This appears to be a specific Makefile for building the "nsrvproxy\_save" binary. Let's go through the important sections and their functionalities:

- The Makefile includes other Makefile fragments (`header.mk`, `nsr.mk`) and sets some build configurations.

- There are several variable assignments, such as `dir`, `pc\_CFLAGS\_$(ddir)`, `INCLUDEDIRS\_$(ddir)`, etc., that define various paths, compiler flags, and include directories used during the build process.

- The `Fnw\_vproxy\_EXES\_nsrvproxy\_save` variable specifies the name and path of the executable binary to be built (`nsrvproxy\_save$E`).

- The `nsrvproxy\_save\_CPPFILES` variable lists the C++ source files required to build the `nsrvproxy\_save` binary.

- The `nsrvproxy\_save\_OBJS` variable specifies the object files to be built from the C++ source files.

- The `nsrvproxy\_save\_SYSLIBS` and `nsrvproxy\_save\_pc\_SYSLIBS` variables define the system libraries that need to be linked with the binary.

- The `nsrvproxy\_save\_LIBS` variable lists various libraries that need to be linked with the binary.

- The `nsrvproxy\_save` target builds the `nsrvproxy\_save` binary using the specified object files and libraries.

- The `nsrvproxy\_save: nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save$E` phony target ensures that the `nsrvproxy\_save` target is always considered out of date and built when invoked.

This Makefile seems to be part of a larger project for building the "nsrvproxy\_save" binary. It includes various build configurations, compiler flags, and libraries necessary for successfully building the binary.

# CMakeLists.txt for nsrvproxy\_save binary

cmake\_minimum\_required(VERSION 3.0)

project(nsrvproxy\_save)

# Enable exception handling and runtime type information

set(CMAKE\_CXX\_FLAGS "${CMAKE\_CXX\_FLAGS} /GR /EHsc /wd4290")

# Enable Link-time Code Generation

set(CMAKE\_EXE\_LINKER\_FLAGS "${CMAKE\_EXE\_LINKER\_FLAGS} /LTCG")

# Set the include directories

set(INCLUDEDIRS\_nsrvproxy\_save

${BOOST\_INC\_PATH}

${INC\_ICONV}

${INC\_LIBACTION\_PATH}

${INC\_LIBJSONCODEC}

${INC\_LIBDDCL}

${INC\_LIBNSR}

${INC\_LIBRMQC}

${INC\_LIBUTIL}

${INC\_LIBNWNFS}

${INC\_LIBXML2}

${INC\_NSRVIM}

${INC\_LIB\_VSPHERE}

${INC\_NWP}

${INC\_RAP}

${LIBPOLICY\_INC\_PATH}

${LIBCURLX\_INC\_PATH}

${LIBJANSSON\_INC\_PATH}

${LIBNSRVPROXY\_INC\_PATH}

)

# Source files

set(nsrvproxy\_save\_CPPFILES

nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save.cpp

nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save\_action.cpp

nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save\_ctx.cpp

nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save\_job.cpp

nsr/vmware/vproxy/nsrvproxy\_save/nsrvproxy\_save\_vm.cpp

nsr/vmware/vproxy/nsrvproxy\_save/vcenter\_container.cpp

nsr/vmware/vproxy/nsrvproxy\_save/vcenter\_vm.cpp

nsr/vmware/vproxy/nsrvproxy\_save/json\_node.cpp

nsr/vmware/vproxy/nsrvproxy\_save/json\_object.cpp

nsr/vmware/vproxy/nsrvproxy\_save/json\_array.cpp

)

# Add executable target

add\_executable(nsrvproxy\_save ${nsrvproxy\_save\_CPPFILES})

# Link libraries

target\_link\_libraries(nsrvproxy\_save

${LIBCURLX\_LIB}

${LIB\_COMMONSSL\_SYSLIBS}

ws2\_32.lib

msxml2.lib

advapi32.lib

Mpr.lib

netapi32.lib

rpcrt4.lib

WbemUuid.lib

)

# Set output name and output directory

set\_target\_properties(nsrvproxy\_save PROPERTIES

OUTPUT\_NAME "nsrvproxy\_save"

RUNTIME\_OUTPUT\_DIRECTORY "${CMAKE\_BINARY\_DIR}/bin/nsr/vmware/vproxy/nsrvproxy\_save"

)

# Install target

install(TARGETS nsrvproxy\_save

DESTINATION "nsr/vmware/vproxy/nsrvproxy\_save"

)