

# APNIC Training

**SANOG**

## Internet Routing Registry (IRR)

July 21, 2010 – Paro, Bhutan

**16 South Asian Network Operators Group (SANOG) Conference**

In conjunction with Bhutan Telecom Ltd.



# Introduction

- Presenters
  - Nurul Islam Roman
    - Training Officer (Technical)
    - [nurul@apnic.net](mailto:nurul@apnic.net)



# Assumptions & Objectives

## Assumptions

- Are current or prospective APNIC members
- Have not submitted many requests
- Are not familiar or up-to-date with address policies
- Are not familiar with procedures
- Are interested in address management

## Objectives

- To provide an understanding of address management
- To provide a working knowledge of the procedures for requesting resources from APNIC and managing these
- To keep membership up-to-date with the latest policies
- Liaise with members.

# What is a Routing Registry?

- A repository (database) of Internet routing policy information
  - Autonomous Systems exchanges routing information via BGP
  - Exterior routing decisions are based on policy based rules
  - However BGP does not provides a mechanism to publish/communicate the policies themselves
  - RR provides this functionality
- Routing policy information is expressed in a series of objects

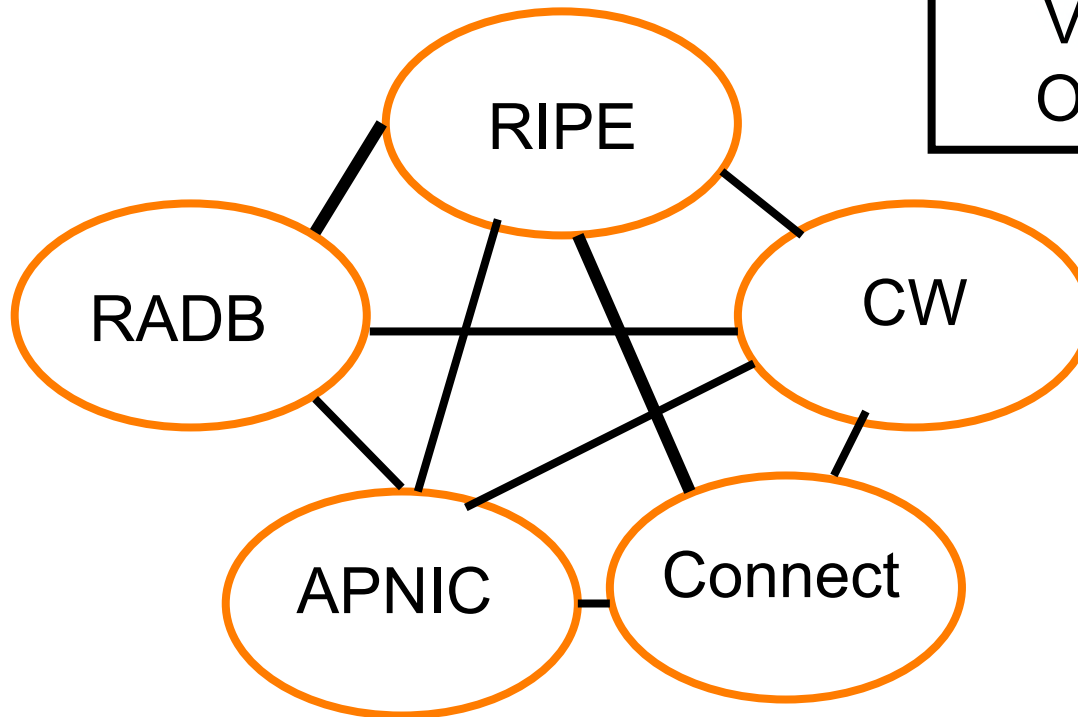
# Routing registry objects

- Route, aut-num, inet-rtr, peering-set, AS-set, rtr-set, filter-set
  - Each object has its own purpose
  - Together express routing policies
- More details covered later

# What is a Routing Registry?

- Global Internet Routing Registry database
  - <http://www.irr.net/>
    - Uses RPSL
- Stability and consistency of routing
  - network operators share information
- Both public and private databases
  - These databases are independent
    - but some exchange data
    - only register your data in one database

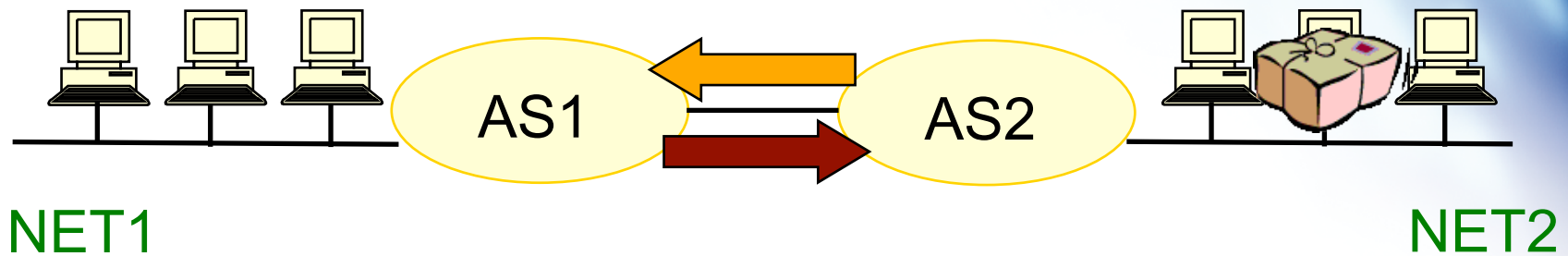
# What is a Routing Registry?



ARIN, ArcStar, FGC,  
Verio, Bconnex,  
Optus, Telstra, ...

IRR = APNIC RR + RIPE DB + RADB + C&W + ARIN + ...

# Representation of routing policy



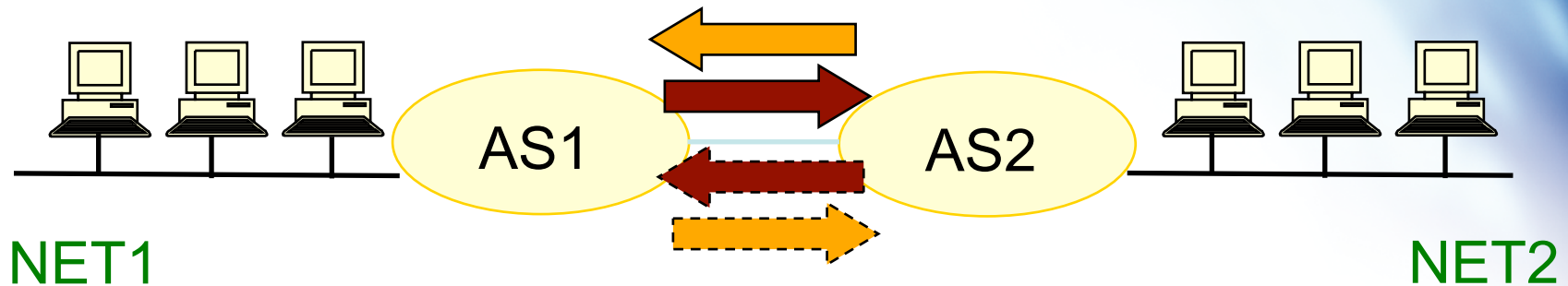
In order for traffic to flow from NET2 to NET1  
between AS1 and AS2:

AS1 has to announce NET1 to AS2 via BGP

And AS2 has to accept this information and use it

Resulting in packet flow from NET2 to NET1

# Representation of routing policy (cont.)



In order for traffic to flow towards from NET1 to NET2:

AS2 must announce NET2 to AS1

And AS1 has to accept this information and use it

Resulting in packet flow from NET 1 to NET2

# What is routing policy?

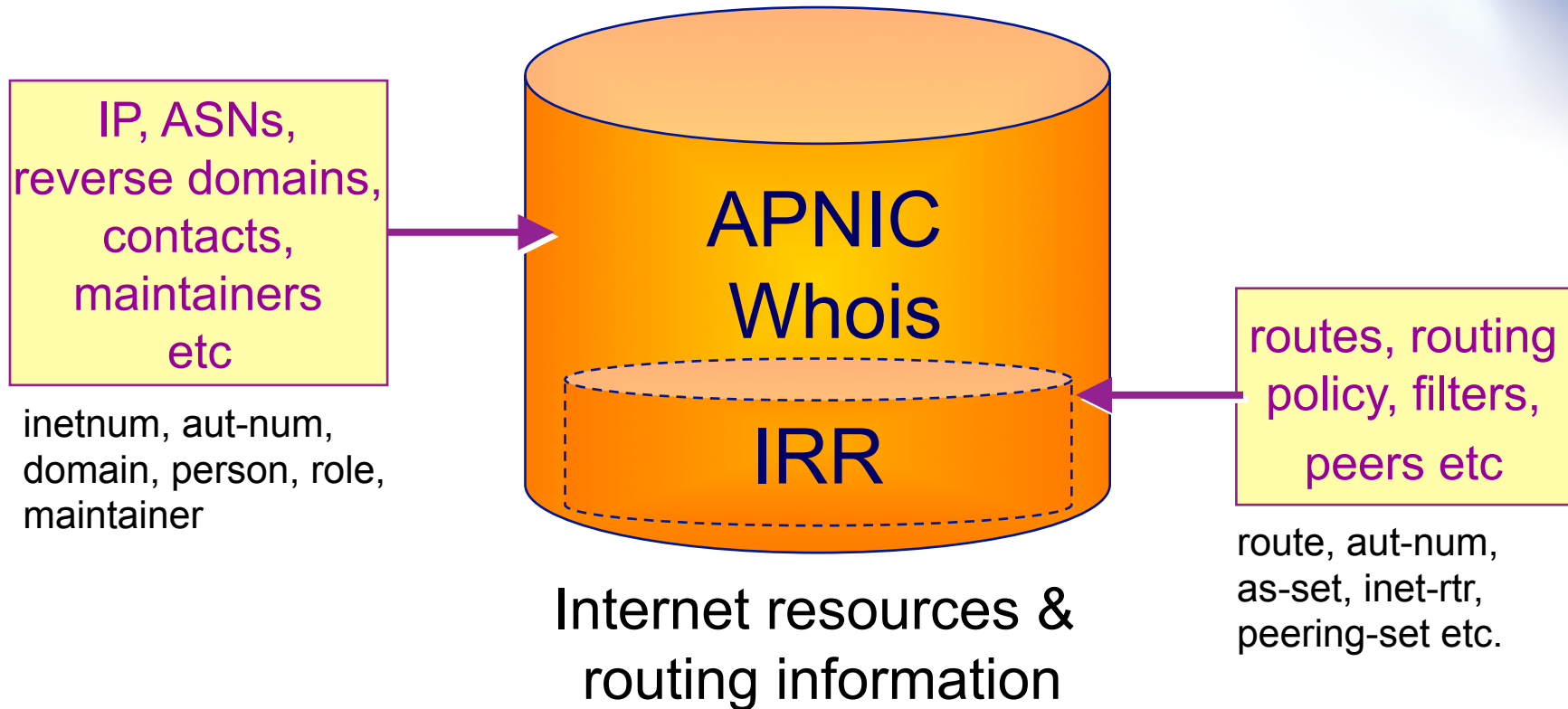
- Description of the routing relationship between autonomous systems
  - Who are my BGP peers?
    - Customer, peers, upstream
  - What routes are:
    - Originated by each neighbour?
    - Imported from each neighbour?
    - Exported to each neighbour?
    - Preferred when multiple routes exist?
  - What to do if no route exists?
  - What routes to aggregate?

# APNIC Database & the IRR

- APNIC whois Database
  - Two databases in one
- Public Network Management Database
  - “whois” info about networks & contact persons
    - IP addresses, AS numbers etc
- Routing Registry
  - contains routing information
    - routing policy, routes, filters, peers etc.
  - APNIC RR is part of the global IRR

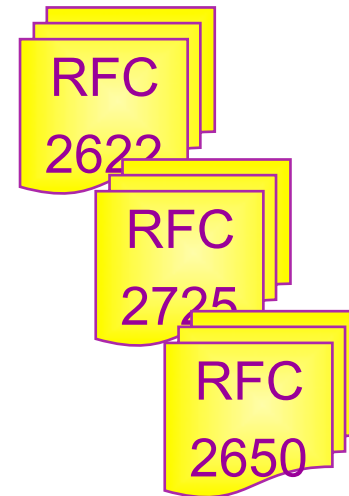
# Integration of Whois and IRR

- Integrated APNIC Whois Database & Internet Routing Registry



# RPSL

- Routing Policy Specification Language
  - Object oriented language
    - Based on RIPE-181
  - Structured whois objects
  
- Higher level of abstraction than access lists
  
- Describes things interesting to routing policy:
  - Routes, AS Numbers ...
  - Relationships between BGP peers
  - Management responsibility
  
- Relevant RFCs
  - Routing Policy Specification Language
  - Routing Policy System Security
  - Using RPSL in Practice

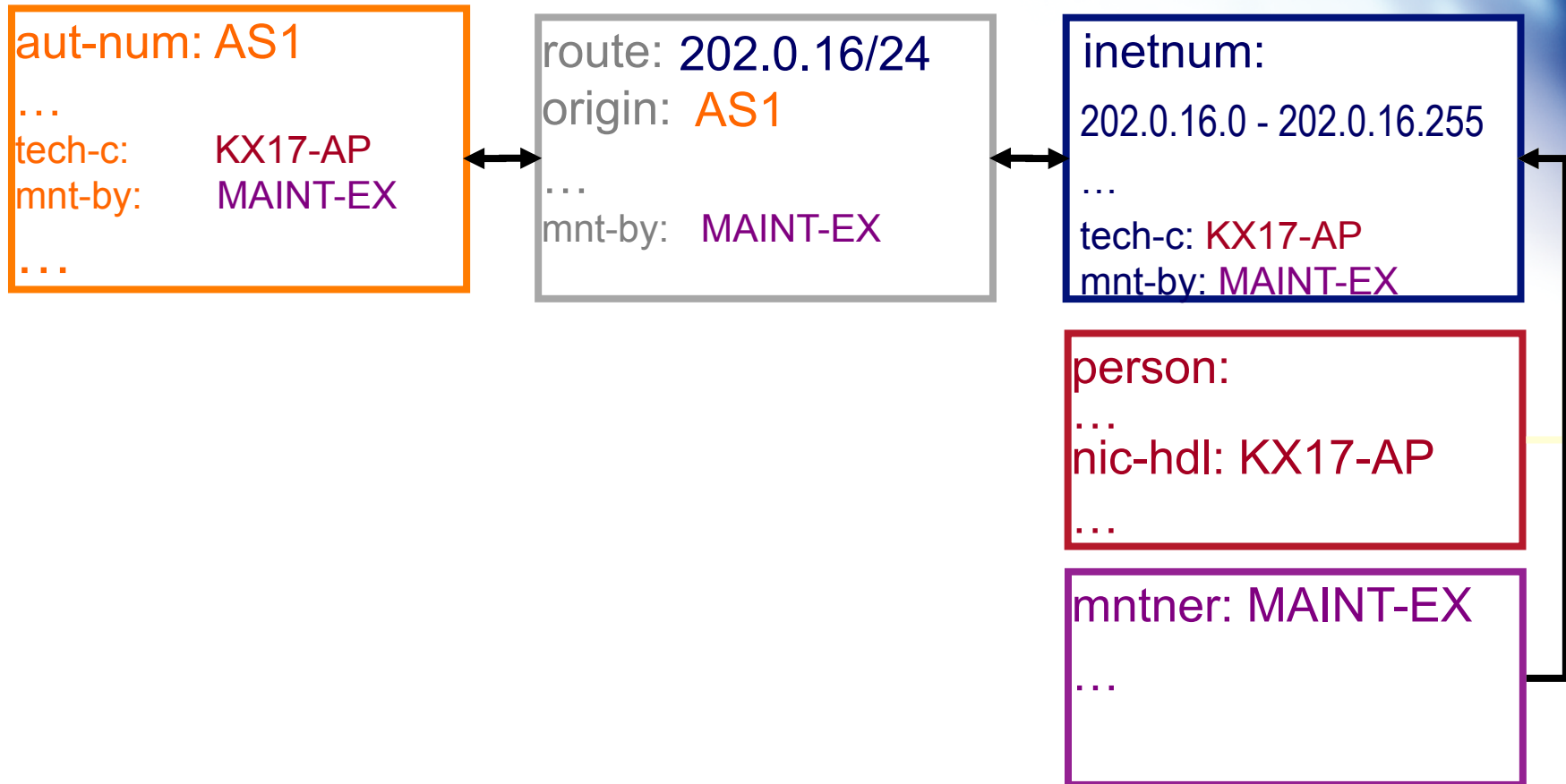


# IRR objects

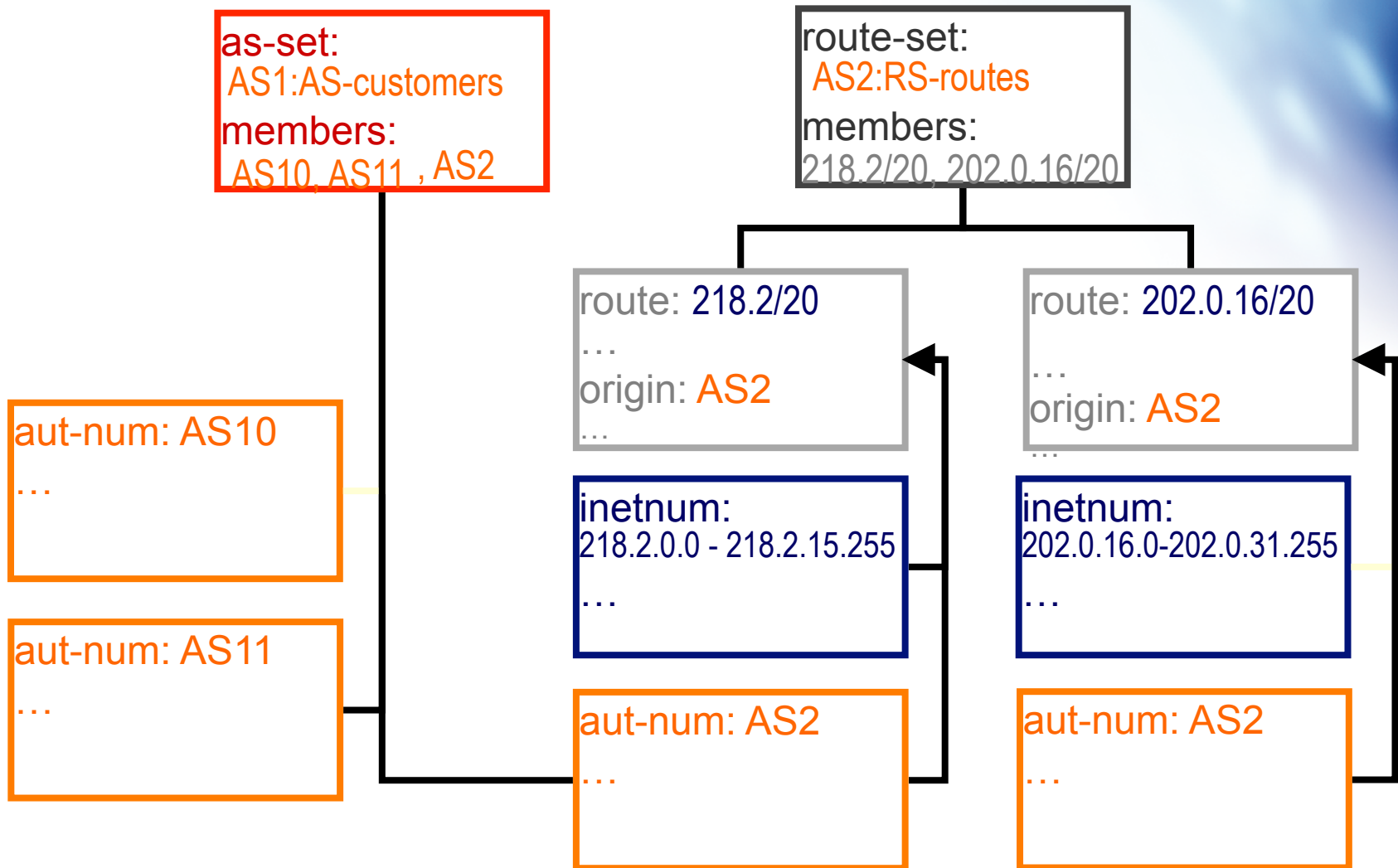
- **route**
  - Specifies interAS routes
- **aut-num**
  - Represents an AS. Used to describe external routing policy
- **inet-rtr**
  - Represents a router
- **peering-set**
  - Defines a set of peerings
- **route-set**
  - Defines a set of routes
- **as-set**
  - Defines a set of **aut-num** objects
- **rtr-set**
  - Defines a set of routers
- **filter-set**
  - Defines a set of routes that are matched by its filter

[www.apnic.net/db/ref/db-objects.html](http://www.apnic.net/db/ref/db-objects.html)

# Inter-related IRR objects



# Inter-related IRR objects



# Hierarchical authorisation

- **mnt-routes**
  - authenticates **creation** of route objects
    - creation of route objects must pass authentication of mntner referenced in the mnt-routes attribute
  - Format:
    - `mnt-routes: <mntner>`

In:

`inetnum`

, `aut-num`

and

`route`

objects

# Authorisation mechanism

```
inetnum:      202.137.181.0 - 202.137.196.255
netname:      SPARKYNET-WF
descr:        SparkyNet Service Provider
...
mnt-by:       APNIC-HM
mnt-lower:    MAINT-SPARKYNET1-WF
mnt-routes:   MAINT-SPARKYNET2-WF
```

This object can only be modified by APNIC

Creation of more specific objects (assignments) within this range has to pass the authentication of MAINT-SPARKYNET

Creation of route objects matching/within this range has to pass the authentication of MAINT-SPARKYNET-WF

# Creating route objects

- Multiple authentication checks:
  - Originating ASN
    - mntner in the mnt-routes is checked
    - If no mnt-routes, mnt-lower is checked
    - If no mnt-lower, mnt-by is checked
  - AND the address space
    - Exact match & less specific route
      - mnt-routes etc
    - Exact match & less specific inetnum
      - mnt-routes etc
  - AND the route object mntner itself
    - The mntner in the mnt-by attribute

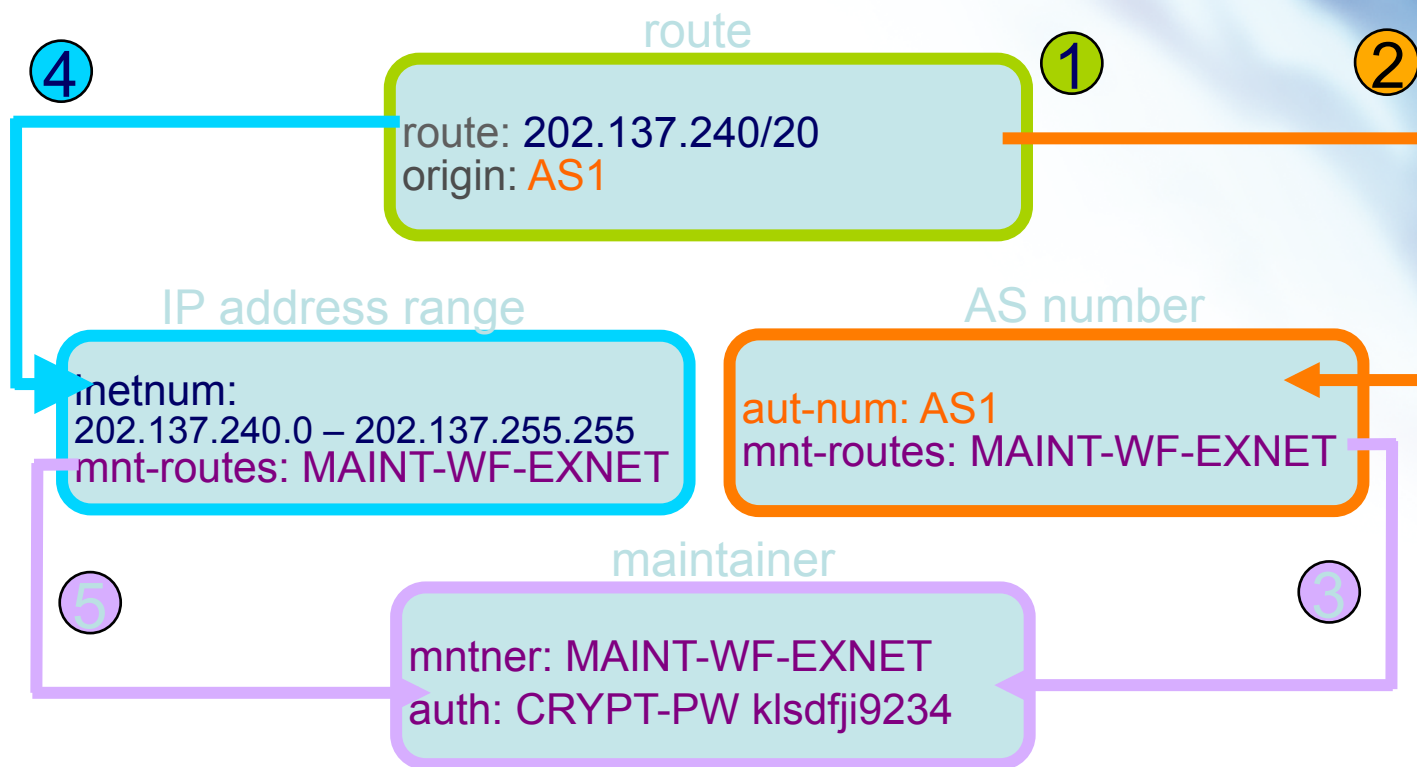
aut-num

inetnum

route  
(encompassing)

route

# Creating route objects



1. Create route object and submit to APNIC RR database
2. DB checks aut-num obj corresponding to the ASN in route obj
3. Route obj creation must pass auth of mntner specified in aut-num *mnt-routes* attribute.
4. DB checks inetnum obj matching/encompassing IP range in route obj
5. Route obj creation must pass auth of mntner specified in inetnum *mnt-routes* attribute.

# Using the Routing Registry

## Overview of the IRRToolSet

# IRRToolSet

- Set of tools developed for using the Internet Routing Registry (IRR)
- Work with Internet routing policies
  - These policies are stored in IRR in the Routing Policy Specification Language (RPSL)
- The goal of the IRRToolSet is to make routing information more convenient and useful for network engineers
  - Tools for automated router configuration,
  - Routing policy analysis
  - On-going maintenance etc.

# IRRToolSet

- History
  - Originated at the USC Information Sciences Institute during 1997-2001 as the Routing Arbiter ToolSet (RAToolSet) project
  - Later migrated to RIPE NCC in order to continue its development and support (RAToolSet was later changed to IRRToolSet)
  - RIPE NCC later transferred maintenance of the tool set to ISC, who began accepting code from the community and providing code maintenance

# IRRToolSet

- Now maintained by ISC:
  - <http://irrtoolset.isc.org>
  - Download: <ftp://ftp.isc.org/isc/IRRToolSet/>
    - Installation needs: lex, yacc and C++ compiler



# Use of RPSL - RtConfig

- RtConfig v4
  - part of IRRToolSet
- Reads policy from IRR (aut-num, route & -set objects) and generates router configuration
  - vendor specific:
    - Cisco, Bay's BCC, Juniper's Junos and Gated/RSd
  - Creates route-map and AS path filters
  - Can also create ingress / egress filters
    - (documentation says Cisco only)

# Why use IRR and RtConfig?

- Benefits of RtConfig
  - Avoid filter errors (typos)
  - Expertise encoded in the tools that generate the policy rather than engineer configuring peering session
  - Filters consistent with documented policy
    - (need to get policy correct though)

# Using RPSL in practice

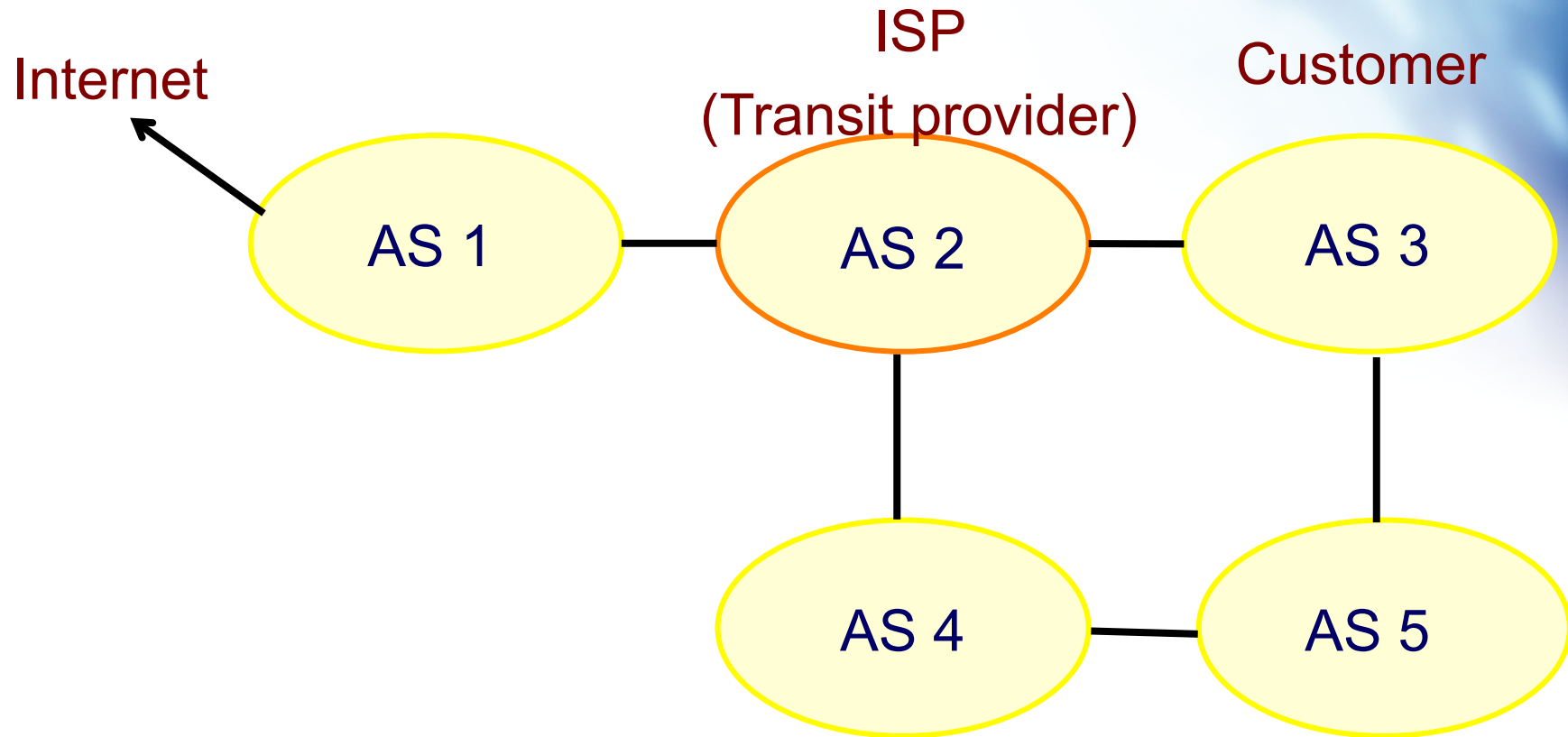
# Overview

- Review examples of routing policies expression
  - Peering policies
  - Filtering policies
  - Backup connection
  - Multihoming policies

## RPSL - review

- Purpose of RPSL
  - Allows specification of your routing configuration in the public IRR
    - Allows you to check “Consistency” of policies and announcements
  - Gives opportunities to consider the policies and configuration of others

# Common peering policies



- Peering policies of an AS
  - Registered in an aut-num object

# Common peering policies

- Policy for AS3 in the AS2 aut-num object

```
aut-num:      AS2
as-name:      SAMPLE-NET
dsescrip:    Sample AS
import:       from AS1 accept ANY
import:       from AS3 accept <^AS3+$>
export:       to AS3 announce ANY
export:       to AS1 announce AS2 AS3
admin-c:      CW89-AP
tech-c:       CW89-AP
mtn-by:       MAINT-SAMPLE-AP
changed:      sample@sample.net
```

# ISP customer – transit provider policies

- Policy for AS3 and AS4 in the AS2 aut-num object

```
aut-num:      AS2
import:       from AS1 accept ANY
import:       from AS3 accept <^AS3+$>
import:       from AS4 accept <^AS4+$>
export:       to AS3 announce ANY
export:       to AS4 announce ANY
export:       to AS1 announce AS2 AS3 AS4
```

# AS-set object

- Describe the customers of AS2

as-set:	AS2:AS-CUSTOMERS
members:	AS3 AS4
changed:	<a href="mailto:sample@sample.net">sample@sample.net</a>
source:	APNIC

# Aut-num object referring as-set object

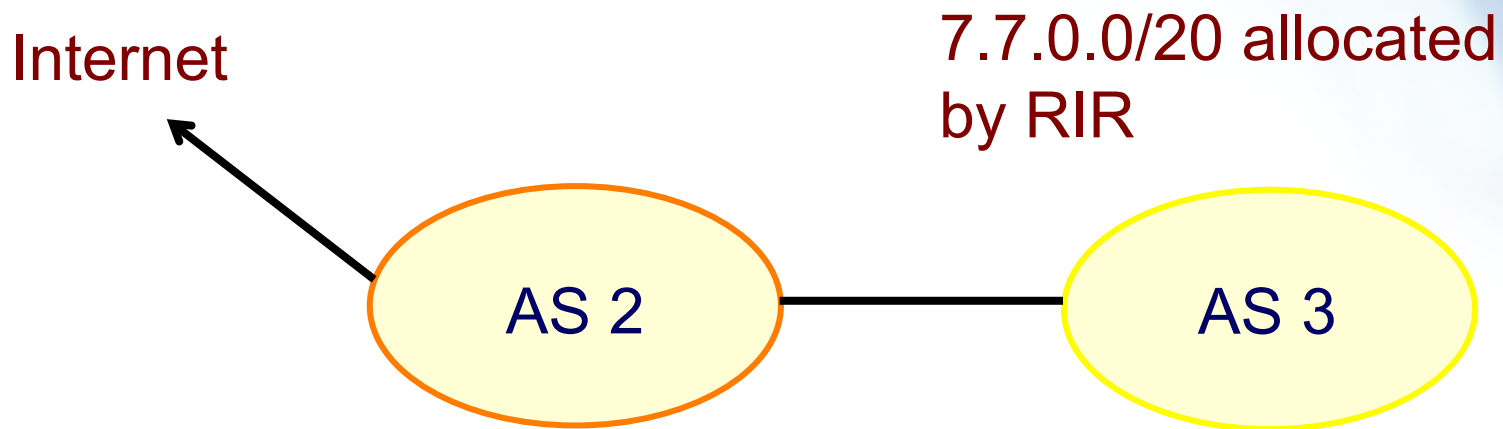
```
aut-num:      AS2
import:       from AS1 accept ANY
import:       from AS2:AS-CUSTOMERS accept
               <^AS2:AS-CUSTOMERS+$>
export:       to AS2:AS-CUSTOMERS announce ANY
export:       to AS1 announce AS2 AS2:AS-
               CUSTOMERS
```

```
aut-num:      AS1
import:       from AS2 accept <^AS2+AS2:AS-
               CUSTOMERS+$>
export:       .....
```

## Express filtering policy

- To limit the routes one accepts from a peer
  - To prevent the improper use of unassigned address space
  - To prevent malicious use of another organisation's address space

# Filtering policy



AS3 wants to announce part or all of 7.7.0.0/20 on the global Internet.

AS2 wants to be certain that it only accepts announcements from AS3 for address space that has been properly allocated to AS3.

# Aut-num object with filtering policy

```
aut-num:      AS2
import:       from AS3 accept { 7.7.0.0/20^20-24 }
.....
```

For an ISP with a growing or changing customer base, this mechanism will not scale well.

Route-set object can be used.

# Route-set

```
route-set:    AS2:RS-ROUTES:AS3
members:     7.7.0.0/20^20-24
changed:     sample@sample.net
source:      APNIC
```

Specifies the set of routes that will be accepted from a given customer

Set names are constructed hierarchically:

AS2 : RS-ROUTES : AS3



indicates whose sets  
these are

indicates peer AS

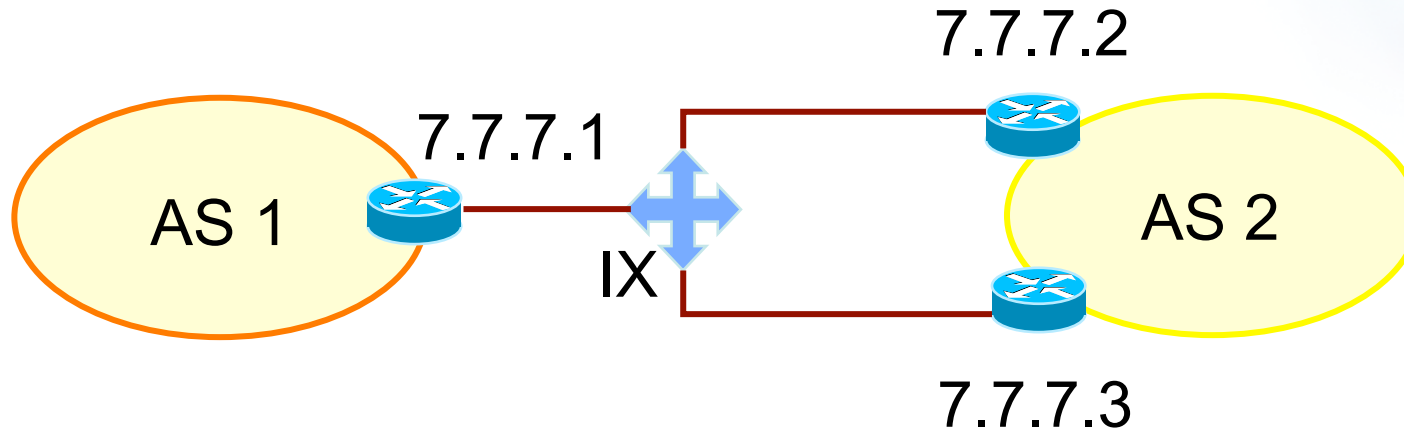
## Filter configuration using route-set – AS2

```
import:    from AS1 accept ANY
import:    from AS3 accept AS2:RS-ROUTES:AS3
import:    from AS4 accept AS2:RS-ROUTES:AS4
export:    to AS2:AS-CUSTOMERS announce ANY
export:    to AS1 announce AS2 AS2:AS-CUSTOMERS
```

RPSL allows the peer's AS number to be replaced by the keyword **PeerAS**

```
import:    from AS2:AS-CUSTOMERS accept
           AS2:RS-ROUTES:PeerAS
```

# Including interfaces in peering definitions: AS1



How to define AS1's routing policy by specifying its boundary router?

## Including interfaces in peering definitions: AS1 (cont.)

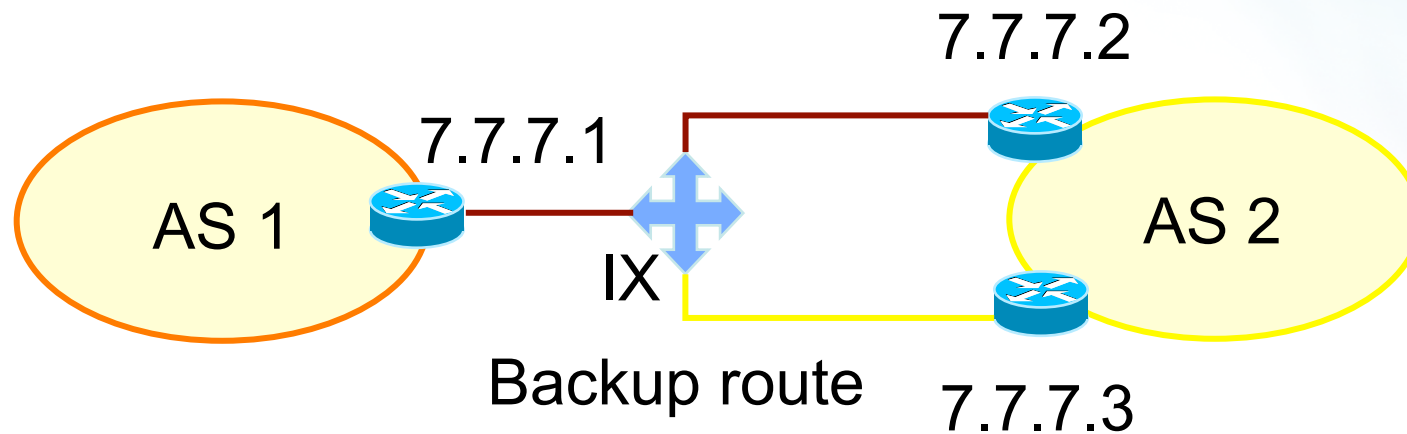
```
aut-num: AS1  
import: from AS2 at 7.7.7.1 accept <^AS2+$>
```

AS1 may want to choose to accept:

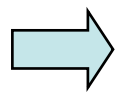
- only those announcements from router 7.7.7.2
- discard those announcements from router 7.7.7.3

```
aut-num: AS1  
import: from AS2 7.7.7.2 at 7.7.7.1 accept <^AS2+$>
```

# Describing simple backup connections: AS1



How to define AS1's routing policy of its backup route?



Use preference

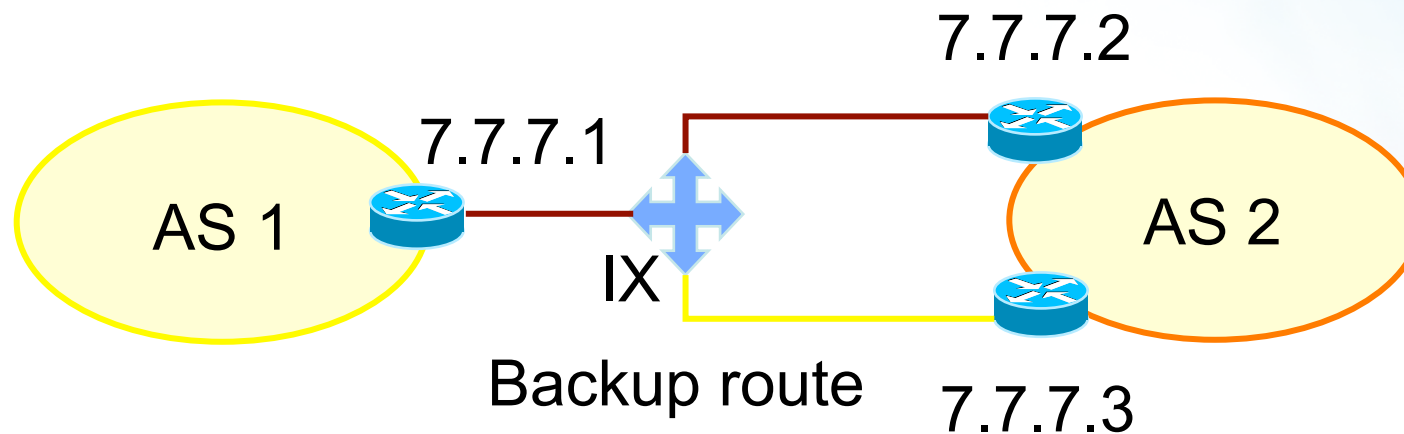
## Describing simple backup connections: AS1 (cont.)

```
aut-num: AS1
import:  from AS2 7.7.7.2 at 7.7.7.1 action pref=10;
         from AS2 7.7.7.3 at 7.7.7.1 action pref=20;
accept <^AS2+$>
```

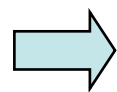
### Use of pref

- pref is opposite to local-pref
- Smaller values are preferred over larger values

## Describing simple backup connections: AS2



How to define AS2's routing policy of AS1's backup route?



multi exit discriminator metric (med) can be used

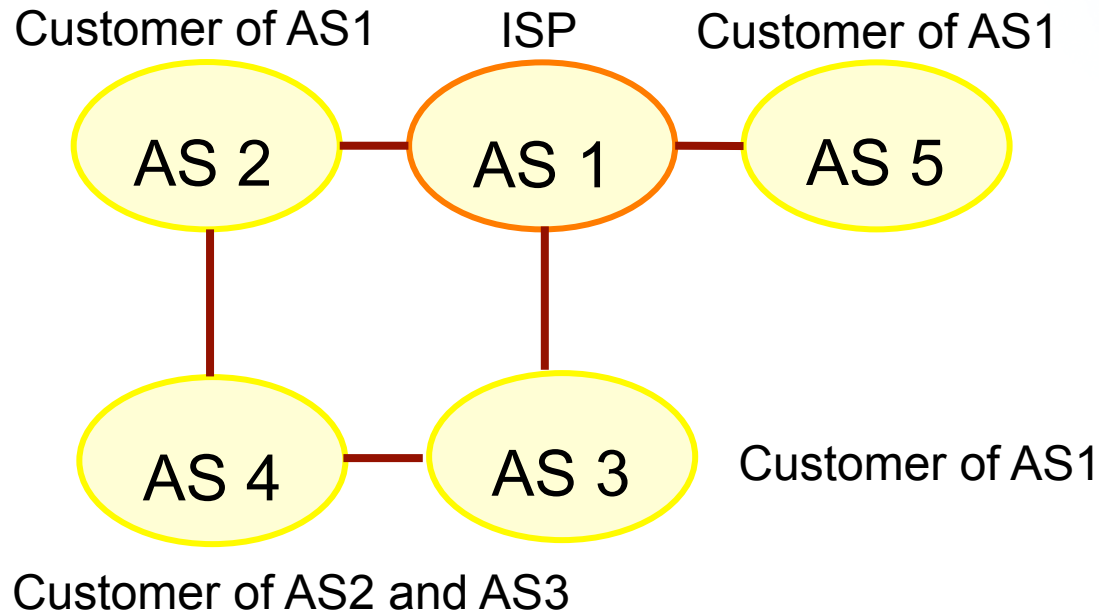
## Describing simple backup connections: AS2 (cont.)

```
aut-num: AS2
export: to AS1 7.7.7.1 at 7.7.7.2 action med=10;
        to AS1 7.7.7.1 at 7.7.7.3 action med=20;
announce <^AS2+$>
```

### Use of med

- Suitable for load balancing including backups

# Multihome routing policy



## AS1's base policy

- Only accepts routes from customers that are originated by the customer
- or by the customer's customers

## Multihome routing policies (cont.)

aut-num: AS1

import: from AS2 accept (AS2 or AS4) AND  
<^AS2+AS4\*\$>

import: from AS3 accept (AS3 or AS4) AND  
<^AS3+AS4\*\$>

import: from AS5 accept AS5 AND <^AS5+\$>

# Questions?

**Thank you!**