V-NetLab Kernel Module Interface Document

Manish Mehra

Nov 23, 2010

HostMgr – Kernel Module Interface

The kernel module acts as a server processing the requests sent by HostMgr. The communication is done using conventional netlink sockets (socket type AF_NETLINK). The following structure is passed by HostMgr to kernel module for issuing different commands:

```
struct vm_packet
{
   enum struct_type   type;     /* message type */
   unsigned int       vnetID;   /* virtual network-id */
   void              *data;     /* opaque data */
};
```

Following are the message types (commands) exchanged:

1. Register interface
2. Unregister interface
3. Load participation table map
4. Unload participation table map
5. Load listen port
6. Unload listen port
7. Load VM to hub map
8. Unload VM to hub map
9. Port query
10. Display

Messages and Message Formats

The following section describes the message formats of different commands issued by the HostMgr to the Kernel Module

1. Register Interface

The message contains the vmnet interface number specifying the vmnet interface suffix (e.g. 1->vmnet1, 2-> vmnet2, etc). On receiving this message, the kernel module installs packet handler for the virual interface.

2. Unregister Interface

Message format similar to Register Interface message. On receiving this message, the kernel module removes the packet handler installed for that virtual interface

3. Load Participation Table Map

HostMgr fills the following structures `PTKey` & `PTValue` and passes on the information to the

kernel module for loading the participation table. These two structures are packed in a single structure and passed as void *data. The kernel module allocates memory for `PTKey` and `PTValue` and inserts the key-value pair in PT hash table indexed on vnet-id.

```
struct PTKey
{
  int srcHubID;       /* Source Hub ID */
  int dstHubID;       /* Destination Hub ID */
};

struct PTValue
{
  int vlen;                   /* local vmnet list length */
  int *VmnetXList;            /* local vmnet interfaces  */
  int mlen;                   /* remote mac address length */
  unsigned char **MACList;    /* list of remote mac addresses */
};
```

4. Unload Participation Table Map

For unloading participation table map of a particular network, the HostMgr just sends the vnetid of the network to the kernel module. On receiving this message, the kernel module deletes the entry correspoding to vnetid from PT hash table.

5. Load Listenport

HostMgr fills the following structures `ListenPortKey` and `ListenPortData`, packs them in one signle structure and passes on the information to the kernel module. The kernel module allocates memory for key and value and insert key-value pair in Listenport hash table.

```
struct ListenPortKey
{
  uint16_t port;            /* port for user ssh connection */
};

struct ListenPortData
{
  short     vmnetX;         /* vmnet if # (vm_mac associated with) */
  uint8_t   vm_mac[6];      /* MAC address of the virtual machine */
  uint32_t  vm_ip;          /* IP  address of the virtual machine */
  uint8_t   vmnet_mac[6];   /* MAC address of the vmnet interface */
  uint32_t  vmnet_ip;       /* IP  address of the vmnet interface */
};
```

6. Unload Listenport

For unloading a listenport, the HostMgr fills `struct ListenPortKey` and passes it to the kernel module. The kernel module removes the corresponding port entry from Listenport hash table

7. Load VMHub Table

HostMgr fills the following structures `VMHubKey` & `VMHubValue`, packs them in a single structure and passes on the information to the kernel module for loading the VMHub table. The kernel module allocates memory for key and value and insert the key-value pair in VMHub hash table indexed on vnet-id.

```
struct VMHubKey
{
   unsigned char hostID[6];
};

struct VMHubValue
{
   int vmnetX;             /* vmnet interface number */
   int hubID;              /* HUB ID */
   int vnetid;             /* virtual network-id */
};
```

8. Unload VMHub Table

For unloading VMHu table map of a particular network, the HostMgr just sends the vnetid of the network to the kernel module. On receiving this message, the kernel module deletes the entry from VMHub hash table.

9. Port Query

HostMgr uses this message to obtain the list of ports which the kernel module is currently listening on. Following strcuture is filled by kernel module and returned back to the HostMgr

```
#define MAX_PORTS_AVAILABLE 100
struct UsedPortList
{
   int  portBitMap[MAX_PORTS_AVAILABLE];
   int  size;
};
```

10. Display

This is a debug message type which can be used to see the current entries in all hash tables maintained by the kernel module viz Participation Table, Listenport Table, VMHub Table and NAT Table.

Kernel Module APIs

1. Netlink packet handler

The kernel module installs the following packet handler during netlink socket creation. This function is invoked everytime HostMgr sends a command to the kernel module

```
void netlink_receiver_skb(struct sk_buff *skb)
```

2. eth0 pakcet handler

Following packet handler is registered at eth0 interface. Whenever a packet is received at eth0, this packet handler is invoked. It first checks if the packet belongs to a virtual network. It then checks if packet is an SSH packet. If so, it forwards the ssh packet to concerned virtual machine. If not, it forwards a copy of packet on each of destination vmnet interfaces.

```
int kmt_rcv(struct sk_buff      *skb,
            struct net_device   *dev,
            struct packet_type  *pt)
```

3. vmnetX pakcet handler

When HostMgr issues a command to register an interface (e.g. vmnet3, etc), the following packet handler is registered for that virtual interface. It receives packets from virtual machine
that is attched to the local vmnet interface and the packets forwarded by eth0 packet handler.

```
int kmt_send(struct sk_buff     *skb,
             struct net_device  *dev,
             struct packet_type *pt)
```