
Your device is not discoverable on the network. Use this in most cases—when connected to a network at home, work, or in a public place.

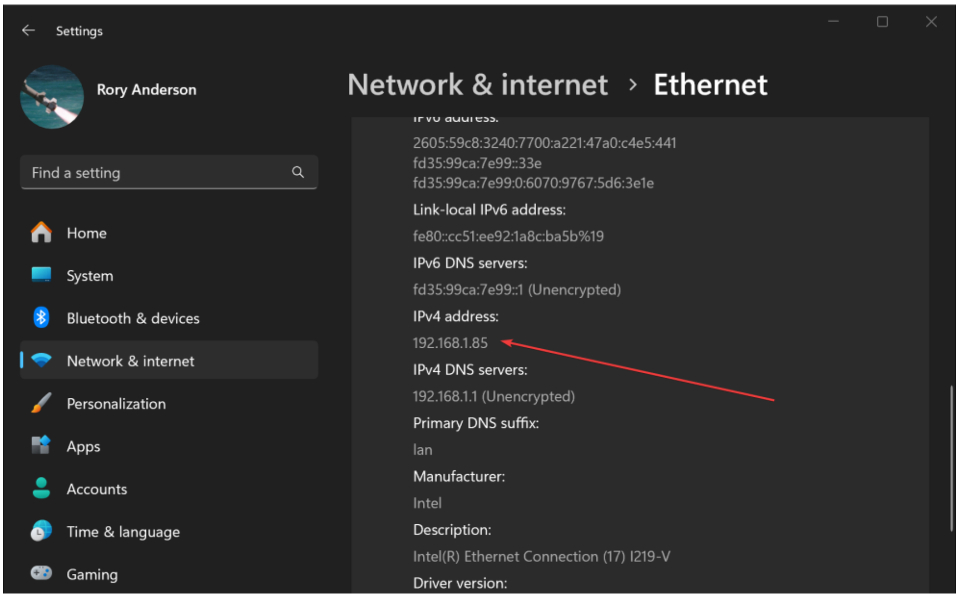
○ Private network
Your device is discoverable on the network. Select this if you need file sharing or use apps that communicate over this network. You should know and trust the people and devices on the network.

Configure firewall and security settings

Authentication settings
Edit

Metered connection
Some apps might work differently to reduce data usage when you're connected to this network
Off

Click on your ethernet adapter. This will be different if you are using WiFi vs a wired network.



IPv6 address:
2605:59c8:3240:7700:a221:47a0:c4e5:441
fd35:99ca:7e99::33e
fd35:99ca:7e99:0:6070:9767:5d6:3e1e

Link-local IPv6 address:
fe80::cc51:ee92:1a8c:ba5b%19

IPv6 DNS servers:
fd35:99ca:7e99::1 (Unencrypted)

IPv4 address:
192.168.1.85

IPv4 DNS servers:
192.168.1.1 (Unencrypted)

Primary DNS suffix:
lan

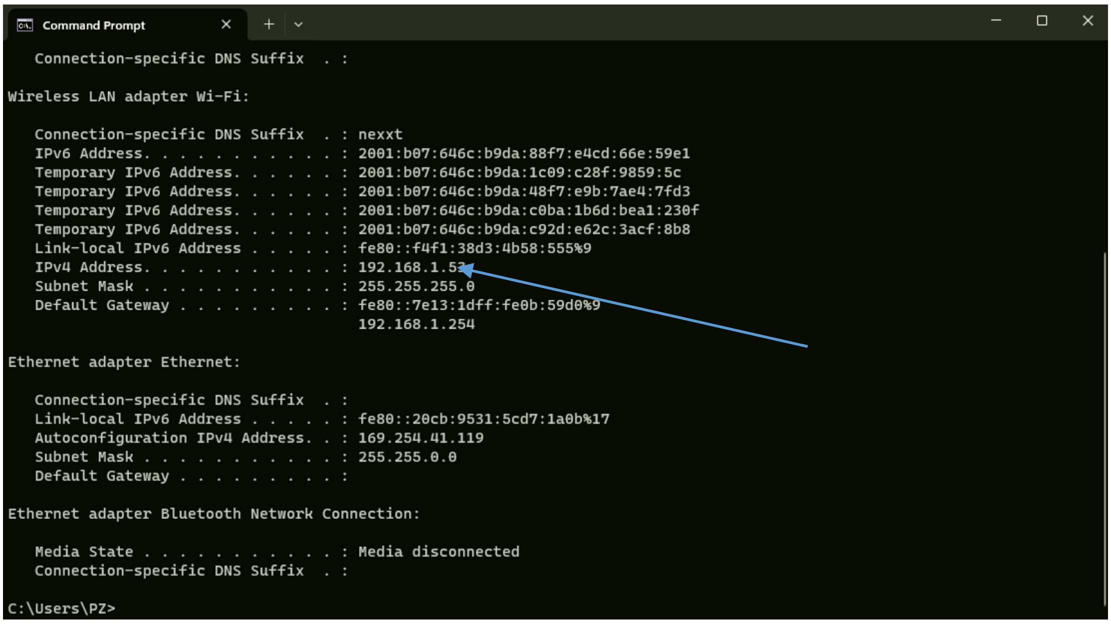
Manufacturer:
Intel

Description:
Intel(R) Ethernet Connection (17) I219-V

Driver version:

Scroll down to find the IPv4 address.

How to find the Server's IP (Using Windows Command Line)



```
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : nexxt
   IPv6 Address. . . . . . . . . . . : 2001:b07:646c:b9da:88f7:e4cd:66e:59e1
   Temporary IPv6 Address. . . . . . : 2001:b07:646c:b9da:1c09:c28f:9859:5c
   Temporary IPv6 Address. . . . . . : 2001:b07:646c:b9da:48f7:e9b:7ae4:7fd3
   Temporary IPv6 Address. . . . . . : 2001:b07:646c:b9da:c0ba:1b6d:bea1:230f
   Temporary IPv6 Address. . . . . . : 2001:b07:646c:b9da:c92d:e62c:3acf:8b8
   Link-local IPv6 Address . . . . . : fe80::f4f1:38d3:4b58:555%9
   IPv4 Address. . . . . . . . . . . : 192.168.1.53
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::7e13:1dff:fe0b:59d0%9
                                       192.168.1.254

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::20cb:9531:5cd7:1a0b%17
   Autoconfiguration IPv4 Address. . : 169.254.41.119
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

C:\Users\PZ>
```

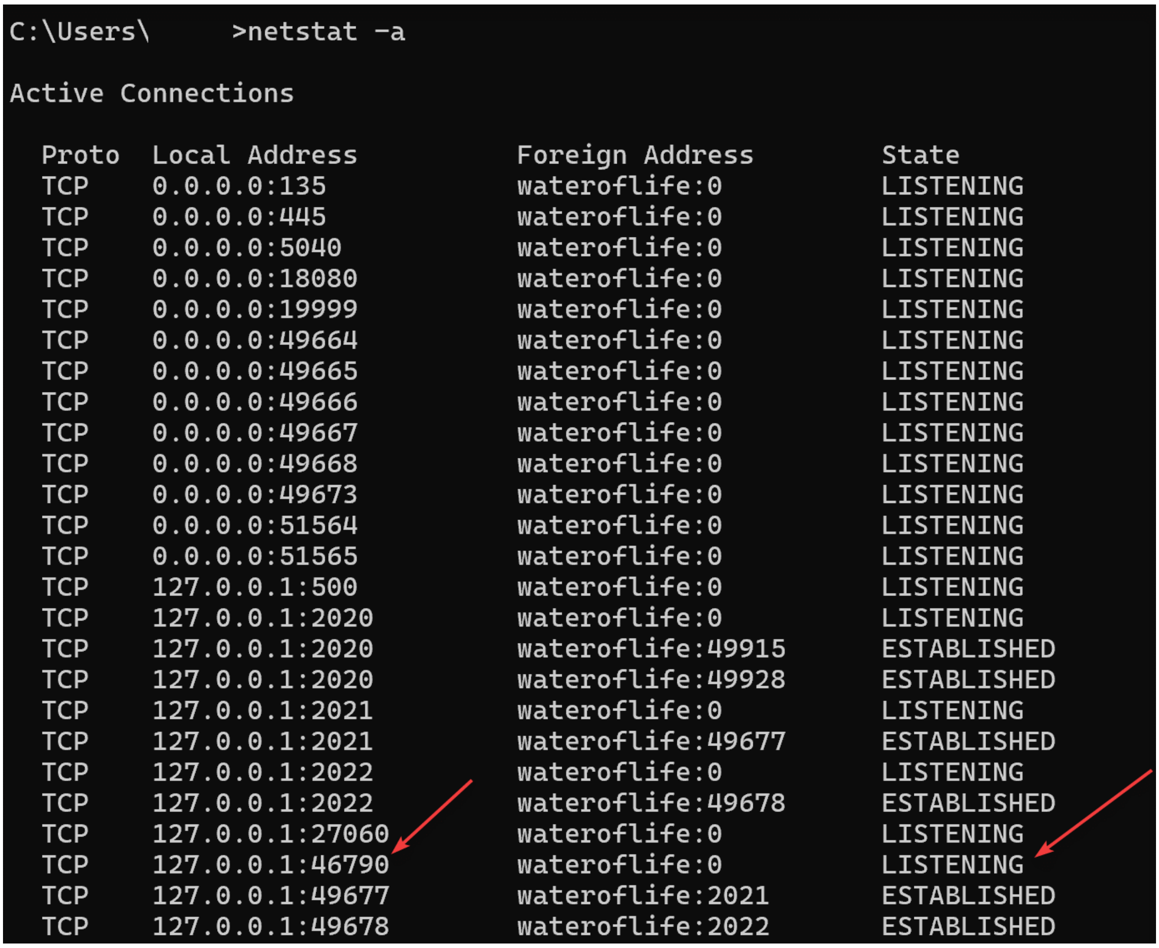
Alternatively use ipconfig at the command line.

Port Configurations and Firewall Settings

Command RTMP uses one unified port across all servers and computers for RTMP functionality. All computers, clients and server, need to use the same port. By default this port is 46790, but can be changed. Command RTMP uses TCP/IP.

The server must be able to accept external connections on the selected port, and thus your firewall settings and other security settings must permit CommandHost.exe to listen on its port for incoming connections.

An example netstat readout, showing CommandHost.exe listening on the default port



```
C:\Users\     >netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            wateroflife:0          LISTENING
  TCP    0.0.0.0:445            wateroflife:0          LISTENING
  TCP    0.0.0.0:5040           wateroflife:0          LISTENING
  TCP    0.0.0.0:18080          wateroflife:0          LISTENING
  TCP    0.0.0.0:19999          wateroflife:0          LISTENING
  TCP    0.0.0.0:49664          wateroflife:0          LISTENING
  TCP    0.0.0.0:49665          wateroflife:0          LISTENING
  TCP    0.0.0.0:49666          wateroflife:0          LISTENING
  TCP    0.0.0.0:49667          wateroflife:0          LISTENING
  TCP    0.0.0.0:49668          wateroflife:0          LISTENING
  TCP    0.0.0.0:49673          wateroflife:0          LISTENING
  TCP    0.0.0.0:51564          wateroflife:0          LISTENING
  TCP    0.0.0.0:51565          wateroflife:0          LISTENING
  TCP    127.0.0.1:500          wateroflife:0          LISTENING
  TCP    127.0.0.1:2020         wateroflife:0          LISTENING
  TCP    127.0.0.1:2020         wateroflife:49915      ESTABLISHED
  TCP    127.0.0.1:2020         wateroflife:49928      ESTABLISHED
  TCP    127.0.0.1:2021         wateroflife:0          LISTENING
  TCP    127.0.0.1:2021         wateroflife:49677      ESTABLISHED
  TCP    127.0.0.1:2022         wateroflife:0          LISTENING
  TCP    127.0.0.1:2022         wateroflife:49678      ESTABLISHED
  TCP    127.0.0.1:27060        wateroflife:0          LISTENING
  TCP    127.0.0.1:46790        wateroflife:0          LISTENING
  TCP    127.0.0.1:49677        wateroflife:2021       ESTABLISHED
  TCP    127.0.0.1:49678        wateroflife:2022       ESTABLISHED
```

Installation and Setup

This manual assumes you have the latest version of CPE available installed correctly on your computer.

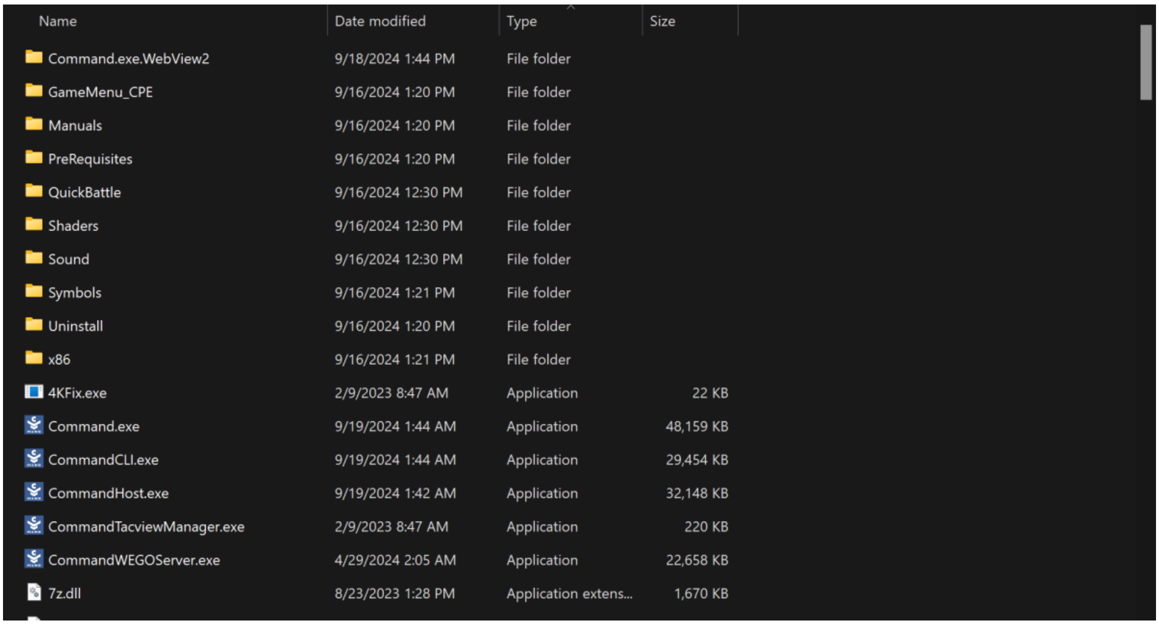
The databases you are using must be distributed to all of the computers, servers and clients, and placed in the normal DB directory.

Place your correct License.dat files on all computers, and the License_MP.dat on the server computer. Contact our tech support for assistance with licensing.

Server Setup

The server for Command RTMP is called **CommandHost.exe**. It is located in the same directory as Command.exe. (By default, C:\Program Files (x86)\Command Professional Edition 2\).

A default C:\Program Files (x86)\Command Professional Edition 2\ directory tree

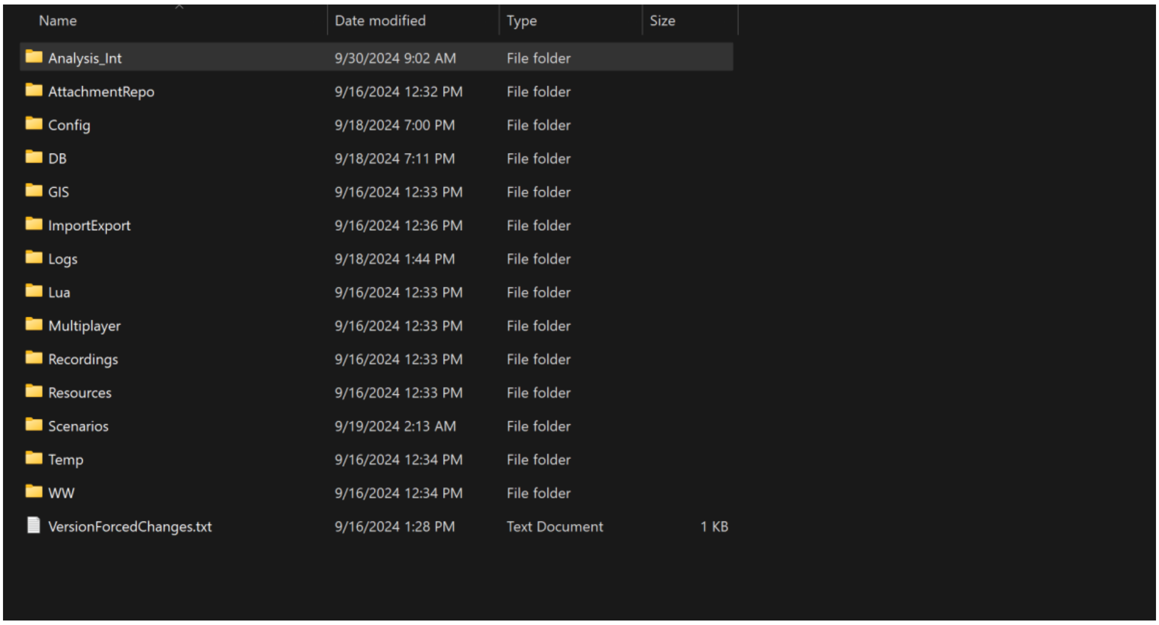


Name	Date modified	Type	Size
Command.exe.WebView2	9/18/2024 1:44 PM	File folder	
GameMenu_CPE	9/16/2024 1:20 PM	File folder	
Manuals	9/16/2024 1:20 PM	File folder	
PreRequisites	9/16/2024 1:20 PM	File folder	
QuickBattle	9/16/2024 12:30 PM	File folder	
Shaders	9/16/2024 12:30 PM	File folder	
Sound	9/16/2024 12:30 PM	File folder	
Symbols	9/16/2024 1:21 PM	File folder	
Uninstall	9/16/2024 1:20 PM	File folder	
x86	9/16/2024 1:21 PM	File folder	
4KFix.exe	2/9/2023 8:47 AM	Application	22 KB
Command.exe	9/19/2024 1:44 AM	Application	48,159 KB
CommandCLI.exe	9/19/2024 1:44 AM	Application	29,454 KB
CommandHost.exe	9/19/2024 1:42 AM	Application	32,148 KB
CommandTacviewManager.exe	2/9/2023 8:47 AM	Application	220 KB
CommandWEGOServer.exe	4/29/2024 2:05 AM	Application	22,658 KB
7z.dll	8/23/2023 1:28 PM	Application extens...	1,670 KB

RTMP can be blocked by antivirus, by power saving modes, by firewalls, by misconfigured network setups.

The Command RTMP server looks for configuration files in the same directory as Command.exe does, by default C:\ProgramData\Command Professional Edition 2\Config\.

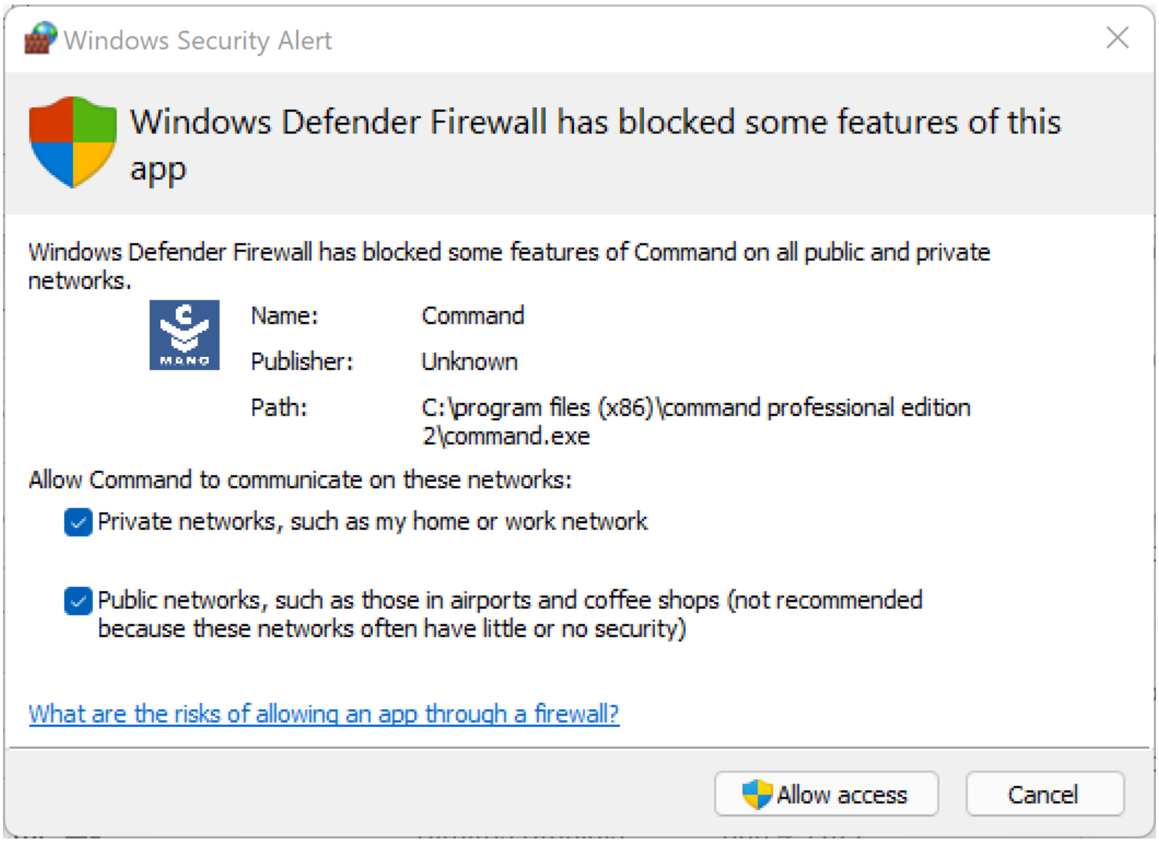
A default C:\ProgramData\Command Professional Edition 2\ directory tree.



Name	Date modified	Type	Size
Analysis_Int	9/30/2024 9:02 AM	File folder	
AttachmentRepo	9/16/2024 12:32 PM	File folder	
Config	9/18/2024 7:00 PM	File folder	
DB	9/18/2024 7:11 PM	File folder	
GIS	9/16/2024 12:33 PM	File folder	
ImportExport	9/16/2024 12:36 PM	File folder	
Logs	9/18/2024 1:44 PM	File folder	
Lua	9/16/2024 12:33 PM	File folder	
Multiplayer	9/16/2024 12:33 PM	File folder	
Recordings	9/16/2024 12:33 PM	File folder	
Resources	9/16/2024 12:33 PM	File folder	
Scenarios	9/19/2024 2:13 AM	File folder	
Temp	9/16/2024 12:34 PM	File folder	
WW	9/16/2024 12:34 PM	File folder	
VersionForcedChanges.txt	9/16/2024 1:28 PM	Text Document	1 KB

CommandHost.exe looks in the Config folder (default, C:\ProgramData\Command Professional Edition 2\Config\) for its settings, including event export settings.

Windows Defender Firewall



Windows Security Alert

Windows Defender Firewall has blocked some features of this app

Windows Defender Firewall has blocked some features of Command on all public and private networks.

Name: Command

Publisher: Unknown

Path: C:\program files (x86)\command professional edition 2\command.exe

Allow Command to communicate on these networks:

☑ Private networks, such as my home or work network

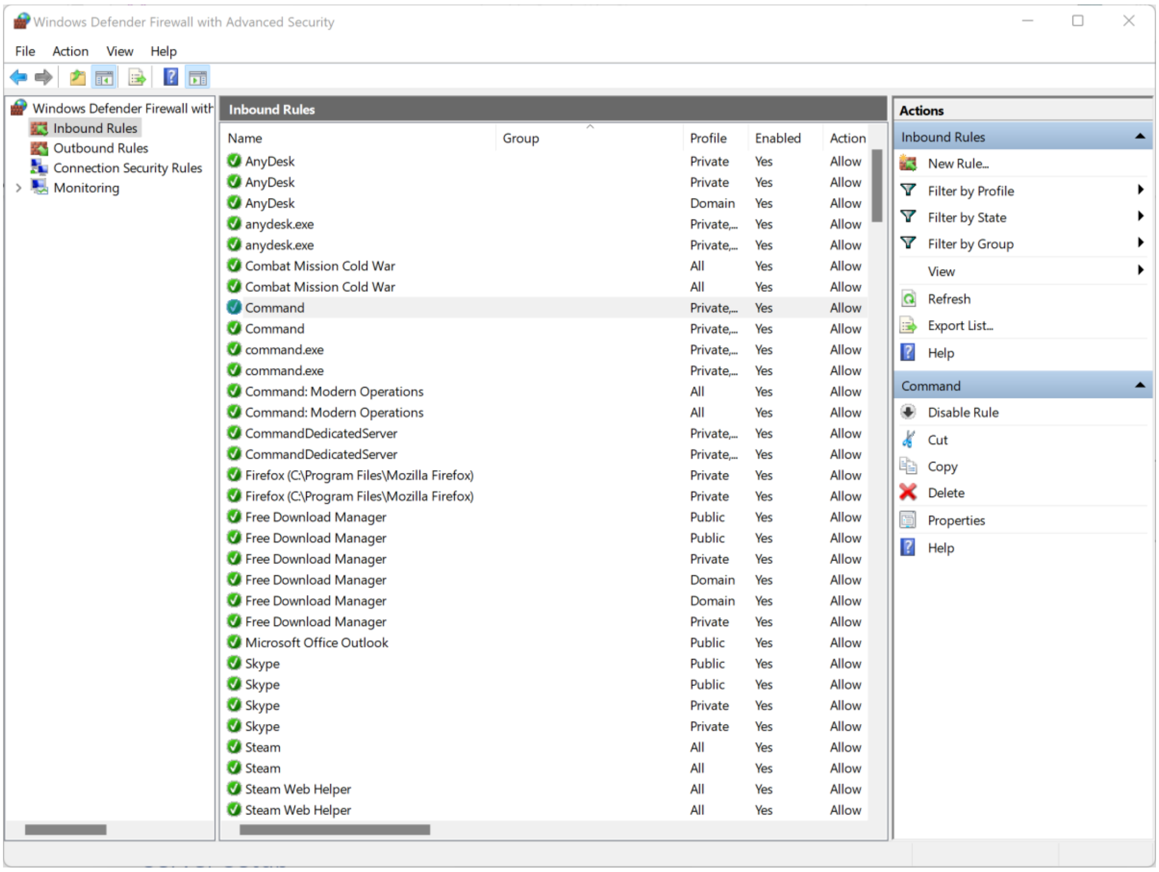
☑ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

When this dialog appears, one must allow Command and CommandHost to communicate across the network profile that corresponds to your computer network. If your network is correctly configured, this will be a “private network” but often computers will have their network profile incorrectly configured, so checking both boxes on this dialog box is recommended as this allows Command through a private network as well as a public network. **This is an essential step, one of the most common errors seen in RTMP installations are firewalls blocking RTMP from communicating on the network.**

This firewall setting can be further configured in the “Windows Defender Firewall with Advanced Security” configuration panel.

If you accidentally press cancel on the Windows Firewall dialog, firewall settings can be set in the Windows Defender Firewall settings.



Name	Group	Profile	Enabled	Action
AnyDesk		Private	Yes	Allow
AnyDesk		Private	Yes	Allow
AnyDesk		Domain	Yes	Allow
anydesk.exe		Private,...	Yes	Allow
anydesk.exe		Private,...	Yes	Allow
Combat Mission Cold War		All	Yes	Allow
Combat Mission Cold War		All	Yes	Allow
Command		Private,...	Yes	Allow
Command		Private,...	Yes	Allow
command.exe		Private,...	Yes	Allow
command.exe		Private,...	Yes	Allow
Command: Modern Operations		All	Yes	Allow
Command: Modern Operations		All	Yes	Allow
CommandDedicatedServer		Private,...	Yes	Allow
CommandDedicatedServer		Private,...	Yes	Allow
Firefox (C:\Program Files\Mozilla Firefox)		Private	Yes	Allow
Firefox (C:\Program Files\Mozilla Firefox)		Private	Yes	Allow
Free Download Manager		Public	Yes	Allow
Free Download Manager		Public	Yes	Allow
Free Download Manager		Private	Yes	Allow
Free Download Manager		Domain	Yes	Allow
Free Download Manager		Domain	Yes	Allow
Free Download Manager		Private	Yes	Allow
Microsoft Office Outlook		Public	Yes	Allow
Skype		Public	Yes	Allow
Skype		Public	Yes	Allow
Skype		Private	Yes	Allow
Skype		Private	Yes	Allow
Steam		All	Yes	Allow
Steam		All	Yes	Allow
Steam Web Helper		All	Yes	Allow
Steam Web Helper		All	Yes	Allow

Configuring Server Settings

Configure the event export settings normally like any sort of Command PE install on the server's side. Client event export settings are not used (as the simulation only runs on the server). These settings are located in your Config folder on the server computer (the computer that will run CommandHost.exe).

Client Setup

Verifying Command.exe Installation

An important stage before attempting a Command RTMP game is to verify that Command PE works on all of the computers. Fire up Command normally on all the client computers, open a scenario (ideally the exact scenario you plan to play) and make sure it works. This checks for the existence of the correct DB on the computer, and checks to make sure Command works on the computer correctly.

Configuring Client Settings (RealtimeMP.ini)

In the Command client side configuration folder (the Config folder on the Client computer) there is a new ini file for preconfiguring Command RTMP called RealtimeMP.ini. This allows you to preset player color, locked side, and player rights (Umpire/Observer/Player).

An example RealtimeMP.ini locking player to "Blue" side for a server at 127.0.0.1.

```
[Player]
Name=Rory Anderson
R=0
G=255
B=255
[Rights]
LockedSide=Blue
Rights=Player
ShowPeerViewports=1
[Default]
IP=127.0.0.1
IP2=192.168.32.2
```

First we have the [**Player**] block:

> **Name**, defines the name of the player. This will be displayed on their viewport, where there mouse is located among other places in game.
>
> **R**,**G**,**B**, these are integers between 0 and 255 that define the color of the player.

Second, we have the [**Rights**] block:

> **LockedSide**, defines the name of the side in the Command scenario that the player will be locked to. This doesn't apply to Umpires. If the side doesn't exist in the scenario, the player will not be locked to a side.
>
> **Rights**, this is the setting that tells RTMP if this client is a Player, Umpire or Observer. Defines the role of the player.
>
> **ShowPeerViewports**, this is 0 or 1 and defines if this client can see other user's viewports.

Finally, we have the [**Default**] block:

> **IP**, defines the IP of the server.
>
> **IP2**, defines the local adapter to bind to. Usually Windows networking doesn't need this to be set, but we've found it useful to have as an option. This should be the IP of the adapter you want to use for RTMP. **This doesn't need to be set usually.**

RealtimeMP.ini examples

A few examples might be in order.

A player with a dark blue color being locked to “Blue Navy” for a server at 192.168.1.45

```
[Player]
Name=Rory Anderson
R=0
G=0
B=255
[Rights]
LockedSide=Blue Navy
Rights=Player
ShowPeerViewports=1
[Default]
IP=192.168.1.45
```

An umpire with a white color. (Note, locked side doesn’t matter to Umpires)

```
[Player]
Name=White Cell
R=255
G=255
B=255
[Rights]
LockedSide=Blue Navy
Rights=Umpire
ShowPeerViewports=1
[Default]
IP=192.168.1.45
```

A red airforce player, with red color, locked to side “Red Airforce”

```
[Player]
Name=43rd Bad Guy Aviation Wing
R=255
G=0
B=0
[Rights]
LockedSide=Red Airforce
Rights=Player
ShowPeerViewports=0
[Default]
IP=192.168.1.45
```

Network Configuration

CommandRTMP doesn’t require an internet connection, but certain tiers of licenses do require an internet connection. For more information about licenses, please contact Matrix Pro tech support.

The fundamental expectation is that all the computers involved in an RTMP wargame are able to communicate in a stable manner over the network with each other. RTMP can be blocked by antivirus, by power saving modes, by firewalls, by misconfigured network setups, and fixing those items are beyond the scope of this document.

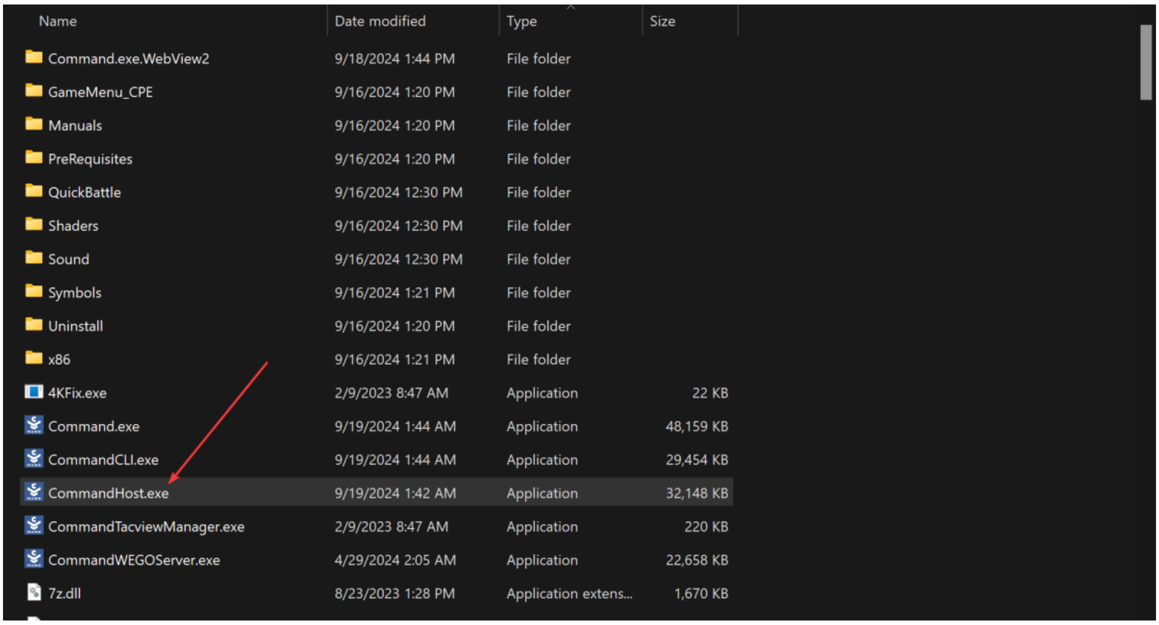
Starting an RTMP Session

At this point you have the server and clients all set up. Your RealtimeMP.ini files are prepared ahead of time for each client and you are ready to start playing Command RTMP across a network.

The basic lifecycle of a Command RTMP Session

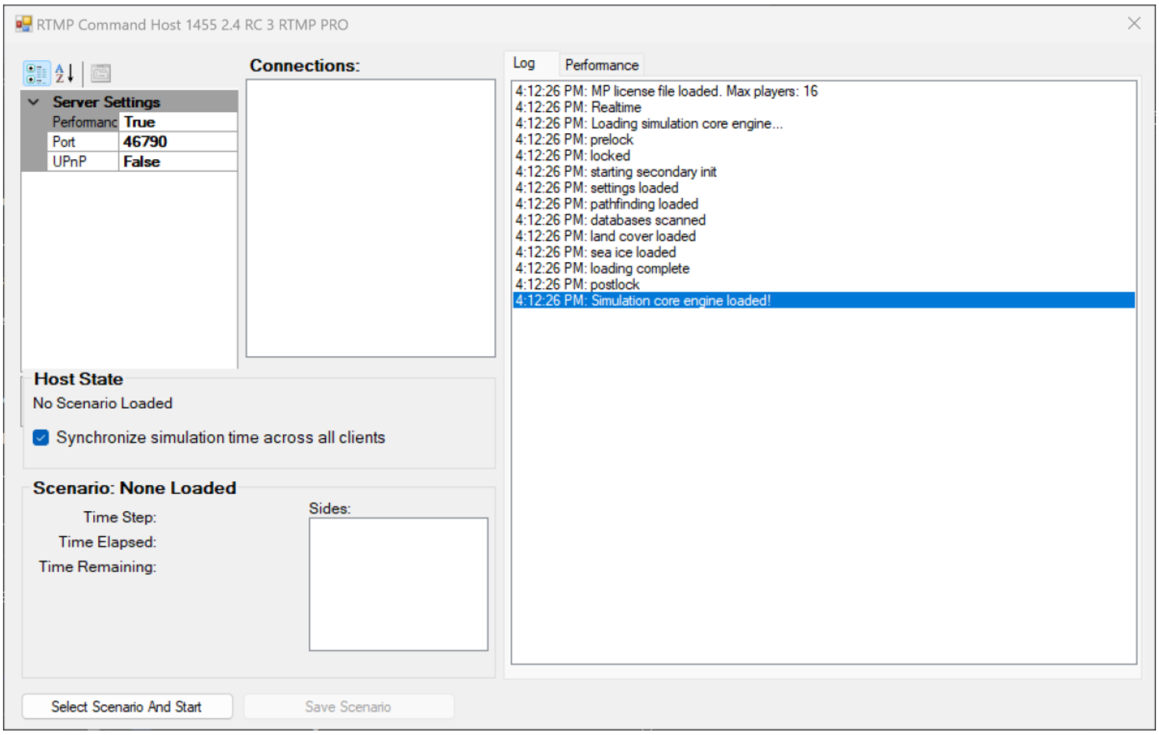
- All computers must be connected together on a network.
- Server computer launches CommandHost.exe
- Server computer selects an initial scenario
- Client computers connect to server
- And now an RTMP game is running!

Launching the Server



Name	Date modified	Type	Size
Command.exe.WebView2	9/18/2024 1:44 PM	File folder	
GameMenu_CPE	9/16/2024 1:20 PM	File folder	
Manuals	9/16/2024 1:20 PM	File folder	
PreRequisites	9/16/2024 1:20 PM	File folder	
QuickBattle	9/16/2024 12:30 PM	File folder	
Shaders	9/16/2024 12:30 PM	File folder	
Sound	9/16/2024 12:30 PM	File folder	
Symbols	9/16/2024 1:21 PM	File folder	
Uninstall	9/16/2024 1:20 PM	File folder	
x86	9/16/2024 1:21 PM	File folder	
4KFix.exe	2/9/2023 8:47 AM	Application	22 KB
Command.exe	9/19/2024 1:44 AM	Application	48,159 KB
CommandCLI.exe	9/19/2024 1:44 AM	Application	29,454 KB
CommandHost.exe	9/19/2024 1:42 AM	Application	32,148 KB
CommandTacviewManager.exe	2/9/2023 8:47 AM	Application	220 KB
CommandWEGOServer.exe	4/29/2024 2:05 AM	Application	22,658 KB
7z.dll	8/23/2023 1:28 PM	Application extens...	1,670 KB

In your Command directory, launch CommandHost.exe


RTMP Command Host 1455 2.4 RC 3 RTMP PRO
Connections:
Log
Performance
Sides:
Select Scenario And Start
Save Scenario

Server Settings

Performanc	True
Port	46790
UPnP	False

Host State

No Scenario Loaded

☑ Synchronize simulation time across all clients

Scenario: None Loaded

Time Step:

Time Elapsed:

Time Remaining:

```
4:12:26 PM: MP license file loaded. Max players: 16
4:12:26 PM: Realtime
4:12:26 PM: Loading simulation core engine...
4:12:26 PM: prelock
4:12:26 PM: locked
4:12:26 PM: starting secondary init
4:12:26 PM: settings loaded
4:12:26 PM: pathfinding loaded
4:12:26 PM: databases scanned
4:12:26 PM: land cover loaded
4:12:26 PM: sea ice loaded
4:12:26 PM: loading complete
4:12:26 PM: postlock
4:12:26 PM: Simulation core engine loaded!
```

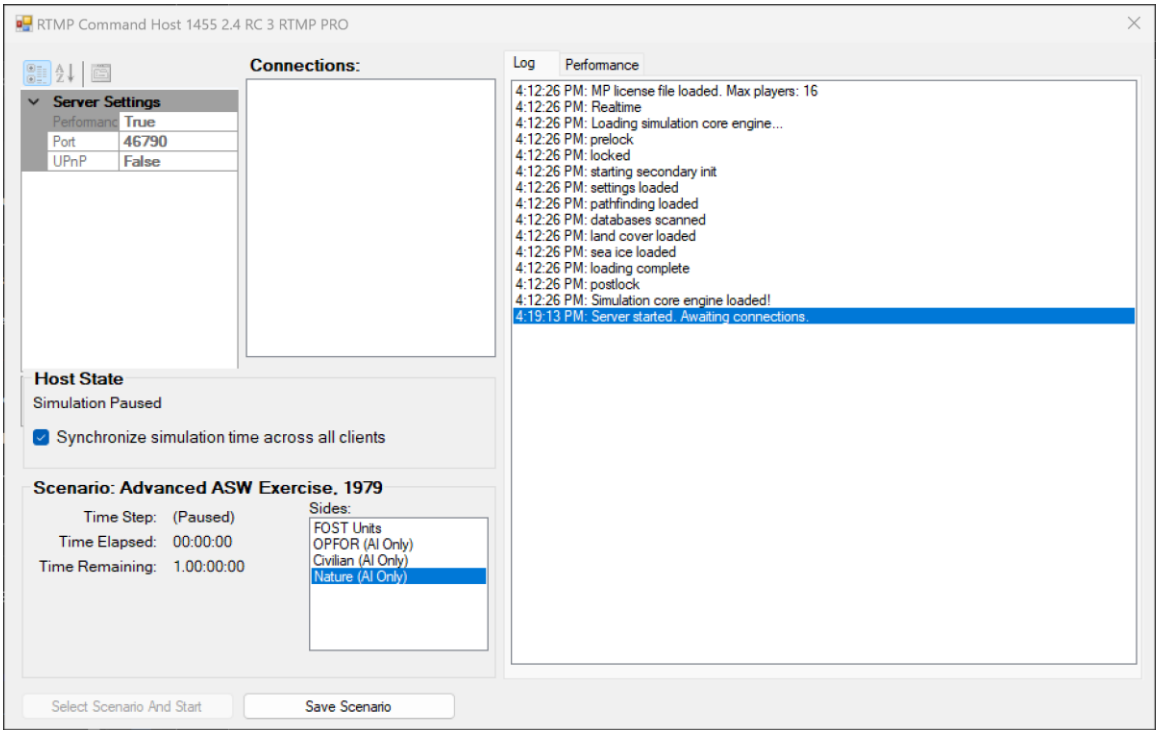
CommandHost.exe, the RTMP server, upon startup, before selecting a scenario.

To launch the server, run CommandHost.exe. It will do some pre-game checks, and when you see “Simulation core engine loaded!” the server is ready to run.

Selecting and Loading a Scenario

Select a scenario with the “Select Scenario And Start” button. Ensure all client computers have the required DBs ahead of time for the scenario.

When you see “Server started. Awaiting connections.” The server has started and is waiting, paused, at the first moment of the scenario, for clients to connect.



RTMP Command Host 1455 2.4 RC 3 RTMP PRO

Server Settings

Performanc	True
Port	46790
UPnP	False

Connections:

```
4:12:26 PM: MP license file loaded. Max players: 16
4:12:26 PM: Realtime
4:12:26 PM: Loading simulation core engine...
4:12:26 PM: prelock
4:12:26 PM: locked
4:12:26 PM: starting secondary init
4:12:26 PM: settings loaded
4:12:26 PM: pathfinding loaded
4:12:26 PM: databases scanned
4:12:26 PM: land cover loaded
4:12:26 PM: sea ice loaded
4:12:26 PM: loading complete
4:12:26 PM: postlock
4:12:26 PM: Simulation core engine loaded!
4:19:13 PM: Server started. Awaiting connections.
```

Host State

Simulation Paused

☑ Synchronize simulation time across all clients

Scenario: Advanced ASW Exercise, 1979

Time Step:	(Paused)
Time Elapsed:	00:00:00
Time Remaining:	1.00:00:00

Sides:
FOST Units
OPFOR (AI Only)
Civilian (AI Only)
Nature (AI Only)

CommandHost.exe, after selecting a scenario. This server is ready to accept client connections!

Server UI Overviews

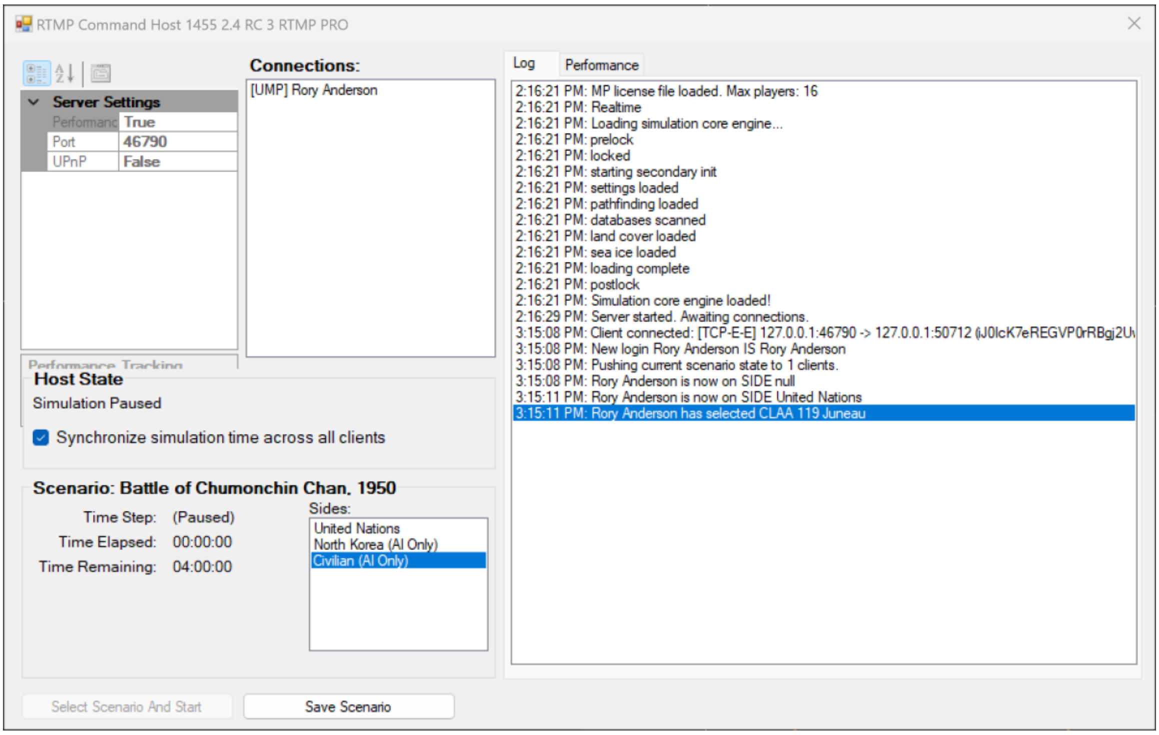
The server UI is intentionally very simple. All it requires is a single button press to select a scenario and start.

There is one other button, Save Scenario, and that creates a save of the current state on the server.

We have a few settings on the top right, set “Performance Tracking” to false for more performance. Set the port before starting the server. Autosave Interval is the number of seconds between automatic autosaves, set to 0 for no periodic server autosaves. (This improves performance)

There is a log of server events on the right. As well as a performance tab, showing the network use of the server.

But that is about it. A very simple server UI.



RTMP Command Host 1455 2.4 RC 3 RTMP PRO

Server Settings

Performanc	True
Port	46790
UPnP	False

Connections:

[UMP] Rory Anderson

Performance Tracking

Host State

Simulation Paused

☑ Synchronize simulation time across all clients

Scenario: Battle of Chumonchin Chan, 1950

Time Step: (Paused)

Time Elapsed: 00:00:00

Time Remaining: 04:00:00

Sides:

United Nations
North Korea (AI Only)
Civilian (AI Only)

```
2:16:21 PM: MP license file loaded. Max players: 16
2:16:21 PM: Realtime
2:16:21 PM: Loading simulation core engine...
2:16:21 PM: prelock
2:16:21 PM: locked
2:16:21 PM: starting secondary init
2:16:21 PM: settings loaded
2:16:21 PM: pathfinding loaded
2:16:21 PM: databases scanned
2:16:21 PM: land cover loaded
2:16:21 PM: sea ice loaded
2:16:21 PM: loading complete
2:16:21 PM: postlock
2:16:21 PM: Simulation core engine loaded!
2:16:29 PM: Server started. Awaiting connections.
3:15:08 PM: Client connected: [TCP-E-E] 127.0.0.1:46790 -> 127.0.0.1:50712 (iJ0lcK7eREGVP0rRBgj2U
3:15:08 PM: New login Rory Anderson IS Rory Anderson
3:15:08 PM: Pushing current scenario state to 1 clients.
3:15:08 PM: Rory Anderson is now on SIDE null
3:15:11 PM: Rory Anderson is now on SIDE United Nations
3:15:11 PM: Rory Anderson has selected CLAA 119 Juneau
```

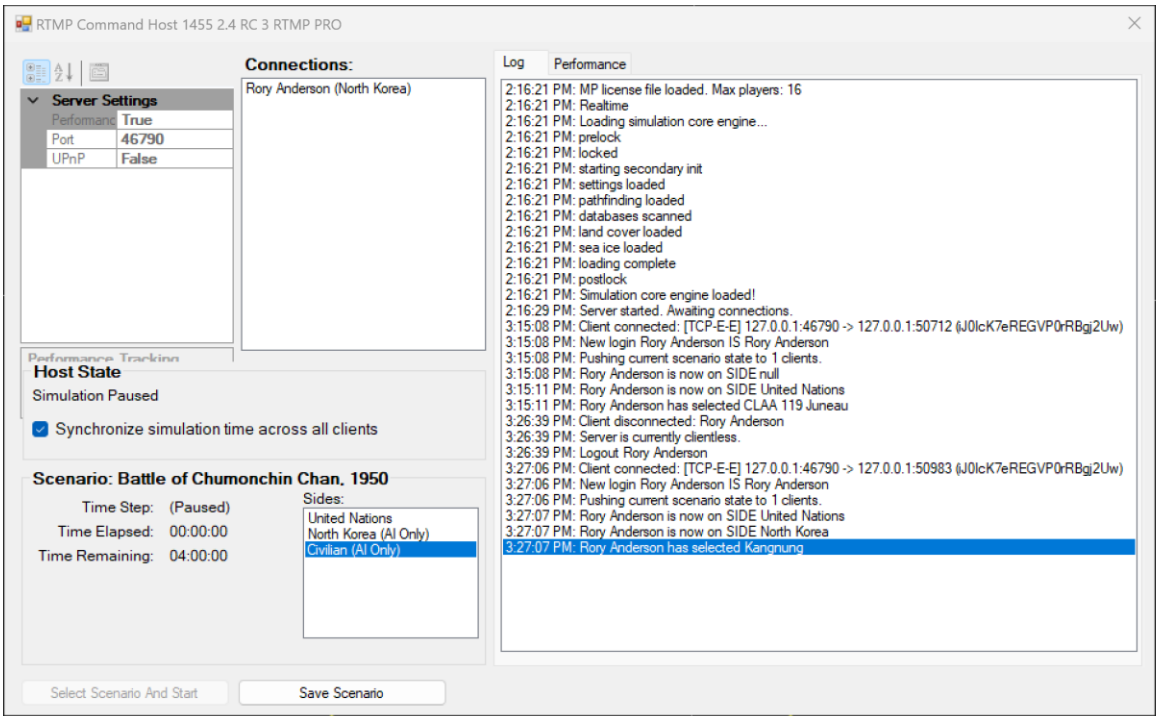
The server UI after a player, with umpire rights, joins.

The connections list, near the left of the UI, lists the currently connected players.

[UMP] indicates the player is an umpire.

[OBS] indicates the player is an observer.

Players without any indication, [UMP] or [OBS] are normal players, without special rights or abilities.



RTMP Command Host 1455 2.4 RC 3 RTMP PRO

Server Settings

Performanc	True
Port	46790
UPnP	False

Connections:

Rory Anderson (North Korea)

Performance Tracking

Host State

Simulation Paused

☑ Synchronize simulation time across all clients

Scenario: Battle of Chumonchin Chan, 1950

Time Step:	(Paused)
Time Elapsed:	00:00:00
Time Remaining:	04:00:00

Sides:

United Nations
North Korea (AI Only)
Civilian (AI Only)

Select Scenario And Start

Save Scenario

Log | Performance

```
2:16:21 PM: MP license file loaded. Max players: 16
2:16:21 PM: Realtime
2:16:21 PM: Loading simulation core engine...
2:16:21 PM: prelock
2:16:21 PM: locked
2:16:21 PM: starting secondary init
2:16:21 PM: settings loaded
2:16:21 PM: pathfinding loaded
2:16:21 PM: databases scanned
2:16:21 PM: land cover loaded
2:16:21 PM: sea ice loaded
2:16:21 PM: loading complete
2:16:21 PM: postlock
2:16:21 PM: Simulation core engine loaded!
2:16:29 PM: Server started. Awaiting connections.
3:15:08 PM: Client connected: [TCP-E-E] 127.0.0.1:46790 -> 127.0.0.1:50712 (iJ0lcK7eREGVP0rRBgj2Uw)
3:15:08 PM: New login Rory Anderson IS Rory Anderson
3:15:08 PM: Pushing current scenario state to 1 clients.
3:15:08 PM: Rory Anderson is now on SIDE null
3:15:11 PM: Rory Anderson is now on SIDE United Nations
3:15:11 PM: Rory Anderson has selected CLAA 119 Juneau
3:26:39 PM: Client disconnected: Rory Anderson
3:26:39 PM: Server is currently clientless.
3:26:39 PM: Logout Rory Anderson
3:27:06 PM: Client connected: [TCP-E-E] 127.0.0.1:46790 -> 127.0.0.1:50983 (iJ0lcK7eREGVP0rRBgj2Uw)
3:27:06 PM: New login Rory Anderson IS Rory Anderson
3:27:06 PM: Pushing current scenario state to 1 clients.
3:27:07 PM: Rory Anderson is now on SIDE United Nations
3:27:07 PM: Rory Anderson is now on SIDE North Korea
3:27:07 PM: Rory Anderson has selected Kangnung
```

The player reconnected as a normal player, without umpire powers.

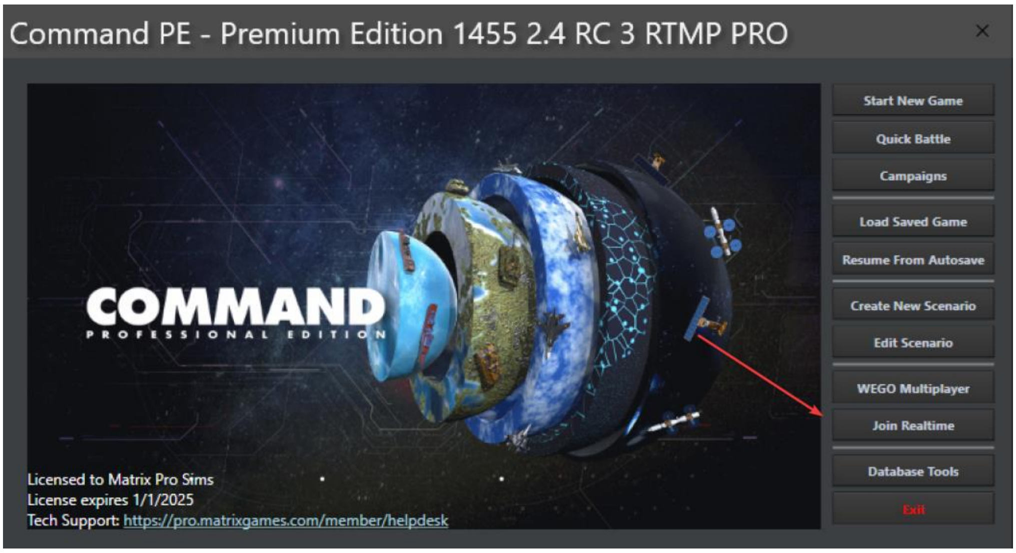
Connecting Clients

Joining a Game

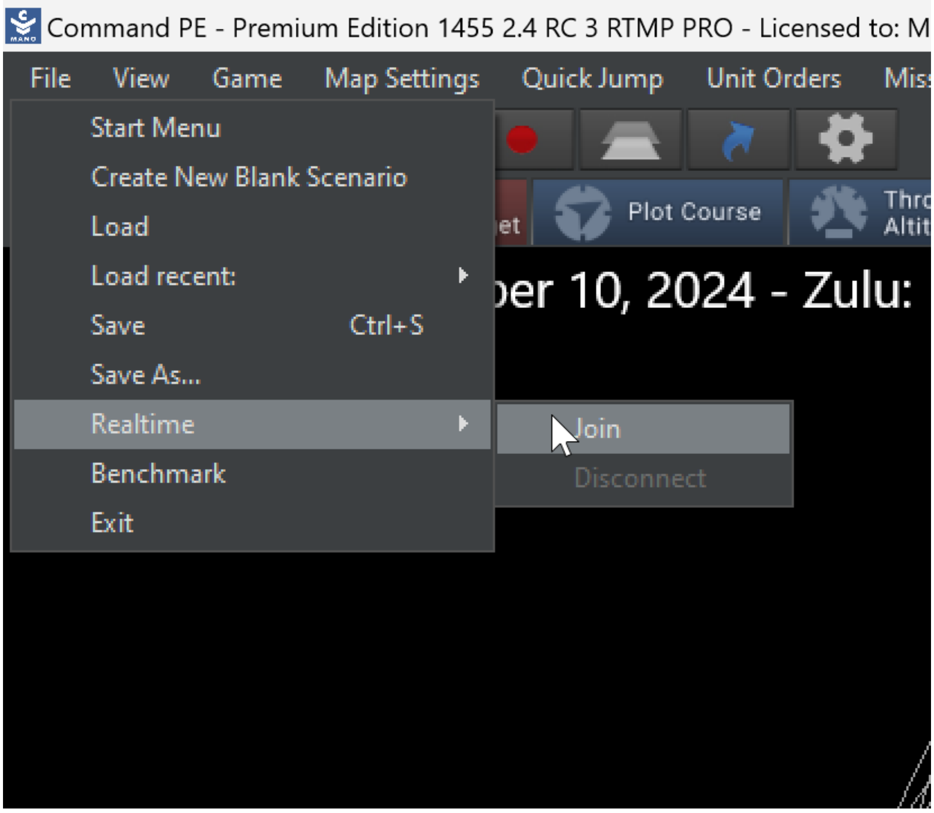
Now, the server is running: lets connect some clients!

First, launch Command like normal.

There are two buttons that allow you to join a game on a server, one on the start screen, and one under File->Realtime on the menu bar.



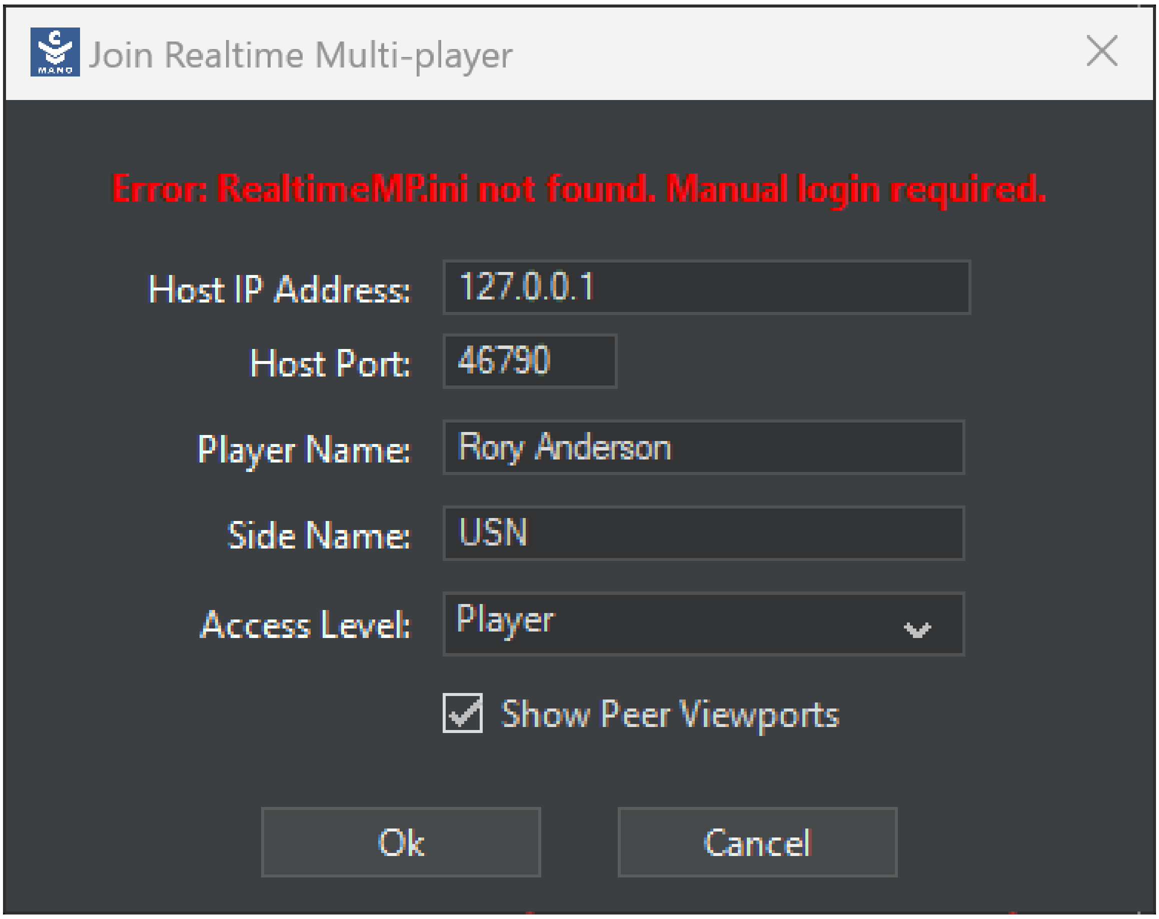
The button “Join Realtime” allows you to connect to a RTMP server.



This menu, under File->Realtime also is equivilient to the button on the start screen above.

Player Configuration (without RealtimeMP.ini)

When joining a server with Command RTMP without a preset RealtimeMP.ini present on the client computer you are presented with this dialog.



Error: RealtimeMP.ini not found. Manual login required.

Host IP Address: 127.0.0.1

Host Port: 46790

Player Name: Rory Anderson

Side Name: USN

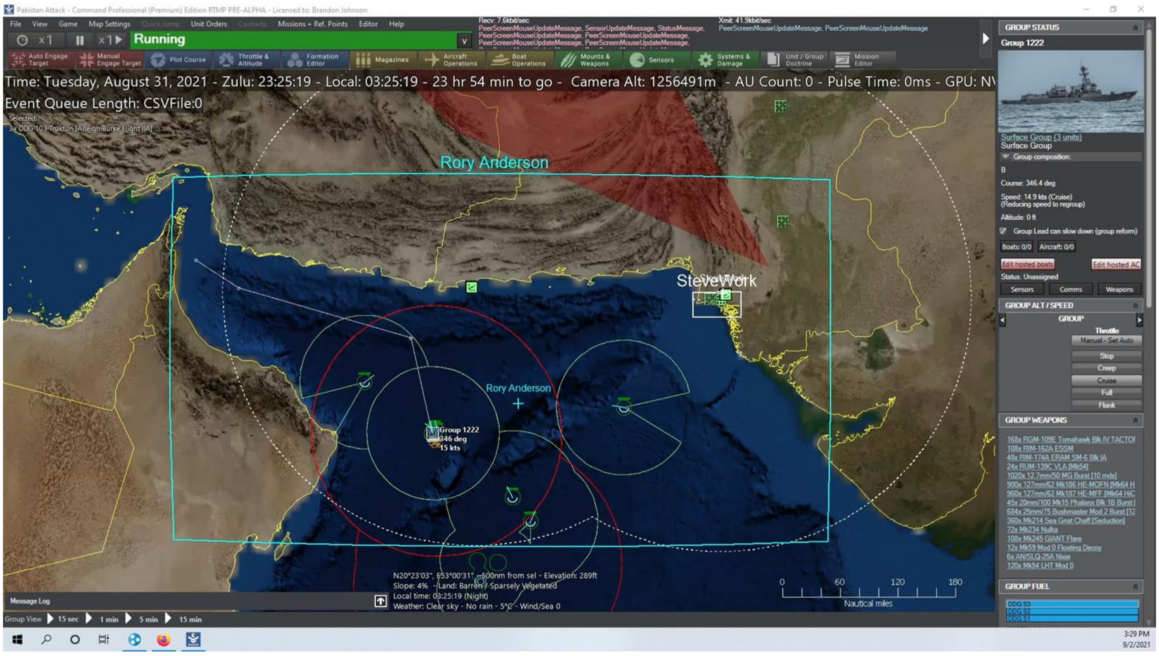
Access Level: Player

☑ Show Peer Viewports

When you don’t have a RealtimeMP.ini…

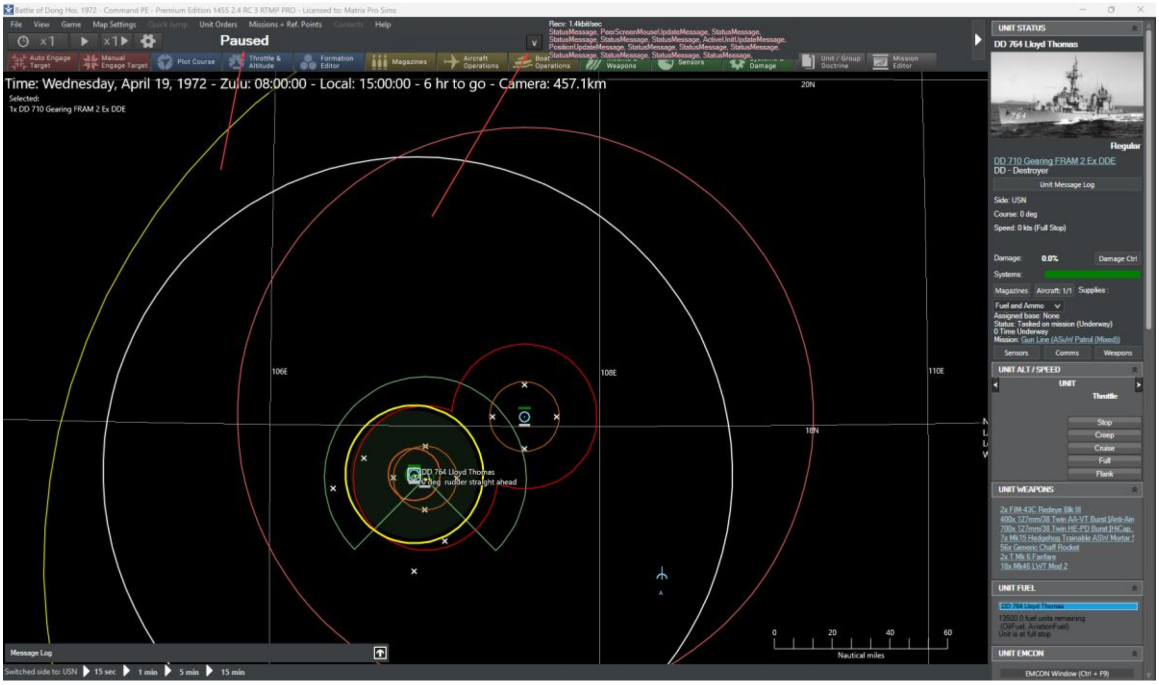
Gameplay Mechanics

So now you are playing RTMP! Congratulations! There is still more to learn.

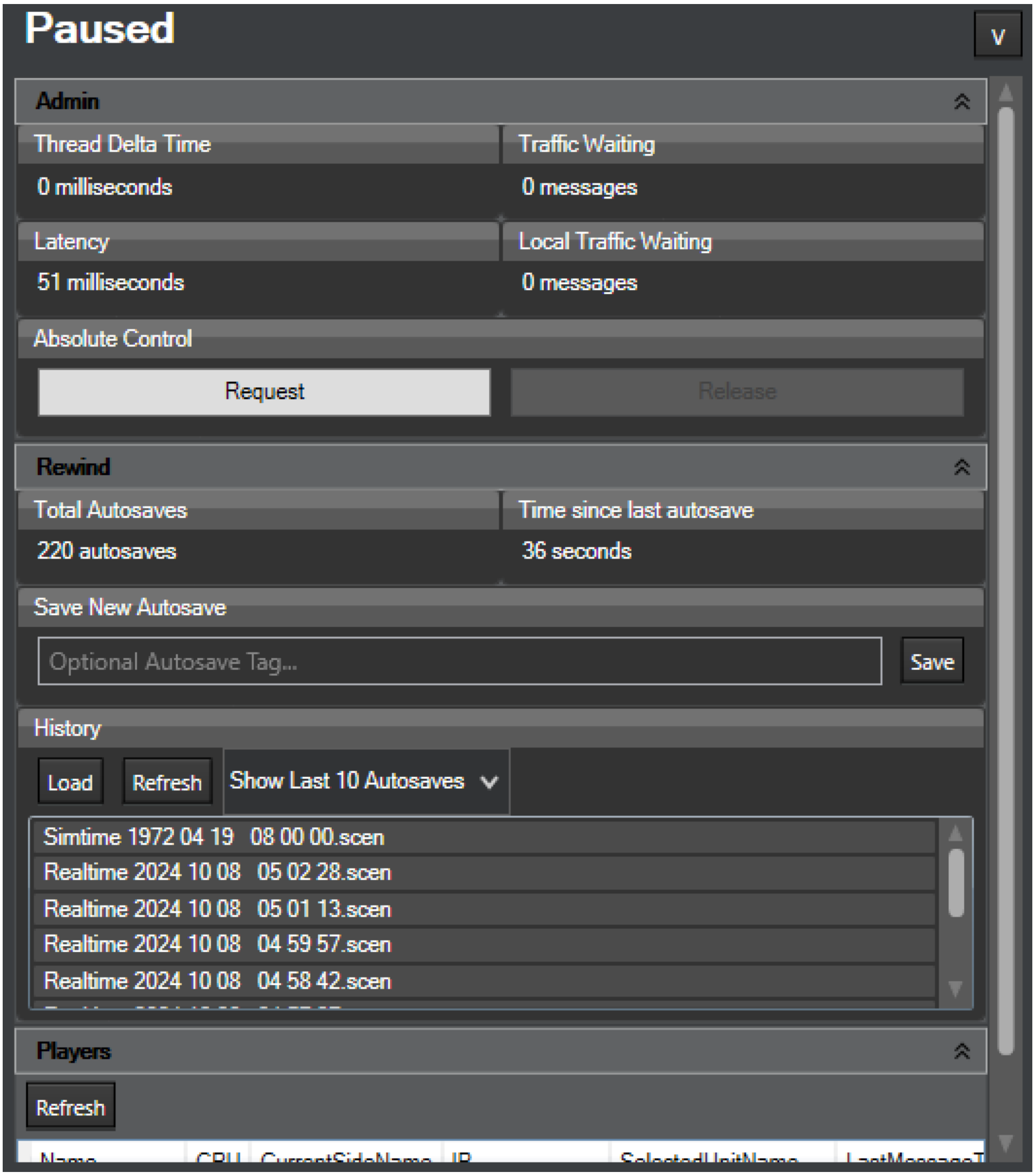

Pakistan Attack - Command Professional (Premium) Edition RTMP PRE-ALPHA - Licensed to: Brandon Johnson
Running
Time: Tuesday, August 31, 2021 - Zulu: 23:25:19 - Local: 03:25:19 - 23 hr 54 min to go - Camera Alt: 1256491m - AU Count: 0 - Pulse Time: 0ms - GPU: N
Event Queue Length: CSVFile:0
Rory Anderson
SteveWork
Group 1222
346 deg
15 kts
GROUP STATUS
Group 1222
Surface Group (3 units)
GROUP ALT / SPEED
GROUP WEAPONS
GROUP FUEL
Message Log
Nautical miles

The RTMP Ribbon

The first thing to cover is the so-called RTMP ribbon. This is the central UI for all things RTMP. It also shows if the game is running or paused, or if some other user has Absolute Control mode (which means you cannot edit the game state, and are waiting for the other user to give up AC mode)


Battle of Dong Hoi, 1972 - Command PE - Premium Edition 1455 2.4 RC 3 RTMP PRO - Licensed to: Matrix Pro Sims
Paused
Time: Wednesday, April 19, 1972 - Zulu: 08:00:00 - Local: 15:00:00 - 6 hr to go - Camera: 457.1km
Selected:
1x DD 710 Gearing FRAM 2 Ex DDE
UNIT STATUS
DD 764 Lloyd Thomas
DD 710 Gearing FRAM 2 Ex DDE
DD - Destroyer
UNIT ALT / SPEED
UNIT WEAPONS
UNIT FUEL
UNIT EMCON
DD 764 Lloyd Thomas
Message Log
Nautical miles

The location of the RTMP ribbon, on top of the normal Command UI. Note the red arrows.



Paused

Admin

Thread Delta Time: 0 milliseconds

Traffic Waiting: 0 messages

Latency: 51 milliseconds

Local Traffic Waiting: 0 messages

Absolute Control

Rewind

Total Autosaves: 220 autosaves

Time since last autosave: 36 seconds

Save New Autosave

History

Show Last 10 Autosaves

Simtime 1972 04 19 08 00 00.scen
Realtime 2024 10 08 05 02 28.scen
Realtime 2024 10 08 05 01 13.scen
Realtime 2024 10 08 04 59 57.scen
Realtime 2024 10 08 04 58 42.scen

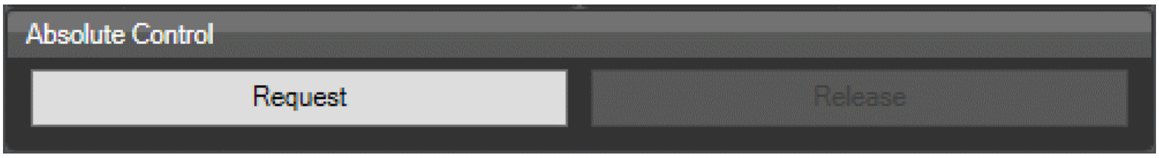
Players

The RTMP ribbon expanded.

Starting from top to bottom, left to right.

Thread Delta Time, Traffic Waiting, Latency and Local Traffic Waiting, these are helpful measures for how overloaded the system is. Higher numbers are worse. These have to do with the underlying network transport layer, and the server’s ability to serve the clients.

Absolute Control (AC)

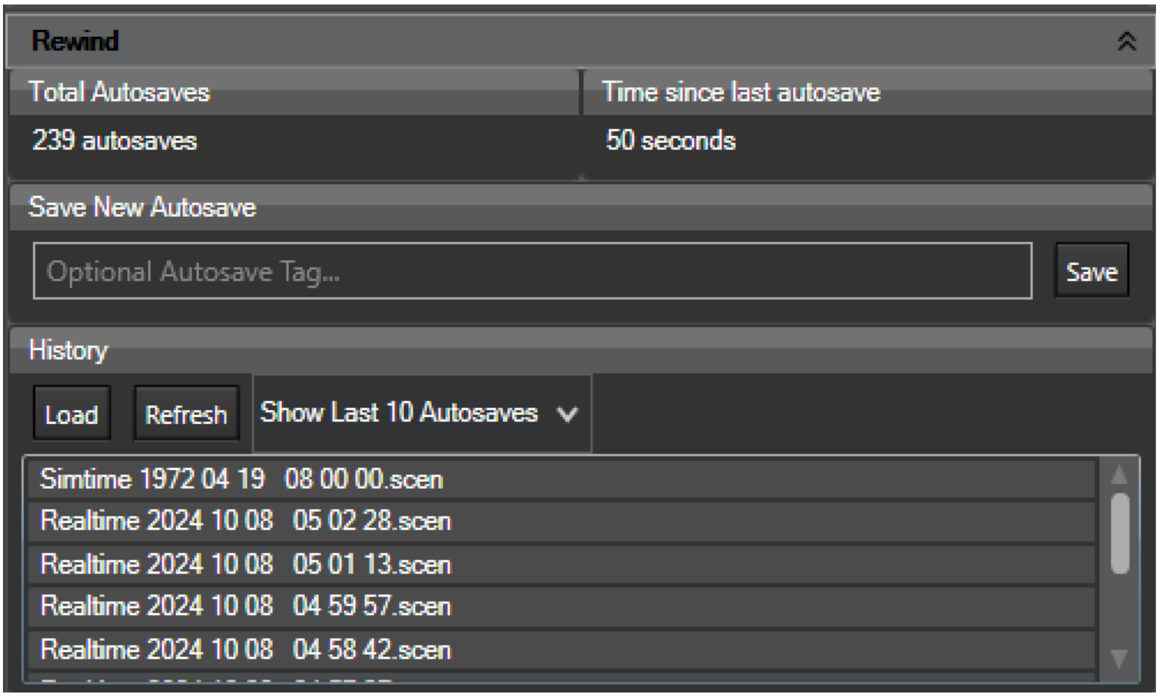


Absolute Control has its own chapter in this document, but here on the RTMP ribbon is how to Request and Release Absolute Control.

Absolute Control pauses the game, and moves control of the game state to the computer that requested it. Outside of Absolute Control (AC), the definitive scenario state is always on the server. When AC is requested by a client, the definitive scenario state is removed from the server and sent to the client. When a client has AC, they have the full scenario state.

Please look at the chapter on Absolute Control.

Autosaves / Rewind



Total Autosaves: 239 autosaves

Time since last autosave: 50 seconds

Save New Autosave

History

Show Last 10 Autosaves

Simtime 1972 04 19 08 00 00.scen
Realtime 2024 10 08 05 02 28.scen
Realtime 2024 10 08 05 01 13.scen
Realtime 2024 10 08 04 59 57.scen
Realtime 2024 10 08 04 58 42.scen

Order Issuing and Execution

Command RTMP plays just like normal Command.

All players can control pause and time acceleration, please set up rules about time acceleration and pausing ahead of time with your players.

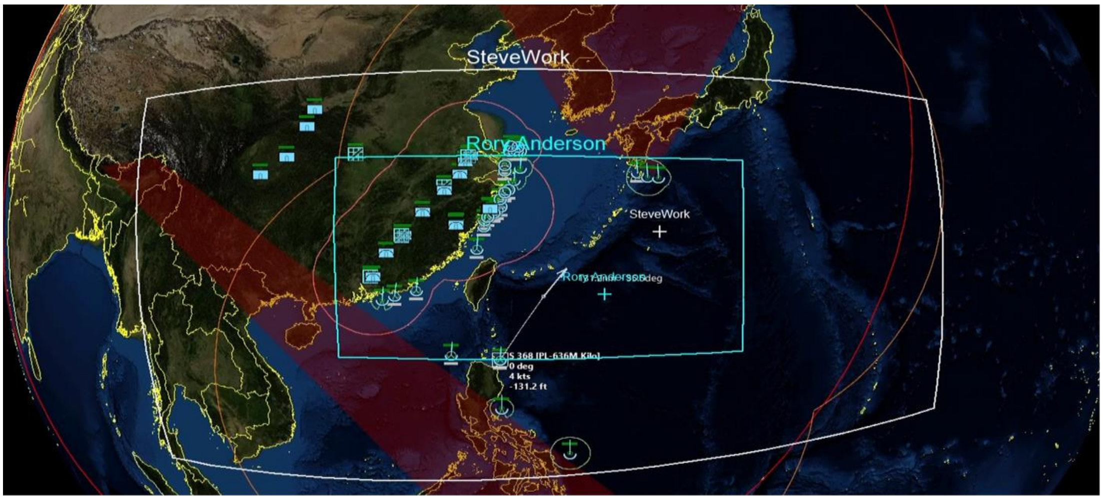
There is no hierarchy or chain of command with players, the last player to issue an order to a unit is the order the unit will follow.

Viewports

One feature of Command RTMP that isn't present in the base game is the ability to see 'viewports.' These are the viewing area, and mouse location, of your fellow players.

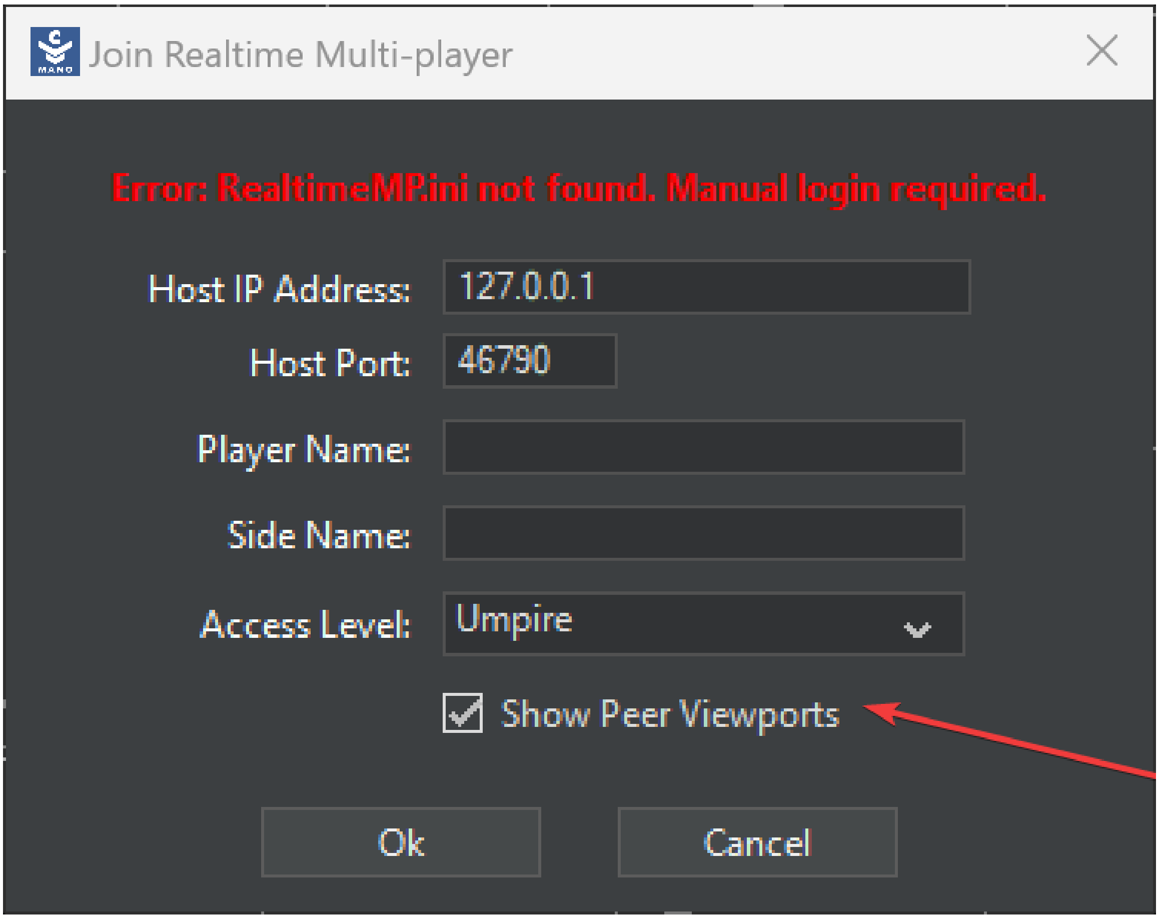
As you pan your map, or move your mouse, your other players on your side can see your movements. And umpires can see all! But players are restricted to just seeing their side's viewports.

This has applications in training, planning and team management.



The 'viewports' are the rectanges labeled "Rory Anderson" and "SteveWork", also note that mouse positions are also visible.

The colors for each player are pre-defined in RealtimeMP.ini. Please see the section on RealtimeMP.ini for more details.



Join Realtime Multi-player

Error: RealtimeMP.ini not found. Manual login required.

Host IP Address: 127.0.0.1

Host Port: 46790

Player Name:

Side Name:

Access Level: Umpire

☑ Show Peer Viewports

This option can disable showing the peer viewports if desired. Also, this is a setting in RealtimeMP.ini called ShowPeerViewports.

Switching Sides (or preventing switching sides) and God's Eye View

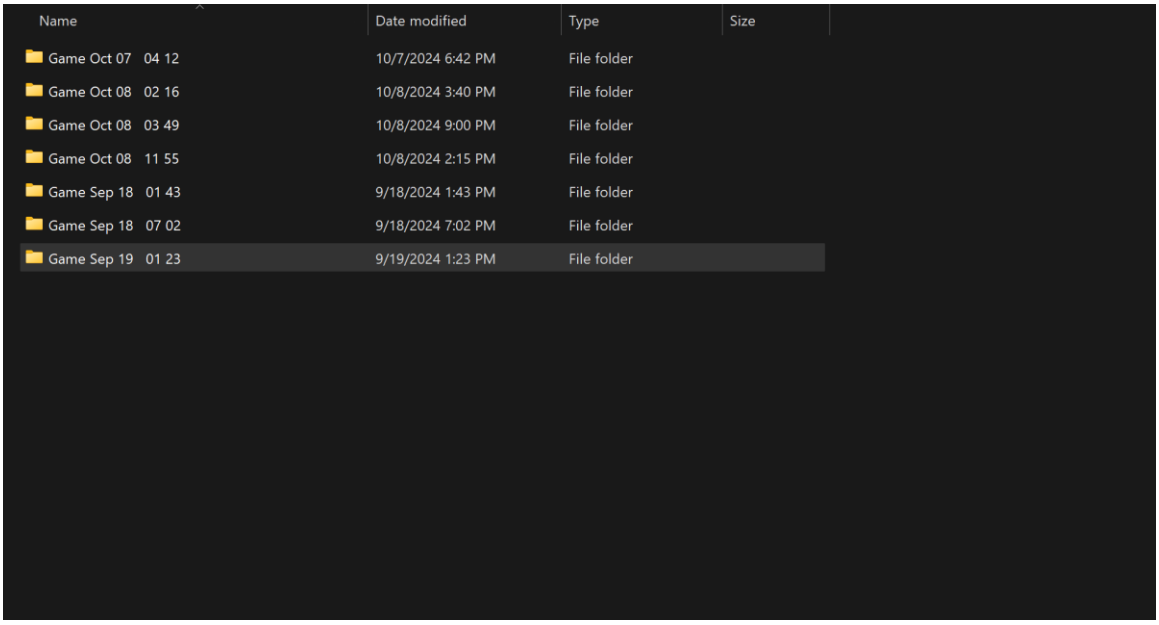
Players who do not have a valid LockedSide defined, can switch sides at will. To lock a player to a side, you need to define a LockedSide in the RealtimeMP.ini or during player connect to a server. The side name and the LockedSide need to have the exact same name. “USN “ and “USN” are considered different sides by this mechanism, note the extra space.

Communication Tools (Lack Thereof)

Command RTMP does not have any built-in communication tools, please use alternative communication tools like in-person vocal communication, cellphones, Discord/Slack/IRC etc.

Saving Games

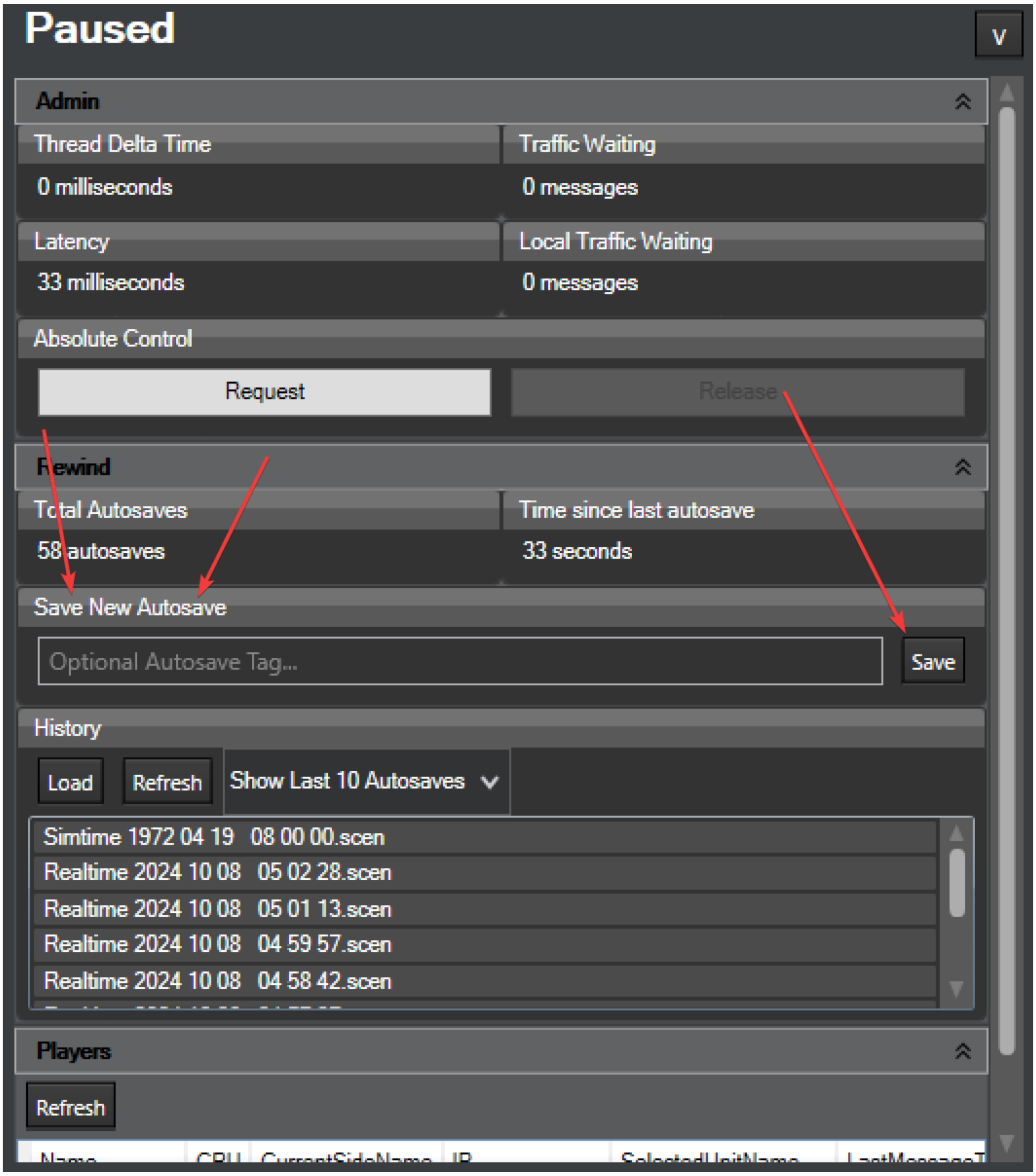
There are three main mechanisms to save the game. By far the simplest is by use of the RTMP ribbon.



Name	Date modified	Type	Size
Game Oct 07 04 12	10/7/2024 6:42 PM	File folder	
Game Oct 08 02 16	10/8/2024 3:40 PM	File folder	
Game Oct 08 03 49	10/8/2024 9:00 PM	File folder	
Game Oct 08 11 55	10/8/2024 2:15 PM	File folder	
Game Sep 18 01 43	9/18/2024 1:43 PM	File folder	
Game Sep 18 07 02	9/18/2024 7:02 PM	File folder	
Game Sep 19 01 23	9/19/2024 1:23 PM	File folder	

The location for automatically saved games is on the **server**’s Scenarios directory, under the sub-directory Realtime (by default, C:\ProgramData\Command Professional Edition 2\Scenarios\Realtime). Each RTMP **server** session is labeled with the date, and the hour and minute of server startup.

RTMP Ribbon Named Autosaves



Paused

Admin

Thread Delta Time: 0 milliseconds

Traffic Waiting: 0 messages

Latency: 33 milliseconds

Local Traffic Waiting: 0 messages

Absolute Control

Rewind

Total Autosaves: 58 autosaves

Time since last autosave: 33 seconds

Save New Autosave

Optional Autosave Tag...

History

Show Last 10 Autosaves

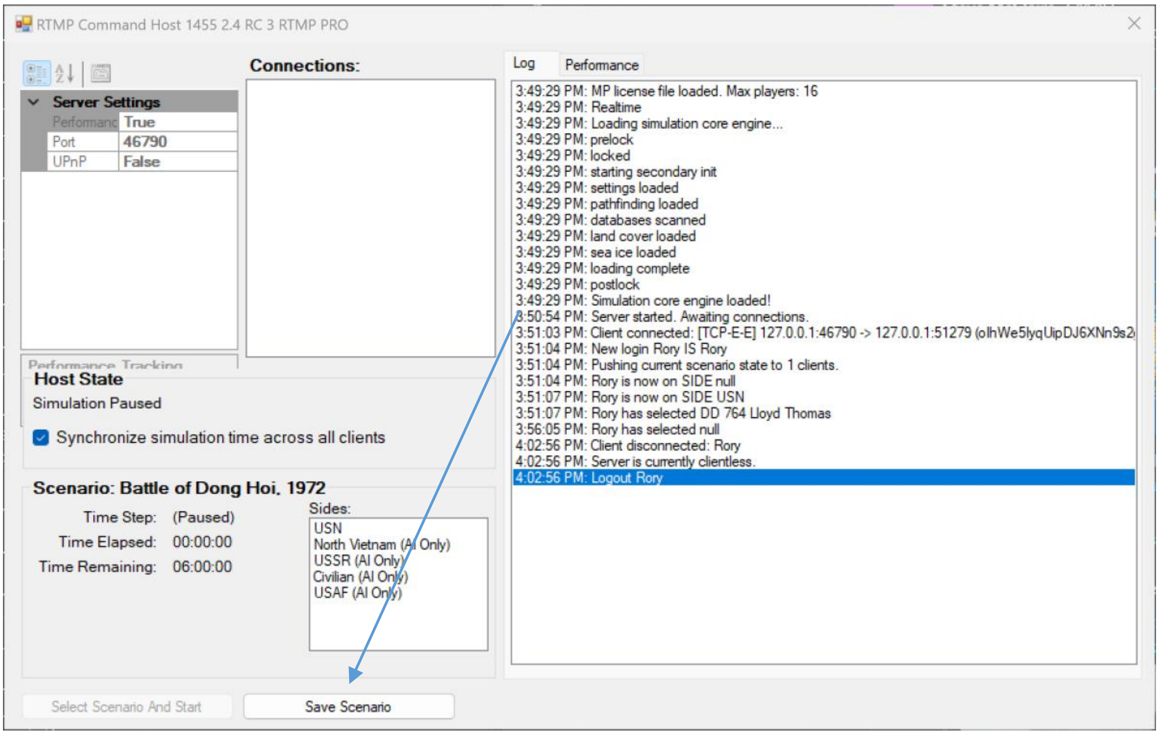
Simtime 1972 04 19 08 00 00.scen
Realtime 2024 10 08 05 02 28.scen
Realtime 2024 10 08 05 01 13.scen
Realtime 2024 10 08 04 59 57.scen
Realtime 2024 10 08 04 58 42.scen

Players

Any player, umpire or observer at any time can trigger an autosave at any time. They can optionally enter a tag to label the autosave, and hit the Save button. The server will then make an autosave, with that optional tag, and place it in the server autosaves directory.

Also, note, the number of autosaves and the time since last autosave are also displayed here.

Server Save Dialog


RTMP Command Host 1455 2.4 RC 3 RTMP PRO
Log
Performance
Performance Tracking
Select Scenario And Start
Save Scenario

Server Settings

Performanc	True
Port	46790
UPnP	False

Connections:

Host State

Simulation Paused

☑ Synchronize simulation time across all clients

Scenario: Battle of Dong Hoi, 1972

Time Step:	(Paused)
Time Elapsed:	00:00:00
Time Remaining:	06:00:00

Sides:

USN
North Vietnam (AI Only)
USSR (AI Only)
Civilian (AI Only)
USAF (AI Only)

```
3:49:29 PM: MP license file loaded. Max players: 16
3:49:29 PM: Realtime
3:49:29 PM: Loading simulation core engine...
3:49:29 PM: prelock
3:49:29 PM: locked
3:49:29 PM: starting secondary init
3:49:29 PM: settings loaded
3:49:29 PM: pathfinding loaded
3:49:29 PM: databases scanned
3:49:29 PM: land cover loaded
3:49:29 PM: sea ice loaded
3:49:29 PM: loading complete
3:49:29 PM: postlock
3:49:29 PM: Simulation core engine loaded!
3:50:54 PM: Server started. Awaiting connections.
3:51:03 PM: Client connected: [TCP-E-E] 127.0.0.1:46790 -> 127.0.0.1:51279 (olhWe5lyqUipDJ6XNn9s2
3:51:04 PM: New login Rory IS Rory
3:51:04 PM: Pushing current scenario state to 1 clients.
3:51:04 PM: Rory is now on SIDE null
3:51:07 PM: Rory is now on SIDE USN
3:51:07 PM: Rory has selected DD 764 Lloyd Thomas
3:56:05 PM: Rory has selected null
4:02:56 PM: Client disconnected: Rory
4:02:56 PM: Server is currently clientless.
4:02:56 PM: Logout Rory
```

This button here opens the Save Scenario interface.

Absolute Control Save Method

A final way, and a very rare way, to save the scenario is for a player to take Absolute Control and go to File->Save. This might be useful if you want a copy on your local computer.

Loading Games

Command RTMP allows you to change scenario in the middle of a multiplayer game, without players needing to reconnect.

First, read the section on Autosave and Rewind, as there is an easy built in way to revert the game state to an earlier game state.

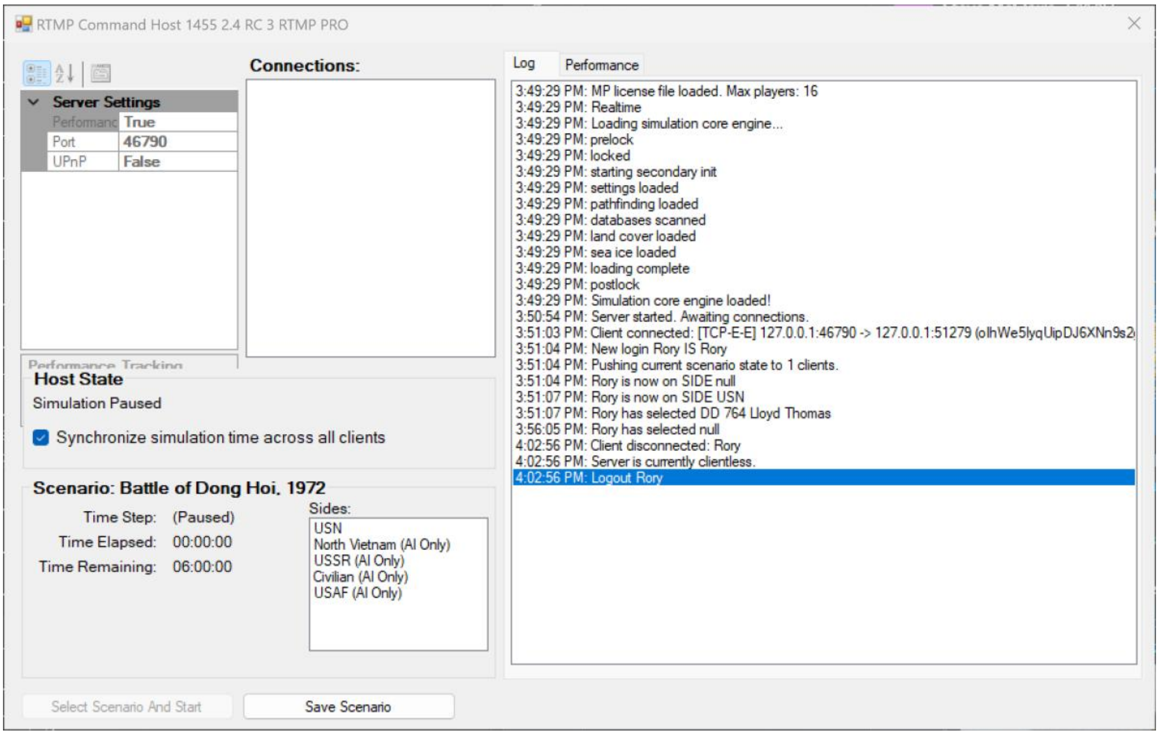
But, lets say you had a server full of 16 connected players, and want to change the scenario to some other, completely different scenario, that uses a different DB. You can!

Take Absolute Control on one of the connected clients. Go to File->Load on the main Command menu. Load your new scenario. Release Absolute Control. Boom! Everyone is now playing the new scenario!

A couple of things: the new scenario's database must be present on all the computers ahead of time, including the server.

And, if you want players to stay locked to the right side, the side names must be the same. For example, if you have 8 players locked to Blue, and 8 players locked to Red using the LockedSide mechanic, then, the new scenario needs to have sides named "Blue" and "Red" to maintain the locked side.

Handling Disconnections


RTMP Command Host 1455 2.4 RC 3 RTMP PRO
Connections:
Log
Performance
Performance Tracking
Select Scenario And Start
Save Scenario

Server Settings

Performanc	True
Port	46790
UPnP	False

Host State

Simulation Paused

☑ Synchronize simulation time across all clients

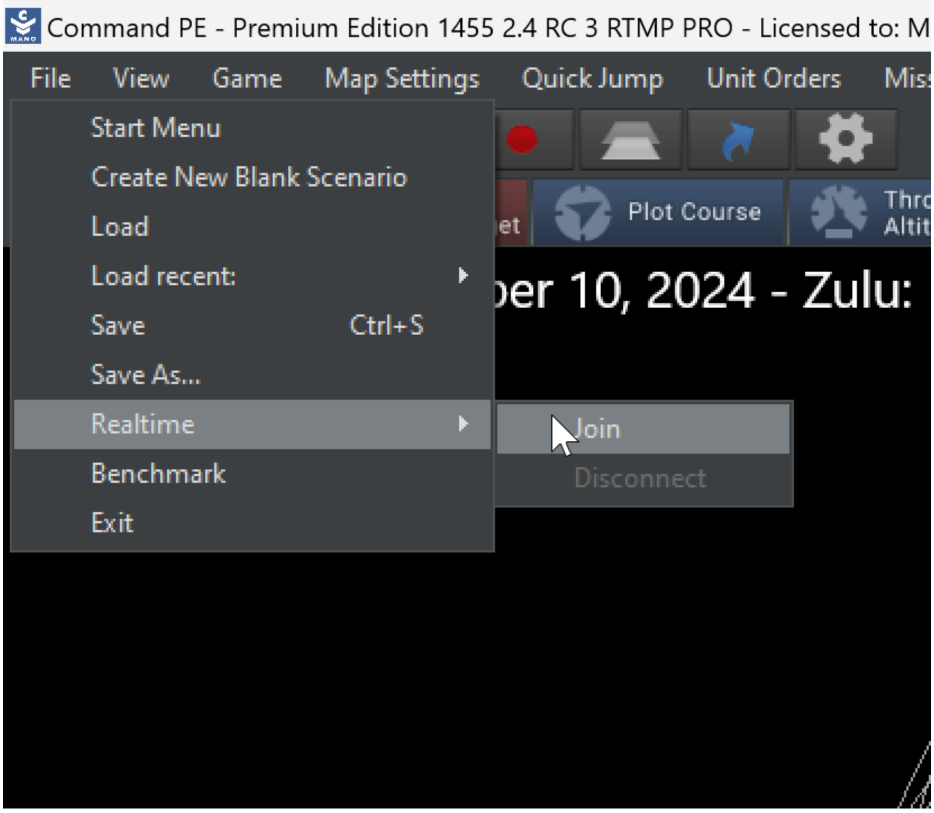
Scenario: Battle of Dong Hoi, 1972

Time Step: (Paused)
Time Elapsed: 00:00:00
Time Remaining: 06:00:00

Sides:
USN
North Vietnam (AI Only)
USSR (AI Only)
Civilian (AI Only)
USAF (AI Only)

```
3:49:29 PM: MP license file loaded. Max players: 16
3:49:29 PM: Realtime
3:49:29 PM: Loading simulation core engine...
3:49:29 PM: prelock
3:49:29 PM: locked
3:49:29 PM: starting secondary init
3:49:29 PM: settings loaded
3:49:29 PM: pathfinding loaded
3:49:29 PM: databases scanned
3:49:29 PM: land cover loaded
3:49:29 PM: sea ice loaded
3:49:29 PM: loading complete
3:49:29 PM: postlock
3:49:29 PM: Simulation core engine loaded!
3:50:54 PM: Server started. Awaiting connections.
3:51:03 PM: Client connected: [TCP-E-E] 127.0.0.1:46790 -> 127.0.0.1:51279 (olhWe5lyqUipDJ6XNn9s2
3:51:04 PM: New login Rory IS Rory
3:51:04 PM: Pushing current scenario state to 1 clients.
3:51:04 PM: Rory is now on SIDE null
3:51:07 PM: Rory is now on SIDE USN
3:51:07 PM: Rory has selected DD 764 Lloyd Thomas
3:56:05 PM: Rory has selected null
4:02:56 PM: Client disconnected: Rory
4:02:56 PM: Server is currently clientless.
4:02:56 PM: Logout Rory
```

The server will automatically pause the game when all clients disconnect. Players can join a game in progress at any time.



When connected to a RTMP game, there are two methods to disconnect. First is to go to File -> Realtime -> Disconnect. And the other is simply to close Command.exe on the client.

Performance Optimization

To maximize performance on the server, do not run anything else of note on the server, including a Command.exe client. Turn Performance Tracking off on the server ahead of time, and set Autosave interval on the server to 0. Do not run any application that uses a lot of networking on the server (eg, bittorrent client, large file transfers over the network, etc).

Autosave and Rewind Feature

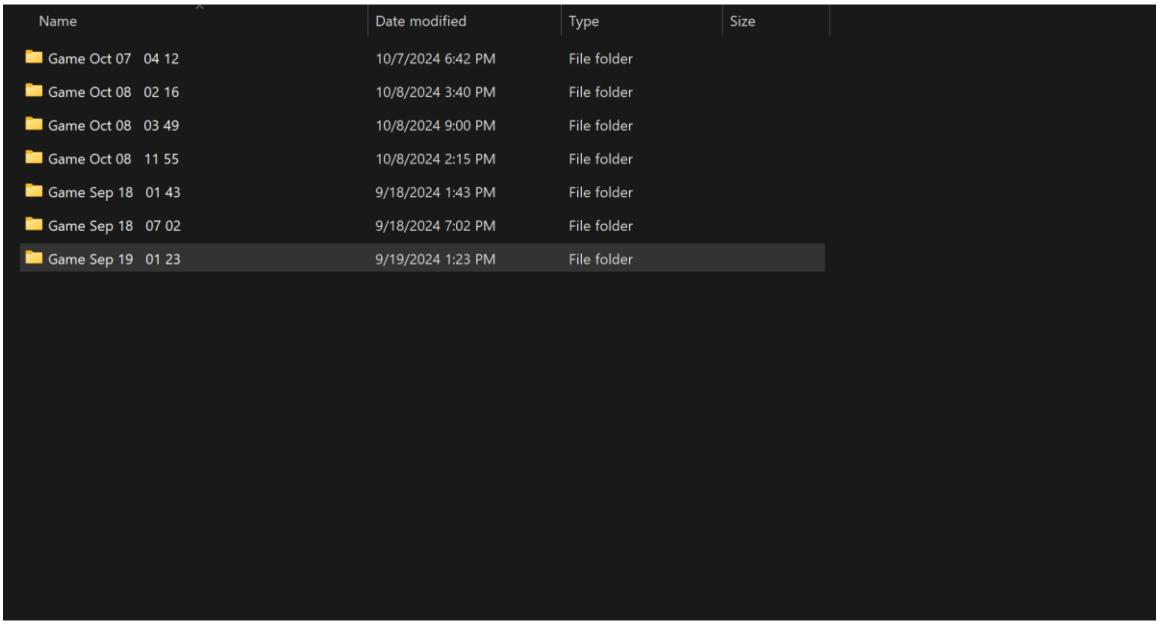
Command RTMP has an autosave feature, and the ability to rewind multiplayer games to a previous autosave.

Command RTMP automatically saves autosaves periodically, the autosave interval is set before starting the server.

When Command RTMP autosaves, it saves TWO copies of the same scenario at a moment in time. This is because the time in the simulation and the time in the real world are different.

“Realtime 2024 10 08   03 36 52.scen” and “Simtime 1950 07 02   09 15 00.scen” are a pair of autosaves, that are exactly the same save. The Realtime autosave is labelled with the real time (the time on the wall clock in reality) and the Simtime autosave is labelled with the in-sim time (the time in Command.exe).

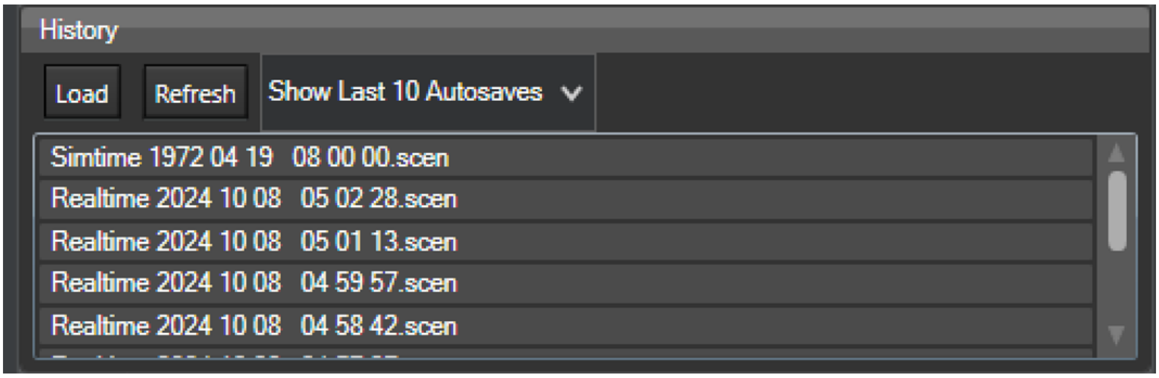
Accessing and Loading Autosaves



Name	Date modified	Type	Size
Game Oct 07 04 12	10/7/2024 6:42 PM	File folder	
Game Oct 08 02 16	10/8/2024 3:40 PM	File folder	
Game Oct 08 03 49	10/8/2024 9:00 PM	File folder	
Game Oct 08 11 55	10/8/2024 2:15 PM	File folder	
Game Sep 18 01 43	9/18/2024 1:43 PM	File folder	
Game Sep 18 07 02	9/18/2024 7:02 PM	File folder	
Game Sep 19 01 23	9/19/2024 1:23 PM	File folder	

The location for automatically saved games is on the **server**'s Scenarios directory, under the sub-directory Realtime (by default, C:\ProgramData\Command Professional Edition 2\Scenarios\Realtime). Each RTMP **server** session is labeled with the date, and the hour and minute of server startup.

Using the Rewind Feature



Simtime 1972 04 19 08 00 00.scen
Realtime 2024 10 08 05 02 28.scen
Realtime 2024 10 08 05 01 13.scen
Realtime 2024 10 08 04 59 57.scen
Realtime 2024 10 08 04 58 42.scen

In the RTMP Ribbon there is a History element. Press "Refresh" to see the latest autosaves. Select one and press Load to load it. Remember to hit refresh to see the latest autosaves.

Absolute Control

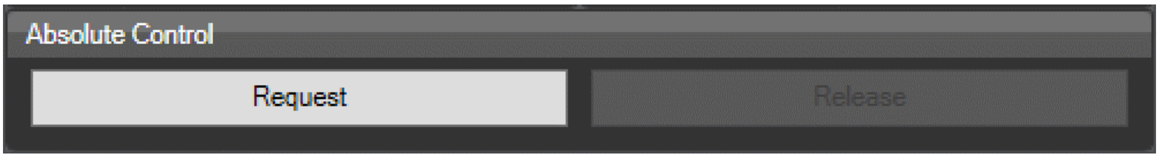
Absolute Control (AC) mode is a special mode in RTMP. It pauses the server, the server saves the scenario to disk, sends the scenario over to the player who requested Absolute Control and that player's computer opens the scenario as though it was normal singleplayer Command.

When that player releases Absolute Control, his computer saves the scenario that he has open to disk and sends it to the server. The server opens this new scenario and the server is now running that new scenario.

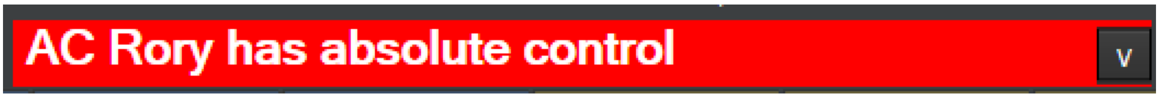
This allows for total control over the scenario, much like in single player. Add/remove sides, edit anything in the scenario, add/remove lua scripts, events, etc.

Activating Absolute Control

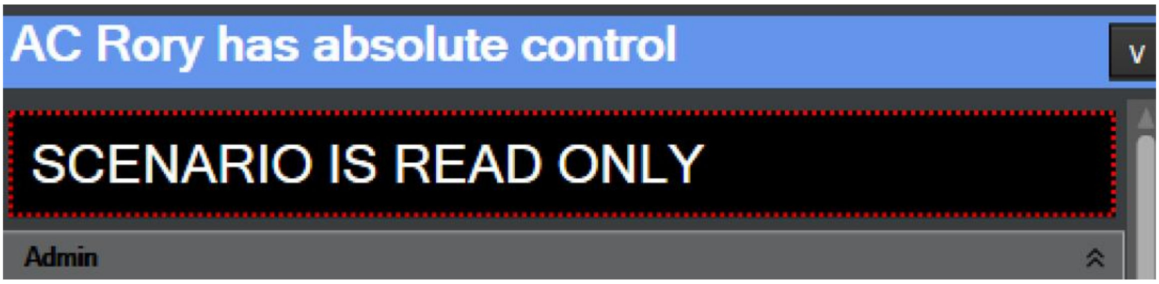
To activate Absolute Control, press this Request button on the RTMP ribbon.



Server will then grant AC access. The player with AC will see this RED banner on their RTMP Ribbon.


AC Rory has absolute control

Other players will see this BLUE banner.


AC Rory has absolute control

SCENARIO IS READ ONLY

When a player has AC, for the rest of the players the scenario is read only. They can view the scenario, but cannot make changes to anything in the game state. (They can seem to make changes, but when AC is released, all their changes will be overwritten).

COMMAND-LINE INTERFACE

(Available in PE Premium)

Starting from version v1.15, a command-line version of CPE is available for use to help automate your analysis workflow.

The CLI version differs in a number of ways compared to the normal graphics user interface (GUI)-driven CPE application. Because it lacks the overhead of the various GUI elements, it runs faster and more efficiently for a given scenario. Because of this, it is better suited for massive parallelized analysis to fully utilize the CPU & RAM resources of modern computers. This ability is further enhanced by its command-line nature: While it is certainly possible to run multiple instances of CPE in parallel, spinning them up is a manual process. In contrast, multiple CLI instances can be orchestrated by launching them from another application or simply by use of a batch file.

To use the CLI, you have to run the program CommandCLI.exe, located in the CPE installation folder (same location as the main Command.exe program). **The CLI version currently supports Monte-Carlo mode analysis (not interactive).**

Command-Line Arguments

To configure its behavior, the CLI version uses a number of command-line arguments.

There are two required command line arguments: **-mode** and -**scenfile**.

-scenfile [e.g. "path"] : Provides CommandCLI.exe with the scenario that you desire to run. Using the quotes is technically not mandatory, but necessary when the scenario file's total path includes space breaks *(e.g. "C:\Users\User1\My Scenarios\Scenario1.scen").*

-mode [MC or I] : Sets CommandCLI.exe into Monte Carlo (MC) or interactive mode.

If set to MC mode, the simulation will load the desired scenario and immediately start running it as fast as possible without waiting for any interaction.

If set to interactive mode, the simulation will load the desired scenario but in a paused state, open a TCP socket port, and wait for instructions by the user through the TCP socket, using the Lua API. Several key Lua methods for manipulating scenario execution are:

- **VP_RunSimulation:** Un-pause the scenario and resume running it as fast as possible
- **VP_PauseSimulation:** Pause the currently running scenario
- **VP_RunToTimeAndHalt** . This allows you to specify that the scenario will pause when it reaches the provided date and/or time (in DD:MM:YYYY / HH:MM:SS format). Usage examples:

  VP_RunToTimeAndHalt({Date='07:02:2023'})
  VP_RunToTimeAndHalt({Time='22:46:23'})
  VP_RunToTimeAndHalt({Date='07:02:2023', Time='22:46:23'})

- **VP_RunForTimeAndHalt** . Similar to the above, this allows you to specify that the scenario will run for X-time (in HH:MM:SS format) and then pause. You can specify a timespan larger than a day by using a suitable hours figure (e.g. 72 hrs. for a 3-day span). Usage example:

VP_RunForTimeAndHalt({Time='28:12:23'})

There are also several optional command line arguments: **-autoexit**, **-outputfolder**, **-iterations, -savemessagelog** and **-recinterval**.

-autoexit [no argument necessary]: By default, when CommandCLI.exe finishes its analysis run or suffers an error of any kind, it will wait for the user to press any key to exit. Using the -autoexit switch will instead cause the program to automatically exit.

-outputfolder [e.g. "path"] : By default, CommandCLI.exe outputs into the Analysis_MC directory in your Command directory. You can override that behavior with the **-outputfolder** command line argument.

-iterations [e.g. 123] : (Applies to MC mode only). By default, CommandCLI.exe will run the provided scenario once. Changing the **-iterations** command line argument allows for more than one simulation to occur in sequence.

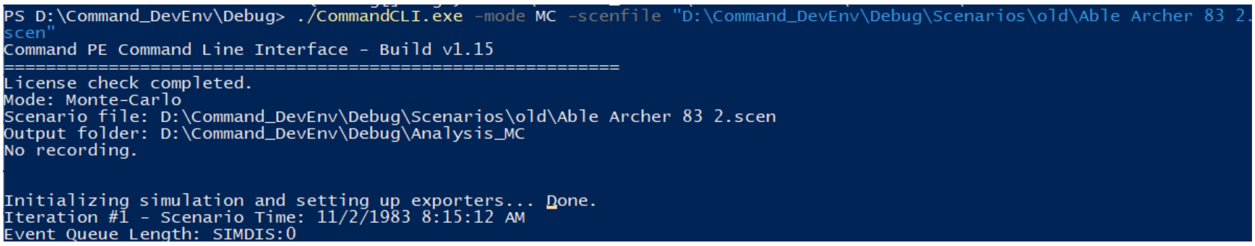
-savemessagelog [no argument necessary]: If added, each iteration run will also output the consolidated message log, and write it on the per-iteration output path.

-recinterval [e.g. 123] : By default, CommandCLI.exe will not use the recorder feature to allow the run to be replayed. Changing the **-recinterval** argument allows for the interval between recorder snapshots to be set. The number provided is the number of seconds between recorder snapshots.

This is an example of a complete call to run an automated analysis:

CommandCLI.exe -mode mc -scenfile "D:\Command\Scenarios\Desert Storm 1991 - Strategic air campaign - Instant Thunder.scen" -iterations 3 -outputfolder "D:\Analysis1" –recinterval 15

CommandCLI can be run via PowerShell, or in the traditional Windows command interpreter (cmd.exe):



```
PS D:\Command_DevEnv\Debug> ./CommandCLI.exe -mode MC -scenfile "D:\Command_DevEnv\Debug\Scenarios\old\Able Archer 83 2.
scen"
Command PE Command Line Interface - Build v1.15
============================================================
License check completed.
Mode: Monte-Carlo
Scenario file: D:\Command_DevEnv\Debug\Scenarios\old\Able Archer 83 2.scen
Output folder: D:\Command_DevEnv\Debug\Analysis_MC
No recording.


Initializing simulation and setting up exporters... Done.
Iteration #1 - Scenario Time: 11/2/1983 8:15:12 AM
Event Queue Length: SIMDIS:0
```

Unlike setting up Monte Carlo in Command.exe, configuration of event export for CommandCLI.exe uses the EventExport.ini file. Please refer to the documentation for Analysis and Data Export for information on how to set up EventExport.ini to capture the information essential to your analysis.

Arguably one of the most useful features of the CLI version is the easy setup for parallel execution. This is an example of how a batch file can be used to spawn three analysis jobs in parallel:

start "Job1" CommandCLI.exe -mode mc -scenfile "D:\Command\Scenarios\Desert Storm 1991 - Strategic air campaign - Instant Thunder.scen" -iterations 3 -outputfolder "D:\Results_Job1" –recinterval 15

```
start “Job2” CommandCLI.exe -mode mc -scenfile "D:\Command\Scenarios\Korea 2017.scen" -iterations 3 -outputfolder “D:\Results_Job2” –recinterval 5
```

```
start “Job3” CommandCLI.exe -mode mc -scenfile "D:\Command\Scenarios\MarshmallowMan.scen" -iterations 3 -outputfolder “D:\Results_Job3” –recinterval 60
```

GENERAL SYSTEM & INFRASTRUCTURE

Additional configuration settings

CPE uses additional configuration settings, which are stored in supplemental configuration files. All configuration files are stored on the **[top-level writable folder]\Config** folder.

Command.ini contains all settings that are applicable to both the commercial and professional/academic editions of CMO.

CPE.ini handles general simulation settings as well as Lua-specific settings.

EventExport.ini stores the settings for event export (see section “Interactive analysis & event export”).

ExternalURLs.ini stores the addresses of external on-demand information providers (tutorial videos, mapping tiles, DB-viewer images etc.). Having these addresses on a separate file outside of the program code makes it far simpler to adjust these links if any of the external sources goes extinct or new alternative sources become available.

SimConnect.ini is responsible for the setting of various remote-entity / distributed-simulation connectors (initially DIS, but also expandable to other types).

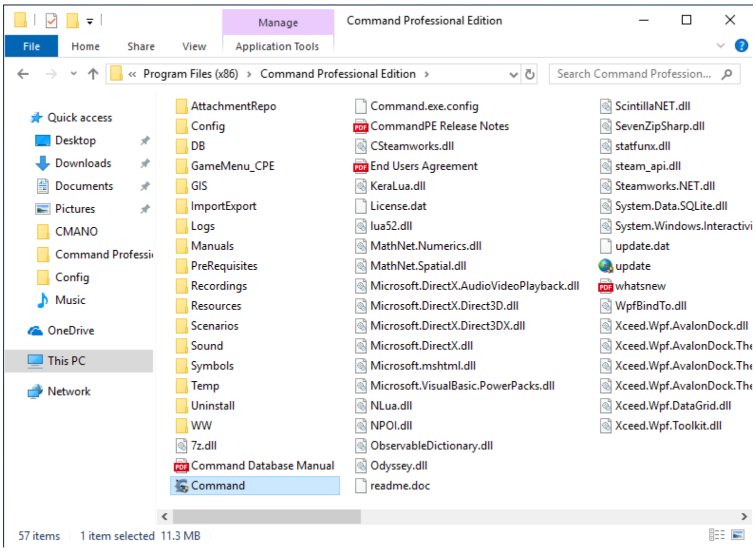
Overrides.ini governs settings for the configurable Mechanics Overrides available in CPE.

Running CPE in high-DPI (4K / UHD) monitors

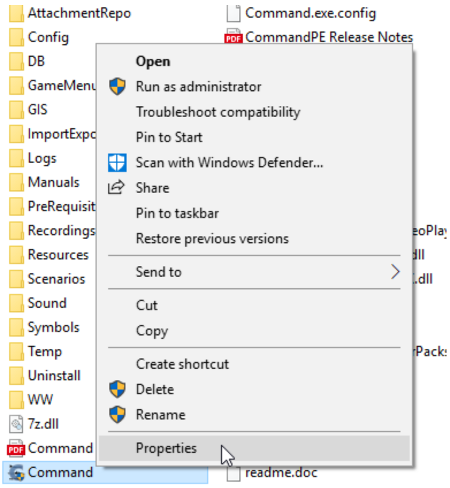
Command PE's user interface is optimized for execution in a 720p (1280x720) or 1080p (1920x1080) screen format, in "standard DPI" desktop settings. As monitors with higher resolution proliferate, CPE's UI code has been continuously adjusted to also function in higher resolution. However, depending on the graphics card used by your system, and particularly in high-DPI desktop settings (https://en.wikipedia.org/wiki/Dots_per_inch#How_Microsoft_Windows_handles_DPI_scaling), some issues may manifest (out-of-memory errors, UI elements misadjusted etc.).

To prevent such issues, a simple manual adjustment is necessary. *(The steps described below are for Windows 10, on earlier versions of Windows the procedure is effectively the same but the steps may be slightly different in execution)*

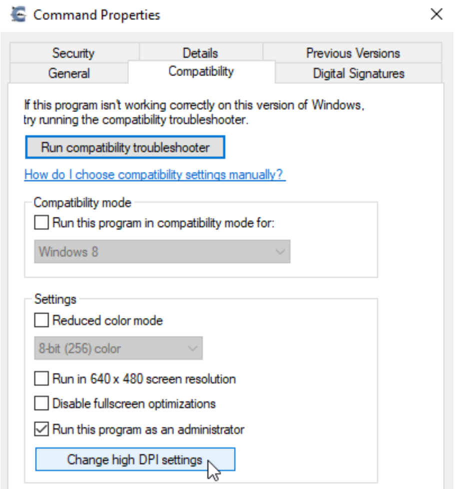
First, using Windows Explorer, navigate to the Command PE installation folder:



Right-click on the Command executable (Command.exe) and select "Properties":



In the "Compatibility" tab, select "Change high DPI settings":



If this program isn't working correctly on this version of Windows, try running the compatibility troubleshooter.

Compatibility mode

☐ Run this program in compatibility mode for:

Windows 8

Settings

☐ Reduced color mode

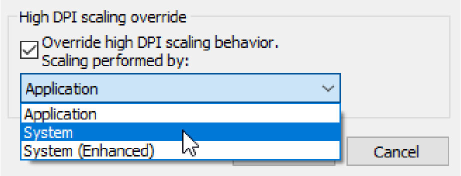
8-bit (256) color

☐ Run in 640 x 480 screen resolution

☐ Disable fullscreen optimizations

☑ Run this program as an administrator

Check the “Override high DPI scaling behavior” checkbox and from the dropdown list select “System”:



High DPI scaling override

☑ Override high DPI scaling behavior.
Scaling performed by:

Application

After this adjustment, Command-PE should run on your high-DPI desktop without any problems. Please contact the tech support personnel with any issues you may encounter in this aspect.

PRO-EXCLUSIVE SIMULATION & UI FEATURES

Communications jamming & disruption

(Available in PE & AE)

The commercial release v1.12 of CMANO introduced rich capabilities for modelling communications disruption, described in summary here: http://www.warfaresims.com/?p=4454

One addition reserved exclusively for Command-PE is the inclusion of integrated communications jamming, without need for any scripting. The model used for the various factors affecting comms-jamming is simpler than the equivalent one used for radar jamming, however it is structured to be extendable/replaceable for interested customers via mechanics overrides (see section "Mechanics Overrides").

The check for comms jamming occurs every 15 seconds of simulated time. Once a unit has all its available communications links effectively jammed, it is removed from its parent side's communications grid and suffers the same "isolation" effects as those in the commercial version of CMO. The unit will remain off-grid as long as its communications remain jammed, and may rejoin its parent grid only if any of its comm devices successfully resists jamming (or the suppression ceases). There are special cases where a unit, while remaining effectively jammed, may nevertheless be able to rejoin its parent comms grid, such as when an aircraft returns to a base or carrier which itself is on-grid.

Some ready-to-use examples of a communications-jamming platform in the Command database (DB3000) are:

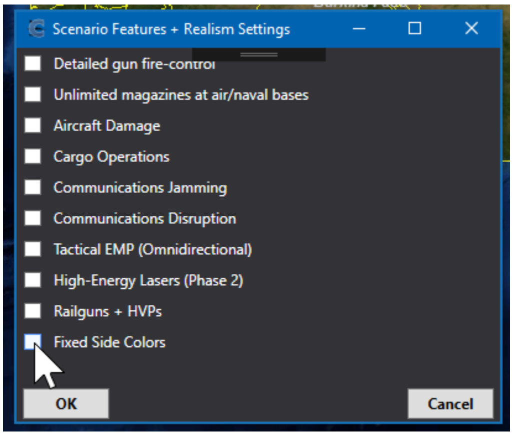
- Aircraft #2316 - EA-6B Prowler ICAP II Blk 89 -- United States (Marine Corps), 1999 – using the USQ-113(V)2 comms intercept & jamming suite
- Aircraft #1091 - EA-18G Growler -- United States (Navy), 2011 – using the improved USQ-113(V)3 system

Fixed Side Colors

By default, Command PE (like its commercial counterpart) uses the standard NTDS/APP-6 color codes for the various unit & contact symbols: blue for friendly, red for hostile, yellow for unknown etc. These colors are based on postures per side. So for example when viewing Side-A's operational picture, Side-A's units are shown in blue; when viewing Side-B's equivalent picture, Side-B's units are shown in blue, etc.

However, in traditional board-based wargames & exercises, by practical necessity, the assigned side colors are fixed: Side-A is constantly blue, for example, and Side-B is constantly red, regardless of which side's viewpoint the player occupies. This convention is frequently carried over to electronic versions of these exercises and may be required in a gaming session that involves CPE.

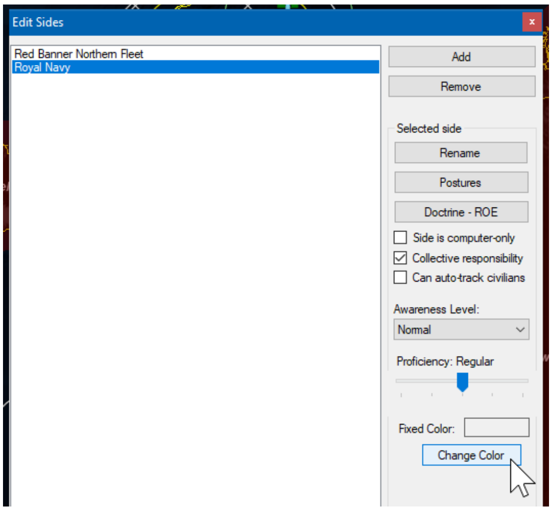
To facilitate this requirement, Command PE allows assigning an arbitrary fixed color per side. This ability is available as a per-scenario optional feature:



Scenario Features + Realism Settings

□ Detailed gun fire-control
□ Unlimited magazines at air/naval bases
□ Aircraft Damage
□ Cargo Operations
□ Communications Jamming
□ Communications Disruption
□ Tactical EMP (Omnidirectional)
□ High-Energy Lasers (Phase 2)
□ Railguns + HVPs
□ Fixed Side Colors

Before being able to enable this feature, it is first necessary to assign a fixed color to each of the sides present in a scenario. This is possible through the "Sides" window interface:



Edit Sides

Red Banner Northern Fleet
Royal Navy

□ Side is computer-only
☑ Collective responsibility
□ Can auto-track civilians

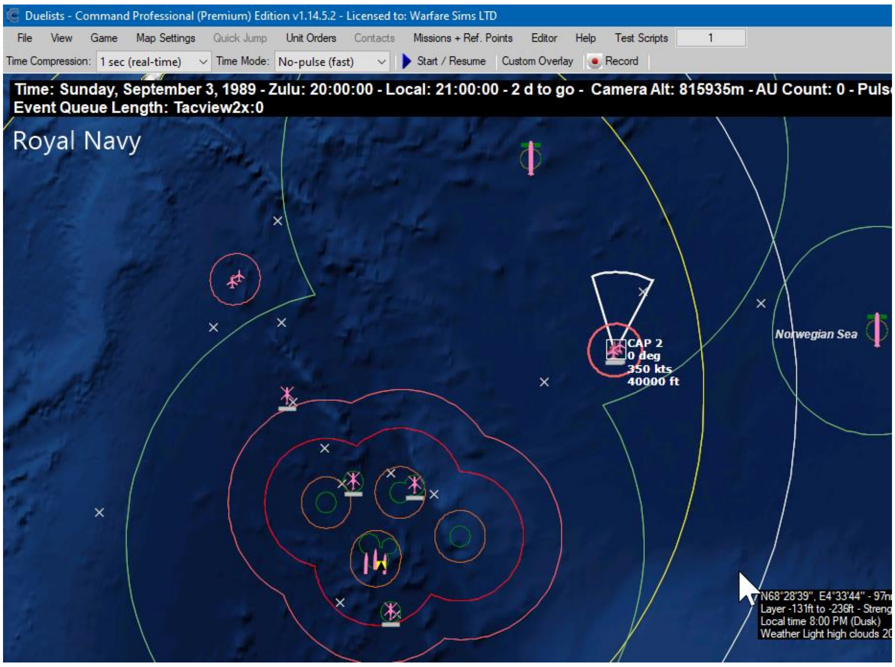
Awareness Level: Normal

Proficiency: Regular

Fixed Color:

Once each side has been assigned a fixed color, the feature can be enabled for the scenario.

This is an example of an own side having been assigned a pink hue as a fixed color:



Duelists - Command Professional (Premium) Edition v1.14.5.2 - Licensed to: Warfare Sims LTD

Time Compression: 1 sec (real-time) Time Mode: No-pulse (fast)

Time: Sunday, September 3, 1989 - Zulu: 20:00:00 - Local: 21:00:00 - 2 d to go - Camera Alt: 815935m - AU Count: 0 - Puls
Event Queue Length: Tacview2x:0

Royal Navy

N68°28'39", E4°33'44" - 97n
Layer -131ft to -236ft - Streng
Local time 8:00 PM (Dusk)
Weather Light high clouds 20

Because of the usage of fixed colors, it can be more difficult to keep track of which side's viewpoint the player is observing. For this reason, while this feature is active, the current side's name is permanently displayed under the status caption bars as a reminder (in the above screenshot, "Royal Navy").

The Amphibious Planner, Operation Planner & Serial Editor

The amphibious and operation planner, together with the serial editor, form a set of common-oriented tools that allow you to orchestrate complex, multi-layered operations (including, but not limited to comprehensive amphibious landings) and execute the different phases of an operation at different time points or based on customized conditions.

For a detailed walkthrough on using these powerful tools to their full effect, see Appendix 3 at the end of this manual.

Multiple Coordinate System Options

Command PE can display global locations using a number of different coordinate systems. The user can select from the following coordinate systems in the "Game Options" window:

- Degrees
- Signed Degree
- Military Grid Reference System (MGRS)
- Universal Trans Mercator (UTM)
- Earth-Centered, Earth-Fixed (ECEF)

Export Losses and Expenditures

Command PE can export a formatted list of all losses and expenditures for the current scenario to a formatted text file. To export the current state of scenario losses and expenditures, open the 'Losses & Expenditures' window from the Game menu and click on the 'Export' button.

The 'CSV' option will create a comma delimited (by default) text file with the following fields. The separator character can be changed to a character other than a comma, if desired. The CSV file will contain the following fields:

<Entry Type>,<ID>,<Quantity>,<Name>

Entry type will be either 'Loss', 'Expenditure', or 'Weapon Loss.' A 'Loss' indicates a unit type, while an 'Expenditure' or 'Weapon Loss' identifies a weapon type.

"ID" indicates the type and database ID of a unit loss (i.e. Aircraft_1234) or the database ID of a weapon loss or expenditure.

"Quantity" indicates the number of losses or expenditures.

“Name” is the unit or weapon name.

The ‘XML’ option will create a formatted XML file with identifying fields for each type of unit lost, weapon expended, and weapon lost in the scenario. Each entry will include the type (‘Unit’ or ‘Weapon’) plus an ID, Quantity, and Name field (formatted as above) for each type of loss or expenditure.

Custom Unit Icons

Beginning from v2.3, the map symbol Icon of individual units can now be modified during runtime, on a persistent level, and through multiple scenario runs.

Custom icons supports:

- Individual units defined by their GUID which is a unique instance of a unit. (ex : one icon for F-15 #1, a different icon for F-15 #2 etc.)
- Individual database ID (DBID) which represent a main entry in the database. (ex : one icon for F-15, another icon for F-16 etc.)
- A platform category (ex : aircraft)

Given the situation, you may want to either change a specific unit’s icon, even temporarily but have simultaneously an icon override for its category. For example, you could have all F-15’s icons changed, but have one specific F-15 unit with its own icon.

This is supported according to an order of priority when performing icon override. Command will override the icons given a logical order. The most global override (platform category) will always be overridden by the most specific one, if the latter exists.

A summary of the precedence rules:

- If a platform category icon overrides exists it will replace the native icon.
- If a DBID (database specific) override exists it will replace all the above.
- If a Guid (unit specific) override exists it will replace all the above.

Category and DBID icon override

Getting started

The custom icon folder and association files template are automatically created, if you have just acquired this new version, you will need to launch Command beforehand for the files and folder to be generated.

Platform categories are currently defined as:

- ship
- aircraft

- vehicle
- facility
- submarine
- weapon
- satellite
- personnel

If you are uncertain about a units platform check the in-game **DBViewer** or the database entry itself.

For example, some "units", such as whales are classified as submarines.

Creating an entry

To create the override, open either **Associations_CWDB.csv** or **Associations_DB3000.csv** in **Symbols\Custom**, depending on which database you have chosen to work upon.

The syntax to create an override is as follows :

#TYPE;DBID;FILENAME;SIZE(optional);ORIENTATION(optional)

Note that the last 2 arguments are optional, this means that both can be ignored when adding an entry.

- **Type** is the database categories as listed in **"platform categories".**
- **DBID** is the actual database entry ID which is unique **within** a unit category. Putting * in place of the DBID designates the entire category to be overridden by this icon.
- **Filename** is the icon's filename, with its extension, which is located in the **Symbols\Custom.** You should only use squared images, with a white base color (to support dynamic coloring), and the picture pixel size should be as small as possible and ideally multiple of 2, or 20px. Large and very large icons (>32px) should be the exception and used only is the custom size argument is large as well.
- **Size** is the icon's size in pixel as drawn in command, the default size is 20px. This argument is optional.
- **Rot** determines whether or not the icon should rotate according to the unit's bearing. Putting a different value or nothing defines the icon with the default orientation (non-rotatable) This argument is optional.

Each parsible line must start with a # character, all other lines are ignored, making it possible to add comments within the file.

To create an override that applies to all ships you can write :

#ship;*;Ships.png

- The * means "all DBID from the category".
- "Ships.png" being the image you have added in the custom icon folder.

To create an icon override specific to a DBID, the second argument must have the corresponding DBID of a platform. The DBID can be found in the DbViewer within Command, or by browsing the database file.

Example 1 :

#aircraft;2846;Drone.png;40;rot

Replaces aircraft DBID 2846 default icon to Drone.png, but also increases its size (from 20px to 40px)

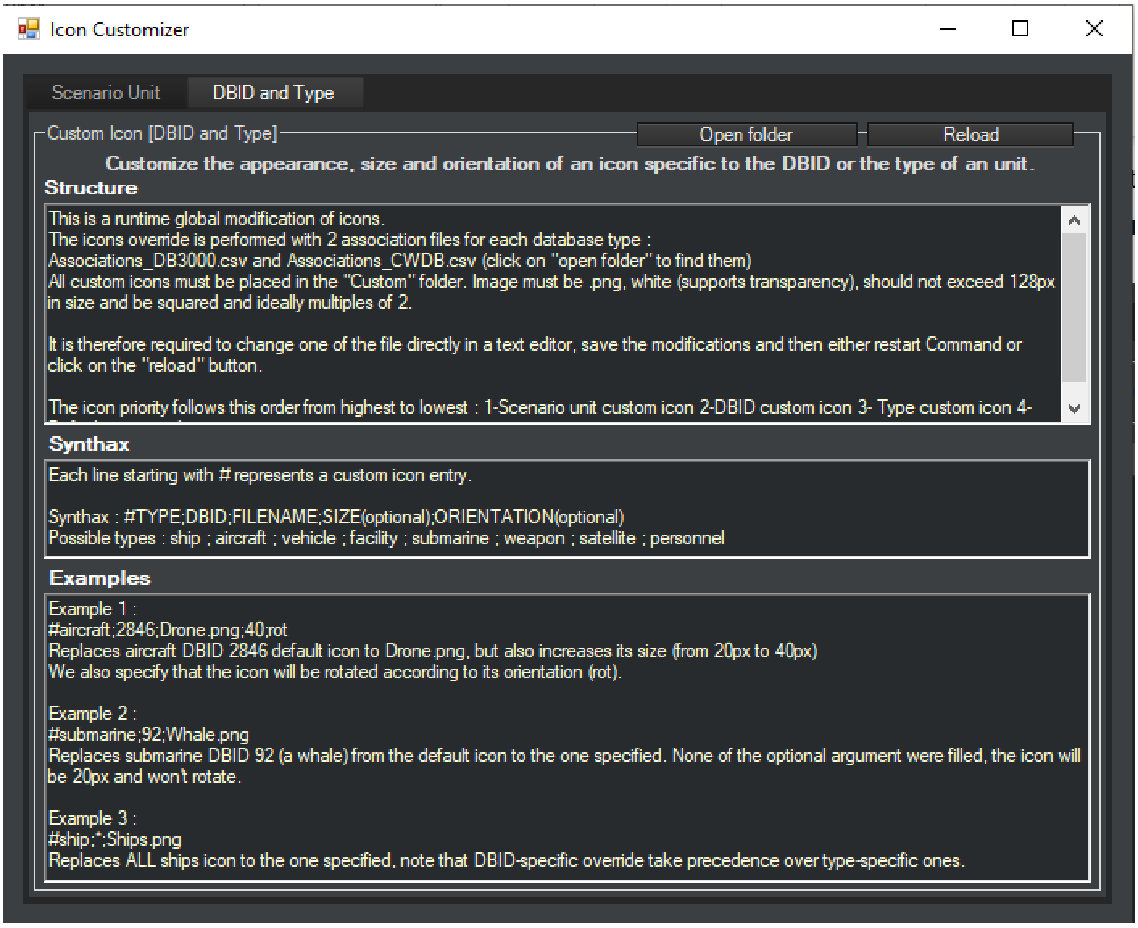
We also specify that the icon will be rotated according to its orientation (rot).

Example 2 :

#submarine;92;Whale.png

Replaces submarine DBID 92 (a whale) from the default icon to the one specified. None of the optional argument were filled, the icon will be 20px and won't rotate.

Instructions are available within Command's UI, this also allows hot reload during runtime, thanks to the "reload" button :



Customize the appearance, size and orientation of an icon specific to the DBID or the type of an unit.

Structure

This is a runtime global modification of icons.
The icons override is performed with 2 association files for each database type :
Associations_DB3000.csv and Associations_CWDB.csv (click on "open folder" to find them)
All custom icons must be placed in the "Custom" folder. Image must be .png, white (supports transparency), should not exceed 128px in size and be squared and ideally multiples of 2.

It is therefore required to change one of the file directly in a text editor, save the modifications and then either restart Command or click on the "reload" button.

The icon priority follows this order from highest to lowest : 1-Scenario unit custom icon 2-DBID custom icon 3- Type custom icon 4-

Synthax

Each line starting with # represents a custom icon entry.

Synthax : #TYPE;DBID;FILENAME;SIZE(optional);ORIENTATION(optional)
Possible types : ship ; aircraft ; vehicle ; facility ; submarine ; weapon ; satellite ; personnel

Examples

Example 1 :
#aircraft;2846;Drone.png;40;rot
Replaces aircraft DBID 2846 default icon to Drone.png, but also increases its size (from 20px to 40px)
We also specify that the icon will be rotated according to its orientation (rot).

Example 2 :
#submarine;92;Whale.png
Replaces submarine DBID 92 (a whale) from the default icon to the one specified. None of the optional argument were filled, the icon will be 20px and won't rotate.

Example 3 :
#ship;*;Ships.png
Replaces ALL ships icon to the one specified, note that DBID-specific override take precedence over type-specific ones.

Scenario-level unit icon override

This low-level override associates an unit's Guid to a custom icon and this information is stored within the save/scenario file. It is not necessary to manually edit any file, since the modification are performed within Command's UI.

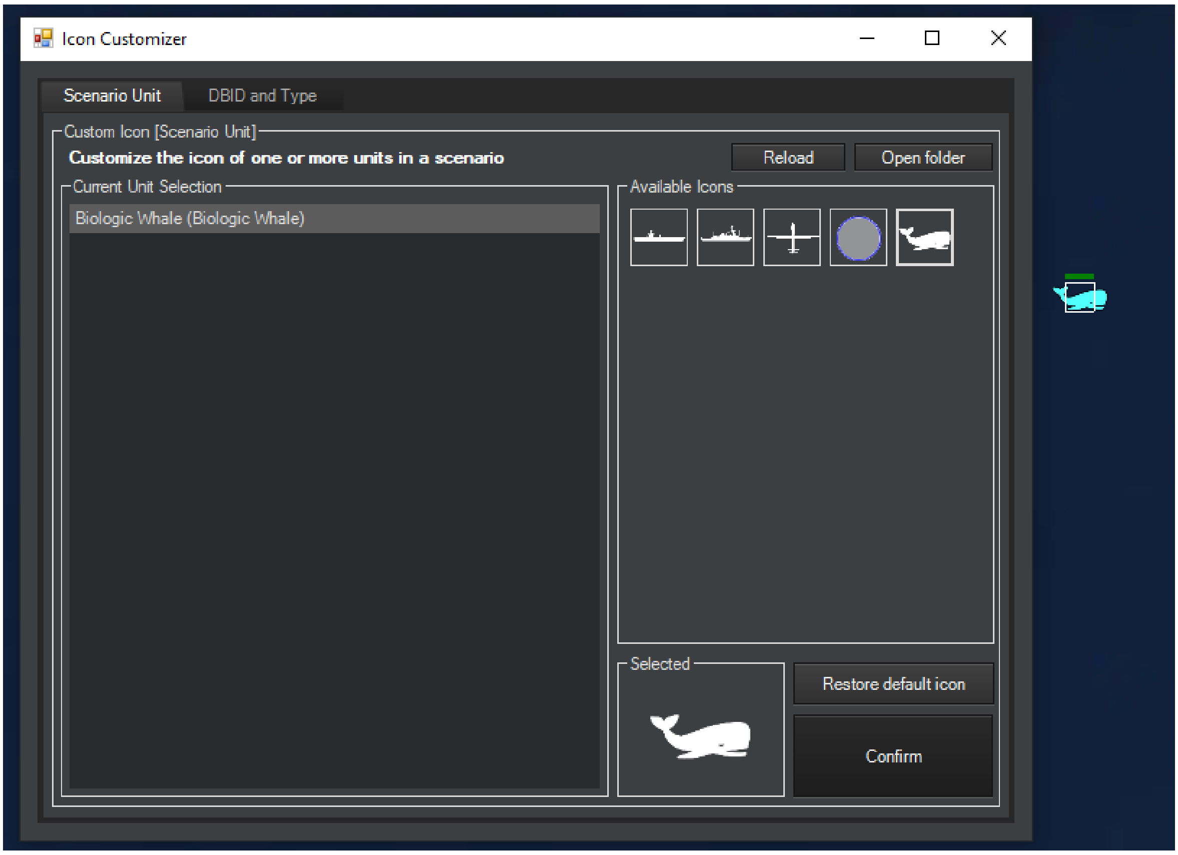
Note that the icon files still need to be present in the right location. If you pass along a scenario to another user and want him to use the custom icon, you must provide him with the custom icons located in **\Symbols\Custom** . If you are uncertain of the location of this folder, run Command, open any scenario, and navigate to the top menu bar **Editor>Customize Icons** and then click on the **"open folder"** button.

Customizing the icons

Select one or more units and go to **[Top bar menu]** Editor>Customize Icons.

This will bring you to an UI listing all icons located in the Custom folder. Note that editing this folder is possible even during runtime, thanks to the "reload" button, which refreshes the available icons.

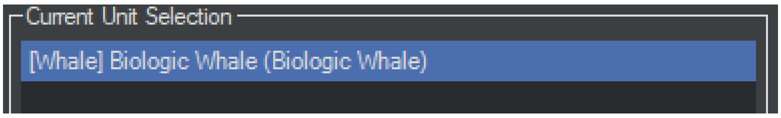
Pick an icon in the displayed list and click on the "confirm" button. The unit's icon will be changed.



Customize the icon of one or more units in a scenario

Biologic Whale (Biologic Whale)

A unit with a customized Icon will have the name of the custom icon in brackets next to its name. The **"Restore default icon"** button will remove the custom icon override for all selected units, even for a unit from a saved scenario.



[Whale] Biologic Whale (Biologic Whale)

Miscellaneous

Pitch Kinematics

Starting from version v1.11.06, air-launched guided weapons follow a much smoother and more realistic trajectory after launch, using true pitch attitude; thus, their horizontal and vertical speeds are adjusted continuously as their orientation towards their target and pitch change. This change means that the projected position-in-time of a weapon, as well as the fuel/energy quantity necessary to achieve the nominal range are calculated differently from the legacy model.

In the commercial version this change is enforced and cannot be disabled. In CPE, to avoid disrupting existing analysis workflows that assume/rely on the legacy model, the improved model is optional (enabled by default) and enabled/disabled through the **CPE.ini** file (see Appendix 1):

UsePitchForWeapons = 1 ← enabled

UsePitchForWeapons = 0 ← disabled

The new model also changes the way that the weapon fuel/energy quantity is calculated (see inset *“Changes to weapon fuel for pitch-enabled weapons”* on the DB editor manual).

DLZ-check override

DLZ calculations (see CMO manual, section “DLZ and why it matters”) are fundamental to weapon launch decisions but can sometimes prevent engagements in which we want to temporarily overlook the kinematic factor.

For this reason, a unit can be instructed to ignore its DLZ calculations prior to firing a weapon *(Unit context menu → Attack Orders → Ignore DLZ)*. This is useful for visualizing a weapon’s trajectory regardless of its DLZ envelope (and validating the DLZ calculations themselves).

ANALYSIS & DATA EXPORT

Interactive analysis & event export

(Available in PE & AE)

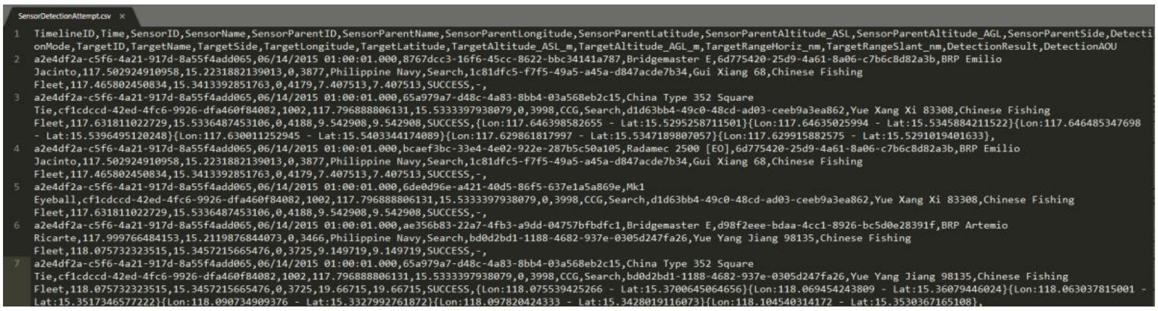
One of the common uses of Command-PE is for analysis of a scenario (or multiple variants of a given scenario with subtle changes) to generate outputs useful for tactical/operational analysis.

Command-PE offers two analysis modes, interactive and non-interactive (aka Monte Carlo mode).

Interactive analysis is very similar to a normal gaming session as on the commercial version of CMO, thus affording the user the flexibility of altering any of the scenario parameters mid-execution at will. The only addition is that, in the background, events of interest are being written to disk (or sent over the network) as configured by the user.

The following destination types for event export are currently supported:

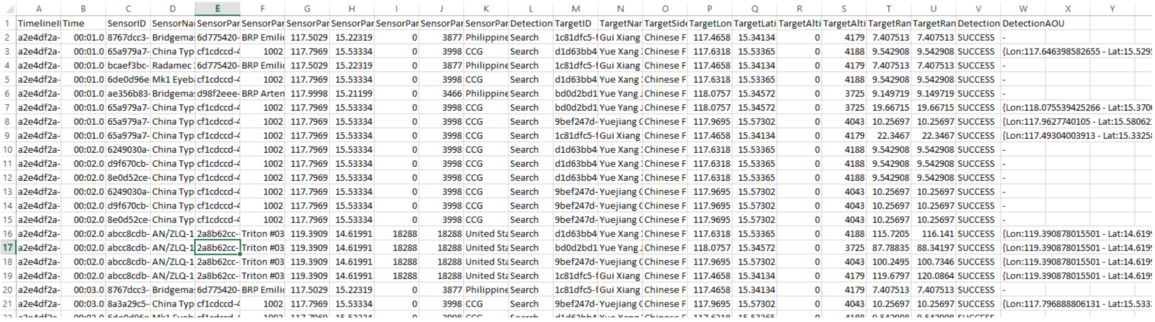
- **CSV format**: Writes events to comma-delimited files, on the path **[top-level writable folder]\Analysis_Int\xxx.csv** (multiple files, one for each event type, and even more if the "SplitFilesBySide" option is enabled)



SensorDetectionAttempt.csv

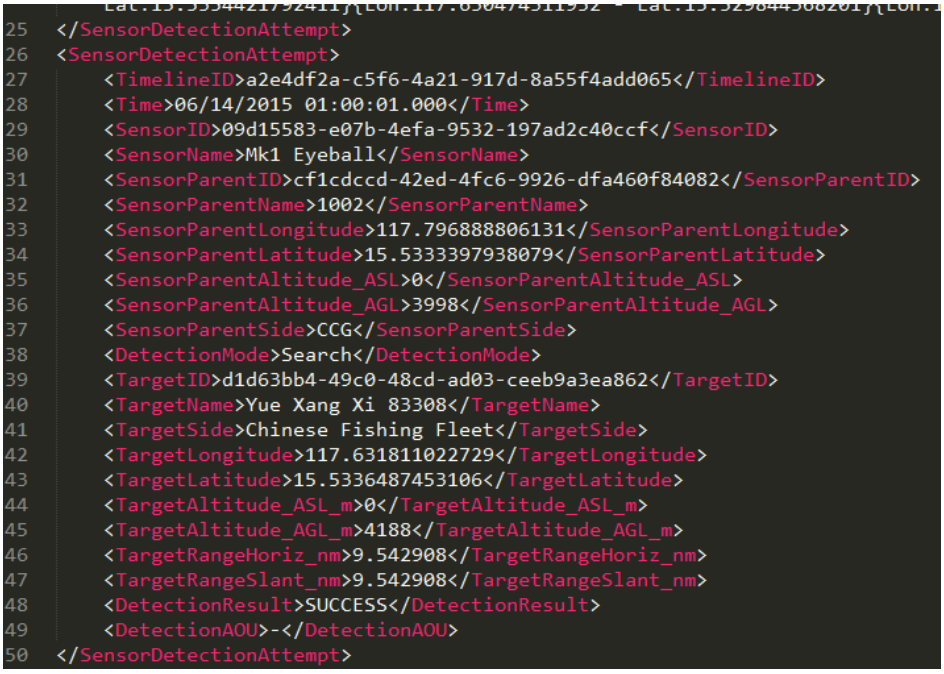
```
TimelineID,Time,SensorID,SensorName,SensorParentID,SensorParentName,SensorParentLongitude,SensorParentLatitude,SensorParentAltitude_ASL,SensorParentAltitude_AGL,SensorParentSide,Detecti
onMode,TargetID,TargetName,TargetSide,TargetLongitude,TargetLatitude,TargetAltitude_ASL_m,TargetAltitude_AGL_m,TargetRangeHoriz_nm,TargetRangeSlant_nm,DetectionResult,DetectionAOU
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,8767dcc3-16f6-45cc-8622-bbc34141a787,Bridgemaster E,6d775420-25d9-4a61-8a06-c7b6c8d82a3b,BRP Emilio
Jacinto,117.502924910958,15.2231882139013,0,3877,Philippine Navy,Search,1c81dfc5-f7f5-49a5-a45a-d847acde7b34,Gui Xiang 68,Chinese Fishing
Fleet,117.465802450834,15.3413392851763,0,4179,7.407513,7.407513,SUCCESS,-,
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,65a979a7-d48c-4a83-8bb4-03a568eb2c15,China Type 352 Square
Tie,cf1cdccd-42ed-4fc6-9926-dfa460f84082,1002,117.796888806131,15.5333397938079,0,3998,CCG,Search,d1d63bb4-49c0-48cd-ad03-ceeb9a3ea862,Yue Xang Xi 83308,Chinese Fishing
Fleet,117.631811022729,15.5336487453106,0,4188,9.542908,9.542908,SUCCESS,{Lon:117.646398582655 - Lat:15.5295258711501}{Lon:117.64635025994 - Lat:15.5345884211522}{Lon:117.646485347698
- Lat:15.5396495120248}{Lon:117.630011252945 - Lat:15.5403344174089}{Lon:117.629861817997 - Lat:15.5347189807057}{Lon:117.629915882575 - Lat:15.5291019401633},
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,bcaef3bc-33e4-4e02-922e-287b5c50a105,Radamec 2500 [EO],6d775420-25d9-4a61-8a06-c7b6c8d82a3b,BRP Emilio
Jacinto,117.502924910958,15.2231882139013,0,3877,Philippine Navy,Search,1c81dfc5-f7f5-49a5-a45a-d847acde7b34,Gui Xiang 68,Chinese Fishing
Fleet,117.465802450834,15.3413392851763,0,4179,7.407513,7.407513,SUCCESS,-,
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,6de0d96e-a421-40d5-86f5-637e1a5a869e,Mk1
Eyeball,cf1cdccd-42ed-4fc6-9926-dfa460f84082,1002,117.796888806131,15.5333397938079,0,3998,CCG,Search,d1d63bb4-49c0-48cd-ad03-ceeb9a3ea862,Yue Xang Xi 83308,Chinese Fishing
Fleet,117.631811022729,15.5336487453106,0,4188,9.542908,9.542908,SUCCESS,-,
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,ae356b83-22a7-4fb3-a9dd-04757bfbdfc1,Bridgemaster E,d98f2eee-bdaa-4cc1-8926-bc5d0e28391f,BRP Artemio
Ricarte,117.999766484153,15.2119876844073,0,3466,Philippine Navy,Search,bd0d2bd1-1188-4682-937e-0305d247fa26,Yue Yang Jiang 98135,Chinese Fishing
Fleet,118.075732323515,15.3457215665476,0,3725,9.149719,9.149719,SUCCESS,-,
a2e4df2a-c5f6-4a21-917d-8a55f4add065,06/14/2015 01:00:01.000,65a979a7-d48c-4a83-8bb4-03a568eb2c15,China Type 352 Square
Tie,cf1cdccd-42ed-4fc6-9926-dfa460f84082,1002,117.796888806131,15.5333397938079,0,3998,CCG,Search,bd0d2bd1-1188-4682-937e-0305d247fa26,Yue Yang Jiang 98135,Chinese Fishing
Fleet,118.075732323515,15.3457215665476,0,3725,19.66715,19.66715,SUCCESS,{Lon:118.075539425266 - Lat:15.3700645064656}{Lon:118.069454243809 - Lat:15.36079446024}{Lon:118.063037815001 -
Lat:15.3517346577222}{Lon:118.090734909376 - Lat:15.3327992761872}{Lon:118.097820424333 - Lat:15.3428019116073}{Lon:118.104540314172 - Lat:15.3530367165108},
```

The generated CSV files can be either left in the "raw" text form, or opened in other applications like MS-Excel for further processing & analysis:



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	TimelineI	Time	SensorID	SensorNa	SensorPar	SensorPar	SensorPar	SensorPar	SensorPar	SensorPar	SensorPar	Detection	TargetID	TargetNar	TargetSide	TargetLon	TargetLati	TargetAlti	TargetAlti	TargetRan	TargetRan	Detection	DetectionAOU
2	a2e4df2a-	00:01.0	8767dcc3-	Bridgemas	6d775420-	BRP Emilio	117.5029	15.22319	0	3877	Philippine	Search	1c81dfc5-	Gui Xiang	Chinese F	117.4658	15.34134	0	4179	7.407513	7.407513	SUCCESS	-
3	a2e4df2a-	00:01.0	65a979a7-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	9.542908	9.542908	SUCCESS	{Lon:117.646398582655 - Lat:15.529
4	a2e4df2a-	00:01.0	bcaef3bc-	Radamec	6d775420-	BRP Emilio	117.5029	15.22319	0	3877	Philippine	Search	1c81dfc5-	Gui Xiang	Chinese F	117.4658	15.34134	0	4179	7.407513	7.407513	SUCCESS	-
5	a2e4df2a-	00:01.0	6de0d96e	Mk1 Eyeba	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	9.542908	9.542908	SUCCESS	-
6	a2e4df2a-	00:01.0	ae356b83-	Bridgemas	d98f2eee-	BRP Arten	117.9998	15.21199	0	3466	Philippine	Search	bd0d2bd1	Yue Yang	Chinese F	118.0757	15.34572	0	3725	9.149719	9.149719	SUCCESS	-
7	a2e4df2a-	00:01.0	65a979a7-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	bd0d2bd1	Yue Yang	Chinese F	118.0757	15.34572	0	3725	19.66715	19.66715	SUCCESS	{Lon:118.075539425266 - Lat:15.370
8	a2e4df2a-	00:01.0	65a979a7-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	10.25697	10.25697	SUCCESS	{Lon:117.9627740105 - Lat:15.58062
9	a2e4df2a-	00:01.0	65a979a7-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	1c81dfc5-	Gui Xiang	Chinese F	117.4658	15.34134	0	4179	22.3467	22.3467	SUCCESS	{Lon:117.49304003913 - Lat:15.3325
10	a2e4df2a-	00:02.0	6249030a-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	9.542908	9.542908	SUCCESS	-
11	a2e4df2a-	00:02.0	d9f670cb-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	9.542908	9.542908	SUCCESS	-
12	a2e4df2a-	00:02.0	8e0d52ce-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	9.542908	9.542908	SUCCESS	-
13	a2e4df2a-	00:02.0	6249030a-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	10.25697	10.25697	SUCCESS	-
14	a2e4df2a-	00:02.0	d9f670cb-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	10.25697	10.25697	SUCCESS	-
15	a2e4df2a-	00:02.0	8e0d52ce-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	10.25697	10.25697	SUCCESS	-
16	a2e4df2a-	00:02.0	abcc8cdb-	AN/ZLQ-1	2a8b62cc-	Triton #03	119.3909	14.61991	18288	18288	United Sta	Search	d1d63bb4	Yue Xang	Chinese F	117.6318	15.53365	0	4188	115.7205	116.141	SUCCESS	{Lon:119.390878015501 - Lat:14.619
17	a2e4df2a-	00:02.0	abcc8cdb-	AN/ZLQ-1	2a8b62cc-	Triton #03	119.3909	14.61991	18288	18288	United Sta	Search	bd0d2bd1	Yue Yang	Chinese F	118.0757	15.34572	0	3725	87.78835	88.34197	SUCCESS	{Lon:119.390878015501 - Lat:14.619
18	a2e4df2a-	00:02.0	abcc8cdb-	AN/ZLQ-1	2a8b62cc-	Triton #03	119.3909	14.61991	18288	18288	United Sta	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	100.2495	100.7346	SUCCESS	{Lon:119.390878015501 - Lat:14.619
19	a2e4df2a-	00:02.0	abcc8cdb-	AN/ZLQ-1	2a8b62cc-	Triton #03	119.3909	14.61991	18288	18288	United Sta	Search	1c81dfc5-	Gui Xiang	Chinese F	117.4658	15.34134	0	4179	119.6797	120.0864	SUCCESS	{Lon:119.390878015501 - Lat:14.619
20	a2e4df2a-	00:03.0	8767dcc3-	Bridgemas	6d775420-	BRP Emilio	117.5029	15.22319	0	3877	Philippine	Search	1c81dfc5-	Gui Xiang	Chinese F	117.4658	15.34134	0	4179	7.407513	7.407513	SUCCESS	-
21	a2e4df2a-	00:03.0	8a3a29c5-	China Typ	cf1cdccd-4	1002	117.7969	15.53334	0	3998	CCG	Search	9bef247d-	Yuejiang	Chinese F	117.9695	15.57302	0	4043	10.25697	10.25697	SUCCESS	{Lon:117.796888806131 - Lat:15.533

- **XML format:** Writes events to XML-formatted files, on the path **[top-level writable folder]\Analysis_Int\EventExport.xml** (single file)



```
</SensorDetectionAttempt>
<SensorDetectionAttempt>
    <TimelineID>a2e4df2a-c5f6-4a21-917d-8a55f4add065</TimelineID>
    <Time>06/14/2015 01:00:01.000</Time>
    <SensorID>09d15583-e07b-4efa-9532-197ad2c40ccf</SensorID>
    <SensorName>Mk1 Eyeball</SensorName>
    <SensorParentID>cf1cdccd-42ed-4fc6-9926-dfa460f84082</SensorParentID>
    <SensorParentName>1002</SensorParentName>
    <SensorParentLongitude>117.796888806131</SensorParentLongitude>
    <SensorParentLatitude>15.5333397938079</SensorParentLatitude>
    <SensorParentAltitude_ASL>0</SensorParentAltitude_ASL>
    <SensorParentAltitude_AGL>3998</SensorParentAltitude_AGL>
    <SensorParentSide>CCG</SensorParentSide>
    <DetectionMode>Search</DetectionMode>
    <TargetID>d1d63bb4-49c0-48cd-ad03-ceeb9a3ea862</TargetID>
    <TargetName>Yue Xang Xi 83308</TargetName>
    <TargetSide>Chinese Fishing Fleet</TargetSide>
    <TargetLongitude>117.631811022729</TargetLongitude>
    <TargetLatitude>15.5336487453106</TargetLatitude>
    <TargetAltitude_ASL_m>0</TargetAltitude_ASL_m>
    <TargetAltitude_AGL_m>4188</TargetAltitude_AGL_m>
    <TargetRangeHoriz_nm>9.542908</TargetRangeHoriz_nm>
    <TargetRangeSlant_nm>9.542908</TargetRangeSlant_nm>
    <DetectionResult>SUCCESS</DetectionResult>
    <DetectionAOU>-</DetectionAOU>
</SensorDetectionAttempt>
```

- **MS-Access database:** Writes to a single MS-Access database file, on the path **[top-level writable folder]\Analysis_Int\EventExport.mdb** (single file)

- **Tacview 1.x/2.x format:** Writes events to a single ACMI format file used by Tacview, on the path: **[top-level writable folder]\Analysis_Int\Tacview1x.acmi** and **Tacview2x.acmi** respectively (see also section “Tacview visualization”).

- **Tacview RT (real-time):** Streams events either locally or over the network to a running Tacview instance configured for real-time streaming (see also section “Tacview visualization”).

- **SIMDIS file format:** Writes events to a single .asi format file used by SIMDIS, on the path: **[top-level writable folder]\Analysis_Int\SIMDIS.asi** (see also section “SIMDIS visualization”).

Additional destination types (other databases, web services etc.) can be added on request.

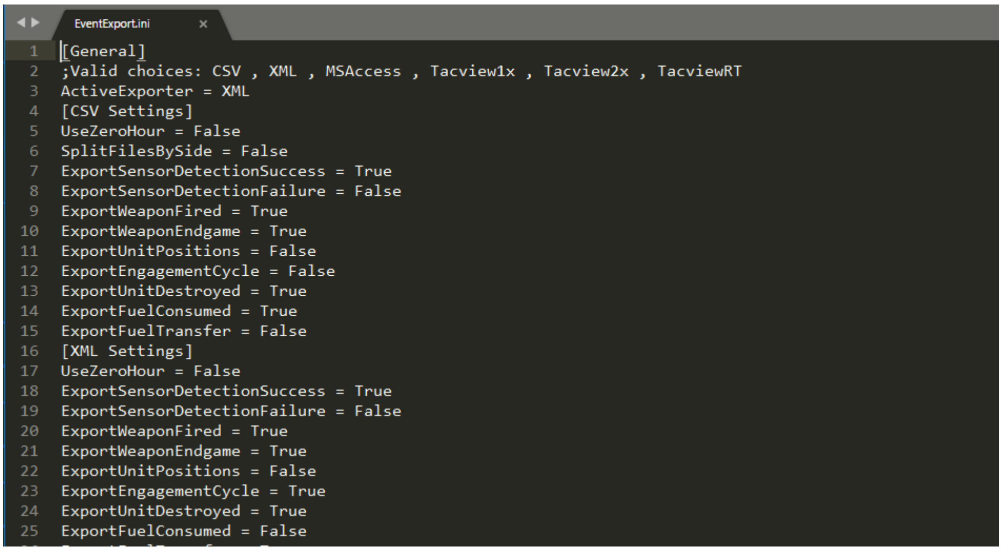
The following event types are currently exportable in both interactive and Monte-Carlo mode:

- Sensor detections (both successes and failures)
- Weapons fired
- Weapon engagements
- Unit destroyed
- Unit damaged
- Unit locations and movements
- Fuel consumption
- Fuel transfers (refueling etc.)
- AI contact-handling pipeline & engagement lifecycle (initial contact creation, contact classification/ID, targeting, weapon assignment, etc.)
- Air operations (aircraft taking off and landing)

- Docking operations (water craft docking / undocking from a mothership)
- Cargo transfer

Any additional event type desired can be added on request.

The selection of the event-export destination, as well as the various parameters, is controlled by the file **[top-level writable folder]\Config\EventExport.ini**:



```
[General]
;Valid choices: CSV , XML , MSAccess , Tacview1x , Tacview2x , TacviewRT
ActiveExporter = XML
[CSV Settings]
UseZeroHour = False
SplitFilesBySide = False
ExportSensorDetectionSuccess = True
ExportSensorDetectionFailure = False
ExportWeaponFired = True
ExportWeaponEndgame = True
ExportUnitPositions = False
ExportEngagementCycle = False
ExportUnitDestroyed = True
ExportFuelConsumed = True
ExportFuelTransfer = False
[XML Settings]
UseZeroHour = False
ExportSensorDetectionSuccess = True
ExportSensorDetectionFailure = False
ExportWeaponFired = True
ExportWeaponEndgame = True
ExportUnitPositions = False
ExportEngagementCycle = True
ExportUnitDestroyed = True
ExportFuelConsumed = False
```

The structure of the settings file allows mix-and-matching different output destinations for different event types. For example, a user may configure unit position updates to be streamed to Tacview for real-time display, but fuel consumption events to be stored to a database for later processing.

Once any of the settings in this file is changed, CPE must be restarted for these changes to take effect.

To run a scenario in interactive analysis mode:

1) Specify one or more export destinations in the **Event Export.ini** file (e.g. “ActiveExporter = CSV, Tacview2x”). For each destination, configure the desired event types to be exported by setting them to “True”.

2) Startup Command, and load and run a scenario as usual. You can visually confirm that the events are exported, by enabling "Show Diagnostics" in the "Game Options" window. The scenario status text (White-text on black-background label on top of the map) will then display, among other scenario information, the line "Event Queue Length: XXX" showing the current queue(s) for the events being exported to disk/network/etc.

Customizing output destination & export rate

By default, the generated outputs of an interactive run (when directed to text files of various formats as opposed to, for example, streamed to database servers) are written on the folder **[top-level writable folder]\Analysis_Int** and are overwritten or appended between successive runs**.** However, it is possible both to change the top-level target folder and also to separate the generated results for each run.

To change the top-level output folder, you can use the **OutputRoot** property on the file **[top-level writable folder]\Config\EventExport.ini**. For example:

```
[General]

; Valid choices: CSV , XML , MSAccess , Tacview1x , Tacview2x,
TacviewRT, SQLServer, SQLite, SIMDIS

ActiveExporter = CSV, Tacview2x, SQLite, TacviewRT

OutputRoot = "C:\InteractiveResults"

[CSV Settings]

UseZeroHour = True

SplitFilesBySide = False

[.....]
```

To separate the outputs of each interactive analysis run, you can use the **PartitionOutputs** property on the file **[top-level writable folder]\Config\EventExport.ini**. For example:

```
[General]

; Valid choices: CSV , XML , MSAccess , Tacview1x , Tacview2x,
TacviewRT, SQLServer, SQLite, SIMDIS

ActiveExporter = CSV, Tacview2x, SQLite, TacviewRT

OutputRoot = "C:\InteractiveResults"

PartitionOutputs = True

[CSV Settings]

UseZeroHour = True

SplitFilesBySide = False

[.....]
```

If the value is set to True, each interactive run (same or different scenario) outputs its results on a subfolder within [OutputRoot] whose name is a combination of the scenario name and date & time of the start of the execution. If the value is not present, or is set to False, all outputs are stored on [OutputRoot] as default.

It is also possible to customize the output rate of the “Unit Positions” export table. Instead of exporting unit positions on every second (the default), a different rate may be set, using the **ExportUnitPositions** value under the **[Output Rate]** configuration section in the file **[top-level writable folder]\Config\EventExport.ini**. For example:

```
[General]

; Valid choices: CSV , XML , MSAccess , Tacview1x , Tacview2x,
TacviewRT, SQLServer, SQLite, SIMDIS

ActiveExporter = CSV, Tacview2x, SQLite, TacviewRT

OutputRoot = "C:\InteractiveResults"

[Output Rate]

ExportUnitPositions = 5SECONDS

[CSV Settings]

UseZeroHour = True

SplitFilesBySide = False

ExportSensorDetectionSuccess = True

[.....]
```

The valid inputs are:

- "CONTINUOUS": Outputs on every sim pulse
- "1SECOND" : Outputs on every second
- "2SECONDS" / "2SECOND" : Outputs every 2 sim seconds
- "5SECONDS" / "5SECOND" : Outputs every 5 sim seconds
- "15SECONDS" / "15SECOND" : Outputs every 15 sim seconds
- "30SECONDS" / "30SECOND" : Outputs every 30 sim seconds
- "1MINUTE" : Outputs every sim minute
- "5MINUTES" / "5MINUTE" : Outputs every 5 sim minutes
- "15MINUTES" / "15MINUTE" : Outputs every 15 sim minutes
- "30MINUTES" / "30MINUTE" : Outputs every 30 sim minutes
- "1HOUR" : Outputs every sim hour
- "6HOURS" / "6HOUR" : Outputs every 6 sim hours
- "12HOURS" / "12HOUR" : Outputs every 12 sim hours
- "24HOURS" / "24HOUR" : Outputs every 24 sim hours
- "MATCHSIMSPEED": The output rate automatically matches the selected time acceleration setting if in interactive mode. For example if the user has set time accel to 1-5 mins, the output rate will be once every 5 sim minutes.

Speed bands: Dynamic export rate based on unit speed

Sometimes it is desirable, instead of a static predefined export interval, to vary the export frequency dynamically per-unit, based on a unit's speed. For example, we may want very rapid updates on ballistic missiles and other exo-atmospheric objects, less frequent updates on aircraft, even less so for ships or ground vehicles, and very sporadic "heartbeat" updated for static facilities. "Speed bands" is a configurable speed interval where we can do just that, by applying a specific position output frequency. This can save both CPU processing power and disk space from redundant data.

To facilitate this, the **EventExport.ini** file has been expanded to include additional per-exporter settings. They are grouped under the headings [General], [CSV Settings], [XML Settings] etc. This allows customizing the export rates not only per speed band but also individually for each exporter.

Here is an example of part of the revised config file with the new settings:

```
[CSV Settings]
UseZeroHour = False
SplitFilesBySide = True
ExportSensorDetectionSuccess = True
ExportSensorDetectionFailure = True
ExportWeaponFired = False
ExportWeaponEndgame = False
ExportUnitPositions = True
ExportEngagementCycle = True
ExportUnitDestroyed = True
ExportFuelConsumed = True
ExportFuelTransfer = True
ExportAirOps = True
ExportDockingOps = True
ConsolidateCSV = False
UseCustomUnitExportFrequency = true
SpeedBandFrequency0To20 = 60000
SpeedBandFrequency20To40 = 30000
SpeedBandFrequency40To80 = 15000
SpeedBandFrequency80To160 = 7500
SpeedBandFrequency160To320 = 3250
SpeedBandFrequency320To640 = 1625
SpeedBandFrequency640To1280 = 800
SpeedBandFrequency1280To2560 = 400
SpeedBandFrequencyOver2560 = 200
SpeedBandFrequencyDynamic = 0
```

(If these settings are not already present on the config files, default values are created for them on first-time use)

Speed band frequency

Here we can define, for each speed bands in knots, the desired export frequency (using simulation time) in milliseconds.

Examples:

SpeedBandFrequency20To40 = 30000

What this actually means:

1 position export entry will be written every 30 simulation seconds, for units having a speed between 20 and 40 knots.

```
SpeedBandFrequency0To20 = 60000

SpeedBandFrequency20To40 = 30000

SpeedBandFrequency40To80 = 15000

SpeedBandFrequency80To160 = 7500

SpeedBandFrequency160To320 = 3250

SpeedBandFrequency320To640 = 1625

SpeedBandFrequency640To1280 = 800

SpeedBandFrequency1280To2560 = 400

SpeedBandFrequencyOver2560 = 200
```

More detail: custom frequency for specific units

If an even finer control over the position output frequency of a unit (or a group of units) is desired, we can edit a separate configuration file called **ExportUnitLocationFrequencySettings.csv** file. This file must remain in the **[WritableRoot]\config** folder and follow the template's format.

Unit's GUID	Override
186a6e85-9be3-4a6b-96bb-32ea33db3c2d	500
525r7e21-3jh1-4a6b-55uu-35e234gb3r2d	300

An example of such a CSV file (the first row is ignored):

```
Unit's GUID,Override

my guid 1,500

my guid 2,300
```

In this example, this first unit will output its position every 0.5 second (500ms)

The second unit will export it every 0.3 seconds (300ms)

The exporter will not use the custom frequency unless **UseCustomUnitExportFrequency** is set to true.

If **UseCustomUnitExportFrequency** is active but no value is found for the unit, the fallback value will be either the dynamic frequency or the defined speed band, as normal.

Dynamic frequency based on unit's speed

SpeedBandFrequencyDynamic will be evaluated if it has any value different to 0. This behavior has priority over the rest except for **Custom frequency for specific units.**

Dynamic frequency works differently to the rest as it changes according to the unit's current speed :

Resulting frequency = SpeedBandFrequencyDynamic / Unit current speed * 1000

Therefore, as a unit goes faster, the frequency will increase, the dynamic frequency controls that rate proportionally.

This is a very powerful tool, especially for large scenarios, it is possible to reduce the amount of written data by orders of magnitude while keeping a high level of accuracy.

A good methodology is to have a target reference accuracy, and then calculate backward the **SpeedBandFrequencyDynamic.**

For example, assume say our goal is to get 1 position update every second for a unit going at 10knots. (Therefore 2 updates a second for units going at 20 knots etc…).

The **SpeedBandFrequencyDynamic = 100** (0.1sec)

Since **10 000 (10sec) = 100 /10 *1000**

Finding the sweet spot for dynamic frequency while having frequency overrides for very specific units allows to create very lightweight analysis in a large scenario while keeping extremely accurate focused data.

Changing settings through scripting

Event export settings can be changed while a scenario is executing using the Lua function Exporter_SetSettings. This allows you to enable or disable exporters and events without requiring a complete restart of Command. For example:

```
        Exporter_SetSetting("General","ActiveExporter","Tacview2x,CSV")
```

The script call above would enable both the Tacview2x exporter and the CSV exporters, just as if you had edited the EventExport.ini file and restarted Command. Note that you must explicitly list all exporters you wish to have enabled, as anything not listed as an ‘active exporter’ will be disabled by default.

You can also use Exporter_SetSettings to change the individual exports for each output type. For example, the following would enable the unit positions event for the CSV exporter, just as if you’d change the ‘CSV Settings’ section of EventExport.ini and restarted Command:

```
        Exporter_SetSetting("CSV Settings","ExportUnitPositions","true")
```

See the Command Professional Edition Lua documentation for further details on the Exporter_SetSettings function.

Some more usage examples:

Exporter_SetSetting("General","ActiveExporter","XML,Tacview1x") *- Set the active exporter to XML and Tacview 1x*

Exporter_SetSetting("Tacview Settings","SpeedBandFrequency0To20","2000") *- Set the Tacview exporter frequency to 2 for the speed band between 0 and 20 knots, meaning that 1 position export entry will be generated every 2 simulation seconds (2000ms), for units having a speed between 0 and 20 knots.*

Scripting walkthrough example

Let us consider an example of using this feature through scripting. To get started, you will perform an export in Tacview mode of a scenario of your choice (preferably a scenario with a good variety of unit's types). Tacview is ideal to visualize the effects of the changes in position export frequencies.

And we will add another exporter during the course of the scenario.

Let us begin with these default values:

```
UseCustomUnitExportFrequency = true

SpeedBandFrequency0To20 = 0

SpeedBandFrequency20To40 = 0

SpeedBandFrequency40To80 = 0

SpeedBandFrequency80To160 = 0

SpeedBandFrequency160To320 = 0

SpeedBandFrequency320To640 = 0

SpeedBandFrequency640To1280 = 0

SpeedBandFrequency1280To2560 = 0

SpeedBandFrequencyOver2560 = 0

SpeedBandFrequencyDynamic = 0
```

Set this, for each exporter:

```
ExportUnitPositions = False
```

And make sure no event exporter is activated:

```
ActiveExporter = None
```

Now open the desired scenario in CPE. Subsequently, open the Lua script console and run :

```
Exporter_SetSetting("General","ActiveExporter","Tacview2x")
```

We are enabling the Tacview exporter.

```
Exporter_SetSetting("Tacview Settings","SpeedBandFrequencyDynamic","200")
```

We are setting **SpeedBandFrequencyDynamic** to 200 for the Tacview exporter. Meaning that we are going to export the unit position every 200 seconds/knots.

Argh! But we forgot to enable the csv exporter, so we need to restart the scenario…..?

Actually, we do not.

The export enables hot loading of exporters without crushing existing ones:

```
Exporter_SetSetting("General","ActiveExporter","Tacview2x,CSV")
```

Make sure, to include all exporters you want to be active, not just the new ones you are adding.

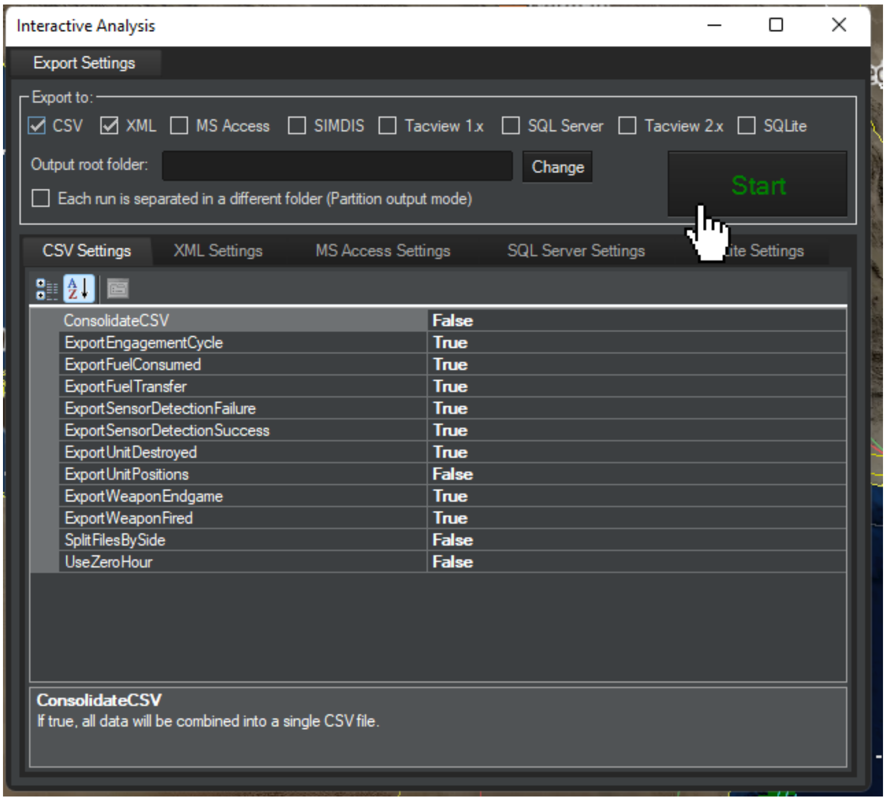
If we had not included Tacview2x in the above Lua command, the Tacview2x exporter would be deactivated and flushed.

Finally, we want to make sure that position export is enabled for CSV export:

```
Exporter_SetSetting("CSV Settings","ExportUnitPositions","true")
```

Interactive analysis settings window

A common drawback of directly editing the EventExport.ini file is that the application must be restarted in order for these changes to take effect. An alternative way to configure the most common analysis & export settings is to use the “Interactive Analysis Settings” window (Editor → Interactive Analysis Settings):



Export to: ☑ CSV ☑ XML □ MS Access □ SIMDIS □ Tacview 1.x □ SQL Server □ Tacview 2.x □ SQLite

Output root folder:

□ Each run is separated in a different folder (Partition output mode)

ConsolidateCSV	False
ExportEngagementCycle	True
ExportFuelConsumed	True
ExportFuelTransfer	True
ExportSensorDetectionFailure	True
ExportSensorDetectionSuccess	True
ExportUnitDestroyed	True
ExportUnitPositions	False
ExportWeaponEndgame	True
ExportWeaponFired	True
SplitFilesBySide	False
UseZeroHour	False

ConsolidateCSV
If true, all data will be combined into a single CSV file.

This makes it possible to change various settings for event export without needing to restart the application. It also makes it possible to start or stop an export sequence at will.

Any changes to any of the listed per-format export settings will take effect immediately if an export is active at the time (ie. there is no need to stop and restart the export process). However, in order to add or remove an export format (e.g. start exporting to XML in addition to CSV), or in order to change the destination file, the currently running export run must be stopped and then resumed.

Note that using the EventExport.ini config file is also still supported, as the raw config file exposes some properties which are not currently displayed on the GUI window (for example, customizing the export rate).

Save scenario as XML by default

(Available in PE & AE)

CPE can be configured to save scenarios in their “raw” XML format instead of compressing them into a binary form inside an XML container. To enable this, the Boolean config value **SaveAsXML** must be set in the file **[top-level writable folder]\Config\CPE.ini .** For example:

```
[General]

UsePitchForWeapons = 1

CustomFolder_GIS =

CustomFolder_DB =

CustomFolder_WW =

SaveAsXML = True

[Lua]

EnableSocket = 0

[.....]
```

If set to True, when saving a scenario manually ("Save" or "Save As..." commands on the main menu), the scenario is saved in its raw XML format. Scenarios thus saved can be subsequently loaded into CPE through the "Editor" menu *(Editor --> Scenario Migration --> Import from file).*

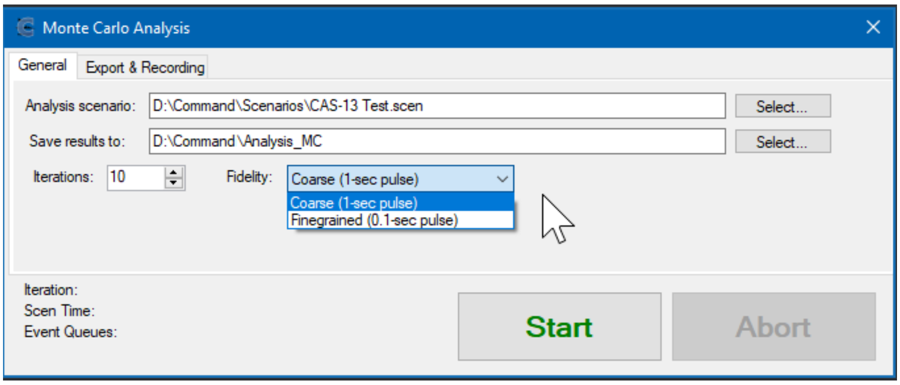
Non-interactive (Monte Carlo) analysis

(Available in PE – Optional in AE)

Monte Carlo-mode analysis is similar to interactive mode in the output options available, but differs from it in a number of other ways:

- The scenario runs in "headless" mode, without any visualization or possibility of intervention by the user.
- The user can set the scenario to be batch-run an arbitrary number of times, to generate multiple result sets for subsequent statistical analysis.
- Because of the headless execution, simulation performance is higher.

To run a scenario in Monte-Carlo mode, from the top-level menu click on "Editor --> Monte-Carlo Analysis" and set the desired settings on the window that appears:



Analysis scenario: D:\Command\Scenarios\CAS-13 Test.scen

Save results to: D:\Command\Analysis_MC

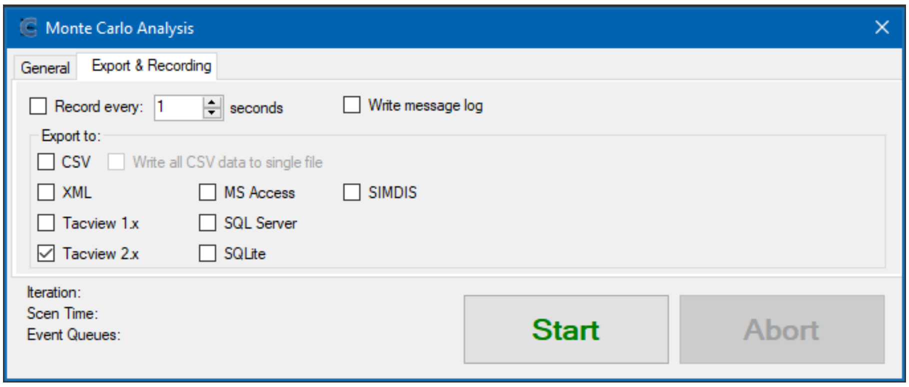
Iterations: 10

Fidelity: Coarse (1-sec pulse)

Coarse (1-sec pulse)
Finegrained (0.1-sec pulse)

Iteration:
Scen Time:
Event Queues:

On first tab, "General", we can select the scenario to be executed, the output folder for the event export results, and the number of iterations. We can also specify if the analysis will run in coarse (1-sec pulse) or fine-grained (0.1-sec pulse) mode.



□ Record every: 1 seconds

□ Write message log

Export to:

□ CSV □ Write all CSV data to single file

□ XML □ MS Access □ SIMDIS

□ Tacview 1.x □ SQL Server

☑ Tacview 2.x □ SQLite

Iteration:
Scen Time:
Event Queues:

On the "Export & Recording" tab we can specify the desired export destination(s). **IMPORTANT NOTE: These selections override the "ActiveExporter" setting in the EventExport.ini file.** As in interactive analysis, we can mix-and-match different destinations for different event types.

One additional option here is setting the scenario to be recorded ("Record every X seconds"). This treats the recorder as another "data export" destination, which can be quite useful when used in combination with the other generated data. (For example, if during analysis we notice an event of interest, and want to understand the context under which it happened, we can use the recorded snapshot to literally jump into the full state of the scenario at that moment and better understand how and why this event happened.). We can also select whether the generated message log will be written on each iteration (checkbox “Write message log”).

Once all desired settings are set, click on “Start” to begin the analysis run. The execution can be paused, resumed and aborted at any point.

Because of the inherent batch-running nature of non-interactive mode, the path locations of the generated files are different. They are grouped under folders representing the iteration under which they were generated. For example, for iteration #3 in an analysis run, the file paths will be:

- For CSV format : [top-level writable folder]\Analysis_MC\3\xxx.csv (multiple files, one for each event type, and even more if the "SplitFilesBySide" option is enabled)
- For XML : [top-level writable folder]\Analysis_MC\3\EventExport.xml (single file)
- For MS-Access: [top-level writable folder]\Analysis_MC\3\EventExport.mdb (single file)
- For Tacview1x/Tacview2x: [top-level writable folder]\Analysis_MC\3\Tacview1x.acmi and Tacview2x.acmi respectively.
- For SIMDIS: [top-level writable folder]\Analysis_MC\3\SIMDIS.asi
- For the .rec file produced by the Command recorder (for playback & resume), if enabled: [top-level writable folder]\Analysis_MC\3\3.rec

Regardless of the number of iterations, there will also be a "Summary.txt" file generated on the [top-level writable folder]\Analysis_MC folder, listing the summary results (losses & expenditures etc.) of each iteration. This allows a quick overview of the analysis run (handy for quickly getting the averages and picking up outliers for subsequent scrutiny).

CAUTION: Each time a new Monte-Carlo run commences, the files generated by the last one are erased. So, if the analysis workflow is along the lines of "let it run, then go over results of run #1 while run #2 executes", it is necessary to copy/move the entire [top-level writable folder]\Analysis_MC folder elsewhere for deeper processing before commencing the next run.

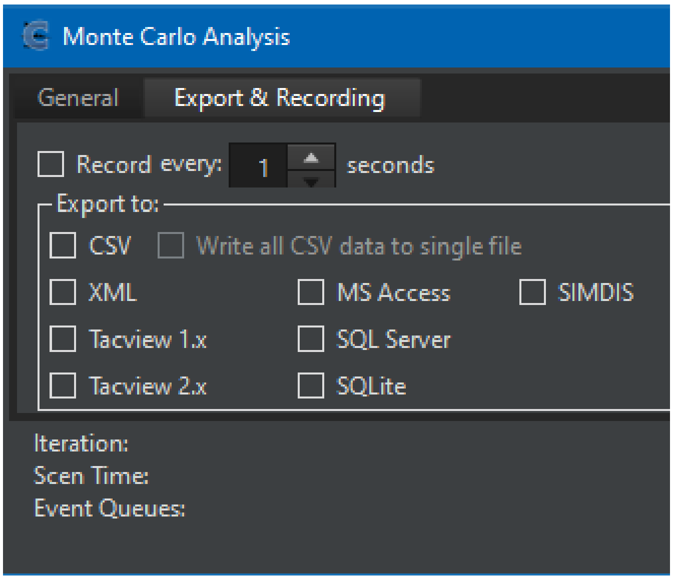
Performance tune-up: Avoiding common pitfalls in interactive & Monte-Carlo analysis

If you encounter a slowdown/freeze in the execution of an interactive analysis or Monte-Carlo run, this can usually be traced to one of two culprits:

1) The scenario itself encounters some performance or RAM bottleneck/hotspot, or
2) There is a bottleneck in the event export itself.

There are two quick checks that you can perform in order to narrow down the likely cause of the issue:

1) Perform a “dry run” of the scenario, with all export options de-selected:



Monte Carlo Analysis

General | Export & Recording

□ Record every: 1 seconds

Export to:

□ CSV □ Write all CSV data to single file

□ XML □ MS Access □ SIMDIS

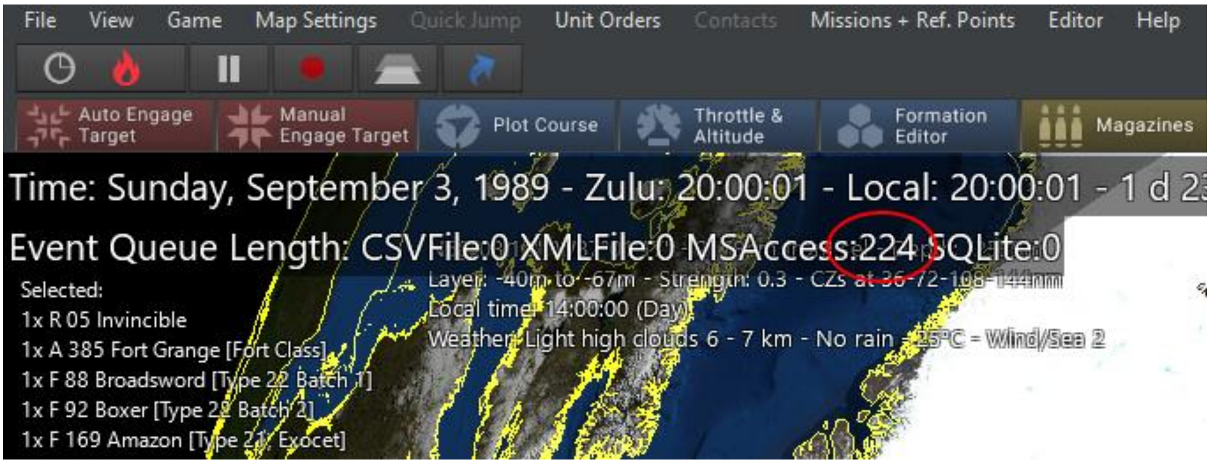
□ Tacview 1.x □ SQL Server

□ Tacview 2.x □ SQLite

Iteration:
Scen Time:
Event Queues:

If the scenario runs for a prolonged time without any slowdown/freeze, it is almost certainly an issue on the export side. If the problem persists, there is probably a problem on the scenario itself.

2) Do a normal test run of the scenario, with your desired destinations enabled, and observe the event-queue numbers shown at the black status bar on the top of the map area. For example:



Time: Sunday, September 3, 1989 - Zulu: 20:00:01 - Local: 20:00:01 - 1 d 2

Event Queue Length: CSVFile:0 XMLFile:0 MSAccess:224 SQLite:0

Selected:
1x R 05 Invincible
1x A 385 Fort Grange [Fort Class]
1x F 88 Broadsword [Type 22 Batch 1]
1x F 92 Boxer [Type 22 Batch 2]
1x F 169 Amazon [Type 21, Exocet]

Layer: -40m to -67m - Strength: 0.3 - CZs at 36-72-108-144nm
Local time: 14:00:00 (Day)
Weather: Light high clouds 6 - 7 km - No rain - 25°C - Wind/Sea 2

If any of the queue numbers displayed there is over 300-400K, there is likely a problem with event export.

If the problem is in the scenario itself:

If possible, it is highly recommended to share the scenario (or a sanitized version of it, if sensitive) with the Command-PE development team. Most/all of the "common" performance culprits have long been weeded out of the sim codebase, so any bottlenecks still appearing are likely to be the result of "perfect storms" of unusual (and individually harmless) circumstances. The dev team can diagnose such issues and usually resolve them promptly (or, if impossible to directly solve, provide advice for workarounds).

If the problem lies with the export of generated events:

This is a far more common case in our experience. In order to run the actual simulation as fast as possible, Command does not export the generated events "in line" with sim execution; instead, it offloads them to dedicated export threads (one per export destination), which buffer and send/write the events as appropriate. These buffers exist as in-memory queues (with one exception, see below), and if for any reason these queues get too large in size (400-500.000 is a common threshold) then the program (and indeed the OS itself, depending on available total memory) may begin to suffer from RAM starvation and exhibit slowdowns and eventually freezes.

In order to understand the best practices for avoiding such a situation, we need a short overview of the nature & peculiarities of each export type:

*** CSV:** This writes the generated events to a comma-delimited raw text file. This is the fastest on-disk option and the most commonly used due to its versatility (the CSV file can be directly opened in tabular form in Excel, imported to a database etc.). A large queue size in this exporter typically means a slow writing speed, which can be the result of:

a) Slow storage medium, i.e. a slow mechanical hard disk (unfortunately this is still common in many laptops). Fast storage hardware is highly recommended: Any SSD is a large improvement over any HDD, and if available a RAM-disk (https://en.wikipedia.org/wiki/RAM_drive) volume is ideal.

b) A particularly intrusive antivirus software. Many AVs monitor file operations carefully and will inspect every write operation to disk for possible malware behavior (Norton Antivirus is a particularly draconian example). This can grind the event export speed to a trickle even on lightning-fast storage hardware. Thankfully, most AVs offer the ability to exclude specific files and folders from real-time monitoring. Therefore, excluding the files/folders in which the events are being written should resolve the problem.

*** XML:** This writes the generated events to an XML-formatted file. This is slightly slower than CSV but otherwise similar in method and performance, and the same pitfalls and guidelines apply.

*** MS Access:** This is the only exporter type that uses a disk-backed queue instead of a RAM-based one, and for good reason: Microsoft's Access database is painfully slow in inserting data, even with superfast storage hardware (the bottleneck is at Access's low-level routines for inserting data to tables), so adopting a RAM-based queue for this exporter would lead to RAM exhaustion almost immediately during a Monte-Carlo run. Technically, the disk-based queue also means that even very large queue sizes are unlikely to cause RAM starvation-related slowdowns/freezes; however, piling generated events to the queue itself is a relatively slow operation (because it actually writes to disk instead of writing to a RAM queue) so the actual sim execution slows in speed. Generally, we do not recommend using Access for any high-frequency events (e.g. unit positions, sensor detection checks, fuel consumption etc.).

*** Tacview1x/Tacview2x/SIMDIS:** These exporter types are similar to CSV & XML in that they write to disk files, which are then read by external applications. The pitfalls and guidelines of CSV also apply here.

*** TacviewRT:** This exporter type buffers Tacview-relevant events to be streamed to a running instance of Tacview in real time. If no Tacview instance is connected to Command, then the buffered events are discarded instead of piling up, so this exporter is highly unlikely to cause any performance issues (there has been no reported issue so far).

*** SQL Server:** Microsoft's flagship DB server solution is currently our fastest option for exporting events. It effortlessly swallows even high-frequency events (e.g. unit positions, sensor detection checks, fuel consumption etc.) from very large scenarios, so we recommend it for such setups. The only case where a queue ballooning has been observed is when the connection to the DB server is temporarily interrupted or broken altogether. Interruptions of up to a few seconds can be compensated by the queue buffer, but longer-duration disruption can cause a problem.

*** SQLite:** This exporter inserts events into an SQLite database. Contrary to SQL Server, SQLite is a file-based DB solution (similar in this respect to MS-Access) but a very fast one, only losing slightly in raw speed to CSV. The pitfalls and guidelines of CSV also apply here.

Regardless of the destination type(s) selected, some performance guidelines are universal:

1) Export only the event types that are of interest.

2) Take advantage of the ability to export different event types to different destinations concurrently. For example, in a large scenario, high-volume events can be directed to a fast consumer like a CSV file or SQL Server or SQLite database, while events typically happening less frequently (e.g. fuel transfer, weapon impacts etc.) may be written to less-performant destinations that may offer other advantages.

3) Experiment! Each scenario is unique in the frequency & volume of events it generates, and underlying hardware is an important factor, so frequent measurements and tailoring of the export settings to the situation at hand are essential for trouble-free statistical runs.

Customizing simulation fidelity

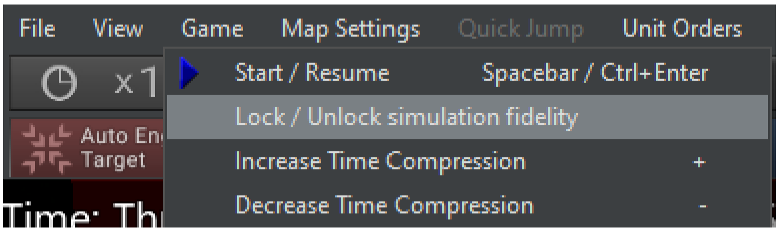
Command is processing simulation into time-slices / steps of variable value. A simulated second in Command can either be performed in one step (time resolution = 1 sec) or ten steps (time resolution = 0.1 sec).

By default, Command uses a “hybrid” resolution mode: Time resolution will work in 0.1-sec steps in all time compression modes except for the ‘turbo’ mode, symbolized with a flame in the UI. When in turbo mode, Command switches to the coarse (1-sec slice) resolution. Therefore, time resolution is tied to the time compression selected.

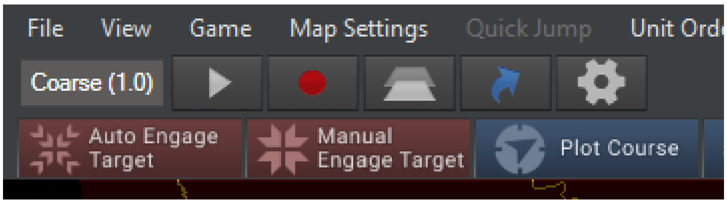
It is, however, sometimes desirable to dynamically alter the time resolution based on simulation events (i.e. use fine-grained resolution when friendly units are in a threat area, while using coarse resolution in other circumstances). CPE provides the ability to perform this both through the user interface and through Lua scripting. It is also possible to “lock” the game resolution throughout the course of the scenario.

Using the interface

The menu button “Game → Lock / Unlock simulation fidelity” will either lock or unlock the time resolution. Locking the time resolution means that time compression will no longer affect the time resolution (ie. the fidelity remains constant regardless of time acceleration):



Enabling this option also disables time compression interactions, and add a new toggle to control the time resolution:



Using Lua

Likewise, we can use the **ScenEdit_SetSimulationFidelity** method to forcefully set the time resolution (valid values are 0.1 and 1.0). This call will implicitly lock the time resolution:

```
ScenEdit_SetSimulationFidelity(Fidelity As Single) As String
```

We can also use the **ScenEdit_LockSimulationFidelity** method to explicitly lock/unlock the time resolution:

```
ScenEdit_LockSimulationFidelity(IsLocked As Boolean)
```

Multiple-instance execution & folder redirection

(Available in PE & AE)

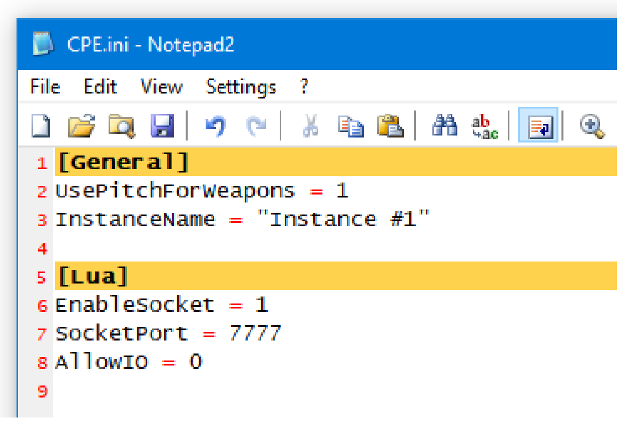
CPE supports running multiple instances of the program in parallel. This can offer two main advantages, particularly in the context of scenario analysis with multiple scenarios or variations of the same basic scenario:

- The multiple scenarios can be set up to run for analysis concurrently on the multiple instances, instead of sequentially. As the outputted results of analysis can be examined as they are generated, well before each analysis run is concluded, this allows much faster comparison of the results of different scenario inputs.

- It allows better utilization of system resources in powerful hardware. In modern PC systems, Command's core simulation engine generally takes advantage of up to 6-7 logical CPU threads; higher utilization is possible but less common (line-of-sight calculations can easily spike to 100% CPU usage). Modern systems can offer more CPU threads than this, and such setups are becoming increasingly affordable. By spawning additional running instances, and provided sufficient RAM is available, Command can make full use of such hardware to dramatically accelerate the analysis cycle.

There are two strategies to utilize this capability:

1) Copy the entire CPE installation folder to a different file system location and run the second instance from there. (Running multiple instances from the same exact file system executable location is not currently supported).
In this usage scenario, it can be useful to visually distinguish the two+ running instances in order to avoid confusion. To support this, the configuration file CPE.ini now supports an additional parameter, "InstanceName". For example:

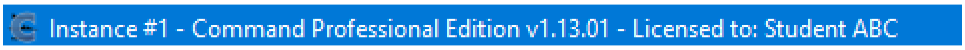


```
[General]
UsePitchForWeapons = 1
InstanceName = "Instance #1"

[Lua]
EnableSocket = 1
SocketPort = 7777
AllowIO = 0

```

This instance name is then displayed on the main window of Command PE when running:



Instance #1 - Command Professional Edition v1.13.01 - Licensed to: Student ABC

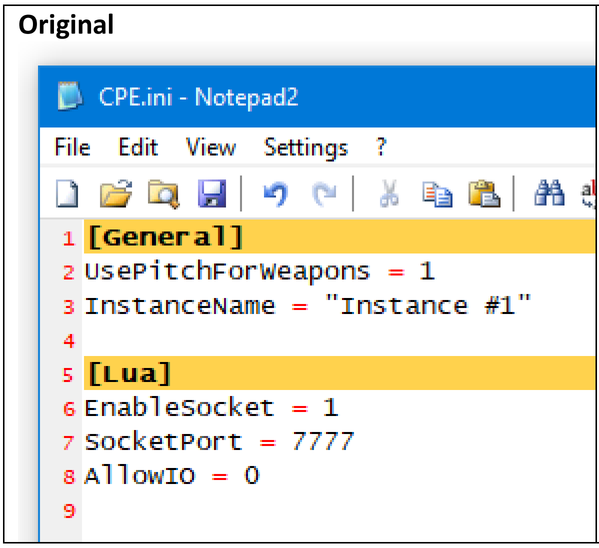
….thus making it easy to distinguish between multiple instances running on the same system.

This method offers simplicity but is somewhat wasteful in storage space, as the entire installation folder, with its gigabytes of data, is duplicated. This leads us to the second method:

2) Configure the mirror copies of the original installation to have their most storage-expensive data folders redirect to the original one. The larger folders in terms of storage are the **\DB** (databases), **\GIS** (GIS and terrain data) and **\WW** (map tiles and additional GIS data) folders. Each of them can be redirected to the equivalent folders of the original installation, dramatically reducing the storage footprint of the "clone" instance(s).

Let us consider an example of the latter method. Assume that the original installation is on the path "C:\CMANO-PE\". We copy the entire installation folder to any other point in the file system. Here is how the configuration files for each instance would be set up:

Original

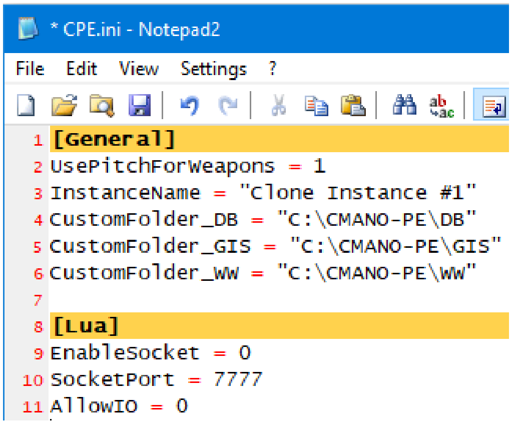


```
[General]
UsePitchForWeapons = 1
InstanceName = "Instance #1"

[Lua]
EnableSocket = 1
SocketPort = 7777
AllowIO = 0

```

Clone 1



```
[General]
UsePitchForWeapons = 1
InstanceName = "Clone Instance #1"
CustomFolder_DB = "C:\CMANO-PE\DB"
CustomFolder_GIS = "C:\CMANO-PE\GIS"
CustomFolder_WW = "C:\CMANO-PE\WW"

[Lua]
EnableSocket = 0
SocketPort = 7777
AllowIO = 0
```

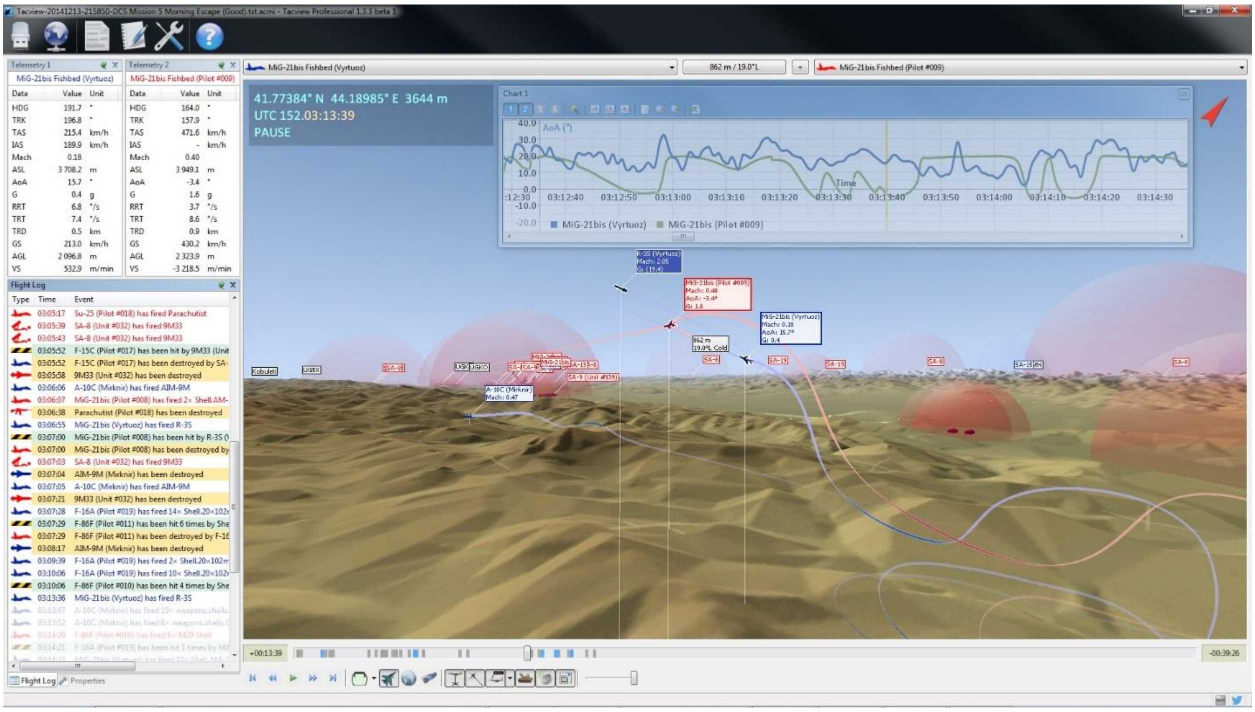
Once the clone instance is properly configured, we can delete its own \DB, \GIS and \WW folders entirely.

In this example, redirecting the data-hungry folders of the clone instance to the equivalent folders of the original installation has enabled us to shrink its storage footprint **from 10-15GB down to 600-700MB**. This makes it practical to host and run multiple instances of the application even in environments where storage space is at a premium (e.g. when running Command on a high-performance, low-capacity storage medium such as an SSD or RAM-drive). It also ensures that all running instances are basing their simulation run on the same read-only data (platform databases, terrain & environment, other GIS info etc.).

A caveat here: The Command PE installer, during the installation, stores to the registry the installation path. This information is subsequently used by follow-on installers, updaters etc. to perform maintenance or updates to the installation. Copies of the installation folder will not receive this maintenance. For this reason, it is good practice to "wipe and re-spawn" the additional copies of the original installation folder after updates, fixes etc. are applied. This holds true for either method described above.

Tacview visualization

(Available in PE & AE)


Tacview-20141213-215850-DCS Mission 5 Morning Escape (Good).txt.acmi - Tacview Professional 1.3.3 beta 1
MiG-21bis Fishbed (Vyrtuoz)
862 m / 19.0°L
MiG-21bis Fishbed (Pilot #009)
41.77384° N 44.18985° E 3644 m
UTC 152.03:13:39
PAUSE
Chart 1
AoA (°)
Time
MiG-21bis (Vyrtuoz)
MiG-21bis (Pilot #009)
Kobuleti
+00:13:39
-00:39:26
Flight Log
Properties

Telemetry 1 – MiG-21bis Fishbed (Vyrtuoz)

Data	Value	Unit
HDG	191.7	°
TRK	196.8	°
TAS	215.4	km/h
IAS	189.9	km/h
Mach	0.18	
ASL	3 708.2	m
AoA	15.7	°
G	0.4	g
RRT	6.8	°/s
TRT	7.4	°/s
TRD	0.5	km
GS	213.0	km/h
AGL	2 096.8	m
VS	532.9	m/min

Telemetry 2 – MiG-21bis Fishbed (Pilot #009)

Data	Value	Unit
HDG	164.0	°
TRK	157.9	°
TAS	471.6	km/h
IAS	-	km/h
Mach	0.40	
ASL	3 949.1	m
AoA	-3.4	°
G	1.6	g
RRT	3.7	°/s
TRT	8.6	°/s
TRD	0.9	km
GS	430.2	km/h
AGL	2 323.9	m
VS	-3 218.5	m/min

Flight Log

Type	Time	Event
	03:05:17	Su-25 (Pilot #018) has fired Parachutist
	03:05:39	SA-8 (Unit #032) has fired 9M33
	03:05:43	SA-8 (Unit #032) has fired 9M33
	03:05:52	F-15C (Pilot #017) has been hit by 9M33 (Unit
	03:05:52	F-15C (Pilot #017) has been destroyed by SA-
	03:05:58	9M33 (Unit #032) has been destroyed
	03:06:06	A-10C (Mirknir) has fired AIM-9M
	03:06:07	MiG-21bis (Pilot #008) has fired 2× Shell.AM-
	03:06:38	Parachutist (Pilot #018) has been destroyed
	03:06:55	MiG-21bis (Vyrtuoz) has fired R-3S
	03:07:00	MiG-21bis (Pilot #008) has been hit by R-3S (\
	03:07:00	MiG-21bis (Pilot #008) has been destroyed by
	03:07:03	SA-8 (Unit #032) has fired 9M33
	03:07:04	AIM-9M (Mirknir) has been destroyed
	03:07:05	A-10C (Mirknir) has fired AIM-9M
	03:07:21	9M33 (Unit #032) has been destroyed
	03:07:28	F-16A (Pilot #019) has fired 14× Shell.20×102r
	03:07:29	F-86F (Pilot #011) has been hit 6 times by She
	03:07:29	F-86F (Pilot #011) has been destroyed by F-16
	03:08:17	AIM-9M (Mirknir) has been destroyed
	03:09:39	F-16A (Pilot #019) has fired 2× Shell.20×102m
	03:10:06	F-16A (Pilot #019) has fired 10× Shell.20×102r
	03:10:06	F-86F (Pilot #010) has been hit 4 times by She
	03:13:36	MiG-21bis (Vyrtuoz) has fired R-3S
	03:13:47	A-10C (Mirknir) has fired 10× weapons.shells.
	03:13:52	A-10C (Mirknir) has fired 8× weapons.shells.(
	03:14:20	F-86F (Pilot #010) has fired 6× M20 Shell
	03:14:21	F-16A (Pilot #019) has been hit 3 times by M2

CPE offers the ability to export simulation data to Tacview, a popular 3D visualization tool (http://www.tacview.net/). This provides improved situational awareness of an engagement as well as more options for visual analysis (such as jumping into the viewpoint of a specific unit, stepping forwards and backwards in time, accelerating playback etc.).

Tacview supports two input modes: Reading from a pre-generated ACMI text file, or consuming a real-time data stream. Command PE supports both these modes.

Exporting to file

To configure exporting to ACMI text file, edit the file **EventExport.ini** and set the **ActiveExporter** property to “Tacview2x” (Tacview 1.x format support is deprecated but maintained for backwards-compatibility):

```
ActiveExporter = Tacview2x
```

(This can be combined with any other destination for which we want to export data, e.g. `ActiveExporter = Tacview2x, CSV`*)*

If performing an interactive analysis run, Command must be restarted in order for this change to take effect. If running a Monte-Carlo session, we can set Tacview as an export destination directly from the Monte-Carlo window.

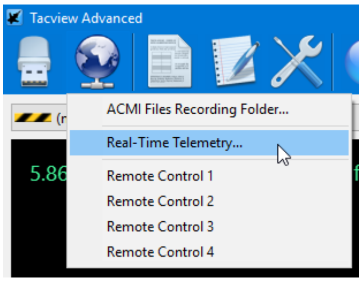
Once the scenario runs and the ACMI file is generated (located at **[top-level writable folder]\Tacview2x.acmi**), we can then open it with Tacview.

Real-time streaming

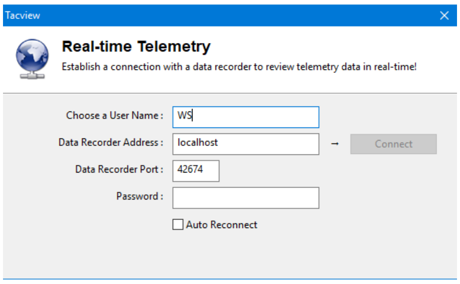
Setting up real-time streaming to Tacview is slightly more involved. First, the **ActiveExporter** property on **EventExport.ini** must be set to "TacviewRT":

```
ActiveExporter = TacviewRT
```

Then Command must be restarted, and the desired scenario loaded (but not yet started). Then start Tacview, and from its main menu select "Recording → Real-Time Telemetry":



The "Real-time Telemetry" window will appear, with several customizable options. Command listens on the default port (42762) and does not use a password for this data stream so no change is necessary. Just click on "Connect", and Tacview should connect to Command's stream socket and begin displaying Command's scenario environment.



Real-time Telemetry

Establish a connection with a data recorder to review telemetry data in real-time!

Choose a User Name : WS

Data Recorder Address : localhost

Data Recorder Port : 42674

Password :

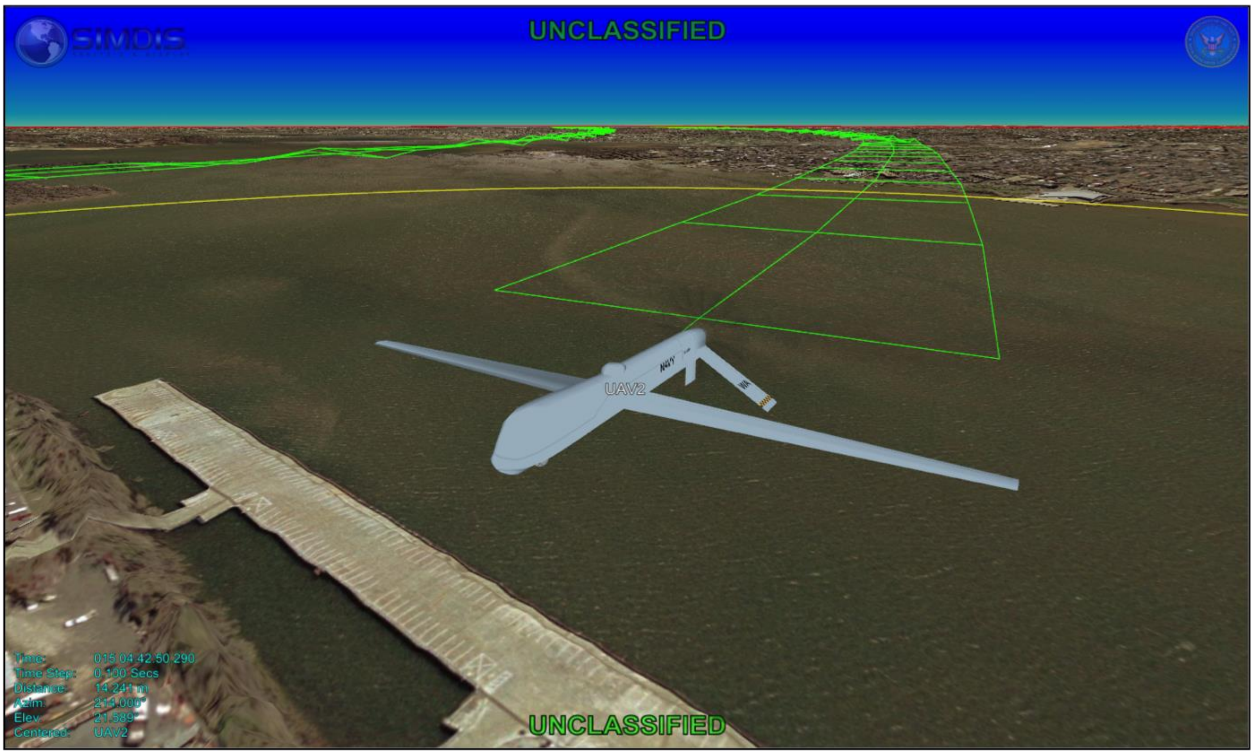
□ Auto Reconnect

Command can also stream to an instance of Tacview running on a different machine on the network. In this case, the "Data Recorder Address" field above will need to be filled in with the IP address of the machine running Command. (The CPE-running machine will also need to have its firewall configured to enable port 42674).

Export support for other 3D visualization tools, platforms & protocols such as CIGI, STK & Cesium can be added upon request.

SIMDIS visualization

(Available in PE & AE)



Time: 015 04:42:50.290
Time Step: 0.100 Secs
Distance: 14.241 m
Azim: 214.000°
Elev: 21.589°
Centered: UAV2

Another 3D visualization option for Command PE is SIMDIS (https://simdis.nrl.navy.mil/), an image-generator and analysis tool produced by the US Office of Naval Research. As with Tacview, SIMDIS provides rich options for visual demonstration & analysis together with complete time- and POV-location control. SIMDIS is currently arguably better suited than Tacview on visualization of naval (and especially underwater) engagements as it offers a wider variety of naval platform models out of the box, as well as importing a wide range of terrain data, including detailed bathymetric surfaces (something that Tacview currently lacks).

Similar to Tacview, the primary means of input is a raw text file in .ASI format. To configure exporting to the .ASI text file, edit the file **EventExport.ini** and set the **ActiveExporter** property to “SIMDIS”:

```
ActiveExporter = SIMDIS
```

(This can be combined with any other destination for which we want to export data, e.g. `ActiveExporter = SIMDIS, CSV`*)*

If performing an interactive analysis run, Command must be restarted in order for this change to take effect. If running a Monte-Carlo session, we can set SIMDIS as an export destination directly from the Monte-Carlo window.

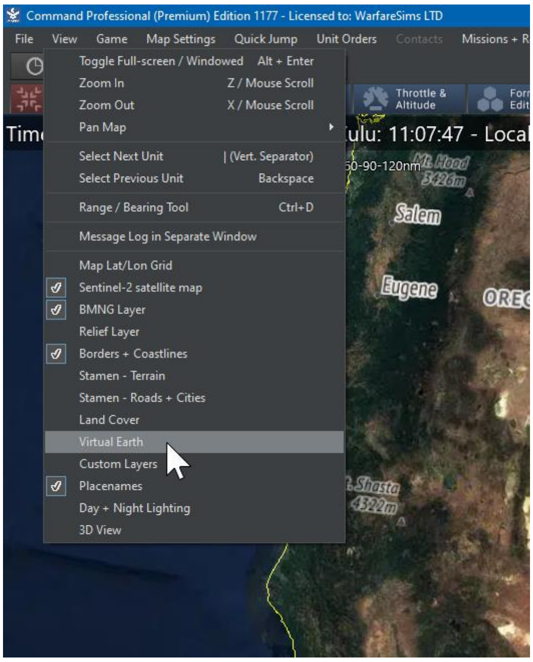
Once the scenario runs and the output file is generated (located at **[top-level writable folder]\SIMDIS.asi**), we can then open it with SIMDIS.

On-demand imagery

(Available in PE & AE)

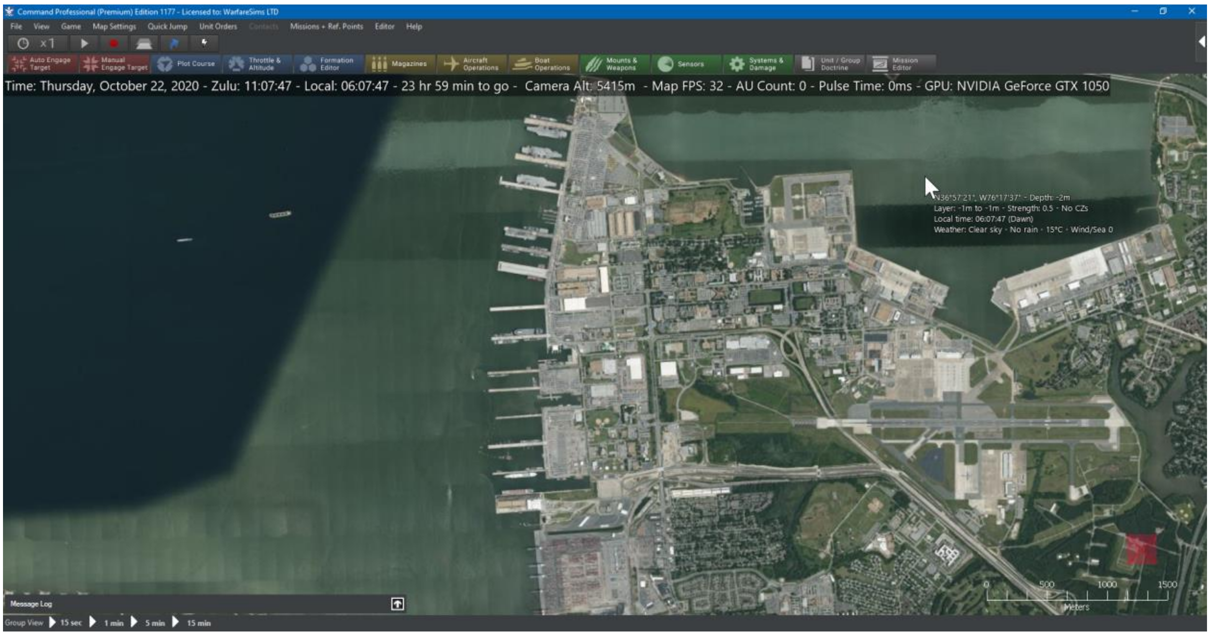
Command-PE offers additional map-layer options compared to the commercial CMO version. One of these additions is the ability to use on-demand overhead imagery from Microsoft's Bing Maps service.

To enable this, simply enable the "Virtual Earth" layer from Command's main menu → View:



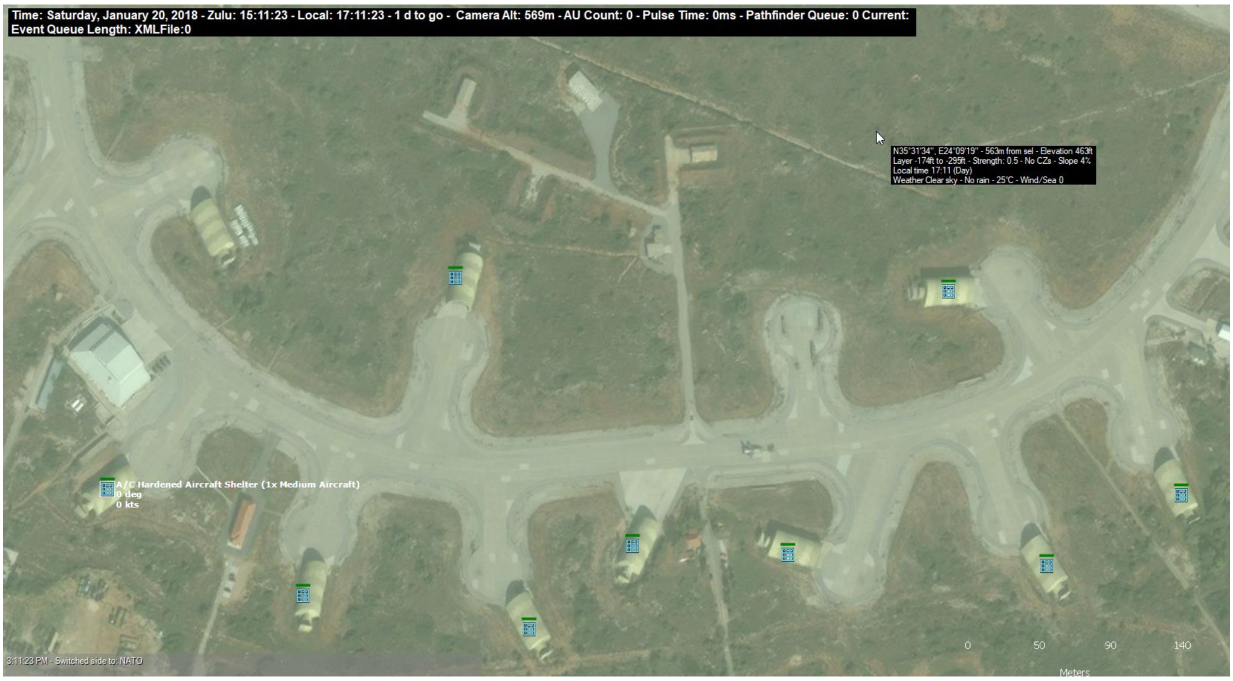
(Virtual Earth was the original name of Microsoft's overhead imagery service before it was renamed to Bing Maps).

It may take a few seconds, but the map view should be populated with imagery from Microsoft's BM servers:


Time: Thursday, October 22, 2020 - Zulu: 11:07:47 - Local: 06:07:47 - 23 hr 59 min to go - Camera Alt: 5415m - Map FPS: 32 - AU Count: 0 - Pulse Time: 0ms - GPU: NVIDIA GeForce GTX 1050

N36°57'21", W76°17'37" - Depth: -2m
Layer: -1m to -1m - Strength: 0.5 - No CZs
Local time: 06:07:47 (Dawn)
Weather: Clear sky - No rain - 15°C - Wind/Sea 0

This ability can be particularly useful when constructing installations that precisely mirror their real-counterparts in layout. In the example below, we are recreating Souda airbase (Crete), by literally placing hardened aircraft shelter (HAS) facilities right at the coordinates of their real-life analogues, with sub-meter precision:



Time: Saturday, January 20, 2018 - Zulu: 15:11:23 - Local: 17:11:23 - 1 d to go - Camera Alt: 569m - AU Count: 0 - Pulse Time: 0ms - Pathfinder Queue: 0 Current:
Event Queue Length: XMLFile:0

N35°31'34", E24°09'19" - 563m from sel - Elevation 463ft
Layer -174ft to -295ft - Strength: 0.5 - No CZs - Slope 4%
Local time 17:11 (Day)
Weather Clear sky - No rain - 25°C - Wind/Sea 0

A/C Hardened Aircraft Shelter (1x Medium Aircraft)
0 deg
0 kts

3:11:23 PM - Switched side to: NATO

The resulting complete installation can then be exported to an .inst file and re-used on any other scenario, just like in the commercial version of CMO.

Additional on-demand imagery sources can be integrated upon request.

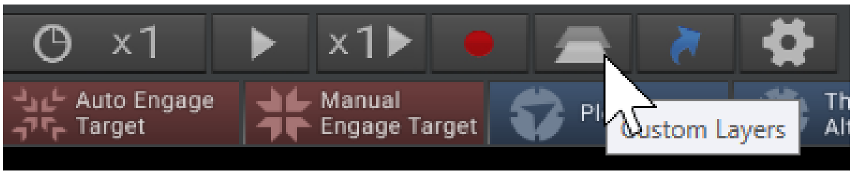
NOTE #1: *This feature works only if the computer running Command-PE has an active Internet connection.*

NOTE #2: *Legal useage of this feature requires that the user is covered by a licensing agreement with Microsoft.*

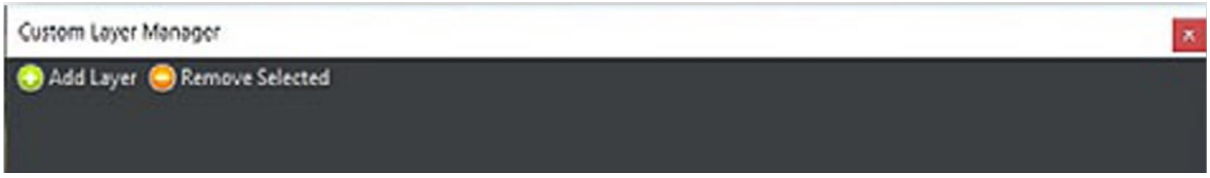
Custom Overlays 101

As in the commercial version of Command, CPE also allows using custom overlay imagery for specific areas. To add overlays to CPE, you must first use third-party GIS/mapping software to generate geo-referenced imagery files; Then you can import them to your scenario using an in-app interface

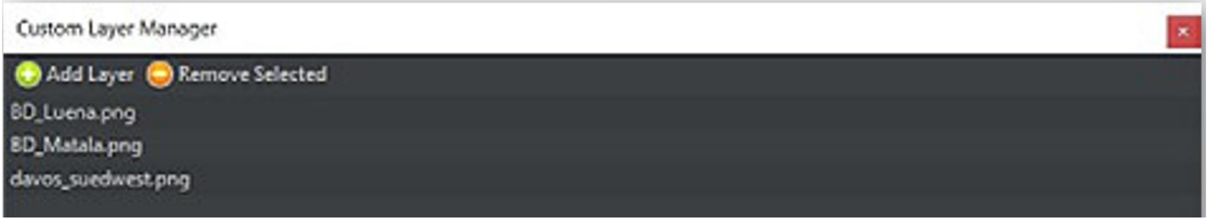
To import the generated georeferenced image into CPE, launch the application, create at least one side (or load a pre-existing scenario) and click the “Custom Overlay” button:



When the Custom Layer Manager appears press the Add Layer button. This will launch a standard Windows “Add File” dialog.



Choose the overlay file and allow the display time to load. Once loaded, you will see it in the bottom list. The most recent overlay will be highest on the list. To remove an overlay, simply select it in the Custom Layer Manager and press the Remove Selected Button.



BD_Luena.png
BD_Matala.png
davos_suedwest.png

DATA & MODEL EDITING

Database Editor

(Available in PE only)

Command-PE includes a comprehensive DB editor that allows complete customization of every aspect of the database. For detailed information on the layout of the database and instructions on how to edit it, please refer to the dedicated DB-editor manual.

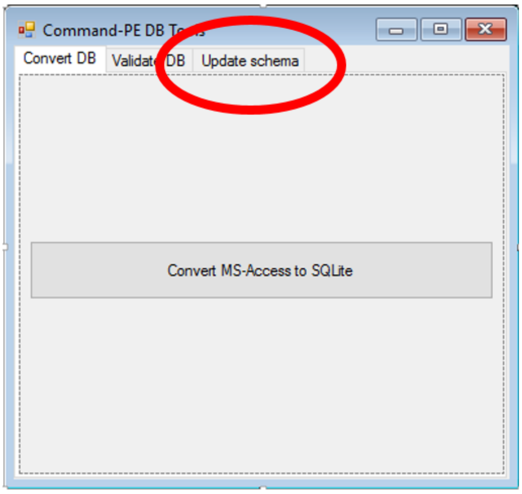
DBTools

Beginning from v1.14, the DB-editing tools and documentation are included in their own zipped file and can be extracted and used anywhere on the file system. Chief among them, the DBTools.exe program is used for a number of functions:

- Database conversion
- Database validation prior to conversion (ensures that no logical errors are present in the database, for example a weapon is present in a platform without the necessary datalink or illuminator for its guidance)
- One-click updating of the database schema to the current version (migrating to the current schema is essential for taking advantage of new simulation features).

Migrating existing DBs to new schema

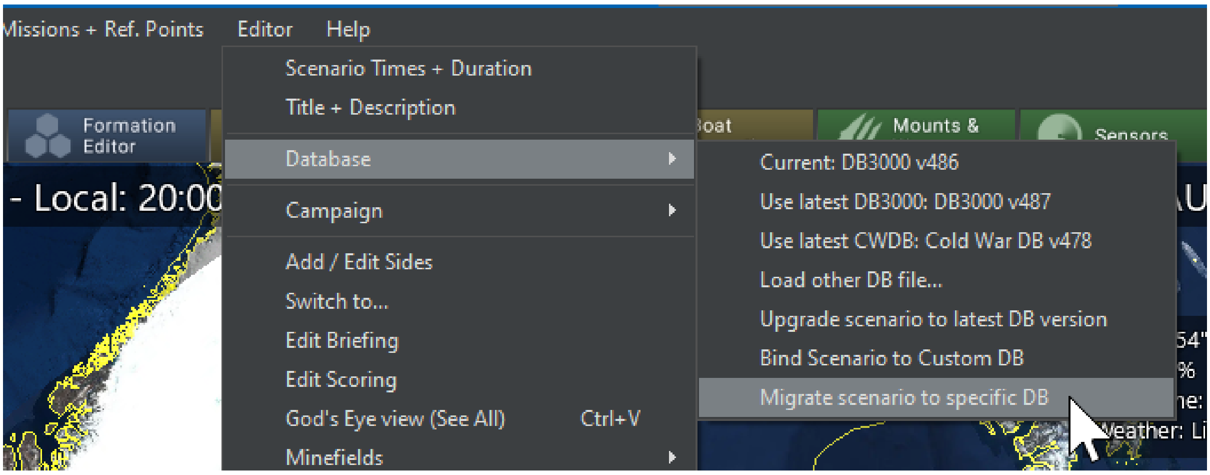
Using the DBTools app, click on the tab "Update schema":



Select the desired database to update, and click the "Update database to current schema" button.

Migrating a scenario to a new custom database

After you have created a new custom database (or new version of an existing one), a common use case is to migrate an existing scenario to it. Command-PE makes this easy with the addition of a tailored function on the "Database" submenu of the Editor menu:



Clicking on “Migrate scenario to specific DB” will bring up a dialog that allows you to select the database file in question. After this, the standard “migrate scenario to DB” window appears and presents the various options for the migration.

Lua I/O

(Available in PE – Optional in AE)

Lua has a number of built-in I/O features for interaction with local storage, network etc. The commercial version of CMO disables these features for security reasons (being able, for example, to read an external text file which can then execute arbitrary code against local system resources could pose a potential attack vector). CPE also disables them by default, but allows optionally enabling them for added functionality. The configuration for this feature is stored on **[top-level writable folder]\Config\CPE.ini**:

[Lua]
AllowIO = 0 ← 0 to disable, 1 to enable.

Command must be restarted in order for changes in these settings to take effect.

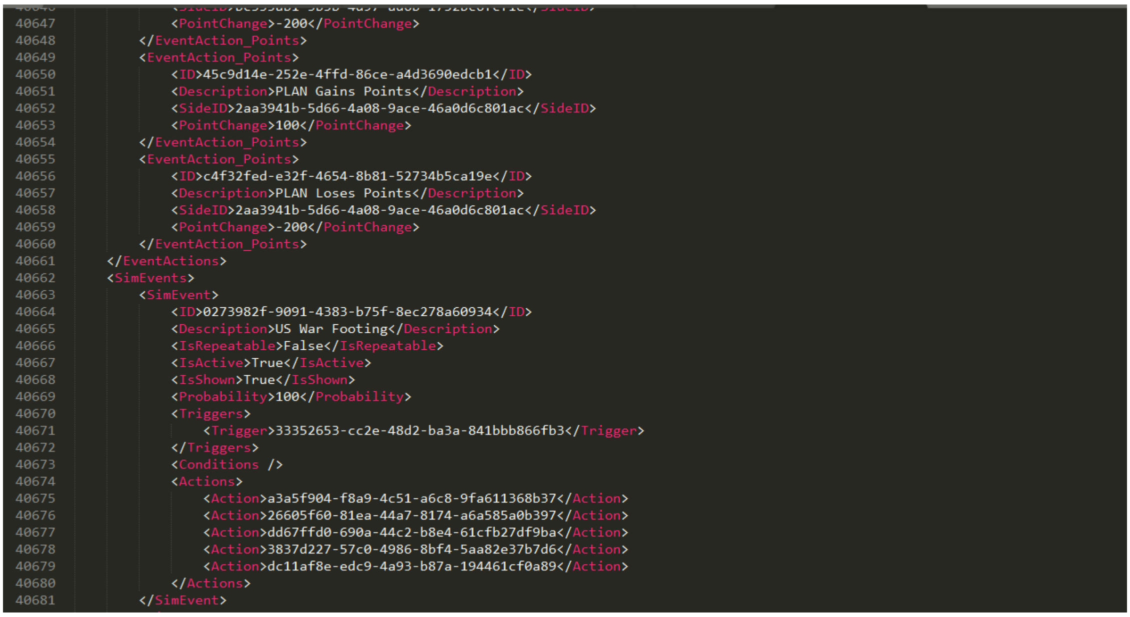
XML import/export

(Available in PE only)

Command-PE offers the ability to export its entire scenario object graph (all its internal objects like sides, missions, objectives, events, units, unit components etc.) to an XML file. This can then be parsed, analysed etc. The XML document can be modified and re-imported into Command, absorbing all the changes affected.

To export the scenario state, from the main menu select **Editor → Scenario Migration → Export to file**.

This is an example of the XML output, after being formatted for legibility:



```
            <PointChange>-200</PointChange>
        </EventAction_Points>
        <EventAction_Points>
            <ID>45c9d14e-252e-4ffd-86ce-a4d3690edcb1</ID>
            <Description>PLAN Gains Points</Description>
            <SideID>2aa3941b-5d66-4a08-9ace-46a0d6c801ac</SideID>
            <PointChange>100</PointChange>
        </EventAction_Points>
        <EventAction_Points>
            <ID>c4f32fed-e32f-4654-8b81-52734b5ca19e</ID>
            <Description>PLAN Loses Points</Description>
            <SideID>2aa3941b-5d66-4a08-9ace-46a0d6c801ac</SideID>
            <PointChange>-200</PointChange>
        </EventAction_Points>
    </EventActions>
    <SimEvents>
        <SimEvent>
            <ID>0273982f-9091-4383-b75f-8ec278a60934</ID>
            <Description>US War Footing</Description>
            <IsRepeatable>False</IsRepeatable>
            <IsActive>True</IsActive>
            <IsShown>True</IsShown>
            <Probability>100</Probability>
            <Triggers>
                <Trigger>33352653-cc2e-48d2-ba3a-841bbb866fb3</Trigger>
            </Triggers>
            <Conditions />
            <Actions>
                <Action>a3a5f904-f8a9-4c51-a6c8-9fa611368b37</Action>
                <Action>26605f60-81ea-44a7-8174-a6a585a0b397</Action>
                <Action>dd67ffd0-690a-44c2-b8e4-61cfb27df9ba</Action>
                <Action>3837d227-57c0-4986-8bf4-5aa82e37b7d6</Action>
                <Action>dc11af8e-edc9-4a93-b87a-194461cf0a89</Action>
            </Actions>
        </SimEvent>
```

After making any changes, the XML can be re-imported via **Editor → Scenario Migration → Import from file**.

This is an additional/alternative way to the Lua API to make changes in the simulation state. It is arguably more powerful, because it allows altering every facet of the sim state whereas the Lua API is precise/"strict" in what allows to change. But this freedom and power also come with risks - there are absolutely no integrity or sanity checks when directly editing the scenario graph, contrary to the careful checks that accompany Lua calls.

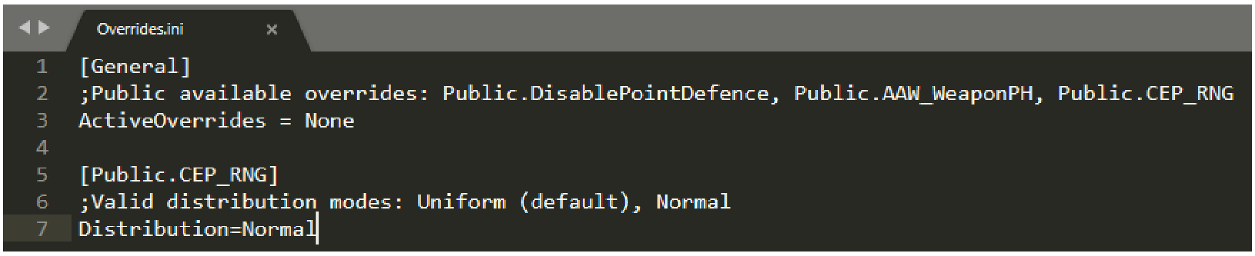
Mechanics Overrides

(Available in PE Premium only)

A common request from users of Command-PE is the ability to substitute Command's own internal models with their own. CPE offers this ability through Mechanics Overrides.

Many of CPE's internal simulation sequences, such as the endgame calculations prior to a guided weapon impact, or sensor detections, or AI decisions etc., are explicitly architected to allow for being extended/overridden by custom methods. These are typically developed on a bespoke basis in order to make use of data in the customer's possession.

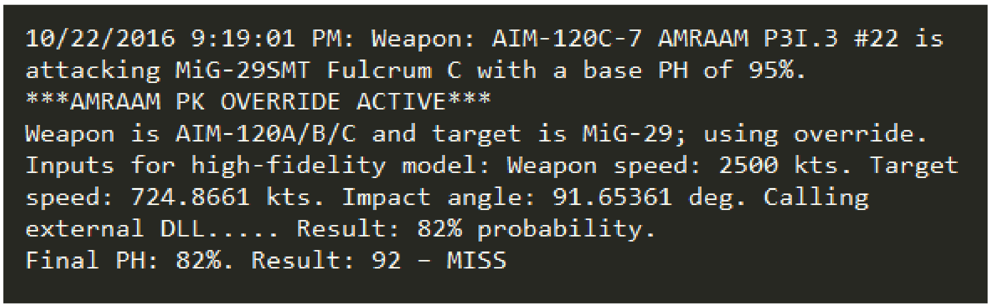
These overrides are also configurable on a per-session basis. The configuration file for them is **Overrides.ini**, located in the **[top-level writable folder]\Config** folder:



```
[General]
;Public available overrides: Public.DisablePointDefence, Public.AAW_WeaponPH, Public.CEP_RNG
ActiveOverrides = None

[Public.CEP_RNG]
;Valid distribution modes: Uniform (default), Normal
Distribution=Normal
```

Here is an example of a finely-tuned override. A customer happened to have in their possession reams of data on the performance of the AIM-120C against the baseline MiG-29S, and wished to use this data for this specific weapon-target combination, while still relying on Command's models for all other engagements. So, the missile endgame pipeline was extended to do precisely that:



```
10/22/2016 9:19:01 PM: Weapon: AIM-120C-7 AMRAAM P3I.3 #22 is
attacking MiG-29SMT Fulcrum C with a base PH of 95%.
***AMRAAM PK OVERRIDE ACTIVE***
Weapon is AIM-120A/B/C and target is MiG-29; using override.
Inputs for high-fidelity model: Weapon speed: 2500 kts. Target
speed: 724.8661 kts. Impact angle: 91.65361 deg. Calling
external DLL..... Result: 82% probability.
Final PH: 82%. Result: 92 - MISS
```

Because the models (and frequently also data) that the customer wishes to use in CPE are often classified, the override mechanism is deliberately very flexible. The process can be as simple as reading a local text file (pre-generated data tables), reading from a database, or alternatively calling a remote endpoint and providing it with all the parameters so that the actual "business logic" is executed on the remote system, the result then being returned to Command.

Sections of the simulation mechanics that can be extended/replaced through overrides include:

- Weapon endgame/impact calculations
- Sensor detections
- Kinematics
- Tactical AI decision making
- Operational-level AI actions

Available public overrides

The default CPE installation contains a number of built-in, public-visible overrides. Additional customer-private overrides are also embedded in the code, and documentation for them is provided separately to respective customers.

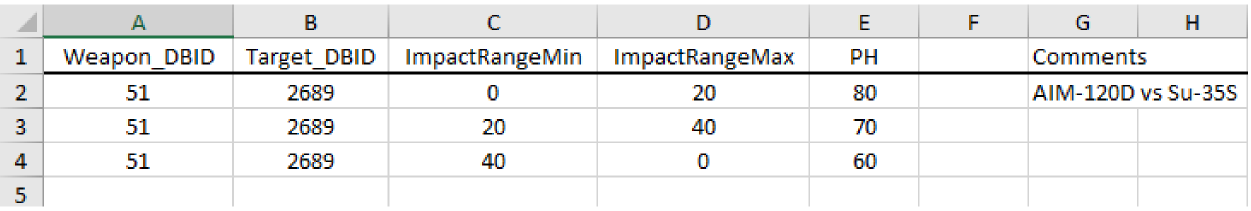
As the Mechanics Overrides framework is deliberately architected for great flexibility, additional overrides can be implemented and added per customer priorities.

The publicly available overrides in v2.2 are:

- **Public.DisablePointDefence** : As the name implies, this override disables the entire point-defence sequence that transpires when a weapon is about to perform its endgame attack on a unit. The point-defence step includes decoys, DECM and expendables (e.g. chaff/flares) attempting to defeat the incoming weapon. Disabling this sequence means that the weapon will definitely reach the final attempt to impact the unit being engaged. This can be useful, for example, to isolate the subsequent endgame calculation or apply a pre-defined hit estimate without concern for point-defence systems interfering.

- **Public.AAW_WeaponPH** : This override replaces the built-in terminal engagement model for guided AAW weapons, and uses pre-defined probability values instead. The values are read from an Excel file, located at:
  [top-level writable folder]\Resources\Overrides\Public.AAW_WeaponPH\Data.xls .

  The file contains some sample representative data:

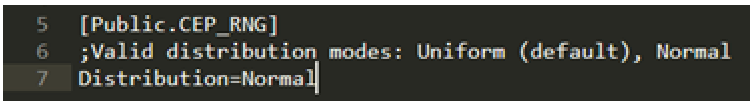


	A	B	C	D	E	F	G	H
1	Weapon_DBID	Target_DBID	ImpactRangeMin	ImpactRangeMax	PH		Comments	
2	51	2689	0	20	80		AIM-120D vs Su-35S	
3	51	2689	20	40	70			
4	51	2689	40	0	60			
5								

  To populate with additional data, the respective weapon and target database record ID s must be provided, as well as the range bracket for which the PH figure is valid. As in the provided sample, a weapon-target pairing can be either a single record (if it is assumed that the PH is constant regardless of range) or multiple records, each using the same weapon-target pairing but corresponding to different range brackets.

  During runtime, the sim code checks if the characteristics of the engagement (weapon DBID, target DBID, and range from launch point) match any of the records on the Excel sheet, and if they do, skips the built-in calculations and instead applies these values.

- **Public.CEP_RNG :** This override replaces the built-in random number generator (RNG) used for CEP-related weapon impact calculations with an alternative one.
  This override is configurable through its entry on Overrides.ini:



```
[Public.CEP_RNG]
;Valid distribution modes: Uniform (default), Normal
Distribution=Normal
```

Whereas the default RNG uses uniform distribution of values, the alternative RNG employs normal distribution, and it is assumed that the impact CEP equals 1.1774 x sigma [standard deviation]. *(This is based on calculations on D.H. Wagner's "Naval Operations Analysis")*.

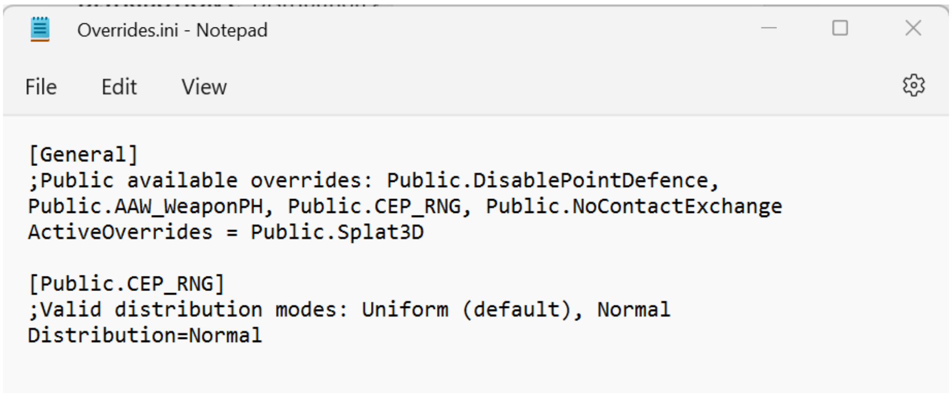
The practical result of using the normal-based RNG is that the concentration of missed impacts is closer to the targeted unit than in the case of uniform distribution (i.e. there are a lot more "near-misses" than far-wide ones). This can be an important factor for area weapons, proximity blast damage calculations etc.

- **Public.NoContactExchange :** By default, allied sides automatically exchange their contact reports & updates. When this override is active, allied sides no longer exchange contact information; instead every such information must be routed manually. This can be useful in explicitly modeling communications between different friendly sides.

- **Public.Splat3D:** This (new in v2.2) override allows customizing the signature type (usually radar, but all signature types are possible) of any aircraft in the simulation, on a per-DBID basis.

  The override is triggered by the following two necessary conditions:

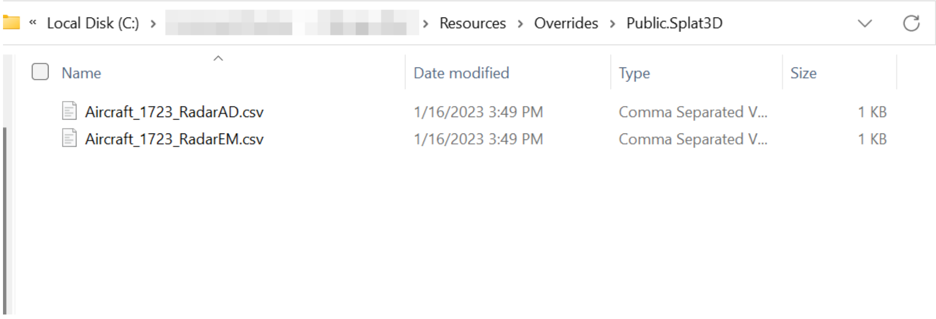
  Public.Splat3D added as an active override in the Overrides.ini file, as in this example:



```
[General]
;Public available overrides: Public.DisablePointDefence,
Public.AAW_WeaponPH, Public.CEP_RNG, Public.NoContactExchange
ActiveOverrides = Public.Splat3D

[Public.CEP_RNG]
;Valid distribution modes: Uniform (default), Normal
Distribution=Normal
```

And the existence of a correctly named csv in the [Writable folder]\Resources\Overrides\Public.Splat3D directory, as in this example:


« Local Disk (C:) › [redacted] › Resources › Overrides › Public.Splat3D

Name	Date modified	Type	Size
Aircraft_1723_RadarAD.csv	1/16/2023 3:49 PM	Comma Separated V...	1 KB
Aircraft_1723_RadarEM.csv	1/16/2023 3:49 PM	Comma Separated V...	1 KB

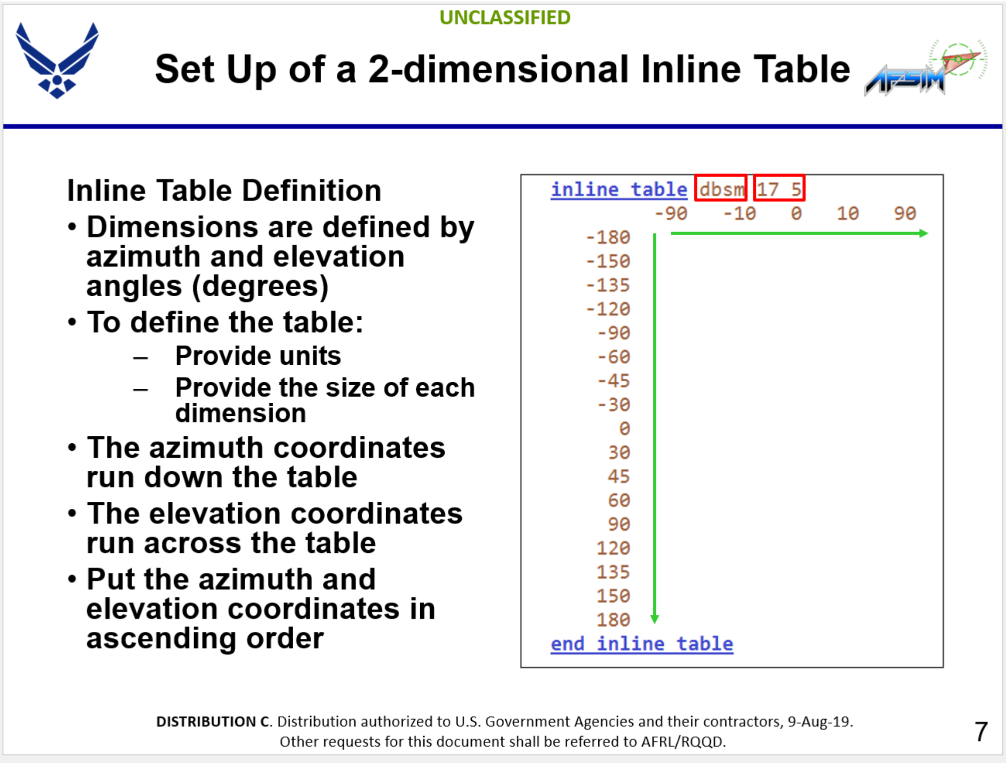
The name of the csv file is very important. There are 3 components to a Splat3D override CSV filename:

Unit type: Currently only Aircraft is supported.

DBID: This is the DBID of the aircraft in the Command database.

Band: RadarAD or RadarEM. Command breaks down the radar spectrum into A-D radar bands and E-M radar bands for RCS calculations.

For the 3D RCS file format, we deliberately follow the format used in similar AFSIM signature files:



UNCLASSIFIED

Set Up of a 2-dimensional Inline Table

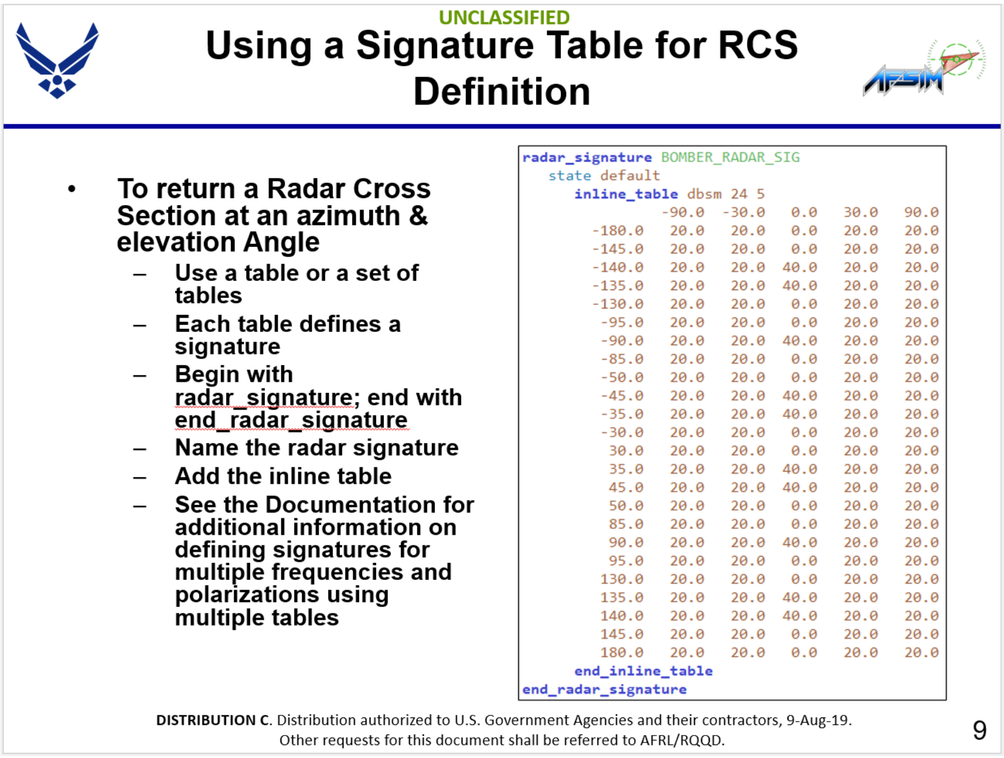
Inline Table Definition

- Dimensions are defined by azimuth and elevation angles (degrees)
- To define the table:
  - Provide units
  - Provide the size of each dimension
- The azimuth coordinates run down the table
- The elevation coordinates run across the table
- Put the azimuth and elevation coordinates in ascending order

```
inline_table dbsm 17 5
            -90   -10   0   10   90
    -180
    -150
    -135
    -120
     -90
     -60
     -45
     -30
       0
      30
      45
      60
      90
     120
     135
     150
     180
end_inline_table
```

DISTRIBUTION C. Distribution authorized to U.S. Government Agencies and their contractors, 9-Aug-19.
Other requests for this document shall be referred to AFRL/RQQD.

7



UNCLASSIFIED

Using a Signature Table for RCS Definition

- To return a Radar Cross Section at an azimuth & elevation Angle
  - Use a table or a set of tables
  - Each table defines a signature
  - Begin with radar_signature; end with end_radar_signature
  - Name the radar signature
  - Add the inline table
  - See the Documentation for additional information on defining signatures for multiple frequencies and polarizations using multiple tables

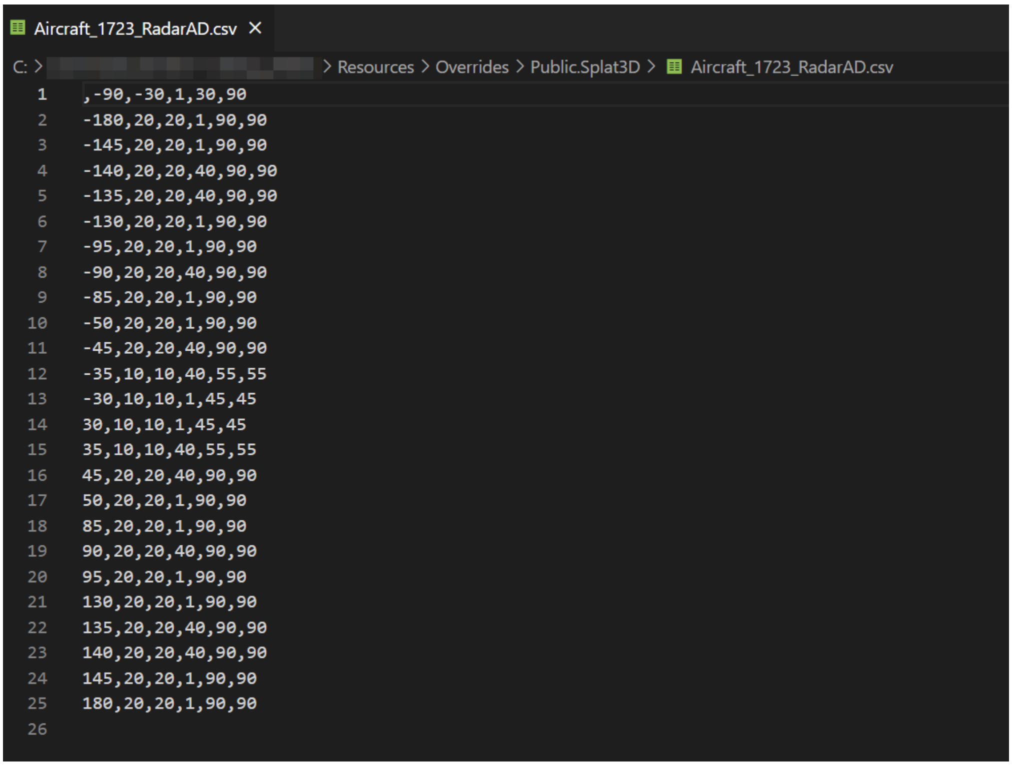
```
radar_signature BOMBER_RADAR_SIG
   state default
      inline_table dbsm 24 5
                -90.0  -30.0   0.0   30.0   90.0
       -180.0   20.0   20.0    0.0   20.0   20.0
       -145.0   20.0   20.0    0.0   20.0   20.0
       -140.0   20.0   20.0   40.0   20.0   20.0
       -135.0   20.0   20.0   40.0   20.0   20.0
       -130.0   20.0   20.0    0.0   20.0   20.0
        -95.0   20.0   20.0    0.0   20.0   20.0
        -90.0   20.0   20.0   40.0   20.0   20.0
        -85.0   20.0   20.0    0.0   20.0   20.0
        -50.0   20.0   20.0    0.0   20.0   20.0
        -45.0   20.0   20.0   40.0   20.0   20.0
        -35.0   20.0   20.0   40.0   20.0   20.0
        -30.0   20.0   20.0    0.0   20.0   20.0
         30.0   20.0   20.0    0.0   20.0   20.0
         35.0   20.0   20.0   40.0   20.0   20.0
         45.0   20.0   20.0   40.0   20.0   20.0
         50.0   20.0   20.0    0.0   20.0   20.0
         85.0   20.0   20.0    0.0   20.0   20.0
         90.0   20.0   20.0   40.0   20.0   20.0
         95.0   20.0   20.0    0.0   20.0   20.0
        130.0   20.0   20.0    0.0   20.0   20.0
        135.0   20.0   20.0   40.0   20.0   20.0
        140.0   20.0   20.0   40.0   20.0   20.0
        145.0   20.0   20.0    0.0   20.0   20.0
        180.0   20.0   20.0    0.0   20.0   20.0
      end_inline_table
end_radar_signature
```

DISTRIBUTION C. Distribution authorized to U.S. Government Agencies and their contractors, 9-Aug-19.
Other requests for this document shall be referred to AFRL/RQQD.

9

The first row and column describe the coordinate of the RCS data value.

The RCS data value is in dbsm. Floating point values are supported.

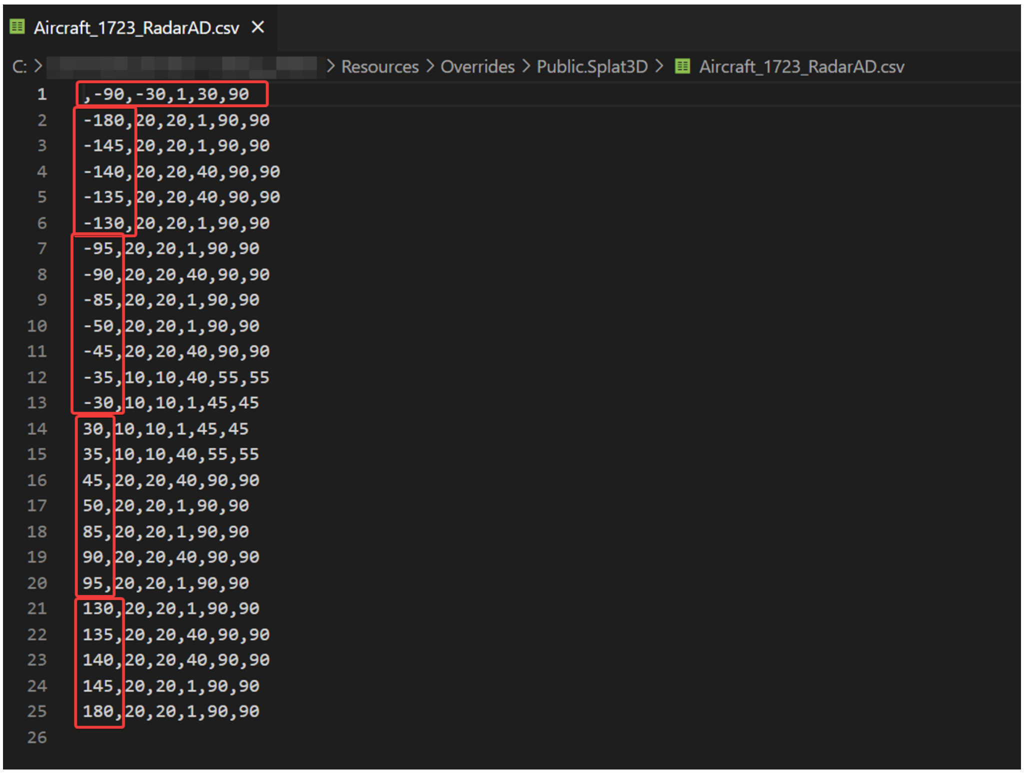

Aircraft_1723_RadarAD.csv ×
C: > [redacted] > Resources > Overrides > Public.Splat3D > Aircraft_1723_RadarAD.csv

```
,-90,-30,1,30,90
-180,20,20,1,90,90
-145,20,20,1,90,90
-140,20,20,40,90,90
-135,20,20,40,90,90
-130,20,20,1,90,90
-95,20,20,1,90,90
-90,20,20,40,90,90
-85,20,20,1,90,90
-50,20,20,1,90,90
-45,20,20,40,90,90
-35,10,10,40,55,55
-30,10,10,1,45,45
30,10,10,1,45,45
35,10,10,40,55,55
45,20,20,40,90,90
50,20,20,1,90,90
85,20,20,1,90,90
90,20,20,40,90,90
95,20,20,1,90,90
130,20,20,1,90,90
135,20,20,40,90,90
140,20,20,40,90,90
145,20,20,1,90,90
180,20,20,1,90,90

```

The first row denotes the pitch (vertical elevation) of the datavalue.

The first column denotes the bearing (horizontal azimuth) of the RCS datavalue:


Aircraft_1723_RadarAD.csv ×
C: > [redacted] > Resources > Overrides > Public.Splat3D > Aircraft_1723_RadarAD.csv

```
,-90,-30,1,30,90
-180,20,20,1,90,90
-145,20,20,1,90,90
-140,20,20,40,90,90
-135,20,20,40,90,90
-130,20,20,1,90,90
-95,20,20,1,90,90
-90,20,20,40,90,90
-85,20,20,1,90,90
-50,20,20,1,90,90
-45,20,20,40,90,90
-35,10,10,40,55,55
-30,10,10,1,45,45
30,10,10,1,45,45
35,10,10,40,55,55
45,20,20,40,90,90
50,20,20,1,90,90
85,20,20,1,90,90
90,20,20,40,90,90
95,20,20,1,90,90
130,20,20,1,90,90
135,20,20,40,90,90
140,20,20,40,90,90
145,20,20,1,90,90
180,20,20,1,90,90

```

For simplicity, the logic currently followed for sampling intermediate values is “snap to nearest-neighbor”. So for example if the RCS value at 150 degrees azimuth is X and the value at 160 degrees is Y, a value query at 153 degrees will return X while a query at 158 degrees will return Y.

Querying custom values through the Lua API

In order to support testing/debugging of the Splat3D override, a new method has been added to the Lua API, which can query the RCS of a target unit, from the direction of a sensor unit in the specified band:

```
Tool_QueryRCS{sensorunitname='Unit B', targetunitname='Unit A',
SIGNATURETYPE='Radar_A_D'}
```

The signature type can be any of:

```
HullSonar_PassiveOnly_VLF
HullSonar_PassiveOnly_LF
HullSonar_PassiveOnly_MF
HullSonar_PassiveOnly_HF
ActiveSonar
Visual_Detect
Visual_ID
IR_Detect
IR_ID
Radar_A_D
Radar_E_M
```

Although developed as a test tool for the Splat3D override, this new Lua method can also be used as a general signature query for any platform, whether the override applies to them or not.

Lua Event Hooks

(Available in PE Premium only)

Code-side mechanics overrides are typically implemented in one of two ways: Oftentimes the end customer full shares the model & data with the CPE development team, and the information is fully absorbed into the redistributed CPE release. In other cases, the customer provides “endpoints” (a text file, a compiled DLL, a web service, a database server etc.) that encapsulate/screen the confidential/proprietary model and the dev team then adds extension “stubs” in the sim code that reach out to these endpoints in order to fulfill the desired integration.

A common attribute of such integrations is that the CPE development team needs to become aware of the customer needs and be involved, to a greater or lesser extent. This is obviously undesirable in cases where even the mere fact of the override desire is confidential (i.e. when a very sensitive new technology or model is desired to be tested in Command’s virtual environment). Lua event-hooks offer a new alternative for such cases.

Event hooks provide a new mechanism for selectively overriding or extending built-in simulation steps. Compared to existing options for mechanics overrides, they provide greater flexibility and complete isolation from the CPE development team (i.e.. Overrides can be developed without any knowledge / involvement / assistance from the dev team).

The development team has added, at specific simulation events, hooks that can be extended through Lua scripting to either provide additional functionality (e.g. extended or special logging for the event taking place) or even the ability to completely override the event about to take place (e.g. replace the weapon endgame calculations). By making these overrides wholly script-based, end users can develop the override logics at their own pace and without any assistance from the CPE dev team.

NOTE: *Some of these events, like unit movement, occur with very high frequency within the simulation. This, combined with the fact that the current Lua environment is strictly single-threaded, means that enabling high-frequency event hooks can result in a dramatic drop in sim performance in any sizable scenario. Please use in moderation!*

As of CPE v2.2.3, the following Lua event hooks are available:

- **WeaponFired** – invoked after a weapon has been fired
- **WeaponImpactsBefore** – invoked before a weapon impacts, with an opportunity for override
- **WeaponImpactsAfter** – invoked after a weapon has impacted
- **SensorCheckBefore** – invoked before a sensor detection check occurs, with an opportunity to override
- **SensorCheckAfter** – invoked after a sensor detection check has occurred, with an opportunity to get a complete report
- **ContactUpdateAfter** – invoked after a new contact has been detected or data on a known contact has been updated
- **ContactUpdateFromOffGridAfter** – invoked after an out-of-communication unit re-establishes communication and shares its contacts.
- **UnitMovesBefore** – invoked before a unit executes its movement action, with an opportunity for override

- **UnitMovesAfter** – invoked after a unit executes its movement action; it cannot override the built-in movement logics, but can supplement them (e.g. implement additional movement due to wind or sea drift)
- **OnStatusChange** – invoked after a unit's air-ops/docking-ops status changes, e.g. an aircraft takes off or a ship docks to a pier.

Let us walk through examples of how we can use the hooks to add custom functionality.

This is a simple script that provides diagnostic information when a weapon is fired:

```
function WeaponFired(target, weapon)
        print('Executing ....')
        print(target)
        print(Weapon)
        local Message = "Missile "..weapon.name.. " fired at " .. target.name
        ScenEdit_MsgBox(Message, 1)
        return {true,"okay"}
end
```

Notice the function signature; this is precisely mapped to the in-sim code and thus must remain unchanged in any custom script in order to correctly communicate with the running simulation.

This function can be "registered" with the running scenario through any number of ways: It can be executed from the built-in "Lua console", or executed as a special action, or executed as part of a normal Lua script (e.g. driven by a built-in simulation event or fed through the external TCP/IP connection. Once registered, the function remains active throughout the execution of the scenario.

Here is an example function for the "Weapon impacts – before" hook:

```
function WeaponImpactsBefore(weapon, target)
        print('Executing ....')
        print(target)
        print(Weapon)
        local Message = "Missile "..weapon.name.. " about to hit "
        if target ~= nil then
                Message = Message .. target.name else Message = Message .. 'nothing'
        end
        ScenEdit_MsgBox(Message, 1)
        return nil
end
```

In this case, by returning "nil" to the simulation, we are signaling that this function is not intended to override the built-in weapon impact mechanics. If, however, we returned any other object or value, we would be signaling that we are handling the impact step within our script. In this way, we can entirely override/replace the internal weapon impact calculations.

Here is an example script for the “Weapon impacts – after” hook:

```
function WeaponImpactsAfter(weapon, target, result)
        print('Executing ....')
        print(target)
        print(Weapon)
        local Message = "Missile "..weapon.name
        if result == nil then Message = Message .. " impacts "
        elseif result == true then Message = Message .. " hits "
        else  Message = Message .. " misses "
        end
        if target ~= nil then
                Message = Message .. target.name
        else
                Message = Message .. 'nothing'
        end
        ScenEdit_MsgBox(Message, 1)
        return nil
end
```

Like the “Weapon is fired” hook, this one is also “one way”, ie. no override of the built-in method is available at the moment.

Finally, an example of a sensor detection check override. In this case we are wiring the "Sensor check – Before" event.

```
function SensorCheckBefore(theSensor, theSensorParent, theTarget)
   local Message = "Sensor: " ..theSensor.name.. " about to attempt detection on: "
..theTarget.name
   print(Message)
   local precise = theSensor:IsPreciseCheck(theSensorParent, theTarget)
   print(precise)
   if precise == true then
      return (precise)
   else
   --build AOU
   local AOU = {}
   AOU = World_GetCircleFromPoint({
         latitude=theTarget.latitude,
         longitude=theTarget.longitude,
         radius=50,
         numpoints = 72})
   print('imprecise with AOU')
   print(AOU)
   return true, AOU
   end
end
```

Note that we are handling precise and imprecise detections different. If a detection is made with a precise sensor, we just return a "true" value and the built-in sensor pipeline handles the rest. If a detection is made using an imprecise sensor (e.g. ESM or passive sonar) we can pass back a list of coordinates that represent the "area of uncertainty" (AOU) of the detection made.

NOTE: *Sample hook scripts for all available event hooks are located on **[Data Folder]\Resources\Lua Hooks**.*

Additional simulation events are planned to be extensible through the Lua event hook framework, in order to offer further oppoirtunities for extensibility. Potential candidates may include:

- AI/OODA decisions (evaluate targets & threats, determine primary target & threat, evaluate own operational status, assign weapons etc.)
- Logistics
- Environmental effects

INTEROPERABILITY

DIS support

(Available in PE Premium only)

Beginning from v1.14, CPE supports the Distributed Interactive Simulation (DIS) protocol for enhanced interoperability with other simulation products. Protocol version compliance is mostly v6 (1998) with some additions from v7 (2012) to implement some of the most recent DIS features like underwater emissions.

NOTE: *DIS connectivity is currently supported only for scenarios using the DB3000 database and its derivatives. CWDB scenarios are not supported at the moment.*

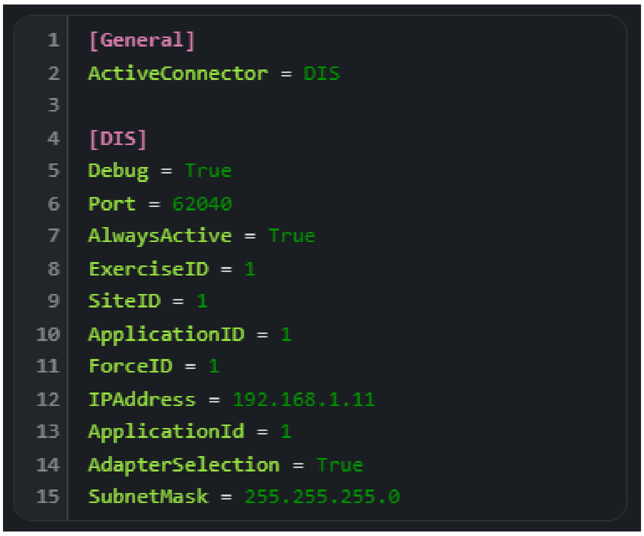
The following PDU types are currently supported:

- Entity state
- Weapon fire
- Weapon detonation
- Start/Resume & Stop/Freeze (receive-only)
- Underwater acoustic emission (passive sonar only)

Command can participate in a DIS session either in a completely "ad hoc" fashion (i.e. join or start a session by transmitting PDUs immediately on startup) without regard for time synchronization, or cooperate with a simulation-controller in an organized manner (do not start transmitting and receiving until it receives a proper Start/Resume PDU). The latter is useful for sessions where time-coordination is essential.

Configuration

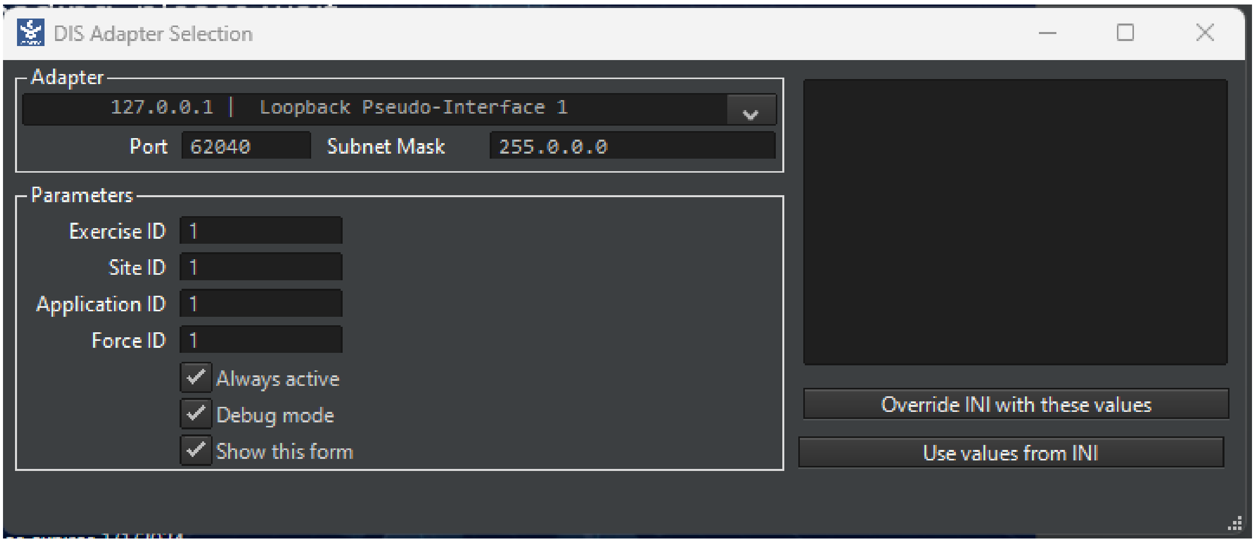
DIS settings are controlled by editing the configuration file **SimConnect.ini**, located in the **[top-level writable folder]\Config** folder:



```
[General]
ActiveConnector = DIS

[DIS]
Debug = True
Port = 62040
AlwaysActive = True
ExerciseID = 1
SiteID = 1
ApplicationID = 1
ForceID = 1
IPAddress = 192.168.1.11
ApplicationId = 1
AdapterSelection = True
SubnetMask = 255.255.255.0
```

Upon startup, if DIS has been enabled then a configuration window appears, which enables confirming or tweaking the DIS connection options:



127.0.0.1 | Loopback Pseudo-Interface 1

Port 62040 Subnet Mask 255.0.0.0

Exercise ID 1

Site ID 1

Application ID 1

Force ID 1

☑ Always active

☑ Debug mode

☑ Show this form

Through this window, users can select the desired network adapter to use, specify the desired port and subnet mask (this can help solve issues in complex network setups) as well as tweak and confirm the DIS-specific options like exercise, force, site and applications IDs.

In order to activate DIS, the “ActiveConnector” property in the “General” section must be set to “DIS”. (The remote-entity framework developed to enable DIS support is structured in such a way as to enable additional connectors in future updates, such as HLA, CIGI, AIS, ADS-B etc.)

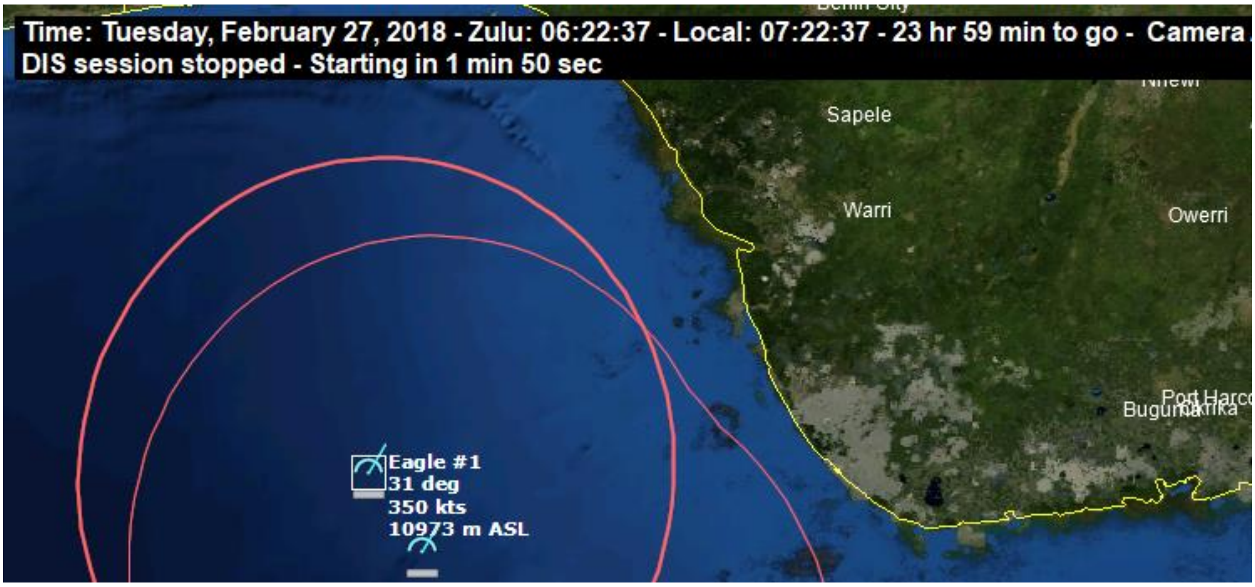
On the DIS-specific settings, the ExerciseID, SiteID, ApplicationID and ForceID values represent the exercise, site, application and force identifiers used when participating in a DIS session.

The “IPAddress” value specifies IP address for broadcasting & receiving PDUs. This address is used both to calculate the broadcast IP address (in combination with the connection’s subnet mask) and also to "pin" the transmission & receival of DIS packets to a specific network interface card (NIC), which can be handy if running on a system with multiple NICs.

The “Port” value indicates the TCP/IP port used both for sending and receiving PDUs (only broadcast mode is currently supported).

The “AlwaysActive” field defines how CPE will join/start a DIS session. If set to “True”, CPE will immediately start broadcasting and receiving messages on startup, without any consideration to time synchronization. This is useful for quick hookup tests or for cases where synchronized simulated time is not essential.
For environments where following a common simulated time is important, or where a session must commence at a specific real-world time, it is recommended to set this value to “False”. In this case, CPE will not immediately join a session but will instead wait for a proper Start/Resume PDU from the controller application, to define the real-world start time and the simulated scenario time.

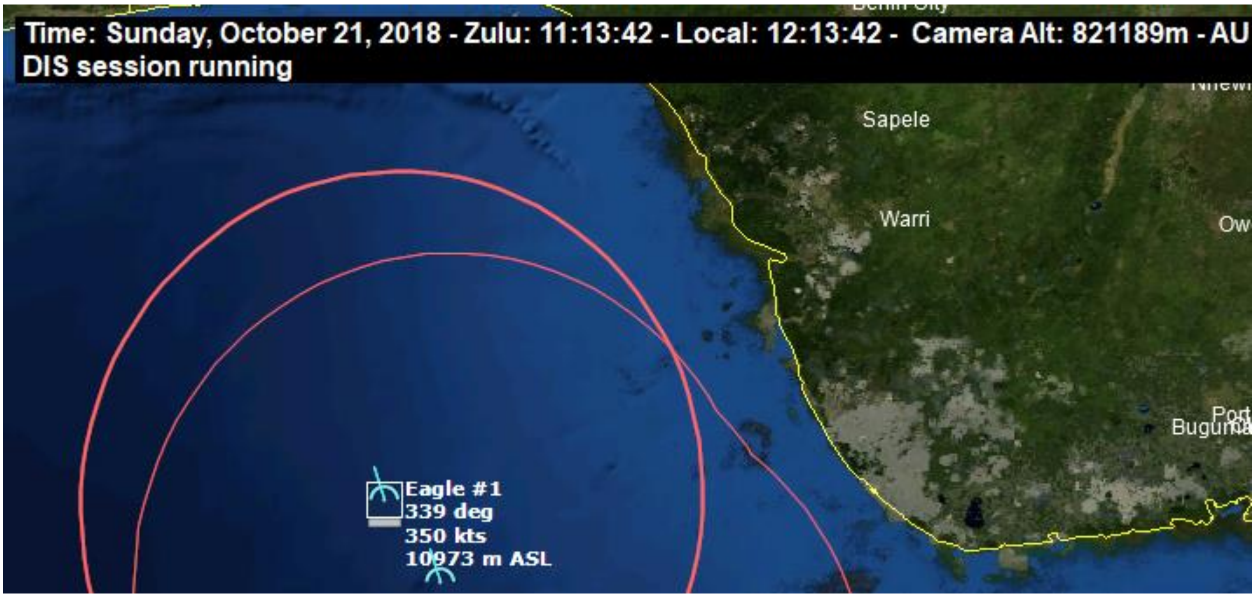
With AlwaysActive set to “False”, after CPE has received a Start/Resume PDU defining when the session will start, it displays a running countdown-to-start on the caption text:



Time: Tuesday, February 27, 2018 - Zulu: 06:22:37 - Local: 07:22:37 - 23 hr 59 min to go - Camera

DIS session stopped - Starting in 1 min 50 sec

Once CPE has actually joined a DIS session (or has automatically started, with AlwaysActive set to "True"), the caption text changes to indicate this:

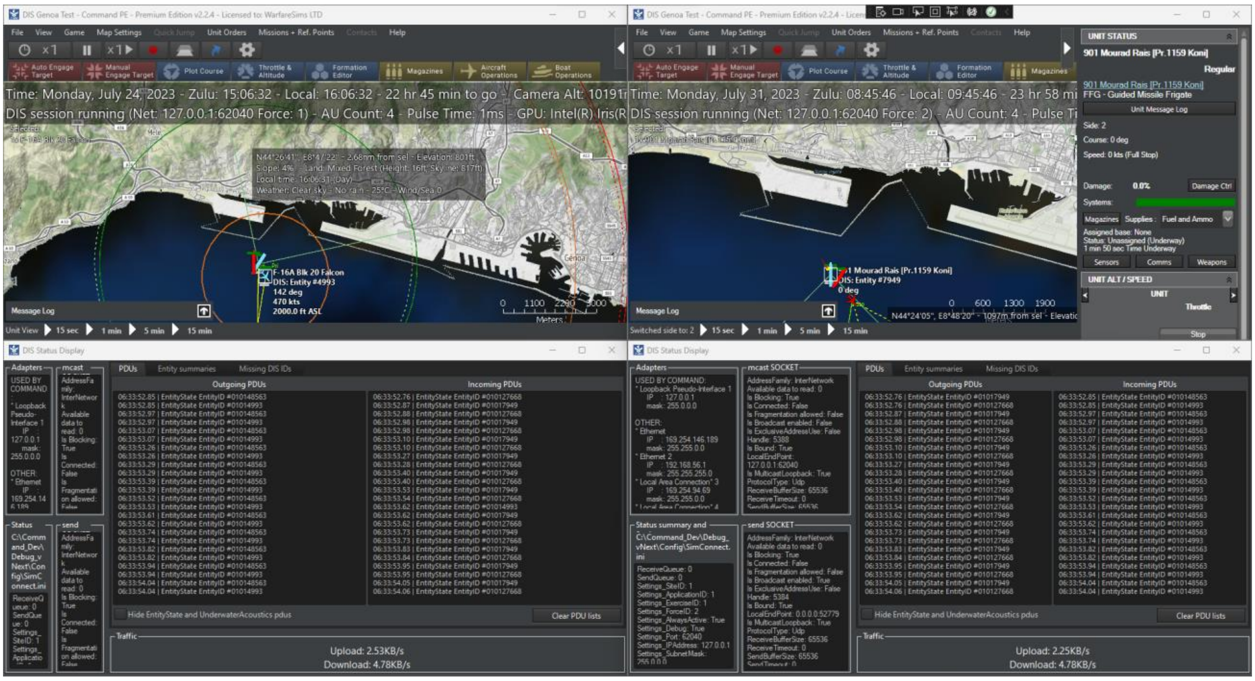


Time: Sunday, October 21, 2018 - Zulu: 11:13:42 - Local: 12:13:42 - Camera Alt: 821189m - AU

DIS session running

Debug mode

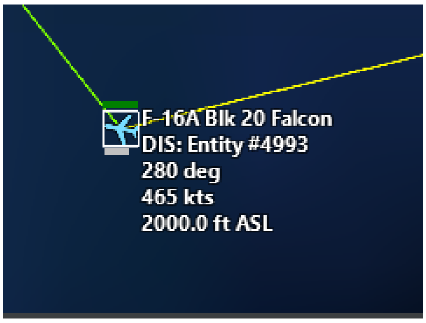
If "Debug mode" is enabled, when the DIS-enabled scenario starts, an additional "Debug" window is shown, which presents various diagnostic information about incoming and outgoing DIS PDUs:



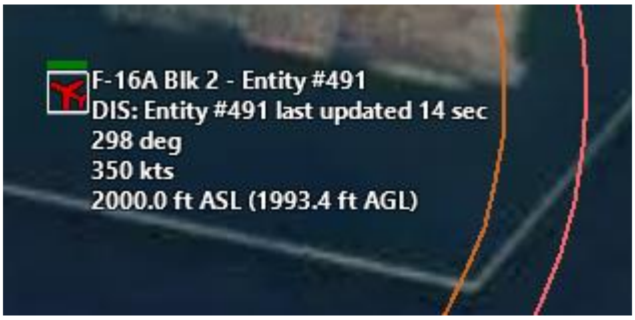
Time: Monday, July 24, 2023 - Zulu: 15:06:32 - Local: 16:06:32 - 22 hr 45 min to go - Camera Alt: 10191
DIS session running (Net: 127.0.0.1:62040 Force: 1) - AU Count: 4 - Pulse Time: 1ms - GPU: Intel(R) Iris(R

Time: Monday, July 31, 2023 - Zulu: 08:45:46 - Local: 09:45:46 - 23 hr 58 mi
DIS session running (Net: 127.0.0.1:62040 Force: 2) - AU Count: 4 - Pulse Ti

Additionally, in debug mode units rendered on the map have additional DIS-related information included on their datablock text, as in these examples:



F-16A Blk 20 Falcon
DIS: Entity #4993
280 deg
465 kts
2000.0 ft ASL



F-16A Blk 2 - Entity #491
DIS: Entity #491 last updated 14 sec
298 deg
350 kts
2000.0 ft ASL (1993.4 ft AGL)

The special case of the UA PDU

The Underwater Acoustic Emission (UA) PDU warrants special mention. One of its essential fields, “Passive Parameter Index”, is used to indicate a unit’s passive sonar signature (ie. emitted noise level).

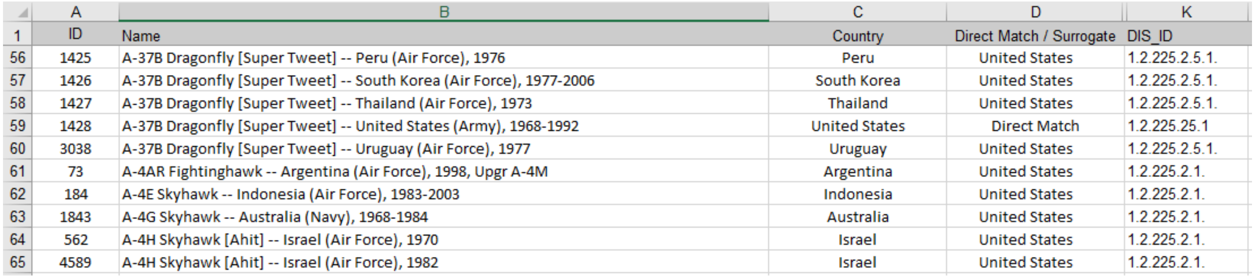
According to the IEEE spec (IEEE 1278.1-2012, section “7.6.4 Underwater Acoustic (UA) PDU”), *“[…] This field indicates which [common acoustic database] record (or file) shall be used in the definition of passive signature (unintentional) emissions of the entity. The indicated database record (or file) shall define all noise generated”*. However, no mention of this “common acoustic database” is present anywhere in the specification; accordingly, it is reasonable to assume that such a resource will be site/user-specific.

Accordingly, Command PE’s default implementation is to use the “Passive Parameter Index” field to store (and retrieve) a unit’s desired speed, ie. the “make turns for X knots” value. This effectively provides the powerplant/shaft operating speed and thus noise. Combined with the flow noise value (obtained by looking up the unit's actual speed from its latest available Entity-state PDU record), this allows determining a unit's complete passive sonar noise profile.

This default behavior for the "Passive Parameter Index" field can be altered via a suitable Mechanics Override.

Associating DIS and CPE entities

The "translation" between the DIS entity notation syntax (e.g. 1.1.225.1.2.0) and CPE's platforms and systems is handled by association tables in Excel files, located in: **[top-level writable folder]\Resources\DIS\ .** Each of the Excel files (Aircraft.xls, Ship.xls etc.) corresponds to the relevant database annex and each record has a matching reference to the appropriate (exact or nearest match) DIS entity record:



	A	B	C	D	K
1	ID	Name	Country	Direct Match / Surrogate	DIS_ID
56	1425	A-37B Dragonfly [Super Tweet] -- Peru (Air Force), 1976	Peru	United States	1.2.225.2.5.1.
57	1426	A-37B Dragonfly [Super Tweet] -- South Korea (Air Force), 1977-2006	South Korea	United States	1.2.225.2.5.1.
58	1427	A-37B Dragonfly [Super Tweet] -- Thailand (Air Force), 1973	Thailand	United States	1.2.225.2.5.1.
59	1428	A-37B Dragonfly [Super Tweet] -- United States (Army), 1968-1992	United States	Direct Match	1.2.225.25.1
60	3038	A-37B Dragonfly [Super Tweet] -- Uruguay (Air Force), 1977	Uruguay	United States	1.2.225.2.5.1.
61	73	A-4AR Fightinghawk -- Argentina (Air Force), 1998, Upgr A-4M	Argentina	United States	1.2.225.2.1.
62	184	A-4E Skyhawk -- Indonesia (Air Force), 1983-2003	Indonesia	United States	1.2.225.2.1.
63	1843	A-4G Skyhawk -- Australia (Navy), 1968-1984	Australia	United States	1.2.225.2.1.
64	562	A-4H Skyhawk [Ahit] -- Israel (Air Force), 1970	Israel	United States	1.2.225.2.1.
65	4589	A-4H Skyhawk [Ahit] -- Israel (Air Force), 1982	Israel	United States	1.2.225.2.1.

Storing these associations on the Excel files means that end-users can easily tailor them to their own custom needs and extend them as appropriate (e.g. add more associations for platforms/systems that they have created) without the intervention of the dev team.

The file should match between the various command instances running DIS and should preferably have only 1-to-1 associations in order to avoid "translation" issues when receiving incoming PDUs. The loaded scenario must contain only units belonging to the side that matches the instance's force-ID, and those units must be properly associated with a DIS ID in the association files.
In case a unit with no matching DIS ID is present in the scenario, that unit **WILL NOT** be broadcast on the DIS protocol and (if debug mode is enabled) an entry about that unit will be visible in the "Missing DIS IDs" tab in the debug window.

TCP/IP-Socket access to Lua API

(Available in PE – Optional in AE)

CPE allows connecting to the Lua API of the currently running scenario through a standard TCP/IP socket. The configuration for this feature is stored on **[top-level writable folder]\Config\CPE.ini**:

```
[Lua]
EnableSocket = 0 🡐 0 to disable, 1 to enable
SocketPort = 7777 🡐 default port, can be changed
EncodingMode = 0 🡐encoding type
```

Command must be restarted in order for changes in these settings to take effect.

The socket server running on the specified port receives all Lua commands and transfers them to the Lua API for the currently running scenario.

The "Encoding Mode" property allows using different encoding types when connecting to Lua TCP socket. Valid values are:

- 0 🡒 Big-endian Unicode (default)
- 1 🡒 Local system default
- 2 🡒 Unicode
- 3 🡒 ASCII
- 7 🡒 UTF-7
- 8 🡒 UTF-8
- 32 🡒 UTF-32

The value set is respect for both incoming and outgoing data (i.e. responses from the Lua API will use the same encoding).

External programs that use this feature must first connect to the specified socket before sending commands. Otherwise a "connection not established" exception is likely to be generated.

The syntax for the Lua scripts is exactly the same as when using the Lua console. The only necessary addition is that, for scripts that execute input without returning a value (e.g. to add a new unit), the script must be predated by the comment "--script ". This signals to the Lua interpreter that the script should be executed without providing any feedback. Methods that do return a value should omit this prefix.

Lua TCP/IP connection examples

1) Simple socket client, written in VB.NET:

```
Imports System.Net.Sockets

Class MainWindow

    Private clientSocket As New System.Net.Sockets.TcpClient()

    Private Sub Button_Click(sender As Object, e As RoutedEventArgs)
        Try
            clientSocket.Connect("127.0.0.1", CInt(TB_SocketPort.Text))
            msg("Client Started")
            MsgBox("Connected!")
        Catch ex As Exception
            MsgBox("Error: " & ex.Message)
        End Try
    End Sub

    Private Sub msg(theStr As String)
        Console.WriteLine(">> " & Trim(theStr))
    End Sub

    Private Sub Button_Click_1(sender As Object, e As RoutedEventArgs)
        Dim serverStream As NetworkStream = clientSocket.GetStream()
        Dim outStream As Byte() =
System.Text.Encoding.BigEndianUnicode.GetBytes(TB_Script.Text)
        serverStream.Write(outStream, 0, outStream.Length)
        serverStream.Flush()

        Dim inStream(clientSocket.ReceiveBufferSize) As Byte
        serverStream.Read(inStream, 0, CInt(clientSocket.ReceiveBufferSize))
        Dim returndata As String =
System.Text.Encoding.BigEndianUnicode.GetString(inStream)
        returndata = returndata.Replace(vbNullChar, "")
        returndata = returndata.Remove(returndata.Length - 1)
        TB_Result.Text = returndata

        'msg("Data from Server : " + returndata)
    End Sub
End Class
```

2) Small, self-contained example in Python:

```
import socket
TCP_IP = '127.0.0.1'
TCP_PORT = 7777
BUFFER_SIZE = 1024
message_string = "--script\nScenEdit_SpecialMessage('playerside','This is written
externally')\nScenEdit_AddUnit({side='playerside',name='FFH 150
ANZAC',type='Ship',dbid=2607,latitude='-32.5144788463646',
longitude='154.963352242518'}) "
message = message_string.encode('UTF-16-BE')
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send(message)
data = s.recv(BUFFER_SIZE)
print(data.decode('utf-8'))
```

3) More elaborate example in Python:

```
import socket
```

```
from Misc import Status, SerpentLogging
from enum import Enum
import re
import LuaHandler
import time

class TCP_Connection:
    TCP_IP = '127.0.0.1'
    TCP_PORT = 7777
    BUFFER_SIZE = 4096
    INPUT_ENCODING = 'UTF-8'
    OUTPUT_ENCODING = 'utf-8'
    SOCKET = None
    CONNECTION_STATUS = Status.NEW

    def __init__(self):
        if TCP_Connection.CONNECTION_STATUS == Status.CONNECTED:
            self.TestConnection()
        else:
            TCP_Connection.SOCKET = socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
            TCP_Connection.SOCKET.connect((TCP_Connection.TCP_IP,
TCP_Connection.TCP_PORT))
            self.TestConnection()

    def TestConnection(self):
        message =
TCP_Connection.Return(LuaHandler.SerpentFunctions.GetBuildNumber())
        if message:
            TCP_Connection.CONNECTION_STATUS = Status.CONNECTED
            SerpentLogging.PrintAndLog(f'Connected to Command via TCP:
{TCP_Connection.TCP_IP}:{TCP_Connection.TCP_PORT}')
            return Status.SUCCESSFUL
        else:
            TCP_Connection.CONNECTION_STATUS = Status.DISCONNECTED
            SerpentLogging.PrintAndLog(f'Unable to connect to Command via TCP:
{TCP_Connection.TCP_IP}:{TCP_Connection.TCP_PORT}','error')
            return Status.FAILED

    @classmethod
    def Return(self, message):
        message = message.encode(TCP_Connection.INPUT_ENCODING)
        TCP_Connection.SOCKET.send(message)
        result = b""
        while True:
            part = TCP_Connection.SOCKET.recv(TCP_Connection.BUFFER_SIZE)
            result += part
            if len(part) < TCP_Connection.BUFFER_SIZE:
                # either 0 or end of data
                break
        result = result.decode(TCP_Connection.OUTPUT_ENCODING)
        return result
```

Using .NET libraries

Beginning in v2.2.4.1, CPE has the ability to load external Microsoft .NET Framework assemblies or libraries and use them just like it uses its built-in components. Both Microsoft official .NET libraries (ie. members of the Common Language Runtime) as well as third-party custom libraries can be used in this way.

For security reasons this ability is disabled by default. To enable it, open the file **[Writable root]\Config\CPE.ini** and, under the [Lua] section, set/add this property: **AllowCLRPackage = 1**.

Two usage examples of this functionality:

```
threading = CLRPackage('System', 'System.Threading')
print('Sleeping...')
threading.Thread.Sleep(10000)
print('Awake!')
```

```
SystemNet = CLRPackage('System', 'System.Net')
local c = SystemNet.WebClient()
print(c)
local s = c:DownloadString('https://httpbin.org/ip')
print (s)
```

FLOATING LICENSE MANAGER

Purpose and overview

By default, CPE searches for a license file (license.dat) placed on the main application root folder. This implementation is geared towards users needing to run CPE on a single machine. There may be times when this arrangement is impractical such as:

- When an analyst wishes to use multiple virtual machines (VMs) that require individual license issue on demand.

- When a team using multiple machines not at the same time.

For this and other similar situations, CPE offers a *floating license* mode that transfers licensure to a server application, allowing individual clients on the same network as the application to request a license on an ad hoc basis. This server application, known as the *floating license manager,* has a certain number of licenses assigned to (such as 50) and it the server operates completely independent of any external links (e.g. the internet). Users on the same network as this server can now reserve licenses when they are running CPE, and then release them when they have closed the application.

Any instance of CPE, whether traditional or command-line, will automatically check for a local license file at startup. If this file is not found, CPE will:

1. Generate a hardware UID and prompt the user to obtain a license.

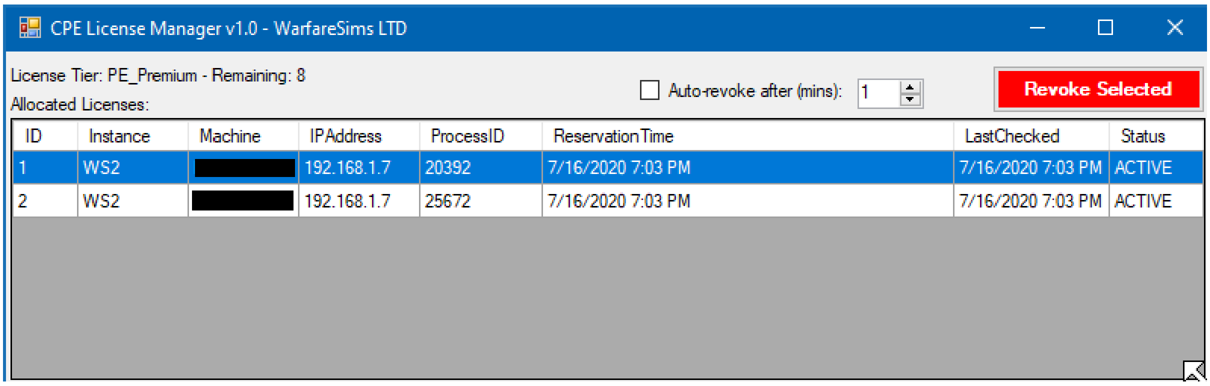
   -or

2. If the CPE instance is confligured to point to the floating license manager, it will automatically attempt contanct in order to obtain a floating license.

If CPE is able to obtain the license, the program works normally. Note that the client will continually query (every few seconds) the floating license manager to check that its license is still valid and has not been revoked by the FLM operator. **This means that clients utilizing floating licenses need to have a reliable, always-on network connection to the running FLM.**

Step by step: Running a client instance using a floating license

This example assumes that you have an FLM instance installed, with its own dedicated license file (license_ILM.dat) present. Start the FLM, and a window such as this should appear:



CPE License Manager v1.0 - WarfareSims LTD

License Tier: PE_Premium - Remaining: 8

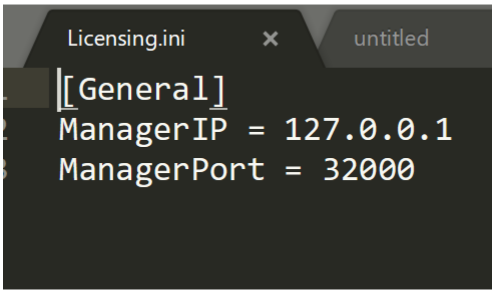
Allocated Licenses:

□ Auto-revoke after (mins): 1

ID	Instance	Machine	IPAddress	ProcessID	ReservationTime	LastChecked	Status
1	WS2	[redacted]	192.168.1.7	20392	7/16/2020 7:03 PM	7/16/2020 7:03 PM	ACTIVE
2	WS2	[redacted]	192.168.1.7	25672	7/16/2020 7:03 PM	7/16/2020 7:03 PM	ACTIVE

This shows that in this case we have a batch of 10 PE-Premium licenses available to allocate, and two of them have already been reserved. By convention, the FLM uses **port 32000** to listen for incoming license requests.

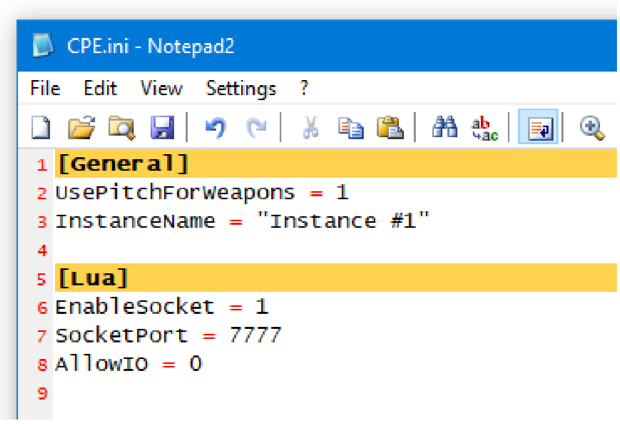
Now go to the desired client instance and add a new file to \Config, called **Licensing.ini** . Its text content should be:



```
[General]
ManagerIP = 127.0.0.1
ManagerPort = 32000
```

In this example, the FLM and the desired client are running on the same machine, so the FLM config will point to the same machine (hence 127.0.0.1) for the address; if the FLM is running on a different machine then that machine's IP address must be entered here.

You will also need to provide a unique instance name, this can be accomplished by setting this on the file \Config\CPE.ini, like this:



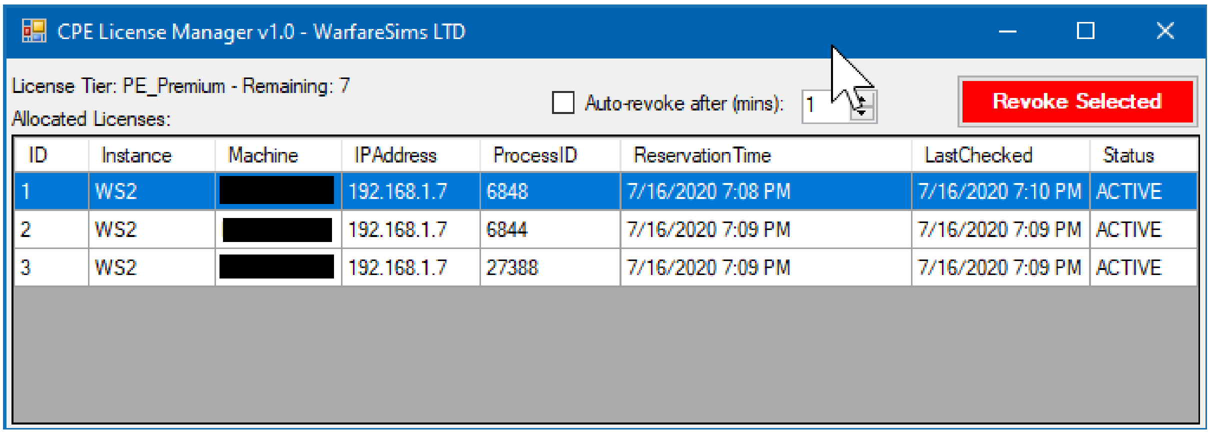
CPE.ini - Notepad2

```
[General]
UsePitchForWeapons = 1
InstanceName = "Instance #1"

[Lua]
EnableSocket = 1
SocketPort = 7777
AllowIO = 0

```

Make sure that there is no local license file present *(ie. license.dat file)* and start up Command.exe or CommandCLI.exe . The client should start up normally, and you will notice a new license reservation appear on the FLM grid:



License Tier: PE_Premium - Remaining: 7

□ Auto-revoke after (mins): 1

Allocated Licenses:

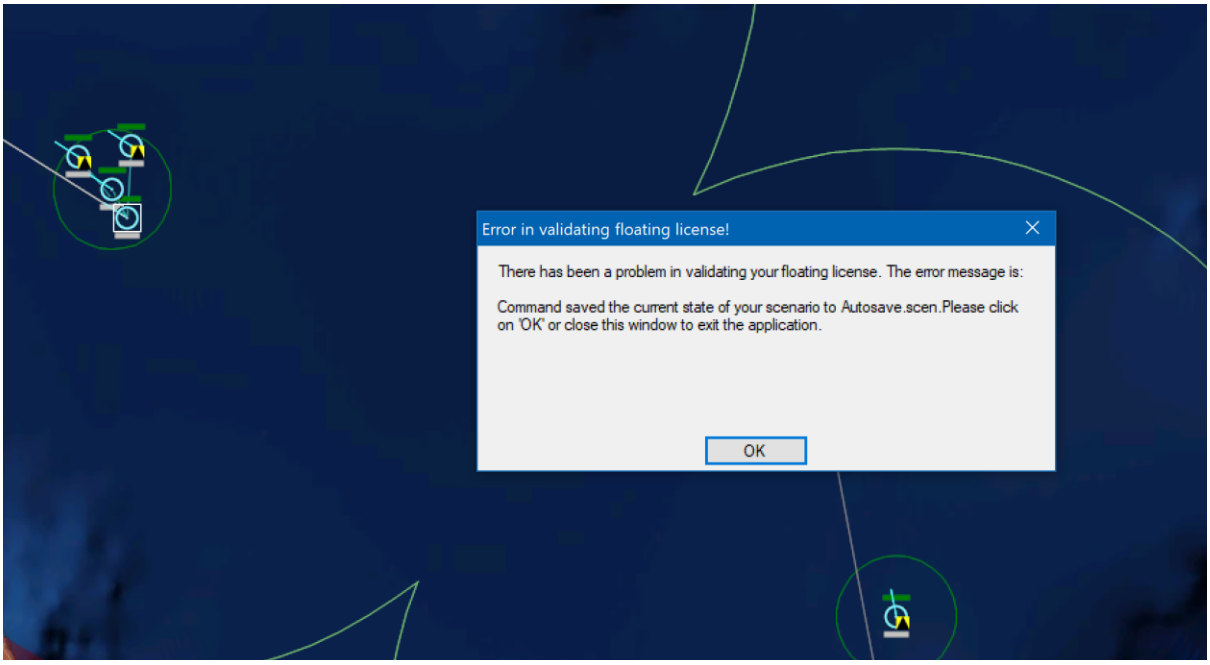
ID	Instance	Machine	IPAddress	ProcessID	ReservationTime	LastChecked	Status
1	WS2	[redacted]	192.168.1.7	6848	7/16/2020 7:08 PM	7/16/2020 7:10 PM	ACTIVE
2	WS2	[redacted]	192.168.1.7	6844	7/16/2020 7:09 PM	7/16/2020 7:09 PM	ACTIVE
3	WS2	[redacted]	192.168.1.7	27388	7/16/2020 7:09 PM	7/16/2020 7:09 PM	ACTIVE

The grid shows a number of information that llaows the operator to identify the CPE instances by both name and machine name and IP address, as well as process ID (since multiple instances of CPE may run on the same machine, using the same instance name if so configured, this is necessary in order to differentiate them).

As the client proceeds with whatever task it is assigned, in the background it regularly polls the FLM to check that its reservation is still valid.

Let us now see what happens when we revoke the allocated license. From the CPE instance's information, identify the relevant allocated license. Select its line in the grid, and click on the button "Revoke Selected".

The selected license should be removed from the FLM's visual grid (and its internal database). After a few seconds, the associated CPE client instance will halt execution and report that it can no longer confirm that its license is valid, and will exit. The precise way that this is reported varies between the full-GUI client and the CLI edition; below is an example from the GUI version:



Error in validating floating license!

There has been a problem in validating your floating license. The error message is:

Command saved the current state of your scenario to Autosave.scen.Please click on 'OK' or close this window to exit the application.

APPENDICES

Appendix 1: Sensor model factors (except sonar)

The following factors are currently taken into account when evaluating sensor detections in the Command-PE simulation:

Radar/OECM model

- Radio Line of Sight (separate checks for horizon, straight-line, OTH-surface wave and OTH-backscatter)
- Specific periscope-search capability (only a few sets have this)
- LDSD limitations (separate full- and limited-LDSD e.g. AWG-10, Saphir-23/25)
- Elevation/lookdown angle
- Horizontal & vertical beamwidth
- Power output (peak)
- Pulse width
- Processing gain
- System noise level
- Operating frequency
- Pulse-repetition frequency (PRF)
- Gain & “effective” power bonuses for PESA & especially AESA antennas
- Target RCS (dependent on aspect and frequency band)
- GMTI only: target speed (fast targets show up more)
- Land targets only: Bonus for high-frequency radars (e.g. MMW)
- Presence and geometry of jamming sources
- Tech generation comparison vs. jamming sources (abstracts ECCM techniques)
- Terrain & sea clutter (slope-dependent for land)
- Weather (rain → propagation loss)

ESM/RWR model

- Radio Line of Sight (separate checks for horizon, straight-line, OTH-surface wave and OTH-backscatter)
- Frequency band compatible match with target emitter
- Elevation/lookdown angle
- Emitter horizontal & vertical beamwidth
- Emitter power output (peak)
- Emitter pulse width
- Receiver processing gain
- Receiver system noise level
- Emitter LPI design
- AESA emitters get significantly reduced “effective” detected power level (simulates spread-spectrum, power management and other LPI techniques)
- Weather (rain → propagation loss)

Infrared model

- Visual Line of Sight (incl. cloud blockage)
- Specific periscope-search capability
- Target IR signature (aspect-dependent, various modifiers for afterburner, rocket boost phase, supersonic/hypersonic etc., distinct values for detection & classification)
- Sensor zoom levels (distinct values for detection, tracking & classification - e.g. narrow-FOV "soda straw" sensors like EO/IR turrets may track & classify at very long range but actually detect at very short)
- Weather (clouds, fog, rain → high absorption)
- Time of day (--> ambient temperature)

Visual model

- Visual Line of Sight (incl. cloud blockage)
- Contrails (atmospheric targets), wakes (sea surface), dust & motion (ground vehicles) - can detect at far greater distances than actual target object under suitable conditions
- Target visual signature (aspect-dependent, various modifiers for afterburner, rocket boost phase, supersonic/hypersonic etc., distinct values for detection & classification)
- Sensor zoom levels (distinct values for detection, tracking & classification - e.g. narrow-FOV "soda straw" sensors like EO/IR turrets may track & classify at very long range but actually detect at very short)
- Weather (clouds, fog, rain → sharp reduction)
- Time of day (--> ambient light)

Laser model

- Visual Line of Sight (incl. cloud blockage)
- Weather (clouds, fog, rain → sharp reduction)

Appendix 2: The sonar model in Command PE

Contrary to the radar/ECM/ESM model, Command-PE's default sonar model does not start from the bare physics/EM formulas to work its way up; instead, it uses an "effects-based" model that adjusts itself to publicly available information on both figures and factors. This allows incorporating new information and additional factors much easier than when a physics-based model is used.

The factors that affect a platform's detectability to a sonar sensor include:

Active sonar

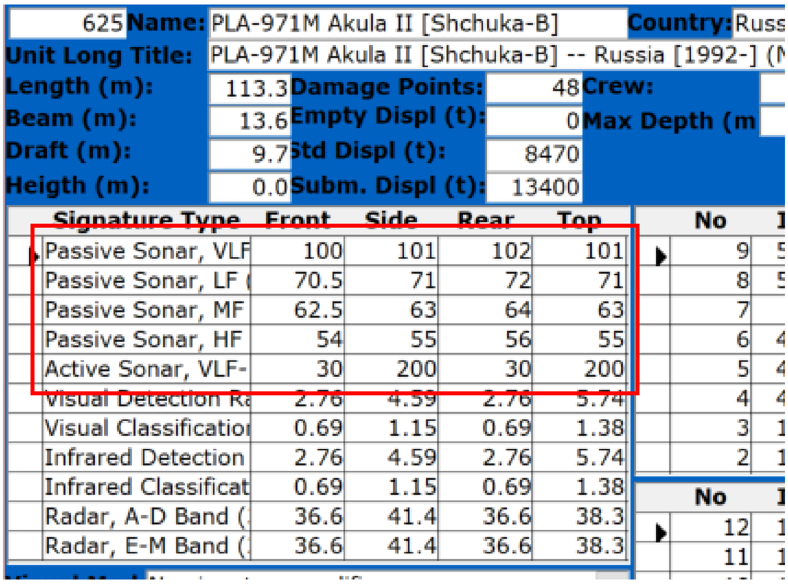
- Underwater Line of Sight
- Sensor platform own powerplant & propeller noise (adjusted for throttle). Propeller noise adjusted depending on type (e.g. shroud).
- Sensor platform own flow noise (adjusted for current speed)
- External towed sensors (VDS, towed array etc.) get significant own-noise reductions
- Explicit sensor depth if different from platform depth (e.g. towed array, VDS, dipping sonar)
- Target echo signature (aspect-, frequency- and platform-dependent)
- Effects of surface ducting (<-- affected by weather)
- Effects of thermal layer
- Effects of deep sound channel
- Effects of reverberation
- Effects of bottom clutter (if target is close to bottom)
- Limits of direct-path propagation
- Bottom-bounce mode (with check for suitable seabed slope)
- Convergence-zone mode (with check for CZ raypath blockage, e.g. by underwater ridge)

Passive sonar & ping-intercept

- Underwater Line of Sight (certain sensors like bottom arrays / SOSUS ignore this)
- Sensor platform own powerplant & propeller noise (adjusted for throttle). Propeller noise adjusted depending on type (e.g. shroud).
- Sensor platform own flow noise (adjusted for current speed)
- External towed sensors (VDS, towed array etc.) get significant own-noise reductions
- Explicit sensor depth if different from platform depth (e.g. towed array, VDS, dipping sonar)
- Target powerplant & propeller noise (aspect-, frequency- and platform-dependent, adjusted for throttle). Propeller noise adjusted depending on type (e.g. shroud).
- Target flow noise (adjusted for current speed)
- Effects of surface ducting (<-- affected by weather)
- Effects of thermal layer
- Effects of deep sound channel
- Effects of reverberation
- Limits of direct-path propagation
- Convergence-zone mode (with check for CZ raypath blockage, e.g. by underwater ridge)

The basics

Each naval platform/system in the simulation database has a set of sonar signatures, as in this example:



625	Name:	PLA-971M Akula II [Shchuka-B]		Country:	Russ
Unit Long Title:		PLA-971M Akula II [Shchuka-B] -- Russia [1992-] (N			
Length (m):	113.3	Damage Points:	48	Crew:	
Beam (m):	13.6	Empty Displ (t):	0	Max Depth (m	
Draft (m):	9.7	Std Displ (t):	8470		
Heigth (m):	0.0	Subm. Displ (t):	13400		

Signature Type	Front	Side	Rear	Top
Passive Sonar, VLF	100	101	102	101
Passive Sonar, LF (	70.5	71	72	71
Passive Sonar, MF	62.5	63	64	63
Passive Sonar, HF	54	55	56	55
Active Sonar, VLF-	30	200	30	200
Visual Detection Ra	2.76	4.59	2.76	5.74
Visual Classificatio	0.69	1.15	0.69	1.38
Infrared Detection	2.76	4.59	2.76	5.74
Infrared Classificat	0.69	1.15	0.69	1.38
Radar, A-D Band (	36.6	41.4	36.6	38.3
Radar, E-M Band (	36.6	41.4	36.6	38.3

No	I
9	5
8	5
7	
6	4
5	4
4	4
3	1
2	1

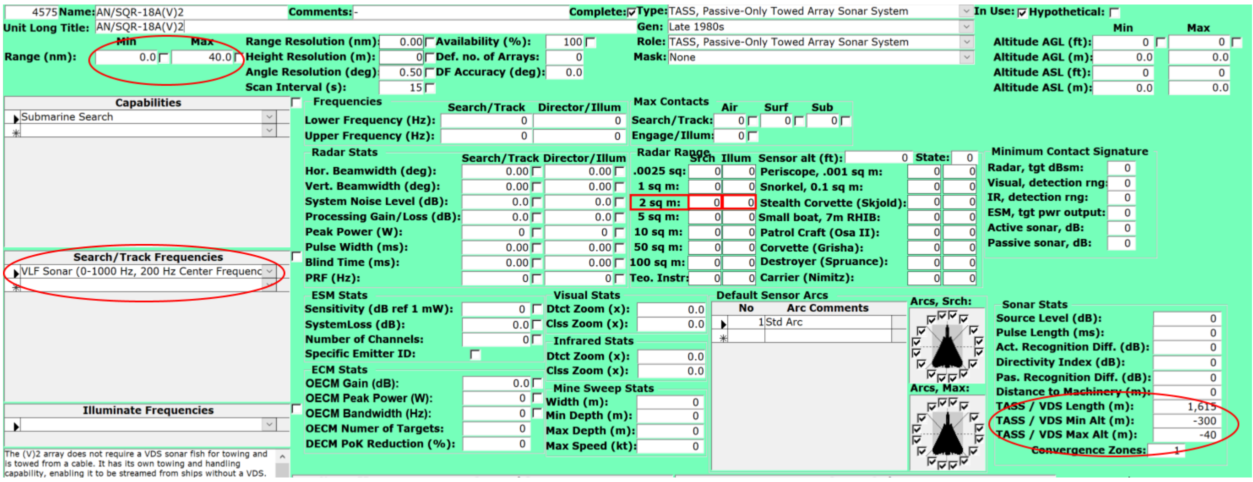
No	I
12	1
11	1

The passive sonar figures represent the noise level of the platform (in decibels) at maximum throttle and maximum through-water speed, at the different acoustic frequency bands. Within the simulation, the target noise is broken down to powerplant and flow noise (for instance, a submarine may be starting to drift after a high-speed dash, in this case its flow noise will be high but its powerplant & propeller noise will be low).

The target noise signatures are compared against "reference maximums", e.g. for submarines the reference maximum is a November-class submarine at flank throttle. For each unit, four discrete acoustic bands are considered: VLF (100Hz reference frequency), LF (3kHz), MF (7.5kHz) and HF (15kHz).

(Note: At frequencies greater than 200 Hz the Source Level falls by 6 dB when the frequency is doubled. This is reflected by the units' passive sonar signature. For example, a sub with a sonar signature of 140dB at 1-200Hz, has a signature of 110dB at 3kHz, 102dB at 7.5kHz, and 94dB at 15kHz)

For sonar sensors, the critical values used by the default model are the maximum detection range (this is the “everything perfect” nominal range against the reference-loudest targets), the search/track frequency bands, and the VDS/TASS properties if applicable:



4575 Name: AN/SQR-18A(V)2 Comments: - Complete: ☑ Type: TASS, Passive-Only Towed Array Sonar System In Use: ☑ Hypothetical: □

Unit Long Title: AN/SQR-18A(V)2 Gen: Late 1980s

Range (nm): Min 0.0 □ Max 40.0 □

Range Resolution (nm): 0.00 □ Availability (%): 100 □ Role: TASS, Passive-Only Towed Array Sonar System

Height Resolution (m): 0 □ Def. no. of Arrays: 0 Mask: None

Angle Resolution (deg): 0.50 □ DF Accuracy (deg): 0.0

Scan Interval (s): 15 □

	Min		Max	
Altitude AGL (ft):	0	□	0	□
Altitude AGL (m):	0.0		0.0	
Altitude ASL (ft):	0		0	
Altitude ASL (m):	0.0		0.0	

Capabilities: Submarine Search

Search/Track Frequencies: VLF Sonar (0-1000 Hz, 200 Hz Center Frequenc

Illuminate Frequencies:

The (V)2 array does not require a VDS sonar fish for towing and is towed from a cable. It has its own towing and handling capability, enabling it to be streamed from ships without a VDS.

Frequencies

	Search/Track	Director/Illum
Lower Frequency (Hz):	0	0
Upper Frequency (Hz):	0	0

Max Contacts

	Air	Surf	Sub
Search/Track:	0 □	0 □	0 □
Engage/Illum:	0 □		

Radar Stats

	Search/Track	Director/Illum
Hor. Beamwidth (deg):	0.00 □	0.00 □
Vert. Beamwidth (deg):	0.00 □	0.00 □
System Noise Level (dB):	0.0 □	0.0 □
Processing Gain/Loss (dB):	0.0 □	0.0 □
Peak Power (W):	0 □	0 □
Pulse Width (ms):	0.00 □	0.00 □
Blind Time (ms):	0.00 □	0.00 □
PRF (Hz):	0 □	0 □

Radar Range

	Srch	Illum	Sensor alt (ft): 0	State: 0	
.0025 sq:	0	0	Periscope, .001 sq m:	0	0
1 sq m:	0	0	Snorkel, 0.1 sq m:	0	0
2 sq m:	0	0	Stealth Corvette (Skjold):	0	0
5 sq m:	0	0	Small boat, 7m RHIB:	0	0
10 sq m:	0	0	Patrol Craft (Osa II):	0	0
50 sq m:	0	0	Corvette (Grisha):	0	0
100 sq m:	0	0	Destroyer (Spruance):	0	0
Teo. Instr:	0	0	Carrier (Nimitz):	0	0

Minimum Contact Signature

Radar, tgt dBsm: 0
Visual, detection rng: 0
IR, detection rng: 0
ESM, tgt pwr output: 0
Active sonar, dB: 0
Passive sonar, dB: 0

ESM Stats

Sensitivity (dB ref 1 mW): 0 □
SystemLoss (dB): 0.0 □
Number of Channels: 0 □
Specific Emitter ID: □

ECM Stats

OECM Gain (dB): 0.0 □
OECM Peak Power (W): 0 □
OECM Bandwidth (Hz): 0 □
OECM Numer of Targets: 0
DECM PoK Reduction (%): 0

Visual Stats

Dtct Zoom (x): 0.0
Clss Zoom (x): 0.0

Infrared Stats

Dtct Zoom (x): 0.0
Clss Zoom (x): 0.0

Mine Sweep Stats

Width (m): 0
Min Depth (m): 0
Max Depth (m): 0
Max Speed (kt): 0

Default Sensor Arcs

No	Arc Comments
1	Std Arc

Sonar Stats

Source Level (dB): 0
Pulse Length (ms): 0
Act. Recognition Diff. (dB): 0
Directivity Index (dB): 0
Pas. Recognition Diff. (dB): 0
Distance to Machinery (m): 0
TASS / VDS Length (m): 1,615
TASS / VDS Min Alt (m): -300
TASS / VDS Max Alt (m): -40
Convergence Zones: 1

The other fields such as source level, pulse length, active recognition difference, directivity index etc. are not currently used by the default model, but can be populated for use by a different model through overrides.

Unless the sensor is a sonobuoy or dipping sonar, its own mounted-platform noise must also be considered. As with the target noise, this is also broken down into flow and powerplant/propeller components and evaluated for self-noise. Outboard sensors (e.g. VDS or towed array) get a significant relief from this, as they are severely less affected.

Line-of-sight is necessary in order to achieve detection, with separate check for blockage from landmass (for example, SOSUS arrays can hear beyond the sonar "horizon" but are still blocked by i.e. an island or peninsula).

Other fundamental factors include:

- Target aspect (critical for active sonar but also makes a difference for passive)
- Target and sensor-parent cavitation (boosts noise)
- Target submarine snorkelling (again increases noise)
- Sensor arc restrictions (e.g. hull sonar baffles)
- Onboard/offboard nature of sensor (offboard sensors get significant own-noise reductions)

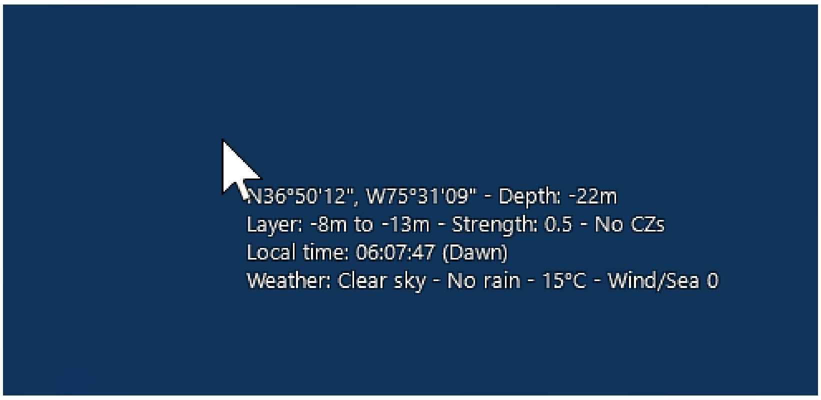
These are the “hunter and target in a perfect day” bare basics. But in real life, and in Command PE, the environment almost always gets a vote.

Surface ducting and thermocline

Command’s sonar model assumes a “hard” limit of 20K yards / 9.5nm for direct-path propagation. The marine environment can modify this limit, however, by boosting or degrading an acoustic signal depending on circumstances.

Surface ducting can extend the direct-path range up to 1.5x the nominal value, in a perfectly calm sea (state 0). As the sea state increases, the ducting ability is rapidly curtailed.

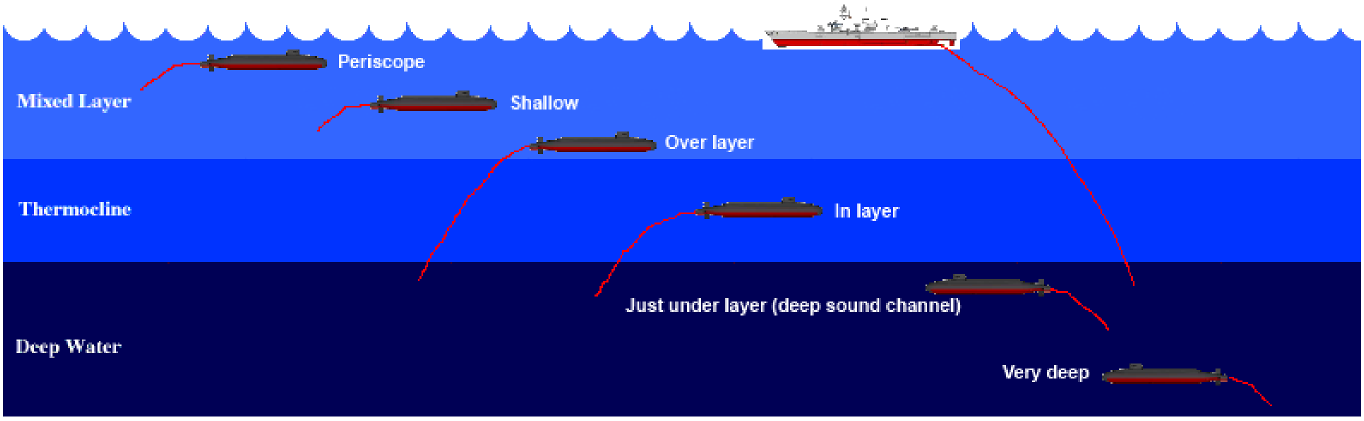
The thermocline layer is one of the most critical environmental factors. Information on the local thermocline conditions is always displayed on the map-cursor databox, as in this example:



N36°50'12", W75°31'09" - Depth: -22m
Layer: -8m to -13m - Strength: 0.5 - No CZs
Local time: 06:07:47 (Dawn)
Weather: Clear sky - No rain - 15°C - Wind/Sea 0

The “Layer: XX ft to YY ft” text describes the floor and ceiling of the thermocline in the area of the cursor; these are automatically calculated by local bottom depth and latitude. The “Strength” value indicates the effectiveness of the thermocline in absorbing sound energy, thus degrading a signal passing through. Note that the layer ceiling and bottom, as well as its "thickness" (and hence absorption rate) are not fixed, but vary according to latitude, local depth and local temperature (the estimated actual values are shown on the map cursor). Trying to manually keep a submarine at the edges of the layer as these values shift can be a very tedious job, hence the depth presets (F2 window) which automate this.

Following is a brief summary of how the different depth ranges (reflected on the depth-presets on the unit throttle/altitude window) are modelled in Command, and the tactical implications of each. In the graphic, towed arrays and/or VDS systems are represented by the red tails on each platform:



Periscope depth: This is the only depth at which subs can use their above-water sensors (periscope, ESM, radar etc.). A strong surface duct is usually present, which magnifies the transmission range of noise well beyond nominal (in fact, in calm seas a surface ship can use this duct to get early warning of threats that are visually/electronically stealthy but acoustically very noisy, such as small high-speed boats). Bad weather significantly degrades this duct's effect. All sub-launched missiles can be launched at this depth. Because of the small water pressure cavitation sets in even at relatively low throttle settings. This is the optimum depth for detection against surface ships (and counter-detection by hull sonars on surface ships, as well as "shallow"-set sonobuoys and dipping sonars).

Shallow: Some (but not all) sub-launched missiles can be launched at this depth. The surface duct is still present but significantly weaker. This depth is suitable for keeping an eye (or ear, as it goes) on surface traffic and staying near (or at) missile-launch conditions while keeping out of the strong surface duct (reduce counter-detection by air/surface) and keeping close to the layer below for a quick dive if necessary (ditto).

Just above layer: Completely out of the surface duct, this depth reduces interaction with air/surface matters even more, at the benefit of ASW. Indeed, this is the ideal hunting ground for ASW-oriented subs with towed arrays. This is because subs at this depth automatically trail their arrays _below_ the layer; thus they maximize their detection range against anything below while masking themselves with the layer (and retaining their above-layer search ability with their hull sonars). Cavitation speed is significantly higher here. No missiles can be used at this depth or below.

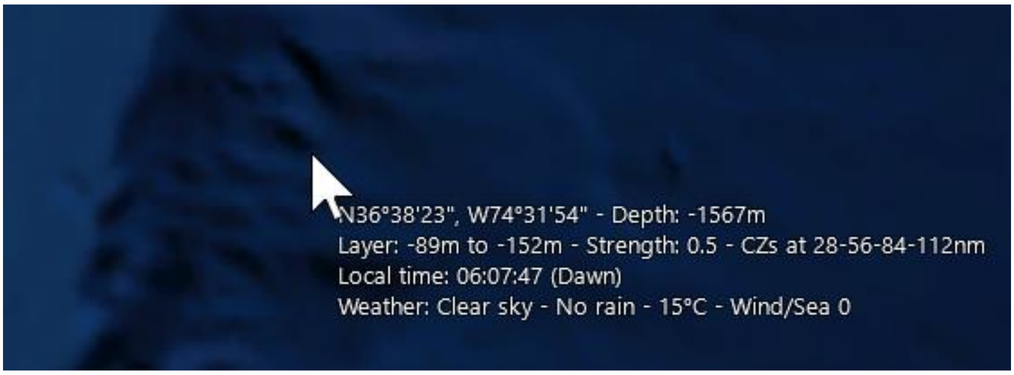
In-layer: Similar to above, the towed array hangs below the layer but the counter-detection reduction is not as great (sound has to go through a lot less to reach an enemy sensor). Also the unpredictable mix-up of warm and cold water at this depth range significantly reduces detection ranges against other subs also in the layer (think Mutara Nebula from Star Trek II).

Just under layer - the Deep Sound Channel: This is by far the most transmission-friendly depth band, greatly magnifying detection ranges. Surface ships and subs just above the layer both trail their VDS or towed arrays in this band. Sonobuoys and dipping sonars in "deep" setting also operate here. As a result, this is the worst place to be if you are hunted by modern ASW forces (or conversely, the best place to be if you want to attract attention). As a consolation, cavitation speed rises even higher. ASW-tasked subs without towed arrays often loiter in this band to maximize the range of their hull sonars. Surface and shallow sonars as well as "shallow"-set sonobuoys and dipping sonars have their ranges drastically cut against under-layer targets, and in some cases may not be able to get through at all.

The Great Deep: The DSC still has some influence here but not as great. Cavitation sets in only if you go flat-out (modern subs do not cavitate at all here, even at flank). If the sea bottom is shallower than the sub's rated depth, the sub can "fly nap of the earth" or even go belly-up (sit on the bottom) and get the benefit of greatly reduced active sonar echo. This is generally the ideal depth for "transit" mode, when the emphasis is on moving from A to B rather than hunting, or for stationary ambush.

Convergence zones

Convergence zone (CZ) detections are possible only if the local depth provides at least 600ft/200m clearance under the target. The local conditions for the presence (or not) of CZs are displayed on the map-cursor databox:



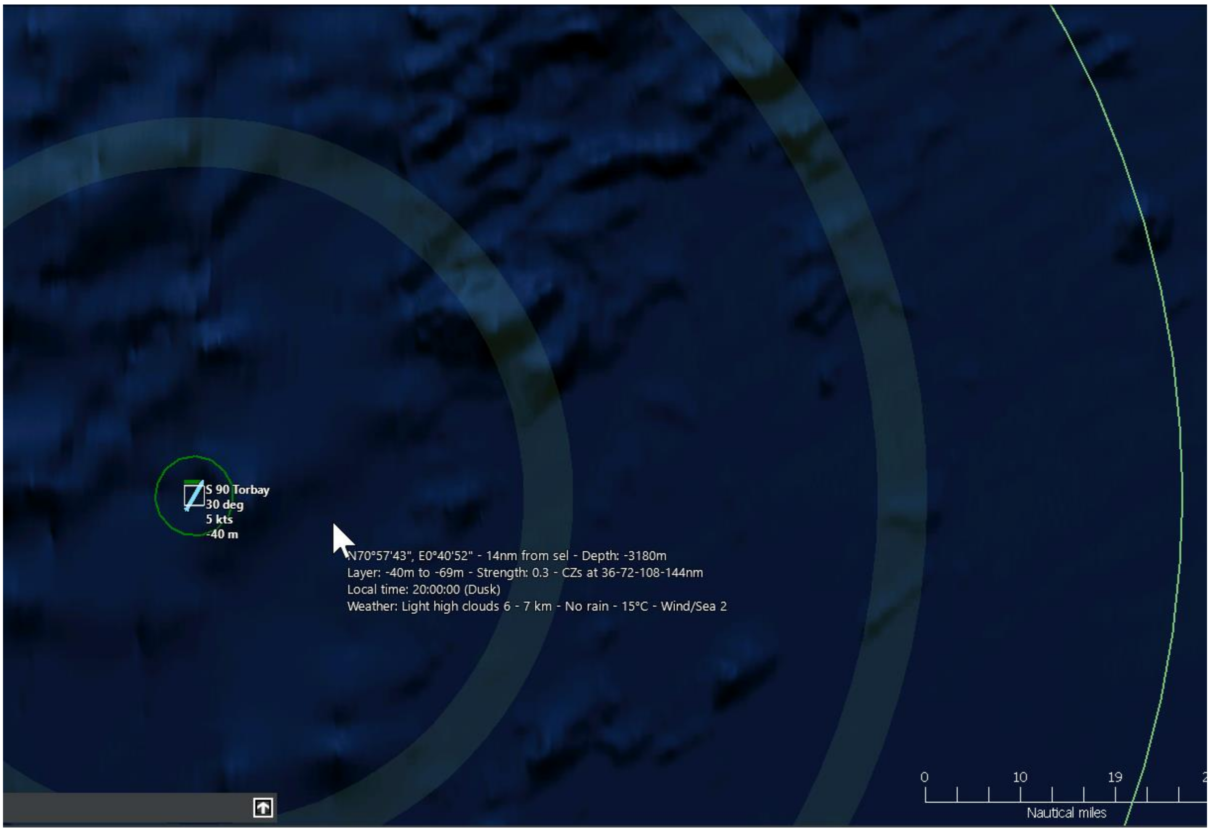
N36°38'23", W74°31'54" - Depth: -1567m
Layer: -89m to -152m - Strength: 0.5 - CZs at 28-56-84-112nm
Local time: 06:07:47 (Dawn)
Weather: Clear sky - No rain - 15°C - Wind/Sea 0

CZ intervals range from 40nm in the poles to 20nm in the equator, depending also on local temperature.

An arbitrary number of CZs are supported, if the sound is powerful enough and the receiver sensitive enough.

The CZ area depth (i.e., ring “thickness”) is assumed to always be 5nm regardless of range, so the actual detection range at each CZ interval may be plus/minus 2.5nm.

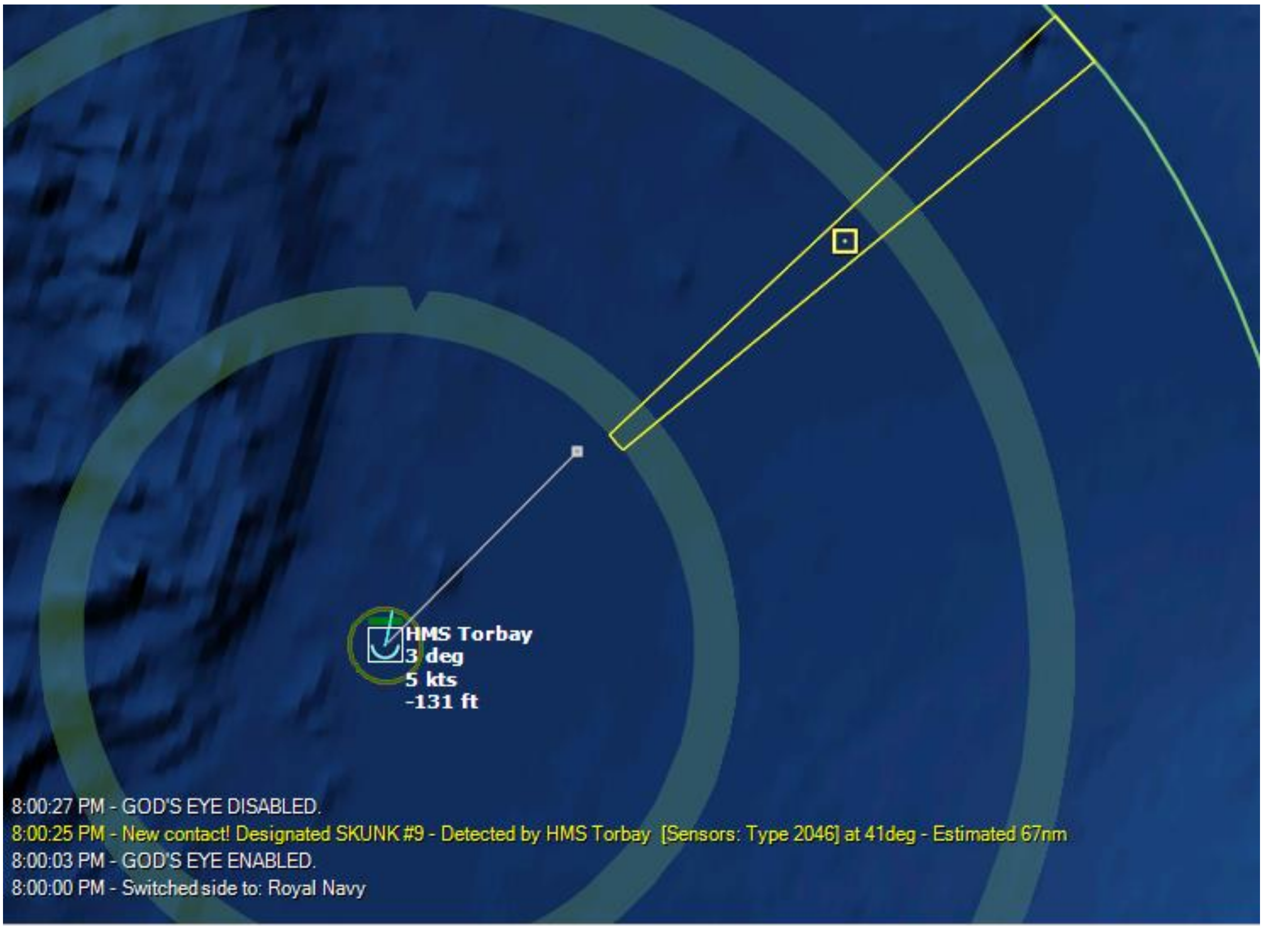
When the “Underwater sensors” map setting is enabled, if we select a unit that can make CZ detections (i.e., has suitably long-ranged sonar and is in deep enough water), the possible CZ rings for this unit are displayed on the map, as in this example:



N70°57'43", E0°40'52" - 14nm from sel - Depth: -3180m
Layer: -40m to -69m - Strength: 0.3 - CZs at 36-72-108-144nm
Local time: 20:00:00 (Dusk)
Weather: Light high clouds 6 - 7 km - No rain - 15°C - Wind/Sea 2

To verify if CZs can indeed form between the sensor and target, the simulation checks not only their respective local depths but also the depths at the nadirs of each CZ curve; so for example the sensor and the target may both have an abyss below them but if an underwater ridge high enough is between them it may well block the CZ path.

Sonar operators are assumed to be proficient enough to discriminate between a direct-path contact and a CZ one (by examining both aural tones and the bearing rate); as a result, the generated AOU will either start from the sensor platform and extend to max direct-path range, or (in the case of a CZ detection) start from the innermost CZ interval and extend to max sensor range. This is an example of a CZ detection as displayed:



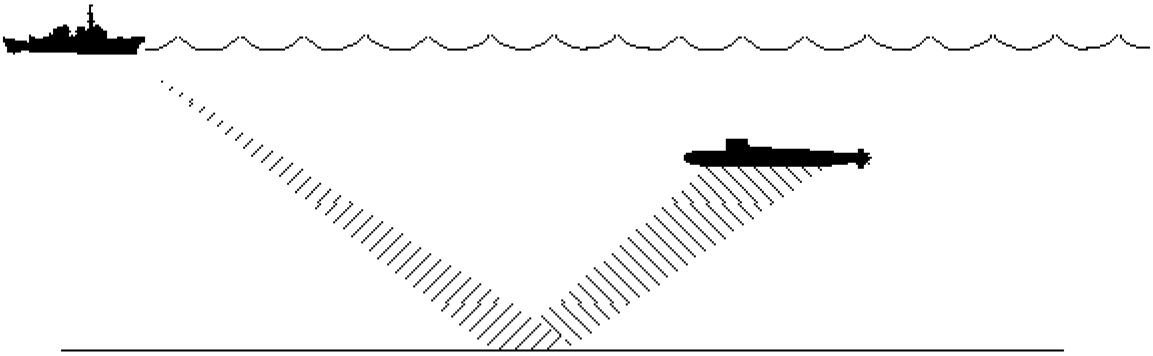
8:00:27 PM - GOD'S EYE DISABLED.
8:00:25 PM - New contact! Designated SKUNK #9 - Detected by HMS Torbay [Sensors: Type 2046] at 41deg - Estimated 67nm
8:00:03 PM - GOD'S EYE ENABLED.
8:00:00 PM - Switched side to: Royal Navy

Reverb, ambient noise & other factors

Noise propagation is also affected by reverberation and ambient noise. This is most evident in shallow environments, where there are more opportunities for sound waves to reflect and bend in unpredictable ways. Reverb affects more the lower-band frequencies (MF and especially HF sonars are more resistant). Ice environments also increase the background noise, making under-ice detections more difficult.

Sonar detections may be affected by target masking. When the sub/ship being detected is within X degrees bearing (relative to the sonar sensor) of another one, the (comparatively) louder one may prevent the other(s) from being detected.
(X = a given angular difference referred to as the "bearing gate". This value decreases in more modern sonar sets (as low as 4 degrees) and increases in older ones (as high as 20 deg). To fool a modern passive sonar by masking under another vessel on the same bearing, you need to get very close to the other unit angularly).

A potential way to get around the degradation of sound signal through the thermocline, is to use bottom bounce. This is used automatically by a sonar platform if suitable conditions (proper hunter-target vertical geometry, sufficiently flat sea bottom etc.) are met:



Bottom bounce offers up to a 20% increase in detection range over nominal direct-path under identical conditions. This will vary according to a number of factors such as the actual bottom slope (a rougher bottom will decrease the benefit).

Technological generation is one of the most crucial factors in a sonar sensor's ability to compensate for degradation from environmental and other factors. A given sonar may have a very high nominal range, but if its tech generation is declared as antiquated, the practical detection range may be severely degraded from the effects of reverberation, ambient noise, masking and other factors.

A practical example

Let us consider a simple example, to illustrate some of the fundamentals of the sonar model.

A Burke-class (Flt IIA - 2015) DDG attempts to passively detect a Type 093 Shang-class submarine (2007) at relatively deep water (686m bottom depth), using its hull SQS-53C(V)1 sonar. The Burke is coming off a sprint and drifting, current speed is 20 knots and falling, throttle is set to creep (5 knots). The Shang is steady at 10 knots, just under the layer (175m depth). The Burke is trailing the Shang at 8.48nm. Local conditions are: Heavy rain, sea state 4, 53%-strong thermal layer between 98-167m, CZs forming at 27-54-81-108nm.

* The sonar bow has LOS to the sub.

* The Burke is drifting with throttle to creep or less, so powerplant noise modifier is essentially minimum and flow noise modifier is 0.45. The total "own noise" modifier is 0.88.

* The Shang has a baseline LF rear-aspect sound signature of 82 vs the reference maximum of 112 (The acoustic signature values are in decibels). The powerplant noise modifier is 0.5 and the flow noise modifier is 0.3, for a total noise modifier of 0.458. The sub is not cavitating. Its "actual" (vs. nominal) sound signature is 0.73

* Taking all the above numbers into account, the sonar's actual detection range is 40nm (max theoretical) * 0.88 (own noise) * 0.458 (target noise) * 0.73 (target noise ratio) = 11.88nm.

* Now we go into propagation and reverb modifiers. No surface ducting in this case (target is too deep), and the thermal layer sharply reduces the detection range by 0.47. No modifiers for shallow water or under-ice here, so the actual feasible detection range is now just 5.63nm. Actual direct-path detection range is 4.68 (nominal 9.5nm affected by the same propagation and reverb modifiers). This is much lower than the 8.48nm distance, so the SQS-53(V)1 sonar does not detect the Shang under these conditions.

Let us check if the destroyer's SQR-20 MFTA towed array fares better (it should, it hangs below the layer and the sub is right in the deep sound channel area): Same own- and target-noise modifiers and base signatures as above. Theoretical max range: 70nm. Practical range after noise modifiers: ~22nm. Not only is the array not degraded by the thermal layer, but the DSC actually grants it a 2x bonus. So detection range jumps to a whopping ~45nm. Direct-path range is similarly boosted by the DSC, and the array has no problem picking up the Shang.

One way to make life quite hard for the DDG would be to place both boats on shallow water. No towed array capability there, and the reverb would sharply cut down practical detection ranges for the LF bow sonar (plus, the sub would be able to hide itself against active pinging by hugging the bottom).

Bring your own model: Mechanics Overrides

The sonar model is extensible and replaceable through the "Mechanics Overrides" feature (see relevant section on this manual). The database sensor annex already includes several fields for properties not used by the default model but used by other popular models such as Odin or Kraken. Customers interested in extending or replacing the default sonar model should contact the Command-PE development team to discuss the most beneficial modification for their needs.

Appendix 3: The Amphibious Planner, Operation Planner and Serial Editor

While this feature uses known military concepts, in practice some terminologies may have a slightly different meanings within the Command simulation, to fit gameplay & analysis requirements, and accommodate with the current state of the simulation.

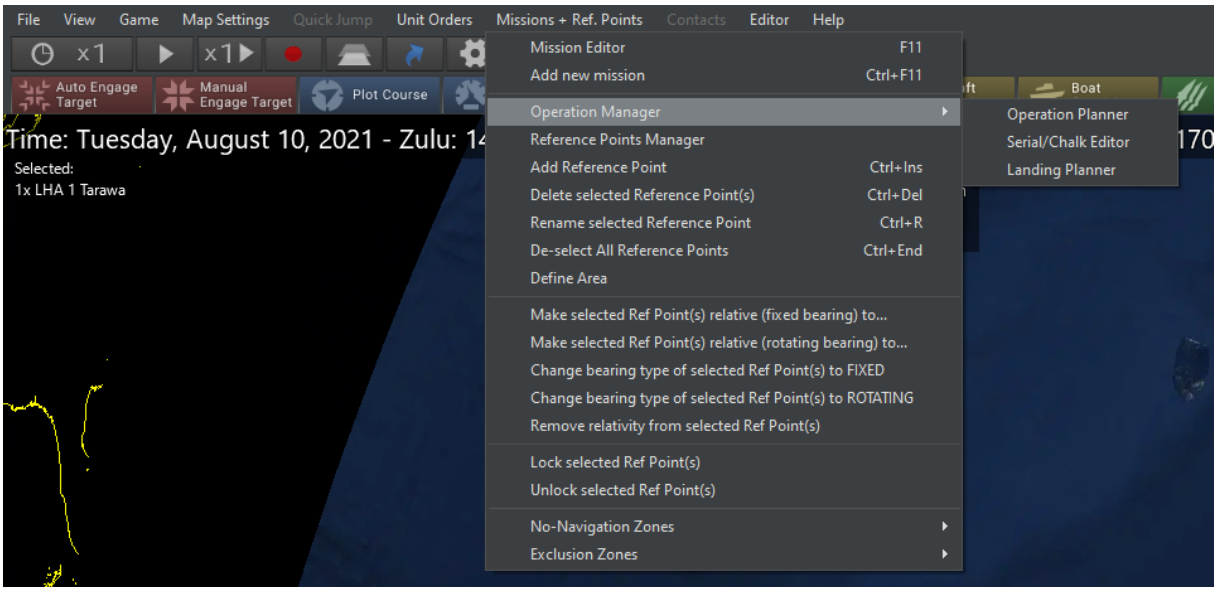
To fully comprehend this extension, it is essential to have a good knowledge of the mission editor in general, and cargo missions in particular.

SUMMARY

- Getting Started
- An amphibious operation from start to end
- Creating zones
- Creating Serials and Chalks
- Creating your landing plan
- Understanding the concept of multi-mission units
- Operation Planner
- Triggers
- Logical Operators
- Lua scripting
- Working with estimation
- Lua Reference
- Import/Export
- Nomenclature

Getting started

You can find the **Landing planner**, the **Serial Editor**, and the **Operation Planner** under the **Missions + Ref. Points -> Operation Manager** menu.


File
View
Game
Map Settings
Quick Jump
Unit Orders
Missions + Ref. Points
Contacts
Editor
Help
x1
x1
Auto Engage Target
Manual Engage Target
Plot Course
Boat
Time: Tuesday, August 10, 2021 - Zulu: 14
Selected:
1x LHA 1 Tarawa
Mission Editor F11
Add new mission Ctrl+F11
Operation Manager
Reference Points Manager
Add Reference Point Ctrl+Ins
Delete selected Reference Point(s) Ctrl+Del
Rename selected Reference Point Ctrl+R
De-select All Reference Points Ctrl+End
Define Area
Make selected Ref Point(s) relative (fixed bearing) to...
Make selected Ref Point(s) relative (rotating bearing) to...
Change bearing type of selected Ref Point(s) to FIXED
Change bearing type of selected Ref Point(s) to ROTATING
Remove relativity from selected Ref Point(s)
Lock selected Ref Point(s)
Unlock selected Ref Point(s)
No-Navigation Zones
Exclusion Zones
Operation Planner
Serial/Chalk Editor
Landing Planner

An amphibious operation from start to finish

The best way to learn the functionalities of this extension is to play with an actual test case.

You can find a scenario called “**Quickstart Landing Planner**” in the “[Program data]\Scenarios\Tutorials\Other tutorials” folder. This is a prebuilt scenario with an existing landing force.

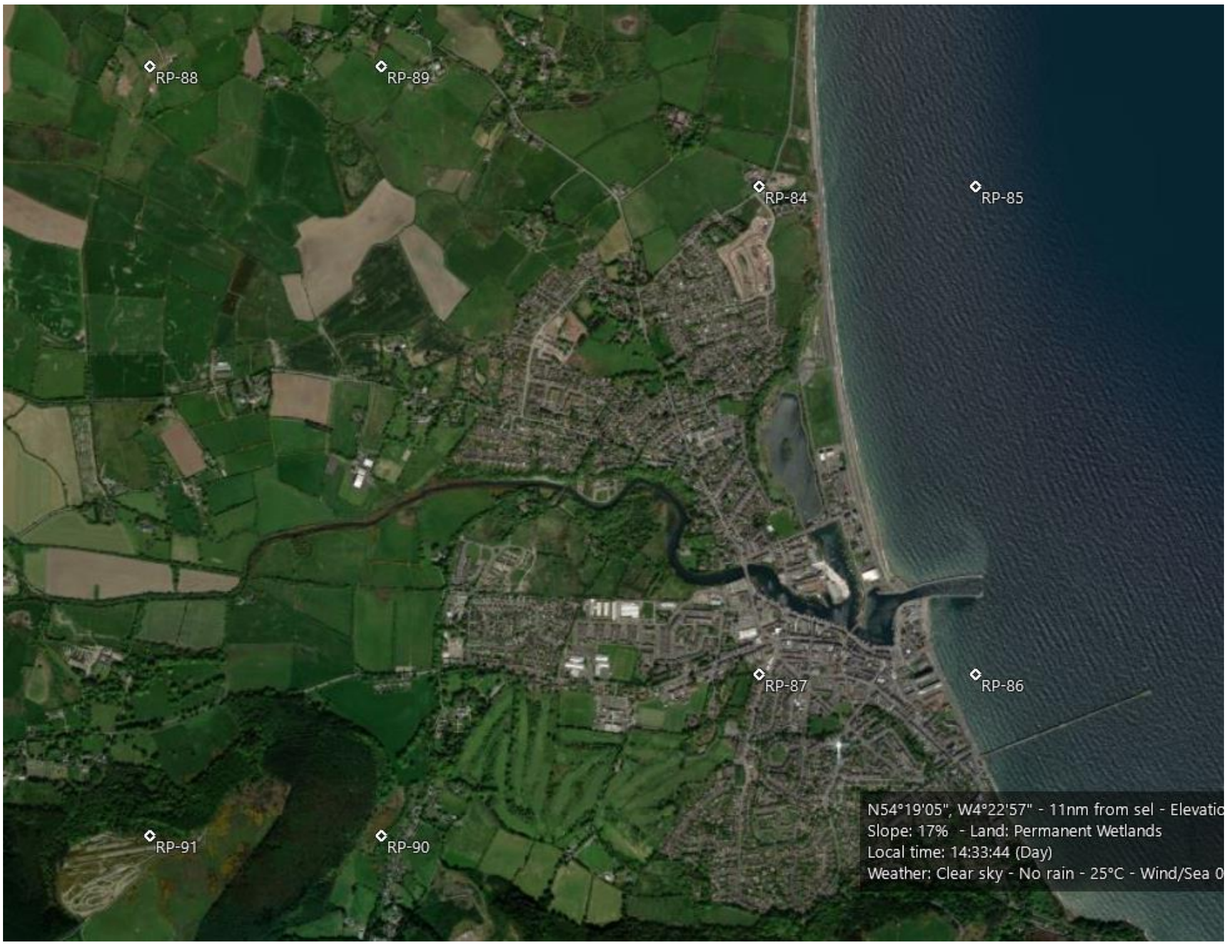
Open this scenario to get started. We will follow a procedure to create a full landing operation and invade the Isle of Man!

Creating zones

The operation planer comes with a new feature allowing you to create stand-alone zones. The first thing to do is to layout your invasion strategy by designating zones. These zones in the future will either be set as **LZ** (Airborne Landing Zone) or **CLZ** (Amphibious landing zone.)

Create a landing zone along the shore for your CLZ (a CLZ should containing both water and land) and a zone inland for your LZ.

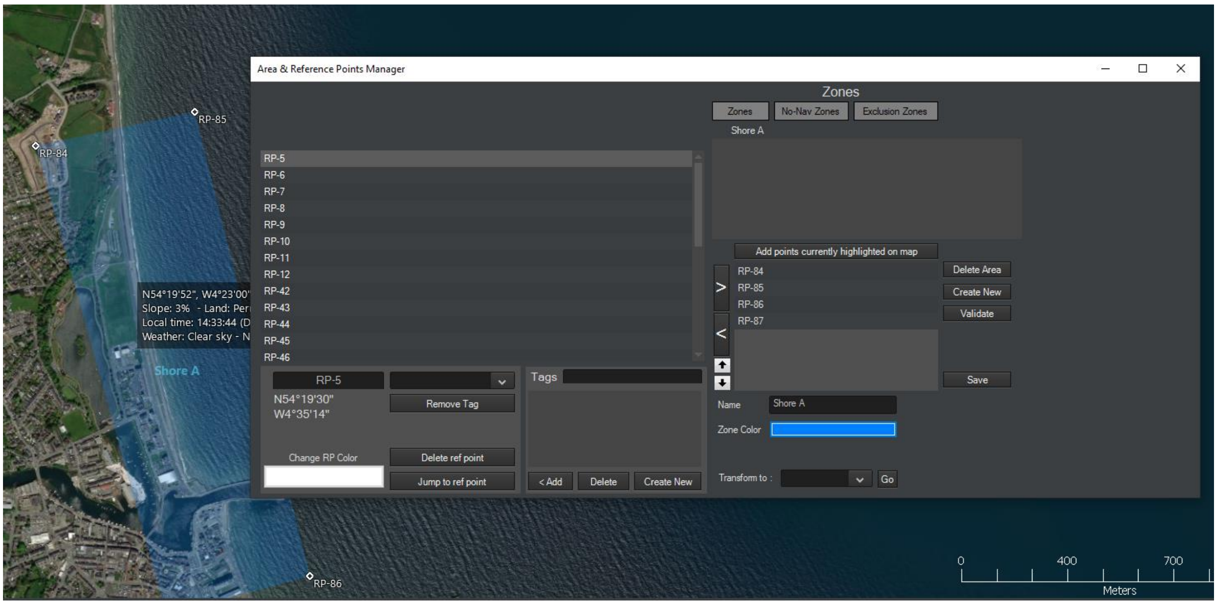
Zones are defined by three or more reference points. To create a new zone, go to the **Missions + Ref. Points -> Define Area** menu item and place your reference points.



N54°19'05", W4°22'57" - 11nm from sel - Elevatio
Slope: 17% - Land: Permanent Wetlands
Local time: 14:33:44 (Day)
Weather: Clear sky - No rain - 25°C - Wind/Sea 0

Go to **Missions + Ref. Points -> Reference Points Manager**

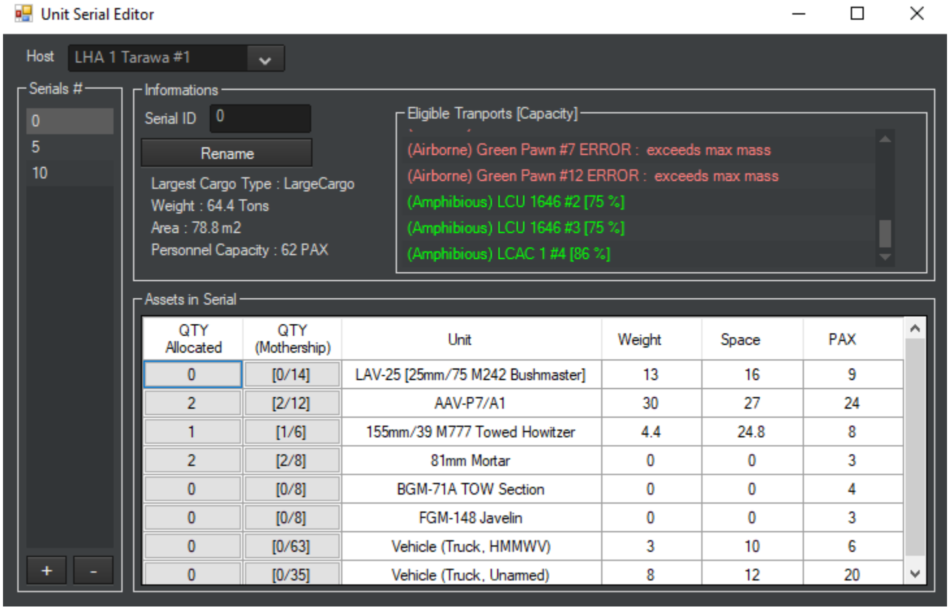
Now select a group of reference points to create your zone from. As shown in the screenshot below, click on "**Add points currently highlighted on map**" and then rename your zone to something specific and select a new color.



You have now defined a new zone. Create as many zones as needed.

Creating Serials and Chalks

A serial defines a group of existing units from a common mothership that will be landed together using a cargo mission. Serials allow you to organize your landing manifest while generating your landing plan through the landing planner. Serials can only be modified, created, or removed from the serials editor. They are not used in the rest of the simulation, outside of the landing planner. A chalk is similar to a serial but is used for an airborne landing instead of a seaborne one. Therefore, the terms serial and chalk are conceptually identical in the Command simulation, and, for clarity, the term “serial” will refer to both in this manual.



Host LHA 1 Tarawa #1

Serial ID 0

Largest Cargo Type : LargeCargo
Weight : 64.4 Tons
Area : 78.8 m2
Personnel Capacity : 62 PAX

(Airborne) Green Pawn #7 ERROR : exceeds max mass
(Airborne) Green Pawn #12 ERROR : exceeds max mass
(Amphibious) LCU 1646 #2 [75 %]
(Amphibious) LCU 1646 #3 [75 %]
(Amphibious) LCAC 1 #4 [86 %]

QTY Allocated	QTY (Mothership)	Unit	Weight	Space	PAX
0	[0/14]	LAV-25 [25mm/75 M242 Bushmaster]	13	16	9
2	[2/12]	AAV-P7/A1	30	27	24
1	[1/6]	155mm/39 M777 Towed Howitzer	4.4	24.8	8
2	[2/8]	81mm Mortar	0	0	3
0	[0/8]	BGM-71A TOW Section	0	0	4
0	[0/8]	FGM-148 Javelin	0	0	3
0	[0/63]	Vehicle (Truck, HMMWV)	3	10	6
0	[0/35]	Vehicle (Truck, Unarmed)	8	12	20

Go to **Missions + Ref. Points -> Operation Manager -> Serial/Chalk Editor**

In this scenario we have a fleet of 2 LHA 1 Tarawa, these are amphibious assault ships transporting troops and cargo, landing crafts and aircrafts. In this manual we will call them **"Mothership".**

The first thing to do is to select a mothership - a "Host".

Select one of the LHA.

On the left, you can see all the serials on this mothership. We currently have no serial.

Click on the "+" button to add one.

Serials ID start at 0 and increment by 5 each time you add one. This is for organizational purpose, in case you'd like to add serials in-between existing serials. A serial's ID is used to reference that serial in the landing planner but has no other significance.

In the "Eligible transports" panel you will see the full list of landing crafts. The percentage at the end is how much (max value of area/weight/pax) the current serial is taking. 375% means that it will theoretically take 4 waves to deliver this serial with the given transport.

Sometimes a transport will be unable to transport a given serial. When this occurs, the transport will be shown in red, and the serial editor will report why the transport is impossible.

The "Information" panel displays the details of the current serial. It records the total weight, area and pax count and the largest cargo type in the serial.

The "Assets in serial" panel displays all of the individual units loaded in the mothership. Each line represents a unit type.

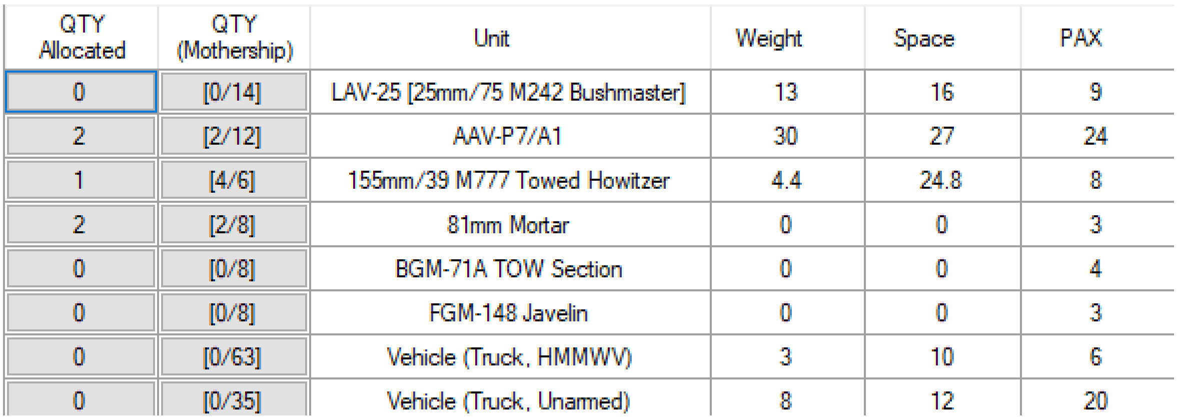
To allocate 1 unit to the current serial, click the corresponding button in the "QTY Allocated" column.

To remove 1 unit from the current serial, click the corresponding button in the "QTY (Mothership)" column.

In the example below we have allocated to the current serial:

- 2 AAV-P7
- 1 Towed Howitzer (155mm)
- 2 Mortar (81mm)

Note the "[4/6]" value for the 155 Howitzer under "QTY (Mothership)" -- this means that a total of 6 howitzers are aboard the mothership, with 4 out of 6 howitzers allocated across all serials, leaving 2 howitzers not yet allocated to a serial.



QTY Allocated	QTY (Mothership)	Unit	Weight	Space	PAX
0	[0/14]	LAV-25 [25mm/75 M242 Bushmaster]	13	16	9
2	[2/12]	AAV-P7/A1	30	27	24
1	[4/6]	155mm/39 M777 Towed Howitzer	4.4	24.8	8
2	[2/8]	81mm Mortar	0	0	3
0	[0/8]	BGM-71A TOW Section	0	0	4
0	[0/8]	FGM-148 Javelin	0	0	3
0	[0/63]	Vehicle (Truck, HMMWV)	3	10	6
0	[0/35]	Vehicle (Truck, Unarmed)	8	12	20

Create as many serials as needed and populate them with units.

Try to make efficient serial depending on how you'd like to land the units. Keep an eye out for serials that can't be transported at all.

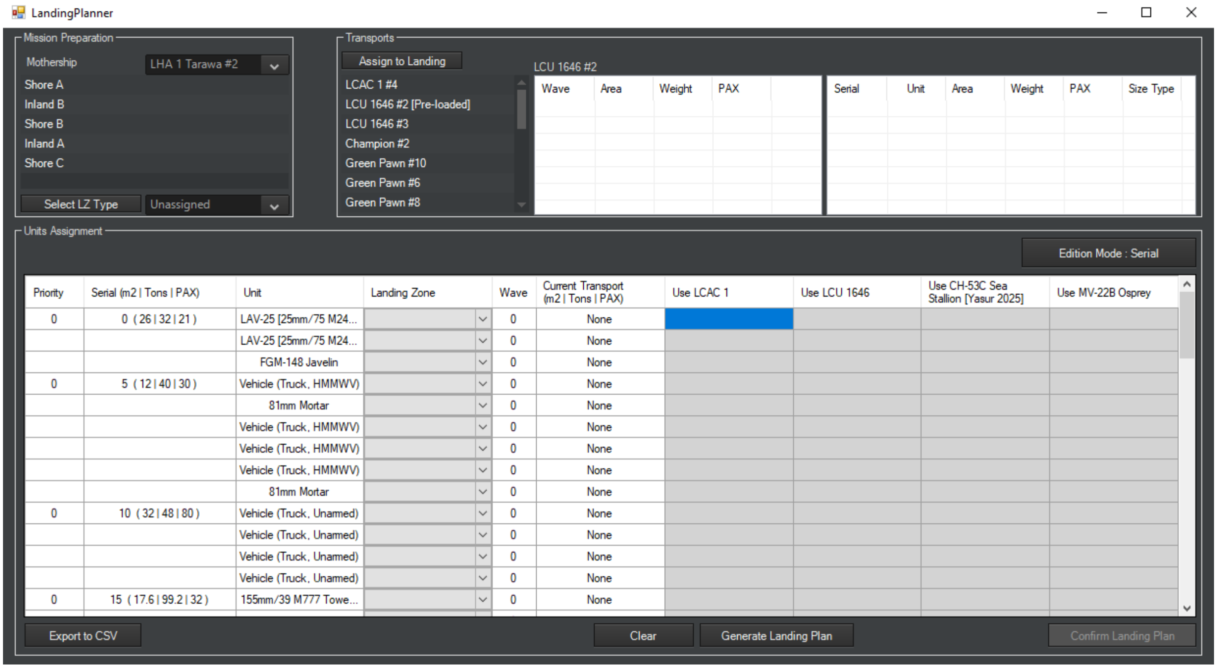
Creating your landing plan

You have your zones.

You have serials.

Now it is time to generate your actual landing plan.

Go to **Missions + Ref. Points -> Operation Manager -> Landing Planner**



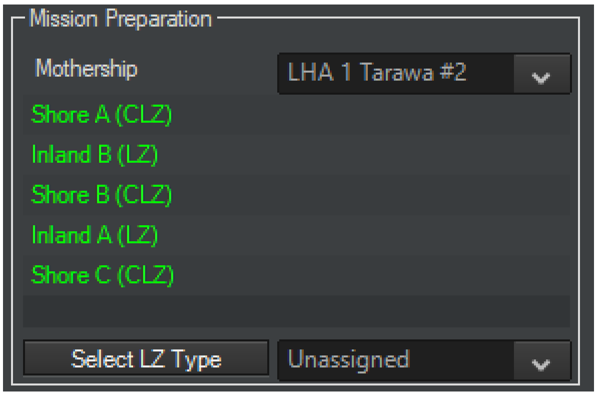
Mothership: LHA 1 Tarawa #2

Priority	Serial (m2 | Tons | PAX)	Unit	Landing Zone	Wave	Current Transport (m2 | Tons | PAX)	Use LCAC 1	Use LCU 1646	Use CH-53C Sea Stallion [Yasur 2025]	Use MV-22B Osprey
0	0 (26 | 32 | 21)	LAV-25 [25mm/75 M24...		0	None				
		LAV-25 [25mm/75 M24...		0	None				
		FGM-148 Javelin		0	None				
0	5 (12 | 40 | 30)	Vehicle (Truck, HMMWV)		0	None				
		81mm Mortar		0	None				
		Vehicle (Truck, HMMWV)		0	None				
		Vehicle (Truck, HMMWV)		0	None				
		Vehicle (Truck, HMMWV)		0	None				
		81mm Mortar		0	None				
0	10 (32 | 48 | 80)	Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
0	15 (17.6 | 99.2 | 32)	155mm/39 M777 Towe...		0	None				

Choose the mothership for this landing plan. A mothership will have a single landing plan.

Then prepare your zones and assign them a type. Designate your amphibious landing zone as “CLZ” and your airborne landing zones as “LZ”.

To designate a zone type, select the zone in the zone list, select the type in the drop down list of types, and then click the “Select LZ Type” button.

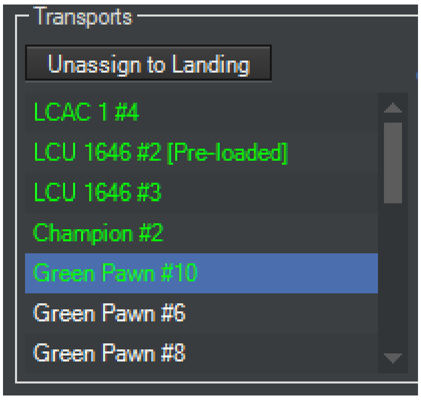


Mission Preparation

Mothership: LHA 1 Tarawa #2

Shore A (CLZ)
Inland B (LZ)
Shore B (CLZ)
Inland A (LZ)
Shore C (CLZ)

Select LZ Type | Unassigned



Transports

Unassign to Landing

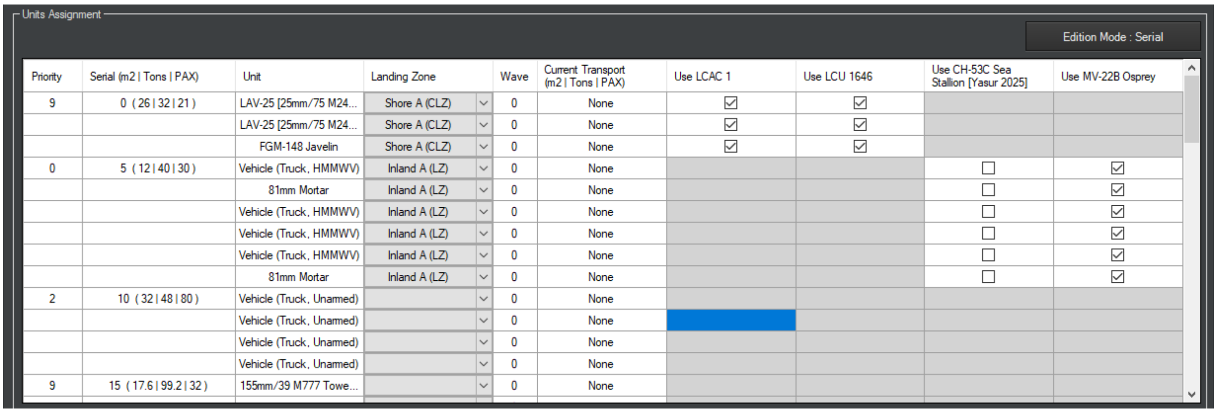
LCAC 1 #4
LCU 1646 #2 [Pre-loaded]
LCU 1646 #3
Champion #2
Green Pawn #10
Green Pawn #6
Green Pawn #8

Once the zones have been designated for your landing plan, assign the transport you wish to allocate.

To allocate a transport, click on the transport unit and then click the “Assign to Landing” button. To unallocate a transport, click the transport and then click “Unassign to Landing.”

Allocating a transport allows the landing planner to consider using that transport when you generate a landing plan. It does not guarantee the transport will be used for the landing.

As a rule of thumb, assign as many transports as possible if you want to deliver your units as fast as possible, in a minimal numbers of waves. But the landing planner leaves you the possibility to keep some reserve units.



Units Assignment

Edition Mode : Serial

Priority	Serial (m2 | Tons | PAX)	Unit	Landing Zone	Wave	Current Transport (m2 | Tons | PAX)	Use LCAC 1	Use LCU 1646	Use CH-53C Sea Stallion [Yasur 2025]	Use MV-22B Osprey
9	0 (26 | 32 | 21)	LAV-25 [25mm/75 M24...	Shore A (CLZ)	0	None	☑	☑		
		LAV-25 [25mm/75 M24...	Shore A (CLZ)	0	None	☑	☑		
		FGM-148 Javelin	Shore A (CLZ)	0	None	☑	☑		
0	5 (12 | 40 | 30)	Vehicle (Truck, HMMWV)	Inland A (LZ)	0	None			☐	☑
		81mm Mortar	Inland A (LZ)	0	None			☐	☑
		Vehicle (Truck, HMMWV)	Inland A (LZ)	0	None			☐	☑
		Vehicle (Truck, HMMWV)	Inland A (LZ)	0	None			☐	☑
		Vehicle (Truck, HMMWV)	Inland A (LZ)	0	None			☐	☑
		81mm Mortar	Inland A (LZ)	0	None			☐	☑
2	10 (32 | 48 | 80)	Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
		Vehicle (Truck, Unarmed)		0	None				
9	15 (17.6 | 99.2 | 32)	155mm/39 M777 Towe...		0	None				

The units Assignment panel lists all your serials, with their contained units. Each line represents a unit within a serial.

Priority

The priority column is where you decide which serial should be transported first. The highest priority is 0, and the lowest is 9. The landing planner generator will take this into account during computation.

Landing Zone

Each serial has a designated landing zone. You decide here where the serial should be delivered.

Transport types

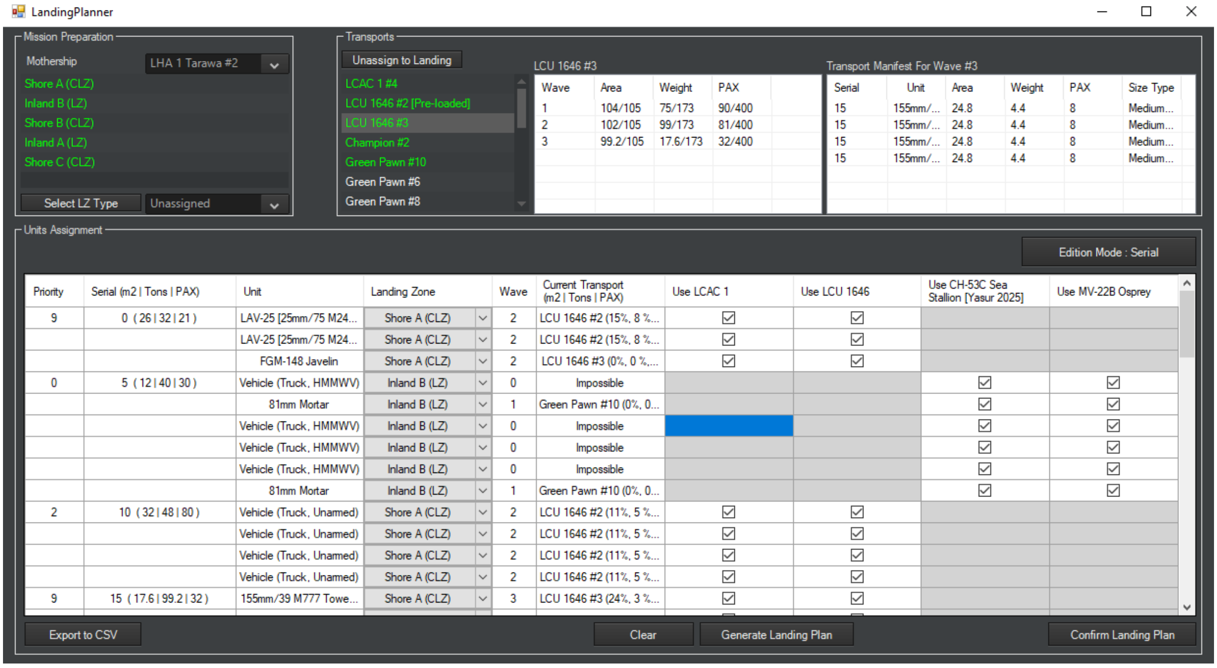
The remaining columns are populated with the different types of transports allocated to the landing plan. Note these are transport types, not individual transport units.

You can designate which transport type(s) can be used to transport any given unit in a serial. Allowing multiple transport type usually allows the landing planner to generate a more efficient landing plan.

Designate a landing zone for each serial, and optionally, tweak the priority and the allowed transport type on the right columns.

When you are happy with the configuration, click on "Generate Landing Plan".

A draft of the landing plan will be generated but not confirmed. At this point you can still make changes and click "Generate Landing Plan" again to draft a new plan.



Mothership: LHA 1 Tarawa #2

Shore A (CLZ)
Inland B (LZ)
Shore B (CLZ)
Inland A (LZ)
Shore C (CLZ)

Select LZ Type: Unassigned

LCAC 1 #4
LCU 1646 #2 [Pre-loaded]
LCU 1646 #3
Champion #2
Green Pawn #10
Green Pawn #6
Green Pawn #8

LCU 1646 #3

Wave	Area	Weight	PAX
1	104/105	75/173	90/400
2	102/105	99/173	81/400
3	99.2/105	17.6/173	32/400

Transport Manifest For Wave #3

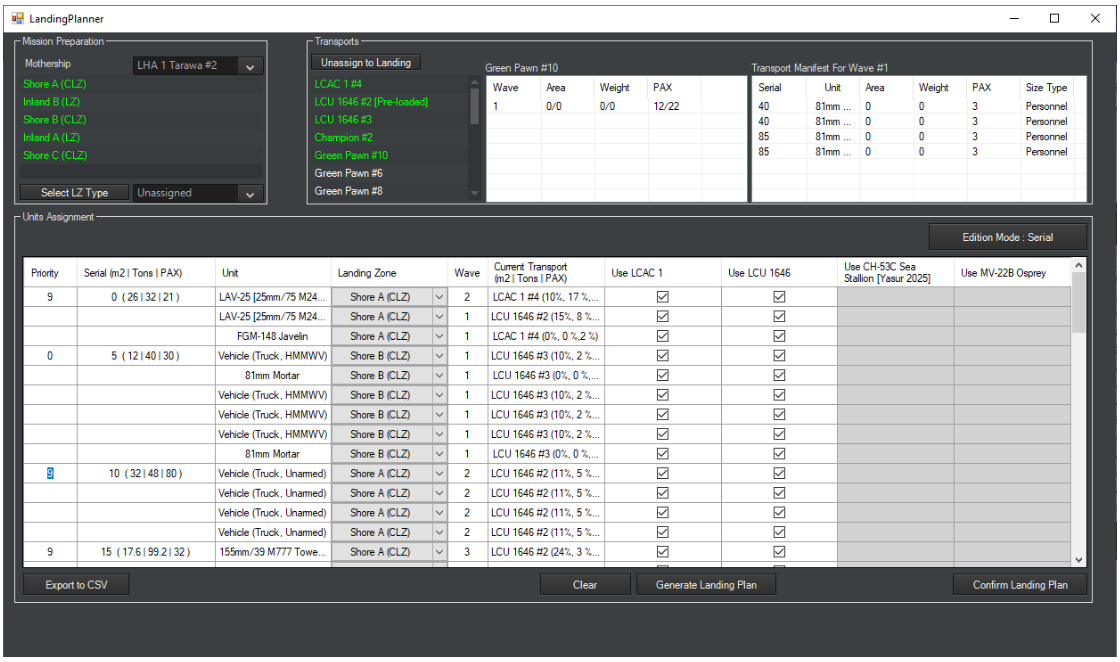
Serial	Unit	Area	Weight	PAX	Size Type
15	155mm/...	24.8	4.4	8	Medium...
15	155mm/...	24.8	4.4	8	Medium...
15	155mm/...	24.8	4.4	8	Medium...
15	155mm/...	24.8	4.4	8	Medium...

Priority	Serial (m2 | Tons | PAX)	Unit	Landing Zone	Wave	Current Transport (m2 | Tons | PAX)	Use LCAC 1	Use LCU 1646	Use CH-53C Sea Stallion [Yasur 2025]	Use MV-22B Osprey
9	0 (26 | 32 | 21)	LAV-25 [25mm/75 M24...	Shore A (CLZ)	2	LCU 1646 #2 (15%, 8 %...	☑	☑		
		LAV-25 [25mm/75 M24...	Shore A (CLZ)	2	LCU 1646 #2 (15%, 8 %...	☑	☑		
		FGM-148 Javelin	Shore A (CLZ)	2	LCU 1646 #3 (0%, 0 %,...	☑	☑		
0	5 (12 | 40 | 30)	Vehicle (Truck, HMMWV)	Inland B (LZ)	0	Impossible			☑	☑
		81mm Mortar	Inland B (LZ)	1	Green Pawn #10 (0%, 0...			☑	☑
		Vehicle (Truck, HMMWV)	Inland B (LZ)	0	Impossible			☑	☑
		Vehicle (Truck, HMMWV)	Inland B (LZ)	0	Impossible			☑	☑
		Vehicle (Truck, HMMWV)	Inland B (LZ)	0	Impossible			☑	☑
		81mm Mortar	Inland B (LZ)	1	Green Pawn #10 (0%, 0...			☑	☑
2	10 (32 | 48 | 80)	Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
9	15 (17.6 | 99.2 | 32)	155mm/39 M777 Towe...	Shore A (CLZ)	3	LCU 1646 #3 (24%, 3 %...	☑	☑		

The landing planner did all the heavy thinking and optimized the manifests as much as possible, taking priority into account.

The landing planner will also warn you of errors. Look at the landing plan above and note that the "Current Transport" has been marked as "Impossible" where we have a serial with trucks assigned to our Ospreys and CH-53C.

Since serial #5 would be better transported aboard an LCU or an LCAC, change the landing zone to "Shore B (CLZ)" and click on "Generate Landing Plan" again.



LandingPlanner

Mission Preparation

Mothership: LHA 1 Tarawa #2

- Shore A (CLZ)
- Inland B (LZ)
- Shore B (CLZ)
- Inland A (LZ)
- Shore C (CLZ)

Select LZ Type: Unassigned

Transports

Unassign to Landing

- LCAC 1 #4
- LCU 1646 #2 [Pre-loaded]
- LCU 1646 #3
- Champion #2
- Green Pawn #10
- Green Pawn #6
- Green Pawn #8

Green Pawn #10

Wave	Area	Weight	PAX
1	0/0	0/0	12/22

Transport Manifest For Wave #1

Serial	Unit	Area	Weight	PAX	Size Type
40	81mm ...	0	0	3	Personnel
40	81mm ...	0	0	3	Personnel
85	81mm ...	0	0	3	Personnel
85	81mm ...	0	0	3	Personnel

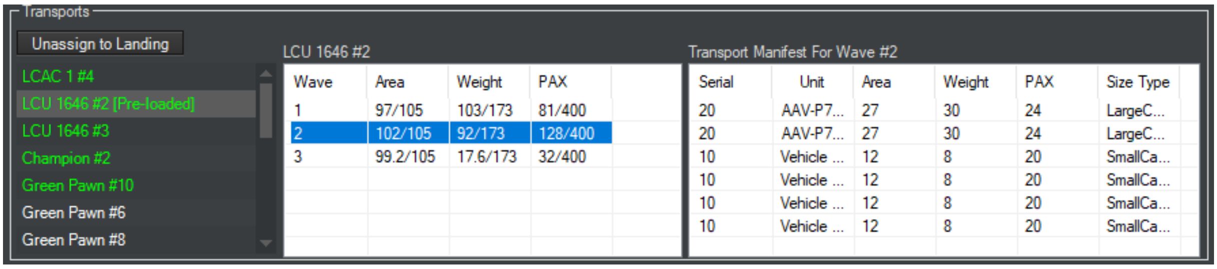
Units Assignment

Edition Mode : Serial

Priority	Serial (m2 | Tons | PAX)	Unit	Landing Zone	Wave	Current Transport (m2 | Tons | PAX)	Use LCAC 1	Use LCU 1646	Use CH-53C Sea Stallion [Yasur 2025]	Use MV-22B Osprey
9	0 (26 | 32 | 21)	LAV-25 [25mm/75 M24...	Shore A (CLZ)	2	LCAC 1 #4 (10%, 17 %,...	☑	☑		
		LAV-25 [25mm/75 M24...	Shore A (CLZ)	1	LCU 1646 #2 (15%, 8 %...	☑	☑		
		FGM-148 Javelin	Shore A (CLZ)	1	LCAC 1 #4 (0%, 0 %,2 %)	☑	☑		
0	5 (12 | 40 | 30)	Vehicle (Truck, HMMWV)	Shore B (CLZ)	1	LCU 1646 #3 (10%, 2 %...	☑	☑		
		81mm Mortar	Shore B (CLZ)	1	LCU 1646 #3 (0%, 0 %,...	☑	☑		
		Vehicle (Truck, HMMWV)	Shore B (CLZ)	1	LCU 1646 #3 (10%, 2 %...	☑	☑		
		Vehicle (Truck, HMMWV)	Shore B (CLZ)	1	LCU 1646 #3 (10%, 2 %...	☑	☑		
		Vehicle (Truck, HMMWV)	Shore B (CLZ)	1	LCU 1646 #3 (10%, 2 %...	☑	☑		
		81mm Mortar	Shore B (CLZ)	1	LCU 1646 #3 (0%, 0 %,...	☑	☑		
9	10 (32 | 48 | 80)	Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
		Vehicle (Truck, Unarmed)	Shore A (CLZ)	2	LCU 1646 #2 (11%, 5 %...	☑	☑		
9	15 (17.6 | 99.2 | 32)	155mm/39 M777 Towe...	Shore A (CLZ)	3	LCU 1646 #2 (24%, 3 %...	☑	☑		

The result is now a viable landing plan.

We also have our transport manifests generated now. Click on an individual transport and then select a wave to see the detailed manifest for this specific transport and wave.



Transports

Unassign to Landing

- LCAC 1 #4
- LCU 1646 #2 [Pre-loaded]
- LCU 1646 #3
- Champion #2
- Green Pawn #10
- Green Pawn #6
- Green Pawn #8

LCU 1646 #2

Wave	Area	Weight	PAX
1	97/105	103/173	81/400
2	102/105	92/173	128/400
3	99.2/105	17.6/173	32/400

Transport Manifest For Wave #2

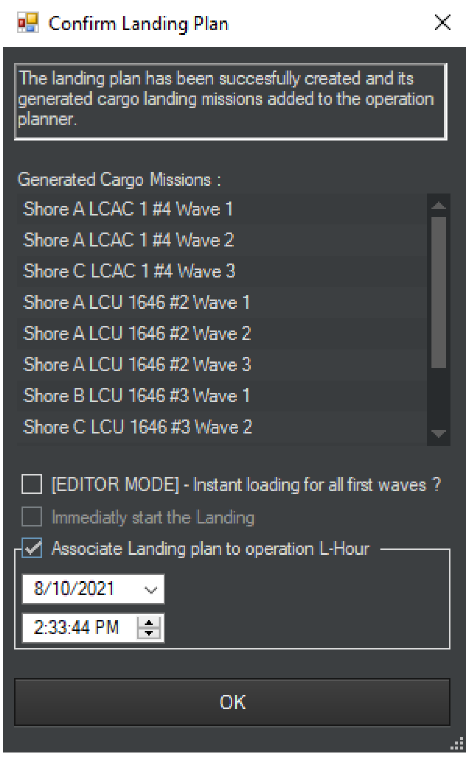
Serial	Unit	Area	Weight	PAX	Size Type
20	AAV-P7...	27	30	24	LargeC...
20	AAV-P7...	27	30	24	LargeC...
10	Vehicle ...	12	8	20	SmallCa...
10	Vehicle ...	12	8	20	SmallCa...
10	Vehicle ...	12	8	20	SmallCa...
10	Vehicle ...	12	8	20	SmallCa...

Do a final check of your landing plan and try to find anything that looks incorrect.

Once you are satisfied with the plan, click on the “Confirm Landing Plan” button.

Pre-Boating special case

The landing planner support special cases when the user decides to load the landing craft prior to the landing operation. These chalks will appear as “pre-boated”. It is necessary to assign them a landing zone, and for obvious reason, the first wave will be dedicated for the pre-boated chalk units.



Confirm Landing Plan

The landing plan has been succesfully created and its generated cargo landing missions added to the operation planner.

Generated Cargo Missions :

Shore A LCAC 1 #4 Wave 1
Shore A LCAC 1 #4 Wave 2
Shore C LCAC 1 #4 Wave 3
Shore A LCU 1646 #2 Wave 1
Shore A LCU 1646 #2 Wave 2
Shore A LCU 1646 #2 Wave 3
Shore B LCU 1646 #3 Wave 1
Shore C LCU 1646 #3 Wave 2

☐ [EDITOR MODE] - Instant loading for all first waves ?

☐ Immediatly start the Landing

☑ Associate Landing plan to operation L-Hour

8/10/2021

2:33:44 PM

A pop-up will open listing all the generated cargo missions.

Each wave is a Command cargo mission.

3 Options are available in this panel:

Instant loading for all first wave?

This editor-only option will make sure that the first wave of every transport is instantaneously embarked. This is used to quickly simulate pre-boating.

Immediately start the landing

The whole landing plan will execute immediately.

Associate Landing plan to operation L-Hour

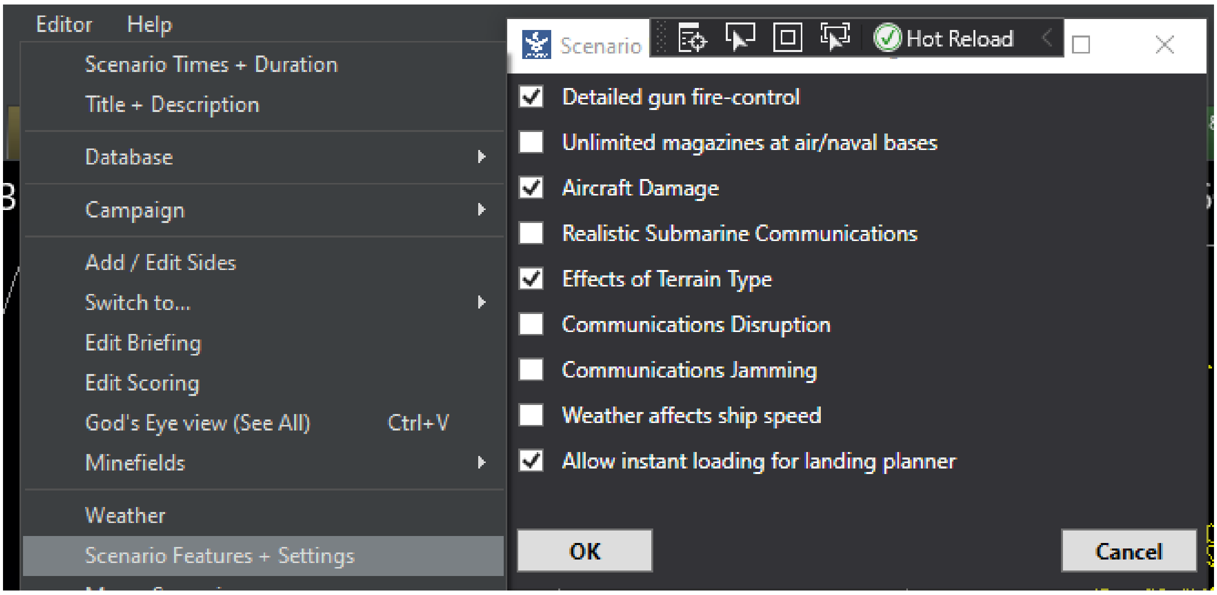
Automatically creates a Master Landing Plan that will act as the L-Hour initial mission in the operation. Meaning that this landing plan will start at the defined L-Hour date and time.

Allowing instant loading for regular player

If you want to allow regular player (non-editor users) to be able to select the immediate landing at first wave feature, you can change this setting in **Editor>Scenario Features + Setting.**

Enable **Allow instant loading for landing planner,** click on OK and make sure to save the scenario.

Any player using this scenario will be able to load instantaneously the cargo for the first wave.



☑ Detailed gun fire-control

☐ Unlimited magazines at air/naval bases

☑ Aircraft Damage

☐ Realistic Submarine Communications

☑ Effects of Terrain Type

☐ Communications Disruption

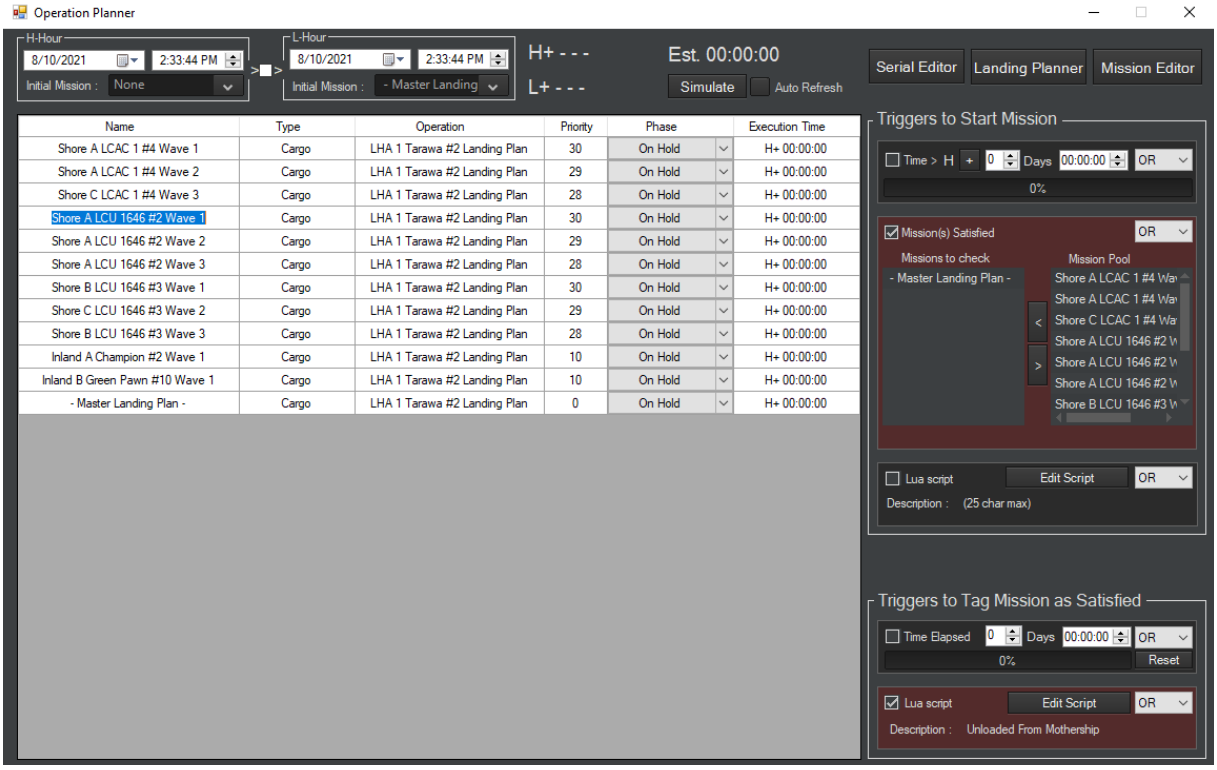
☐ Communications Jamming

☐ Weather affects ship speed

☑ Allow instant loading for landing planner

Once you are satisfied with the 3 options, click on “OK”, the landing plan will be generated and added to the operation planner.

Go to **Missions + Ref. Points -> Operation Manager -> Operation Planner**



Operation Planner

H-Hour: 8/10/2021 2:33:44 PM — Initial Mission : None

L-Hour: 8/10/2021 2:33:44 PM — Initial Mission : - Master Landing

H+ - - -

L+ - - -

Est. 00:00:00

Simulate □ Auto Refresh

Name	Type	Operation	Priority	Phase	Execution Time
Shore A LCAC 1 #4 Wave 1	Cargo	LHA 1 Tarawa #2 Landing Plan	30	On Hold	H+ 00:00:00
Shore A LCAC 1 #4 Wave 2	Cargo	LHA 1 Tarawa #2 Landing Plan	29	On Hold	H+ 00:00:00
Shore C LCAC 1 #4 Wave 3	Cargo	LHA 1 Tarawa #2 Landing Plan	28	On Hold	H+ 00:00:00
Shore A LCU 1646 #2 Wave 1	Cargo	LHA 1 Tarawa #2 Landing Plan	30	On Hold	H+ 00:00:00
Shore A LCU 1646 #2 Wave 2	Cargo	LHA 1 Tarawa #2 Landing Plan	29	On Hold	H+ 00:00:00
Shore A LCU 1646 #2 Wave 3	Cargo	LHA 1 Tarawa #2 Landing Plan	28	On Hold	H+ 00:00:00
Shore B LCU 1646 #3 Wave 1	Cargo	LHA 1 Tarawa #2 Landing Plan	30	On Hold	H+ 00:00:00
Shore C LCU 1646 #3 Wave 2	Cargo	LHA 1 Tarawa #2 Landing Plan	29	On Hold	H+ 00:00:00
Shore B LCU 1646 #3 Wave 3	Cargo	LHA 1 Tarawa #2 Landing Plan	28	On Hold	H+ 00:00:00
Inland A Champion #2 Wave 1	Cargo	LHA 1 Tarawa #2 Landing Plan	10	On Hold	H+ 00:00:00
Inland B Green Pawn #10 Wave 1	Cargo	LHA 1 Tarawa #2 Landing Plan	10	On Hold	H+ 00:00:00
- Master Landing Plan -	Cargo	LHA 1 Tarawa #2 Landing Plan	0	On Hold	H+ 00:00:00

Triggers to Start Mission

□ Time > H + 0 Days 00:00:00 OR — 0%

☑ Mission(s) Satisfied OR

Missions to check: - Master Landing Plan -

Mission Pool: Shore A LCAC 1 #4 Wa / Shore A LCAC 1 #4 Wa / Shore C LCAC 1 #4 Wa / Shore A LCU 1646 #2 W / Shore A LCU 1646 #2 W / Shore A LCU 1646 #2 W / Shore B LCU 1646 #3 W

□ Lua script — Edit Script — OR

Description : (25 char max)

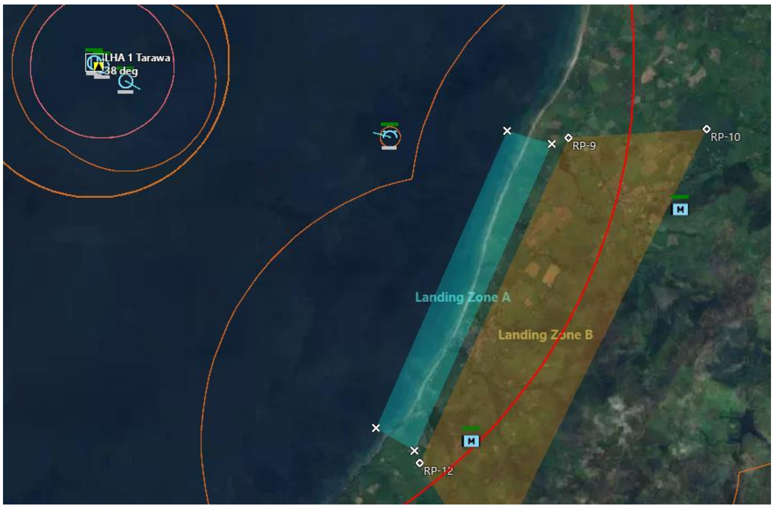
Triggers to Tag Mission as Satisfied

□ Time Elapsed 0 Days 00:00:00 OR — 0% — Reset

☑ Lua script — Edit Script — OR

Description : Unloaded From Mothership

The Operation Planner lists all the missions of your side. We can see the generated cargo missions from the landing planner. Thankfully the landing planner takes care of all triggers, lua scripts, mission dependencies, multi-mission assignments, phase selections, priorities, etc. We now have a functional landing operation.



Landing operation in progress.

Self-transportation and amphibious vehicles

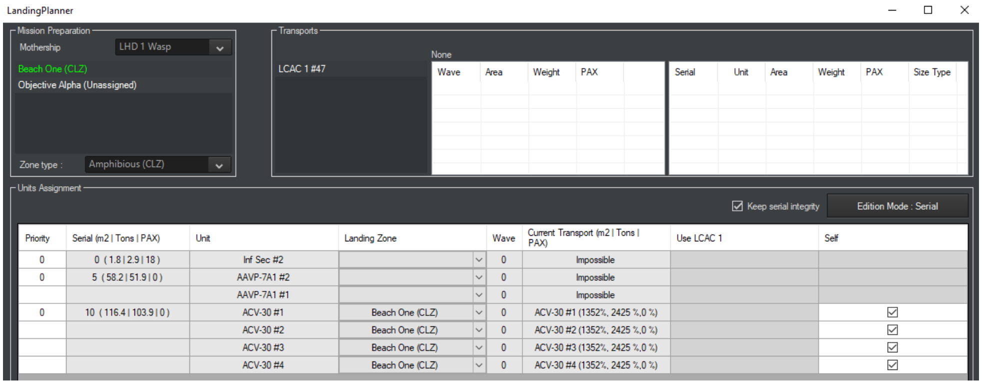
The landing planner allows to set eligible units to transport themselves. The only eligible units at the moment are ground units with amphibious capabilities.

Amphibious vehicles not counted as transports; we consider them as part of the serial.

The “self” transport option mean that the unit will uses its own means to go to the landing zone. This option has priority, it will therefore ignore all other option if the “self” option is checked.

In order to enable this option, the serial must contain exclusively amphibious vehicles. If you have a serial with 9 amphibious vehicles and 1 non-amphibious unit, none of these units will have the “self” option activated.

To see this in action, please open the **Quickstart Amphibious vehicles and Active Units** scenario provided with this manual.



Mothership: LHD 1 Wasp

Beach One (CLZ)
Objective Alpha (Unassigned)

Zone type : Amphibious (CLZ)

LCAC 1 #47

None

Wave	Area	Weight	PAX

Serial	Unit	Area	Weight	PAX	Size Type

☑ Keep serial integrity

Priority	Serial (m2 | Tons | PAX)	Unit	Landing Zone	Wave	Current Transport (m2 | Tons | PAX)	Use LCAC 1	Self
0	0 (1.8 | 2.9 | 18)	Inf Sec #2		0	Impossible		
0	5 (58.2 | 51.9 | 0)	AAVP-7A1 #2		0	Impossible		
		AAVP-7A1 #1		0	Impossible		
0	10 (116.4 | 103.9 | 0)	ACV-30 #1	Beach One (CLZ)	0	ACV-30 #1 (1352%, 2425 %,0 %)		☑
		ACV-30 #2	Beach One (CLZ)	0	ACV-30 #2 (1352%, 2425 %,0 %)		☑
		ACV-30 #3	Beach One (CLZ)	0	ACV-30 #3 (1352%, 2425 %,0 %)		☑
		ACV-30 #4	Beach One (CLZ)	0	ACV-30 #4 (1352%, 2425 %,0 %)		☑

Before moving on operation planner, it is essential to understand a new core feature: Multi-missioned units.

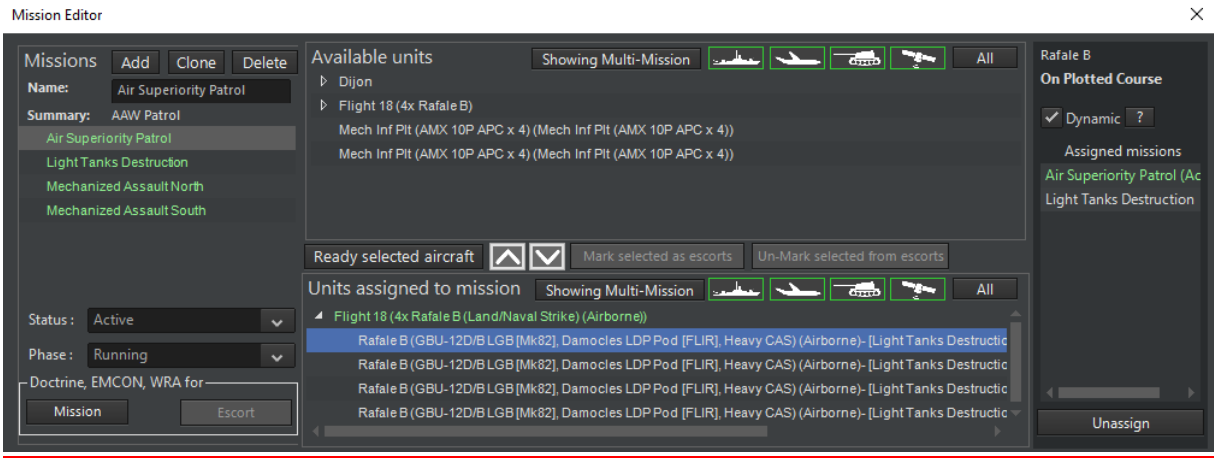
Understanding dynamic units (aka multi-missioned units)

Previously, in Command, a given unit could only be assigned to a single mission. If you wanted to assign the unit to another mission, you would have to manually unassign it from the current mission and then assign it to a new one.

This extension adds the possibility for a single unit to be assigned to multiple missions. Of course, the unit can only execute one mission at a time, but you can now prioritize which mission it should execute. Mission priority is set via the operation planner.

Let’s assume we have a scenario where a group of aircrafts is assigned to a patrol mission. These aircrafts should patrol an area from a given time and then start a strike mission against a group of tanks. Without the operation planner, the player would need to track the time and then manually switch each aircraft to a new mission, at the right moment. Thanks to the operation planner such behavior can be automated, and in more complex environment, our units could even behave like a reactive AI, aware of the simulation at the strategical level.

IMPORTANT ! A unit without “Dynamic” Checked will be ignored by the operation planner dynamic assignment, if you want a unit to work with operation planner’s feature you **MUST** designate it as a dynamic unit.


Mission Editor

Missions Add Clone Delete

Name: Air Superiority Patrol

Summary: AAW Patrol

Air Superiority Patrol
Light Tanks Destruction
Mechanized Assault North
Mechanized Assault South

Status : Active

Phase : Running

Doctrine, EMCON, WRA for
Mission Escort

Available units — Showing Multi-Mission — All

- Dijon
- Flight 18 (4x Rafale B)
- Mech Inf Plt (AMX 10P APC x 4) (Mech Inf Plt (AMX 10P APC x 4))
- Mech Inf Plt (AMX 10P APC x 4) (Mech Inf Plt (AMX 10P APC x 4))

Ready selected aircraft — Mark selected as escorts — Un-Mark selected from escorts

Units assigned to mission — Showing Multi-Mission — All

- Flight 18 (4x Rafale B (Land/Naval Strike) (Airborne))
  - Rafale B (GBU-12D/B LGB [Mk82], Damocles LDP Pod [FLIR], Heavy CAS) (Airborne)- [Light Tanks Destructic
  - Rafale B (GBU-12D/B LGB [Mk82], Damocles LDP Pod [FLIR], Heavy CAS) (Airborne)- [Light Tanks Destructic
  - Rafale B (GBU-12D/B LGB [Mk82], Damocles LDP Pod [FLIR], Heavy CAS) (Airborne)- [Light Tanks Destructic
  - Rafale B (GBU-12D/B LGB [Mk82], Damocles LDP Pod [FLIR], Heavy CAS) (Airborne)- [Light Tanks Destructic

Rafale B

On Plotted Course

✓ Dynamic ?

Assigned missions

Air Superiority Patrol (Ac
Light Tanks Destruction

Unassign

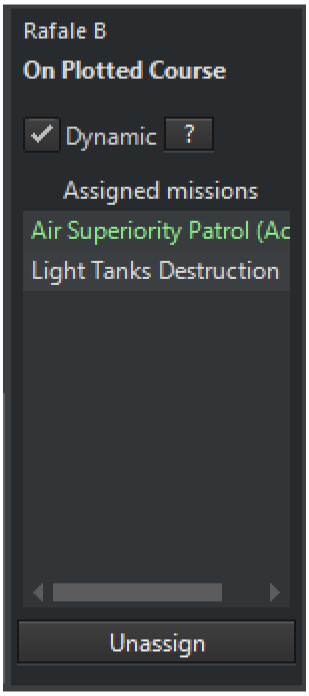
Open the scenario called “**Quickstart Operation Planner”.**

Open the mission editor.

Select “Air Superiority Patrol” mission in the list and make sure both filters are set to “Showing Multi-Mission”.

Select one of the Rafale in flight 18.

Notice that since we have toggle “Showing Multi-Mission” we are now allowed to assign multiple mission to a unit, they are still shown in the “Available units” list despite having an assigned mission.


Rafale B

On Plotted Course

✓ Dynamic ?

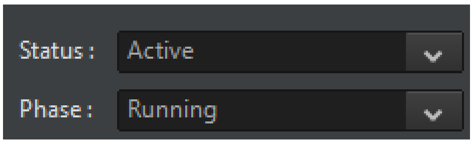
Assigned missions

Air Superiority Patrol (Ac
Light Tanks Destruction

Unassign

This panel on the right side of the mission editor shows all missions assigned to this unit. The mission in green in the one currently executed by the unit. Missions don’t have an order of execution, but a priority, which is set in the operation planner, as we will see later.

The “Dynamic” checkbox allows the unit to be assigned to multiple missions, units are all unchecked by default to reflect default command behavior.


Status : Active

Phase : Running

Below the mission list we can see the current status and phase of the mission. We will see later in the operation planner chapter as “Phase” is a new way of managing the mission dynamically and is closely related to multi-mission.

It is on this screen that you decide to add the unit to all of the missions you want it assigned to. At this point you don’t have worry about priority or to select a current mission as the operation planner will manage this for you.

In above’s example we see that Rafale B is assigned to both the “Air Superiority Patrol” and “Light Tanks Destruction” missions, and the active mission for this unit is the one in green: “Air Superiority

Patrol." Depending on the configuration of the operation planer, this unit may automatically switch to the "Light Tanks Destruction" mission at some point in the future.

Don't worry if the multi-mission mechanism is not entirely clear to you yet, as the mission editor is only half the story. The next chapter on operation planner will explain the other half.

Mobile facility vs Ground unit

One of the most significant change brought by the operation planner, is the possibility to have units dynamically change mission. Not all types of units are eligible for this kind of behavior.

From a minimalist ground unit implementation, Command then introduced fully simulated ground units.

It is important to understand the core difference between:

- **Ground units** as facility which is a legacy implementation where ground unit are assimilated as moving "facility"
- **Mobile facilities**, a new implementation, which works like others active units such as aircraft, ships etc…

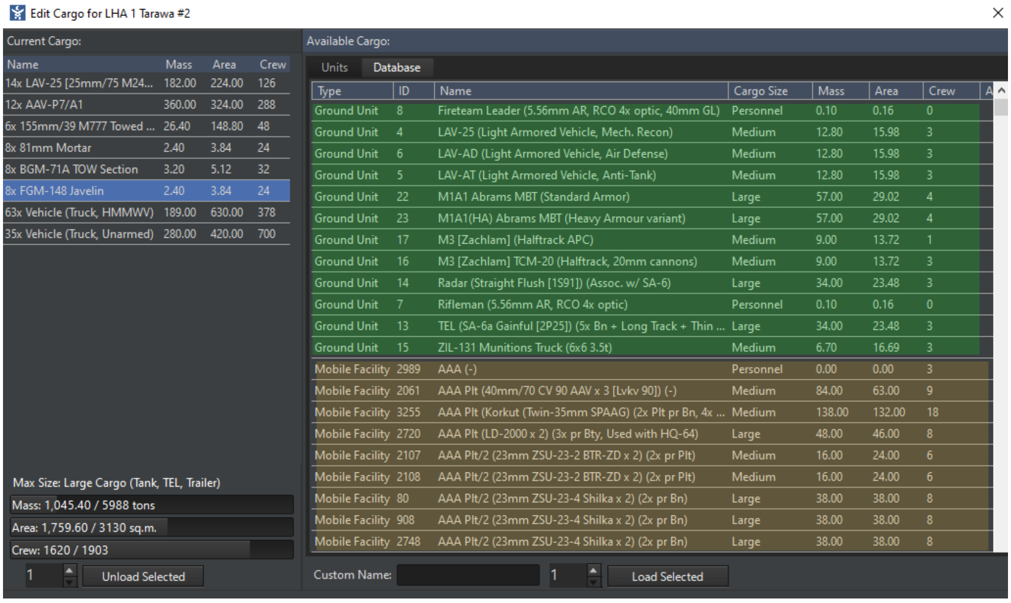
The first hold a group of units abstractly represented as "mounts", while the last is a fully simulated individual unit.

The core difference that interests us here is that ground units as facility are transported as cargo which doesn't have an existence in the simulation until it has landed (and spawned).

This means that we can't assign or queue them a mission until they have landed, and you will not be able to achieve a fully autonomous behavior for your units, in this case.

Since the 8th November 2021 update we allow cargo operation for active units, meaning that all these limitation are now removed. However, you must have the right methodology to achieve this.

Unless you have to achieve backward compatibility or if a database entry is missing, you MUST use ground units NOT mobiles facilities to use the operation planner at its fullest.



Edit Cargo for LHA 1 Tarawa #2

Current Cargo:

Name	Mass	Area	Crew
14x LAV-25 [25mm/75 M24...	182.00	224.00	126
12x AAV-P7/A1	360.00	324.00	288
6x 155mm/39 M777 Towed ...	26.40	148.80	48
8x 81mm Mortar	2.40	3.84	24
8x BGM-71A TOW Section	3.20	5.12	32
8x FGM-148 Javelin	2.40	3.84	24
63x Vehicle (Truck, HMMWV)	189.00	630.00	378
35x Vehicle (Truck, Unarmed)	280.00	420.00	700

Max Size: Large Cargo (Tank, TEL, Trailer)

Mass: 1,045.40 / 5988 tons

Area: 1,759.60 / 3130 sq.m.

Crew: 1620 / 1903

Available Cargo:

Type	ID	Name	Cargo Size	Mass	Area	Crew
Ground Unit	8	Fireteam Leader (5.56mm AR, RCO 4x optic, 40mm GL)	Personnel	0.10	0.16	0
Ground Unit	4	LAV-25 (Light Armored Vehicle, Mech. Recon)	Medium	12.80	15.98	3
Ground Unit	6	LAV-AD (Light Armored Vehicle, Air Defense)	Medium	12.80	15.98	3
Ground Unit	5	LAV-AT (Light Armored Vehicle, Anti-Tank)	Medium	12.80	15.98	3
Ground Unit	22	M1A1 Abrams MBT (Standard Armor)	Large	57.00	29.02	4
Ground Unit	23	M1A1(HA) Abrams MBT (Heavy Armour variant)	Large	57.00	29.02	4
Ground Unit	17	M3 [Zachlam] (Halftrack APC)	Medium	9.00	13.72	1
Ground Unit	16	M3 [Zachlam] TCM-20 (Halftrack, 20mm cannons)	Medium	9.00	13.72	3
Ground Unit	14	Radar (Straight Flush [1S91]) (Assoc. w/ SA-6)	Large	34.00	23.48	3
Ground Unit	7	Rifleman (5.56mm AR, RCO 4x optic)	Personnel	0.10	0.16	0
Ground Unit	13	TEL (SA-6a Gainful [2P25]) (5x Bn + Long Track + Thin ...	Large	34.00	23.48	3
Ground Unit	15	ZIL-131 Munitions Truck (6x6 3.5t)	Medium	6.70	16.69	3
Mobile Facility	2989	AAA (-)	Personnel	0.00	0.00	3
Mobile Facility	2061	AAA Plt (40mm/70 CV 90 AAV x 3 [Lvkv 90]) (-)	Medium	84.00	63.00	9
Mobile Facility	3255	AAA Plt (Korkut (Twin-35mm SPAAG) (2x Plt pr Bn, 4x ...	Medium	138.00	132.00	18
Mobile Facility	2720	AAA Plt (LD-2000 x 2) (3x pr Bty, Used with HQ-64)	Large	48.00	46.00	8
Mobile Facility	2107	AAA Plt/2 (23mm ZSU-23-2 BTR-ZD x 2) (2x pr Plt)	Medium	16.00	24.00	6
Mobile Facility	2108	AAA Plt/2 (23mm ZSU-23-2 BTR-ZD x 2) (2x pr Plt)	Medium	16.00	24.00	6
Mobile Facility	80	AAA Plt/2 (23mm ZSU-23-4 Shilka x 2) (2x pr Bn)	Large	38.00	38.00	8
Mobile Facility	908	AAA Plt/2 (23mm ZSU-23-4 Shilka x 2) (2x pr Bn)	Large	38.00	38.00	8
Mobile Facility	2748	AAA Plt/2 (23mm ZSU-23-4 Shilka x 2) (2x pr Bn)	Large	38.00	38.00	8

Custom Name:

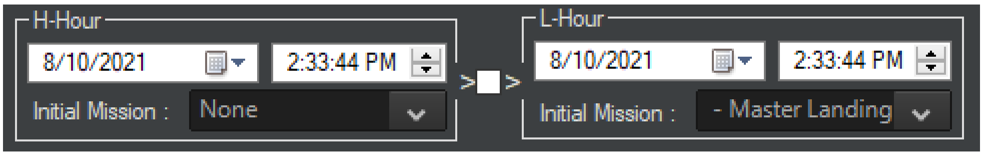
Operation Planner

Open the operation planner: **Missions + Ref. Points -> Operation Manager -> Landing Planner**

Creating a landing plan gave you a quick introduction to the operation planner. But it can be used beyond landing operations – it is a complex and general-purpose framework that works alongside many of the new features in Command.

The operation planner adds a new level of interaction between missions and allows units to be dynamically reassigned from one mission to another.

Since the operation planner is sometimes tied to a landing plan, we have here the concepts of H-Hour and L-Hour to indicate overall operational time. These can be of course be ignored, or used in a different purpose.



H-Hour
8/10/2021
2:33:44 PM
Initial Mission : None

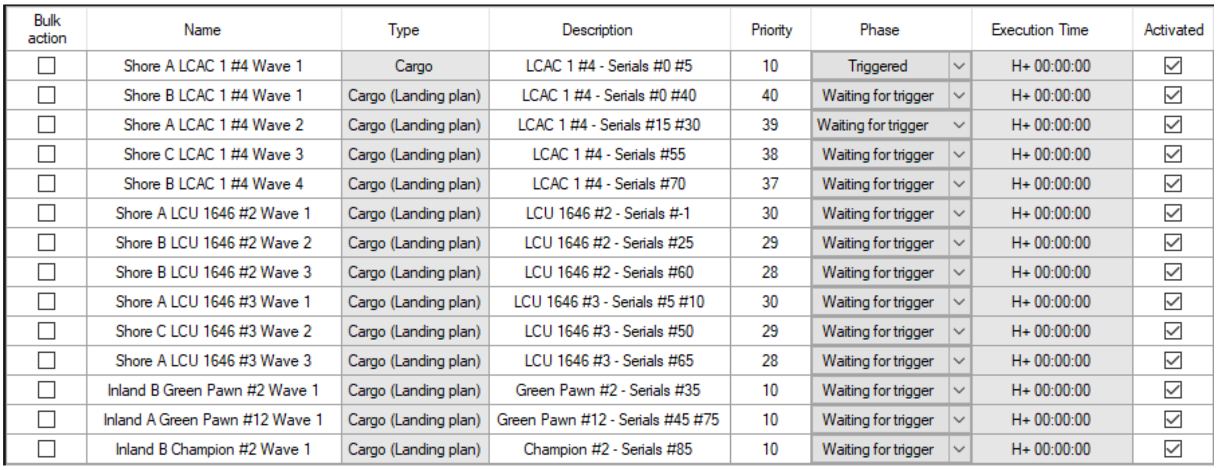
L-Hour
8/10/2021
2:33:44 PM
Initial Mission : - Master Landing

On the top left, can be defined the H-Hour and the L-Hour.

The H-Hour box on the left is where we can define the date and time to start the H-Hour mission. The H-Hour mission is the initial mission in the operation. L-Hour box works identically to H-Hour but they are independent.

Once an H-Hour or L-Hour is hit the respective initial mission can no longer be changed.

The checkbox in the middle ties the H-Hour to the L-Hour, meaning that the time separation will be constant between H-Hour and L-Hour when you modify the time for either.



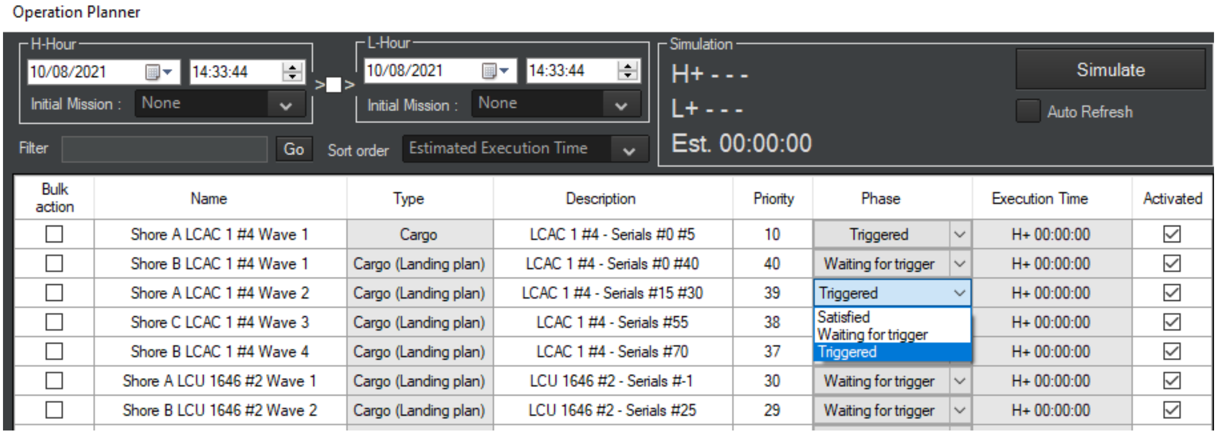
Bulk action	Name	Type	Description	Priority	Phase	Execution Time	Activated
☐	Shore A LCAC 1 #4 Wave 1	Cargo	LCAC 1 #4 - Serials #0 #5	10	Triggered	H+ 00:00:00	☑
☐	Shore B LCAC 1 #4 Wave 1	Cargo (Landing plan)	LCAC 1 #4 - Serials #0 #40	40	Waiting for trigger	H+ 00:00:00	☑
☐	Shore A LCAC 1 #4 Wave 2	Cargo (Landing plan)	LCAC 1 #4 - Serials #15 #30	39	Waiting for trigger	H+ 00:00:00	☑
☐	Shore C LCAC 1 #4 Wave 3	Cargo (Landing plan)	LCAC 1 #4 - Serials #55	38	Waiting for trigger	H+ 00:00:00	☑
☐	Shore B LCAC 1 #4 Wave 4	Cargo (Landing plan)	LCAC 1 #4 - Serials #70	37	Waiting for trigger	H+ 00:00:00	☑
☐	Shore A LCU 1646 #2 Wave 1	Cargo (Landing plan)	LCU 1646 #2 - Serials #-1	30	Waiting for trigger	H+ 00:00:00	☑
☐	Shore B LCU 1646 #2 Wave 2	Cargo (Landing plan)	LCU 1646 #2 - Serials #25	29	Waiting for trigger	H+ 00:00:00	☑
☐	Shore B LCU 1646 #2 Wave 3	Cargo (Landing plan)	LCU 1646 #2 - Serials #60	28	Waiting for trigger	H+ 00:00:00	☑
☐	Shore A LCU 1646 #3 Wave 1	Cargo (Landing plan)	LCU 1646 #3 - Serials #5 #10	30	Waiting for trigger	H+ 00:00:00	☑
☐	Shore C LCU 1646 #3 Wave 2	Cargo (Landing plan)	LCU 1646 #3 - Serials #50	29	Waiting for trigger	H+ 00:00:00	☑
☐	Shore A LCU 1646 #3 Wave 3	Cargo (Landing plan)	LCU 1646 #3 - Serials #65	28	Waiting for trigger	H+ 00:00:00	☑
☐	Inland B Green Pawn #2 Wave 1	Cargo (Landing plan)	Green Pawn #2 - Serials #35	10	Waiting for trigger	H+ 00:00:00	☑
☐	Inland A Green Pawn #12 Wave 1	Cargo (Landing plan)	Green Pawn #12 - Serials #45 #75	10	Waiting for trigger	H+ 00:00:00	☑
☐	Inland B Champion #2 Wave 1	Cargo (Landing plan)	Champion #2 - Serials #85	10	Waiting for trigger	H+ 00:00:00	☑

The spreadsheet in the center lists all the side's missions. Most columns are informational. Generally, you will be dealing with 2 columns: **Priority** and **Phase**.

Mission priority

The priority of a mission does NOT designate his supposed position in a mission execution queue. It indicates to its assigned units how important this mission is **at this moment**. The mission priority is used by units assigned to multiple missions to decide which mission to execute at any given time.
A unit having a mission in “On Hold” phase will not have this mission evaluated for the active mission evaluation.

Mission phases


Operation Planner

H-Hour: 10/08/2021 14:33:44, Initial Mission : None

L-Hour: 10/08/2021 14:33:44, Initial Mission : None

Simulation: H+ - - -, L+ - - -, Est. 00:00:00

☐ Auto Refresh

Filter: Sort order: Estimated Execution Time

Bulk action	Name	Type	Description	Priority	Phase	Execution Time	Activated
☐	Shore A LCAC 1 #4 Wave 1	Cargo	LCAC 1 #4 - Serials #0 #5	10	Triggered	H+ 00:00:00	☑
☐	Shore B LCAC 1 #4 Wave 1	Cargo (Landing plan)	LCAC 1 #4 - Serials #0 #40	40	Waiting for trigger	H+ 00:00:00	☑
☐	Shore A LCAC 1 #4 Wave 2	Cargo (Landing plan)	LCAC 1 #4 - Serials #15 #30	39	Triggered	H+ 00:00:00	☑
☐	Shore C LCAC 1 #4 Wave 3	Cargo (Landing plan)	LCAC 1 #4 - Serials #55	38		H+ 00:00:00	☑
☐	Shore B LCAC 1 #4 Wave 4	Cargo (Landing plan)	LCAC 1 #4 - Serials #70	37		H+ 00:00:00	☑
☐	Shore A LCU 1646 #2 Wave 1	Cargo (Landing plan)	LCU 1646 #2 - Serials #-1	30	Waiting for trigger	H+ 00:00:00	☑
☐	Shore B LCU 1646 #2 Wave 2	Cargo (Landing plan)	LCU 1646 #2 - Serials #25	29	Waiting for trigger	H+ 00:00:00	☑

A mission phase is a new concept introduced with the operation planner extension, it is not related to the mission status and should not be mistaken with it.

Waiting for trigger: The mission may (or may not) have already started yet but queued units won’t evaluate this mission when choosing one to be assigned to.

Satisfied: The mission is considered to have achieved enough of its objective for its current assigned dynamic units to consider other missions, but a satisfied mission does not mean its ending. This tag is used for mission triggers and for multi-mission priority.

Triggered: A running mission will become a valid candidate for multi-mission units assigned to it. Therefore, a queued unit to this mission may become assigned to it.

A unit assigned to multiple missions will pick the most appropriate mission to be assigned to, based on the priority and the phase of the mission. unless it is already active on a mission that is in “running” phase, or if all assigned missions are in “On Hold” phase.
A unit that is active in a mission in “Satisfied” phase will still evaluate the satisfied mission and may continue that mission if no other missions are available.

Name

Just like for operation, the mission’s name helps for organization. However, user’s made Lua scripts might reference a mission by its name and it is recommended to be careful when changing mission’s name in such situation.

Description

This is purely an informational tag and does not affect the simulation at the moment. Use and edit this field to organize yourself.
Generated mission will usually contain some generated information.

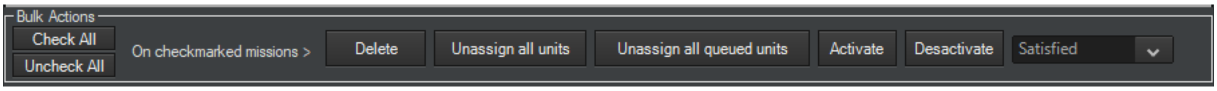
Type

This is the mission sub type as defined in Command's simulation.

Execution Time

The time, relative to H-Hour at which the mission is estimated to begin. See the chapter "Working with estimation" for more details.

Bulk actions



Bulk action tools are located on the bottom of the operation planner. These are used to select multiple mission at once and perform simultaneous modifications on them.

Filter

The filter tool is used to do a specific term research on mission and filter the result of this search query.

Triggers

This chapter is the core of the operation planner and landing planner capabilities. It allows relationships between mission and a dynamic approach when executing missions.

A trigger is like a set of a condition and an action, if the condition is met, then we do an action.

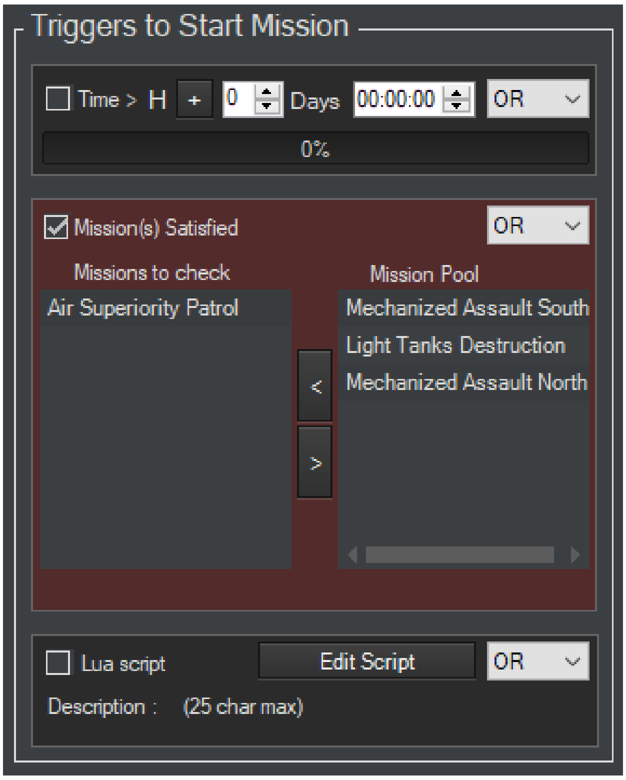
Command evaluates all these triggers every second.

For mission, there are 2 types of actions:

- We start a mission (we set the mission's phase as "Running").
- We finish a mission (we set the mission's phase as "Satisfied").

It is important to understand that Command doesn't have a concept of mission completion, when tagging a mission as satisfied, command only indicates that the mission have enough fulfilled its objective to allow its assigned units to evaluate other mission assignment options. Of course all relevant missions have already implicit mission completion mechanisms, a cargo mission will stop operation once the task is done, a strike mission won't launch again to strike a nonexistent targets, etc…

These 2 actions are tied to a set of conditions.


Triggers to Start Mission
□ Time > H + 0 Days 00:00:00 OR
0%
☑ Mission(s) Satisfied OR
Missions to check
Air Superiority Patrol
Mission Pool
Mechanized Assault South
Light Tanks Destruction
Mechanized Assault North
<
>
□ Lua script Edit Script OR
Description : (25 char max)

Starting a mission with triggers

Select any of the mission and look on the right side of the operation planner window. Notice the main block called “Triggers to Start Mission”, and how it is separated into 3 smaller blocks.

These smaller blocks are individual conditions. Checking them means this specific condition will be evaluated.

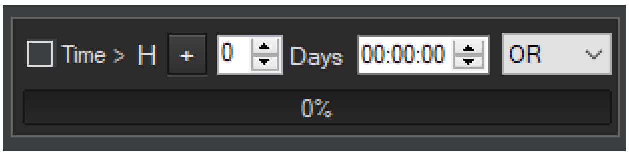
The dropdown on the right of each smaller block is called a conditional operator.

As you can see there are 3 types of triggers that can start a mission:

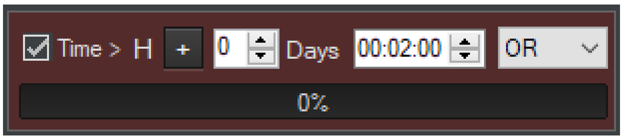
- A time based trigger
- A mission dependency trigger
- A Lua script trigger

Time-based trigger

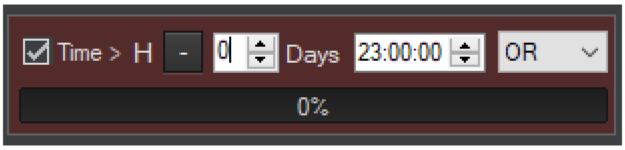
This will be triggered when the scenario date reaches the defined H-Hour plus or minus a given duration.

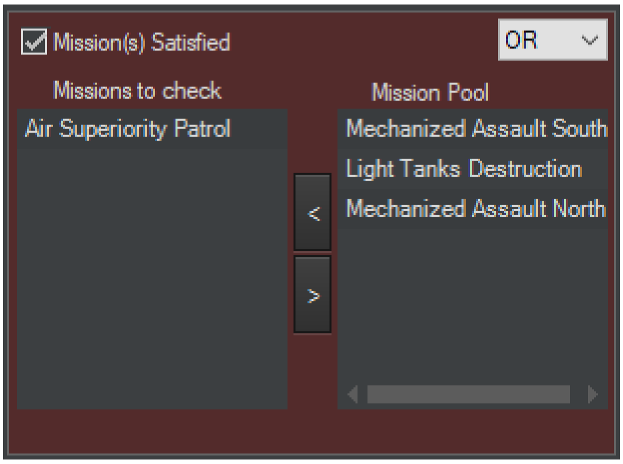


*Exemple #1: This trigger will be **true** if we reach H+ 2 hours*



*Exemple #2: This trigger will be **true** if we reach H- 23 hours*





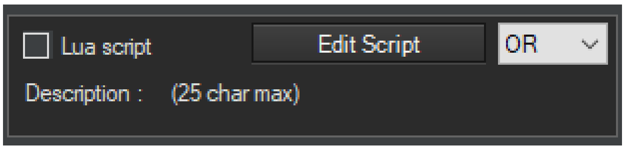
Mission dependency trigger

This will be triggered if all missions in “missions to check” are in “Satisfied” phase.

In this example, it will be triggered when “Air Superiority Patrol” mission is in “Satisfied” Phase.

Lua script trigger

This will be triggered if the Lua script contained returns TRUE as a value. See the “Lua scripting” chapter for more information.



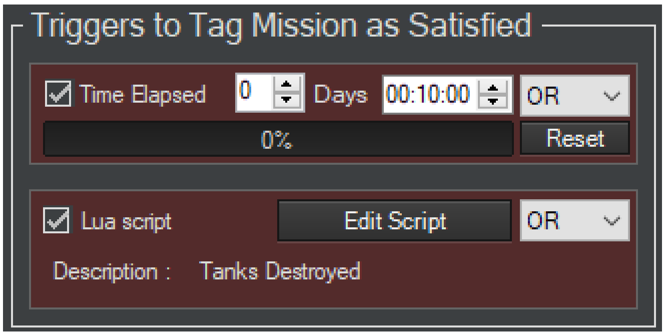
□ Lua script | Edit Script | OR

Description : (25 char max)

“Finishing” a mission with triggers

It was mentioned earlier that Command does not have an explicit concept of finished mission.
The triggers to tag a mission as “satisfied” work just like the one to start it.

The first trigger is a time based one, it tracks the elapsed time since the mission’s phase has been set to “Running” and will be true once the defined time elapsed.



Triggers to Tag Mission as Satisfied

☑ Time Elapsed 0 Days 00:10:00 OR

0% Reset

☑ Lua script | Edit Script | OR

Description : Tanks Destroyed

The second one is a Lua script trigger and work identically to the one in the mission start trigger – it returns the Boolean value of the contained lua script.

Logical Operators

Triggers, in Command, are tied with logical operators.

Take a look at the picture on the right, representing a set of triggers to start a mission:

We have set all 3 triggers to be checked. On the right of each triggers you can notice a dropdown when you can selected either the OR or AND operator.

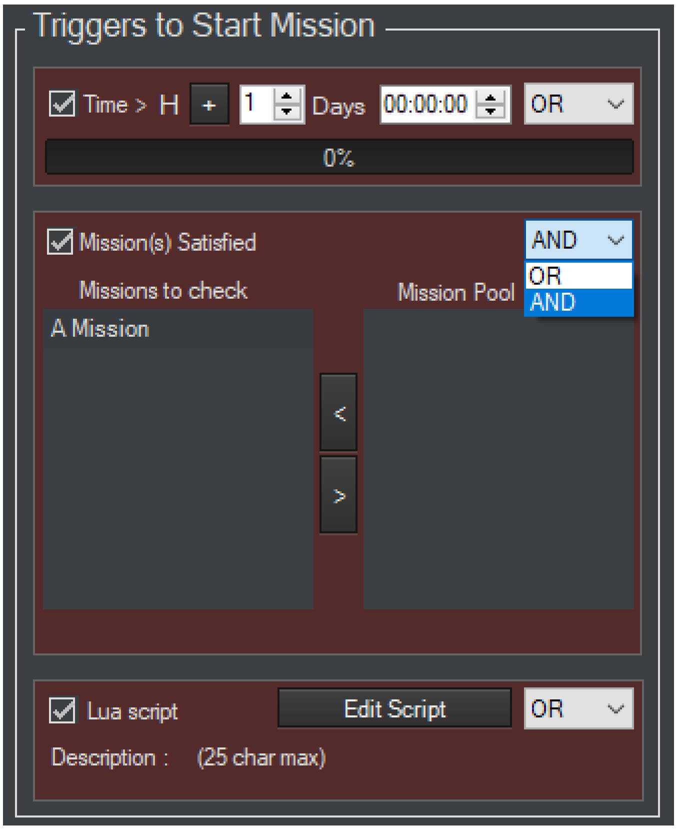
For the triggers to return TRUE and therefore start the mission (set its phase to “Running”) each checked trigger is evaluated.

We see in this example that the first trigger has “OR” operator, the second “AND,” the third “OR.”

What it means in this situation is that the second trigger (the one with “AND”) must be true.

In addition the “AND” trigger needing to be true, either of the first or third “OR” triggers must also be true.

If all conditions are met and the current mission’s phases is “On Hold” then the mission will change it phase to “Running”.



Triggers to Start Mission

☑ Time > H + 1 Days 00:00:00 OR

0%

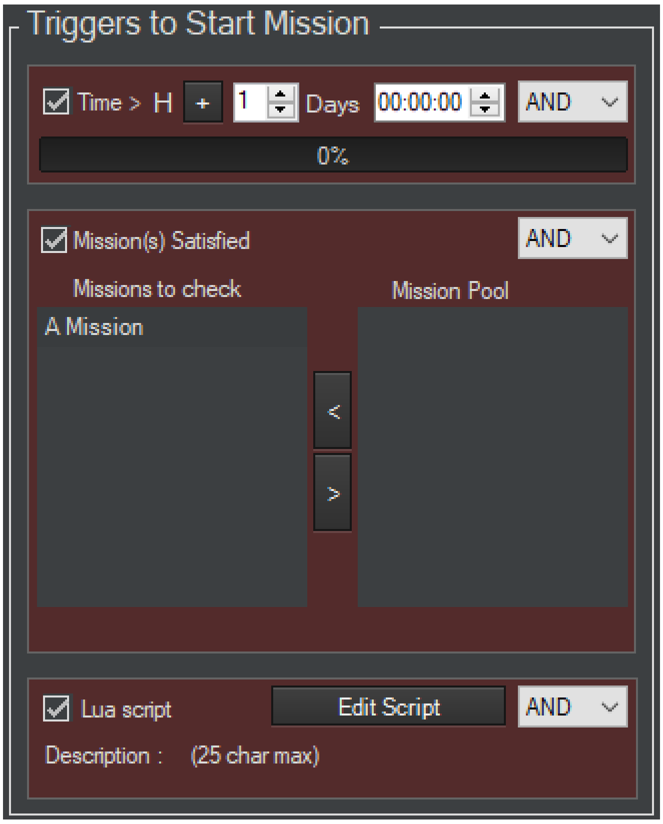
☑ Mission(s) Satisfied AND

Missions to check | Mission Pool

A Mission

☑ Lua script | Edit Script | OR

Description : (25 char max)



Triggers to Start Mission

☑ Time > H + 1 Days 00:00:00 AND

0%

☑ Mission(s) Satisfied AND

Missions to check: A Mission

Mission Pool

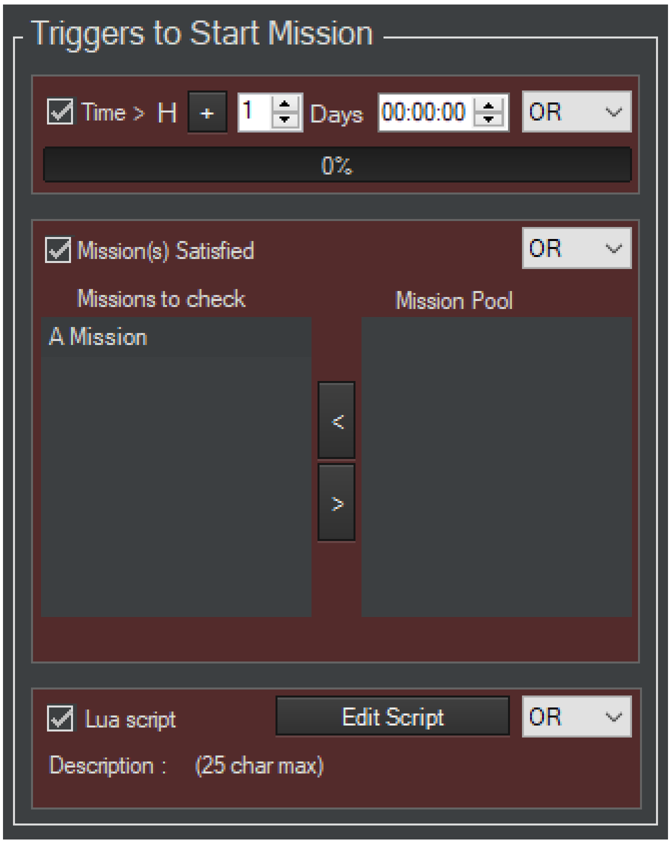
☑ Lua script Edit Script AND

Description : (25 char max)

Another example:

All triggers are checked (and will be evaluated), and each of them have the “AND” operator.

This means that the missions will start when all conditions are true.



Triggers to Start Mission

☑ Time > H + 1 Days 00:00:00 OR

0%

☑ Mission(s) Satisfied OR

Missions to check: A Mission

Mission Pool

☑ Lua script Edit Script OR

Description : (25 char max)

Final example:

All triggers have an “OR” operator, meaning

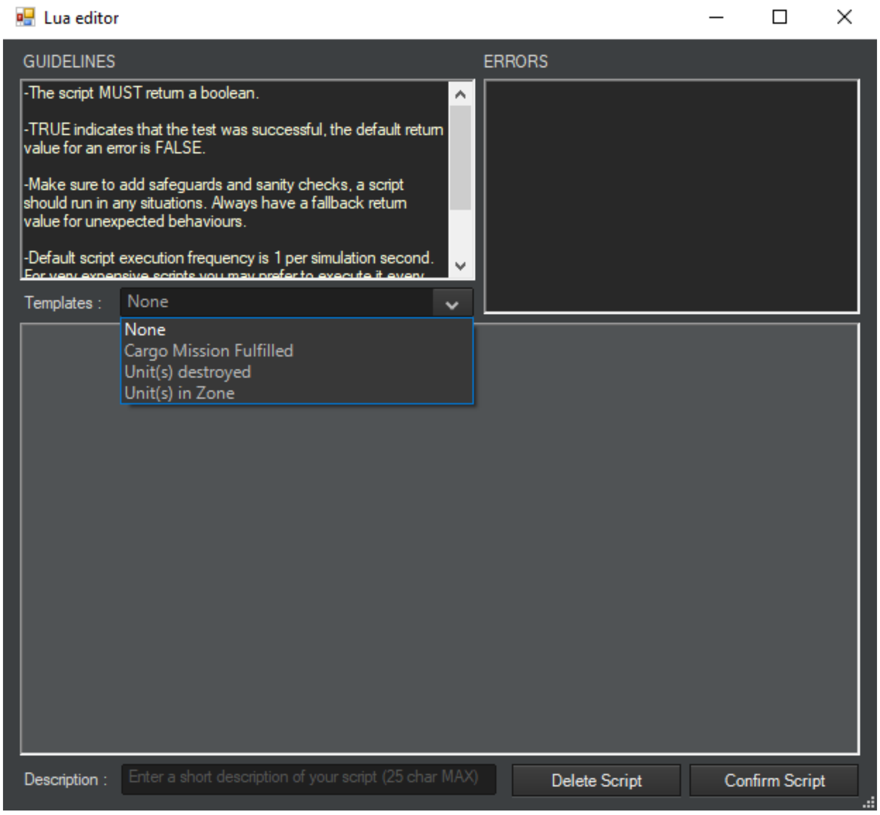
That the mission will start if any of the conditions is true.

Lua scripting

Command has already a powerful Lua scripting framework. The operation planner allows the integration of your script as a trigger.

Clicking on “Edit Script” in either the “Triggers to Start Mission” or the “Triggers to Tag Mission as Satisfied” group will bring you to an interface where you can add your script in.

Just like the rest of the triggers, the Lua Script trigger will be executed each second for its associated mission. The Lua script must return a Boolean.



-The script MUST return a boolean.

-TRUE indicates that the test was successful, the default return value for an error is FALSE.

-Make sure to add safeguards and sanity checks, a script should run in any situations. Always have a fallback return value for unexpected behaviours.

-Default script execution frequency is 1 per simulation second. For very expensive scripts you may prefer to execute it every

Working with estimation

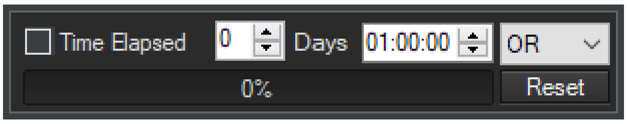
Command is such a complex simulation that giving an accurate estimation of an operation duration could take up to hours of computation. The operation planner provides an instantaneous estimation at the price of reliability.

The operation planner estimation takes into account all the time based triggers and mission dependencies, it simulates a run and then display the estimated execution tie for each mission.

This estimation work only thanks to the user’s input on triggers.

The special case of Lua script

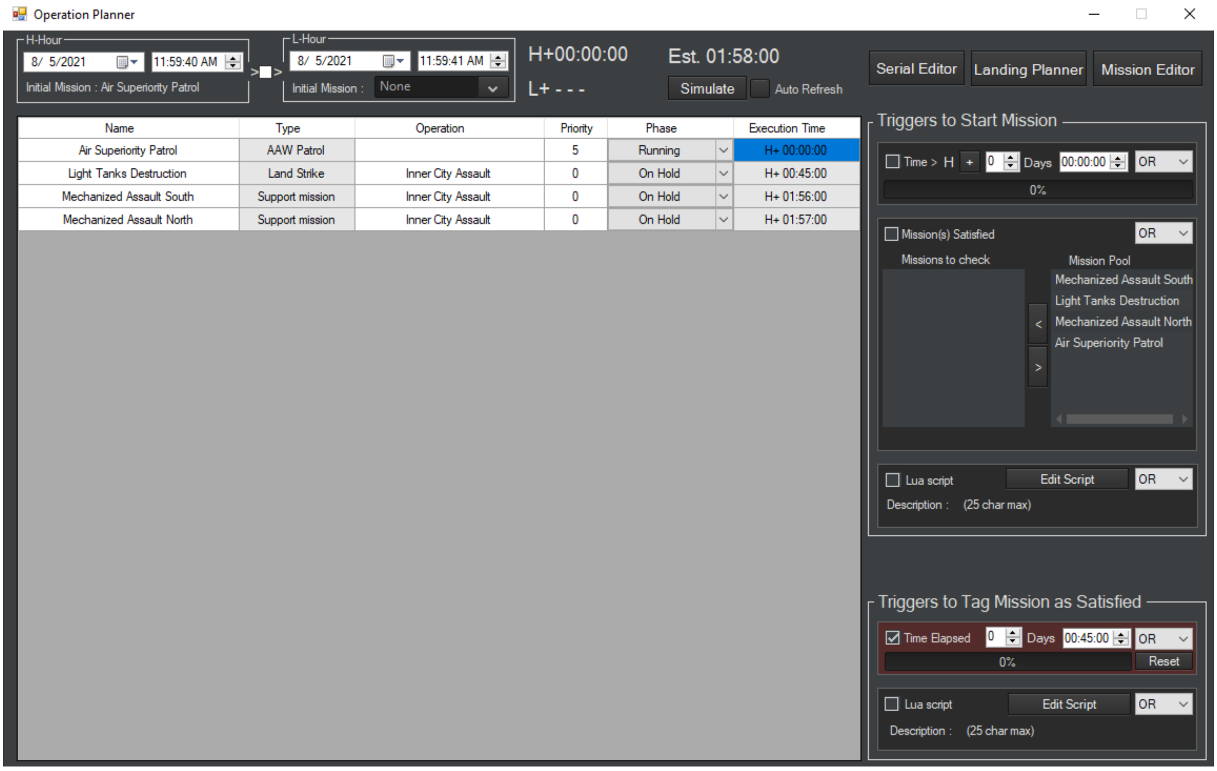
You may have noticed that not all triggers are time-based, some depends on lua script and cannot be reliably predicted. In this case, you will have to manually input a value into the trigger in this trigger, shown on the right.


□ Time Elapsed 0 Days 01:00:00 OR

0% Reset

It is not necessary to check (enable) the trigger, having an unchecked "Time Elapsed" trigger with a value basically tells the operation planner: "Only use this trigger when estimating execution time". In this example, we assume the mission will be satisfied after 1 hour.

If everything is properly configured, clicking on the "Simulate" button will calculate the execution time, relative to H-Hour for each mission.


Operation Planner

H-Hour: 8/ 5/2021 11:59:40 AM
Initial Mission : Air Superiority Patrol

L-Hour: 8/ 5/2021 11:59:41 AM
Initial Mission : None

H+00:00:00

L+ - - -

Est. 01:58:00

Simulate □ Auto Refresh

Name	Type	Operation	Priority	Phase	Execution Time
Air Superiority Patrol	AAW Patrol		5	Running	H+ 00:00:00
Light Tanks Destruction	Land Strike	Inner City Assault	0	On Hold	H+ 00:45:00
Mechanized Assault South	Support mission	Inner City Assault	0	On Hold	H+ 01:56:00
Mechanized Assault North	Support mission	Inner City Assault	0	On Hold	H+ 01:57:00

Triggers to Start Mission

□ Time > H + 0 Days 00:00:00 OR

0%

□ Mission(s) Satisfied OR

Missions to check

Mission Pool: Mechanized Assault South, Light Tanks Destruction, Mechanized Assault North, Air Superiority Patrol

□ Lua script Edit Script OR

Description : (25 char max)

Triggers to Tag Mission as Satisfied

☑ Time Elapsed 0 Days 00:45:00 OR

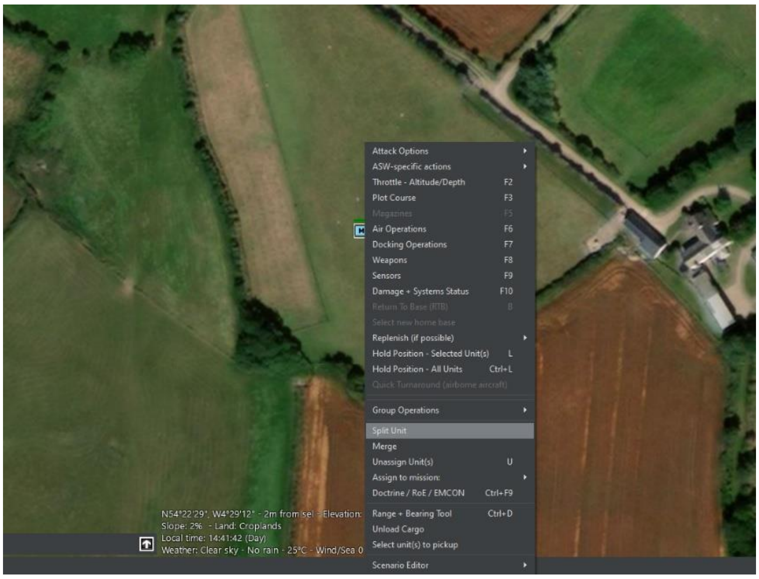
0% Reset

□ Lua script Edit Script OR

Description : (25 char max)

How to Split/Merge ground units (facilities)

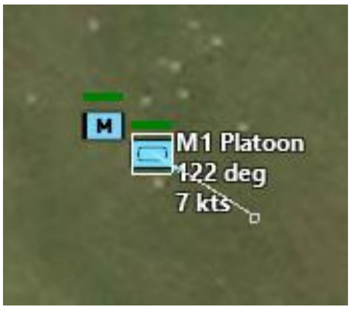
Legacy ground units associated as “facilities” can now be rearranged through this new tool.

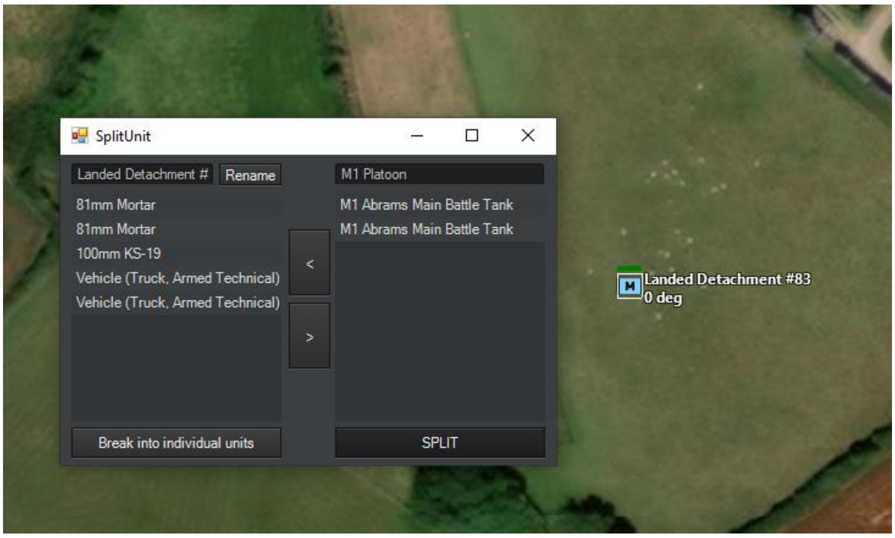


N54°22'29", W4°29'12" - 2m from sel - Elevation:
Slope: 2% - Land: Croplands
Local time: 14:41:42 (Day)
Weather: Clear sky - No rain - 25°C - Wind/Sea 0

Select an eligible unit, such as a landed detachment, right click on it to bring the context menu and click on “Split unit”.

This brings you this menu, with the details of the detachment.





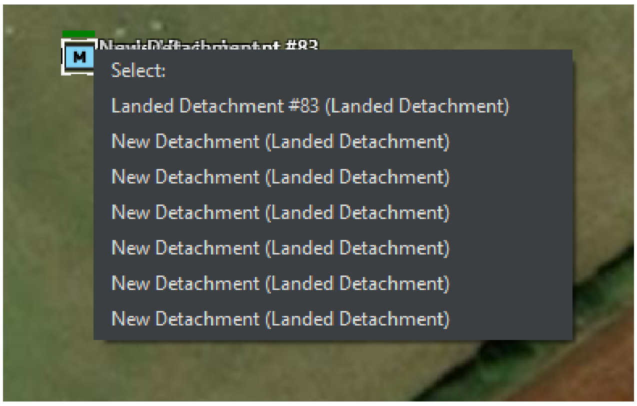
Landed Detachment #

81mm Mortar
81mm Mortar
100mm KS-19
Vehicle (Truck, Armed Technical)
Vehicle (Truck, Armed Technical)

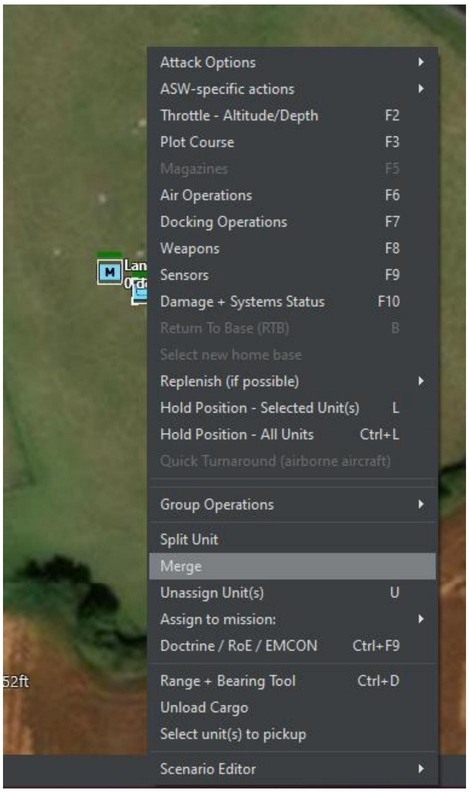
M1 Platoon

M1 Abrams Main Battle Tank
M1 Abrams Main Battle Tank

You can also break the detachment into individual units with a single click


Select:
Landed Detachment #83 (Landed Detachment)
New Detachment (Landed Detachment)
New Detachment (Landed Detachment)
New Detachment (Landed Detachment)
New Detachment (Landed Detachment)
New Detachment (Landed Detachment)
New Detachment (Landed Detachment)

Two eligible units can be merge together:



Mission Lua Wrapper

PriorityWeight (Read/Write) [int]

How important this mission is (0 = most important) affects multi-mission units assignment dynamic.

OperationName (Read/Write) [string]

Used to organize mission into groups, it is an informational element shown on the operation planner UI.

Completion (Read/Write) [float]

From 0.0 to 1.0, it designates how complete this mission is. This value is mostly unused in most cases at the moment and should be ignored.

Phase (Read/Write) [ENUM]

The phase as interpreted by the operation planner, this rules the whole dynamic in the operation planner and the assignment of multi-mission units.

MissionPhase ENUM:

None = 999

Auto = 0

Completed = 10

Active = 20

OnHold = 30

MISSION START BY TIME TRIGGER

MissionStartTrigger_Time (Read/Write) [int]

The H +/- time condition for the trigger.

MissionStartTrigger_Time_Enabled (Read/Write) [Bool]

Is the trigger used or not?

MissionStartTrigger_Time_LastResult (Read) [Bool]

What was the last result of the trigger the last time it has been evaluated?

MissionStartTrigger_Time_Operator (Read/Write) [Bool]

The trigger's operator. True = 'AND', False = 'OR'

MISSION START BY MISSION COMPLETED

MissionStartTrigger_MissionCompleted (Read/Write) [NLua.LuaTable]

The condition for the trigger, defined as all the missions that need to be completed (phase as "satisfied") for the trigger to check.

Exemple :

```
>>  local mission = ScenEdit_GetMission('USA','Shore A LCAC 1 #4 Wave 1')
print(mission.MissionStartTrigger_MissionCompleted)
mission.MissionStartTrigger_MissionCompleted = {"LTACBA-0HMBO9DVSMOPI","LTACBA-0HMBO9DVSMOP9","LTACBA-0HMBO9DVSMOPB"}
print(mission.MissionStartTrigger_MissionCompleted)
{ [1] = 'LTACBA-0HMBO9DVSMOPI' }
{ [1] = 'LTACBA-0HMBO9DVSMOPI', [2] = 'LTACBA-0HMBO9DVSMOP9', [3] = 'LTACBA-0HMBO9DVSMOPB' }
```

MissionStartTrigger_MissionCompleted_Enabled (Read/Write) [Bool]

Is the trigger used or not?

MissionStartTrigger_MissionCompleted_LastResult (Read) [Bool]

What was the last result of the trigger the last time it has been evaluated?

MissionStartTrigger_MissionCompleted_Operator (Read/Write) [Bool]

The trigger's operator. True = 'AND', False = 'OR'

MISSION START BY LUA CONDITION

MissionStartTrigger_LUADescription (Read/Write) [String]

The label of the lua script as an indicator of its purpose.

MissionStartTrigger_LUA (Read/Write) [String]

The lua script to be executed for this trigger conditions, must return a boolean as a condition result.

MissionStartTrigger_LUA_Enabled (Read/Write) [Bool]

Is the trigger used or not?

MissionStartTrigger_LUA_LastResult (Read) [Bool]

What was the last result of the trigger the last time it has been evaluated?

MissionStartTrigger_LUA_Operator (Read/Write) [Bool]

The trigger's operator. True = 'AND', False = 'OR'

MISSION COMPLETED ("SATISFIED") BY ELAPSED TIME OF MISSION EXECUTION

MissionCompletedTrigger_ElapsedTime (Read/Write) [int]

The elapsed time since mission execution for the trigger's condition to return true.

MissionCompletedTrigger_ElapsedTime_Current (Read/Write) [int]
The current elapsed time.

MissionCompletedTrigger_ElapsedTime_Enabled (Read/Write) [Bool]
Is the trigger used or not?

MissionCompletedTrigger_ElapsedTime_LastResult (Read) [Bool]
What was the last result of the trigger the last time it has been evaluated?

MissionCompletedTrigger_ElapsedTime_Operator (Read/Write) [Bool]
The trigger's operator. True = 'AND', False = 'OR'

MISSION COMPLETED ("SATISFIED") BY LUA CONDITION

MissionCompletedTrigger_LUADescription (Read/Write) [String]
The label of the lua script as an indicator of its purpose.

MissionCompletedTrigger_LUA (Read/Write) [String]
The lua script to be executed for this trigger conditions, must return a boolean as a condition result.

MissionCompletedTrigger_LUA_Enabled (Read/Write) [Bool]
Is the trigger used or not?

MissionCompletedTrigger_LUA_LastResult (Read) [Bool]
What was the last result of the trigger the last time it has been evaluated?

MissionCompletedTrigger_LUA_Operator (Read/Write) [Bool]
The trigger's operator. True = 'AND', False = 'OR'

Operation Lua Wrapper

The 'Operation' object can be fetched from the side object. There is only one operation object per side.

LHourMission (Read/Write) [LuaWrapper_Mission]

The mission designated to start at L-Hour. As soon as the L-Hour is reached, the mission will change phase to "Running".

HHourMission (Read/Write) [LuaWrapper_Mission]

The mission designated to start at H-Hour. As soon as the H-Hour is reached, the mission will change phase to “Running”.

HHour (Read/Write) [String]

The time set as H-Hour.

LHour (Read/Write) [String]

The time set as L-Hour.

HHourEffectiveStartTime (Read/Write) [String]

The effective time at which the H-Hour started counting. Sometimes the actual H-Hour can be different from the one planned initially, or in the past.

LHourEffectiveStartTime (Read/Write) [String]

The effective time at which the L-Hour started counting. Sometimes the actual L-Hour can be different from the one planned initially, or in the past.

H_LHourAreRelative (Read/Write) [String]

Whether or not the H-Hour and the L-Hour have a static time delta. With this option on TRUE, the H-hour and L-hour will always have the same time difference and will adjust automatically when changing either the H-Hour or the L-Hour

Unit Lua Wrapper

AllowMultiMission (Read/Write) [Bool]

Is the mission allowed to have multiple missions assigned to it and will it be considered by the operation planner logic?

AssignedMissionsQueue (Read/Write) [NLua.LuaTable]

The list of missions assigned to this unit. This is the missions the operation planner evaluate every tick.

Serial/Chalk Lua Wrapper

Serials/Chalks can be fetched from the side object the name of the collection is “Chalks”.

ID (Read/Write) [Int]

The serial’s visible ID.

AssociatedMothership (Read) [string]

The GUID of the mothership this chalk is associated to.

LargestCargoType (Read) [CargoType ENUM]

Get the largest cargo this serial currently contains.

Enum CargoType:
NoCargo = 0
Personnel = 1000
SmallCargo = 2000
MediumCargo = 3000
LargeCargo = 4000
VLargeCargo

Mass (Read) [float]

The total mass of the serial.

Area (Read) [float]

The total area taken by the serial.

PAX (Read) [float]

The total personnel this serial contains.

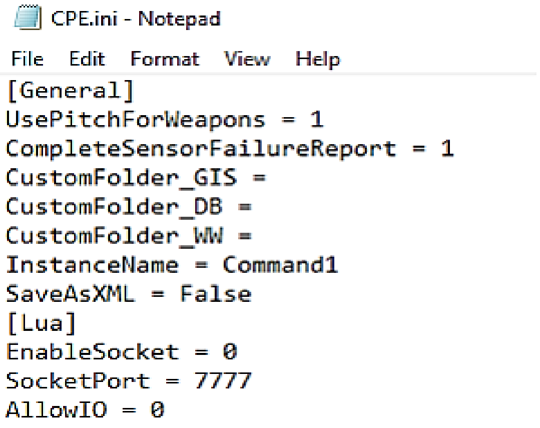
Appendix 4: Complete Sensor Detection Reports

The “Sensor Detection” event type exported by Command PE’s event-export framework, while providing a lot of useful contextual information on a detection check, does not provide more detailed feedback on specifically why a detection failed. CPE’s sensor model is comprehensive in its mirroring of real-life factors that can prevent a detection (everything from line-of-sight blockage to the detrimental effect of weather conditions, for example), and sometimes it is useful to an analyst to understand which specific factors are the culprits in a detection failure.

Beginning with v2.2, Command PE can provide a complete sensor detection failure report, if desired. For each failed sensor detection check, this full report will lists every possible failure condition that either caused the detection failure, or would have caused a detection failure in the absence of other conditions. A complete sensor detection report can be can be helpful to validate sensors created in a custom database or to gain a deeper understanding of your scenarios analysis results.

IMPORTANT NOTE*: By default, Command’s sensor detection pipeline is optimized for performance & scalability, with careful arrangement of the detection steps to enable speed-enhancing shortcuts to be taken where possible (for example, CPU-inexpensive checks are performed first, so if these fail then more expensive operations can be skipped). Enabling complete sensor detection reports will force the simulation to completely ignore such shortcuts and evaluate every possible sensor condition that could have resulted in a detection failure. This requires additional CPU processing for every failed sensor detection check, and this is likely to significantly reduce the speed at which the simulation runs. Therefore, complete sensor failure reports should be carefully managed and enabled only when you require this additional data.*

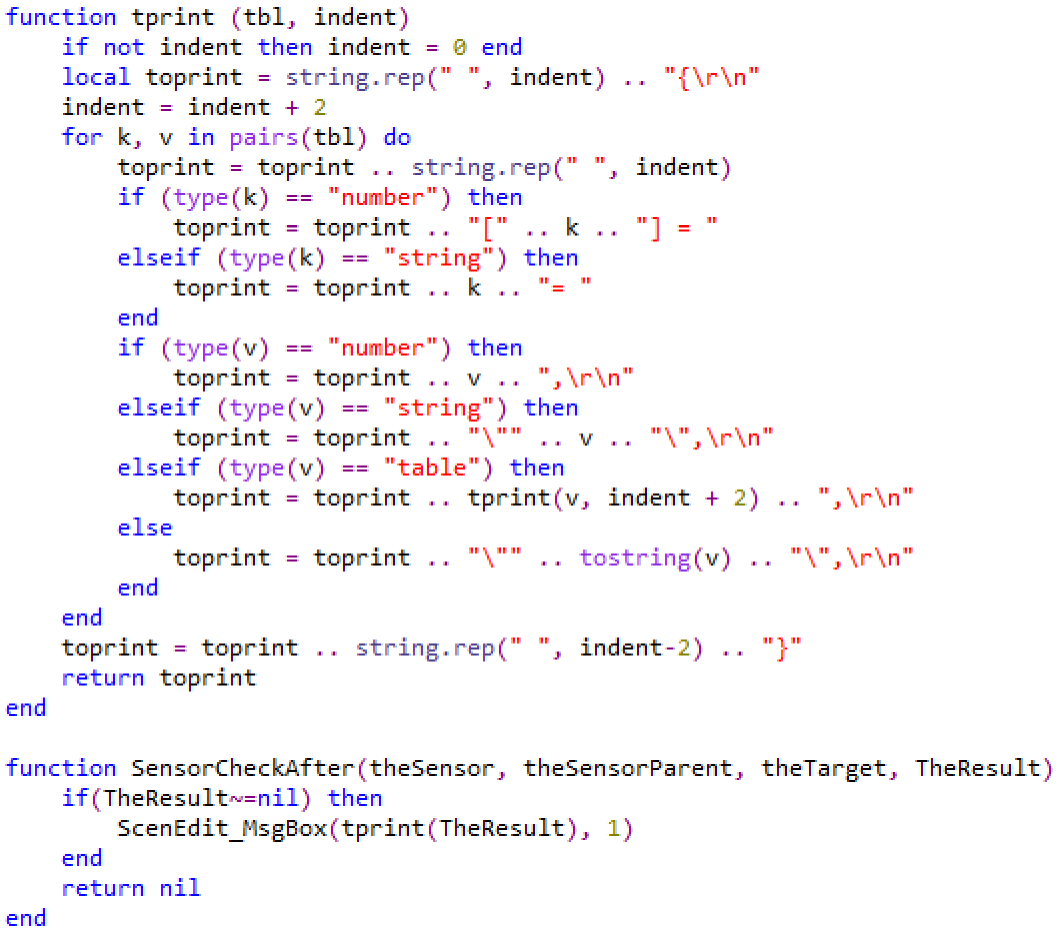
To enable complete sensor failure reports, add the line “CompleteSensorFailureReports = 1” to the ‘General’ section of the CPE.INI text file in your Command Professional Edition ‘Config’ folder and restart Command.



```
[General]
UsePitchForWeapons = 1
CompleteSensorFailureReport = 1
CustomFolder_GIS =
CustomFolder_DB =
CustomFolder_WW =
InstanceName = Command1
SaveAsXML = False
[Lua]
EnableSocket = 0
SocketPort = 7777
AllowIO = 0
```

To retrieve the data for a complete sensor failure report you must then add a SensorCheckAfter Lua event hook to your scenario. For information on this hook, please see the manual section on ‘Lua Event Hooks’ above. The fourth parameter in your SensorCheckAfter Lua hook function will be passed a table of failure results which can then be parsed, displayed, or written to a file by your Lua script, as desired.

Each entry in the failure results Lua table will contain a pair of values – the first a numeric value indicating the relative order the failure cases were evaluated in, and the second a specific reason for each failure case. The following script demonstrates one way you might process and report the failure cases. In this example the results from the failure report is passed to the function ‘tprint’ to convert them to a human-readable format, then displayed with a message box.



```
function tprint (tbl, indent)
    if not indent then indent = 0 end
    local toprint = string.rep(" ", indent) .. "{\r\n"
    indent = indent + 2
    for k, v in pairs(tbl) do
        toprint = toprint .. string.rep(" ", indent)
        if (type(k) == "number") then
            toprint = toprint .. "[" .. k .. "] = "
        elseif (type(k) == "string") then
            toprint = toprint .. k .. "= "
        end
        if (type(v) == "number") then
            toprint = toprint .. v .. ",\r\n"
        elseif (type(v) == "string") then
            toprint = toprint .. "\"" .. v .. "\",\r\n"
        elseif (type(v) == "table") then
            toprint = toprint .. tprint(v, indent + 2) .. ",\r\n"
        else
            toprint = toprint .. "\"" .. tostring(v) .. "\",\r\n"
        end
    end
    toprint = toprint .. string.rep(" ", indent-2) .. "}"
    return toprint
end

function SensorCheckAfter(theSensor, theSensorParent, theTarget, TheResult)
    if(TheResult~=nil) then
        ScenEdit_MsgBox(tprint(TheResult), 1)
    end
    return nil
end
```

Sensor Detection Results Hierarchy

Complete sensor failure reports will enumerate all possible sources of sensor detection failures starting with the most general-case failure and then proceeding to more sensor-specific failure cases, listing all of the failure cases that apply to that detection attempt. The reported results were evaluated by the simulation in the order given by the first value in each entry in the results Lua table (enumerated from 0…N) while the corresponding reason for failure can be found in the lists given below.

A successful sensor detection attempt will report a single Lua table entry with the index ‘0’ and a GENERAL_SuccessfullDetection_<type> result.

General Failures

```
GENERAL_Range = 1
GENERAL_TargetSuitability = 3
GENERAL_CoveringArc = 5
GENERAL_MaxAltitude_AGL = 7
GENERAL_MinAltitude_AGL = 9
GENERAL_MaxAltitude_ASL = 11
GENERAL_MinAltitude_ASL = 13
GENERAL_Other = 15
```

```
GENERAL_WeaponESM_UnsuitableTarget_Satellite = 17
GENERAL_WeaponESM_UnsuitableTarget_AircraftOrWeapon = 19
GENERAL_WeaponESM_UnsuitableTarget_ShipOrFacility = 21

GENERAL_SuccessfullDetection_FromOneOfThePCLS = 22
GENERAL_FailedDetection_FromAllThePCLS = 23

GENERAL_SuccessfullDetection_IR = 24
GENERAL_FailedDetection_IR = 25

GENERAL_SuccessfullDetection_PingIntercept = 26
GENERAL_FailedDetection_PingIntercept = 27

GENERAL_SuccessfullDetection_Visual = 28
GENERAL_FailedDetection_Visual = 29

GENERAL_SuccessfullDetection_Radar = 30
GENERAL_FailedDetection_Radar = 31

GENERAL_SuccessfullDetection_ESM = 32
GENERAL_FailedDetection_ESM = 33

GENERAL_SuccessfullDetection_HullandBottomFixed_PassiveSonar = 34
GENERAL_FailedDetection_HullandBottomFixed_PassiveSonar = 35

GENERAL_SuccessfullDetection_HullandBottomFixed_ActiveSonar = 36
GENERAL_FailedDetection_HullandBottomFixed_ActiveSonar = 37

GENERAL_SuccessfullDetection_DippingAndVDS_PassiveSonar = 38
GENERAL_FailedDetection_DippingAndVDS_PassiveSonar = 39

GENERAL_SuccessfullDetection_DippingAndVDS_ActiveSonar = 40
GENERAL_FailedDetection_DippingAndVDS_ActiveSonar = 41

GENERAL_SuccessfullDetection_MAD = 42
GENERAL_FailedDetection_MAD = 43
```

RADAR Specific Failure Cases (100-199)

```
RADAR_RadarLOS_SurfaceWave = 101
RADAR_RadarLOS = 103
RADAR_RadarLOS_Terrain = 105
RADAR_PureIlluminatorFailedDetection = 107
RADAR_PureIlluminatorSuccessfulDetection = 109
RADAR_SubUnitAbovePeriscopeDepthNotDetectable_BiologicsOrFalseContact = 111
RADAR_SubUnitAbovePeriscopeDepthNotDetectable_NoPeriscopeOrSnorkel = 113
RADAR_CantDetectPeriscope = 115
RADAR_BlindCone = 117
RADAR_LDSD_Limited_LookDown15deg = 119
RADAR_LDSD_Limited_LookDown5deg = 121
RADAR_Other = 123
RADAR_ECM_Equation = 125
RADAR_Successful_OTHBackscatter = 126
RADAR_ABM_HorizonFailed = 127
RADAR_ABM_HorizonSuccessful = 129
```

Visual Specific Failure Cases (200-299)

```
VISUAL_VerticalViewAngle45deg = 201
VISUAL_AverageCockpitVisibility_VerticalViewAngle25deg = 203
```

```
        VISUAL_RNG_ExcellentCockpitVisibility = 205
        VISUAL_RNG_AverageCockpitVisibility = 207
        VISUAL_RNG_PoorCockpitVisibility = 209
        VISUAL_LOS = 211
        VISUAL_MaxNominalRange = 213
        VISUAL_MaxSpecificRange = 215
        VISUAL_ABM_Horizon_Failed = 217
        VISUAL_ABM_Horizon_Successful = 218


IR Specific Failure Cases (400-499)

        IR_LOSVisual = 401
        IR_MaxRange = 403
        IR_MaxRange_SpecificToTarget = 405
        IR_NoPeriscopeSearchCapability = 407
        IR_ABM_BeyondHorizon = 409


Ping-Intercept Specific Failure Cases (500-599)

        PING_LOSSonar = 501
        PING_MaxRange = 503
        PING_NoInterceptionFromAnyEmitter = 505


Active Sonar Specific Failure Cases (600-699)

        ACTIVESONAR_NotOperating = 601
        ACTIVESONAR_LOSSonar = 603
        ACTIVESONAR_AboveDeafSpeed = 605
        ACTIVESONAR_MaxRangeSpecificTarget = 607
        ACTIVESONAR_SuccessfulDirectDetection = 608
        ACTIVESONAR_SuccessfulIndirectDetection = 610
        ACTIVESONAR_SuccessfulBottomBounceDetection = 612
        ACTIVESONAR_SuccessfulDetectionThroughCZ = 614
        ACTIVESONAR_Other = 615


Passive Sonar Specific Failure Cases (700-799)

        PASSIVESONAR_NotOperating = 701
        PASSIVESONAR_SuccessfulPingIntercept = 702
        PASSIVESONAR_FailedPingIntercept = 703
        PASSIVESONAR_AboveDeafSpeed = 705
        PASSIVESONAR_LOSSonar = 707
        PASSIVESONAR_MaxNominalRange = 709
        PASSIVESONAR_SignalMaskedByAnother = 711
        PASSIVESONAR_MaxActualRange = 713
        PASSIVESONAR_LandmassInBetween = 715
        PASSIVESONAR_SuccessfulDetectionThroughCZ = 716
        PASSIVESONAR_CZBlockingAndNoDirectPath = 717


MAD Specific Failure Cases (800-899)

        MAD_UnitIsSubBiologics = 801
        MAD_UnitIsTorpedo = 803
        MAD_MaxEffectiveRange = 805
        MAD_IsNotBoatType = 807
```

RADAR Illuminator Specific Failure Cases (900-999)

```
        RADARILLUMINATION_RadarHorizonLOS_SurfaceWave = 901
        RADARILLUMINATION_RadarHorizonLOS = 903
        RADARILLUMINATION_RadarHorizonLOS_Failed = 905
        RADARILLUMINATION_CoveringArc = 907
        RADARILLUMINATION_Other = 909
        RADARILLUMINATION_ECMEquation = 911
        RADARILLUMINATION_ABM_Horizon_Failure = 913
        RADARILLUMINATION_ABM_Horizon_Success = 914
        RADARILLUMINATION_OTH_Success = 916
        RADARILLUMINATION_OTH_Failure = 917
```

ESM Specific Failure Cases (1000-1099)

```
        ESM_RadarHorizonLOS_SurfaceWave = 1001
        ESM_RadarHorizonLOS = 1003
        ESM_RadarHorizonLOS_Terrain = 1005
        ESM_RWRvsJamming = 1007
        ESM_Equation = 1009
        ESM_OTHSW_Emitter_MaxRange = 1011
        ESM_OTHSW_Emitter_LOS = 1013
        ESM_EmitterNotActive = 1015
        ESM_IncompatibleFrequency = 1017
        ESM_EmitterNotOperating = 1019
        ESM_MaxRange = 1021
        ESM_Success = 1022
        ESM_Failure = 1023
```

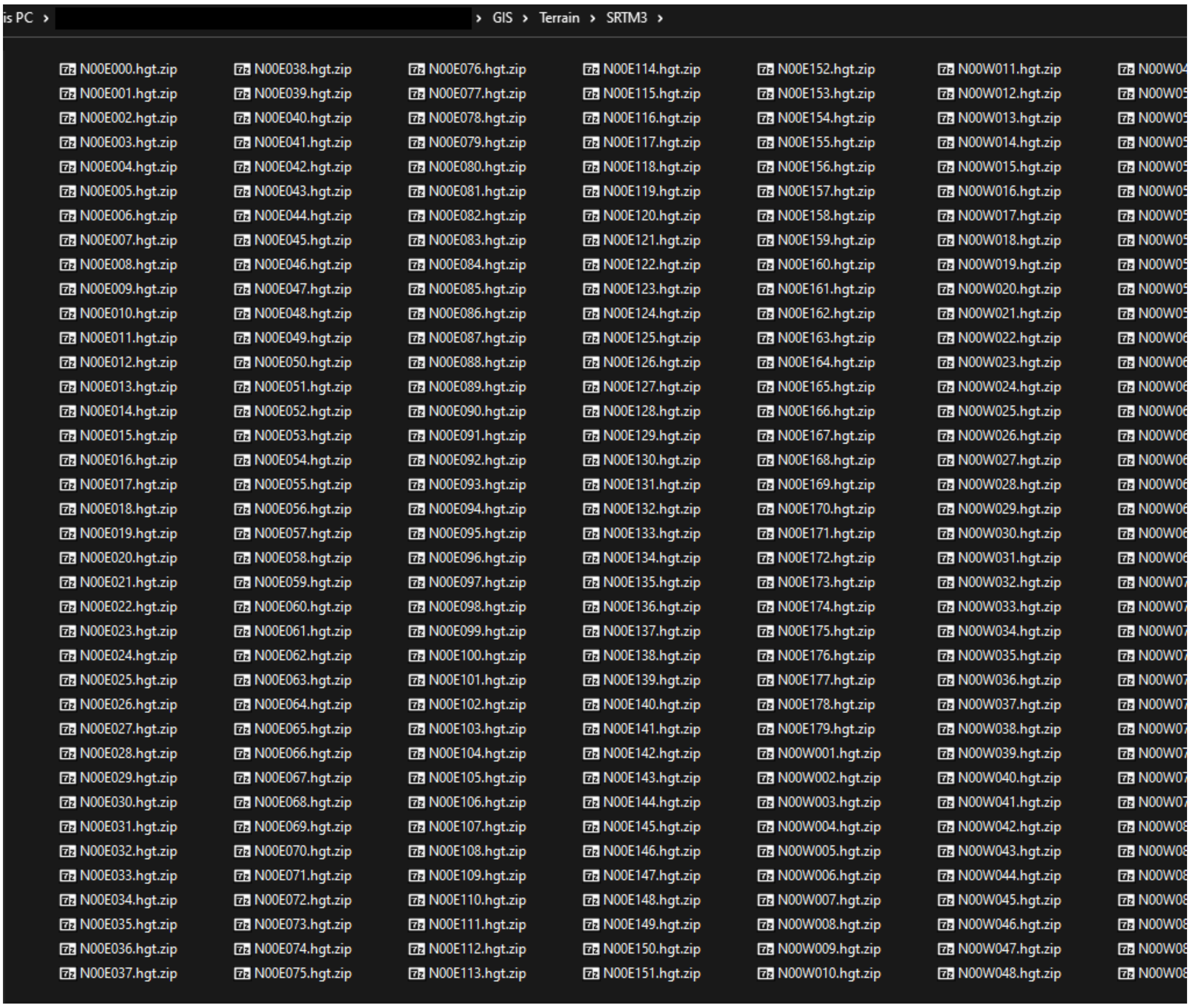
Appendix 5: Custom Terrain Editing: How to make a fictional island

With the introduction of the SRTM3 terrain in CPE 2.0+, it is now much easier to alter the elevation data for any area in the world; the classic example of this requirement is “I want to make a hypothetical island in the middle of the ocean/sea”. So let us consider the steps necessary for this.

NOTE: *Although it is technically possible to alter the terrain elevation of the legacy SRTM30PLUS elevation data, the process has complications and gotchas that make it more trouble than it’s worth, especially with the availability of the new SRTM3 elevation set. For this reason, only editing SRTM3 data is supported. This means that, as a pre-requisite, the add-on “HD pack” must be installed prior to the procedure.*

1) Altering the terrain elevation file(s)

The first step in creating our fictional island is to edit the actual terrain files. The SRTM3 files are stored by default in the folder [writable root]\GIS\Terrain\SRTM3. Here is an example of said folder:



is PC › [redacted] › GIS › Terrain › SRTM3 ›

```
N00E000.hgt.zip
N00E001.hgt.zip
N00E002.hgt.zip
N00E003.hgt.zip
N00E004.hgt.zip
N00E005.hgt.zip
N00E006.hgt.zip
N00E007.hgt.zip
N00E008.hgt.zip
N00E009.hgt.zip
N00E010.hgt.zip
N00E011.hgt.zip
N00E012.hgt.zip
N00E013.hgt.zip
N00E014.hgt.zip
N00E015.hgt.zip
N00E016.hgt.zip
N00E017.hgt.zip
N00E018.hgt.zip
N00E019.hgt.zip
N00E020.hgt.zip
N00E021.hgt.zip
N00E022.hgt.zip
N00E023.hgt.zip
N00E024.hgt.zip
N00E025.hgt.zip
N00E026.hgt.zip
N00E027.hgt.zip
N00E028.hgt.zip
N00E029.hgt.zip
N00E030.hgt.zip
N00E031.hgt.zip
N00E032.hgt.zip
N00E033.hgt.zip
N00E034.hgt.zip
N00E035.hgt.zip
N00E036.hgt.zip
N00E037.hgt.zip
N00E038.hgt.zip
N00E039.hgt.zip
N00E040.hgt.zip
N00E041.hgt.zip
N00E042.hgt.zip
N00E043.hgt.zip
N00E044.hgt.zip
N00E045.hgt.zip
N00E046.hgt.zip
N00E047.hgt.zip
N00E048.hgt.zip
N00E049.hgt.zip
N00E050.hgt.zip
N00E051.hgt.zip
N00E052.hgt.zip
N00E053.hgt.zip
N00E054.hgt.zip
N00E055.hgt.zip
N00E056.hgt.zip
N00E057.hgt.zip
N00E058.hgt.zip
N00E059.hgt.zip
N00E060.hgt.zip
N00E061.hgt.zip
N00E062.hgt.zip
N00E063.hgt.zip
N00E064.hgt.zip
N00E065.hgt.zip
N00E066.hgt.zip
N00E067.hgt.zip
N00E068.hgt.zip
N00E069.hgt.zip
N00E070.hgt.zip
N00E071.hgt.zip
N00E072.hgt.zip
N00E073.hgt.zip
N00E074.hgt.zip
N00E075.hgt.zip
N00E076.hgt.zip
N00E077.hgt.zip
N00E078.hgt.zip
N00E079.hgt.zip
N00E080.hgt.zip
N00E081.hgt.zip
N00E082.hgt.zip
N00E083.hgt.zip
N00E084.hgt.zip
N00E085.hgt.zip
N00E086.hgt.zip
N00E087.hgt.zip
N00E088.hgt.zip
N00E089.hgt.zip
N00E090.hgt.zip
N00E091.hgt.zip
N00E092.hgt.zip
N00E093.hgt.zip
N00E094.hgt.zip
N00E095.hgt.zip
N00E096.hgt.zip
N00E097.hgt.zip
N00E098.hgt.zip
N00E099.hgt.zip
N00E100.hgt.zip
N00E101.hgt.zip
N00E102.hgt.zip
N00E103.hgt.zip
N00E104.hgt.zip
N00E105.hgt.zip
N00E106.hgt.zip
N00E107.hgt.zip
N00E108.hgt.zip
N00E109.hgt.zip
N00E110.hgt.zip
N00E111.hgt.zip
N00E112.hgt.zip
N00E113.hgt.zip
N00E114.hgt.zip
N00E115.hgt.zip
N00E116.hgt.zip
N00E117.hgt.zip
N00E118.hgt.zip
N00E119.hgt.zip
N00E120.hgt.zip
N00E121.hgt.zip
N00E122.hgt.zip
N00E123.hgt.zip
N00E124.hgt.zip
N00E125.hgt.zip
N00E126.hgt.zip
N00E127.hgt.zip
N00E128.hgt.zip
N00E129.hgt.zip
N00E130.hgt.zip
N00E131.hgt.zip
N00E132.hgt.zip
N00E133.hgt.zip
N00E134.hgt.zip
N00E135.hgt.zip
N00E136.hgt.zip
N00E137.hgt.zip
N00E138.hgt.zip
N00E139.hgt.zip
N00E140.hgt.zip
N00E141.hgt.zip
N00E142.hgt.zip
N00E143.hgt.zip
N00E144.hgt.zip
N00E145.hgt.zip
N00E146.hgt.zip
N00E147.hgt.zip
N00E148.hgt.zip
N00E149.hgt.zip
N00E150.hgt.zip
N00E151.hgt.zip
N00E152.hgt.zip
N00E153.hgt.zip
N00E154.hgt.zip
N00E155.hgt.zip
N00E156.hgt.zip
N00E157.hgt.zip
N00E158.hgt.zip
N00E159.hgt.zip
N00E160.hgt.zip
N00E161.hgt.zip
N00E162.hgt.zip
N00E163.hgt.zip
N00E164.hgt.zip
N00E165.hgt.zip
N00E166.hgt.zip
N00E167.hgt.zip
N00E168.hgt.zip
N00E169.hgt.zip
N00E170.hgt.zip
N00E171.hgt.zip
N00E172.hgt.zip
N00E173.hgt.zip
N00E174.hgt.zip
N00E175.hgt.zip
N00E176.hgt.zip
N00E177.hgt.zip
N00E178.hgt.zip
N00E179.hgt.zip
N00W001.hgt.zip
N00W002.hgt.zip
N00W003.hgt.zip
N00W004.hgt.zip
N00W005.hgt.zip
N00W006.hgt.zip
N00W007.hgt.zip
N00W008.hgt.zip
N00W009.hgt.zip
N00W010.hgt.zip
N00W011.hgt.zip
N00W012.hgt.zip
N00W013.hgt.zip
N00W014.hgt.zip
N00W015.hgt.zip
N00W016.hgt.zip
N00W017.hgt.zip
N00W018.hgt.zip
N00W019.hgt.zip
N00W020.hgt.zip
N00W021.hgt.zip
N00W022.hgt.zip
N00W023.hgt.zip
N00W024.hgt.zip
N00W025.hgt.zip
N00W026.hgt.zip
N00W027.hgt.zip
N00W028.hgt.zip
N00W029.hgt.zip
N00W030.hgt.zip
N00W031.hgt.zip
N00W032.hgt.zip
N00W033.hgt.zip
N00W034.hgt.zip
N00W035.hgt.zip
N00W036.hgt.zip
N00W037.hgt.zip
N00W038.hgt.zip
N00W039.hgt.zip
N00W040.hgt.zip
N00W041.hgt.zip
N00W042.hgt.zip
N00W043.hgt.zip
N00W044.hgt.zip
N00W045.hgt.zip
N00W046.hgt.zip
N00W047.hgt.zip
N00W048.hgt.zip
N00W04
N00W05
N00W05
N00W05
N00W05
N00W05
N00W05
N00W05
N00W05
N00W05
N00W05
N00W06
N00W06
N00W06
N00W06
N00W06
N00W06
N00W06
N00W06
N00W06
N00W06
N00W07
N00W07
N00W07
N00W07
N00W07
N00W07
N00W07
N00W07
N00W07
N00W07
N00W08
N00W08
N00W08
N00W08
N00W08
N00W08
N00W08
```

The data is split into individual files, each representing a 1x1-degree grid on earth. Their naming makes it easy to identify which region each file represents; for example, the file N88W098 covers the 88 deg North, 98 deg West grid.

IMPORTANT: *Before proceeding, make sure that you back up all the terrain files that you intend to modify. Assuming that you know in advance the coordinates of the area you will edit, this is a matter of finding (through the naming scheme) and backing up only the files that are going to be altered. If you are uncertain about which files to select, a complete backup of this folder is also possible (though rather storage size-intensive).*

The SRTM3 elevation files use the default SRTM file format, .HGT: https://www.lifewire.com/hgt-file-2621580 . This is a popular digital elevation model (DEM) format throughout the GIS industry, and is supported by numerous GIS or DEM applications such as ArcGIS, VTBuilder, DG Terrain Viewer, Global Mapper etc.

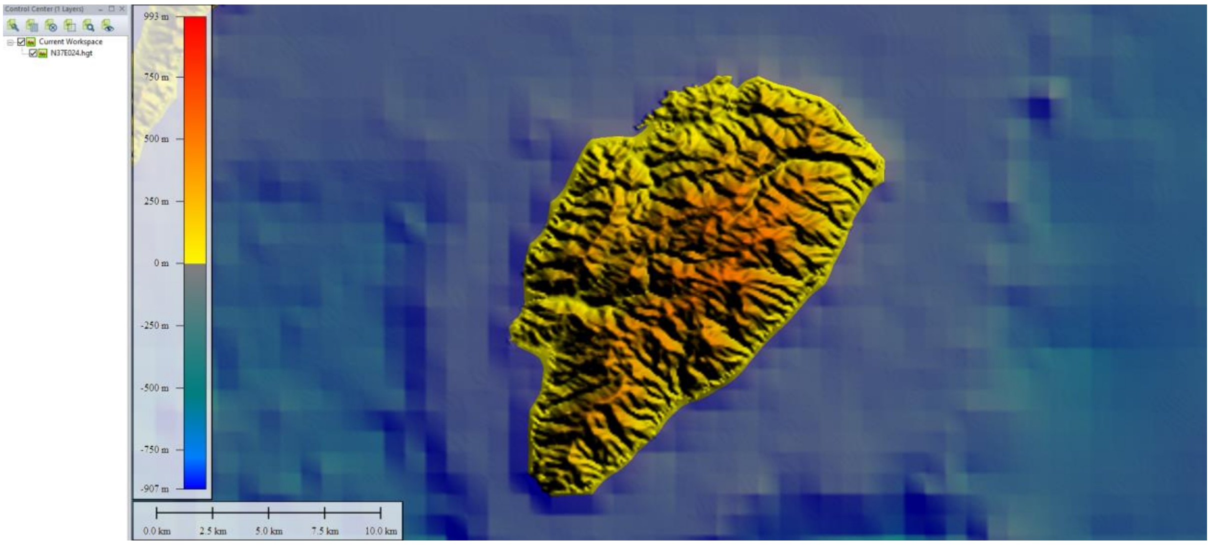
So: Find the grid file of interest, unzip it if necessary (as shown in the screenshot above, elevation grid files are by default packed in individual zip files to reduce their disk footprint) and load it up in your GIS/DEM application of choice. Perform the desired elevation changes, and save the affected grid file(s) again.

Overwrite the default elevation files with the modified copies, and restart CPE. Now if you hover the map cursor in the area where you have raised the elevation level, the elevation value displayed by the map cursor should map the changes you have made.

2) Applying a matching raster overlay

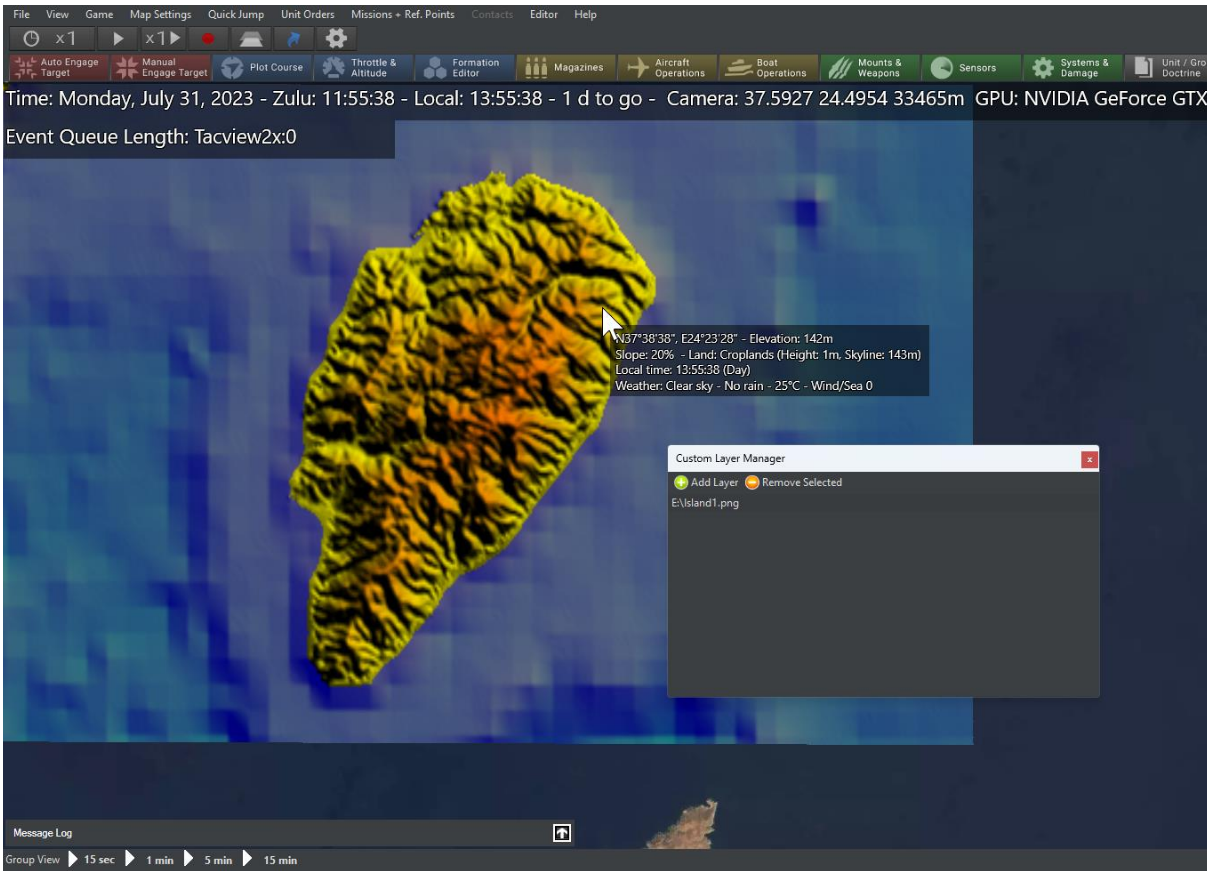
CPE's multitude of bundled map raster layers are based on actual real maps and are thus matched to the real-life global terrain elevation data. Thus, we now have a visual mismatch in the area where we created our fictional island: The built-in map layers show us empty sea, but out map cursor is clearly telling us there is above-ASL terrain elevation (and indeed we can place ground units on the raised area). How do we reconcile the mismatch? By creating and applying a custom raster overlay.

The simplest way (an accurate way too) to create this overlay is to again use a GIS/DEM application together with our custom elevation data. So again load the modified elevation grid, and use the application to create and export a suitable georeferenced raster image:



CPE can use raw-bitmap (bmp, **not recommended**), jpeg- and png-format files for overlays. Whatever the format, the overlay file must also have a matching geo-reference file (.bpw, .jgw and .pgw respectively) in order to be accurately placed on CPE's map.

Now we can go ahead and load this overlay in CPE:



Time: Monday, July 31, 2023 - Zulu: 11:55:38 - Local: 13:55:38 - 1 d to go - Camera: 37.5927 24.4954 33465m GPU: NVIDIA GeForce GTX

Event Queue Length: Tacview2x:0

N37°38'38", E24°23'28" - Elevation: 142m
Slope: 20% - Land: Croplands (Height: 1m, Skyline: 143m)
Local time: 13:55:38 (Day)
Weather: Clear sky - No rain - 25°C - Wind/Sea 0

Custom Layer Manager

Add Layer Remove Selected

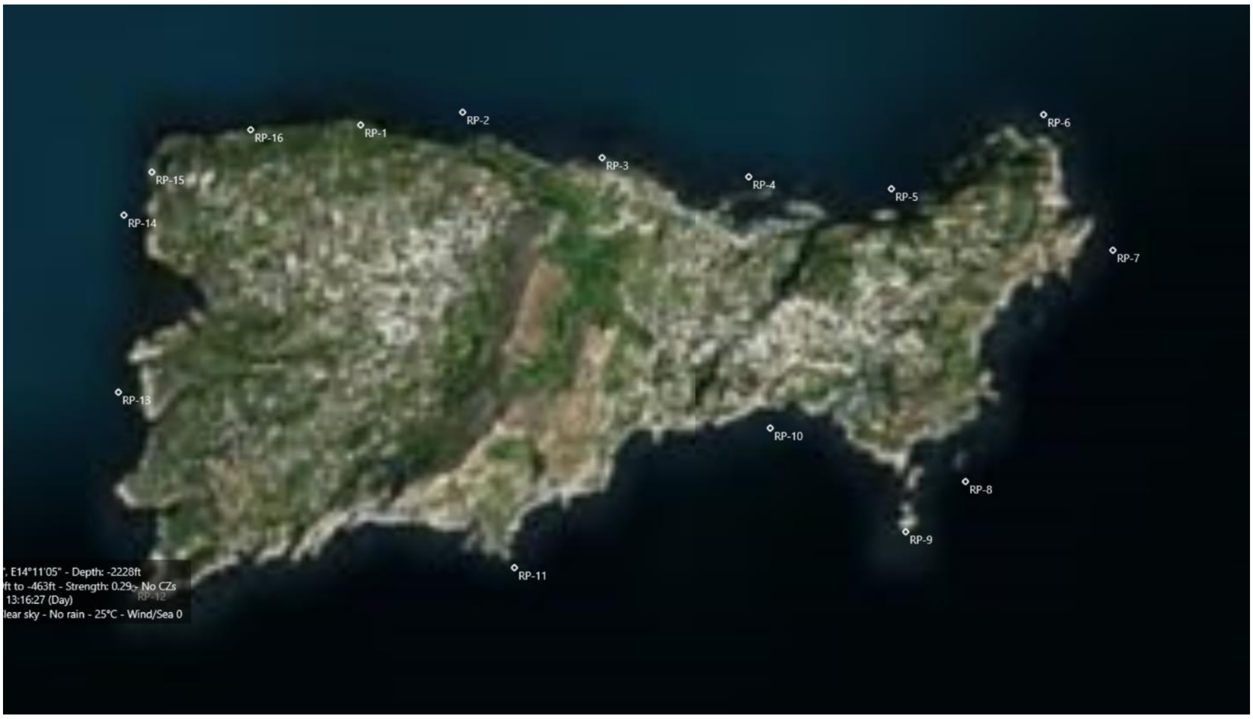
E:\Island1.png

Note that now the visual overlay matches our custom-defined elevation: We now actually see the fictional island we created, and the elevation values displayed at the map-cursor databox match our visual expectation.

3) Using CEZs to finetune terrain types & culture-height

Custom Environment Zones (CEZ) let the user create areas in which the terrain and weather data are custom defined; this gives Command an unprecedented flexibility. The user can have a snowy area adjacent to a warm desert and can even flood entire zones to simulate torrential rain or other natural disasters.
It is also entirely possible to rise a landmass in the middle of the ocean and define a specific weather for this new island.

The first step to create a CEZ is to define an area:



', E14°11'05" - Depth: -2228ft
ft to -463ft - Strength: 0.29 - No CZs
13:16:27 (Day)
lear sky - No rain - 25°C - Wind/Sea 0

After the area is defined, the user must open the Area & Reference Points Manager and click on **Cust. Env. Zones.** (Green square)

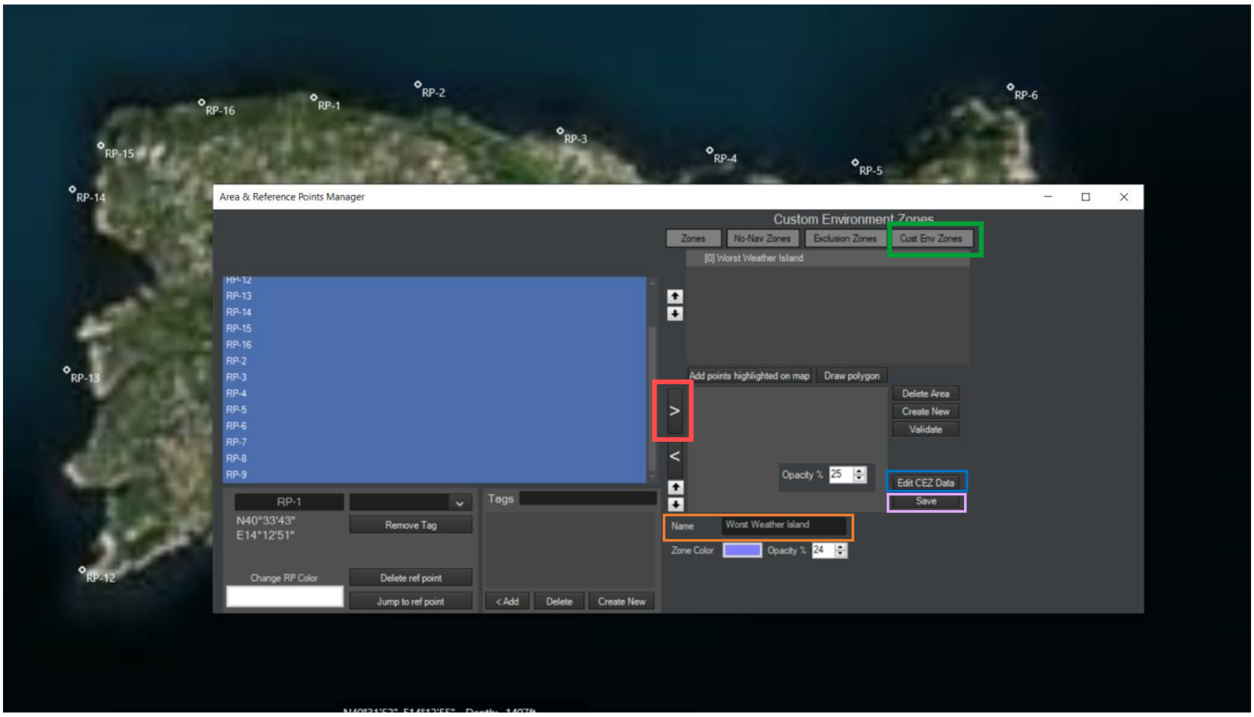
This will open the CEZ panel in which the user can define and edit CEZ data.

To do that the procedure is:

1) Define the CEZ Name (Orange square)

2) Select all the points that will define the perimeter of the new Zone and add them with the ">" button (Red Square)

3) Save so that the new CEZ is created (Purple Square)



RP-1
N40°33'43"
E14°12'51"

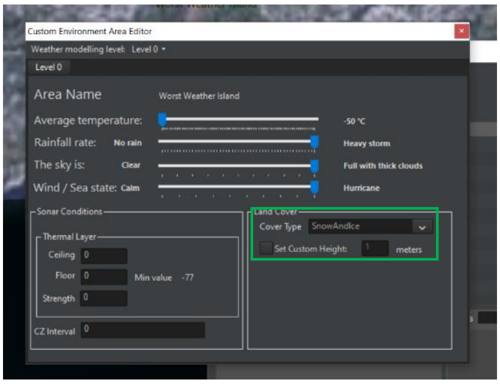
Opacity % 25

Name: Worst Weather Island

Zone Color: Opacity % 24

4) The user can now edit the environment of the CEZ by clicking on the Edit CEZ Data (Blue Square)

From this window plenty of data can be defined including, but not limited to, average temperature of the zone, rainfall rate, thermal layer (ignored by the simulation if the area is overland) and terrain type and altitude (Green Square)



Weather modelling level: Level 0

Area Name: Worst Weather Island

Average temperature: -50 °C

Rainfall rate: No rain – Heavy storm

The sky is: Clear – Full with thick clouds

Wind / Sea state: Calm – Hurricane

Ceiling: 0

Floor: 0 Min value -77

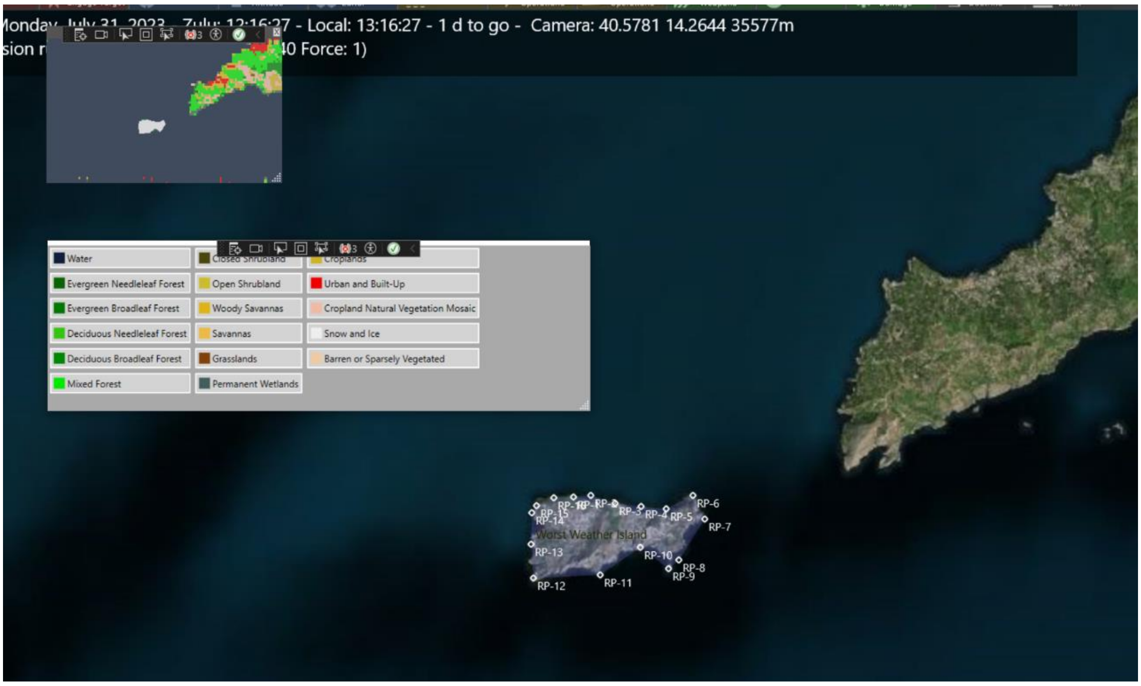
Strength: 0

CZ Interval: 0

Cover Type: SnowAndIce

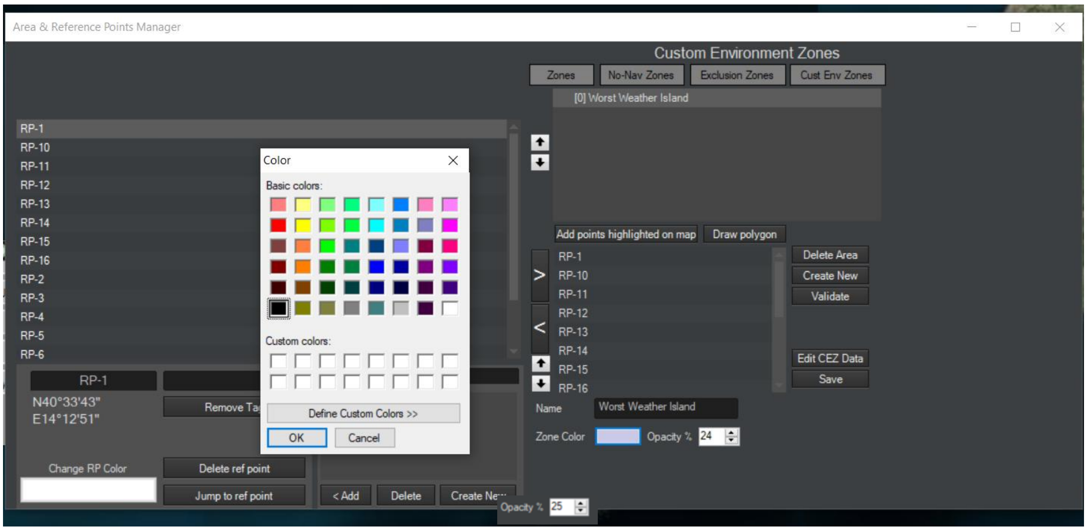
□ Set Custom Height: 1 meters

Once this is done simply close this window. You will immediately see the selected data being reflected on the map. In the attached picture we can see the just created CEZ (Snow and Ice) displayed on the map and the minimap.



londay, July 31, 2023 - Zulu: 12:16:27 - Local: 13:16:27 - 1 d to go - Camera: 40.5781 14.2644 35577m
sion r 0 Force: 1)

Another very requested feature was to be able to define a custom color and a custom transparency for every CEZ, this can be done in the window “Area & Reference Point Manager”:



Area & Reference Points Manager

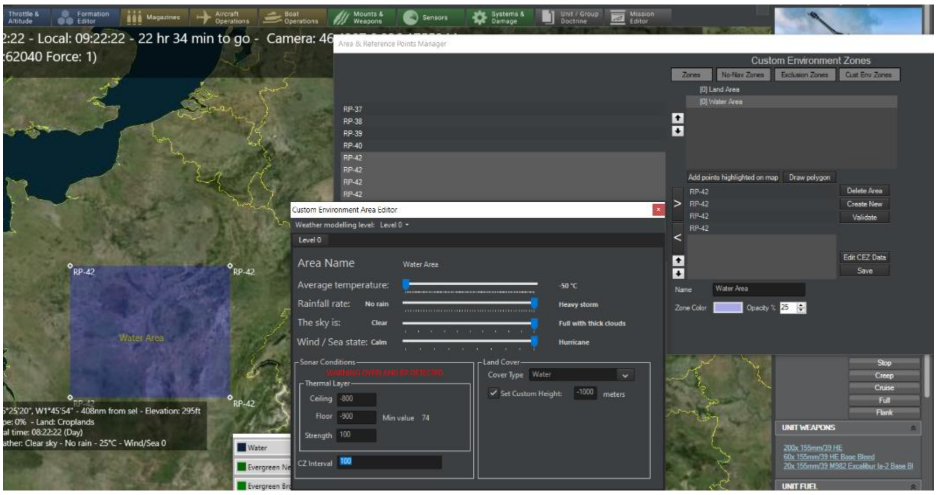
Custom Environment Zones

[0] Worst Weather Island

RP-1
N40°33'43"
E14°12'51"

Name: Worst Weather Island
Zone Color
Opacity %: 24
Opacity %: 25

As mentioned, it’s also possible to generate a water area in a place that would normally be a landmass. This is done in the same way as previously shown. The user only has to choose “Water” in the Land Cover section of the CEZ area Editor, it’s also possible to define Thermal layer and CZ Interval for the newly defined water area.



Custom Environment Area Editor

Weather modelling level: Level 0

Level 0

Area Name: Water Area

Average temperature: -50 °C

Rainfall rate: No rain – Heavy storm

The sky is: Clear – Full with thick clouds

Wind / Sea state: Calm – Hurricane

Sonar Conditions

WARNING! OVERLAND RP DETECTED

Thermal Layer

Ceiling: -800

Floor: -900 Min value 74

Strength: 100

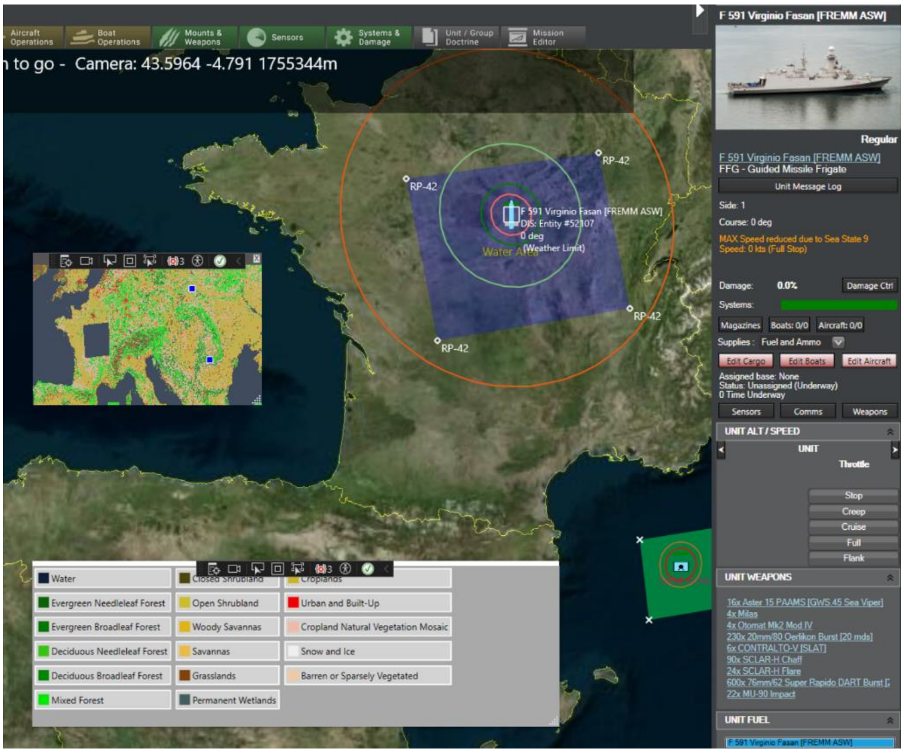
CZ Interval: 100

Land Cover

Cover Type: Water

☑ Set Custom Height: -1000 meters

Again, all the changes are immediately shown on the map and every unit will treat the new water area as any other water area on the map.



F 591 Virginio Fasan [FREMM ASW]

Regular

F 591 Virginio Fasan [FREMM ASW]
FFG - Guided Missile Frigate

Side: 1

Course: 0 deg

MAX Speed reduced due to Sea State 9
Speed: 0 kts (Full Stop)

Damage: 0.0%

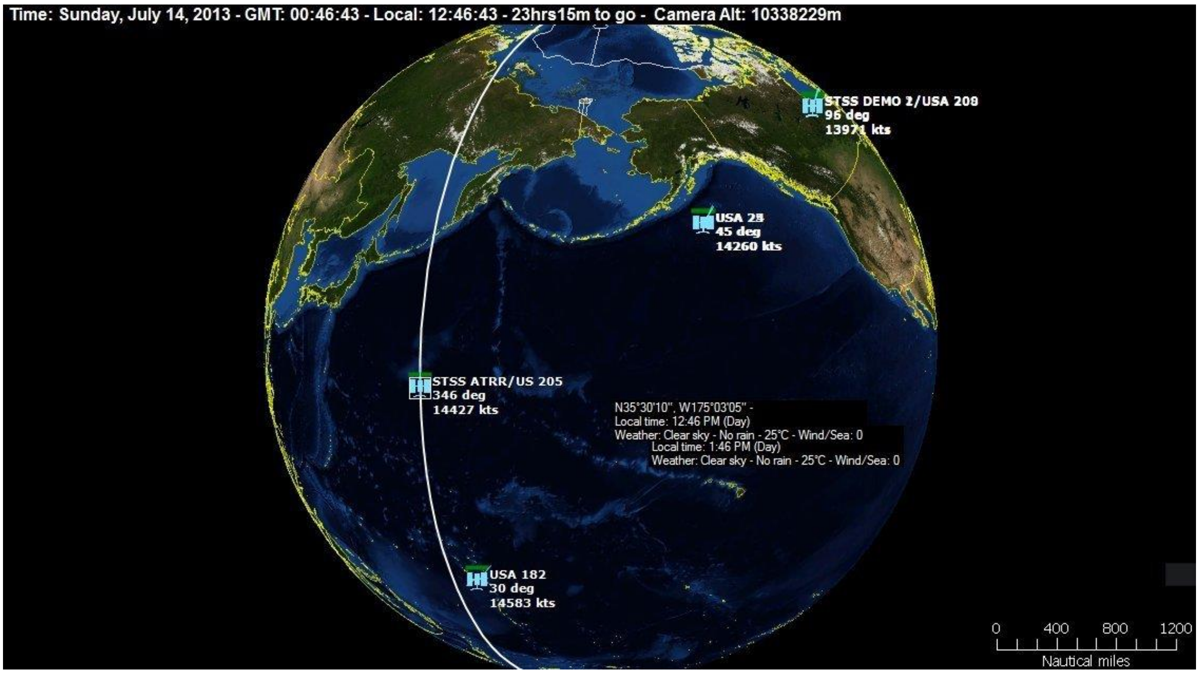
Supplies : Fuel and Ammo

Assigned base: None
Status: Unassigned (Underway)
0 Time Underway

UNIT WEAPONS

16x Aster 15 PAAMS [GWS 45 Sea Viper]
4x Milas
4x Otomat Mk2 Mod IV
230x 20mm/80 Oerlikon Burst [20 rnds]
6x CONTRALTO-V [SLAT]
90x SCLAR-H Chaff
24x SCLAR-H Flare
600x 76mm/62 Super Rapido DART Burst [
22x MU-90 Impact

Appendix 6: CPE Space Operations – Present and Future


Time: Sunday, July 14, 2013 - GMT: 00:46:43 - Local: 12:46:43 - 23hrs15m to go - Camera Alt: 10338229m
STSS DEMO 2/USA 208
96 deg
13971 kts
USA 25
45 deg
14260 kts
STSS ATRR/US 205
346 deg
14427 kts
N35°30'10", W175°03'05" -
Local time: 12:46 PM (Day)
Weather: Clear sky - No rain - 25°C - Wind/Sea: 0
Local time: 1:46 PM (Day)
Weather: Clear sky - No rain - 25°C - Wind/Sea: 0
USA 182
30 deg
14583 kts
0
400
800
1200
Nautical miles

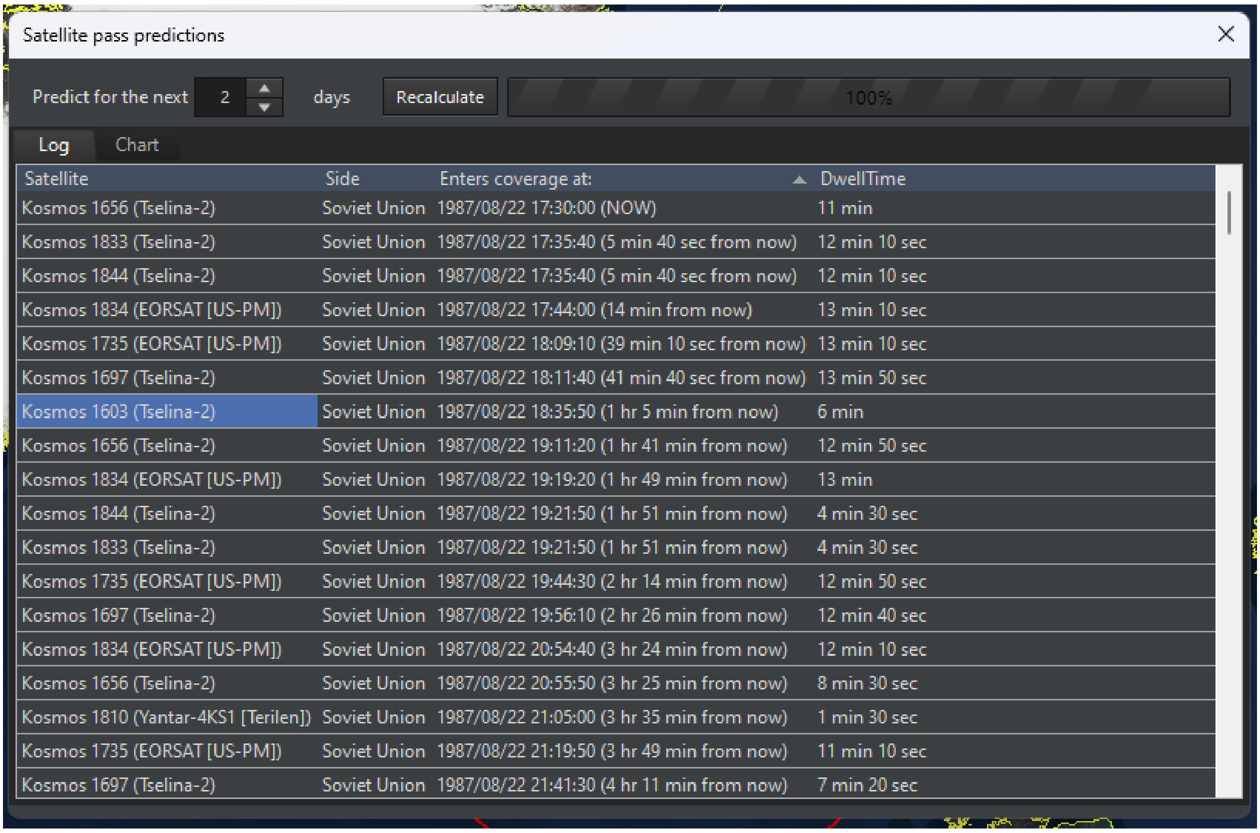
Command includes an elaborate series of near-space assets with a particular emphasis on intelligence satellites, both strictly military and “civilian” or dual-role.

Satellites are currently modelled as a search / surveillance /reconnaissance asset, capable of detecting & tracking targets as well as performing follow-up surveillance (e.g. air-ops activity on airfields) and battle-damage assessment (BDA) on existing contacts. Broad area scans as well as close-look capability are modelled by having extra sensors that cannot perform search but have high zoom ability (ie. observe in detail targets previously located by other means). Sensor swath, altitude and slant-range limitations are all taken into account.

Users can place satellite platforms in true-to-life orbits using provided TLE data, either from the DB or through scripting. The orbital characteristics can also by dynamically updated through scripting, so it's possible to "mirror" sat movements in Command to match external-provided real-time data.

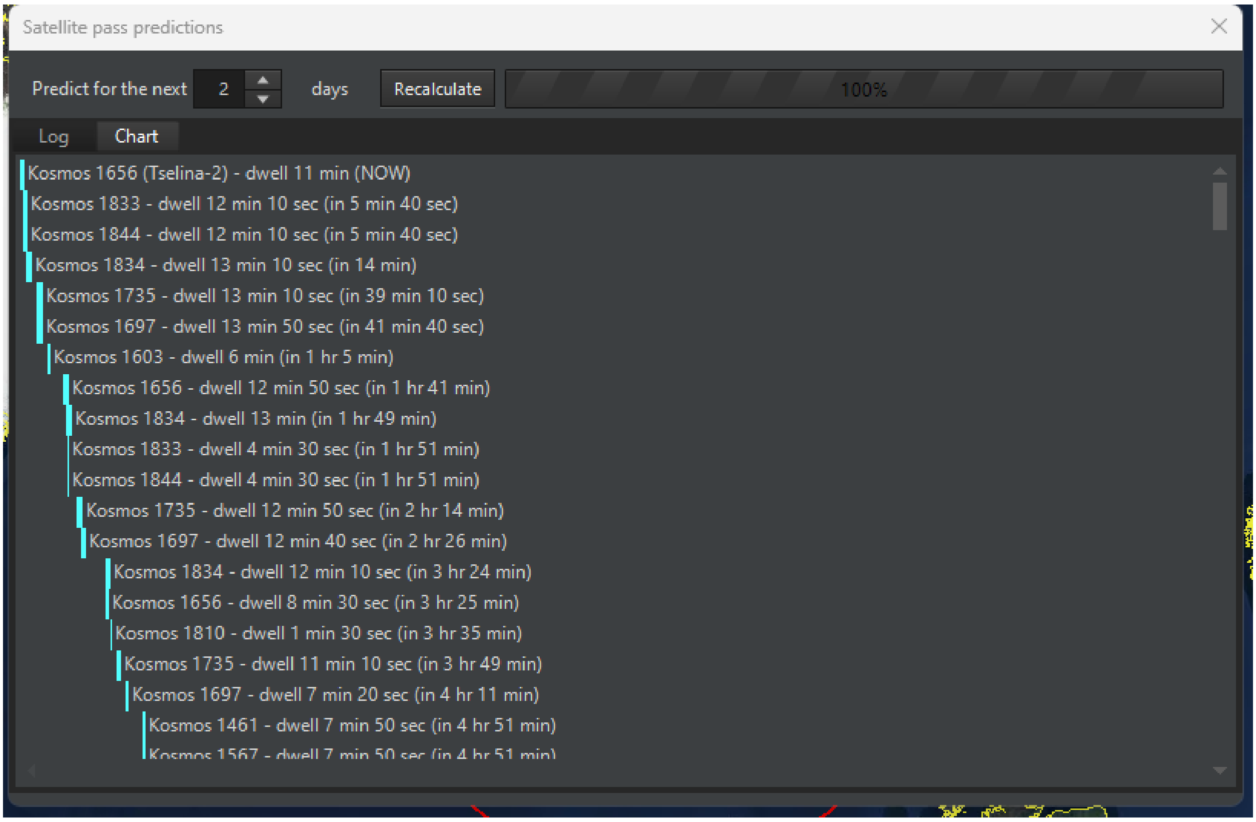
Satellites can be engaged and negated with ASATs (direct-ascent only, both surface and air-launched) and high-energy lasers. Such engagements are evaluated as one-shot kills; there is currently no gradual damage model for satellites as exists for other platform types.

Command provides a comprehensive UI for predicting future passes and dwell times over area-of-interest. This allows effective coordination with other assets (for example, to provide pre-strike reconnaissance or post-strike BDA). This is an example of the information in tabular format:



Satellite	Side	Enters coverage at:	DwellTime
Kosmos 1656 (Tselina-2)	Soviet Union	1987/08/22 17:30:00 (NOW)	11 min
Kosmos 1833 (Tselina-2)	Soviet Union	1987/08/22 17:35:40 (5 min 40 sec from now)	12 min 10 sec
Kosmos 1844 (Tselina-2)	Soviet Union	1987/08/22 17:35:40 (5 min 40 sec from now)	12 min 10 sec
Kosmos 1834 (EORSAT [US-PM])	Soviet Union	1987/08/22 17:44:00 (14 min from now)	13 min 10 sec
Kosmos 1735 (EORSAT [US-PM])	Soviet Union	1987/08/22 18:09:10 (39 min 10 sec from now)	13 min 10 sec
Kosmos 1697 (Tselina-2)	Soviet Union	1987/08/22 18:11:40 (41 min 40 sec from now)	13 min 50 sec
Kosmos 1603 (Tselina-2)	Soviet Union	1987/08/22 18:35:50 (1 hr 5 min from now)	6 min
Kosmos 1656 (Tselina-2)	Soviet Union	1987/08/22 19:11:20 (1 hr 41 min from now)	12 min 50 sec
Kosmos 1834 (EORSAT [US-PM])	Soviet Union	1987/08/22 19:19:20 (1 hr 49 min from now)	13 min
Kosmos 1844 (Tselina-2)	Soviet Union	1987/08/22 19:21:50 (1 hr 51 min from now)	4 min 30 sec
Kosmos 1833 (Tselina-2)	Soviet Union	1987/08/22 19:21:50 (1 hr 51 min from now)	4 min 30 sec
Kosmos 1735 (EORSAT [US-PM])	Soviet Union	1987/08/22 19:44:30 (2 hr 14 min from now)	12 min 50 sec
Kosmos 1697 (Tselina-2)	Soviet Union	1987/08/22 19:56:10 (2 hr 26 min from now)	12 min 40 sec
Kosmos 1834 (EORSAT [US-PM])	Soviet Union	1987/08/22 20:54:40 (3 hr 24 min from now)	12 min 10 sec
Kosmos 1656 (Tselina-2)	Soviet Union	1987/08/22 20:55:50 (3 hr 25 min from now)	8 min 30 sec
Kosmos 1810 (Yantar-4KS1 [Terilen])	Soviet Union	1987/08/22 21:05:00 (3 hr 35 min from now)	1 min 30 sec
Kosmos 1735 (EORSAT [US-PM])	Soviet Union	1987/08/22 21:19:50 (3 hr 49 min from now)	11 min 10 sec
Kosmos 1697 (Tselina-2)	Soviet Union	1987/08/22 21:41:30 (4 hr 11 min from now)	7 min 20 sec

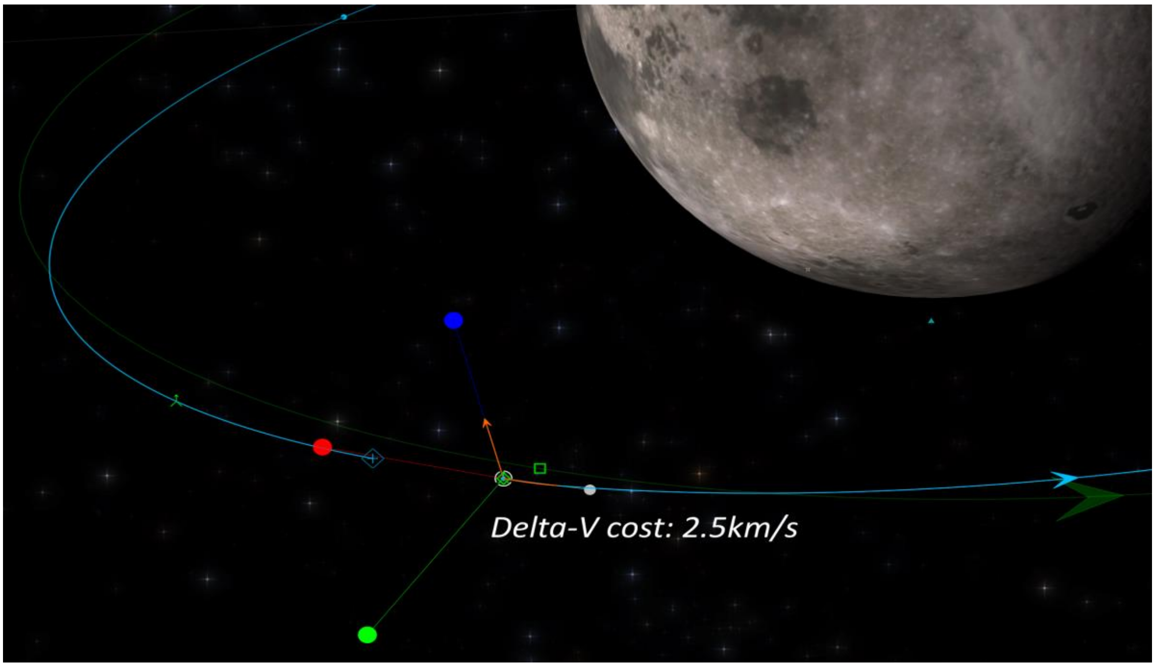
The very same information can also be presented in graphical/chart form, to allow quicker understanding of the future windows of availability:



Kosmos 1656 (Tselina-2) - dwell 11 min (NOW)
Kosmos 1833 - dwell 12 min 10 sec (in 5 min 40 sec)
Kosmos 1844 - dwell 12 min 10 sec (in 5 min 40 sec)
Kosmos 1834 - dwell 13 min 10 sec (in 14 min)
Kosmos 1735 - dwell 13 min 10 sec (in 39 min 10 sec)
Kosmos 1697 - dwell 13 min 50 sec (in 41 min 40 sec)
Kosmos 1603 - dwell 6 min (in 1 hr 5 min)
Kosmos 1656 - dwell 12 min 50 sec (in 1 hr 41 min)
Kosmos 1834 - dwell 13 min (in 1 hr 49 min)
Kosmos 1833 - dwell 4 min 30 sec (in 1 hr 51 min)
Kosmos 1844 - dwell 4 min 30 sec (in 1 hr 51 min)
Kosmos 1735 - dwell 12 min 50 sec (in 2 hr 14 min)
Kosmos 1697 - dwell 12 min 40 sec (in 2 hr 26 min)
Kosmos 1834 - dwell 12 min 10 sec (in 3 hr 24 min)
Kosmos 1656 - dwell 8 min 30 sec (in 3 hr 25 min)
Kosmos 1810 - dwell 1 min 30 sec (in 3 hr 35 min)
Kosmos 1735 - dwell 11 min 10 sec (in 3 hr 49 min)
Kosmos 1697 - dwell 7 min 20 sec (in 4 hr 11 min)
Kosmos 1461 - dwell 7 min 50 sec (in 4 hr 51 min)
Kosmos 1567 - dwell 7 min 50 sec (in 4 hr 51 min)

While the current space ops model obviously offers a lot of functionality, there is a diverse list of additional capabilities that are available as future growth options. Such potential additions include:

- Sat-to-sat and sat-to-earth engagements using energy weapons (lasers / EMPs / HEMW etc.)
- Improved damage model (similar to extended aircraft damage model)
- Integration with other existing space-oriented software (e.g. AGI STK)
- Sat-to-sat engagements with kinetic weaponry (the weapons themselves essentially become short-lived satellites)
- Space-to-earth engagements (e.g. orbital bombardment)
- Co-orbital ASATs
- Implementation of Kalman filtering on uncertain detections/contacts (e.g. SIGINT) to more faithfully model generated "uncertainty area" of each contact
- Better modelling of "looking through soda-straw" FOV limitations (ie. cannot scan a wide area if the cameras are close-focused on single target) and impact of spacecraft maneuverability/point-ability on number of targets visited per pass segment. (Currently a sat in Command can close-inspect all targets under its swatch in a single-pass, which is not feasible in real life ops)
- Delta-V budgeting for altering orbital parameters *("We can shift that KH-11 to pass sooner and at lower altitude over area-X, but it will cost 12% of its available fuel. Worth it?")*



- A suitable user interface for per-pass intel collection tasking integrated/coordinated with other intel assets (air/sea/land/SOF units etc.)

- A substantially more space ops-centric user interface and main map (using Command's existing sim-core but with UI elements oriented to the needs & priorities of the space ops community).
  One such representative UI sample is shown: