User Chains

Version: 2115

Table of Contents

User Chains

User chains are configured via configmgr with the "loadchain" and "addchain" wan.conf commands.

The loadchain command takes a module name and reference name as arguments.

```
loadchain <module_path/module_name> <reference name>
```

The addchain command takes the reference name used in the loadchain command, Inetics chain insertion point a priority value (-2 to +2) and optional arguments to the chain. Special default priority values are defined for each insertion point. Using the default priority values will ensure the user chain will be added after any standard Inetics chains.

```
addchain <reference name> <insertion point> <priority> [additional arguments ...]
```

The reference name can be any alpha-numeric string. The chain insertion points outlined in the Inetics white paper are:

- HARDWARE INTERFACE CONTROL (HIC)

  Description: Hardware-specific driver with interrupt service routine (ISR) and raw data I/O routines.
  Function: Moves streaming network data between host memory and the network interface card.
  Examples: PCI 201-ADSL HIC, PCI 530 series HIC, and PCI 1000 series HIC.

- ENCODED DATA PROCESSOR (EDP)

  Description: Processing stage for custom driver software components that manipulate Layer 2.
  Function: Provides a chain for installing custom software that processes data link encapsulation.
  Examples: Customized frame relay processing; test software that injects data link encapsulation errors.

- DATA LINK PROTOCOL (DLP)

  Description: Driver component for processing standards-based WAN protocols.
  Function: Strips and encapsulates standard data link protocols.
  Examples: Cisco HDLC, bisync HDLC, frame relay, PPP, multilink PPP, and ATM.

- DECODED DATA PROCESSOR (DDP)

  Description: Processing stage for custom driver software components that manipulate Layer 3.
  Function: Provides a chain for installing custom software that processes network encapsulation.
  Examples: Firewalls, tunnels, and VPNs; quality of service (QoS) and multicasting applications; test software to inject network encapsulation errors; and custom TCP/IP extensions.

Priority values allow the user chain to be inserted before or after other user chains and standard Inetics chains. Zero is the standard Inetics value so specifying a negative number means your user chain is inserted before the standard chains and a positive number means your user chain is inserted after the standard chains. Configmgr

also allows you to specify special default keywords like rx_ddp_default which inserts your chain after any standard Inetics chains.

Transmit insertion points in execution order

Insertion Point	Default Priority	Description
tx_fromsys	tx_fromsys_default	Processes data before handing off to the kernel.
tx_ddp	tx_ddp_default	Processes data without L2 headers.
tx_dlp	tx_dlp_default	Inserts L2 headers.
tx_edp	tx_edp_default	Processes data with L2 headers.
tx_hic	tx_hic_default	Processes data before handing off to the hardware.

Receive insertion points in execution order

Insertion Point	Default Priority	Description
rx_hic	rx_hic_default	Processes data coming from the hardware.
rx_edp	rx_edp_default	Processes data with L2 headers.
rx_dlp	rx_dlp_default	Removes L2 headers.
rx_ddp	rx_ddp_default	Processes data without L2 headers.
rx_tosys	rx_tosys_default	Processes data before handing off to the kernel.

Example loading the packet logger chain on the receive ddp chain (after the L2 headers have been removed)

```
loadchain logger.o logger
!
interface Serial0
encapsulation ppp
addchain logger rx_ddp rx_ddp_default
ip address 192.168.10.1 255.255.255.252
!
```

Creating your own User Chain

Start by copying an example user chain from the Inetics source directory *src/user_chains* that matches your project requirements. Examples include:

Chain Name	Description
logger.c	Simple packet logging chain.
bridge_monitor.c	Received packets have an Ethernet header prepended before being passed up to the kernel. This is used in monitoring applications where WAN data is tapped and forwarded via Ethernet to a collector.

dev_reroute.c	Received packets have their device pointer changed so it appears they are received on a different device.
mirror.c	Received packets are copied and retransmitted to another device.

Each user chain has per-instance storage via the chain->data pointer. Some of the sample chains use this pointer to hold information such as the mirroring output device or Ethernet header.

Arguments can be passed into the chain via the config() callback function pointer. Set your user chain config callback as demonstrated by the mirror.c sample chain.