

**CESNET technical report number 30/2003**

# **IOSConvert: IOS to XML router configuration file converter - command reference**

**Miroslav Matuska**

10.12.2003

## **1 Introduction**

This document describes the IOSConvert program. It is one of the front-ends of the Netopeer router configuration system. Netopeer is a part of Liberouter project [Lib]. Purpose of the Netopeer system is to create a software system for platform-independent configuration of routers and entire networks. Therefore, for backward compatibility with currently used routers, conversion tool from platform dependent to platform independent configuration is needed.

The Cisco IOS frontend is designed to perform batch conversion from the IOS configuration format into platform independent XML format, according to Netopeer DTD (see [Lho03], samples in CVS). It parses IOS file (Cisco router configuration) and creates XML DOM tree containing same configuration information. The DOM tree is then written into file. All supported Cisco commands are converted into XML elements or attributes

The main task of this converter is to transform existing Cisco configuration files into Netopeer compatible configurations in order to use them on other platform or to process them in other front-end. The program runs in batch mode, eg. input IOS file is converted into output XML file, no command line interface is provided. The resulting XML file can be put into repository, loaded into other application (XML parser) an so on.

IOSConvert program is written in C/C++ language under common Liberouter license.

## **2 Installation requirements**

Packages: *Universal XML parser library (other part of Netopeer system) or libxml2, libxml2-devel*

Compiler: GNU C (gcc/g++)

### 3 Installation

1. Download the source tarball (or checkout the CVS).
2. Unzip and untar if needed, chdir into sources directory.
3. Run `./configure -with-extra-includes=/usr/include/libxml2 [-with-extra-libs=/usr/lib]` where you specify the path of header (include) and library files of libxml2
4. Run `make`. This compiles the program.
5. Now you can run the `iosconvert` program in the `iosconvert` directory.

### 4 Usage

`iosconvert infile outfile [-ndD]`

The *infile* file is the source IOS configuration file and must exist.

The *outfile* file is the resulting XML file. If this file does not exist, it will be created, otherwise it will be overwritten.

Switches:

`-d` prints small debugging info (unknown commands)

`-D` prints large debugging info (all parsed commands)

`-n` does not print any line number info

no switch prints only line number info

### 5 Command reference

This section describes each supported IOS command and their Netopeer XML equivalents. It provides the reference for the "status-quo" of the IOS supported commands. Unlisted commands found in IOS configuration file are put into `<platform-special>` element (see below).

#### 5.1

!

##### 5.1.1 Description

Comment, everything on the line is ignored. It can be also finishing character for previous section, if it is only character on the line.

### **5.1.2 Example**

! This is just a comment for following section

### **5.1.3 XML Equivalent**

None

## **5.2**

### ***access-list***

#### **5.2.1 Description**

Unnamed access list, both standard and extended are supported. See the ACL section below.

#### **5.2.2 Example**

```
access-list 100 permit tcp host 120.25.35.1 any eq telnet
access-list 11 deny ip host 120.25.35.1 any
```

#### **5.2.3 XML Equivalent**

See below in ACL section

## **5.3**

### ***banner motd, banner login***

#### **5.3.1 Description**

Login banner text (like /etc/issue). First two characters are taken as terminating and the text is processed until the second occurrence of terminating characters. Banner motd and login are taken as equal.

#### **5.3.2 Example**

```
banner login ^CWelcome to burcak!^C
```

#### **5.3.3 XML Equivalent**

```
<banner>Welcome to burcak!</banner>
```

## **5.4**

### ***end***

#### **5.4.1 Description**

End of whole configuration, data after this command are not processed

## **5.5**

### ***hostname***

#### **5.5.1 Description**

Name of the router

#### **5.5.2 Example**

```
hostname router-prg-c7200
```

#### **5.5.3 XML Equivalent**

```
<system hostname="router-prg-c7200">
```

## **5.6**

### ***interface***

#### **5.6.1 Description**

Start of a interface subsection - commands related to one interface only. Valid for usual L2 interfaces, Tunnels and VLANs

#### **5.6.2 Example**

```
interface FastEthernet10/0/1
```

#### **5.6.3 XML Equivalent**

```
<device id="cd1" name="FastEthernet10/0/1">
```

#### **5.6.4 Example**

```
interface FastEthernet10/0/1.1
```

#### **5.6.5 XML Equivalent**

```
<vlan-interface device="cd1" id="cd1.1" name="FastEthernet 10/0/1.1">
```

### **5.6.6 Example**

```
interface Tunnel1
```

### **5.6.7 XML Equivalent**

```
<tunnel id="ct1" name="Tunnel1">
```

### **5.6.8 Subcommands**

## **5.7**

### ***(interface) arp***

#### **5.7.1 Description**

Set the type of ARP protocol. Valid type are: arpa, frame-relay, snap, probe. DTD equivalent is to enable ARP only.

#### **5.7.2 Example**

```
arp arpa
```

#### **5.7.3 XML Equivalent**

```
<device arp="on">
```

#### **5.7.4 Example**

```
arp frame-relay
```

#### **5.7.5 XML Equivalent**

```
<device arp="on">
```

#### **5.7.6 Example**

```
arp snap
```

#### **5.7.7 XML Equivalent**

```
<device arp="on">
```

#### **5.7.8 Example**

```
arp probe
```

### **5.7.9 XML Equivalent**

```
<device arp="on">
```

## **5.8**

### ***(interface) description***

#### **5.8.1 Description**

Specifies text info (description) for the current interface

#### **5.8.2 Example**

```
description Link to server segment
```

#### **5.8.3 XML Equivalent**

```
<description>Link to server segment</description>
```

## **5.9**

### ***(interface) duplex***

#### **5.9.1 Description**

Set duplex operation mode on Ethernet devices.

#### **5.9.2 Example**

```
duplex full
```

#### **5.9.3 XML Equivalent**

```
<device duplex="full">
```

#### **5.9.4 Example**

```
duplex half
```

#### **5.9.5 XML Equivalent**

```
<device duplex="half">
```

#### **5.9.6 Example**

```
duplex auto
```

### **5.9.7 XML Equivalent**

```
<device duplex="auto">
```

## **5.10**

### ***(interface) encapsulation (1.form)***

#### **5.10.1 Description**

Set the type of VLAN. Valid types are: dot1q (IEEE 802.1q), isl (Cisco Inter Switch Link)

#### **5.10.2 Example**

```
encapsulation dot1q
```

#### **5.10.3 XML Equivalent**

```
<vlan-interface encapsulation="802.1q">
```

#### **5.10.4 Example**

```
encapsulation isl
```

#### **5.10.5 XML Equivalent**

```
<vlan-interface encapsulation="isl">
```

## **5.11**

### ***(interface) encapsulation (2.form)***

#### **5.11.1 Description**

Set the type of layer-2 protocol. Valid types are: ppp (Point-to-Point Protocol), x25, smds (switched multimegabit data service), atm-dxi

#### **5.11.2 Example**

```
encapsulation ppp
```

#### **5.11.3 XML Equivalent**

```
<device encapsulation="ppp">
```

#### **5.11.4 Example**

```
encapsulation frame-relay
```

#### **5.11.5 XML Equivalent**

```
<device encapsulation="frame-relay">
```

#### **5.11.6 Example**

```
encapsulation x25
```

#### **5.11.7 XML Equivalent**

```
<device encapsulation="x25">
```

#### **5.11.8 Example**

```
encapsulation smds
```

#### **5.11.9 XML Equivalent**

```
<device encapsulation="smds">
```

#### **5.11.10 Example**

```
encapsulation atm-dxi
```

#### **5.11.11 XML Equivalent**

```
<device encapsulation="atm-dxi">
```

### **5.12**

#### ***(interface) ip address***

##### **5.12.1 Description:**

Set the IPv4 address and network mask of this interface. Address family is ipv4

##### **Example:**

```
ip address 192.168.1.2 255.255.255.0
```

### 5.12.2 XML Equivalent

```
<ipv4-address address="192.168.1.2" device="cdX" masklen="24" af="ipv4"/>
```

## 5.13

### *(interface) ip address XXX secondary*

#### 5.13.1 Description

Set the secondary IPv4 address and network mask of this interface.

#### 5.13.2 Example

```
ip address 192.168.1.3 255.255.255.0 secondary
```

#### 5.13.3 XML Equivalent

```
<ipv4-address address="192.168.1.2" device="cdX" masklen="24"  
             role="secondary"/>
```

## 5.14

### *(interface) ip dvmrp, ip pim*

#### 5.14.1 Description

Enables multicast routing.

#### 5.14.2 XML Equivalent

```
<device multicast="on">
```

## 5.15

### *(interface) ip rip send version*

#### 5.15.1 Description

Sets the version of sent RIP information on this interface

#### 5.15.2 Example

```
ip rip send version 2
```

#### 5.15.3 XML Equivalent

```
<rip-interface send=yes out-version="2"/>
```

## 5.16

### *(interface) ip rip receive version*

#### 5.16.1 Description

Sets the version of received RIP information on this interface, both versions are received by default

#### 5.16.2 Example

```
ip rip receive version 2
```

#### 5.16.3 XML Equivalent

```
<rip-interface receive=yes in-version="2">
```

#### 5.16.4 Example

```
ip rip receive version 1 2
```

#### 5.16.5 XML Equivalent

```
<rip-interface receive=yes in-version="both">
```

## 5.17

### *(interface) ip rip authentication mode*

#### 5.17.1 Description

Sets the RIP authentication type on this interface. Valid types are: text (plain-text), md5 (hashed text)

#### 5.17.2 Example

```
ip rip authentication mode text
```

#### 5.17.3 XML Equivalent

```
<rip-interface>  
  <rip-authentication mode="simple">  
</rip-interface>
```

#### 5.17.4 Example

```
ip rip authentication mode md5
```

### 5.17.5 XML Equivalent

```
<rip-interface>  
  <rip-authentication mode="md5">  
</rip-interface>
```

## 5.18

### *(interface) ip rip authentication key-chain*

#### 5.18.1 Description

Sets the RIP authentication key-chain on this interface. It should be defined before in the "key" section.

#### 5.18.2 Example

```
ip rip authentication key-chain mychain
```

#### 5.18.3 XML Equivalent

```
<rip-interface>  
  <rip-authentication password="SeCrEt">  
</rip-interface>
```

(where "mychain" is the name for key chain that includes the string "SeCrEt" as the password, see key section)

## 5.19

### *(interface) ip access-group XXX in*

#### 5.19.1 Description

Applies access control list (packet filter chain) on the input traffic on this interface.

#### 5.19.2 Example

```
ip access-group 101 in
```

#### 5.19.3 XML Equivalent

```
<device filter-in="101"/>
```

## 5.20

### *(interface) ip access-group XXX out*

#### 5.20.1 Description

Applies access control list (packet filter chain) on the output traffic on this interface.

#### 5.20.2 Example

```
ip access-group 101 out
```

#### 5.20.3 XML Equivalent

```
<device filter-out="101"/>
```

## 5.21

### *(interface) ipv6 address*

#### 5.21.1 Description

Set the IPv6 address and network mask of this interface. Address family (af attribute) is ipv6. Attribute "link" specifies forced link-local address on this interface

#### 5.21.2 Example

```
ip address 2001:718:1f02::1/64
```

#### 5.21.3 XML Equivalent

```
<ipv6-address address="2001:718:1f02::1" device="cd1"  
              masklen="64" af="ipv6"/>
```

#### 5.21.4 Example

```
ip address FFFE:718:1f02::1/64 link-local
```

#### 5.21.5 XML Equivalent

```
<ipv6-address address="FFFE:718:1f02::1" device="cd1"  
              masklen="64" af="ipv6" scope="link"/>
```

## 5.22

### *(interface) ipv6 rip ripng enable*

#### 5.22.1 Description

Enabling the RIPng protocol distribution on this interface

#### 5.22.2 XML Equivalent

```
<ripng>  
  <ripng-interface device="cd1"/>  
</ripng>
```

## 5.23

### *(interface) mac-address*

#### 5.23.1 Description

Set the MAC address of NIC on this device.

#### 5.23.2 Example

```
mac-address 00:00:a0:3f:19:0d
```

#### 5.23.3 XML Equivalent

```
<device macaddr="00:00:a0:3f:19:0d"/>
```

## 5.24

### *(interface) mtu*

#### 5.24.1 Description

Set the Maximum Trasmit Unit parameter of the link.

#### 5.24.2 Example

```
mtu 512
```

#### 5.24.3 XML Equivalent

```
<device mtu="512"/>
```

## **5.25**

### ***(interface) no***

#### **5.25.1 Description**

Negativize the command on the rest of line.

#### **5.25.2 Example**

```
no ip address
```

#### **5.25.3 XML Equivalent**

None

## **5.26**

### ***(interface) shutdown***

#### **5.26.1 Description**

Administratively disable the interface.

#### **5.26.2 XML Equivalent**

```
<device disable="yes"/>
```

## **5.27**

### ***(interface) speed***

#### **5.27.1 Description**

Set the speed in Mbit/s on Ethernet devices.

#### **5.27.2 Example**

```
speed auto
```

#### **5.27.3 XML Equivalent**

```
<device speed="auto"/>
```

#### **5.27.4 Example**

```
speed 100
```

### **5.27.5 XML Equivalent**

```
<device speed="100"/>
```

### **5.27.6 Example**

```
speed 10
```

### **5.27.7 XML Equivalent**

```
<device speed="10"/>
```

## **5.28**

### ***(interface) tunnel checksum***

#### **5.28.1 Description**

Enables the checking of checksum in tunnel. Default is: off

#### **5.28.2 XML Equivalent**

```
<tunnel checksum="on"/>
```

## **5.29**

### ***(interface) tunnel key***

#### **5.29.1 Description**

Tunnel selector or security key

#### **5.29.2 Example**

```
tunnel key mykey
```

#### **5.29.3 XML Equivalent**

```
<tunnel key="mykey"/>
```

## **5.30**

### ***(interface) tunnel mode***

#### **5.30.1 Description**

Set the tunnel encapsulation type. Valid types are: gre

### **5.30.2 Example**

```
tunnel mode gre
```

### **5.30.3 XML Equivalent**

```
<tunnel mode="gre"/>
```

## **5.31**

### ***(interface) tunnel path-mtu-discovery***

#### **5.31.1 Description**

Enables the path MTU discovery in tunnel. Default is: off

#### **5.31.2 Example**

```
tunnel path-mtu-discovery
```

#### **5.31.3 XML Equivalent**

```
<tunnel pmtudisc="on"/>
```

## **5.32**

### ***(interface) tunnel sequence-datagrams***

#### **5.32.1 Description**

Enables the checking of the correct sequence of datagrams in tunnel. Default is: off

#### **5.32.2 Example**

```
tunnel sequence-datagrams
```

#### **5.32.3 XML Equivalent**

```
<tunnel sequence="on"/>
```

## **5.33**

### ***(interface) tunnel source tunnel-source-interface***

#### **5.33.1 Description**

Set the source point of the tunnel - interface.

### 5.33.2 Example

```
tunnel source Serial0/1
```

### 5.33.3 XML Equivalent

```
<tunnel>  
  <tunnel-source-interface device="cd4"/>  
</tunnel>
```

Where "cd4" is device "Serial0/1".

## 5.34

### *(interface) tunnel source tunnel-source-address*

#### 5.34.1 Description

Set the source point of the tunnel - address.

#### 5.34.2 Example

```
tunnel source 132.177.5.10
```

#### 5.34.3 XML Equivalent

```
<tunnel>  
  <tunnel-source-address address="132.177.5.10"/>  
</tunnel>
```

## 5.35

### *(interface) tunnel destination tunnel-destination-address*

#### 5.35.1 Description

```
tunnel destination 222.22.2.3
```

#### 5.35.2 Example

! This is just a comment for following section

#### 5.35.3 XML Equivalent

```
<tunnel>  
  <tunnel-destination-address address="222.22.2.3"/>  
</tunnel>
```

## 5.36

### *(interface) tunnel ttl*

#### 5.36.1 Description

Set the TTL parameter of the tunnel.

#### 5.36.2 Example

```
tunnel ttl 64
```

#### 5.36.3 XML Equivalent

```
<tunnel ttl="64"/>
```

## 5.37

### *(interface) tx-ring-limit*

#### 5.37.1 Description

Set the length of transmitting (output) data queue on interface.

#### 5.37.2 Example

```
tx-ring-limit 100
```

#### 5.37.3 XML Equivalent

```
<device txqlen="100"/>
```

## 5.38

### *ip access-list*

#### 5.38.1 Description

Specifies the section the named packet filter chain (access control list). Both standard and extended ACLs are supported. See the section of ACLs below.

#### 5.38.2 Example

```
ip access-list standard mylist-stand  
  deny 10.1.0.0 0.0.255.255  
  permit any any
```

```
ip access-list extended mylist-ext
  deny ip 10.1.0.0 0.0.255.255 10.2.6.0 0.0.0.255 eq telnet
  permit icmp any any 10 12
```

### 5.38.3 XML Equivalent

See below (ACL section)

## 5.39

### *ip domain-list*

#### 5.39.1 Description

Specifies a domain name for domain lookups. There can be more domain-list commands in one configuration. If both domain-name and domain-list commands are present, domain-list is used.

#### 5.39.2 Example

```
ip domain-list cesnet.cz
ip domain-list muni.cz
```

#### 5.39.3 XML Equivalent

```
<system>
  <dns>
    <search>
      <domain-suffix suffix="cesnet.cz"/>
      <domain-suffix suffix="muni.cz"/>
    </search>
  </dns>
</system>
```

## 5.40

### *ip domain-name*

#### 5.40.1 Description

Specifies a domain name for domain lookups. There can be only one domain-name command in one configuration. If both domain-name and domain-list commands are present, domain-list is used.

#### 5.40.2 Example

```
ip domain-name muni.cz
```

### 5.40.3 XML Equivalent

```
<system>
  <dns>
    <search>
      <domain-suffix suffix="muni.cz"/>
    </search>
  </dns>
</system>
```

## 5.41

### *ip multicast-routing*

#### 5.41.1 Description

Enables multicast routing on given machine.

#### 5.41.2 XML Equivalent

```
<system multicast-routing="on">
```

## 5.42

### *ip name-server*

#### 5.42.1 Description

Specifies a DNS server in numeric form for domain lookups.

#### 5.42.2 Example

```
ip name-server 192.168.1.1
```

#### 5.42.3 XML Equivalent

```
<system>
  <dns>
    <dns-server address="192.168.1.1"/>
  </dns>
</system>
```

## 5.43

### *ip route*

#### 5.43.1 Description

Specifies a static IPv4 route in form: target network+subnet mask+next hop. Next hop could be IP address or name of point-to-point interface. If the name of "next-hop" interface is "Null", then the packet will be discarded silently (route into blackhole).

#### 5.43.2 Example

```
ip route 192.168.1.0 255.255.255.0 192.168.3.1 110
```

#### 5.43.3 XML Equivalent

```
<static-routes>  
  <route af="ipv4" preference="110">  
    <destination address="192.168.1.0" length="24"/>  
    <nexthop via="192.168.3.1"/>  
  </route>  
</static-routes>
```

#### 5.43.4 Example

```
ip route 192.168.1.0 255.255.255.0 Null
```

#### 5.43.5 XML Equivalent

```
<static-routes>  
  <route af="ipv4" type="blackhole">  
    <destination address="192.168.1.0" length="24"/>  
  </route>  
</static-routes>
```

## 5.44

### *ipv6 route*

#### 5.44.1 Description

Specifies a static IPv6 route in form: target network+subnet mask+next hop. Next hop could be IP address or name of point-to-point interface. If the name of "next-hop" interface is "Null", then the packet will be discarded silently (route into blackhole).

### 5.44.2 Example

```
ipv6 route 2001:718:0:4::/64 2001:718:1f00::1
```

### 5.44.3 XML Equivalent

```
<static-routes>  
  <route af="ipv6">  
    <destination address="2001:718:0:4::" length="64"/>  
    <nexthop via="2001:718:1f00::1"/>  
  </route>  
</static-routes>
```

### 5.44.4 Example

```
ipv6 route 2001:718:0:4::/64 Null
```

### 5.44.5 XML Equivalent

```
<static-routes>  
  <route af="ipv6" type="blackhole">  
    <destination address="2001:718:0:4::" length="64"/>  
  </route>  
</static-routes>
```

## 5.45

### *ipv6 router ripng*

#### 5.45.1 Description

Declares a section of global commands for IPv6 RIPng

#### 5.45.2 Subcommands

## 5.46

*(ipv6 router ripng) redistribute [ static | connected | isis  
| ospf | bgp ]*

#### 5.46.1 Description

Includes IPv6 routes from other routing protocols in RIPng updates (via routing table, routes are exported from routing table)

### 5.46.2 Example

```
redistribute static
```

### 5.46.3 XML Equivalent

```
<ripng>  
  <route-export chain="ripng-export"/>  
</ripng>  
  
<routing>  
  <route-filters>  
    <route-filter-chain id="ripng-export"  
      name="ripng-export-filter-chain">  
      <route-filter-rule>  
        <route-match-list>  
          <match-route-source source="static"/>  
        </route-match-list>  
        <route-action-list>  
          <accept-action/>  
        </route-action-list>  
      </route-filter-rule>  
    </route-filter-chain>  
  </route-filters>  
</routing>
```

## 5.47

### *ipv6 unicast routing*

#### 5.47.1 Description

Enables IPv6 unicast routing on given machine.

#### 5.47.2 XML Equivalent

None

## 5.48

### *key-chain*

#### 5.48.1 Description

Start of key chain section

### **5.48.2 Example**

key chain mychain

### **5.48.3 XML Equivalent**

None (converted into elements that use this key chain, please see RIP authentication section)

### **5.48.4 Subcommands**

## **5.49**

### ***(key-chain) key-string***

#### **5.49.1 Description**

Specifies the text string (password) for this key chain.

#### **5.49.2 Example**

key string SeCrEt

#### **5.49.3 XML Equivalent**

None (converted into elements that use this key chain, see RIP authentication section.)

## **5.50**

### ***no***

#### **5.50.1 Description**

Negativize the command on the rest of line.

## **5.51**

### ***no ip split-horizon***

#### **5.51.1 Description**

Turns off split horizon for RIP on all RIP enabled interfaces

### 5.51.2 XML Equivalent

```
<rip>  
  <rip-interface device="cd4" split-horizon="no"/>  
  <rip-interface device="cd5" split-horizon="no"/>  
</rip>
```

(Where cd4 and cd5 are interfaces, that run RIP)

## 5.52

### *router rip*

#### 5.52.1 Description

Declares a section of global commands for IPv4 RIP.

#### 5.52.2 XML Equivalent

```
<rip disable="no">
```

#### 5.52.3 Subcommands

## 5.53

### *(router rip) network*

#### 5.53.1 Description

Includes IPv4 network prefix in RIP updates

#### 5.53.2 Example

```
network 146.1.0.0  
network 146.2.0.0
```

#### 5.53.3 XML Equivalent

```
<rip>  
  <rip-interface device="cd4"/>  
  <rip-interface device="cd5"/>  
</rip>
```

(Where cd4 corresponds to the interface with 146.1.x.x/16 subnet and cd5 corresponds to the interface with 146.2.x.x/16 subnet)

## **5.54**

### ***(router rip) distance***

#### **5.54.1 Description**

Specifies administrative distance of RIP routes (120 by default).

#### **5.54.2 Example**

```
distance 120
```

#### **5.54.3 XML Equivalent**

```
<rip preference="120"/>
```

## **5.55**

### ***(router rip) default-metric***

#### **5.55.1 Description**

Specifies the default metric of RIP routes.

#### **5.55.2 Example**

```
default-metric 2
```

#### **5.55.3 XML Equivalent**

```
<rip default-metric="2"/>
```

## **5.56**

### ***(router rip) neighbor***

#### **5.56.1 Description**

Specifies unicast RIP neighbor (mostly on NBMA networks)

#### **5.56.2 Example**

```
neighbor 192.168.1.16
```

#### **5.56.3 XML Equivalent**

```
<rip>  
  <rip-neighbor address="192.168.1.16"/>  
</rip>
```

## 5.57

### *(router rip) passive-interface*

#### 5.57.1 Description

Disables any RIP activity on specified interface

#### 5.57.2 Example

```
passive-interface FastEthernet0/1
```

#### 5.57.3 XML Equivalent

```
<rip-interface device="cd4" send="no" receive="no"/>
```

Where "cd4" is the "FastEthernet0/1 interface."

## 5.58

### *(router rip) redistribute [ static | connected | isis | ospf | bgp ]*

#### 5.58.1 Description

Includes IPv4 routes from other routing protocols in RIP updates (via routing table)

#### 5.58.2 XML Equivalent

```
<rip>  
  <route-export chain="rip-export"/>  
</rip>
```

(see RIPng "redistribute" section to see "rip-export" static routes definition - same as "ripng-export")

## 5.59

### *(router rip) version*

#### 5.59.1 Description

Specifies version of RIP protocol (can be overwritten on interfaces). Default value is to accept version 1 and 2 and distribute only version 1 broadcasts.

### 5.59.2 XML Equivalent

```
<rip version="12in1out"/>
```

(by default whe no other IOS command is present)

### 5.59.3 Example

```
version 1
```

### 5.59.4 XML Equivalent

```
<rip version="1"/>
```

### 5.59.5 Example

```
version 2
```

### 5.59.6 XML Equivalent

```
<rip version="2"/>
```

## 5.60

### *version*

#### 5.60.1 Description

Specifies the version of IOS system for which the configuration file was intended

#### 5.60.2 Example

```
version 12.2
```

#### 5.60.3 XML Equivalent

```
<platform-special platform="ios" version="12.2"/>
```

## 5.61

### *\*Anything else that is not covered by DTD\**

#### 5.61.1 Description

Unknown parts of IOS configuration (=those that are not currently supported by DTD) are placed into "platform-special" element. Netopeer Cisco backend

may use these information to restore complete configuration (next to XML data) for Cisco router. The aim of this element is to allow working with whole Cisco IOS configuration on Cisco routers in Netopeer without losing information even if some IOS commands are not supported in DTD. Supported commands are moved into other XML elements and the rest is put here. If some commands of configuration sub-section (eg. interface or routing protocols subsection) are unsupported, then subsection is made in the "platform-special" element also (separated by the exclamation mark "!"; and only unsupported commands are put into it). Unsupported flags in ACLs rules are ignored now (these are not put into "platform-special" element now).

### 5.61.2 Example

```
router bgp 100
  neighbor 152.13.12.11
  !
```

### 5.61.3 XML Equivalent

```
<platform-special platform="ios">
router bgp 100
  neighbor 152.13.12.11
  !
</platform-special>
```

## 6 Access Control Lists

Access Control List is group of lines (=criterias+actions) that are matched against data to accept or reject these data by given criteria. Cisco IOS uses two types of IP access lists: simpler "standard" ACLs and complex ACLs called "extended". Standard ACLs:

- operate on Layer 3 (IP packets)
- uses source address to match accept/reject rule

Extended ACLs:

- operate on higher layer
- can use source and destination address, number of port or type of service to match accept/reject rule

#### 6.0.4 Example of numbered ACL:

```
access-list 100 permit tcp any host 120.25.35.1 eq telnet
access-list 100 permit tcp any host 120.25.35.1 eq http
access-list 100 permit tcp any host 120.25.35.1 eq 443
access-list 100 deny ip any host 120.25.35.1
```

#### 6.0.5 Example of two named ACLs:

```
ip access-list standard mylist-stand
deny 10.1.0.0 0.0.255.255
deny 10.2.0.0 0.0.255.255
deny 10.3.0.0 0.0.255.255
permit any any
```

```
ip access-list extended mylist-ext
permit tcp 10.1.0.0 0.0.255.255 host 10.2.6.1 eq telnet
deny tcp 10.1.0.0 0.0.255.255 10.2.6.0 0.0.0.255 eq telnet
permit icmp any any 10 12
```

As you can see, named ACL has a kind of "header" with identification info. Header starts a section of lines that belong to same access list (specified in header). Numbered ACLs do not have the first "header" line, they include identification info on each line in the list. Each line is one rule in the list. They are interpreted consecutive from the first one to the last one. If a match is found, appropriate action (specified on matched line) is taken and the processing of the ACL is finished. If no match is found, the implicit "reject" action is taken

General template for specifying standard numbered ACLs is:

```
access-list number action source_address_with_mask log
```

General template for specifying extended numbered ACLs is:

```
access-list number action protocol source_address_with_mask
[source_port] destination_address_with_mask
[destination_port] [various_protocol_options] [log | log-input]
```

Template for the named form of ACL is similar - remove initial "*access-list number*" string

Legend:

**number** is unique identifier of ACL, it shows which lines belong to same ACL (to the same packet filter chain)

**action** can be either deny (means that matching packet will be rejected), permit (means that matching packet will be accepted) or remark. Remark means that this line is not rule, just remark for list maintainer (the line contains no more information than this remark)

Following values are criteria:

**protocol** (only in extended ACL) - specifies type of protocol that has to be in packet (IP, TCP, UDP, ICMP)

**source\_address\_with\_mask** and **destination\_address\_with\_mask** are ranges of IP addresses (eg. subnets if the masks are contiguous) that will be matched against source or destination address of the packet. Masks are IOS specific - wildcard masks, bitwise NOT of normal subnet mask. The address and mask is separated with a space. So if we want to match whole 168.12.0.0/16 subnet, we use notation *168.12.0.0 0.0.255.255*

There are few abbreviations for this field: *any* means *0.0.0.0 0.0.0.0* (match is in every case), *host* means that the wildcard mask is *0.0.0.0* (match only on one specified host, not on subnet)

**source\_port** and **destination\_port** fields are used for TCP and UDP protocols (no use for IP or ICMP, of course). Match should be done on exact port numbers (keyword *eq*) or port range (keywords *range, lt, gt, neq*). These keywords are used like this:

*eq 22* - match is made when the port equals to 22

*range 0 1024* - match is made when the port is between 0 and 1024

*lt 128* - match is made when the port is lesser than 128

*gt 1024* - match is made when the port is greater than 1024

*neq 80* - match is made when the port not equals to 80

*neq http* - match is made when the port not equals to 80 (same as previous line but different notation)

The port number can be specified either as an decimal number or as text string (for well-known ports like 21=ftp, 23=telnet, 25=smtp, 80=http)

**various\_protocol\_options** are one or more of the following:

- *urg, rsh, pst, fin, ack, established, syn* are state bits of TCP connection in TCP header. If a keyword is present, then desired bit in TCP header must be set (=1). Established means that syn bit must be 0 (allows only established connections)
- *dscp, tos* matches against certain value of TOS and DSCP fields in IP header

- *fragments* means that this rule is valid only for subsequent fragments of fragmented IP packet
- for ICMP protocol : *type* and *code* values could be specified as two consecutive numbers in this field (text representation of ICMP values is not supported in IOSConvert now).

**log** or **log-input** means that packet match here should be logged (without this field, only action is taken and no logging is provided)

### 6.0.6 XML representation

One line of access list means one `<packet-filter-rule>` element. Actions and logging functions are specified in `<packet-action-list>` element (subelement of `<packet-filter-rule>`). `<accept-action>` is used for permit, `<drop-action>` is used for deny and `<log-action>` element is used for logging.

### 6.0.7 Example

```
access-list XXX permit XXX
access-list XXX permit XXX log
access-list XXX deny XXX log-input
```

### 6.0.8 XML Equivalent

```
<packet-action-list>
  <accept-action count="no" />
</packet-action-list>

<packet-action-list>
  <accept-action count="no" />
  <log-action count="no" />
</packet-action-list>

<packet-action-list>
  <drop-action count="no" />
  <log-action count="no" level="notice"/>
</packet-action-list>
```

Remarks are stored in `<description>` element (subelement of `<packet-filter-rule>`).

### 6.0.9 Example

```
access-list 100 remark This is remark to the next rule in chain...
```

### 6.0.10 XML Equivalent

```
<packet-filters>
  <packet-filter-chain default-policy="drop-action" id="100">

    <packet-filter-rule>
      <description>
        This is remark to the next rule in chain...
      </description>
    </packet-filter-rule>

  </packet-filter-chain>
</packet-filters>
```

Source and destination addresses with their subnet masks are represented as *<prefix-list>* elements and referenced only in *<packet-match-list>* element.

### 6.0.11 Example

cp2 represents "any" keyword, cp3 represents "host 217.155.135.2"

### 6.0.12 XML Equivalent

```
<prefix-lists>

  <prefix-list id="cp2" name="cp2">
    <match-prefix address="0.0.0.0" length="32" af="ipv4" />
  </prefix-list>

  <prefix-list id="cp3" name="cp3">
    <match-prefix address="217.155.135.2" length="32" af="ipv4" />
  </prefix-list>

</prefix-lists>
```

Protocol criteria field is represented by following elements (all of the are subelements of *<packet-match-list>*)

### 6.0.13 Example for IP(v4) protocol

```
access-list NUMBER ACTION ip SRC_ADDRESS_MASK DST_ADDRESS_MASK
                    fragments [dscp 12 | tos 3]
```

#### 6.0.14 XML Equivalent

```
<match-ipv4 fragments="subseq">
  <match-source list="SRC_ADDRESS_MASK"/>
  <match-destination list="DST_ADDRESS_MASK"/>
  [<match-dsfield dscp="12"/> | <match-ecnfield ect="on" ce="on"/>]
</match-ipv4>
```

IPv6 element is similar to IPv4 element.

#### 6.0.15 Example for ICMP protocol

```
access-list NUMBER ACTION icmp SRC_ADDRESS_MASK DST_ADDRESS_MASK 15 12
```

#### 6.0.16 XML Equivalent

```
<match-icmp type="15" code="12"/>
```

#### 6.0.17 Example for TCP protocol

```
access-list NUMBER ACTION tcp SRC_ADDRESS_MASK gt 1024
                        DST_ADDRESS_MASK eq 22 ack syn
```

#### 6.0.18 XML Equivalent

```
<match-tcp>
  <match-source-port-range lo="1024" hi="65535" negate="no"/>
  <match-destination-port-range lo="22" hi="22" negate="no"/>
  <match-tcp-flags syn="on" ack="on" .../>
</match-tcp>
```

#### 6.0.19 Example for UDP protocol

```
access-list NUMBER ACTION udp SRC_ADDRESS_MASK lt 1024
                        DST_ADDRESS_MASK neq 53
```

#### 6.0.20 XML Equivalent

```
<match-udp>
  <match-source-port-range lo="0" hi="1024" negate="no"/>
  <match-destination-port-range lo="53" hi="53" negate="yes"/>
</match-udp>
```

The criteria are specified in *<packet-match-list>* element (subelement of *<packet-filter-rule>*).

### 6.0.21 Example

```
access-list 1 permit host 217.155.135.2
access-list 1 deny any
```

### 6.0.22 XML Equivalent

```
<packet-filters>
  <packet-filter-chain default-policy="drop-action" id="1">

    <packet-filter-rule>
      <packet-match-list>
        <match-ipv4 fragment="all">
          <match-source list="cp3" negate="no" />
        </match-ipv4>
      </packet-match-list>
    <packet-action-list>
      <accept-action count="no" />
    </packet-action-list>
  </packet-filter-rule>

  <packet-filter-rule>
    <packet-match-list>
      <match-ipv4 fragment="all">
        <match-source list="cp2" negate="no" />
      </match-ipv4>
    </packet-match-list>
    <packet-action-list>
      <drop-action count="no" />
    </packet-action-list>
  </packet-filter-rule>

</packet-filter-chain>
</packet-filters>
```

## 7 Remarks

- ESP protocol is not supported now (no support in DTD).
- IPv6 addresses access-lists are on basic level of support due to developing syntax of IPv6 ACLs in Cisco IOS (see [Gro01])
- MAC addresses access-lists are supported, but currently only on single host basis (no "net"-mask or groups of MAC addresses).

## 8 References

### References

- [Lho03] Lhotka Ladislav, *XML schema for router configuration data: An annotated DTD*  
CESNET Technical Report 2/2003
- [Cis02] Cisco Systems, Inc., *Cisco IOS Command References Master Index, Release 12.0*  
<http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/rbkixol.htm>  
2002
- [Gro01] Grossetete Patrick, *Cisco IOS IPv6 Access Control Lists*  
[http://www.cisco.com/warp/public/732/Tech/ipv6/docs/ipv6\\_acls0403.ppt](http://www.cisco.com/warp/public/732/Tech/ipv6/docs/ipv6_acls0403.ppt),  
Cisco Systems, Inc., 2001
- [Lib] CESNET z.s.p.o., *Liberouter project, PC based IPv6 router*  
<http://www.liberouter.org>