Linux 下移动 IPv6 实现指南

Lars Strand lars@unik.no

2003 年 7 月 30 日

内容

1 介绍

本文档描述了在 Linux 平台下的软件和建立与使用移动 IPv6 的步骤。draft-ieft.mipv6-ipv6”回答了移动 IP 的概念与使用它的原因。

1.1 移动 IP 的概念

每个移动节点总是由其家乡地址识别，而于其当前接入 Internet 位置无关。当离开家乡链路时，移动节点也与一个转交地址相关联，该转交地址包含了移动节点当前位置的信息。寻址到移动节点家乡地址的 IPv6 报文分组经过其家乡代理（HA）被透明的路由到其转交地址。该协议使 IPv6 节点能够缓存移动节点家乡地址和其转交地址的绑定，然后在直接发送目的为移动节点的所有报文分组到该转交地址。

1.2 为何使用移动 IP?

若在 IPv6 中没有对移动性的特定支持，因为路由是根据报文分组目的 IP 地址中的子网前缀进行路由，则当移动节点离开其家乡链路（其家乡 IPv6 子网前缀使用的链路）时，发送给移动节点（主机或路由器）的报文分组不能够到达。为保证移动情况下的持续通信，移动节点在每次移动到一个新链路时都能改变其 IP 地址，但移动节点在改变位置时就无法保持传输层和更高层的连接。因为移动计算机在 IPv6 部署期间可能占 Internet 上通信设备的大多数或起码是可观比例，所以 IPv6 移动性支持尤为重要。

全部细节，参见移动 IPv6 移动性支持（RFC3775）（见“资源”部分）。

1.3 工作机制

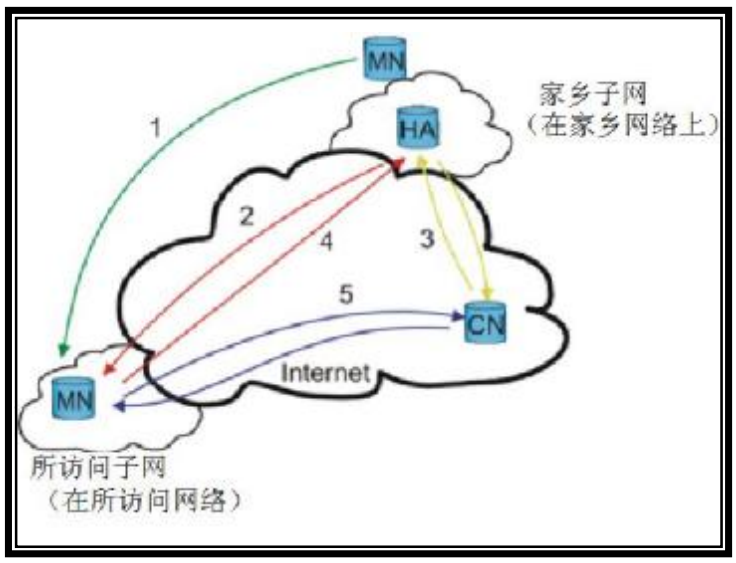


图 1：移动 IP

1. 移动节点（MN）到达外地网络并得到一个新转交地址。
2. MN 执行与其家乡代理（HA）的绑定更新（新转交地址在 HA 那里注册）。HA 发送一条绑定确认给 MN。
3. 通信对端（CR）要联系 MN。HA 截获目的是 MN 的报文分组。
4. 然后 HA 使用 MN 的转交地址从 CR 通过隧道发送所有报文分组到 MN。

5. 当 MN 应答 CR 时，它可以使用其当前的转交地址（执行与 CR 的绑定）并与 CR 直接通信（优化路由）或者通过隧道经过 HA 传输所有报文分组。

解释如图 1 所示意。

2 IPv6

IP 版本 6（IPv6）是一个新版本的 Internet 协议，作为 IP 版本 4（IPv4）[RFC-791]的后续版本而设计。IPv6 与 IPv4 相比主要有以下变化：

扩展的寻址功能

首部格式简化

对扩展与选项的改善支持

流标签功能

认证与保密功能

为充分理解 MIPv6 的工作原理，您应当对 IPv6 无状态自动配置有基本了解。您可对 RFC2462 中的 IPv6 无状态地址自动配置进行研究。

通常需了解 IPv6 更多信息，请访问 IPv6 工作组（IETF）网站-见链接的资源部分。

3 Linux 下的 MIPv6 实现

目前有两个可用的 Linux 下移动 IPv6 实现的版本。英国的兰开夏大学有最早（?）的实现(http://www.c8-ipv6.lancs.ac.uk/MobileIP/)。所支持的最新内核版本为 2.1.90，与 IETF 移动 IPv6 草案第五版（目前是第 24 版）相兼容。代码与站点从 1998 开始就一直未更新，因此它被认为是过时的。

另外一个不断更新的实现由 HUT（赫尔辛基技术大学）开发。所支持的最新内核版本是 2.6.8.1。需要文档与软件请访问 http://www.mipl.mediapoli.com/或浏览邮件压缩文档。

3.1 给内核打补丁

HUT MIPv6 需要内核补丁。该实现对 IPv6 内核栈作了修改，因此需要重新编译内核。实现软件包中有很好的安装过程帮助文档，但我会给出一个简要的指南。

请注意！MN 和 HA 不再需要两个不同的内核。只需对一个内核进行编译以提供对 MN 与 HA 的支持。不可能同时作为 MN 和 HA 运行；而是根据所加载的模块来进行选择。

1. 从 http://www.mobile-ipv6.org 下载最新的 Linux MIPv6 源代码。目前最新发布的版本是：mipv6-2.0-rc2-linux-2.6.8.1。考虑到稳定性问题，我们使用 mipv6-1.1-v2.4.26。最后 4 个数字对应于应使用补丁的 Linux 内核版本。

```
# cd /usr/local/src
# wget http://www.mobile-ipv6.org/download/mipv6-1.1-v2.4.26.tar.gz
# tar zxfv mipv6-1.1-v2.4.26.tar.gz
```

2.从 ftp.kernel.org 下载并解压对应的 Linux 内核版本：

```
# cd /usr/src
# wget ftp://ftp.kernel.org/pub/linux/kernel/v2.4/linux-2.4.26.tar.bz2
# tar jxvf linux-2.4.26.tar.bz2
```

```
# ln -s linux-2.4.26 linux
# cd linux
```

3.应用 MIPv6 补丁：

```
# patch -p1 --dry-run < /usr/local/src/mipv6-1.1-v2.4.26/mipv6-1.1-v2.4.26.patch
```

--dry-run 检查补丁能否正确应用。若返回任何错误结果，则不应继续进行。若一切正常，则

```
# patch -p1 < /usr/local/src/mipv6-1.0-v2.4.22/mipv6-1.1-v2.4.26.patch
```

4．删除源代码中残留的.o 文件以及其它从属文件。

```
#make mrproper
```

5.现在开始对内核树进行配置。运行 **make menuconfig。**MIPv6 选项在“Networking Options”内。以下选项应当出现。

```
CONFIG_EXPERIMENTAL=y
CONFIG_SYSCTL=y
CONFIG_PROC_FS=y
CONFIG_MODULES=y
CONFIG_NET=y
CONFIG_NETFILTER=y
CONFIG_UNIX=y
CONFIG_INET=y
CONFIG_IPV6=m
CONFIG_IPV6_SUBTREES=y
CONFIG_IPV6_IPV6_TUNNEL=m
CONFIG_IPV6_MOBILITY=m
CONFIG_IPV6_MOBILITY_MN=m
CONFIG_IPV6_MOBILITY_HA=m
```

因为 MIPL 开发工作还在进行中，所以您可能设置：

```
CONFIG_IPV6_MOBILITY_DEBUG=y
```

使用调试信息，更易于发现问题所在。调试信息也非常有助于报告 bug。

为确保开启了所有正确选项，您可运行包含在 MIPL 中的一个小的 shell 脚本：**chkconf_kernel.sh**

6.接下来您应当编译并安装内核。

提示：为更容易的将该内核与其它内核相区分，您可以改变**/usr/src/linux/Makefile** 中的**"EXTRAVERSION"** 变量，比如**"-MIPv6-1"**。

```
#make dep
#make bzImage
#make clean
#make modules
#make modules_install //生成了/lib/modules/2.4.26
```

7．把 **bzImage** 复制到 **boot** 目录并在**/boot** 下生成 **initrd img** 文件：

```
#cp /usr/src/linux/arch/i386/boot/bzImage /boot
#cd /boot
#mkinitrd initrd-2.4.26.img 2.4.26
```

8.修改 **grub** 设置：

```
#vi /etc/grub.conf
```

添加以下几行：

```
title Red Hat Linux (test)
root (hd0,2)
```

kernel /boot/bzImage ro root=/dev/hda3（根据实际情况）
initrd /boot/initrd-2.4.26.img

3.2 用户空间工具

用户空间工具 **mipdiag**，配置文件和初始化脚本必须作为模块安装才能正确运行：

```
# cd /usr/local/src/mipv6-1.1-v2.4.26
# ./configure
# make && make install
```

3.3 MIPv6 设备节点

MIPv6 模块也需要一个新的设备节点表项。执行命令：

```
# mknod /dev/mipv6_dev c 0xf9 0
```

3.4 自动启动

1. *Red Hat:*

所有初始化脚本都位于/etc/init.d/，它们符号链接到正确的运行级（/etc/rcX.d/），您可执行命令：

```
# chkconfig --add mobile-ip6
```

以使 MIPv6 随系统启动，或

```
# chkconfig --del mobile-ip6
```

使 MIPv6 不随系统启动。

4 实验床

现在您应该有了一个打上 MIPL 补丁的可正常运行的内核，安装了用户级工具并启用了引导时的自动启动。若出现任何错误，仔细对照以上各部分。

4.1 实验场景

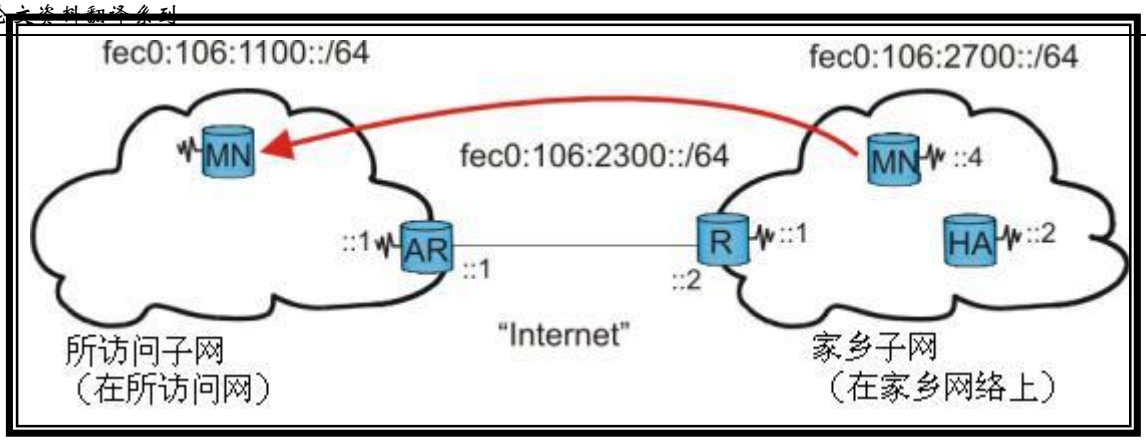
我们在实验床中使用站点本地地址。您也可使用全球地址，但一定要*注意链路本地地址无效！*实验床包含 4 个节点；见图“移动 IPv6 实验床”。

1.*HA-家乡代理：*HA 位于家乡网络，地址为 **fec0:106:2700::2**，有一个无线网络网卡。

2. *MN-移动节点：*当 MN 在“家乡网络”时，其地址为 **fec0:106:2700::4**。当 MN 移动到另一个网络时，它获得一个新的“转交”地址。

3.*R-Router：*这是位于家乡网络和 Internet 之间的路由器。它有一个无线网卡，地址为 **fec0:106:2700::1** 和一个有线网卡，地址为 **fec0:106:2300::2**。

*4.AR-Access Router:*AR 和 R 之间的链路是我们的“internet”-但在该实验场景中只是一根网线（可以是任何网络）。AR 有两个网卡；有线网卡地址为 **fec0:106:2300::1**，无线网卡地址为 **fec0:106:1100::1**。



移动 IPv6 实验床

4.2 配置过程详解

4.2.1 搭建功能完备的 IPv6 网络

在我们开始测试移动 IP 以前，需要一个功能完备的 IPv6 网络。所有节点之间应能相互 ping 通。*这是关键部分*。比如，若 AR 无法 ping 通 HA，就没有绑定更新。

我会给出搭建网络的方法和使用 IPv6 运行的简单步骤。搭建 IPv6 网络的更多信息，参见 Peter Bieringer 的 Linux IPv6 HOWTO。

为简单起见，我禁用了加密功能-*注意：在与无线网络打交道时，您应总使用加密！也请注意不同的无线网络有不同的 ESSID！*

1.*MN*：移动节点有一个无线网卡。应禁用转发功能，但应接受自动配置与路由广播：

```
# iwconfig eth0 mode ad-hoc essid homenet enc off
# ifconfig eth0 inet6 add fec0:106:2700::4/64
# echo "0" > /proc/sys/net/ipv6/conf/eth0/forwarding
# echo "1" > /proc/sys/net/ipv6/conf/eth0/autoconf
# echo "1" > /proc/sys/net/ipv6/conf/eth0/accept_ra
# echo "1" > /proc/sys/net/ipv6/conf/eth0/accept_redirects
# /etc/init.d/mobile-ip6 start
```

2.*HA*：家乡代理有一个无线网卡。应启用转发，因为它使用正常路由来将截获的报文分组从物理网卡发送到虚拟隧道网卡。*注意：必须添加缺省路由，否则在与位于外地 LAN 上的 MN 通信时会出现问题。一个可能的解决方法是使用 HA 作为家乡网络的缺省路由。*

```
# iwconfig eth0 mode ad-hoc essid homenet enc off
# ifconfig eth0 inet6 add fec0:106:2700::2/64
# echo "1" > /proc/sys/net/ipv6/conf/eth0/forwarding
# echo "0" > /proc/sys/net/ipv6/conf/eth0/autoconf
# echo "0" > /proc/sys/net/ipv6/conf/eth0/accept_ra
# echo "0" > /proc/sys/net/ipv6/conf/eth0/accept_redirects
# ip route add ::/0 via fec0:106:2700::1
# /etc/init.d/mobile-ip6 start
```

3.*R*(家乡)路由器有两个网卡；一个无线与一个有线。路由器必须启用转发功能。

```
# ifconfig eth0 inet6 add fec0:106:2300::2/64
# iwconfig eth1 mode ad-hoc essid homenet enc off
```

```
# ifconfig eth1 inet6 add fec0:106:2700::1/64
# echo "1" > /proc/sys/net/ipv6/conf/all/forwarding
# echo "0" > /proc/sys/net/ipv6/conf/all/autoconf
# echo "0" > /proc/sys/net/ipv6/conf/all/accept_ra
# echo "0" > /proc/sys/net/ipv6/conf/all/accept_redirects
# ip route add fec0:106:1100::/64 via fec0:106:2300::1
```

4.*AR*:接入路由器（位于外地链路上）也有两个网卡；一个无线和一个有线。必须启用转发功能。

```
# ifconfig eth0 inet6 add fec0:106:2300::1/64
# iwconfig eth1 mode ad-hoc essid visitnet enc off
# ifconfig eth1 inet6 add fec0:106:1100::1/64
# echo "1" > /proc/sys/net/ipv6/conf/all/forwarding
# echo "0" > /proc/sys/net/ipv6/conf/all/autoconf
# echo "0" > /proc/sys/net/ipv6/conf/all/accept_ra
# echo "0" > /proc/sys/net/ipv6/conf/all/accept_redirects
# ip route add fec0:106:2700::/64 via fec0:106:2300::2
```

您可不修改 proc 变量，而使用 sysctl。

注意：我们在实验床设置了静态路由。所有主机之间应该都可以相互 ping 通。

4.2.2 配置移动 IPv6

MIPv6 设置中最后要在 network-mip6.conf 中进行配置。RedHat 系统中，该文件位于/etc/sysconfig/。该文件内容应非常容易理解。

1.*HA:*HA 配置文件应包含以下设置：# cat /etc/network-mip6.conf

```
# Home Agent configuration file
FUNCTIONALITY=ha
DEBUGLEVEL=1
MIN_TUNNEL_NR=1
MAX_TUNNEL_NR=5
TUNNEL_SITELOCAL=yes
```

2.*MN:*MN 配置文件看起来应该像这样：

```
# cat /etc/network-mip6.conf

# Mobile Node configuration file
FUNCTIONALITY=mn
DEBUGLEVEL=1
TUNNEL_SITELOCAL=yes
MIN_TUNNEL_NR=1
MAX_TUNNEL_NR=3
HOMEDEV=mip6mnha1
HOMEADDRESS=fec0:106:2700::4/64    # MN's home adress
HOMEAGENT=fec0:106:2700::2/64      # HA's address
```

3.接着，启动移动 IP：

```
# /etc/init.d/mobile-ip6 start
Starting Mobile IPv6: OK
```

您通过在 HA 上执行 **ifconfig** 命令来验证其是否启动。若建立了隧道，ip6tnl1，则启动了移动 ip6：

```
# ifconfig
eth1      Link encap:Ethernet  HWaddr 00:02:2D:2D:DE:79
          inet6 addr: fec0:106:2700::2/64 Scope:Site
          inet6 addr: fe80::202:2dff:fe2d:de79/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:618 errors:6 dropped:6 overruns:0 frame:6
          TX packets:1485 errors:22 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:87914 (85.8 KiB)  TX bytes:252596 (246.6 KiB)
          Interrupt:3 Base address:0x100

ip6tnl1   Link encap:UNSPEC HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00 ①
          UP POINTOPOINT RUNNING NOARP  MTU:1460  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:576 (576.0 b)  TX bytes:624 (624.0 b)

ip6tnl2   Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00②
          UP RUNNING NOARP  MTU:1460  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:560 (560.0 b)  TX bytes:560 (560.0 b)
```

①隧道建立，准备连接。

②另外一条隧道就绪。

您也会看到 mipv6 模块被加载（MN）。

```
# lsmod
Module                  Size  Used by    Not tainted
mip6_mn                59888   0   (unused)
ipv6_tunnel            11448   1   [mip6_mn]
mip6_base              40728   0   [mip6_mn]
ipv6                  179764  -1   [mip6_mn ipv6_tunnel mip6_base]
...
```

4.2.3 在 AR 上配置 radvd

当 MN 来到一个新网络时，它进行链路本地地址配置，若成功则进入下一阶段。[RFC2462]（IPv6 无状态地址自动配置）描述了下一阶段：

“自动配置的下一阶段包括获得一条路由广播或确定没有路由器的存在。若存在路由器，则它们会发送具体规定主机应进行何种自动配置的路由器广播。若路由器不存在，则应当触发有状态自动配置。”

“路由器定期发送路由器广播，但连续广播之间的延迟通常比执行自动配置的主机愿意等待的时间更长。为快速获得一条广播，主机向全路由器组播组发送一条或多条路由器请求”。—第 8 页。

这是我们使用 RADVD 所在。

与 IPv6 无状态地址自动配置相关的更多细节请阅读[RFC2462]。

我们在 AR 的无线网卡上配置 RADVD。radvd.conf 文件应包含以下内容：

```
# cat /etc/radvd.conf
interface eth1
{
AdvSendAdvert on;
AdvIntervalOpt on;

MinRtrAdvInterval 3;
MaxRtrAdvInterval 10;
AdvHomeAgentFlag off;

prefix fec0:106:1100::/64
{
AdvOnLink on;
AdvAutonomous on;
AdvRouterAddr on;
};
};
```

然后我们启动它：

```
# /etc/init.d/radvd start
```

您现在应可使用 **radvdump** 命令来确定所有 radvd 报文都正确的定期发送：

```
# radvdump
Router advertisement from fe80::202:2dff:fe54:d1b2 (hoplimit 255)
Received by interface eth1
# Note: {Min,Max}RtrAdvInterval cannot be obtained with radvdump
AdvCurHopLimit: 64
AdvManagedFlag: off
AdvOtherConfigFlag: off
AdvHomeAgentFlag: off
AdvReachableTime: 0
AdvRetransTimer: 0
Prefix fec0:106:1100::/64
AdvValidLifetime: 2592000
AdvPreferredLifetime: 604800
AdvOnLink: on
```

AdvAutonomous: on
AdvRouterAddr: off
AdvSourceLLAddress: 00 02 2D 54 D1 B2

注意！当在 HA 上使用 radvd 并启用“autoconf”（在 proc 中）时，除静态地址以外，您也会在 MN 上得到一个自动生成的 IPv6 地址：

4.2.4 在 HA 上配置 radvd

为使 MN 能知道其位于家乡网络，HA 也应发出 RA。因此我们应该也在 HA 上启用 RADVD。/etc/radvd.conf 文件应包含以下内容：

```
# cat /etc/radvd.conf
interface eth0
{
AdvSendAdvert on;
MaxRtrAdvInterval 3;
MinRtrAdvInterval 1;
AdvIntervalOpt off;
AdvHomeAgentFlag on;
HomeAgentLifetime 10000;
HomeAgentPreference 20;
AdvHomeAgentInfo on;
prefix fec0:106:2700::2/64
{
AdvRouterAddr on;
AdvOnLink on;
AdvAutonomous on;
AdvPreferredLifetime 10000;
AdvValidLifetime 12000;
};
};
```

也应在 HA 上执行 **radvdump** 命令以检查是否发送了 radvd 报文：

```
# radvdump
    Router advertisement from fe80::202:2dff:fe54:d11e (hoplimit 255)
    Received by interface eth0
        # Note: {Min,Max}RtrAdvInterval cannot be obtained with radvdump
        AdvCurHopLimit: 64
        AdvManagedFlag: off
        AdvOtherConfigFlag: off
        AdvHomeAgentFlag: on
        AdvReachableTime: 0
        AdvRetransTimer: 0
        Prefix fec0:106:2700::2/64
                AdvValidLifetime: 12000
                AdvPreferredLifetime: 10000
                AdvOnLink: on
```

```
                    AdvAutonomous: on
                    AdvRouterAddr: on
            AdvSourceLLAddress: 00 02 2D 54 D1 1E
            AdvHomeAgentInfo:
                    HomeAgentPreference: 20
                    HomeAgentLifetime: 1000
```

```
    # ifconfig eth0
    eth0    Link encap:Ethernet    HWaddr 00:90:7D:F3:03:1A
            inet6 addr: fec0:106:2700:0:290:7dff:fef3:31a/64 Scope:Site ①
            inet6 addr: fec0:106:2700::4/64 Scope:Site                  ②
            inet6 addr: fe80::290:7dff:fef3:31a/64 Scope:Link           ③
            UP BROADCAST RUNNING MULTICAST    MTU:1500    Metric:1
            RX packets:513 errors:89 dropped:89 overruns:0 frame:85
            TX packets:140 errors:41 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:100
            RX bytes:56084 (54.7 Kb)    TX bytes:19212 (18.7 Kb)
            Interrupt:3 Base address:0x100
```

①

一个新的（多余的）自动生成的地址。因为我们在/proc/sys/net/ipv6/conf/eth0/autoconf 中设置 autoconf 为 1，所以 MN 会产生一个由 HA 的前缀和其 MAC 地址组合成的新的地址。我不认为能避免这个地址的产生。

②

原始静态 IPv6 地址。

③

引导时生成的链路本地地址。

5 若干实验

5.1 预测试

如上进行每项配置；家乡网络与所访问网络具有不同的 ESSID 尤为重要。

当您在 MN 上启动移动 IPv6 时，您会看到组播路由器请求报文：

```
    # tcpdump -i eth0 -vv ip6 or proto ipv6

    ...
    13:32:54.681763 fe80::202:a5ff:fe6f:a08a > ff02::2: icmp6: router solicitation \
     (src lladdr: 0:2:a5:6f:a0:8a) (len 16, hlim 255)

    13:32:55.681763 fe80::202:a5ff:fe6f:a08a > ff02::2: icmp6: router solicitation \
     (src lladdr: 0:2:a5:6f:a0:8a) (len 16, hlim 255)
```

```
13:32:57.681765 fe80::202:a5ff:fe6f:a08a > ff02::2: icmp6: router solicitation \
 (src lladdr: 0:2:a5:6f:a0:8a) (len 16, hlim 255)
...
```

5.2 移动检测

通常情况下移动检测采用邻居不可达性检测来在缺省路由器不在双向可达时进行检测。此时，移动节点必须发现一台缺省路由器（通常位于新链路上）。

为便于了解所发生的情况，您应为每条命令使用一个 xterm 窗口：

```
# watch ifconfig eth0
# watch route -A inet6
# tcpdump -i eth0 -vv ip6 or proto ipv6
```

为“移动”到另一个网络，您可在 MN 上执行以下命令：

```
# iwconfig eth1 essid visitnet
```

则 MN 位于另一个无线网络，因为它发送出了“路由器请求”（组播），AR 将使用其前缀进行响应。然后 MN 会使用所接收到的前缀和其 MAC 地址来进行自身配置。若您输入 **ifconfig eth0**，您会看到新的 IPv6 地址：

```
# ifconfig eth0
      eth0    Link encap:Ethernet    HWaddr 00:90:7D:F3:03:1A
              inet6 addr: fec0:106:1100:0:290:7dff:fef3:31a/64 Scope:Site ①
              inet6 addr: fec0:106:2700:0:290:7dff:fef3:31a/64 Scope:Site ②
              inet6 addr: fec0:106:2700::4/64 Scope:Site                  ③
              inet6 addr: fe80::290:7dff:fef3:31a/64 Scope:Link           ④
              UP BROADCAST RUNNING MULTICAST    MTU:1500    Metric:1
              RX packets:854 errors:154 dropped:154 overruns:0 frame:148
              TX packets:293 errors:58 dropped:0 overruns:0 carrier:0
              collisions:0 txqueuelen:100
              RX bytes:96536 (94.2 Kb)    TX bytes:44664 (43.6 Kb)
              Interrupt:3 Base address:0x100
```

①

组合 AR 的前缀与 MAC 地址所生成的新的“外地”地址

②

多余的家乡网络地址（因为 HA radvd 报文与 MN autoconf 设置为“true”）。

③

“原始”（家乡）地址

④

引导时生成的链路本地地址

几乎同时，MN 会执行与 HA 的绑定更新。在 tcpduom 窗口中，您会看到几个报文分组发送到 HA。为对绑定更新已发送以及来自 MN 的确认进行验证：

```
# mipdiag -s
        Mobile IPv6 Statistics
        NEncapsulations                 : 0
        NDecapsulations                 : 0
        NBindUpdatesRcvd                : 0
        NBindAcksRcvd                   : 1       ①
```

```
        NBindNAcksRcvd                  : 0
        NBindRqsRcvd                    : 0
        NBindUpdatesSent                : 1      ②
        NBindAcksSent                   : 0
        NBindNAcksSent                  : 0
        NBindRqsSent                    : 0
        NBindUpdatesDropAuth            : 0
        NBindUpdatesDropInvalid     : 0
        NBindUpdatesDropMisc            : 0
        NBindAcksDropAuth               : 0
        NBindAcksDropInvalid            : 0
        NBindAcksDropMisc               : 0
        NBindRqsDropAuth                : 0
        NBindRqsDropInvalid             : 0
        NBindRqsDropMisc                : 0
```

①

接受到一条绑定 ACK 报文。

②

发送一条绑定 UPDATE 报文。

您也可用以下命令对绑定进行验证（在 MN 上）：

```
# mipdiag -l
        Mobile IPv6 Binding update list
        Recipient CN: fec0:106:2700::2
        BINDING home address: fec0:106:2700::4 care-of address: fec0:106:1100:0:290:7dff:fef3:31a
                expires: 936 sequence: 0 state: 1
                delay: 3 max delay 32 callback time: 736
```

您也可在 HA 上用 statistics 选项（-s）和“binding cache”（-c）选项对其进行验证：

```
        # mipdiag -c
        Mobile IPv6 Binding cache
        Home Address        Care-of Address                         Lifetime    Type
        fec0:106:2700::4    fec0:106:1100:0:290:7dff:fef3:31a    971         2
```

5.3 ping6

您可尝试从 Mn ping AR 的 eth1（fec0:106:1100::1）：

```
# ping6 fec0:106:1100::1
PING fec0:106:1100::1(fec0:106:1100::1) from fec0:106:2700::4 : 56 data bytes
64 bytes from fec0:106:1100::1: icmp_seq=1 ttl=62 time=8.01 ms
64 bytes from fec0:106:1100::1: icmp_seq=2 ttl=62 time=8.02 ms
...
```

使用 tcpdump 命令可以看到报文分组如何移动：

```
12:13:51.789688 fec0:106:1100:0:202:a5ff:fe6f:a08a > fec0:106:2700::2: \  ①
fec0:106:2700::4 > fec0:106:1100::1: icmp6: echo request \ ②
(len 64, hlim 64) (len 104, hlim 255)

12:13:51.797675 fec0:106:2700::2 > fec0:106:1100:0:202:a5ff:fe6f:a08a: \ ③
fec0:106:1100::1 > fec0:106:2700::4: icmp6: echo reply \
(len 64, hlim 62) (len 104, hlim 253)
```

①

报文分组首先使用 MN 的新 IPv6 地址从 MN 到 HA。

②

接着从 HA 到 AR。

③

然后 AR 对 HA 进行响应并将报文分组通过隧道传输给 MN。

现在可看到更新后的统计结果（MN 上的）：

```
# mipdiag -s
Mobile IPv6 Statistics
NEncapsulations                 : 56
NDecapsulations                 : 25
...
```

5.4 内核 IP 路由表

MIPv6 所做的一件有意思的事情是改变到隧道的缺省路由。新的缺省路由成为：

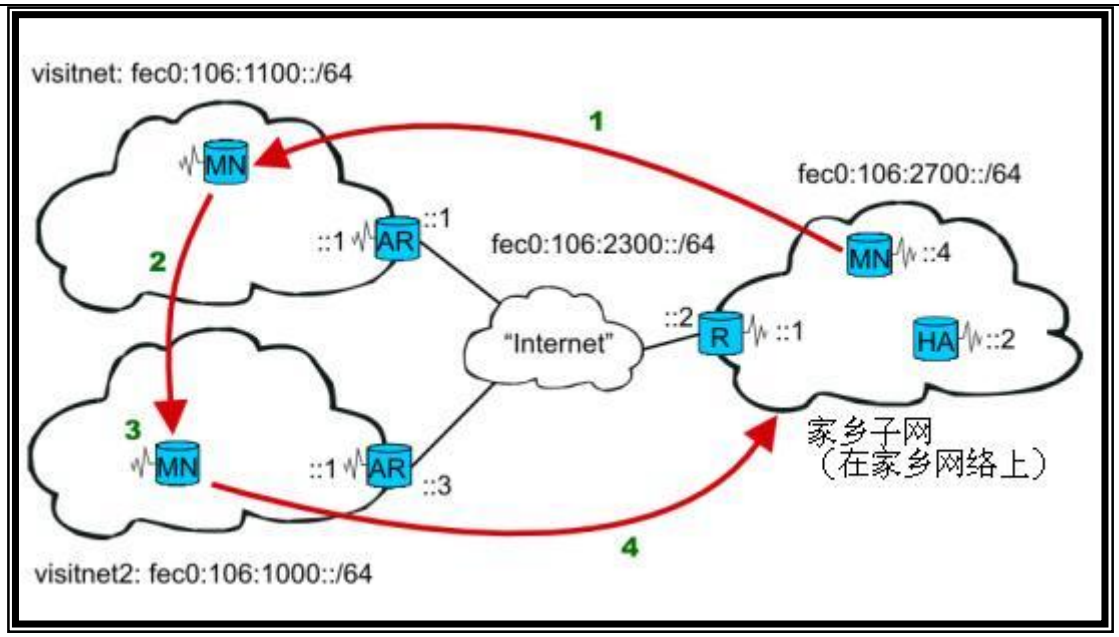
```
# route -A inet6
Kernel IPv6 routing table
Destination          Next Hop          Flags Metric Ref      Use Iface
::/0                 ::                UD    64     0              0 ip6tnl1
....
```

若这样没有添加缺省路由，则可手动进行添加：

```
# ip route ::/0 via dev ip6tnl
```

5.5 移动时经过数个外地 LAN

移动到数个所访问的网络，与移动到一个网络完全相同。您所必考虑的唯一一点是将为每个所访问的网络生成新地址。



MN 移动经过数个不同的 LAN

1.MN 先访问'visitnet'，我们前面已经实现。

2.接着 MN 从'visitnet' t 移动到'visitnet2'。

3.在'visitnet2'时，MN 生成一个新 IPv6 地址并向 HA 发送一条新的绑定更新。

4.然后 MN 返回家乡网络（见下一部分）。

除使用地址 **fec0:106:1000::/6** 代替 **fec0:106:1100::/64** 之外，位于"visitnet2"的 AR 被配置为另一个 AR（位于"visitnet"）。

执行以下命令（在 MN 上），使移动节点从'visitnet'移动到'visitnet2'：

```
# iwconfig eth0 essid visitnet2
```

然后您会看到 MN 配置自身到新网络：

```
# ifconfig eth0
eth1    Link encap:Ethernet    HWaddr 00:90:7D:F3:03:1A
          inet6 addr: fec0:106:1000:0:290:7dff:fef3:31a/64 Scope:Site ①
          inet6 addr: fec0:106:1100:0:290:7dff:fef3:31a/64 Scope:Site
          inet6 addr: fec0:106:2700:0:290:7dff:fef3:31a/64 Scope:Site
          inet6 addr: fec0:106:2700::4/64 Scope:Site
          inet6 addr: fe80::290:7dff:fef3:31a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST    MTU:1500    Metric:1
          RX packets:1073 errors:212 dropped:212 overruns:0 frame:204
          TX packets:371 errors:72 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:120340 (117.5 Kb)    TX bytes:56912 (55.5 Kb)
          Interrupt:3 Base address:0x100
```

①

在'visitnet2'的新的自动配置的地址。

注意！当到达一个新网络时，您可能要在 MN 上重新启动移动 ipv6。

```
# /etc/init.d/mobile-ip6 restart
Stopping Mobile IPv6: OK
Starting Mobile IPv6: OK
```

则 MN 会向 HA 发送一条新绑定更新。注意新的“转交地址”：

```
# mipdiag -l
Mobile IPv6 Binding update list
Recipient CN: fec0:106:2700::2
BINDING home address: fec0:106:2700::4 care-of address: fec0:106:1000:0:290:7dff:fef3:31a
        expires: 973 sequence: 14 state: 1
        delay: 3 max delay 32 callback time: 773
```

您也会看到 HA 上的“绑定缓存”已经更新：

```
# mipdiag -c
Mobile IPv6 Binding cache
Home Address          Care-of Address                    Lifetime    Type
fec0:106:2700::4      fec0:106:1000:0:290:7dff:fef3:31a   943         2
```

5.6 返回家乡网络

您可以执行以下命令使 MN 返回家乡网络：

```
# iwconfig eth0 essid homenet
```

因为 HA 使用设置了 HA-位（AdvHomeAgentFlag）发送出了 radvd 报文，所以 MN 就知道其返回了家乡网络，见 4.2.4 节。

因为 HA 的绑定缓存信息已经清除（空的），所以您可看到 MN“返回了家乡网络”：

```
Mobile IPv6 Binding cache
Home Address          Care-of Address                    Lifetime    Type
```

5.7 实时测试-平滑切换

为获得对移动 IP 机制的生动体验，启动 GnomeMeeting 程序。注意!您必须使用最新的 GnomeMeeting 以获得对 IPv6 的支持！然后进行“移动”，您可看到几乎平滑的切换。

使用提供 IPv6 支持的 GnomeMeeting 测试两个无线网络之间的漫游情况。

6 FAQ

1.*问：为何必须创建/dev/mipv6_dev 表项？*

答：dev 文件主要是为了用户空间工具、mipdiag 能够通过设备文件使用 ioctl 调用对内核参数进行修改。mknod 使用移动 ip6 模块可识别的参数创建了特殊的设备文件。

2. *问：有任何对内核 2.6.x 支持吗？*

答：这是 MIPL 邮件列表上 Henrik Petander 的回答：

“这是对 MIPL 2.6 内核系列状态的简要概述：”

“我们已与 USAGI 计划组合作完成了移动 IPv6 的内核基础设施。基础设施实现了路由优化、隧道与策略路由。”

“现在我们正在对用户空间 daemon 进行开发，它负责处理 MIPv6 信令并对内核部分的运行进行控制。用户空间部分的进展也相当顺利。但是，仍缺乏协议逻辑性，因此实际上还没有可供用户进行测试的部分。我们应该在三月底准备好功能正常，可供测试的原型。”

3. *问：MIPL 是否支持 IPsec？*

答：2.4.x 不支持 IPsec。2.6 系列的 MIPL 一开始就会支持 IPsec。您可使用第三方的 IPsec 实现。

4. *问：我怎样能控制用于 MN 与 CN 之间通信的路由的类型（通过 HA 的隧道或使用绑定更新/确认的直接通信）？*

答：您可采用以下方法实现控制：

/proc/sys/conf/net/ipv6/mobility/accept_return_routability

若您不想使用返回路由可达性与路由优化，就设置其为 0：

echo 0 > /proc/sys/..../accept_return_routability

则 MN 会仅通过家乡隧道与 CN 进行通信。

5. *问：不同的无线网络能否使用不同的 ESSID/WEP 密钥？*

答：可以，但您必须在到达新网络时进行改变。MIPv6 无法自动这么做。

6. *问：如果 MN 已移动经过几个所访问的 LAN，然后返回家乡网络；网卡还是有来自所有所访问网络的自动生成的 IPv6 地址！有任何方法能够“清空/删除”这些地址吗？*

答：不，我不知道任何能够自动删除这些地址的方法，但您可以手动删除：

ifconfig eth0 inet6 del <ipv6-address>

7. *问：主机 B 有两块网卡，分配了两个不同的子网。当我在主机 A 上 pingB 时，没有任何响应！为什么？主机 A 知道主机 B（子网）的位置！*

答：主机 B 不知道主机 A 的位置（B 不知道 A 所在网络的位置），因此您必须添加一条路由表项：

ip route add fec0:106:2700::/64 via fec0:106:2300::1

或

route -A inet6 add fec0:106:2700::/64 gw fec0:106:2300::1 dev eth0

8. *问：在 IPv6 中如何设置缺省网关？*

答：使用传统“路由”：

route -A inet6 add default gw <ipv6-host>

或更加新的 "ip" 命令：

ip route ::/0 via <ipv6-host>

9. *问：为何主机在要求路由器请求时，发送组播地址而不是任播地址？*

答：因为主机需要从所有而不仅仅是任何一台路由器得到答案。这样就能获得所有参数并选择最“佳”的缺省路由器。

10. *问：为何 MN 没注意到它已经移动了？*

答：它认为以前的路由器仍然可达。可能是路由器广播的很长寿命导致的。检查路由器上发送路由器广播的程序的配置。若程序支持路由器广播间隔配置，则您可通过设置间隔使用为 on 来帮助 MN 进行移动检测。细节见 **man radvd.conf** 的执行结果。

7 有用资源

1.Linux 下移动 IPv6 实现 http://www.mipl.mediapoli.com/
2.移动 IP 工作组（IETF）http://www.ietf.org/html.charters/mobileip-charter.html
3.移动 IPv6 移动性支持（RFC3775）http://www.ietf.org/rfc/rfc3775.txt
4. IPv6 工作组 (IETF) http://www.ietf.org/html.charters/ipv6-charter.html
5. RFC2460 Internet 协议，第 6 版（IPv6）规范 http://www.ietf.org/rfc/rfc2460.txt
6. RFC2461 IP 第 6 版(IPv6)邻居发现 http://www.ietf.org/rfc/rfc2461.txt
7. RFC2462 IPv6 Stateless Address Autoconfiguration http://www.ietf.org/rfc/rfc2462.txt
8. Peter Bieringer 的 Linux IPv6 指南(英文) http://ldp.linux.no/HOWTO/Linux+IPv6-HOWTO/
9. IPv6 目前对网络用程序支持情况 http://www.deepspace6.net/docs/ipv6_status_page_apps.html
10. linux 内核编译手记 http://www.20cn.net/ns/cn/zs/data/20030317231808.htm

8 版权、致谢与其它

8.1 版权与许可

8.2 该文档如何产生

8.3 反馈

8.4 致谢