

Coova-Chilli_1.3.0_20141128_HHLB

The default Coova-Chilli on [LEDE 17.01](#) is the [1.3.0+20141128](#).

This version doesn't expose all of its compile options.

Layout

Under the *feeds/packages/net/coova-chilli* directory is the [Makefile](#), [Config.in](#), [patches](#) and [files](#) directories specific to your OpenWRT/LEDE coova-chilli build target platform.

This directory is also referenced as a symbolic link in *package/feeds/packages/coova-chilli*.

Makefile

Open the *package/feeds/packages/coova-chilli/Makefile*:

```
#
# Copyright (C) 2007-2014 OpenWrt.org
#
# This is free software, licensed under the GNU General Public License v2.
# See /LICENSE for more information.
#

include $(TOPDIR)/rules.mk

PKG_NAME:=coova-chilli
PKG_VERSION:=1.3.0+20141128
PKG_MAINTAINER:=Imre Kaloz <kaloz@openwrt.org>
PKG_LICENSE:=GPL-2.0+
PKG_LICENSE_FILES:=COPYING
PKG_RELEASE:=4

PKG_SOURCE_PROTO:=git
PKG_SOURCE_URL:=git://github.com/coova/coova-chilli
PKG_SOURCE_SUBDIR:=$(PKG_NAME)-$(PKG_VERSION)
PKG_SOURCE_VERSION:=b93de20a288c01c2ba28e96e31ad6da01627f45f
PKG_SOURCE:=$(PKG_NAME)-$(PKG_VERSION).tar.gz
PKG_MD5SUM:=2adb27ec56172b18c5beee359dd7898d

PKG_INSTALL:=1

PKG_CONFIG_DEPENDS := \
    COOVACHILLI_MINIPORTAL \
    COOVACHILLI_REDIR \
    COOVACHILLI_USERAGENT \
    COOVACHILLI_DNSLOG \
    COOVACHILLI_UAMDOMAINFILE \
    COOVACHILLI_LARGE_LIMITS \
    COOVACHILLI_NOSSL \
    COOVACHILLI_MATRIXSSL \
    COOVACHILLI_CYASSL \
    COOVACHILLI_OPENSSL

include $(INCLUDE_DIR)/package.mk
include $(INCLUDE_DIR)/kernel.mk

define Package/coova-chilli
    SUBMENU:=Captive Portals
    SECTION:=net
    CATEGORY:=Network
    DEPENDS:=+kmod-tun +librt +COOVACHILLI_MATRIXSSL:libmatrixssl +COOVACHILLI_CYASSL:libcyassl
+COOVACHILLI_OPENSSL:libopenssl
    TITLE:=Wireless LAN HotSpot controller (Coova Chilli Version)
    URL:=http://www.coova.org/CoovaChilli
    MENU:=1
endef

define Package/coova-chilli/description
```

CoovaChilli is an open source access controller for wireless LAN access points and is based on ChilliSpot. It is used for authenticating users of a wireless (or wired) LAN. It supports web based login (UAM) which is today's standard for public HotSpots and it supports Wireless Protected Access (WPA) which is the standard of the future. Authentication, authorization and accounting (AAA) is handled by your favorite radius server.

endif

```
define Package/coova-chilli/config
    source "$(SOURCE)/Config.in"
endif
```

```
define KernelPackage/ipt-coova
    URL:=http://www.coova.org/CoovaChilli
    SUBMENU:=Netfilter Extensions
    DEPENDS:=coova-chilli +kmod-ipt-core +libxtables
    TITLE:=Coova netfilter module
    FILES:=$(PKG_BUILD_DIR)/src/linux/xt_*.${LINUX_KMOD_SUFFIX}
    AUTOLOAD:=$(call AutoProbe,xt_coova)
endif
```

```
define KernelPackage/ipt-coova/description
    Netfilter kernel module for CoovaChilli
    Includes:
    - coova
endif
```

DISABLE_NLS=

TARGET_CFLAGS += \$(FPIC)

```
CONFIGURE_VARS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)"
```

```
MAKE_FLAGS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)"
```

```
MAKE_INSTALL_FLAGS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)" \
    INSTALL_MOD_PATH="$(PKG_INSTALL_DIR)"
```

```
define Build/Prepare
$(call Build/Prepare/Default)
    ( cd $(PKG_BUILD_DIR) ; \
        [ -f ./configure ] || { \
            ./bootstrap ; \
        } \
    )
endif
```

```
define Build/Configure
$(call Build/Configure/Default, \
    $(if $(CONFIG_COOVACHILLI_PROXY),--enable,--disable)-chilliproxy \
    $(if $(CONFIG_COOVACHILLI_REDIR),--enable,--disable)-chilliredir \
    $(if $(CONFIG_COOVACHILLI_DNSLOG),--enable,--disable)-dnslog \
    $(if $(CONFIG_COOVACHILLI_MINIPORTAL),--enable,--disable)-miniportal \
    $(if $(CONFIG_COOVACHILLI_USERAGENT),--enable,--disable)-useragent \
    $(if $(CONFIG_COOVACHILLI_LARGE_LIMITS),--enable,--disable)-large_limits \
    $(if $(CONFIG_COOVACHILLI_UAMDOMAINFILE),--enable,--disable)-uamdomainfile \
    $(if $(CONFIG_COOVACHILLI_MATRIXSSL),--with,--without)-matrixssl \
    $(if $(CONFIG_COOVACHILLI_CYSSL),--with,--without)-cyassl \
    $(if $(CONFIG_COOVACHILLI_OPENSSL),--with,--without)-openssl \
    $(if $(CONFIG_PACKAGE_kmod-ipt-coova),--with-nfcoova) \
)
endif
```

```
define Package/coova-chilli/conffiles
```

```

/etc/config/chilli
endif

define Package/coova-chilli/install
    $(INSTALL_DIR) $(1)/etc
    $(INSTALL_CONF) $(PKG_INSTALL_DIR)/etc/chilli.conf $(1)/etc/
    $(INSTALL_DIR) $(1)/etc/chilli
    $(CP) $(PKG_INSTALL_DIR)/etc/chilli/* $(1)/etc/chilli/
    $(INSTALL_DIR) $(1)/etc/hotplug.d/iface
    $(INSTALL_DATA) ./files/chilli.hotplug $(1)/etc/hotplug.d/iface/30-chilli
    $(INSTALL_DIR) $(1)/usr/sbin
    $(INSTALL_BIN) $(PKG_INSTALL_DIR)/usr/sbin/chilli* $(1)/usr/sbin/
    $(INSTALL_DIR) $(1)/usr/lib/
    $(CP) $(PKG_INSTALL_DIR)/usr/lib/lib*.so.* $(1)/usr/lib/
    $(if $(CONFIG_PACKAGE_kmod-ipt-coova), \
        $(INSTALL_DIR) $(1)/usr/lib/iptables; \
        $(CP) $(PKG_INSTALL_DIR)/usr/lib/iptables/lib*.so $(1)/usr/lib/iptables/ \
    )
    $(INSTALL_DIR) $(1)/etc/init.d
    $(INSTALL_BIN) files/chilli.init $(1)/etc/init.d/chilli
    $(INSTALL_DIR) $(1)/etc/config
    $(INSTALL_DATA) files/chilli.config $(1)/etc/config/chilli
    $(INSTALL_DIR) $(1)/lib/firewall
    $(CP) files/chilli.firewall $(1)/lib/firewall/chilli.sh
endif

$(eval $(call BuildPackage,coova-chilli))
$(eval $(call KernelPackage,ipt-coova))

```

Following the compile definitions from the file **coova-chilli-1.3.0+20141128/configure.in**:

```

#                                     -*- Autoconf -*-
# Process this file with autoconf to produce a configure script.

AC_PREREQ(2.57)
AC_INIT(coova-chilli, 1.3.1-svn, support@coova.com)
AC_CONFIG_SRCDIR([src/chilli.c])

AM_INIT_AUTOMAKE

AM_CONFIG_HEADER([config.h])

AC_CONFIG_MACRO_DIR([m4])

# Checks for programs.
AC_PROG_CC

#JJAKO Check for libtool
AC_PROG_LIBTOOL
AM_PROG_LIBTOOL
AC_ARG_PROGRAM

# Checks for libraries.

if test -r /dev/bpf -o -h /dev/bpf ; then
    V_DEV=bpf
    AC_DEFINE(HAVE_CLONING_BPF,1,[define if you have a cloning BPF device])
elif test -r /dev/bpf0 ; then
    V_DEV=bpf
elif test -r /usr/include/net/pfild.h ; then
    V_DEV=pf
elif test -r /dev/enet ; then
    V_DEV=enet
elif test -r /dev/nit ; then
    V_DEV=snit
elif test -r /usr/include/sys/net/nit.h ; then
    V_DEV=nit
elif test -r /usr/include/linux/socket.h ; then
    V_DEV=linux

```

```

elif test -r /usr/include/net/raw.h ; then
    V_DEV=snoop
elif test -r /usr/include/odmi.h ; then
    V_DEV=bpff
elif test -c /dev/bpff0 ; then # check again in case not readable
    V_DEV=bpff
elif test -r /usr/include/sys/dlpi.h ; then
    V_DEV=dlpi
elif test -c /dev/enet ; then # check again in case not readable
    V_DEV=enet
elif test -c /dev/nit ; then # check again in case not readable
    V_DEV=snit
else
    V_DEV=null
fi
AC_MSG_RESULT($V_DEV)

# Do capture-mechanism-dependent tests.
case "$V_DEV" in
linux)
    AC_LBL_TPACKET_STATS
    AC_LBL_LINUX_TPACKET_AUXDATA_TP_VLAN_TCI
    ;;
esac

# Checks for header files.
AC_HEADER_STDC
AC_CHECK_HEADERS([arpa/inet.h errno.h fcntl.h dirent.h \
    netdb.h netinet/in.h netinet/tcp.h \
    stdint.h stddef.h stdlib.h string.h \
    sys/ioctl.h sys/socket.h linux/sysinfo.h sys/sysinfo.h \
    sys/param.h sys/time.h time.h \
    sys/ipc.h sys/msg.h signal.h \
    sys/wait.h sys/un.h ifaddrs.h \
    sys/stat.h sys/types.h regex.h \
    syslog.h poll.h sys/epoll.h \
    unistd.h endian.h libgen.h \
    asm/types.h pwd.h grp.h ])

AC_CHECK_HEADERS([resolv.h net/route.h net/if.h net/if_arp.h net/if_tun.h net/ethernet.h], [], [],
[
#include <stdio.h>
#ifdef STDC_HEADERS
# include <stdlib.h>
# include <stddef.h>
#else
# ifdef HAVE_STDLIB_H
# include <stdlib.h>
# endif
#endif
#ifdef HAVE_SYS_TYPES_H
# include <sys/types.h>
#endif
#ifdef HAVE_SYS_SOCKET_H
# include <sys/socket.h>
#endif
#ifdef HAVE_NETINET_IN_H
# include <netinet/in.h>
#endif
])

# Checks for typedefs, structures, and compiler characteristics.
AC_C_CONST
AC_TYPE_MODE_T
AC_HEADER_TIME

# Checks for library functions.
AC_FUNC_FORK
AC_PROG_GCC_TRADITIONAL
AC_FUNC_MEMCMP
AC_FUNC_SELECT_ARGTYPES

```

```

AC_CHECK_FUNCS([gethostbyname getprotoent gettimeofday inet_ntoa \
memset select socket strdup strerror strstr strtol getline dirname \
glob getaddrinfo getnameinfo getifaddrs sysinfo])
AC_CHECK_LIB(resolv, res_init)

AC_ARG_ENABLE(chilliquery, [AC_HELP_STRING([--disable-chilliquery],[Disable chilli_query]),
    enable_chilliquery=$enableval, enable_chilliquery=yes)

if test x"$enable_chilliquery" = xyes; then
    AC_DEFINE(ENABLE_CHILLIQUERY,1,[Define to enable chilli_query])
fi

AM_CONDITIONAL(WITH_CHILLIQUERY, [test x"$enable_chilliquery" != xno])

AC_ARG_ENABLE(leakybucket, [AC_HELP_STRING([--disable-leakybucket],[disable use of leaky bucket shaping]),
    enable_leakybucket=$enableval, enable_leakybucket=yes)

if test x"$enable_leakybucket" = xyes; then
    AC_DEFINE(ENABLE_LEAKYBUCKET,1,[Define to enable Chilli Leaky Bucket shaping])
fi

AC_ARG_ENABLE(uamanyip, [AC_HELP_STRING([--disable-uamanyip],[disable use of uamanyip]),
    enable_uamanyip=$enableval, enable_uamanyip=yes)

if test x"$enable_uamanyip" = xyes; then
    AC_DEFINE(ENABLE_UAMANYIP,1,[Define to enable uamanyip])
fi

AC_ARG_ENABLE(uamuiport, [AC_HELP_STRING([--disable-uamuiport],[disable use of uamuiport]),
    enable_uamuiport=$enableval, enable_uamuiport=yes)

if test x"$enable_uamuiport" = xyes; then
    AC_DEFINE(ENABLE_UAMUIPORT,1,[Define to enable uamuiport])
fi

AC_ARG_ENABLE(accounting-onoff, [AC_HELP_STRING([--disable-accounting-onoff],[disable use of Accounting-On and Accounting-Off]),
    enable_accounting_onoff=$enableval, enable_accounting_onoff=yes)

if test x"$enable_accounting_onoff" = xyes; then
    AC_DEFINE(ENABLE_ACCOUNTING_ONOFF,1,[Define to enable Accounting-On and Accounting-Off])
fi

AC_ARG_ENABLE(tap, [AC_HELP_STRING([--disable-tap],[Disable support for tap interface (tun only)]),
    enable_tap=$enableval, enable_tap=yes)

if test x"$enable_tap" = xyes; then
    AC_DEFINE(ENABLE_TAP,1,[Define to enable Chilli tap support])
fi

AC_ARG_ENABLE(tcpreset, [AC_HELP_STRING([--disable-tcpreset],[Disable support for TCP reset of filtered connections]),
    enable_tcpreset=$enableval, enable_tcpreset=yes)

if test x"$enable_tcpreset" = xyes; then
    AC_DEFINE(ENABLE_TCPRESET,1,[Define to enable TCP reset support])
fi

AC_ARG_ENABLE(radproxy, [AC_HELP_STRING([--disable-radproxy],[Disable support RADIUS (EAP) Proxy]),
    enable_radproxy=$enableval, enable_radproxy=yes)

if test x"$enable_radproxy" = xyes; then
    AC_DEFINE(ENABLE_RADPROXY,1,[Define to enable Chilli RADIUS (EAP) Proxy support])
fi

AC_ARG_ENABLE(json, [AC_HELP_STRING([--disable-json],[Disable support for JSON]),
    enable_json=$enableval, enable_json=yes)

if test x"$enable_json" = xyes; then
    AC_DEFINE(ENABLE_JSON,1,[Define to enable Chilli JSON])
fi

```

```

AC_ARG_ENABLE(debug, [AC_HELP_STRING([--disable-debug],[Disable debugging messages]]),
    enable_debug=$enableval, enable_debug=yes)

if test x"$enable_debug" = xyes; then
    AC_DEFINE(ENABLE_DEBUG,1,[Define to enable debugging])
fi

AC_ARG_ENABLE(dhcpradius, [AC_HELP_STRING([--disable-dhcpradius],[Disable support DHCP/RADIUS integration]]),
    enable_dhcpradius=$enableval, enable_dhcpradius=yes)

if test x"$enable_dhcpradius" = xyes; then
    AC_DEFINE(ENABLE_DHCPRADIUS,1,[Define to enable DHCP/RADIUS integration])
fi

AC_ARG_ENABLE(wpad, [AC_HELP_STRING([--enable-wpad],[Enable support WPAD]]),
    enable_wpad=$enableval, enable_wpad=no)

if test x"$enable_wpad" = xyes; then
    AC_DEFINE(ENABLE_WPAD,1,[Define to enable WPAD])
fi

AC_ARG_ENABLE(gardenaccounting, [AC_HELP_STRING([--enable-gardenaccounting],[Enable walled garden
accounting]]),
    enable_gardenaccounting=$enableval, enable_gardenaccounting=no)

if test x"$enable_gardenaccounting" = xyes; then
    AC_DEFINE(ENABLE_GARDENACCOUNTING,1,[Define to enable walled garden accounting])
fi

AC_ARG_ENABLE(gardenext, [AC_HELP_STRING([--enable-gardenext],[Enable extended walled garden features]]),
    enable_gardenext=$enableval, enable_gardenext=no)

if test x"$enable_gardenext" = xyes; then
    AC_DEFINE(ENABLE_GARDENEXT,1,[Define to enable extended walled garden features])
fi

AC_ARG_ENABLE(inspect, [AC_HELP_STRING([--enable-inspect],[Enable inspect feature in cmdsock]]),
    enable_inspect=$enableval, enable_inspect=no)

if test x"$enable_inspect" = xyes; then
    AC_DEFINE(ENABLE_INSPECT,1,[Define to enable inspect feature in cmdsock])
fi

AC_ARG_ENABLE(coa, [AC_HELP_STRING([--disable-coa],[Disable CoA RADIUS support]]),
    enable_coa=$enableval, enable_coa=yes)

if test x"$enable_coa" = xyes; then
    AC_DEFINE(ENABLE_COA,1,[Define for CoA RADIUS support])
fi

AC_ARG_ENABLE(dhcpt, [AC_HELP_STRING([--enable-dhcpt],[Enable support for DHCP option setting]]),
    enable_dhcpt=$enableval, enable_dhcpt=no)

if test x"$enable_dhcpt" = xyes; then
    AC_DEFINE(ENABLE_DHCPT,1,[Define to enable DHCP option setting])
fi

AC_ARG_ENABLE(debug2, [AC_HELP_STRING([--enable-debug2],[Enable verbose debugging]]),
    enable_debug2=$enableval, enable_debug2=no)

if test x"$enable_debug2" = xyes; then
    AC_DEFINE(ENABLE_DEBUG2,1,[Define to enable verbose debugging])
fi

AC_ARG_ENABLE(sessgarden, [AC_HELP_STRING([--enable-sessgarden],[Enable support for session-based walled
garden]]),
    enable_sessgarden=$enableval, enable_sessgarden=no)

if test x"$enable_sessgarden" = xyes; then

```

```

    AC_DEFINE(ENABLE_SESSGARDEN,1,[Define to enable Chilli session walled garden])
fi

AC_ARG_ENABLE(sessproxy, [AC_HELP_STRING([--enable-sessproxy],[Enable support for per session postauth proxy])],
    enable_sessproxy=$enableval, enable_sessproxy=no)

if test x"$enable_sessproxy" = xyes; then
    AC_DEFINE(ENABLE_SESSPROXY,1,[Define to enable per session postauth proxy])
fi

AC_ARG_ENABLE(sessdhcp, [AC_HELP_STRING([--enable-sessdhcp],[Enable support for per session DHCP relay])],
    enable_sessdhcp=$enableval, enable_sessdhcp=no)

if test x"$enable_sessdhcp" = xyes; then
    AC_DEFINE(ENABLE_SESSDHCP,1,[Define to enable per session DHCP relay])
fi

AC_ARG_ENABLE(sessdns, [AC_HELP_STRING([--enable-sessdns],[Enable support for per session DNS enforcement])],
    enable_sessdns=$enableval, enable_sessdns=no)

if test x"$enable_sessdns" = xyes; then
    AC_DEFINE(ENABLE_SESSDNS,1,[Define to enable per session DNS enforcement])
fi

AC_ARG_ENABLE(chillixml, [AC_HELP_STRING([--enable-chillixml],[Enable use of chillixml])],
    enable_chillixml=$enableval, enable_chillixml=no)

if test x"$enable_chillixml" = xyes; then
    AC_DEFINE(ENABLE_CHILLIXML,1,[Define to enable Chilli XML])
fi

AC_ARG_ENABLE(proxyvsa, [AC_HELP_STRING([--enable-proxyvsa],[Enable support for VSA attribute proxy])],
    enable_proxyvsa=$enableval, enable_proxyvsa=no)

if test x"$enable_proxyvsa" = xyes; then
    AC_DEFINE(ENABLE_PROXYVSA,1,[Define to enable VSA proxy])
fi

AC_ARG_ENABLE(dnslog, [AC_HELP_STRING([--enable-dnslog],[Enable support to log DNS name queries])],
    enable_dnslog=$enableval, enable_dnslog=no)

if test x"$enable_dnslog" = xyes; then
    AC_DEFINE(ENABLE_DNSLOG,1,[Define to logging of DNS requests])
fi

AC_ARG_ENABLE(ipwhitelist, [AC_HELP_STRING([--enable-ipwhitelist],[Enable file based IP white list])],
    enable_ipwhitelist=$enableval, enable_ipwhitelist=no)

if test x"$enable_ipwhitelist" = xyes; then
    AC_DEFINE(ENABLE_IPWHITELIST,1,[Define to support file based whitelists])
fi

AC_ARG_ENABLE(uamdomainfile, [AC_HELP_STRING([--enable-uamdomainfile],[Enable loading of mass uamdomains from file])],
    enable_uamdomainfile=$enableval, enable_uamdomainfile=no)

if test x"$enable_uamdomainfile" = xyes; then
    AC_DEFINE(ENABLE_UAMDOMAINFILE,1,[Define to support loading of uamdomains (with regex) from file])
fi

AC_ARG_ENABLE(redirdnsreq, [AC_HELP_STRING([--enable-redirdnsreq],[Enable the sending of a DNS query on redirect])],
    enable_redirdnsreq=$enableval, enable_redirdnsreq=no)

if test x"$enable_redirdnsreq" = xyes; then
    AC_DEFINE(ENABLE_REDIRDNSREQ,1,[Define to DNS query on redirect to pick up dynamic walled garden])
fi

AC_ARG_ENABLE(ieee8021q, [AC_HELP_STRING([--disable-ieee8021q],[disable support for IEEE 802.1Q])],
    enable_ieee8021q=$enableval, enable_ieee8021q=yes)

```

```

if test x"$enable_ieee8021q" = xyes; then
  AC_DEFINE(ENABLE_IEEE8021Q,1,[Define to enable Chilli IEEE 802.1Q])
fi

AC_ARG_ENABLE(largelimits, [AC_HELP_STRING([--enable-largelimits],[Enable larger limits for use with non-
embedded systems])],
  enable_largelimits=$enableval, enable_largelimits=no)

if test x"$enable_largelimits" = xyes; then
  AC_DEFINE(ENABLE_LARGELIMITS,1,[Enable larger limits for use with non-embedded systems])
fi

AC_ARG_WITH([openssl],
  [AS_HELP_STRING([--with-openssl], [enable support for openssl])],[],[with_openssl=no])

OPENSSL=
AS_IF([test x"$with_openssl" != xno],
  [AC_CHECK_LIB([crypto], [CRYPTO_malloc],
    [AC_SUBST([LIBSSL], ["-lssl -lcrypto"])
    AC_DEFINE([HAVE_OPENSSL], [1],
      [Define if you have openssl])
    ],
    [AC_MSG_FAILURE(
      [--with-openssl was given, but test for openssl failed])],
    [-lssl -lcrypto])])

AM_CONDITIONAL(WITH_OPENSSL, [test x"$with_openssl" != xno])

AC_ARG_WITH([matrixssl],
  [AS_HELP_STRING([--with-matrixssl], [enable support for matrixssl])],[],[with_matrixssl=no])

AS_IF([test x"$with_matrixssl" != xno],
  [AC_CHECK_LIB([matrixssl], [matrixSslOpen],
    [AC_SUBST([LIBSSL], ["-lmatrixssl"])
    AC_DEFINE([HAVE_MATRIXSSL], [1],
      [Define if you have matrixssl])
    ],
    [AC_MSG_FAILURE(
      [--with-matrixssl was given, but test for matrixssl failed])],
    [-lmatrixssl])])

AS_IF([test x"$with_matrixssl" != xno],
  [AC_CHECK_HEADERS([matrixSsl.h matrixSsl/matrixSsl.h])])

AM_CONDITIONAL(WITH_MATRIXSSL, [test x"$with_matrixssl" != xno])

AC_ARG_WITH([cyassl],
  [AS_HELP_STRING([--with-cyassl], [enable support for cyassl])],[],[with_cyassl=no])

AS_IF([test x"$with_cyassl" != xno],
  [AC_CHECK_LIB([cyassl], [CyaSSL_Init],
    [AC_SUBST([LIBSSL], ["-lcyassl"])
    AC_DEFINE([HAVE_CYASSL], [1],
      [Define if you have cyassl])
    ],
    [AC_MSG_FAILURE(
      [--with-cyassl was given, but test for cyassl failed])],
    [-lcyassl])])

AS_IF([test x"$with_cyassl" != xno],
  [AC_CHECK_HEADERS([cyassl/ssl.h ssh.h])])

AM_CONDITIONAL(WITH_CYASSL, [test x"$with_cyassl" != xno])

AM_CONDITIONAL(WITH_SSL, [test x"$with_openssl" != xno || test x"$with_matrixssl" != xno || test
x"$with_cyassl" != xno])

AC_ARG_WITH([matrixssl-cli],
  [AS_HELP_STRING([--with-matrixssl-cli], [enable matrixssl client use])],[],[with_matrixssl_cli=no])

```

```

AM_CONDITIONAL(WITH_MATRIXSSL_CLI, [test x"$with_matrixssl_cli" != xno])

AC_ARG_WITH([nfqueue],
  [AS_HELP_STRING([--with-nfqueue], [enable support for netfilter_queue])],[],[with_nfqueue=no])

AS_IF([test x"$with_nfqueue" != xno],
  [AC_CHECK_LIB([netfilter_queue], [nfq_open],
    [AC_SUBST([LIBNETFILTER_QUEUE], ["-lnetfilter_queue -lnfnetworklink"])
    AC_DEFINE([HAVE_NETFILTER_QUEUE], [1],
      [Define if you have netfilter_queue])
    ],
    [AC_MSG_FAILURE(
      [--with-nfqueue was given, but test for netfilter_queue failed])],
    [-lnetfilter_queue -lnfnetworklink])])

AM_CONDITIONAL(WITH_NETFILTER_QUEUE, [test x"$with_nfqueue" != xno])

AC_ARG_WITH([avl],
  [AS_HELP_STRING([--with-avl], [enable support for avl library])],[],[with_avl=no])

AS_IF([test x"$with_avl" != xno],
  [AC_DEFINE([HAVE_AVL], [1], [Define to use avl library])])

AM_CONDITIONAL(WITH_AVL, [test x"$with_avl" != xno])

AC_ARG_WITH([nfcoova],
  [AS_HELP_STRING([--with-nfcoova], [enable support for coova netfilter module])],[],[with_nfcoova=no])

AS_IF([test x"$with_nfcoova" != xno],
  [AC_DEFINE([HAVE_NETFILTER_COOVA], [1], [Define to use coova kernel module])])

AM_CONDITIONAL(WITH_NETFILTER_COOVA, [test x"$with_nfcoova" != xno])

AC_ARG_WITH([sfhash],
  [AS_HELP_STRING([--without-sfhash], [enable SuperFastHash use])],[],[with_sfhash=yes])

AS_IF([test x"$with_sfhash" != xno],
  [AC_DEFINE([HAVE_SFHASH], [1], [Define to use SuperFastHash])])

AM_CONDITIONAL(WITH_SFHASH, [test x"$with_sfhash" != xno])

AC_ARG_WITH([lookup3],
  [AS_HELP_STRING([--with-lookup3], [enable Jenkins lookup3 use])],[],[with_lookup3=no])

AS_IF([test x"$with_lookup3" != xno],
  [AC_DEFINE([HAVE_LOOKUP3], [1], [Define to use Jenkins lookup3])])

AM_CONDITIONAL(WITH_LOOKUP3, [test x"$with_lookup3" != xno])

AC_ARG_WITH([patricia],
  [AS_HELP_STRING([--with-patricia], [enable Patricia use])],[],[with_patricia=no])

AS_IF([test x"$with_patricia" != xno],
  [AC_DEFINE([HAVE_PATRICIA], [1], [Define to include Patricia])])

AM_CONDITIONAL(WITH_PATRICIA, [test x"$with_patricia" != xno])

AC_ARG_ENABLE([authedallowed],
  [AS_HELP_STRING([--enable-authedallowed], [enable Authorized Garden])],[],[with_authedallowed=no])

AS_IF([test x"$with_authedallowed" != xno],
  [AC_DEFINE([ENABLE_AUTHEDALLOWED], [1], [Define to include Authenticated Garden])])

AM_CONDITIONAL(WITH_AUTHEDALLOWED, [test x"$with_authedallowed" != xno])

AC_ARG_WITH([ipv6],
  [AS_HELP_STRING([--without-ipv6], [enable IPv6])],[],[with_ipv6=yes])

AS_IF([test x"$with_ipv6" != xno],

```

```

[AC_DEFINE([ENABLE_IPV6], [1], [Define to use IPv6]))

AM_CONDITIONAL(WITH_IPV6, [test x"$with_ipv6" != xno])

AC_ARG_WITH([pcap],
  [AS_HELP_STRING([--with-pcap], [enable support for pcap]),[],[with_pcap=no])

AS_IF([test x"$with_pcap" != xno],
  [AC_CHECK_LIB([pcap], [pcap_open_live],
    [AC_SUBST([LIBPCAP], ["-lpcap"])
    AC_DEFINE([USING_PCAP], [1],
      [Define if you have pcap enabled])
    ],
    [AC_MSG_FAILURE(
      [--with-pcap was given, but test for pcap failed]),
    [-lpcap])])

AM_CONDITIONAL(WITH_PCAP, [test x"$with_pcap" != xno])

AC_ARG_WITH([curl],
  [AS_HELP_STRING([--with-curl], [enable support for curl]),[],[with_curl=no])

AS_IF([test x"$with_curl" != xno],
  [AC_CHECK_LIB([curl], [curl_global_init],
    [AC_SUBST([LIBCURL], ["-lcurl -lz"])
    AC_DEFINE([USING_CURL], [1],
      [Define if you have curl enabled])
    ],
    [AC_MSG_FAILURE(
      [--with-curl was given, but test for curl failed]),
    [-lcurl -lz -lssl -lcrypto])])

AM_CONDITIONAL(WITH_CURL, [test x"$with_curl" != xno])

AC_ARG_WITH([mmap],
  [AS_HELP_STRING([--with-mmap], [enable support for mmap]),[],[with_mmap=no])

AS_IF([test x"$with_mmap" != xno],
  [AC_DEFINE([USING_MMMap], [1], [Define if you have mmap enabled])])

AC_ARG_WITH([poll],
  [AS_HELP_STRING([--with-poll], [enable support for poll]),[],[with_poll=no])

AS_IF([test x"$with_poll" != xno],
  [AC_DEFINE([USING_POLL], [1], [Define if you have poll() enabled])])

AC_ARG_WITH([ipc-msg],
  [AS_HELP_STRING([--with-ipc-msg], [enable support for msgsnd/msgrcv SV IPC]),[],[with_ipcmmsg=no])

AS_IF([test x"$with_ipcmmsg" != xno],
  [AC_DEFINE([USING_IPC_MSG], [1], [Define to use SV IPC message queue])])

AC_ARG_ENABLE([binstatusfile], [AC_HELP_STRING([--enable-binstatusfile], [Enable support for binary status
file])],
  enable_binstatfile=$enableval, enable_binstatfile=no)

if test x"$enable_binstatfile" = xyes; then
  AC_DEFINE(ENABLE_BINSTATFILE,1,[Define to enable binary status file])
fi

AC_ARG_ENABLE([statusfile], [AC_HELP_STRING([--enable-statusfile], [Enable support for status file])],
  enable_statfile=$enableval, enable_statfile=no)

if test x"$enable_statfile" = xyes || test x"$enable_binstatfile" = xyes; then
  AC_DEFINE(ENABLE_STATFILE,1,[Define to enable status file])
fi

AC_ARG_ENABLE([chilliproxy], [AC_HELP_STRING([--enable-chilliproxy], [Enable support for HTTP AAA Proxy])],
  enable_chilliproxy=$enableval, enable_chilliproxy=no)

if test x"$enable_chilliproxy" = xyes; then

```

```

    AC_DEFINE(ENABLE_CHILLIPROXY,1,[Define to enable HTTP AAA Proxy])
fi

AM_CONDITIONAL(WITH_CHILLIPROXY, [test x"$enable_chilliproxy" = xyes])

AC_ARG_ENABLE(multiroute, [AC_HELP_STRING([--enable-multiroute],[Enable support for multiple routes])],
    enable_multiroute=$enableval, enable_multiroute=no)

if test x"$enable_multiroute" = xyes; then
    AC_DEFINE(ENABLE_MULTIROUTE,1,[Define to enable multiple routes])
fi

AM_CONDITIONAL(WITH_MULTIROUTE, [test x"$enable_multiroute" = xyes])

AC_ARG_ENABLE(multilan, [AC_HELP_STRING([--enable-multilan],[Enable support for multiple LANs])],
    enable_multilan=$enableval, enable_multilan=no)

if test x"$enable_multilan" = xyes; then
    AC_DEFINE(ENABLE_MULTILAN,1,[Define to enable multiple LANs])
fi

AM_CONDITIONAL(WITH_MULTILAN, [test x"$enable_multilan" = xyes])

AC_ARG_ENABLE(chilliradsec, [AC_HELP_STRING([--enable-chilliradsec],[Enable support for RadSec AAA Proxy])],
    enable_chilliradsec=$enableval, enable_chilliradsec=no)

if test x"$enable_chilliradsec" = xyes; then
    AC_DEFINE(ENABLE_CHILLIRADSEC,1,[Define to enable RadSec AAA Proxy])
fi

AM_CONDITIONAL(WITH_CHILLIRADSEC, [test x"$enable_chilliradsec" = xyes])

AC_ARG_ENABLE(chilliredir, [AC_HELP_STRING([--enable-chilliredir],[Enable support for Redir server])],
    enable_chilliredir=$enableval, enable_chilliredir=no)

if test x"$enable_chilliredir" = xyes; then
    AC_DEFINE(ENABLE_CHILLIREDIR,1,[Define to enable Redir server])
fi

AM_CONDITIONAL(WITH_CHILLIREDIR, [test x"$enable_chilliredir" = xyes])

AC_ARG_ENABLE(chilliscript, [AC_HELP_STRING([--enable-chilliscript],[Enable support for chilli_script
helper])],
    enable_chilliscript=$enableval, enable_chilliscript=no)

if test x"$enable_chilliscript" = xyes; then
    AC_DEFINE(ENABLE_CHILLIScript,1,[Define to enable chilli_script helper])
fi

AM_CONDITIONAL(WITH_CHILLIScript, [test x"$enable_chilliscript" = xyes])

AC_ARG_ENABLE(cluster, [AC_HELP_STRING([--enable-cluster],[Enable support for clustering])],
    enable_cluster=$enableval, enable_cluster=no)

if test x"$enable_cluster" = xyes; then
    AC_DEFINE(ENABLE_CLUSTER,1,[Define to enable cluster])
fi

AM_CONDITIONAL(WITH_CLUSTER, [test x"$enable_cluster" = xyes])

AC_ARG_ENABLE(sessionstate, [AC_HELP_STRING([--enable-sessionstate],[Enable extended use of the ChilliSpot-
Session-State attribute])],
    enable_sessionstate=$enableval, enable_sessionstate=no)

if test x"$enable_sessionstate" = xyes; then
    AC_DEFINE(ENABLE_SESSIONSTATE,1,[Define to enable extended use of the ChilliSpot-Session-State attribute])
fi

AC_ARG_ENABLE(sessionid, [AC_HELP_STRING([--enable-sessionid],[Enable the use of the ChilliSpot-Session-Id
attribute])],
    enable_sessionid=$enableval, enable_sessionid=no)

```

```

if test x"$enable_sessionid" = xyes; then
    AC_DEFINE(ENABLE_SESSIONID,1,[Define to enable the use of the ChilliSpot-Session-Id attribute])
fi

AC_ARG_ENABLE(apsessionid, [AC_HELP_STRING([--enable-apsessionid],[Enable the use of the ChilliSpot-AP-Session-Id attribute])],
    enable_apsessionid=$enableval, enable_apsessionid=no)

if test x"$enable_apsessionid" = xyes; then
    AC_DEFINE(ENABLE_APSESSIONID,1,[Define to enable the use of the ChilliSpot-AP-Session-Id attribute])
fi

AC_ARG_ENABLE(chillispotconfig, [AC_HELP_STRING([--disable-chillispotconfig],[Disable the use of the ChilliSpot-Config attribute])],
    enable_chillispotconfig=$enableval, enable_chillispotconfig=yes)

if test x"$enable_chillispotconfig" = xyes; then
    AC_DEFINE(ENABLE_CHILLISPOTCONFIG,1,[Define to enable the use of the ChilliSpot-Config attribute])
fi

AC_ARG_ENABLE(mdns, [AC_HELP_STRING([--enable-mdns],[Enable support for MDNS])],
    enable_mdns=$enableval, enable_mdns=no)

if test x"$enable_mdns" = xyes; then
    AC_DEFINE(ENABLE_MDNS,1,[Define to enable mDNS])
fi

AM_CONDITIONAL(WITH_MDNS, [test x"$enable_mdns" = xyes])

AC_ARG_ENABLE(netbios, [AC_HELP_STRING([--enable-netbios],[Enable support for NetBIOS])],
    enable_netbios=$enableval, enable_netbios=no)

if test x"$enable_netbios" = xyes; then
    AC_DEFINE(ENABLE_NETBIOS,1,[Define to enable NetBIOS])
fi

AM_CONDITIONAL(WITH_NETBIOS, [test x"$enable_netbios" = xyes])

AC_ARG_ENABLE(ieee8023, [AC_HELP_STRING([--enable-ieee8023],[Enable support for IEEE 802.3 Ethernet])],
    enable_ieee8023=$enableval, enable_ieee8023=no)

if test x"$enable_ieee8023" = xyes; then
    AC_DEFINE(ENABLE_IEEE8023,1,[Define to enable IEEE 802.3])
fi

AM_CONDITIONAL(WITH_IEEE8023, [test x"$enable_ieee8023" = xyes])

AC_ARG_ENABLE(pppoe, [AC_HELP_STRING([--enable-pppoe],[Enable support for PPPoE])],
    enable_pppoe=$enableval, enable_pppoe=no)

if test x"$enable_pppoe" = xyes; then
    AC_DEFINE(ENABLE_PPPOE,1,[Define to enable PPPoE])
fi

AM_CONDITIONAL(WITH_PPPOE, [test x"$enable_pppoe" = xyes])

AC_ARG_ENABLE(l2tpppp, [AC_HELP_STRING([--enable-l2tpppp],[Enable support for L2TP/PPP Tunneling])],
    enable_l2tpppp=$enableval, enable_l2tpppp=no)

if test x"$enable_l2tpppp" = xyes; then
    AC_DEFINE(ENABLE_L2TP_PPP,1,[Define to enable L2TP/PPP])
fi

AM_CONDITIONAL(WITH_L2TP_PPP, [test x"$enable_l2tpppp" = xyes])

AC_ARG_ENABLE(eapol, [AC_HELP_STRING([--enable-eapol],[Enable support for EAPOL])],
    enable_eapol=$enableval, enable_eapol=no)

if test x"$enable_eapol" = xyes; then
    AC_DEFINE(ENABLE_EAPOL,1,[Define to enable EAPOL])
fi

```

```

fi

AM_CONDITIONAL(WITH_EAPOL, [test x"$enable_eapol" = xyes])

AC_ARG_ENABLE(miniportal, [AC_HELP_STRING([--enable-miniportal],[Enable support Coova miniportal])],
    enable_miniportal=$enableval, enable_miniportal=no)

if test x"$enable_miniportal" = xyes ; then
    AC_DEFINE(ENABLE_MINIportal,1,[Define to enable Coova miniportal])
fi

AM_CONDITIONAL(WITH_MINIportal, [test x"$enable_miniportal" = xyes])

AC_ARG_ENABLE(miniconfig, [AC_HELP_STRING([--enable-miniconfig],[Enable support minimal cmdline config])],
    enable_miniconfig=$enableval, enable_miniconfig=no)

if test x"$enable_miniconfig" = xyes ; then
    AC_DEFINE(ENABLE_MINICONFIG,1,[Define to enable minimal cmdline config])
fi

AM_CONDITIONAL(WITH_MINICONFIG, [test x"$enable_miniconfig" = xyes])

AC_ARG_ENABLE(ewtapi, [AC_HELP_STRING([--enable-ewtapi],[Enable support for CoovaEWT API])],
    enable_ewtapi=$enableval, enable_ewtapi=no)

if test x"$enable_ewtapi" = xyes; then
    AC_DEFINE(ENABLE_EWTAPI,1,[Define to enable CoovaEWT API])
fi

AC_ARG_ENABLE(libjson, [AC_HELP_STRING([--enable-libjson],[Enable support for libjson])],
    enable_libjson=$enableval, enable_libjson=no)

AM_CONDITIONAL(WITH_EWTAPI, [test x"$enable_ewtapi" = xyes])
AM_CONDITIONAL(WITH_JSONLIB, [test x"$enable_ewtapi" = xyes || test x"$enable_libjson" = xyes])

AC_ARG_ENABLE(ssdp, [AC_HELP_STRING([--enable-ssdp],[Enable support for Simple Service Discovery Protocol])],
    enable_ssdp=$enableval, enable_ssdp=no)

if test x"$enable_ssdp" = xyes ; then
    AC_DEFINE(ENABLE_SSDP,1,[Define to enable Simple Service Discovery Protocol])
fi

AC_ARG_ENABLE(layer3, [AC_HELP_STRING([--enable-layer3],[Enable support for Layer3 only operation])],
    enable_layer3=$enableval, enable_layer3=no)

if test x"$enable_layer3" = xyes ; then
    AC_DEFINE(ENABLE_LAYER3,1,[Define to enable Layer3 only support])
fi

AC_ARG_ENABLE(modules, [AC_HELP_STRING([--enable-modules],[Enable dynamically loadable modules
(experimental)])],
    enable_modules=$enableval, enable_modules=no)

if test x"$enable_modules" = xyes ; then
    AC_DEFINE(ENABLE_MODULES,1,[Define to enable dynamically loadable modules])
fi

AM_CONDITIONAL(WITH_MODULES, [test x"$enable_modules" = xyes])

AC_ARG_ENABLE(extadmvsas, [AC_HELP_STRING([--enable-extadmvsas],[Enable admin-user VSA attribute support])],
    enable_extadmvsas=$enableval, enable_extadmvsas=no)

if test x"$enable_extadmvsas" = xyes; then
    AC_DEFINE(ENABLE_EXTADMVSA,1,[Define to enable admin-user VSA support])
fi

AC_ARG_ENABLE(redirinject, [AC_HELP_STRING([--enable-redirinject],[Enable Redir content-injection support])],
    enable_redirinject=$enableval, enable_redirinject=no)

if test x"$enable_redirinject" = xyes; then
    AC_DEFINE(ENABLE_REDIRINJECT,1,[Define to Redir content-injection support])

```

```

fi

AM_CONDITIONAL(WITH_REDIRINJECT, [test x"$enable_redirinject" = xyes])

AC_ARG_ENABLE(netnat, [AC_HELP_STRING([--enable-netnat],[Enable net interface nat (experimental)]),
    enable_netnat=$enableval, enable_netnat=no)

if test x"$enable_netnat" = xyes ; then
    AC_DEFINE(ENABLE_NETNAT,1,[Define to enable network interface nat])
fi

AM_CONDITIONAL(WITH_NETNAT, [test x"$enable_netnat" = xyes])

AC_ARG_ENABLE(useragent, [AC_HELP_STRING([--enable-useragent],[Enable recording user-agent]),
    enable_useragent=$enableval, enable_useragent=no)

if test x"$enable_useragent" = xyes ; then
    AC_DEFINE(ENABLE_USERAGENT,1,[Define enable the recording of user-agent])
fi

AC_ARG_ENABLE(acceptlanguage, [AC_HELP_STRING([--enable-acceptlanguage],[Enable recording Accept-Language]),
    enable_acceptlanguage=$enableval, enable_acceptlanguage=no)

if test x"$enable_acceptlanguage" = xyes ; then
    AC_DEFINE(ENABLE_ACCEPTLANGUAGE,1,[Define enable the recording of accept-language])
fi

AC_ARG_ENABLE(location, [AC_HELP_STRING([--enable-location],[Enable Location Awareness]),
    enable_location=$enableval, enable_location=no)

if test x"$enable_location" = xyes ; then
    AC_DEFINE(ENABLE_LOCATION,1,[Define enable Location Awareness])
fi

AM_CONDITIONAL(WITH_LOCATION, [test x"$enable_location" = xyes])

AC_ARG_ENABLE(forcedns, [AC_HELP_STRING([--enable-forcedns],[Enable the forcing (NAT) of DNS]),
    enable_forcedns=$enableval, enable_forcedns=no)

if test x"$enable_forcedns" = xyes ; then
    AC_DEFINE(ENABLE_FORCEDNS,1,[Enable the forcing (NAT) of DNS])
fi

AC_CHECK_LIB(rt, clock_gettime)

AC_ARG_ENABLE(config,
    [ --enable-config=file],
    [ if test "x$enableval" != "x" ; then
        config_file=$enableval
      else
        echo "Error! Needs an argument"
        exit -1
      fi
    ])

if test x"$config_file" != x ; then
    AC_DEFINE_UNQUOTED(ENABLE_CONFIG,"$config_file",[none])
fi

AM_CONDITIONAL(WITH_CONFIG, [test x"$enable_config" = xyes])

AC_SUBST(sysconfdir)
AC_CONFIG_FILES([Makefile
    bstring/Makefile
    src/Makefile
    src/mssl/Makefile
    www/Makefile
    doc/Makefile
    conf/Makefile

```

```
debian/Makefile
miniportal/Makefile
json/Makefile
distro/Makefile
distro/suse/coova-chilli.spec
distro/nethserver/coova-chilli.spec
distro/redhat/coova-chilli.spec])
```

AC_OUTPUT

We expose more compile options in the **Makefile**.

```
#
# Copyright (C) 2007-2014 OpenWrt.org
#
# This is free software, licensed under the GNU General Public License v2.
# See /LICENSE for more information.
#

include $(TOPDIR)/rules.mk

PKG_NAME:=coova-chilli
PKG_VERSION:=1.3.0+20141128
PKG_MAINTAINER:=Imre Kaloz <kaloz@openwrt.org>
PKG_LICENSE:=GPL-2.0+
PKG_LICENSE_FILES:=COPYING
PKG_RELEASE:=4

PKG_SOURCE_PROTO:=git
PKG_SOURCE_URL:=git://github.com/coova/coova-chilli
PKG_SOURCE_SUBDIR:=$(PKG_NAME)-$(PKG_VERSION)
PKG_SOURCE_VERSION:=b93de20a288c01c2ba28e96e31ad6da01627f45f
PKG_SOURCE:=$(PKG_NAME)-$(PKG_VERSION).tar.gz
PKG_MD5SUM:=2adb27ec56172b18c5beee359dd7898d

PKG_INSTALL:=1

PKG_CONFIG_DEPENDS := \
    COOVACHILLI_DISABLE_CHILLIQUERY \
    COOVACHILLI_DISABLE_LEAKYBUCKET \
    COOVACHILLI_DISABLE_UAMANYIP \
    COOVACHILLI_DISABLE_UAMUIPORT \
    COOVACHILLI_DISABLE_ACCOUNTING_ONOFF \
    COOVACHILLI_DISABLE_TAP \
    COOVACHILLI_DISABLE_TCPRESET \
    COOVACHILLI_DISABLE_RADPROXY \
    COOVACHILLI_DISABLE_JSON \
    COOVACHILLI_DISABLE_DEBUG \
    COOVACHILLI_DISABLE_DHCPRADIUS \
    COOVACHILLI_ENABLE_WPAD \
    COOVACHILLI_ENABLE_GARDENACCOUNTING \
    COOVACHILLI_ENABLE_GARDENEXT \
    COOVACHILLI_ENABLE_INSPECT \
    COOVACHILLI_DISABLE_COA \
    COOVACHILLI_ENABLE_DHCPD \
    COOVACHILLI_ENABLE_DEBUG2 \
    COOVACHILLI_ENABLE_SESSGARDEN \
    COOVACHILLI_ENABLE_SESSPROXY \
    COOVACHILLI_ENABLE_SESSDHCP \
    COOVACHILLI_ENABLE_SESSDNS \
    COOVACHILLI_ENABLE_CHILLIXML \
    COOVACHILLI_ENABLE_PROXYVSA \
    COOVACHILLI_ENABLE_DNSLOG \
    COOVACHILLI_ENABLE_IPWHITELIST \
    COOVACHILLI_ENABLE_UAMDOMAINFILE \
    COOVACHILLI_ENABLE_REDIRDNSREQ \
    COOVACHILLI_DISABLE_IEEE8021Q \
    COOVACHILLI_ENABLE_LARGE_LIMITS \
    COOVACHILLI_WITH_NFQUEUE \
```

```

COOVACHILLI_WITH_AVL \
COOVACHILLI_WITH_NFCOOVA \
COOVACHILLI_WITHOUT_SFHASH \
COOVACHILLI_WITH_LOOKUP3 \
COOVACHILLI_WITH_PATRICIA \
COOVACHILLI_ENABLE_AUTHEDALLOWED \
COOVACHILLI_WITHOUT_IPV6 \
COOVACHILLI_WITH_PCAP \
COOVACHILLI_WITH_CURL \
COOVACHILLI_WITH_MMAP \
COOVACHILLI_WITH_POLL \
COOVACHILLI_WITH_IPC_MSG \
COOVACHILLI_ENABLE_BINSTATUSFILE \
COOVACHILLI_ENABLE_STATUSFILE \
COOVACHILLI_ENABLE_CHILLIPROXY \
COOVACHILLI_ENABLE_MULTIRROUTE \
COOVACHILLI_ENABLE_MULTILAN \
COOVACHILLI_ENABLE_CHILLIRADSEC \
COOVACHILLI_ENABLE_CHILLIREDIR \
COOVACHILLI_ENABLE_CHILLISCRIP \
COOVACHILLI_ENABLE_CLUSTER \
COOVACHILLI_ENABLE_SESSIONSTATE \
COOVACHILLI_ENABLE_SESSIONID \
COOVACHILLI_ENABLE_APSESSIONID \
COOVACHILLI_DISABLE_CHILLISPOTCONFIG \
COOVACHILLI_ENABLE_MDNS \
COOVACHILLI_ENABLE_NETBIOS \
COOVACHILLI_ENABLE_IEEE8023 \
COOVACHILLI_ENABLE_PPPOE \
COOVACHILLI_ENABLE_L2TPPPP \
COOVACHILLI_ENABLE_EAPOL \
COOVACHILLI_ENABLE_MINIPORTAL \
COOVACHILLI_ENABLE_MINICONFIG \
COOVACHILLI_ENABLE_EWTAPI \
COOVACHILLI_ENABLE_LIBJSON \
COOVACHILLI_ENABLE_SSDP \
COOVACHILLI_ENABLE_LAYER3 \
COOVACHILLI_ENABLE_MODULES \
COOVACHILLI_ENABLE_EXTADMVSA \
COOVACHILLI_ENABLE_REDIRINJECT \
COOVACHILLI_ENABLE_NETNAT \
COOVACHILLI_ENABLE_USERAGENT \
COOVACHILLI_ENABLE_ACCEPTLANGUAGE \
COOVACHILLI_ENABLE_LOCATION \
COOVACHILLI_ENABLE_FORCEDNS \
COOVACHILLI_DISABLE_SSL \
COOVACHILLI_ENABLE_OPENSSL \
COOVACHILLI_ENABLE_MATRIXSSL \
COOVACHILLI_ENABLE_MATRIXSSL_CLI \
COOVACHILLI_ENABLE_CYASSL

```

```

include $(INCLUDE_DIR)/package.mk
include $(INCLUDE_DIR)/kernel.mk

```

```

define Package/coova-chilli
  SUBMENU:=Captive Portals
  SECTION:=net
  CATEGORY:=Network
  DEPENDS:= \
    +kmod-tun \
    +librt \
    +COOVACHILLI_ENABLE_MATRIXSSL:libmatrixssl \
    +COOVACHILLI_ENABLE_CYASSL:libcyassl \
    +COOVACHILLI_ENABLE_OPENSSL:libopenssl \
    +COOVACHILLI_ENABLE_LIBJSON:libjson-c \
    +COOVACHILLI_WITH_NFQUEUE:libnetfilter-queue \
    +COOVACHILLI_WITH_NFQUEUE:libnfnetlink \
    +COOVACHILLI_WITH_NFQUEUE:libmnl \
    +COOVACHILLI_WITH_PCAP:libpcap \
    +COOVACHILLI_WITH_CURL:libcares
  TITLE:=Wireless LAN HotSpot controller (Coova Chilli Version)
endef

```

```

URL:=http://www.coova.org/CoovaChilli
MENU:=1
endif

define Package/coova-chilli/description
    CoovaChilli is an open source access controller for wireless LAN
    access points and is based on ChilliSpot. It is used for authenticating
    users of a wireless (or wired) LAN. It supports web based login (UAM)
    which is today's standard for public HotSpots and it supports Wireless
    Protected Access (WPA) which is the standard of the future.
    Authentication, authorization and accounting (AAA) is handled by your
    favorite radius server.
endif

define Package/coova-chilli/config
    source "$(SOURCE)/Config.in"
endif

define KernelPackage/ipt-coova
    URL:=http://www.coova.org/CoovaChilli
    SUBMENU:=Netfilter Extensions
    DEPENDS:=coova-chilli +kmod-ipt-core +libxtables
    TITLE:=Coova netfilter module
    FILES=$(PKG_BUILD_DIR)/src/linux/xt_*.$(LINUX_KMOD_SUFFIX)
    AUTOLOAD:=$(call AutoProbe,xt_coova)
endif

define KernelPackage/ipt-coova/description
    Netfilter kernel module for CoovaChilli
    Includes:
    - coova
endif

DISABLE_NLS=

TARGET_CFLAGS += $(FPIC)

CONFIGURE_VARS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)"

MAKE_FLAGS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)"

MAKE_INSTALL_FLAGS += \
    ARCH="$(LINUX_KARCH)" \
    KERNEL_DIR="$(LINUX_DIR)" \
    INSTALL_MOD_PATH="$(PKG_INSTALL_DIR)"

define Build/Prepare
$(call Build/Prepare/Default)
    ( cd $(PKG_BUILD_DIR) ; \
        [ -f ./configure ] || { \
            ./bootstrap ; \
        } \
    )
endif

define Build/Configure
$(call Build/Configure/Default, \
    $(if $(CONFIG_COOVACHILLI_DISABLE_CHILLIQUERY),--disable-chilliquery) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_LEAKYBUCKET),--disable-leakybucket) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_UAMANYIP),--disable-uamanyip) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_UAMUIPORT),--disable-uamuiport) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_ACCOUNTING_ONOFF),--disable-accounting-onoff) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_TAP),--disable-tap) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_TCPRESET),--disable-tcpreset) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_RADPROXY),--disable-radproxy) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_JSON),--disable-json) \
    $(if $(CONFIG_COOVACHILLI_DISABLE_DEBUG),--disable-debug) \

```

```
$(if $(CONFIG_COOVACHILLI_DISABLE_DHCPDPRADII),--disable-dhcpdpradii) \
$(if $(CONFIG_COOVACHILLI_ENABLE_WPAD),--enable-wpad) \
$(if $(CONFIG_COOVACHILLI_ENABLE_GARDENACCOUNTING),--enable-gardenaccounting,) \
$(if $(CONFIG_COOVACHILLI_ENABLE_GARDENEXT),--enable-gardenext) \
$(if $(CONFIG_COOVACHILLI_ENABLE_INSPECT),--enable-inspect) \
$(if $(CONFIG_COOVACHILLI_DISABLE_COA),--disable-coa) \
$(if $(CONFIG_COOVACHILLI_ENABLE_DHCPDPT),--enable-dhcpdpt) \
$(if $(CONFIG_COOVACHILLI_ENABLE_DEBUG2),--enable-debug2) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSGARDEN),--enable-sessgarden) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSPROXY),--enable-sessproxy) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSDHCP),--enable-sessdhcp) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSDNS),--enable-sessdns) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CHILLIXML),--enable-chillixml) \
$(if $(CONFIG_COOVACHILLI_ENABLE_PROXYVSA),--enable-proxyvsa) \
$(if $(CONFIG_COOVACHILLI_ENABLE_DNSLOG),--enable-dnslog) \
$(if $(CONFIG_COOVACHILLI_ENABLE_IPWHITELIST),--enable-ipwhitelist) \
$(if $(CONFIG_COOVACHILLI_ENABLE_UAMDOMAINFILE),--enable-uamdomainfile) \
$(if $(CONFIG_COOVACHILLI_ENABLE_REDIRDNSREQ),--enable-redirdnsreq) \
$(if $(CONFIG_COOVACHILLI_DISABLE_IEEE8021Q),--disable-ieee8021q) \
$(if $(CONFIG_COOVACHILLI_ENABLE_LARGEIMLITS),--enable-largeimits) \
$(if $(CONFIG_COOVACHILLI_ENABLE_OPENSSL),--with-openssl) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MATRIXSSL),--with-matrixssl) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MATRIXSSL_CLI),--with-matrixssl-cli) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CYASSL),--with-cyassl) \
$(if $(CONFIG_COOVACHILLI_WITH_NFQUEUE),--with-nfqueue) \
$(if $(CONFIG_COOVACHILLI_WITH_AVL),--with-avl) \
$(if $(CONFIG_COOVACHILLI_WITH_NFCOOVA),--with-nfcoova) \
$(if $(CONFIG_COOVACHILLI_WITHOUT_SFHASH),--without-sfhash) \
$(if $(CONFIG_COOVACHILLI_WITH_LOOKUP3),--with-lookup3) \
$(if $(CONFIG_COOVACHILLI_WITH_PATRICIA),--with-patricia) \
$(if $(CONFIG_COOVACHILLI_ENABLE_AUTHEDALLOWED),--enable-authedallowed) \
$(if $(CONFIG_COOVACHILLI_WITHOUT_IPV6),--without-ipv6) \
$(if $(CONFIG_COOVACHILLI_WITH_PCAP),--with-pcap) \
$(if $(CONFIG_COOVACHILLI_WITH_CURL),--with-curl) \
$(if $(CONFIG_COOVACHILLI_WITH_MMAP),--with-mmap) \
$(if $(CONFIG_COOVACHILLI_WITH_POLL),--with-poll) \
$(if $(CONFIG_COOVACHILLI_WITH_IPC_MSG),--with-ipc-msg) \
$(if $(CONFIG_COOVACHILLI_ENABLE_BINSTATUSFILE),--enable-binstatusfile) \
$(if $(CONFIG_COOVACHILLI_ENABLE_STATUSFILE),--enable-statusfile) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CHILLIPROXY),--enable-chilliproxy) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MULTIRROUTE),--enable-multiroute) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MULTILAN),--enable-multilan) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CHILLIRADSEC),--enable-chilliradsec) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CHILLIREDIR),--enable-chilliredir) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CHILLISCRIP),--enable-chilliscrpt) \
$(if $(CONFIG_COOVACHILLI_ENABLE_CLUSTER),--enable-cluster) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSIONSTATE),--enable-sessionstate) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SESSIONID),--enable-sessionid) \
$(if $(CONFIG_COOVACHILLI_ENABLE_APSESSIONID),--enable-apsessionid) \
$(if $(CONFIG_COOVACHILLI_DISABLE_CHILLISPOTCONFIG),--disable-chillispotconfig) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MDNS),--enable-mdns) \
$(if $(CONFIG_COOVACHILLI_ENABLE_NETBIOS),--enable-netbios) \
$(if $(CONFIG_COOVACHILLI_ENABLE_IEEE8023),--enable-ieee8023) \
$(if $(CONFIG_COOVACHILLI_ENABLE_PPPOE),--enable-pppoe) \
$(if $(CONFIG_COOVACHILLI_ENABLE_L2TPPPP),--enable-l2tp PPP) \
$(if $(CONFIG_COOVACHILLI_ENABLE_EAPOL),--enable-eapol) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MINIportal),--enable-miniportal) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MINICONFIG),--enable-miniconfig) \
$(if $(CONFIG_COOVACHILLI_ENABLE_EWTAPI),--enable-ewtapi) \
$(if $(CONFIG_COOVACHILLI_ENABLE_LIBJSON),--enable-libjson) \
$(if $(CONFIG_COOVACHILLI_ENABLE_SSDP),--enable-ssdp) \
$(if $(CONFIG_COOVACHILLI_ENABLE_LAYER3),--enable-layer3) \
$(if $(CONFIG_COOVACHILLI_ENABLE_MODULES),--enable-modules) \
$(if $(CONFIG_COOVACHILLI_ENABLE_EXTADMVSA),--enable-extadmvs) \
$(if $(CONFIG_COOVACHILLI_ENABLE_REDIRINJECT),--enable-redirinject) \
$(if $(CONFIG_COOVACHILLI_ENABLE_NETNAT),--enable-netnat) \
$(if $(CONFIG_COOVACHILLI_ENABLE_USERAGENT),--enable-useragent) \
$(if $(CONFIG_COOVACHILLI_ENABLE_ACCEPTLANGUAGE),--enable-acceptlanguage) \
$(if $(CONFIG_COOVACHILLI_ENABLE_LOCATION),--enable-location) \
$(if $(CONFIG_COOVACHILLI_ENABLE_FORCEDNS),--enable-forcedns) \
$(if $(CONFIG_PACKAGE_kmod-ipt-coova),--with-nfcoova) \
```

```

    )
endif

define Package/coova-chilli/conffiles
/etc/config/chilli
endif

define Package/coova-chilli/install
$(INSTALL_DIR) $(1)/etc
$(INSTALL_CONF) $(PKG_INSTALL_DIR)/etc/chilli.conf $(1)/etc/
$(INSTALL_DIR) $(1)/etc/chilli
$(CP) $(PKG_INSTALL_DIR)/etc/chilli/* $(1)/etc/chilli/
$(INSTALL_DIR) $(1)/etc/hotplug.d/iface
$(INSTALL_DATA) ./files/chilli.hotplug $(1)/etc/hotplug.d/iface/30-chilli
$(INSTALL_DIR) $(1)/usr/sbin
$(INSTALL_BIN) $(PKG_INSTALL_DIR)/usr/sbin/chilli* $(1)/usr/sbin/
$(INSTALL_DIR) $(1)/usr/lib/
$(CP) $(PKG_INSTALL_DIR)/usr/lib/lib*.so.* $(1)/usr/lib/
$(if $(CONFIG_PACKAGE_kmod-ipt-coova), \
    $(INSTALL_DIR) $(1)/usr/lib/iptables; \
    $(CP) $(PKG_INSTALL_DIR)/usr/lib/iptables/lib*.so $(1)/usr/lib/iptables/ \
)
$(INSTALL_DIR) $(1)/etc/init.d
$(INSTALL_BIN) files/chilli.init $(1)/etc/init.d/chilli
$(INSTALL_DIR) $(1)/etc/config
$(INSTALL_DATA) files/chilli.config $(1)/etc/config/chilli
$(INSTALL_DIR) $(1)/lib/firewall
$(CP) files/chilli.firewall $(1)/lib/firewall/chilli.sh
endif

$(eval $(call BuildPackage,coova-chilli))
$(eval $(call KernelPackage,ipt-coova))

```

Config.in

Following the compile definitions from the file **coova-chilli-1.3.0+20141128/configure.in**:

Open the **package/feeds/packages/coova-chilli/Config.in** file:

```

# CoovaChilli advanced configuration

menu "Configuration"
    depends on PACKAGE_coova-chilli

config COOVACHILLI_PROXY
    bool "Enable support for chilli proxy. Required for AAA Proxy through http"
    default n

config COOVACHILLI_REDIR
    bool "Enable support for redir server. Required for uamregex"
    default n

config COOVACHILLI_MINIPORTAL
    bool "Enable support Coova miniportal"
    default n

config COOVACHILLI_USERAGENT
    bool "Enable recording user-agent"
    default n

config COOVACHILLI_DNSLOG
    bool "Enable support to log DNS name queries"
    default n

config COOVACHILLI_UAMDOMAINFILE
    bool "Enable loading of mass uamdomains from file"
    default n

config COOVACHILLI_LARGE_LIMITS
    bool "Enable larger limits for use with non-embedded systems"
    default n

choice
    prompt "SSL library"
    default COOVACHILLI_NOSSL

config COOVACHILLI_NOSSL
    bool "No SSL support"

config COOVACHILLI_MATRIXSSL
    bool "MatrixSSL"

config COOVACHILLI_CYASSL
    bool "CyaSSL"

config COOVACHILLI_OPENSSL
    bool "OpenSSL"

endchoice

endmenu

```

Patch with the following content:

```

# CoovaChilli advanced configuration

menu "Configuration"
    depends on PACKAGE_coova-chilli

config COOVACHILLI_DISABLE_CHILLIQUERY
    bool "Disable chilli_query"
    default n

config COOVACHILLI_DISABLE_LEAKYBUCKET
    bool "Disable use of leaky bucket shaping"
    default n

```

```
config COOVACHILLI_DISABLE_UAMANYIP
    bool "Disable use of uamanyip"
    default n

config COOVACHILLI_DISABLE_UAMUIPORT
    bool "Disable use of uamuiport"
    default n

config COOVACHILLI_DISABLE_ACCOUNTING_ONOFF
    bool "Disable use of Accounting-On and Accounting-Off"
    default n

config COOVACHILLI_DISABLE_TAP
    bool "Disable support for tap interface (tun only)"
    default n

config COOVACHILLI_DISABLE_TCPRESET
    bool "Disable support for TCP reset of filtered connections"
    default n

config COOVACHILLI_DISABLE_RADPROXY
    bool "Disable support RADIUS (EAP) Proxy"
    default n

config COOVACHILLI_DISABLE_JSON
    bool "Disable support for JSON"
    default n

config COOVACHILLI_DISABLE_DEBUG
    bool "Disable debugging messages"
    default n

config COOVACHILLI_DISABLE_DHCPRADIUS
    bool "Disable support DHCP/RADIUS integration"
    default n

config COOVACHILLI_ENABLE_WPAD
    bool "Enable support WPAD"
    default n

config COOVACHILLI_ENABLE_GARDENACCOUNTING
    bool "Enable walled garden accounting"
    default n

config COOVACHILLI_ENABLE_GARDENEXT
    bool "Enable extended walled garden features"
    default n

config COOVACHILLI_ENABLE_INSPECT
    bool "Enable inspect feature in cmdsock"
    default n

config COOVACHILLI_DISABLE_COA
    bool "Disable CoA RADIUS support"
    default n

config COOVACHILLI_ENABLE_DHCPOPT
    bool "Enable support for DHCP option setting"
    default n

config COOVACHILLI_ENABLE_DEBUG2
    bool "Enable verbose debugging"
    default n

config COOVACHILLI_ENABLE_SESSGARDEN
    bool "Enable support for session-based walled garden"
    default n

config COOVACHILLI_ENABLE_SESSPROXY
    bool "Enable support for per session postauth proxy"
    default n
```

```
config COOVACHILLI_ENABLE_SESSDHCP
    bool "Enable support for per session DHCP relay"
    default n

config COOVACHILLI_ENABLE_SESSDNS
    bool "Enable support for per session DNS enforcement"
    default n

config COOVACHILLI_ENABLE_CHILLIXML
    bool "Enable use of chillixml"
    default n

config COOVACHILLI_ENABLE_PROXYVSA
    bool "Enable support for VSA attribute proxy"
    default n

config COOVACHILLI_ENABLE_DNSLOG
    bool "Enable support to log DNS name queries"
    default n

config COOVACHILLI_ENABLE_IPWHITELIST
    bool "Enable file based IP white list"
    default n

config COOVACHILLI_ENABLE_UAMDOMAINFILE
    bool "Enable loading of mass uamdomains from file"
    default n

config COOVACHILLI_ENABLE_REDIRDNSREQ
    bool "Enable the sending of a DNS query on redirect"
    default n

config COOVACHILLI_DISABLE_IEEE8021Q
    bool "Disable support for IEEE 802.1Q"
    default n

config COOVACHILLI_ENABLE_LARGE_LIMITS
    bool "Enable larger limits for use with non-embedded systems"
    default n

config COOVACHILLI_WITH_NFQUEUE
    bool "Enable support for netfilter_queue"
    default n

config COOVACHILLI_WITH_AVL
    bool "Enable support for AVL library"
    default n

config COOVACHILLI_WITH_NFCOOVA
    bool "Enable support for coova netfilter module"
    default n

config COOVACHILLI_WITHOUT_SFHASH
    bool "Disable SuperFastHash use"
    default n

config COOVACHILLI_WITH_LOOKUP3
    bool "Enable Jenkins lookup3 use"
    default n

config COOVACHILLI_WITH_PATRICIA
    bool "Enable Patricia use"
    default n

config COOVACHILLI_ENABLE_AUTHEDALLOWED
    bool "Enable Authorized Garden"
    default n

config COOVACHILLI_WITHOUT_IPV6
    bool "Enable IPv6"
```

```
    default n

config COOVACHILLI_WITH_PCAP
    bool "Enable support for pcap"
    default n

config COOVACHILLI_WITH_CURL
    bool "Enable support for curl"
    default n

config COOVACHILLI_WITH_MMAP
    bool "Enable support for mmap"
    default n

config COOVACHILLI_WITH_POLL
    bool "Enable support for poll"
    default n

config COOVACHILLI_WITH_IPC_MSG
    bool "Enable support for msgsnd/msgrcv SV IPC"
    default n

config COOVACHILLI_ENABLE_BINSTATUSFILE
    bool "Enable support for binary status file"
    default n

config COOVACHILLI_ENABLE_STATUSFILE
    bool "Enable support for status file"
    default n

config COOVACHILLI_ENABLE_CHILLIPROXY
    bool "Enable support for HTTP AAA Proxy"
    default n

config COOVACHILLI_ENABLE_MULTIRoute
    bool "Enable support for multiple routes"
    default n

config COOVACHILLI_ENABLE_MULTILAN
    bool "Enable support for multiple LANs"
    default n

config COOVACHILLI_ENABLE_CHILLIRADSEC
    bool "Enable support for RadSec AAA Proxy"
    default n

config COOVACHILLI_ENABLE_CHILLIREDIR
    bool "Enable support for Redir server"
    default n

config COOVACHILLI_ENABLE_CHILLIScript
    bool "Enable support for chilli_script helper"
    default n

config COOVACHILLI_ENABLE_CLUSTER
    bool "Enable support for clustering"
    default n

config COOVACHILLI_ENABLE_SESSIONSTATE
    bool "Enable extended use of the CoovaChilli-Session-State attribute"
    default n

config COOVACHILLI_ENABLE_SESSIONID
    bool "Enable the use of the CoovaChilli-Session-Id attribute"
    default n

config COOVACHILLI_ENABLE_APSESSIONID
    bool "Enable the use of the CoovaChilli-AP-Session-Id attribute"
    default n

config COOVACHILLI_DISABLE_CHILLISPOTCONFIG
```

```
    bool "Disable the use of the ChilliSpot-Config attribute"
    default n

config COOVACHILLI_ENABLE_MDNS
    bool "Enable support for MDNS"
    default n

config COOVACHILLI_ENABLE_NETBIOS
    bool "Enable support for NetBIOS"
    default n

config COOVACHILLI_ENABLE_IEEE8023
    bool "Enable support for IEEE 802.3 Ethernet"
    default n

config COOVACHILLI_ENABLE_PPPOE
    bool "Enable support for PPPoE"
    default n

config COOVACHILLI_ENABLE_L2TPPPP
    bool "Enable support for L2TP/PPP Tunneling"
    default n

config COOVACHILLI_ENABLE_EAPOL
    bool "Enable support for EAPOL"
    default n

config COOVACHILLI_ENABLE_MINIPORTAL
    bool "Enable support Coova miniportal"
    default n

config COOVACHILLI_ENABLE_MINICONFIG
    bool "Enable support minimal cmdline config"
    default n

config COOVACHILLI_ENABLE_EWTAPI
    bool "Enable support for CoovaEWT API"
    default n

config COOVACHILLI_ENABLE_LIBJSON
    bool "Enable support for libjson"
    default n

config COOVACHILLI_ENABLE_SSDP
    bool "Enable support for Simple Service Discovery Protocol"
    default n

config COOVACHILLI_ENABLE_LAYER3
    bool "Enable support for Layer3 only operation"
    default n

config COOVACHILLI_ENABLE_MODULES
    bool "Enable dynamically loadable modules (experimental)"
    default n

config COOVACHILLI_ENABLE_EXTADMVSA
    bool "Enable admin-user VSA attribute support"
    default n

config COOVACHILLI_ENABLE_REDIRECTINJECT
    bool "Enable Redir content-injection support"
    default n

config COOVACHILLI_ENABLE_NETNAT
    bool "Enable net interface nat (experimental)"
    default n

config COOVACHILLI_ENABLE_USERAGENT
    bool "Enable recording user-agent"
    default n
```

```

config COOVACHILLI_ENABLE_ACCEPTLANGUAGE
    bool "Enable recording Accept-Language"
    default n

config COOVACHILLI_ENABLE_LOCATION
    bool "Enable Location Awareness"
    default n

config COOVACHILLI_ENABLE_FORCEDNS
    bool "Enable the forcing (NAT) of DNS"
    default n

choice
    prompt "SSL Library"
    default COOVACHILLI_DISABLE_SSL

config COOVACHILLI_DISABLE_SSL
    bool "Disable SSL support"

config COOVACHILLI_ENABLE_OPENSSL
    bool "Enable support for OpenSSL"

config COOVACHILLI_ENABLE_MATRIXSSL
    bool "Enable support for MatrixSSL"

config COOVACHILLI_ENABLE_CYASSL
    bool "Enable support for CyaSSL"

endchoice

config COOVACHILLI_ENABLE_MATRIXSSL_CLI
    bool "Enable MatrixSSL client use"
    depends on COOVACHILLI_ENABLE_MATRIXSSL
    default n

endmenu

```

Coova-Chilli arguments

Open the [coova-chilli-1.3.0+20141128/src/cmdline.ggo](https://github.com/Coova/Coova-Chilli-1.3.0+20141128/src/cmdline.ggo):

```

# chilli - ChilliSpot.org. A Wireless LAN Access Point Controller
# Copyright (C) 2003, 2004, 2005 Mondru AB.
# Copyright (c) 2006-2012 David Bird (Coova Technologies)
#
# The contents of this file may be used under the terms of the GNU
# General Public License Version 2, provided that the above copyright
# notice and this permission notice is included in all copies or
# substantial portions of the software.
#
# The initial developer of the original code is
# Jens Jakobsen <jj@chillispot.org>
#

args "--no-handle-help --no-handle-version"

option "fg"          f "Run in foreground"          flag off
option "debug"       d "Run in debug mode"          flag off
option "bin"         b "Binary config file" string no
option "reload"      r "Send reload after reconfigure" flag off
option "forgiving"   F "Forgiving of many configuration errors" flag off
option "debugfacility" - "Which modules to print debug messages for" int default="1" no
option "logfacility"  - "Syslog facility to be used for logging" int default="-1" no

option "conf"        c "Read configuration file"      string no
option "interval"    - "Re-read configuration file at this interval" int default="3600" no
option "pidfile"     - "Filename of process id file"  string no
option "statedir"    - "Directory of nonvolatile data" string no
option "uid"         - "UID to run as, after being started as root" int default="0" no

```

```

option "gid" - "GID to run as, after being started as root" int default="0" no

# NET/TUN parameters
option "net" n "Network" string default="192.168.182.0/24" no
option "dhcpstart" - "Network DHCP Starting IP" int default="0" no
option "dhcpend" - "Network DHCP Ending IP" int default="0" no
option "dhcpbroadcast" - "Always broadcast DHCP responses" flag off
option "dynip" - "Dynamic IP address pool" string no
option "nodynip" - "No Dynamic IP assignment" flag off
option "statip" - "Static IP address pool" string no
option "uamanyipex" - "Network to exclude from uamanyip" string no
option "uamnatanyipex" - "Network to exclude from uamnatanyip" string no
option "dns1" - "Primary DNS server IP address" string no
option "dns2" - "Secondary DNS server IP address" string no
option "domain" - "Domain to use for DNS lookups" string default="coova.org" no
option "ipup" - "Script to run after link-up" string no
option "ipdown" - "Script to run after link-down" string no
option "conup" - "Script to run after user logon" string no
option "condown" - "Script to run after user logoff" string no
option "macup" - "Script to run after initial DHCP" string no
option "macdown" - "Script to run after DHCP release" string no
option "vlanupdate" - "Script to run after 802.1Q/VLAN id assignment" string no
option "vlanlocation" - "VLAN is to be used as location" flag off
option "locationupdate" - "Script to run after change of location" string no
option "locationstopstart" - "Restart the RADIUS accounting stream with new session-id" flag off
option "locationcopycalled" - "Copy Called-Station-Id from proxy RADIUS" flag off
option "locationimmediateupdate" - "Immediately do an accounting interim update on proxy accounting" flag off
option "locationopt82" - "Use DHCP Option 82 for location" flag off

option "txqlen" - "TX Queue length for tun interface (linux only)" int default="100" no
option "tundev" - "TUN/TAP Device, as in tun0 or tap1" string no
option "mtu" - "MTU given in DHCP" int default="1500" no
option "autostatip" - "Auto- static ip assignment" int default="0" no

option "mmapring" - "Enable use of MMAP Rings (in Linux only)" flag off
option "ringsize" - "TX/RX Ring Size (in kbytes; linux only)" int default="0" no
option "sndbuf" - "SNDBUF size (in kb)" int default="0" no
option "rcvbuf" - "RCVBUF size (in kb)" int default="0" no

option "childmax" - "Maximum number of child processes" int default="128" no

option "peerid" - "Cluster peer ID" int default="0" no
option "peerkey" - "Cluster Blowfish key" string no

# Radius parameters
option "radiuslisten" - "IP address to send from" string no
option "radiusserver1" - "IP address of radius server 1" string no
option "radiusserver2" - "IP address of radius server 2" string no

# option "radiusacctserver1" - "IP address of radius server 1 for accounting" string no
# option "radiusacctserver2" - "IP address of radius server 2 for accounting" string no
# option "radiusadmserver1" - "IP address of radius server 1 for Administrative-User" string no
# option "radiusadmserver2" - "IP address of radius server 2 for Administrative-User" string no

option "radiusauthport" - "Authentication UDP port of radius server" int default="1812" no
option "radiusacctport" - "Accounting UDP port of radius server" int default="1813" no
option "radiussecret" - "Radius shared secret" string default="testing123" no

# option "radiusacctsecret" - "Radius shared secret for accounting" string no
# option "radiusadmsecret" - "Radius shared secret for Administrative-User" string no

option "radiustimeout" - "Retry timeout in seconds" int default="10" no
option "radiusretry" - "Total number of retries" int default="4" no
option "radiusretrysec" - "Number of retries before using secondary" int default="2" no
option "radiusnasid" - "Radius NAS-Identifier" string default="nas01" no
option "radiuslocationid" - "WISPr Location ID" string no
option "radiuslocationname" - "WISPr Location Name" string no
option "locationname" - "Location Name" string no
option "radiusnasporttype" - "Radius NAS-Port-Type" int default="19" no
option "coaport" - "Radius disconnect port to listen to" int default="0" no
option "coanoipcheck" - "Allow radius disconnect from any IP" flag off

```

```

option "noradallow" - "Allow all sessions when RADIUS is not available" flag off

# Radius proxy parameters
option "proxylisten" - "Proxy IP address to listen on" string no
option "proxyport" - "Proxy UDP port to listen on (0 is off)" int default="0" no
option "proxycclient" - "IP address of proxy client(s)" string no
option "proxysecret" - "Radius proxy shared secret" string no
option "proxymacaccept" - "Auto-accept non-EAP requests on proxy port" flag off
option "proxyonacct" - "Proxy through to chilli RADIUS accounting packets" flag off

option "proxylocattr" - "The attribute in proxy to be used for the loc= query string parameter" string no
multiple

# DHCP parameters
option "dhcpif" - "Local Ethernet interface" string no
option "moreif" - "Multi-LAN more interfaces" string no multiple
option "dhcpmac" - "DHCP Interface MAC Address" string no
option "dhcpmacset" - "Option to have dhcpif configured with dhcpmac" flag off
option "nexthop" - "Next Hop MAC address" string no
option "dhcpradius" - "Map certain DHCP options to RADIUS attributes" flag off
option "dhcpgateway" - "DHCP gateway addresss for relay" string no
option "dhcpgatewayport" - "DHCP gateway port for relay" int default="67" no
option "dhcprelayagent" - "DHCP relay agent IP addresss (default uamlisten)" string no
option "lease" - "Lease time to allocate to clients" int default="600" no
option "leaseplus" - "Lease grace period time before removal" int default="60" no
option "noc2c" - "Setup clients for /32 network" flag off

# EAPOL parameters
option "eapolenable" - "Enable IEEE 802.1x authentication" flag off

# UAM parameters
option "uamserver" - "URL of authentication web server" string no
option "uamhomepage" - "URL of homepage to redirect unauthenticated users to" string no
option "uamsecret" - "Shared secret between uamserver and chilli" string no
option "uamlisten" - "IP address to listen to for authentication requests" string no
option "dhcplisten" - "IP address for DHCP default gateway (defaults to uamlisten)" string no
option "uamport" - "TCP port to bind to for authentication requests" int default="3990" no
option "uamuiport" - "TCP port to bind to for UAM UI requests" int default="3991" no
option "uamallowed" - "Resources exempt from access check " string no multiple
option "uamdomain" - "Domain name allowed (active dns filtering; one per line!) " string no multiple
option "uamdomainttl" - "DNS TTL to use (rewrite) when query matches a uamdomain" int default="60" no
option "uamregex" - "Regular expression to match URLs (one per line) " string no multiple
option "nosystemdns" - "Do not attempt to use the system DNS for DHCP" flag off
option "uamanydns" - "Allow client to use any DNS server" flag off
option "uamanyip" - "Allow client to use any IP Address" flag off
option "uamnatanyip" - "Source NAT clients using anyip to an IP of dynip pool" flag off
option "wisprlogin" - "A specific WISPr login url to be used" string no
option "nouamsuccess" - "Do not return to the UAM server on success, original url instead" flag off
option "nowispr1" - "Do not offer WISPr 1.0 XML" flag off
option "nowispr2" - "Do not offer WISPr 2.0 XML" flag off
option "uamlogoutip" - "HTTP Auto-Logout IP Address" string default="1.0.0.0" no
option "uamaliasip" - "Special IP Address aliased (redirect) to uamlisten/uamport" string default="1.0.0.1" no
option "uamaliasname" - "Special simple hostname (no dots) to be resolved to uamaliasip" string no
option "uamhostname" - "Special simple hostname (no dots) to be resolved to uamlisten" string no

option "authedallowed" - "Resources exempt from session limitations" string no multiple
option "uamauthedallowed" - "Use uamallowed as resources exempt from session limitations" flag off

option "uamaaaurl" - "UAM AAA URL specifying the URL to use for the Chilli HTTP AAA" string no
option "domaindnslocal" - "Option to consider all hostnames in domain as local" flag off
option "radsec" - "Use RadSec tunnig (requires SSL; not compatible with uamaaaurl)" flag off

option "defsessiontimeout" - "Default session-timeout if not returned by RADIUS" long default="0" no
option "defidletimeout" - "Default idle-timeout if not returned by RADIUS" int default="0" no
option "defbandwidthmaxdown" - "Default WISPr-Bandwidth-Max-Down if not returned by RADIUS" long default="0" no
option "defbandwidthmaxup" - "Default WISPr-Bandwidth-Max-Up if not returned by RADIUS" long default="0" no
option "definteriminterval" - "Default interim-interval for accounting if not returned by RADIUS" int default="300" no

```

```

option "bwbucketupsize" - "Define the up-bound 'leaky bucket' size" int default="0" no
option "bwbucketdsize" - "Define the down-bound 'leaky bucket' size" int default="0" no
option "bwbucketminsize" - "Define the minimum 'leaky bucket' size" int default="0" no

# MAC authentication
option "macauth" - "Authenticate based on MAC address" flag off
option "macreauth" - "Re-Authenticate based on MAC address for every initial URL redirection" flag off
option "macauthdeny" - "Deny access (even UAM) to MAC addresses given Access-Reject" flag off
option "macallowed" - "List of allowed MAC addresses" string no multiple
option "macsuffix" - "Suffix to add to the MAC address" string no
option "macpasswd" - "Password used when performing MAC authentication" string no
option "macallowlocal" - "Do not use RADIUS for authenticating the macallowed" flag off
option "strictmacauth" - "Be strict about MAC Auth (no DHCP reply until we get RADIUS reply)" flag off
option "strictdhcp" - "Be strict about only allocating dyn-pool from DHCP" flag off

# "local" content
option "wwwdir" - "Local content served by chilli (for splash page, etc)" string no
option "wwwbin" - "Script binary (such as haserl) for simple web programming" string no
option "uamui" - "Program in inetd style to handle all uam requests" string no

# Centralized Configuration
option "adminuser" - "RADIUS administrative user login username" string no
option "adminpasswd" - "RADIUS administrative user login password" string no
option "adminupdatefile" - "File for administrative user ChilliSpot-Config settings" string no
option "rtmonfile" - "File to update with routing settings" string no

option "ethers" - "File containing a mapping of MAC addresses to static IP addresses" string no

# "Location-Aware"
option "nasmac" - "Unique MAC address of the NAS (called-station-id)" string no
option "nasip" - "Unique IP address of the NAS (nas-ip-address)" string no
option "ssid" - "SSID of the session" string no
option "vlan" - "VLAN of the session" string no
option "ieee8021q" - "Support 802.1Q VLAN tagging" flag off
option "only8021q" - "Support 802.1Q VLAN tagged traffic only" flag off

# Command Socket Support
option "cmdsocket" - "path to the command unix socket" string no
option "cmdsocketport" - "Port of command socket. Only used if cmdsocket is not defined." int default="42424" no

option "radiusoriginalurl" - "Turn on the sending of ChilliSpot-OriginalURL in Access-Request" flag off
option "swapoctets" - "Swap the meaning of input/output octets/packets" flag off
option "usestatusfile" - "Use the status file to keep track of sessions" string no
option "statusfilesave" - "Aggressively save the status of sessions to status file" flag off
option "localusers" - "File keep 'Local' usernames and passwords" string no
option "postauthproxy" - "IP of an upstream transparent proxy" string no
option "postauthproxyport" - "Port of an upstream transparent proxy" int default="0" no
option "postauthproxyssl" - "Send SSL to upstream transparent proxy" flag off
option "wpaguests" - "Allow WPA 'Guest' access" flag off
option "openidauth" - "Allow OpenID authentication" flag off
option "papalwaysok" - "Always allow 'PAP' authentication (deprecated; always on)" flag off
option "mschapv2" - "Use MSCHAPv2 authentication where possible" flag off
option "chillixml" - "Use ChilliSpot XML in WISPr blocks" flag off
option "acctupdate" - "Allow updating of session attributes in Accounting-Response" flag off
option "dnsparanoia" - "Inspect DNS packets and drop responses with any non- A, CNAME, SOA, or MX records (to prevent dns tunnels)" flag off
option "seskeepalive" - "Keep sessions 'alive' after a restart of the server" flag off
option "wpadpacfile" - "WPAD PAC file location" string no

option "usetap" - "Use a TAP instead of TUN (linux only)" flag off
option "noarpentries" - "Do not create arp table entries in when using TAP." flag off
option "routeif" - "Turns on 'multi-routing' and defines default route" string no
option "framedservice" - "Use Service-Type = Framed instead of Login" flag off
option "tcpwin" - "Change TCP window size to this value to help prevent congestion" int default="0" no
option "scalewin" - "Scale the TCP window when bandwidth shaping" flag off
option "tcpmss" - "Change TCP maximum window size (mss) option in TCP traffic" int default="0" no
option "maxclients" - "Maximum number of clients/subscribers" int default="512" no
option "dhcphashsize" - "Size of DHCP/MAC hash table" int default="56" no
option "radiusqsize" - "Size of RADIUS queue table" int default="0" no

```

```

option "challenge timeout" - "Timeout in seconds for the generated challenge" int default="600" no
option "challenge timeout2" - "Timeout in seconds for challenge during login" int default="1200" no

option "redir" - "Enable redir (redirection) daemon" flag off
option "inject" - "Enable redir injection" string no
option "injectext" - "Enable redir injection extended script" string no
option "injectwispr" - "Enable redir injection of WISPr" flag off

option "redirurl" - "Send redirection URL in UAM parameters instead of HTTP redirect" flag off

option "routeonetone" - "When using routeif, do one-to-one NAT" flag off

option "nouser gardendata" - "Do not consider walled garden for authorized session accounting" flag off
option "uamgardendata" - "Enable a RADIUS accounting session for walled garden data based on IP lease life-cycle" flag off
option "uamotherdata" - "Enable accounting for dropped 'Other' of IP session" flag off

option "sslkeyfile" - "SSL private key file in PEM format" string no
option "sslkeypass" - "SSL private key password" string no
option "sslcertfile" - "SSL certificate file in PEM format" string no
option "sslcafile" - "SSL CA certificate file in PEM format" string no
option "unixipc" - "The UNIX IPC Filename to use when compiled with --with-unixipc" string no
option "uamallowpost" - "Enable to allow a HTTP POST to the standard uamport interface" flag off
option "natip" - "IP to use when doing nat on WAN (routeidx)" string no
option "natport" - "Port to use when doing nat on the WAN (routeidx)" int default="0" no
option "redirssl" - "Enable redirection of SSL/HTTP port (requires SSL support)" flag off
option "uamuissl" - "Enable SSL/HTTPS support on the uamuiport" flag off

option "dnslog" - "Log DNS requests to a file." string no
option "ipwhitelist" - "Binary IP White List file" string no
option "uamdomainfile" - "Load uamdomains (regex) from file" string no

option "layer3" - "Layer3 only" flag off
option "ipsrclallowed" - "List of allowed source IP addresses, for Layer3" string no multiple
option "patricia" - "Use patricia tries for walled garden lookup" flag off
option "redirdnsreq" - "Send DNS query on redirect to pick of DNS based walled garden" flag off

option "kname" - "Enable the use of the coova kernel module instance of this namem" string no
option "moddir" - "Directory for dynamically loaded modules" string no
option "module" - "Dynamically loaded module" string no multiple

option "dhcpcpt" - "Set a DHCP option using a hex string" string no multiple

option "extadmvsas" - "Extended administrative-user VSA script support" string no multiple

option "dhcpcnotidle" - "DHCP counted for preventing idle-timeout" flag off

option "forcedns1" - "Force all DNS to a specific address" string no
option "forcedns1port" - "Force all DNS to a specific port" int default="0" no

option "forcedns2" - "Force all secondary DNS to a specific address" string no
option "forcedns2port" - "Force all secondary DNS to a specific port" int default="0" no

option "ipv6" - "Enable IPv6 support" flag off
option "ipv6mode" - "IPv6 mode is either 6and4 (default), 4to6, or 6to4" string no
option "ipv6only" - "Enable IPv6-Only" flag off

```

Check the boolean settings section in *package/feeds/packages/coova-chilli/files/chilli.init* with the content of the previous file.

All the variables with the mention **flag on** or **flag off** match the boolean settings section.

```

option "dhcpcbroadcast" - "Always broadcast DHCP responses" flag off
option "dynip" - "Dynamic IP address pool" string no
option "nodynip" - "No Dynamic IP assignment" flag off

```

dhcpcbroadcast and **nodynip** are attributes of boolean settings while **dynip** is not.

Here is the *chilli.init* who matches the *coova-chilli-1.3.0+20141128*:

```

#!/bin/sh /etc/rc.common

START=30
STOP=90

config_cb() {
    chilli_inst=$2
    if [ "$chilli_inst" != "" ]
    then
        rm -f /var/run/chilli_${chilli_inst}.*
        chilli_conf=/var/run/chilli_${chilli_inst}.conf
        eval "start_chilli_${chilli_inst}=1"
    fi
}

option_cb() {
    case "$1" in
        # UCI settings
            network)
                . /lib/functions/network.sh
                local ifname
                network_get_device ifname $2
                echo "dhcpif=\"${ifname}\"" >> $chilli_conf
                ;;
            disabled)
                eval "start_chilli_${chilli_inst}=0"
                ;;
        # boolean settings
            dhcpbroadcast|nodynip|vlanlocation|locationstopstart|
            locationcopycalled|locationimmediateupdate|locationopt82|mmapstring|
            coanoipcheck|noradallow|proxymacaccept|proxyonacct|
            dhcpmacset|dhcpradius|noc2c|eapolenable|nosystemdns|
            uamanydns|uamanyip|uamnatanyip|nouamsuccess|nowispr1|nowispr2|
            uamauthedallowed|domaindnslocal|radsec|macauth|macreauth|
            macauthdeny|macallowlocal|strictmacauth|strictdhcp|ieee8021q|only8021q|
            radiusoriginalurl|swapoctets|statusfilesave|postauthproxyssl|wpaguests|
            openidauth|papalwaysok|mschapv2|chillixml|acctupdate|dnssparanoia|
            seskeepalive|usetap|noarpentries|framedservice|scalewin|redir|
            injectwispr|redirurl|routeonetone|nousergardendata|uamgardendata|
            uamotherdata|uamallowpost|redirssl|uamuissl|layer3|patricia|redirdnsreq|
            dhcpnotidle|ipv6|ipv6only)
                [ "$2" = "true" -o "$2" = "1" ] && echo "$1" >> $chilli_conf
                ;;
            *)
                echo "$1=\"${2}\"" >> $chilli_conf
                ;;
    esac
}

start_chilli() {
    local cfg="$1"
    local start_chilli=$(eval "echo \${start_chilli}_$cfg")
    [ "$start_chilli" = "0" ] && return
    local base=/var/run/chilli_${cfg}
    chilli -c ${base}.conf \
        --pidfile ${base}.pid \
        --cmdsocket ${base}.sock \
        --unixipc ${base}.ipc &
}

start() {
    config_load chilli
    config_foreach start_chilli chilli
}

stop() {
    for pID in $( pgrep chilli )
    do
        kill $pID
    done
}

```

```
done
rm -f /var/run/chilli*
}
```

Reference

<https://github.com/coova/coova-chilli/issues/200>

<https://github.com/coova/coova-chilli/issues/263>