CSave: Installation, Configuration, and Operation

HS Software

1. Installation - Unix, Linux, BSD, and Windows (Cygwin)

The first thing you need to do is set up your build environment - install protobuf, gcc, pkg-config, make, and git. After you've installed everything, retrieve the source code for protobuf-c by executing "git clone https://github.com/protobuf-c/protobuf-c.git". Next, enter the source code directory (cd protobuf-c) and execute "./autogen.sh", "./configure", "make -j4", and "make install". If the shell script isn't executable, do a "chmod +x autogen.sh" before attempting to compile. Once you have successfully installed protobuf-c, return to your home directory - execute "cd" - and download the CSave source code with "git clone https://github.com/HackerSmacker/CSave.git". Enter the CSave source code directory with "cd CSave" and compile it with "make". Continue to the Operations section.

2. Installation - Windows (Native)

You need to compile libprotobuf-c with MINGW64 and use the headers/libraries/executables from that and put them into the Lib, Include, and Bin folders, respectively, of your VSC installation.

3. Installation - OS/2 (Watcom C/C++ 11)

Download the source code and transfer it to your OS/2 machine. Please note that you will have to cross-compile libprotobuf-c with GCC targeting OS/2. Transfer libprotobuf-c.lib and the header files over to OS/2. You may need to compile the C protobuf files with GCC, the rest will compile cleanly with Watcom.

4. Installation - OS/2 (EMX/ArcaOS Packages)

Follow the Unix installation instructions written above. The same process should apply. You might need to edit the Makefile to

5. Installation - OpenVMS 8.4 (DEC C/C++)

CSave does not cleanly compile with the DEC C compiler. Use a cross compiler or use GNV. Make sure your GCC version in GNV is 8.0.0 or newer, as the protobuf-c outputs will not compile.

6. Operation

6.1. Converting saves

You must convert saves from the "binary format" (AKA .sav) to "protobuf" format (the raw data) before you can actually use them. To acomplish this, do: "SaveToProto input.sav output.proto". Warning: this tool is currently not finished. Use Apocalyptech's bl3-cli-saveedit: "python -m bl3save.cli_edit -o protobuf in.sav out.proto". I will finish SaveToProto soon.

6.2. Dumping save information

6.2.1. Getting XP information

To determine what level the save file is at, run this command: "SaveUnpack in.proto | grep SKL | head -n 3". The XP value will coorespond to the level. Examine CommonVars.h to figure out the base XP required to be at that level.

6.2.2. Getting Mayhem Mode information

The editor can display Mayhem Mode stats with "SaveUnpack in.sav | grep MHM". Playthrough 0 is NVHM and playthrough 1 is TVHM. Of course, you could always add another playthrough and set it active, because UVHM can't come soon enough.

6.2.3. Getting name, GUID, ID, slot number, and TVHM status

This info appears at the top of SaveUnpack. Specifically, you can run "SaveUnpack in.sav | grep GEN" to pull the info off of the top.

6.2.4. Getting SDU info

"SaveUnpack in.sav | grep SDU"

6.3. Editing saves

6.4. Comprehensive List of SaveGenerate commands

6.4.1. quit, exit

Exit from SaveGenerate. The output file will be generated and saved. Please now run ProtoToSave to pack the save file into a BL3 binary format save.

6.4.2. set name

Set the player's preferred name. It will prompt for a string.

6.4.3. set class

Sets the player's class. Takes an integer. 0 is Amara, 1 is FL4K, 2 is Moze, and 3 is Zane.

6.4.4. set sdu

Sets SDU values. Iterates through each SDU and prompts for a new level. Press Enter to use the previous value, or specify a blank line if using a script file. If you are using a script file and you have too many blank lines, the editor will ignore them. If you don't have enough blank lines, the editor will set that SDU to zero.

6.4.5. set mayhemlevel

Set the Mayhem Mode level for any playthrough you want. It will first prompt for the Mayhem level you want, then it will prompt for what playthrough you want to update. Playthrough 0 is NVMH, and 1 is TVHM.

6.4.6. set expoints

Set the total amount of experience points. This does not mean set the level - setting the EXP level affects the level because you need a certain amount of EXP to clear that level. Prompts for an integer value.

6.4.7. set level

Sets the player level by getting the level, and setting the EXP to the minimum required to clear that level. Prompts for an integer. The max accepted value is 80, although this will not be accepted by the game, and will instead drop you down to the current level cap.

6.4.8. set guardianrank

This feature is currently not implemented. Check back later for an update.

6.4.9. set money

Set how much money you have. Takes an integer.

6.4.10. set eridium

Sets how much Eridium you have. Takes an integer.

6.4.11. unlock skilltree

Enable the selection of all skills on the tree. Does not coorespond to how many skill points you have.

6.4.12. set skillpoints

Sets how many skill points you have. Takes an integer - there does not appear to be a cap on this value.

6.5. Comprehensive list of SaveUnpack message prefixes

CSAV001GEN - General information
CSAV001CLS - Player class information
CSAV001SKL - Skill points, XP, skills, and tree information
CSAV001SDU - SDU information
CSAV001VEH - Vehicle parts, loadouts, and configurations
CSAV001MSN - Missions/quests
CSAV001AMO - Ammo and grenades
CSAV001GRD - Guardian rank, level, perks, and rewards
CSAV001ROM - Crew quarters/bedroom information (including guns on the rack)
CSAV001ECH - ECHO logs
CSAV001FTM - Fast Travel machines: blacklisted, active, and reachable
CSAV001INV - Inventory: backpack and equipped

CSAV001CUS - Customizations: color, skin, emotes
CSAV001CHL - Challenge information
CSAV001ICL - Money (ICL means Inventory Category List)
CSAV001MHM - Mayhem Mode information