```
************************************************************
    8319 Fri Jul 12 15:14:15 2019
new/usr/src/cmd/svc/milestone/net-routing-setup
want to add link-local routes first
************************************************************
```

```
#!/sbin/sh
#
# CDDL HEADER START
#
# The contents of this file are subject to the terms of the
# Common Development and Distribution License (the "License").
# You may not use this file except in compliance with the License.
#
# You can obtain a copy of the license at usr/src/OPENSOLARIS.LICENSE
# or http://www.opensolaris.org/os/licensing.
# See the License for the specific language governing permissions
# and limitations under the License.
#
# When distributing Covered Code, include this CDDL HEADER in each
# file and include the License file at usr/src/OPENSOLARIS.LICENSE.
# If applicable, add the following below this CDDL HEADER, with the
# fields enclosed by brackets "[]" replaced with your own identifying
# information: Portions Copyright [yyyy] [name of copyright owner]
#
# CDDL HEADER END
#
#
# Copyright (c) 1999, 2010, Oracle and/or its affiliates. All rights reserved.
#
# Copyright (c) 2012 Joyent, Inc.  All rights reserved.

# This script configures IP routing.

. /lib/svc/share/smf_include.sh

set -o xtrace

#
# In a shared-IP zone we need this service to be up, but all of the work
# it tries to do is irrelevant (and will actually lead to the service
# failing if we try to do it), so just bail out.
# In the global zone and exclusive-IP zones we proceed.
#
smf_configure_ip || exit $SMF_EXIT_OK

#
# If routing.conf file is in place, and has not already been read in
# by previous invokation of routeadm, legacy configuration is upgraded
# by this call to "routeadm -u".  This call is also needed when
# a /var/svc/profile/upgrade file is found, as it may contain routeadm commands
# which need to be applied.  Finally, routeadm starts in.ndpd by
# enabling the ndp service (in.ndpd), which is required for IPv6 address
# autoconfiguration. It would be nice if we could do this in
# network/loopback, but since the SMF backend is read-only at that
# point in boot, we cannot.
#
/sbin/routeadm -u

#
# Are we routing dynamically? routeadm(1M) reports this in the
# "current" values of ipv4/6-routing - if either are true, we are running
# routing daemons (or at least they are enabled to run).
#
dynamic_routing_test=‘/sbin/routeadm -p | \
nawk ’/^ipv[46]-routing [.]*/ { print $2 }’  | /usr/bin/grep "current=enabled"‘
if [ -n "$dynamic_routing_test" ]; then
```



```
        dynamic_routing="true"
fi

#
# Configure default IPv4 routers using the local "/etc/defaultrouter"
# configuration file.  The file can contain the hostnames or IP
# addresses of one or more default routers.  If hostnames are used,
# each hostname must also be listed in the local "/etc/hosts" file
# because NIS is not running at the time that this script is
# run.  Each router name or address is listed on a single line by
# itself in the file.  Anything else on that line after the router’s
# name or address is ignored.  Lines that begin with "#" are
# considered comments and ignored.
#
# The default routes listed in the "/etc/defaultrouter" file will
# replace those added by the kernel during diskless booting.  An
# empty "/etc/defaultrouter" file will cause the default route
# added by the kernel to be deleted.
#
# Note that the default router file is ignored if we received routes
# from a DHCP server.  Our policy is to always trust DHCP over local
# administration.
#
smf_netstrategy

#
# Read /etc/inet/static_routes.vmadm and add each route.
#
if [ -f /etc/inet/static_routes.vmadm ]; then
        echo "Adding vmadm persistent link-local routes:"
        /usr/bin/egrep -v "^(#|$)"  /etc/inet/static_routes.vmadm | /usr/bin/egr
                /usr/sbin/route add $line
        done
fi

if [ "$_INIT_NET_STRATEGY" = "dhcp" ] && \
    [ -n "‘/sbin/dhcpinfo Router‘" ]; then
        defrouters=‘/sbin/dhcpinfo Router‘
elif [ -f /etc/defaultrouter ]; then
        defrouters=‘/usr/bin/grep -v \^\# /etc/defaultrouter | \
            /usr/bin/awk ’{print $1}’‘
        if [ -n "$defrouters" ]; then
                #
                # We want the default router(s) listed in
                # /etc/defaultrouter to replace the one added from the
                # BOOTPARAMS WHOAMI response but we must avoid flushing
                # the last route between the running system and its
                # /usr file system.
                #

                # First, remember the original route.
                shift $#
                set -- ‘/usr/bin/netstat -rn -f inet | \
                    /usr/bin/grep ’^default’‘
                route_IP="$2"

                #
                # Next, add those from /etc/defaultrouter.  While doing
                # this, if one of the routes we add is for the route
                # previously added as a result of the BOOTPARAMS
                # response, we will see a message of the form:
                #       "add net default: gateway a.b.c.d: entry exists"
                #
                do_delete=yes
                for router in $defrouters; do
                        route_added=‘/usr/sbin/route -n add default \
```

```
                                    -gateway $router‘
                                res=$?
                                set -- $route_added
                                [ $res -ne 0 -a "$5" = "$route_IP:" ] && do_delete=no
                        done

                        #
                        # Finally, delete the original default route unless it
                        # was also listed in the defaultrouter file.
                        #
                        if [ -n "$route_IP" -a $do_delete = yes ]; then
                                /usr/sbin/route -n delete default \
                                    -gateway $route_IP >/dev/null
                        fi
                else
                        /usr/sbin/route -fn > /dev/null
                fi
else
        defrouters=
fi

#
# Use routeadm(1M) to configure forwarding and launch routing daemons
# for IPv4 and IPv6 based on preset values.  These settings only apply
# to the global zone.  For IPv4 dynamic routing, the system will default
# to disabled if a default route was previously added via BOOTP, DHCP,
# or the /etc/defaultrouter file.  routeadm also starts in.ndpd.
#
if [ "$dynamic_routing" != "true"  ] && [ -z "$defrouters" ]; then
        #
        # No default routes were setup by "route" command above.
        # Check the kernel routing table for any other default
        # routes.
        #
        /usr/bin/netstat -rn -f inet | \
            /usr/bin/grep default >/dev/null 2>&1 && defrouters=yes
fi

#
# The routeadm/ipv4-routing-set property is true if the administrator
# has run "routeadm -e/-d ipv4-routing".  If not, we revert to the
# appropriate defaults.  We no longer run "routeadm -u" on every boot
# however, as persistent daemon state is now controlled by SMF.
#
ipv4_routing_set=‘/usr/bin/svcprop -p routeadm/ipv4-routing-set $SMF_FMRI‘
smartos_param=‘/usr/bin/bootparams | grep "^smartos"‘
if [ -z "$defrouters" ] && [ "$smartos_param" != "" ]; then
        #
        # Set default value for ipv4-routing to enabled.  If routeadm -e/-d
        # has not yet been run by the administrator, we apply this default.
        # The -b option is project-private and informs routeadm not
        # to treat the enable as administrator-driven.
        #
        /usr/sbin/svccfg -s $SMF_FMRI \
            setprop routeadm/default-ipv4-routing = true
        if [ "$ipv4_routing_set" = "false" ]; then
                /sbin/routeadm -b -e ipv4-routing -u
        fi
else
        #
        # Default router(s) have been found,  so ipv4-routing default value
        # should be disabled.  If routaedm -e/d has not yet been run by
        # the administrator, we apply this default.  The -b option is
        # project-private and informs routeadm not to treat the disable as
        # administrator-driven.
        #
```



```
        /usr/sbin/svccfg -s $SMF_FMRI \
            setprop routeadm/default-ipv4-routing = false
        if [ "$ipv4_routing_set" = "false" ]; then
                /sbin/routeadm -b -d ipv4-routing -u
        fi
fi

#
# See if static routes were created by install. If so, they were created
# under /etc/svc/volatile. Copy them into their proper place.
#
if [ -f /etc/svc/volatile/etc/inet/static_routes ]; then
        echo "Installing persistent routes"
        if [ -f /etc/inet/static_routes ]; then
                cat /etc/svc/volatile/etc/inet/static_routes | grep -v ’^#’ \
                    >> /etc/inet/static_routes
        else
                cp /etc/svc/volatile/etc/inet/static_routes \
                    /etc/inet/static_routes
        fi
        /usr/bin/rm /etc/svc/volatile/etc/inet/static_routes

fi

#
# Read /etc/inet/static_routes and add each route.
#
if [ -f /etc/inet/static_routes ]; then
        echo "Adding persistent routes:"
        /usr/bin/egrep -v "^(#|$)" /etc/inet/static_routes | while read line; do
                /usr/sbin/route add $line
        done
fi

#
# Read /etc/inet/static_routes.vmadm and add each non-link-local route.
# Read /etc/inet/static_routes.vmadm and add each route.
#
if [ -f /etc/inet/static_routes.vmadm ]; then
        echo "Adding vmadm persistent routes:"
        /usr/bin/egrep -v "^(#|$)" /etc/inet/static_routes.vmadm | /usr/bin/egre
        /usr/bin/egrep -v "^(#|$)" /etc/inet/static_routes.vmadm | while read li
                /usr/sbin/route add $line
        done
fi

#
# Log the result
#
echo "Routing setup complete:"
/usr/bin/netstat -rn

# Clear exit status.
exit $SMF_EXIT_OK
```