

Using Linux Guests As Layer 3 Routers Between Guest LAN and Hipersocket LANs

Sine Nomine Associates
43596 Blacksmith Square
Ashburn, VA 20147

2006 White Paper Series
April 25, 2006

Copyright © 2006 Sine Nomine Associates

All rights reserved. The contents of this material may be reproduced or transmitted freely provided this copyright notice is retained in its entirety in all copies.

Document History

<i>Date</i>	<i>Revision/ Reissue</i>	<i>Nature of Change</i>
April 25, 2006	Version 01	New Release

Table of Contents

1	IP LAYER 3 ROUTING BASICS.....	1-1
1.1	Next-Hop Routing	1-1
1.2	Default Routes	1-2
1.3	Route Preferences.....	1-2
1.4	Destination Specification and Granularity.....	1-2
1.5	The Importance of Netmasks.....	1-2
2	NETWORK DIAGRAM AND ELEMENT DESCRIPTION.....	2-1
2.1	Network Diagram.....	2-1
2.2	Description of Elements	2-2
2.2.1	z/OS LPAR.....	2-2
2.2.2	z/VM LPAR.....	2-2
2.2.3	Linux Router	2-2
2.2.4	Linux Guests 1, 2, and 3.....	2-2
2.2.5	Guest LAN.....	2-2
2.2.6	Hardware Hipersocket LAN.....	2-2
2.2.7	Corporate Network	2-2
2.2.8	Router in Corporate Network.....	2-3
2.3	Network Addressing Plan	2-3
2.3.1	z/OS System	2-3
2.3.2	z/VM System.....	2-3
2.3.3	Linux Router	2-3
2.3.4	Linux Guests 1, 2, and 3.....	2-3
2.3.5	Guest LAN.....	2-4
2.3.6	Hardware Hipersocket LAN.....	2-4
2.3.7	Corporate Network	2-4
3	ROUTING DESIGN AND ROUTING TABLE CONTENTS	3-5
3.1	z/OS LPAR.....	3-5
3.2	z/VM LPAR.....	3-5

- 3.3 **Linux Router..... 3-5**
- 3.4 **Linux Guests 1, 2, and 3 3-5**
- 3.5 **Router in Corporate Network..... 3-5**

- 4 OTHER SETTINGS REQUIRED.....4-6**
- 4.1 **z/OS LPAR..... 4-6**
- 4.2 **Linux Router..... 4-6**
- 4.3 **Linux Guests 1, 2, and 3 4-6**
- 4.4 **Router in Corporate Network..... 4-6**

List of Tables

There are no tables in this document.

List of Figures

Figure 1: Example Network Diagram.....2-1

1 IP Layer 3 Routing Basics

IP version 4 (and IPv6 as well) routing is based on each node in a network maintaining a decision table of what network destinations can be reached via each of the connected interfaces on the host, and optionally a default destination to use if nothing more specific can be located. This paper provides a short (and simplified) example of using a host system IP stack as a IP packet router between two or more network segments.

This example network is a common configuration used with Linux on zSeries configurations where a Linux guest is used as a router between a hardware hipersocket LAN and a z/VM guest LAN, and is generic enough to provide an example of the major issues with using Linux guests as layer 3 routers. This document does not contain the exact configuration commands to implement the example network, but using this example as a discussion document with corporate network engineering will aid in understanding and configuring routing to provide reliable operation in this type of configuration.

1.1 Next-Hop Routing

As mentioned above, IP routing depends not on a globally maintained list of destinations (like SNA or NJE), but on each node having a knowledge of what link to use to get to the “next hop” to the destination, allowing each node to make incremental and independent decisions on packet routing at each point in the network.

A routing decision is based on pattern matching, e.g. the routing code compares the destination address of a IP packet against a table of patterns maintained in the IP stack, selecting the closest match based on the number of bits specified to match in the routing table entry. Once a match is identified, the IP stack routing table supplies an IP address that is the address of the “next hop” – e.g., send the packet *there*, and that host will figure out what to do with it. Once that destination address is identified, the node identifies which interface can reach that address and queues the packet appropriately. At no point in IP routing design do we rely on any given node knowing the entire network topology – *we just need to know who to punt to for the next step in the chain.*

It’s also important to remember that there needs to be *both* a path from the source to the destination, but also a path *back* from the destination to the original source of the packet or communication won’t occur – the two do NOT have to be the SAME path, but a path in both directions must exist. It’s helpful to visualize this as a pair of separate flows of data, one in each direction – for communication to occur, one has to be able to get there AND back. One may take different roads in each direction, but there has to be some path between endpoints in both directions.

Example:

If you have a packet destined to 192.168.199.4, and you have the following routing table entries:

```
192.168.0.0      via      192.168.0.1
192.168.199.0   via      192.168.199.1
```

A correctly implemented IP stack will queue the packet on the 192.168.199.1 interface – it has the most specific route.

Note also that being directly connected to a network segment inserts an implicit route for the interface address and the network mask of that interface, e.g., if you have an interface on 192.168.199.1 and 192.168.101.1 and both have a 255.255.255.0 network mask, you will have implicit routes for:

192.168.199.0/24 via 192.168.199.1

192.168.101.0/24 via 192.168.101.1

1.2 Default Routes

If there is no specific route matching the destination for the packet, or there is only one way for a system to reach the outside world, most systems supply a “default” route – a destination to use if there is nothing more specific to choose. Since this is the “route of last resort” – you’ve already tried all the other possibilities without a match – there can be only ONE default route at any node in the network.

1.3 Route Preferences

As mentioned above, the route that is the most specific match to a destination is always preferred over a more generic route, but sometimes you may need to provide two equally valid ways to get to a destination, but indicate an administrative preference to use one over another (unless the preferred route is unavailable). Route “weights” or “metrics” allow expressing an administrative preference when two routes would otherwise be equally good.

1.4 Destination Specification and Granularity

Route patterns are specified by the number of bits in the pattern that match the destination – the more bits that match, the more specific the match. The notation used is the base address, a slash (/) and the number of bits that are used in the pattern to match.

Example:

192.168.0.0/16 matches any 192.168.x.x address

192.168.199.0/24 matches only 192.168.199.x address (eg, more specific than the other example)

See Google or Wikipedia articles on classless routing and CIDR for details of what’s doing on here – the critical piece is that the higher number of bits specified in the pattern indicates a more specific match.

1.5 The Importance of Netmasks

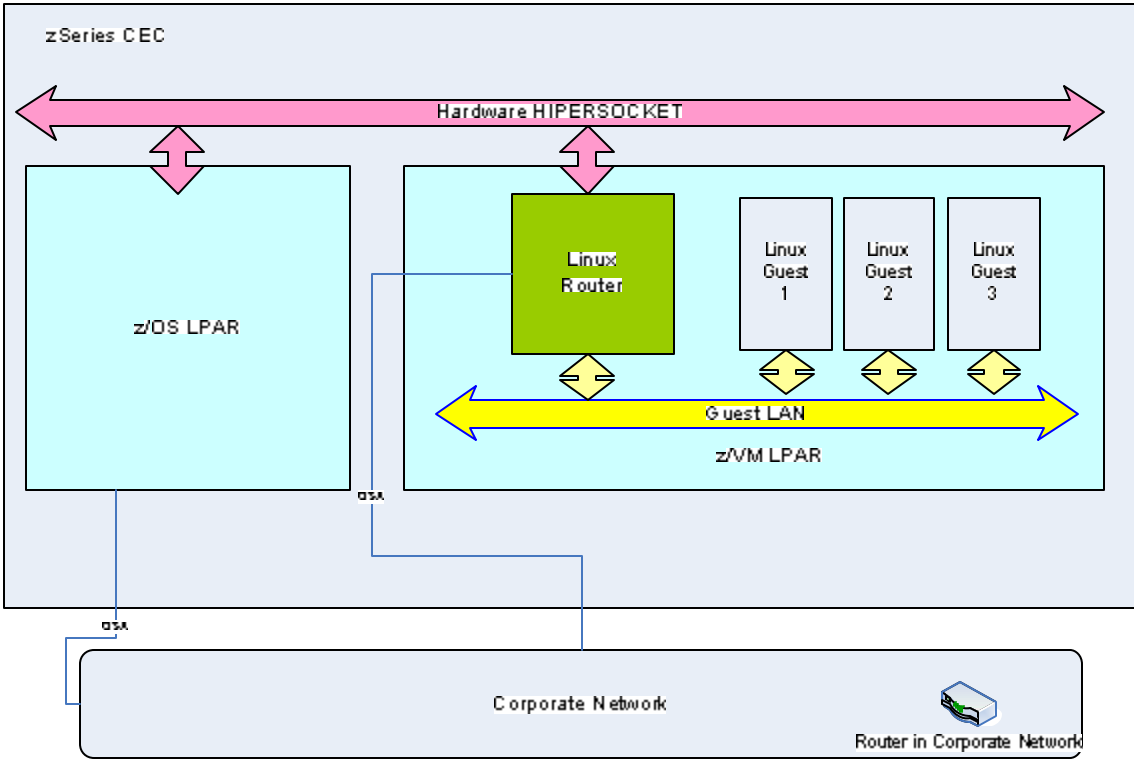
Network masks are not directly related to routing, but are important in that IP uses the network mask value to determine if a destination is on the same network segment as itself, and invokes the routing selection process to reach the destination if it decides it’s not on the same network.

Boiling down a lot of rules – if you want two nodes to communicate directly, there needs to be a path between them, they need to have addresses in the same subnet, and their netmask values must match. Otherwise, they’ll try to go through the routing process.

2 Network Diagram and Element Description

2.1 Network Diagram

The rest of this document will use this diagram as the example network. It provides a number of interesting elements, and illustrates how routing plays a role in using Linux guests as routers.



2.2 Description of Elements

In our example network, we have a number of elements. This section describes each.

2.2.1 z/OS LPAR

In the sample network, the z/OS LPAR has two network connections, one to the hardware hipersocket LAN, and a real OSA connecting to the corporate network. The hipersocket connection should be used only for traffic to Linux Guests 1, 2, and 3. All remaining traffic should travel via the OSA.

2.2.2 z/VM LPAR

The z/VM LPAR acts strictly as a container for the Linux router and Linux Guests 1, 2, and 3, providing the guest LAN simulation. The VM TCP/IP stack is not necessary or required here.

2.2.3 Linux Router

The Linux Router provides packet forwarding from outside desktop clients to Linux Guests 1, 2, and 3, and packet forwarding from Linux Guests 1, 2, and 3 to the z/OS LPAR. Use of packet filters such as `iptables` can prevent desktop clients from traversing the hardware hipersocket LAN to z/OS, as well as use of routing tables. The Linux Router is connected to an external OSA, the guest LAN inside the z/VM LPAR, and the hardware hipersocket LAN connecting to the z/OS LPAR.

2.2.4 Linux Guests 1, 2, and 3

Linux Guests 1, 2, and 3 provide application services to desktop clients and to the z/OS LPAR, possibly accessing a database or z/OS application to present data to a client. Their only connection to the outside world is to the z/VM-simulated guest LAN.

2.2.5 Guest LAN

The guest LAN segment simulated by z/VM connects Linux Guests 1, 2, and 3 to the Linux Router. It is the only method for internal guests to communicate to z/OS or to the outside world.

2.2.6 Hardware Hipersocket LAN

The hardware hipersocket LAN connects the Linux Router and the z/OS LPAR. No traffic from the outside of the zSeries CEC should traverse this connection – all traffic from Linux Guests 1, 2, and 3 must pass through the Linux Router.

2.2.7 Corporate Network

The corporate network is the ordinary conglomeration of networking hardware and gadgets present in any company network. Desktop clients or servers needing to connect to z/OS

applications communicate via z/OS LPAR OSA. Traffic from clients to Linux Guest 1, 2, or 3 must traverse the Linux Router.

2.2.8 Router in Corporate Network

As part of the corporate network, routers and other network devices must be aware of the network topology and routing information. These devices need to be aware that Linux Guests 1, 2, and 3 are reached through the Linux Router.

2.3 Network Addressing Plan

For clarity, the network addressing is not shown in the diagram.

2.3.1 z/OS System

The z/OS system has two network interfaces, one connected to the hipersocket and one connected to an OSA.

In our example, the hipersocket interface is assigned 172.28.1.1 and a network mask of 255.255.255.248. The OSA adapter is assigned 192.204.203.99 and a network mask of 255.255.255.0. The z/OS system default route is 192.204.203.1.

2.3.2 z/VM System

The z/VM system plays no role in the IP routing solution, and thus is not relevant here other than as a container for the Linux guests.

2.3.3 Linux Router

The Linux router has three interfaces, one on the hipersocket LAN, one on the guest LAN, and one external OSA.

In our example, the hipersocket interface is assigned 172.28.1.2 and a network mask of 255.255.255.248. The OSA adapter is assigned 192.204.203.100 and a network mask of 255.255.255.0. The guest LAN interface is assigned 172.28.10.1 and a network mask of 255.255.255.0.

The default route for the Linux Router is 192.204.203.1.

2.3.4 Linux Guests 1, 2, and 3

Linux Guests 1, 2, and 3 have one interface each, attached to the guest LAN segment.

Linux Guest 1 is assigned 172.28.10.2 and a network mask of 255.255.255.0. The default route for Linux Guest 1 is 172.28.10.1 (the Linux Router)

Linux Guest 2 is assigned 172.28.10.3 and a network mask of 255.255.255.0. The default route for Linux Guest 2 is 172.28.10.1 (the Linux Router)

Linux Guest 3 is assigned 172.28.10.4 and a network mask of 255.255.255.0. The default route for Linux Guest 3 is 172.28.10.1 (the Linux Router)

2.3.5 Guest LAN

The guest LAN segment is assigned address range 172.28.10.0 -- 172.28.10.255 and uses a 24 bit subnet mask (255.255.255.0).

2.3.6 Hardware Hipersocket LAN

The hardware hipersocket LAN segment is assigned address range 172.28.1.1 -- 172.28.1.4 and uses a 30 bit subnet mask (255.255.255.248).

2.3.7 Corporate Network

Corporate network hosts are assigned addresses in 192.204.203.x.

3 Routing Design and Routing Table Contents

To successfully implement our sample network, here are the contents of the routing tables necessary to implement the routing policy described in section 2.

3.1 z/OS LPAR

The z/OS LPAR needs only a specific route to indicate that Linux Guests 1, 2 and 3 are reachable via the hipersocket interface, and all other traffic can be handled via the default route. The specific route can be injected as a static route, or learned via a dynamic routing protocol, but the gist of it is the routing table must include an entry similar to:

172.28.1.0/24 via 172.28.1.2

This indicates that all the 172.28.1.x hosts are reachable via the Linux Router.

3.2 z/VM LPAR

No entries are necessary in z/VM. In this configuration, z/VM can be considered as another host connected to the corporate network and treated as such.

3.3 Linux Router

The Linux router needs no special routing, as it has implicit routes for all connected networks by virtue of having interfaces connected to all 3 network segments. The default route causes traffic from outside to flow through the direct OSA connection, and packets from Linux Guest 1, 2, and 3 are routed to z/OS based on destination address. Hipersocket LAN traffic is handled similarly.

3.4 Linux Guests 1, 2, and 3

Linux Guests 1, 2, and 3 need only a default route pointing to the guest LAN interface on the Linux Router, as there is only one possible ingress and exit for all traffic. The Linux Router handles all decisions.

3.5 Router in Corporate Network

Routers and other devices in the corporate network need routes indicating that Linux Guest 1, 2 and 3 are reachable via the Linux Router. Routing tables may look something like this:

172.28.10.0/24 via 192.204.203.100

4 Other Settings Required

4.1 z/OS LPAR

Other than injecting the static route to indicate that the guest LAN addresses are reachable via the Linux Router, no special configuration of z/OS is required. Use of MPROUTE to receive dynamic routing updates is recommended, but not required.

4.2 Linux Router

IP forwarding must be enabled to allow the Linux guest to forward packets not addressed to it. No additional configuration is required. A dynamic routing protocol daemon such as zebra or quagga is recommended, but not required.

4.3 Linux Guests 1, 2, and 3

Only a default route is necessary. No dynamic routing protocol is needed or desirable.

4.4 Router in Corporate Network

Configuration of the corporate network devices to accept dynamic routing updates from z/OS and the Linux Router is recommended for maximum resilience, but a working configuration can be produced using static routes. This is not maintainable on a large scale or in a complex network configuration.