



deploy

IPv6 deployment considerations +
transitions

János Mohácsi
IPv6 workshop, Skopje
29-30 June 2011

Copy ...Rights

This slide set is the ownership of the 6DISS project via its partners

The Powerpoint version of this material may be reused and modified only with written authorization

Using part of this material must mention 6DISS courtesy

PDF files are available from www.6diss.org

Looking for a contact ?

- **Mail to : martin.potts@martel-consulting.ch**
- **Or bernard.tuy@renater.fr**



Droits d'auteur ...

L'ensemble des présentations utilisées dans le cadre de cet atelier est la propriété de 6DISS, représenté par ses différents partenaires.

La version Powerpoint des présentations peut être réutilisée et modifiée après qu'une autorisation écrite ait été obtenue

L'usage de tout ou partie de ce matériel doit mentionné que sa source est le projet 6DISS

La version PDF des présentations est disponible sur www.6diss.org

Pour tout contact :

- **Mail à Martin.Potts@martel-consulting.ch**
- **Ou Bernard.Tuy@renater.fr**

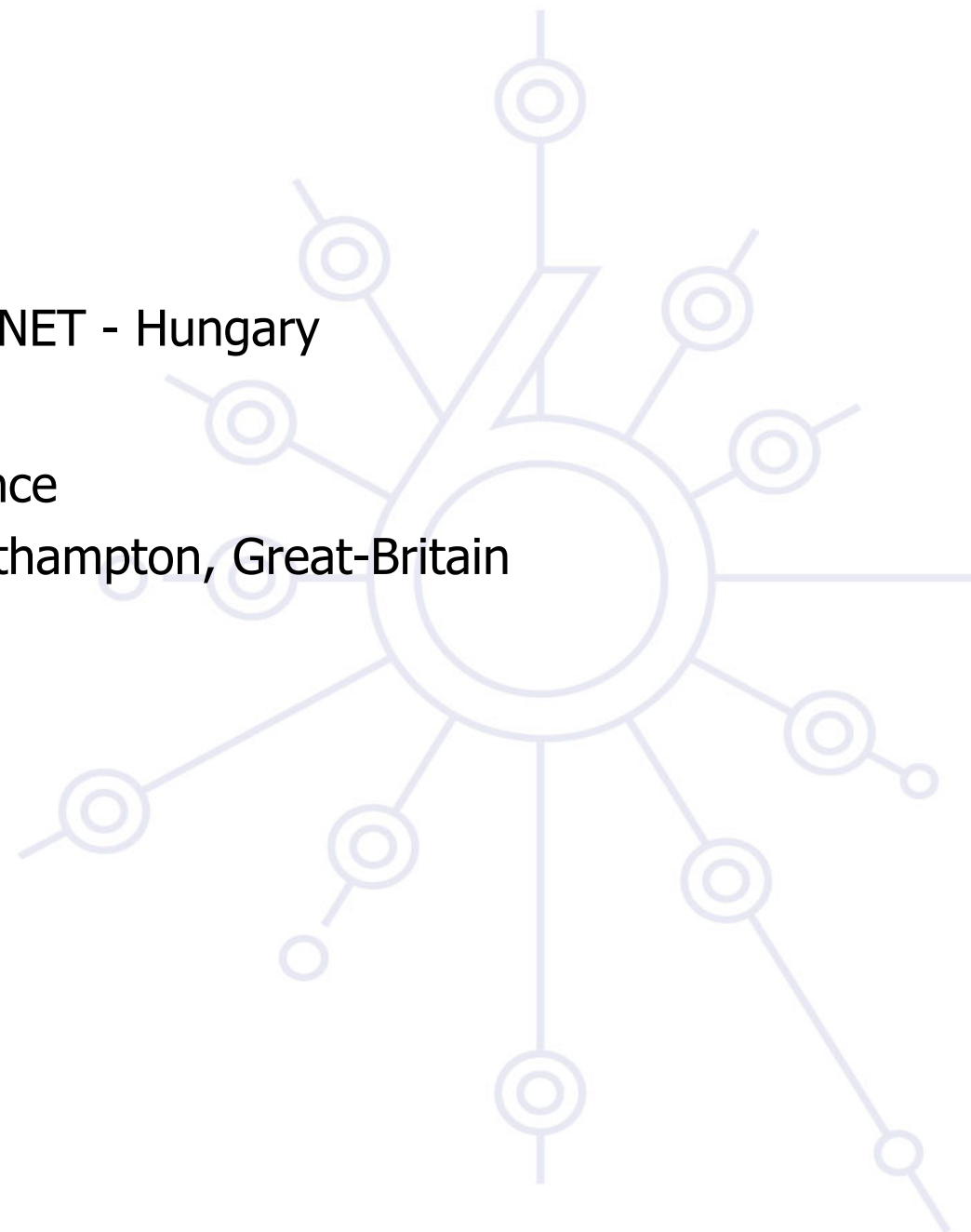
Contributions

Main authors

- János Mohácsi, NIIF/HUNGARNET - Hungary

Contributors

- Jérôme Durand, Renater, France
- Tim Chown, University of Southampton, Great-Britain
- B. Tuy, Renater, France



Updates

B. Tuy	19/11/2008
J. Mohácsi	12/03/2009
J. Mohácsi	30/10/2009
J. Mohácsi	24/02/2010



Warning ...

This module is under work (it's evolving still rapidly ...)

- ***here are ideas drawn from experienced people***
- ***it's out of scope to recommend every one to do the same***
- ***Every campus is specific and thinking what to do and how to do it beforehand is a must***

Good luck !

Outline

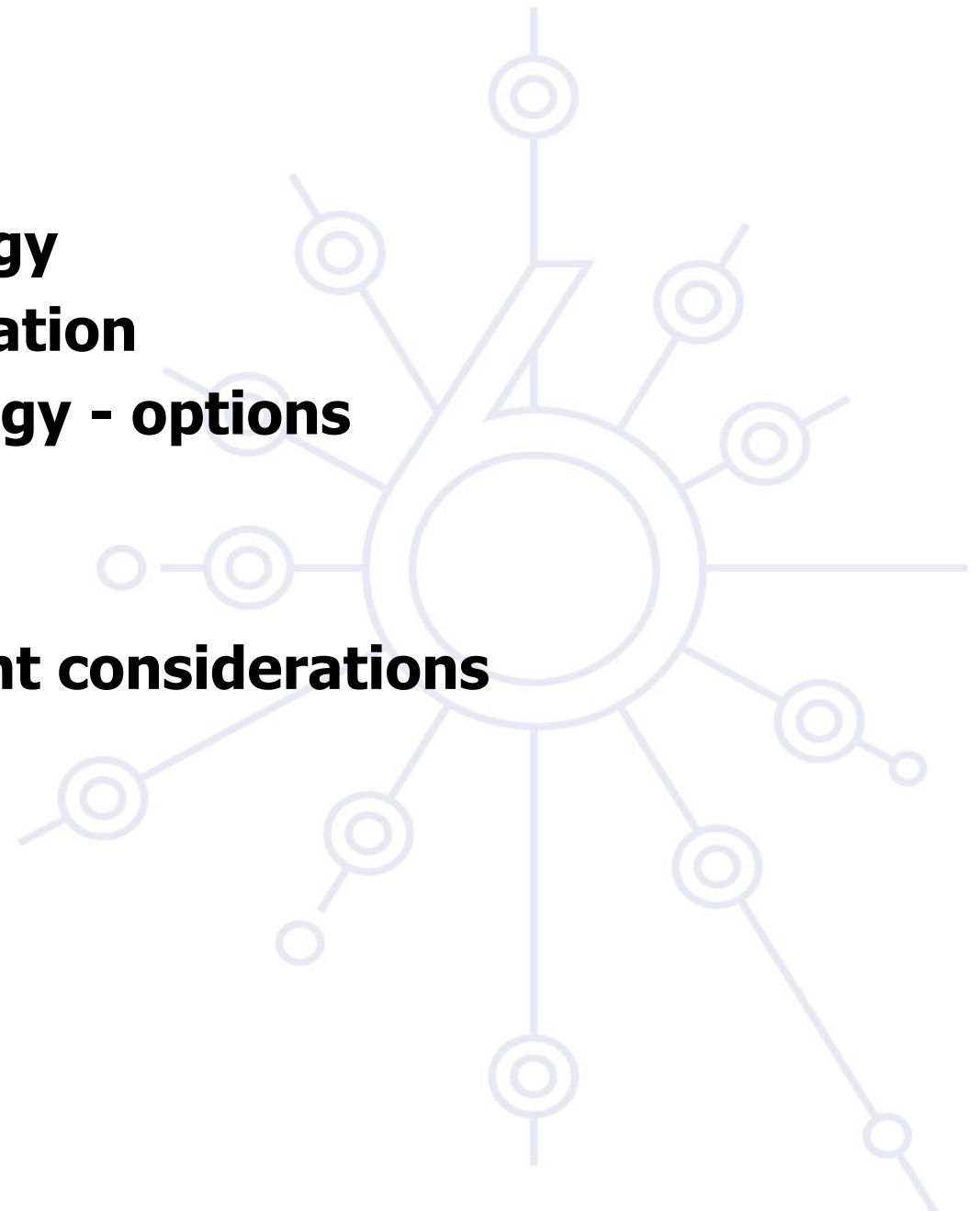
Campus deployment strategy

Campus IPv6 address allocation

Campus deployment topology - options

Campus services

Service provider deployment considerations



Outline

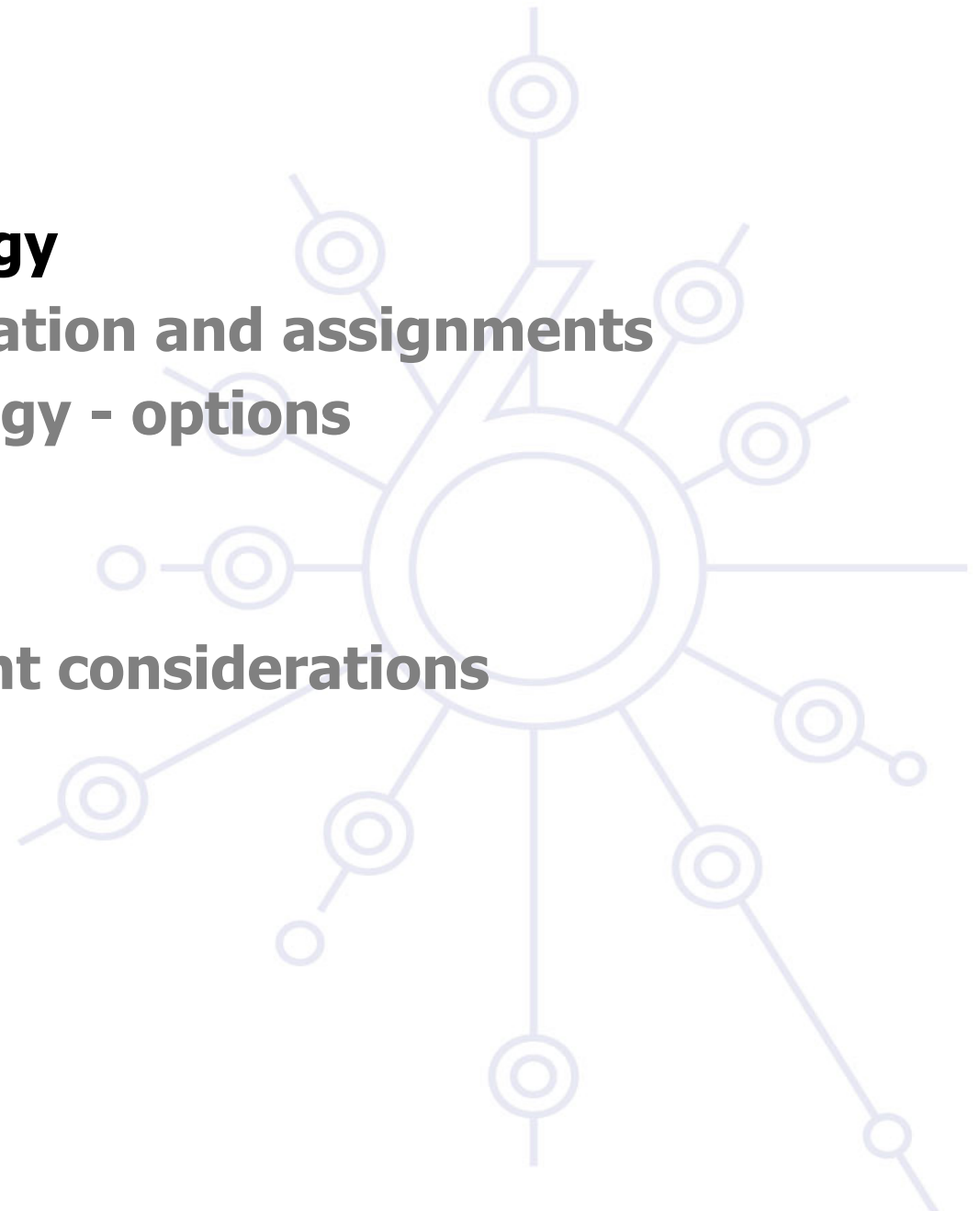
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

Campus services

Service provider deployment considerations



Questions to solve

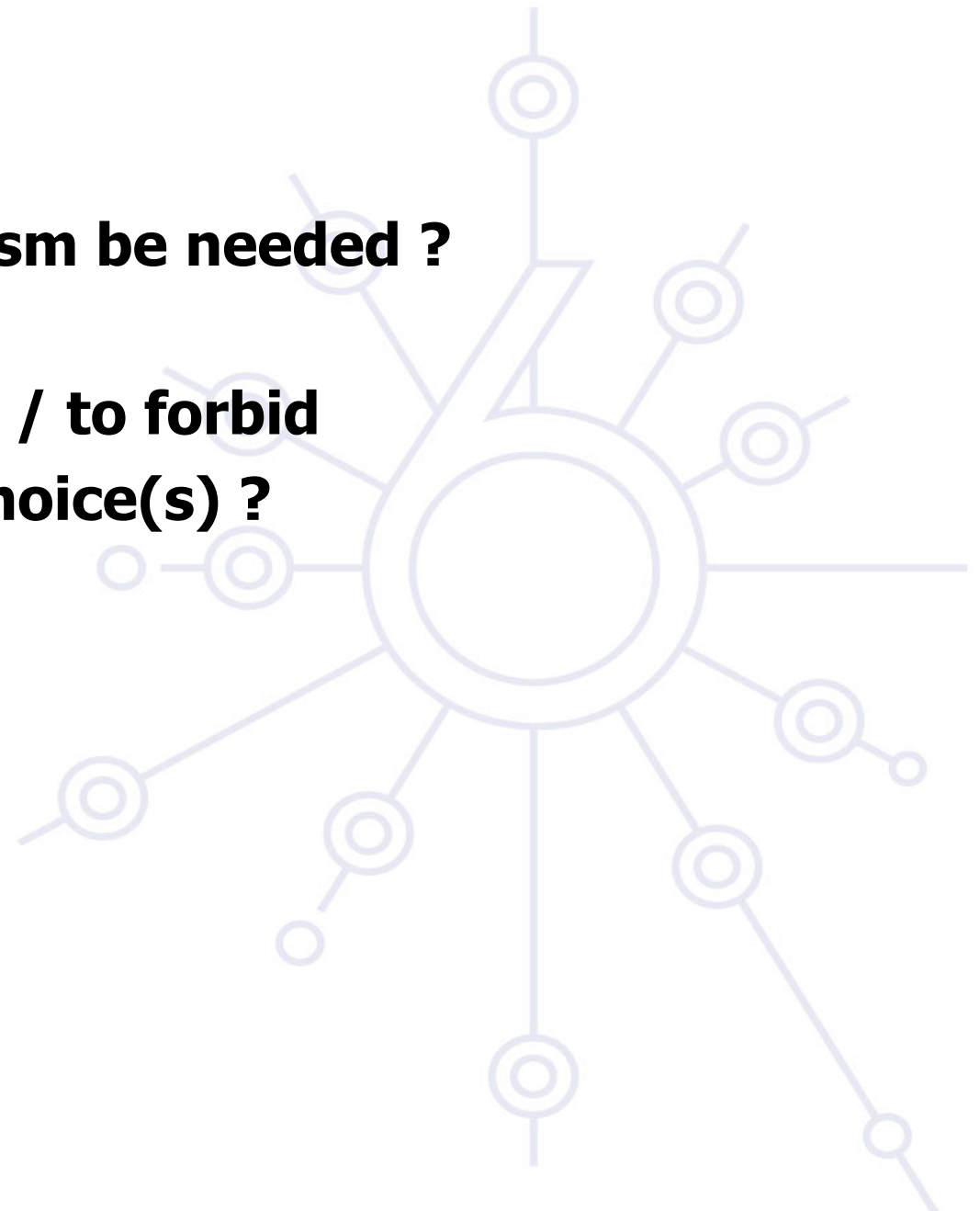
Will a coexistence mechanism be needed ?

If yes,

**Which mechanism to use / to forbid
reasons of this (these) choice(s) ?**

What policy to specify ?

How to implement it ?



Various Campus transition approaches

*IPv4 will be used for years after IPv6 has been deployed
Then both versions of the IP protocol will have to coexist*

Dual Stack

- Servers/clients speaking both protocols
- Application/service can select either protocol to use

Tunneling (“connecting IPv6 clouds”)

- IPv6 packet is data payload of IPv4 packet/or MPLS frames

Translation methods (“IPv4<->IPv6 services”)

- Layer 3: Rewriting IP header information (NAT-PT)
- Layer 4: Rewriting TCP headers
- Layer 7: Application layer gateways (ALGs)

Benefits of dual-stack deployment

By deploying dual-stack, you can test IPv6-only devices/services without disrupting IPv4 connectivity

Dual stack IPv6 + IPv4 NAT: legacy IPv4 applications (email, www) can be used next to new IPv6 applications (p2p, home networking, ...)

- IPv6 offers the next generation of applications

1: Dual-stack

Support both protocols on selected links (and nodes)

Requires support in:

- Host platforms
- Router platforms
- Applications and services
 - e.g. web, DNS, SMTP

Adds considerations for

- Security in all components
- New policies dependent on IPv6-specific features

Can run global IPv6 alongside NAT-ed IPv4

Dual-stack issues

Application must choose which IP protocol to use

- DNS returns IPv4 (A record) and IPv6 addresses (AAAA record)
- e.g. MS Internet Explorer prefers IPv6
- Don't advertise AAAA record for a host unless you have good IPv6 connectivity (for all services on host)

Enabling IPv6 should not adversely impact IPv4 performance

- Consider whether IPv6 tunnels use router CPU for example

Security should be no worse

- Hosts listen on both protocols; secure both

Aside: IPv4 mapped addresses

An IPv6 address used to represent an IPv4 address

A socket API may receive an IPv4 connection as an IPv6 address, known as an IPv4-mapped address

- Format is `::ffff:<ipv4-address>`
- e.g. `::ffff:152.78.64.1`

NB: This is one socket for both address families

Should not be seen 'on the wire', i.e. not as source or destination address

May appear in log files, depending on how the application handles a connection

Typically seen in dual-stack deployments

2:Tunnelling

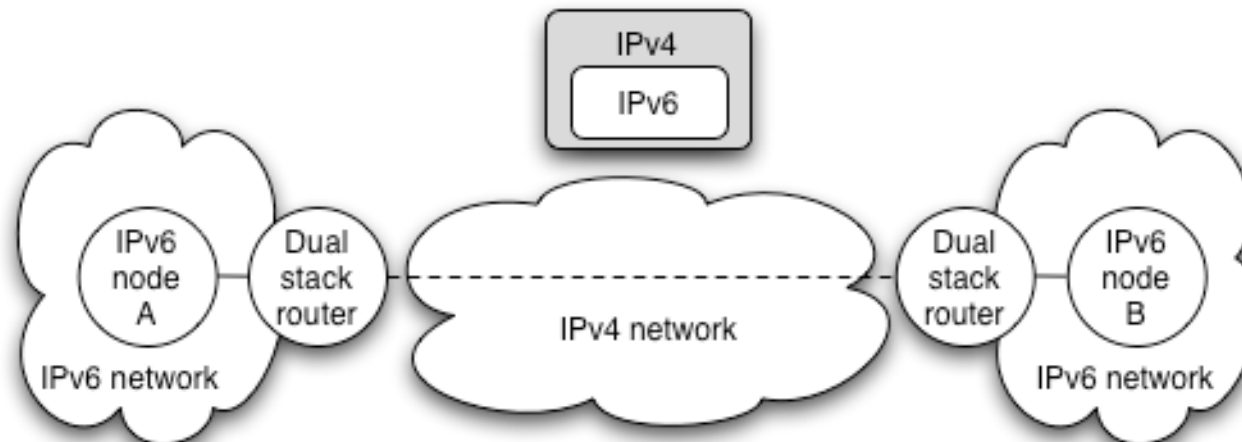
Initially IPv6 in IPv4, (much) later IPv4 in IPv6

So, IPv6 packets are encapsulated in IPv4 packets

- IPv6 packet is payload of IPv4 packet

Usually used between edge routers to connect IPv6 'islands'

- Edge router talks IPv6 to internal systems
- Encapsulates IPv6 in IPv4 towards remote tunnel endpoint



Packet delivery over the tunnel

IPv6 node A sends packet to IPv6 node B

- Routed internally to edge router A

Edge router A sees destination network B is reachable over tunnel interface

- Encapsulates IPv6 packet in IPv4 packet(s)
- Sends resulting IPv4 packet(s) to edge router B
- Delivered over existing IPv4 Internet infrastructure

Edge router B decapsulates IPv6 packet from payload of received IPv4 packet

- Packet routed internally in network B to node B
- Node B receives the IPv6 packet

Fragmentation

IPv6 requires that packet fragmentation only occurs at end systems, not on intermediate routers

- Use Path Maximum Transmission Unit (PMTU) Discovery to choose the MTU
- Achieved using special ICMP messages
- Minimum MTU is 1280 bytes in IPv6

When tunnelling IPv6 in IPv4, the IPv4 packets may be fragmented

- Depends on the IPv4 packet size
- Additional IPv6 headers (e.g. Authentication Header) will affect this

Manual or automatic?

Can create tunnels manually or automatically

Manual tunnels

- Requires manual configuration, at both ends
 - Usually just one command/config line in the router at each end
 - Agreement on addresses to use for interfaces
- Good from a management perspective: you know who your tunnels are created with

Automatic tunnelling

- Tunnels created on demand without manual intervention
- Includes 6to4/6rd (RFC3056/RFC5569 and 5969)
 - Quite popular in SOHO deployments
- Also: ISATAP and Teredo

Configured tunnels

Very easy to setup and configure

Good management potential

- ISP configures all tunnels, so is in control of its deployment
- This is the current approach used by many NRENs (including UKERNA and RENATER) to connect academic sites/users over IPv6 where native IPv6 connectivity is not available

Usually used router-to-router or host-to-router

- Desirable to allow end user to register (and subsequently authenticate) to request a tunnel
- The IPv6 Tunnel Broker (RFC3053) offers such a system, usually for host-to-router connectivity, but sometimes for router-to-router.

Tunnel broker

Very popular in IPv6 user community

Most well-known broker is www.freenet6.net

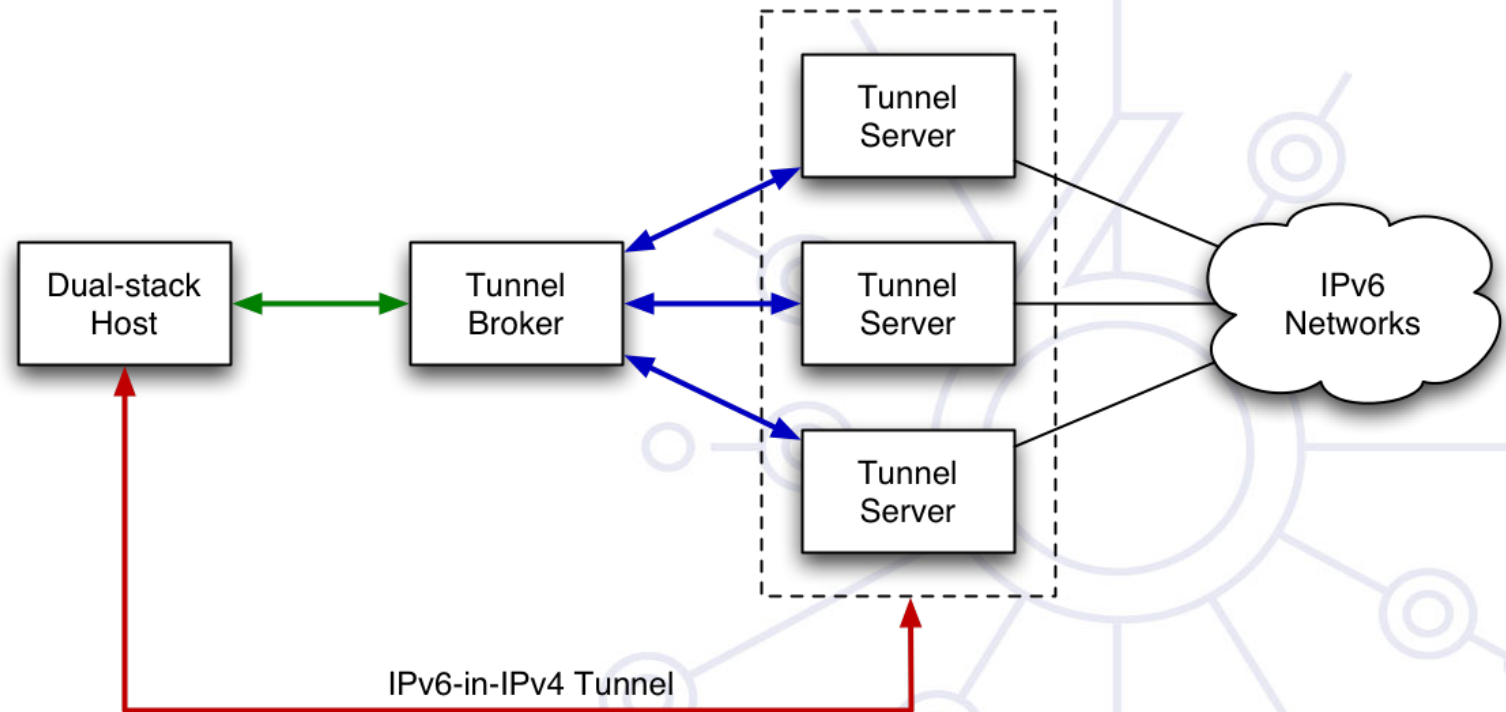
- Hosted in Canada by Hexago

General mode of operation is:

- User/client registers with the broker system
- A tunnel is requested from a certain IPv4 address
- The broker sets up its end of the requested tunnel on its tunnel server
- The broker communicates the tunnel settings to the user, for client-side configuration

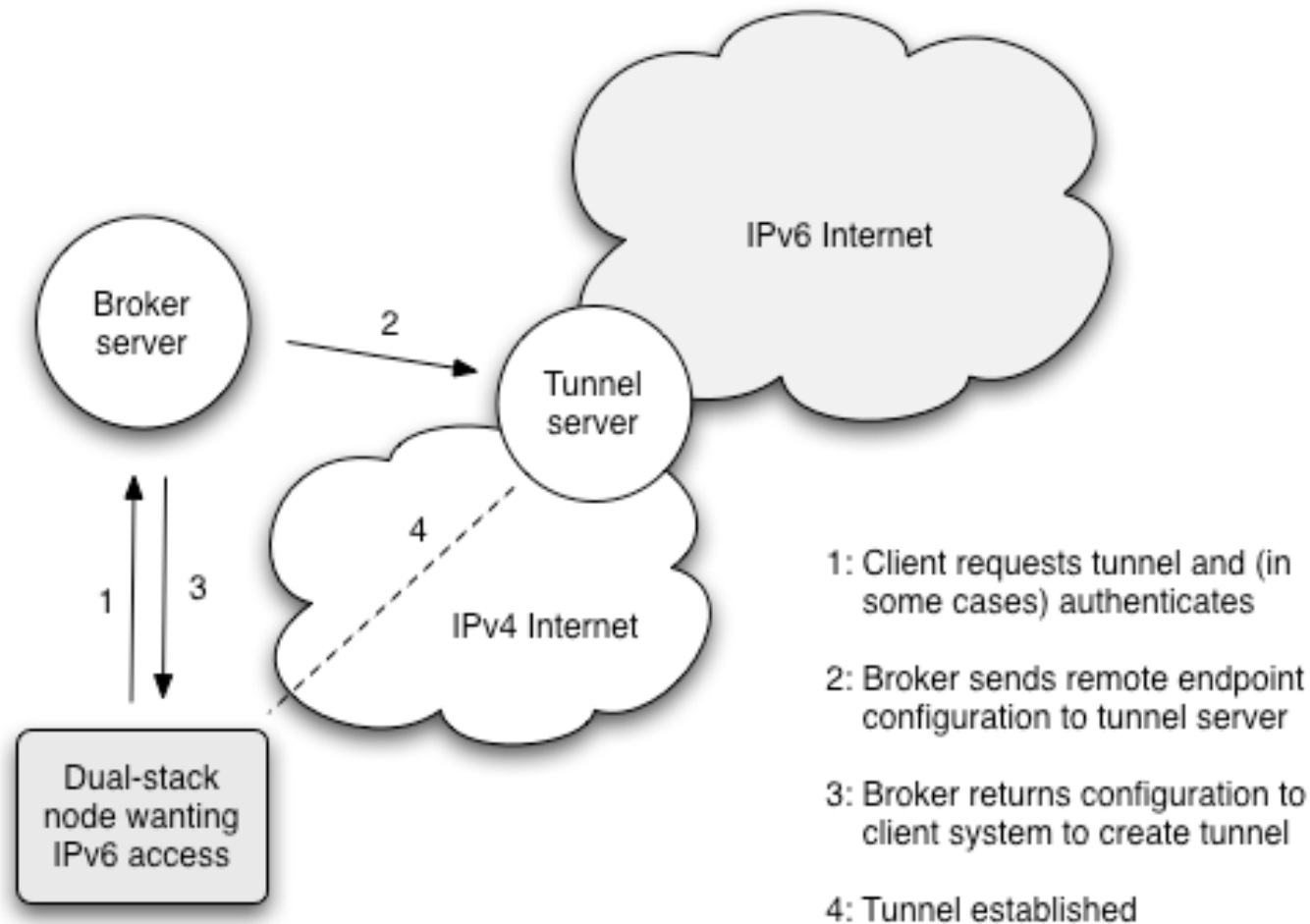
Can traverse a NAT, e.g. if UDP tunnelling used

Broker: systems view



1. User connects to Tunnel Broker web interface requesting tunnel
2. TB returns script to create tunnel to the Tunnel Server, and informs TS of new client
3. Client executes script, and gains access to IPv6 networks via the TS

Broker: Logical view



Broker issues

Broker's key advantage is its manageability

- ISP can track usage levels

A few downsides:

- If broker is topologically remote, round trip times for data may suffer
 - e.g. using freenet6 in Canada to reach UK sites
- Not well-suited if IPv4 address is dynamic
 - Common problem in home DSL networks
- Client tool required to operate through a NAT
- If using a remote tunnel broker, your own ISP may not perceive a demand for IPv6

6to4

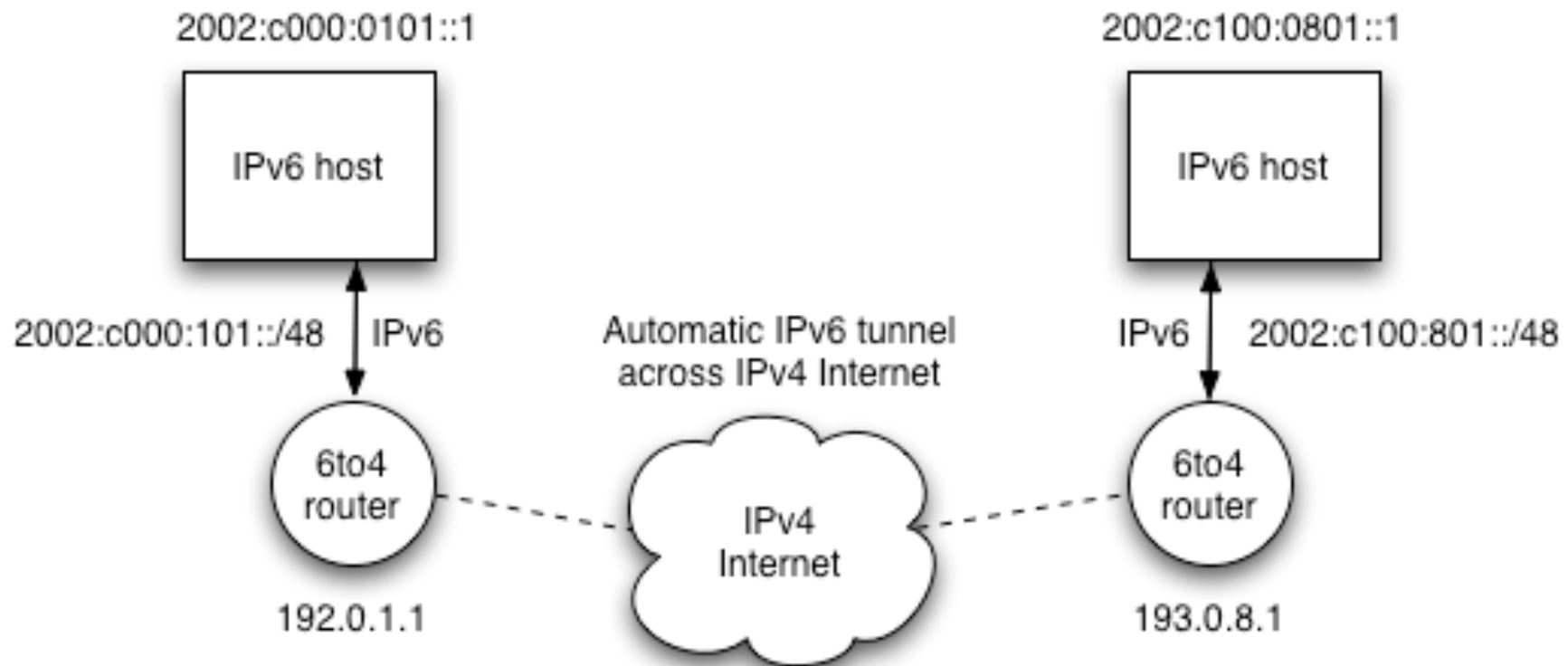
In its basic configuration, 6to4 is used to connect two IPv6 islands across an IPv4 network

Uses special 'trick' for the 2002::/16 IPv6 prefix that is reserved for 6to4 use

- Next 32 bits of the prefix are the 32 bits of the IPv4 address of the 6to4 router
- For example, a 6to4 router on 192.0.1.1 would use an IPv6 prefix of 2002:c000:0101::/48 for its site network

When a 6to4 router sees a packet with destination prefix 2002::/16, it knows to tunnel the packet in IPv4 towards the IPv4 address indicated in the next 32 bits

6to4 basic overview



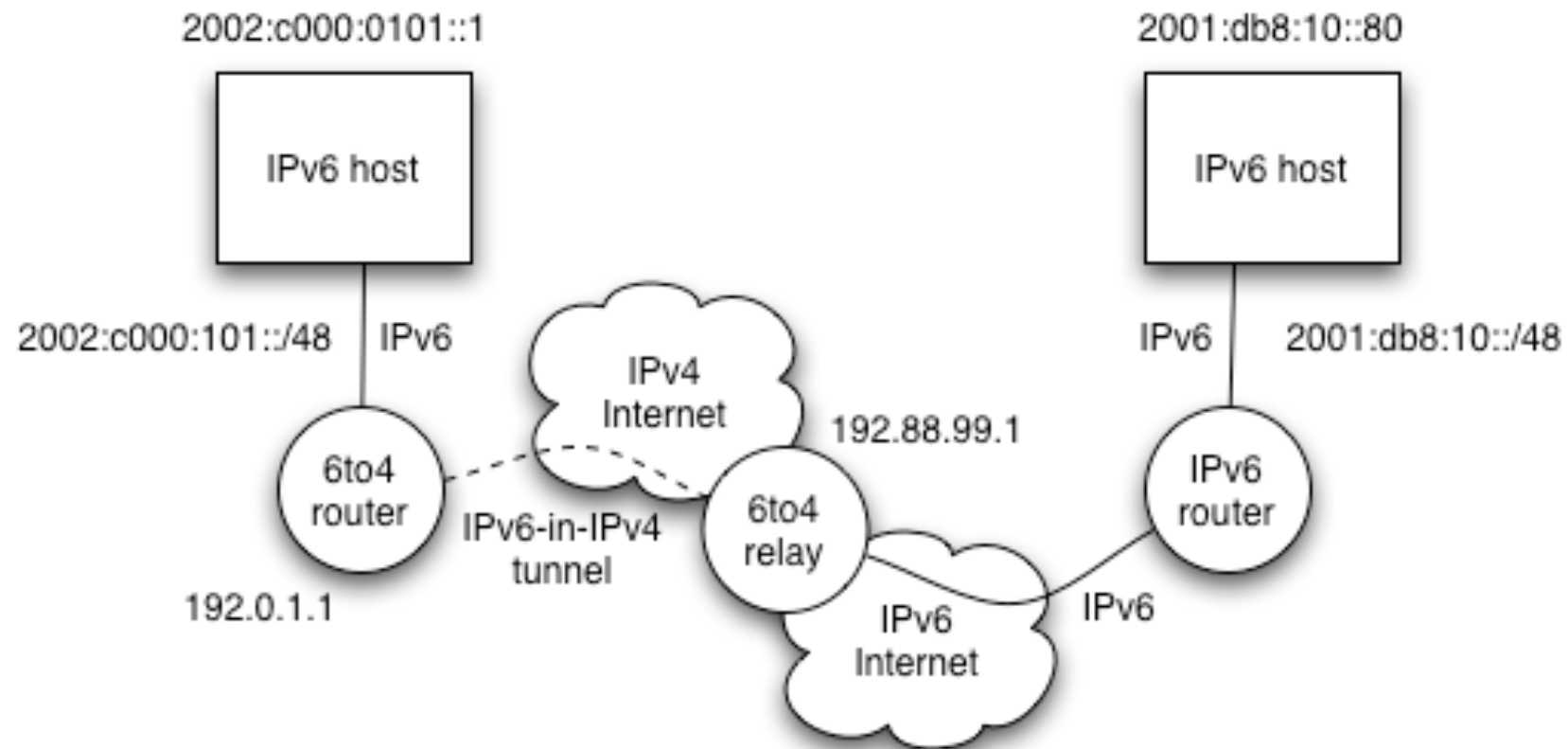
6to4 relay

A 6to4 relay has a 6to4 interface and a 'real' IPv6 interface

Two cases to consider:

- IPv6 packets sent from a 6to4 site to a destination address outside 2002::/16 are tunnelled using 6to4 to the relay, are decapsulated, and then forwarded on the relay's 'real' IPv6 interface to the destination site
 - The 6to4 relay is advertised on a well-known IPv4 anycast address 192.88.99.1.
- IPv6 packets sent from a 'real' IPv6 site towards an address using the 2002::/16 prefix (a 6to4 site) are routed to the 6to4 relay and then tunnelled using 6to4 to the destination 6to4 site
 - The relay advertises 2002::/16 to connected IPv6 neighbours

6to4 with relay



6to4 issues

In principle 6to4 is attractive

- But there are operational concerns

Problem 1: possible relay abuse

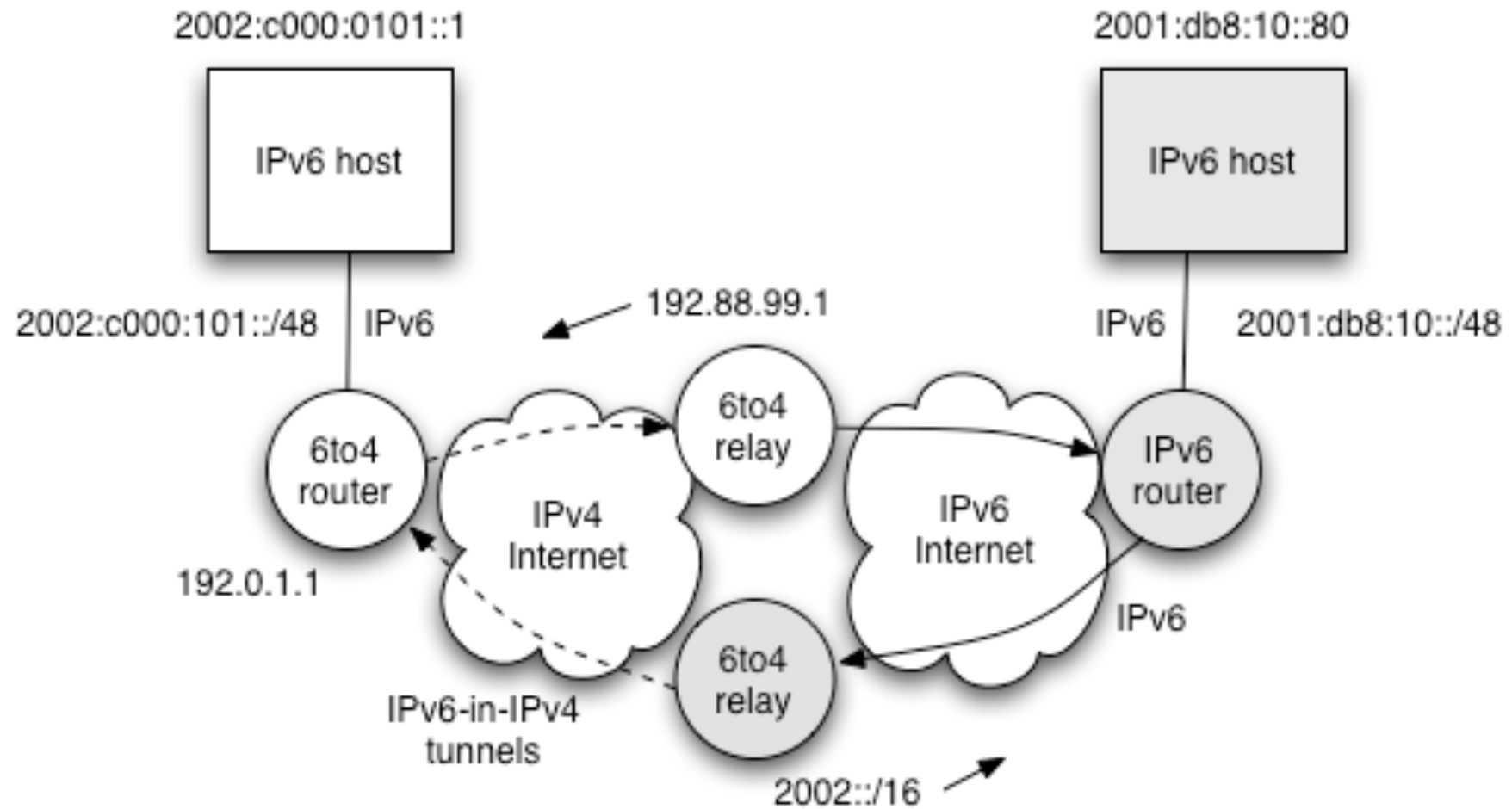
- Relay could be used for a DoS attack
- Tunnelled IPv6 traffic addresses may be spoofed

Problem 2: asymmetric model/reliability

- The 6to4 site may use a different 6to4 relay to the 'real' IPv6 site
- One of the sites may not see a 6to4 relay at all, if ISPs choose to only deploy relays for their own customers, and thus filter routing information

But for 6to4 relay to 6to4 relay operation, it's good

Asymmetric 6to4



ISATAP

Intra-Site Automatic Tunnel Addressing Protocol (RFC4214)

- Automatic tunneling
- Designed for use *within* a site
- Used where dual-stack nodes are sparsely deployed in the site (very early deployment phase)

Host-to-host or host-to-router automatic tunnels

- Uses a specific EUI-64 host address format
- Format can be recognised and acted upon by ISATAP-aware nodes and routers

The EUI-64 is formed by

- A reserved IANA prefix (00-00-5e)
- A fixed 8-bit hex value (fe)
- The 32-bit IPv4 address of the node
- Toggling the globally unique (u) bit

3: Translation

When an IPv4-only system needs to communicate with an IPv6-only system, translation is required

Can be done at various layers

Network layer

- Rewrite IP headers

Transport layer

- Use a TCP relay

Application layer

- Use an application layer gateway (ALG)

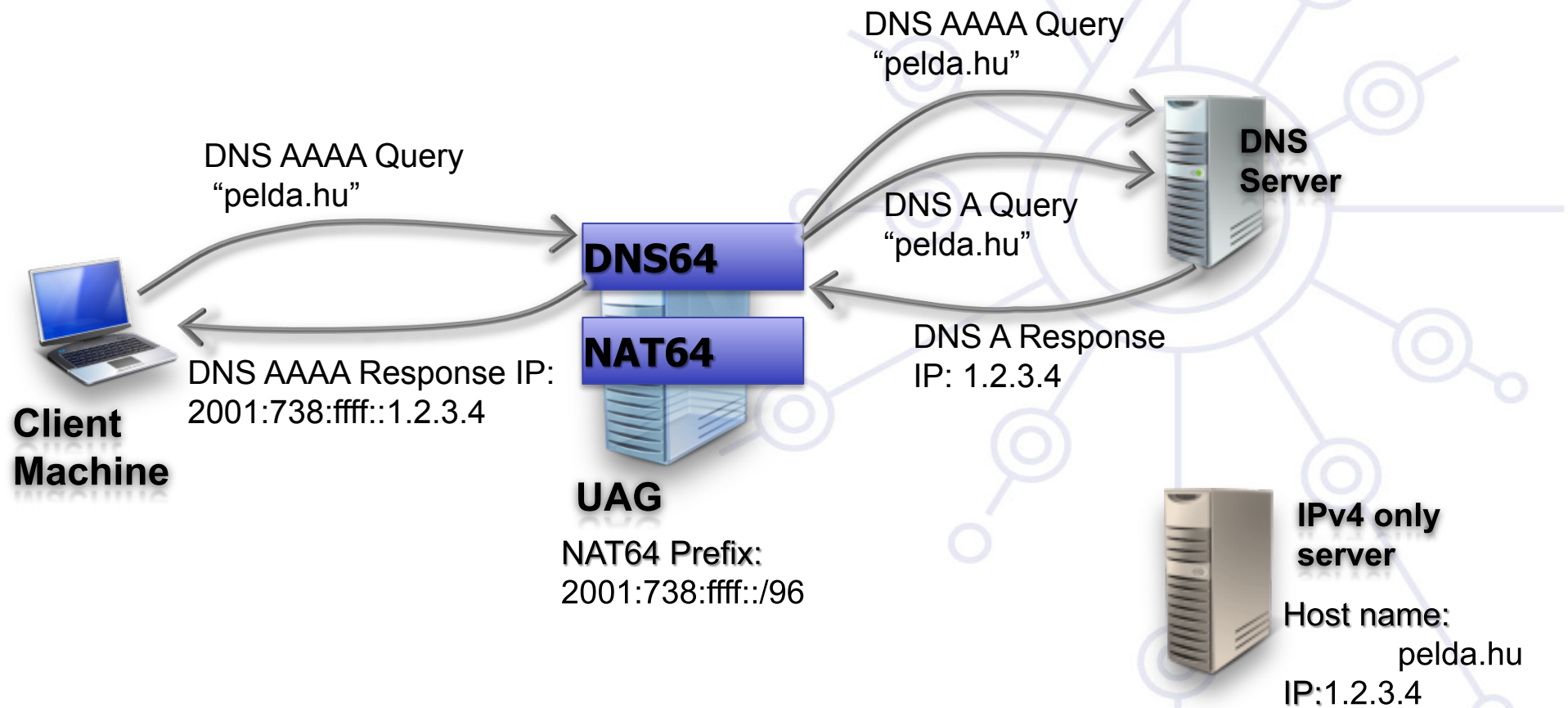
Ideally avoid translation

- Use IPv4 to speak to IPv4 systems and IPv6 for IPv6 systems

NAT64, DNS64 /1

1. IPv6only client asks a IPv4 only service via DNS

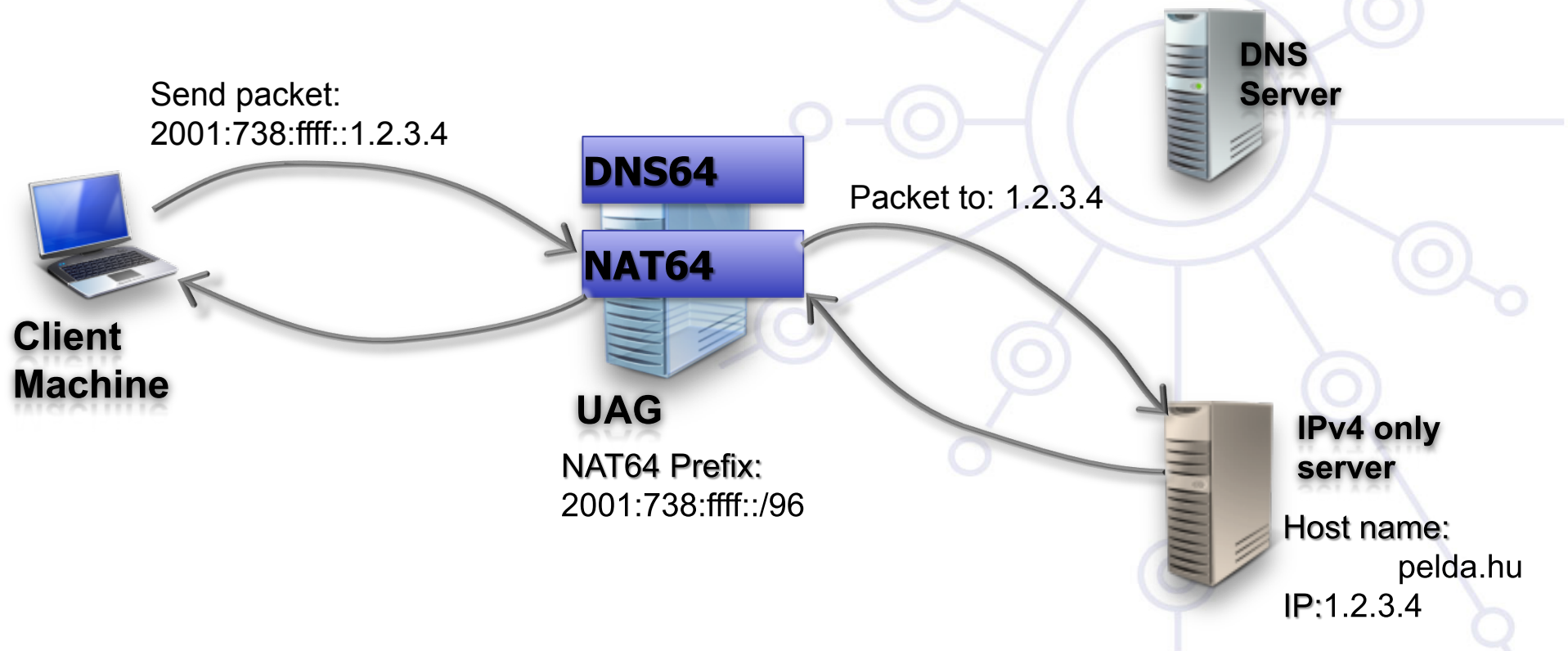
DNS64: 2001:738:0:3::2



NAT64, DNS64 /2

2: IPv6 only clients sends packets to IPv4-only servers
ecdsys

DNS64: 2001:738:0:3::2



Campus deployment plan /1

1. Obtain global IPv6 address space from your ISP

- LIRs usually have a /32 prefix from RIPE NCC/RIRs (e.g. NRENs)
- Customers will get a /48 prefix from LIRs (e.g. Universities)

2. Obtain external connectivity

- You can do dual-stack connectivity
- Many universities will use a tunnel to get IPv6 service
 - in this case be sure that nobody can abuse your tunnel – use filtering

Campus deployment plan /2

3. Internal deployment

- Determine an IPv6 firewall/security policy
 - The IPv4 firewall/security policy is a good start
- Develop an IPv6 address plan for your site
- Determine an address management policy (RA/DHCPv6/Static?)
- Migrate to dual-stack infrastructure on the wire
 - Network links become IPv6 enabled
- Enable IPv6 services and applications
 - Starting with DNS
- Enable IPv6 on host systems (Linux, WinXP, Vista, Mac OS X...)
- Enable management and monitoring tools

Outline

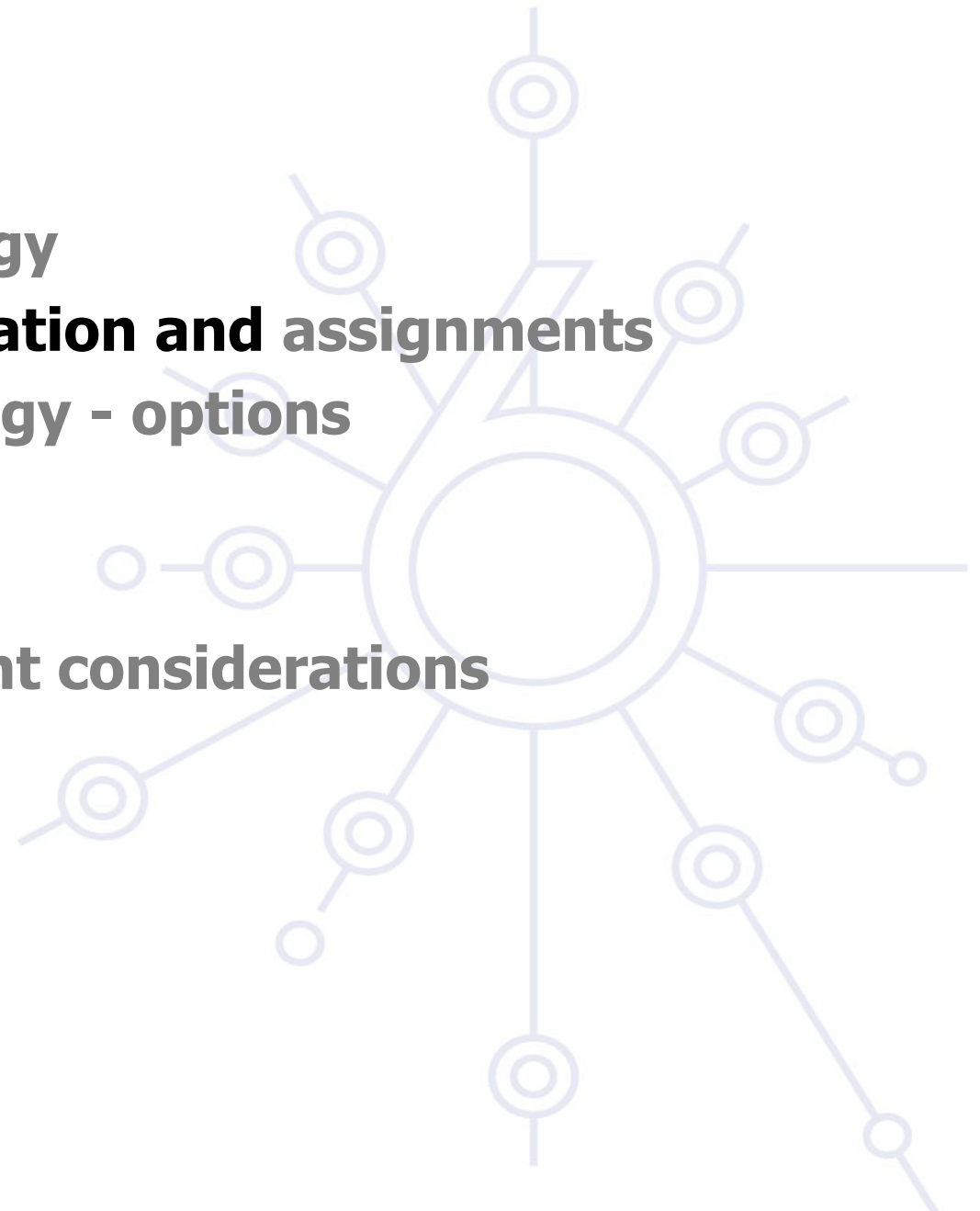
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

Campus services

Service provider deployment considerations



Goals of IPv6 addressing plan

Easier security policy implementation

Easier address source tracing

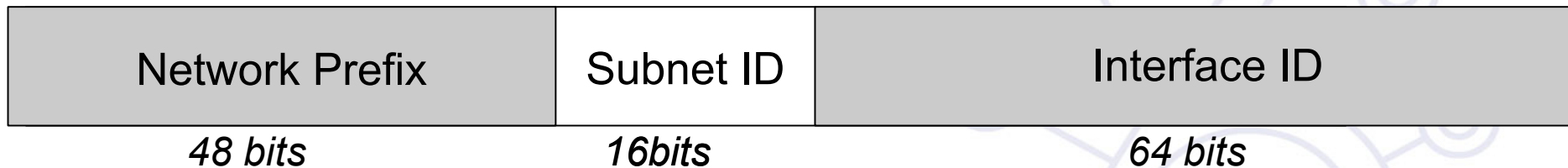
More scalable than with IPv4

Enable better network management



Campus Addressing

Most sites will receive /48 assignments:



16 bits left for subnetting - what to do with them?

Two main questions to answer:

⇒ **How many topologically different “zones” can be identified ?**

- Existing ones or new ones to be created for whatever (good) reason

⇒ **How many networks (subnets) are needed within these zones ?**

Example network « zones »

Zone description	Nb of subnets
Upstream interco and infrast	16
Administration services	4
Medical Sciences dept	32
Dept A	16
Dept B	16
...	

Campus Addressing - site level subnetting - methods -1

1. Sequentially, e.g.

- 0000
- 0001
- ...
- FFFF

- 16 bits = 65535 subnets

⇒ Reserve prefixes for further allocation

0020/60

0030/60

Subnet ID	Zone description
0000 / 60	BB Infrastructure
0010 / 60	Administration
0020 / 59	Medical Sciences dept
0040 / 60	Dept A
0050 / 60	Dept B
...	...

Campus Addressing - site level subnetting - methods 2

2. Following existing IPv4:

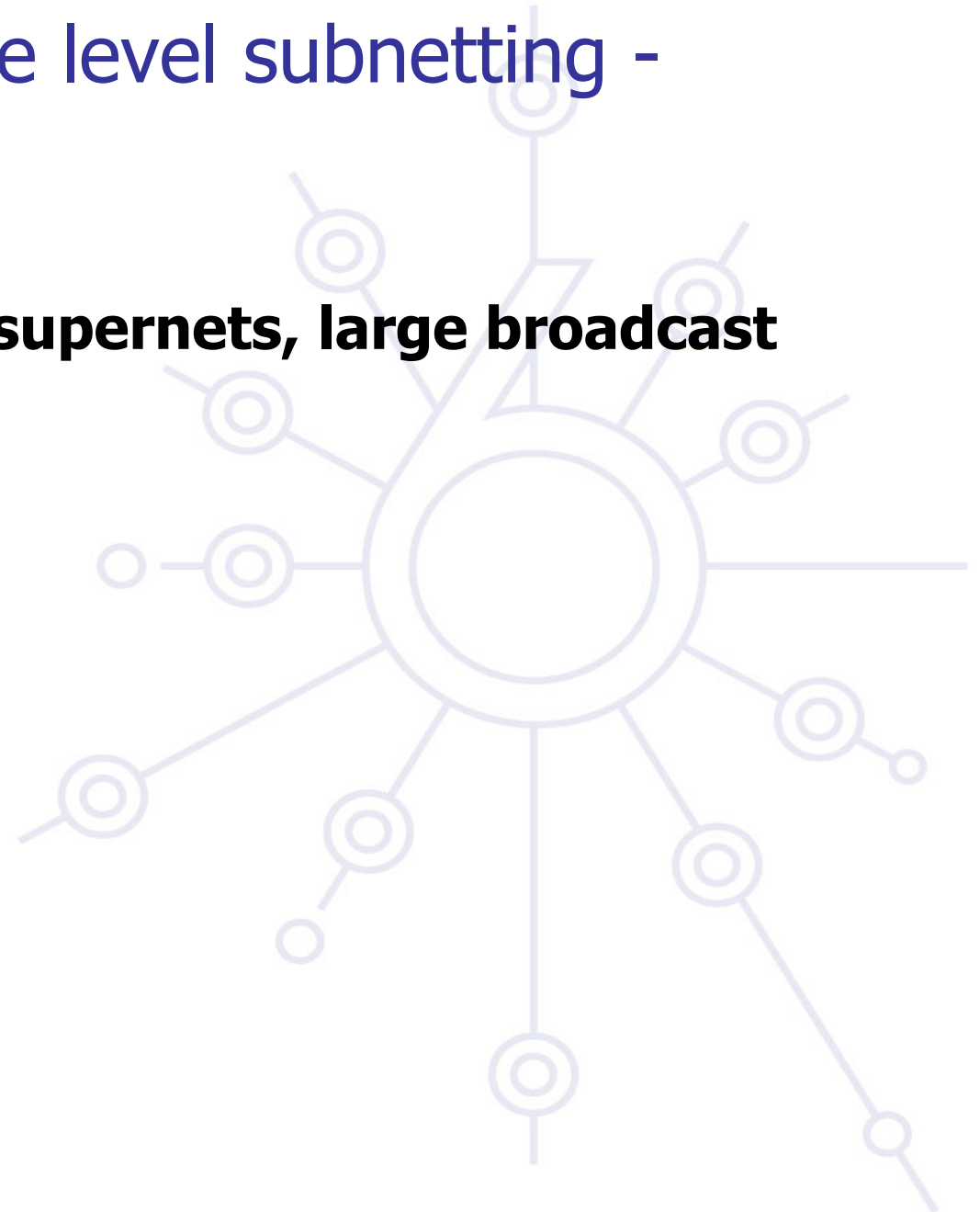
- Subnets or combinations of nets & subnets, or VLANs, etc., e.g.
- IPv4 subnets:
 - 152.66.**60**.0/24 003c
 - 152.66.**91**.0/24 005b
 - 152.66.**156**.0/24 009c
- VLANs:
 - VLAN id 100 0100 (w/o decimal/hex conversion)
 or 0064 (w dec/hex conversion)

⇒ *Best to start thinking about it*

Campus Addressing - site level subnetting - methods 3

3. Topological/aggregating reflecting wiring plants, supernets, large broadcast domains, etc.

- Main library = 0010/60
 - Floor in library = 001a/64
- Computing center = 0200/56
 - Student servers = 02c0/64
- Medical school = c000/52
- and so on. . .



Campus Addressing - site level subnetting – methods 4

Location-Use Type oriented subnetting

Network Prefix	Location	Purpose	Subnetting	Interface ID
----------------	----------	---------	------------	--------------

Location	Purpose	Subnetting	Description
0/52			Building A
	00/56		Servers
	01/56		Students
		0100/64	Students lab 1
		0101/64	Students lab 2
1/52			Building B
	10/56		Grid server
		1000/64	Frontends to Grid
		1001/64	Computational node set 1
		1002/64	Computational node set 2
3/52			Non-location based networks
	30/56		VPNs

Location: 4-8 bits

Purpose: 4-8 bits

Subnetting: 4-8 bits

Purpose and
location field can be
swapped

Example network - topological aggregation + sequential allocation

Zone description	Nb of subnets
Upstream interco and infrast	16
Administration services	4
Medical Sciences dept	32
Dept A	16
Dept B	16
...	

IPv6 subnet prefix allocations (ex.)

Subnet ID	Subnet prefix allocation	Description
0000 / 60		BB Infrastructure
	0000/64	Upstream interconnection
	0001/64	Campus architecture (DMZ)
	...	
	000B/64	Campus architecture
	...	
	000F	...
0010 / 60		Administration
	0010/64	Campus interco
	0011/64	Registration
	0012/64	Finance dept

IPv6 subnet prefix allocations ex. /2

Subnet ID	Subnet prefix allocation	Description
0020 / 60		Medical Sciences dept
	0020/64	Upstream interconnection
	0021/64	Nobel group
	...	
0030 / 60	Reserved	Medical Sciences dept
0040 / 60		Dept A
...		...

New Things to Think About

You can use “all 0s” and “all 1s”! (0000, ffff)

You’re not limited to the usual 254 hosts per subnet!

- LANs with lots of L2 switch allow for larger broadcast domains (with tiny collision domains), perhaps thousands of hosts/LAN...

No “secondary address” (though >1 address/interface)

No tiny subnets either (no /30, /31, /32)

- plan for what you need for backbone blocks, loopbacks, etc.

You should use /64 per links

- Especially if you plan to use autoconfiguration!
- If you allocate global addressess interconnection links - not necessary in every case

New Things to Think About /2

Every /64 subnet has far more than enough addresses to contain all of the computers on the planet, and with a /48 you have 65536 of those subnets

- use this power wisely!

With so many subnets your IGP may end up carrying thousands of routes

- consider internal topology and aggregation to avoid future problems.

Start thinking of better structure of your network...

New Things to Think About /3

Renumbering will likely be a fact of life. Although v6 does make it easier, it still isn't pretty. . .

- Avoid using numeric addresses at all costs
- Avoid hard-configured addresses on hosts except for servers (this is very important for DNS servers) – use the feature that you can assign more than one IPv6 address to an interface (IPv6 alias address for servers)
- Anticipate that changing ISPs will mean renumbering
- An ISP change will impact the first 48 bits, you can keep the last 80 unchanged in every host/server's address.

Address conservation usually not an issue

DHCPv6 might help

More discussion about the subnet sizes

/48 – Organisation/site

/64 – Subnets

/128 – hosts

Links subnet sizes:

Link local only: can be problematical with traceroute6 – ipv6 unnumbered

/127: the all-zeros address is supposed to be the any router anycast address although this is not widely implemented today - see more RFC 3627, RFC 6164

/126: works although there are some address reserved for anycast stuff

/120: no clashes with anycast addresses

/112: alignment is on a nice colon boundary

/64: based on RFC 3513, Allows to use EUI-64 addressing
advisable for point-multipoint and broadcast link scenarios

Outline

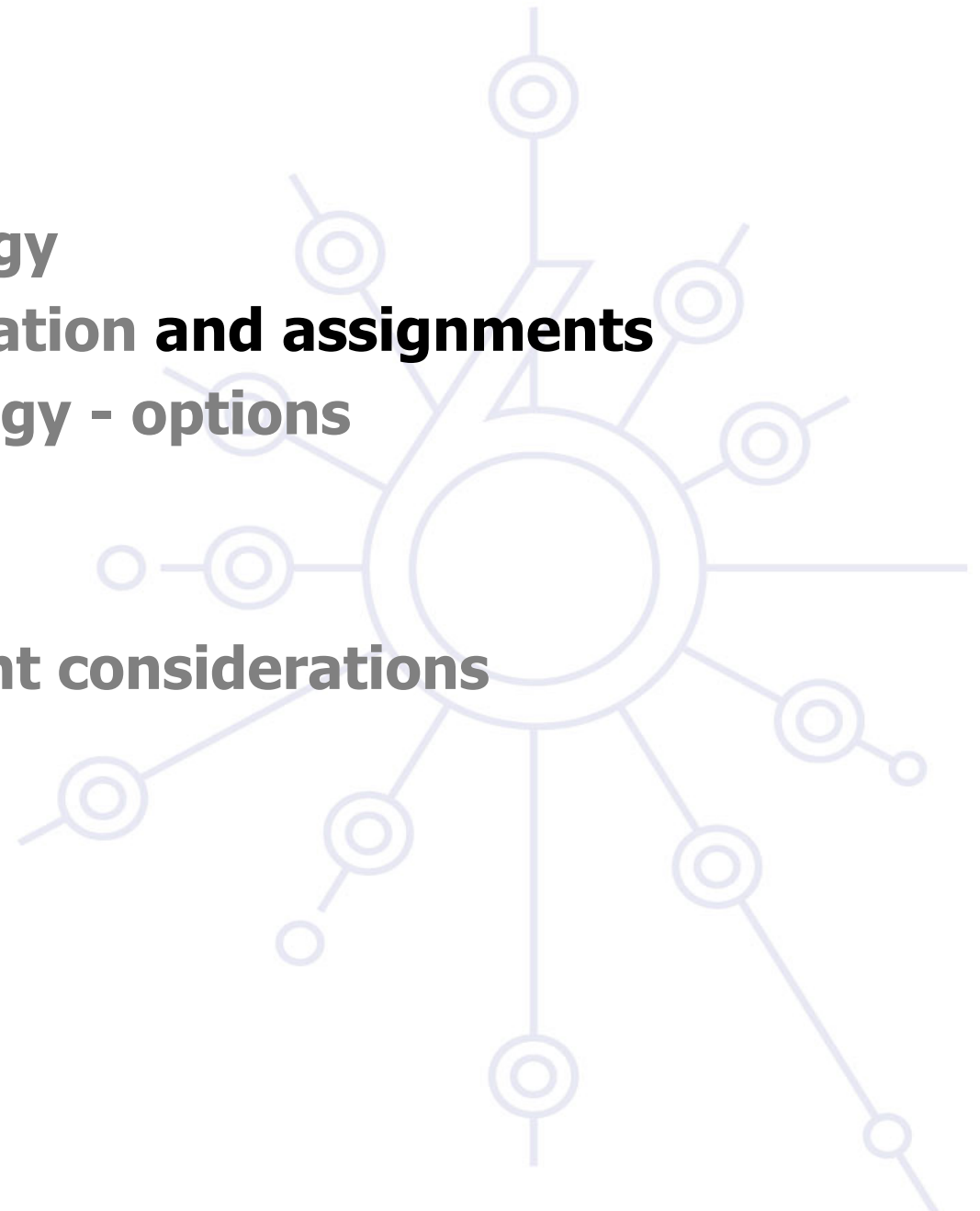
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

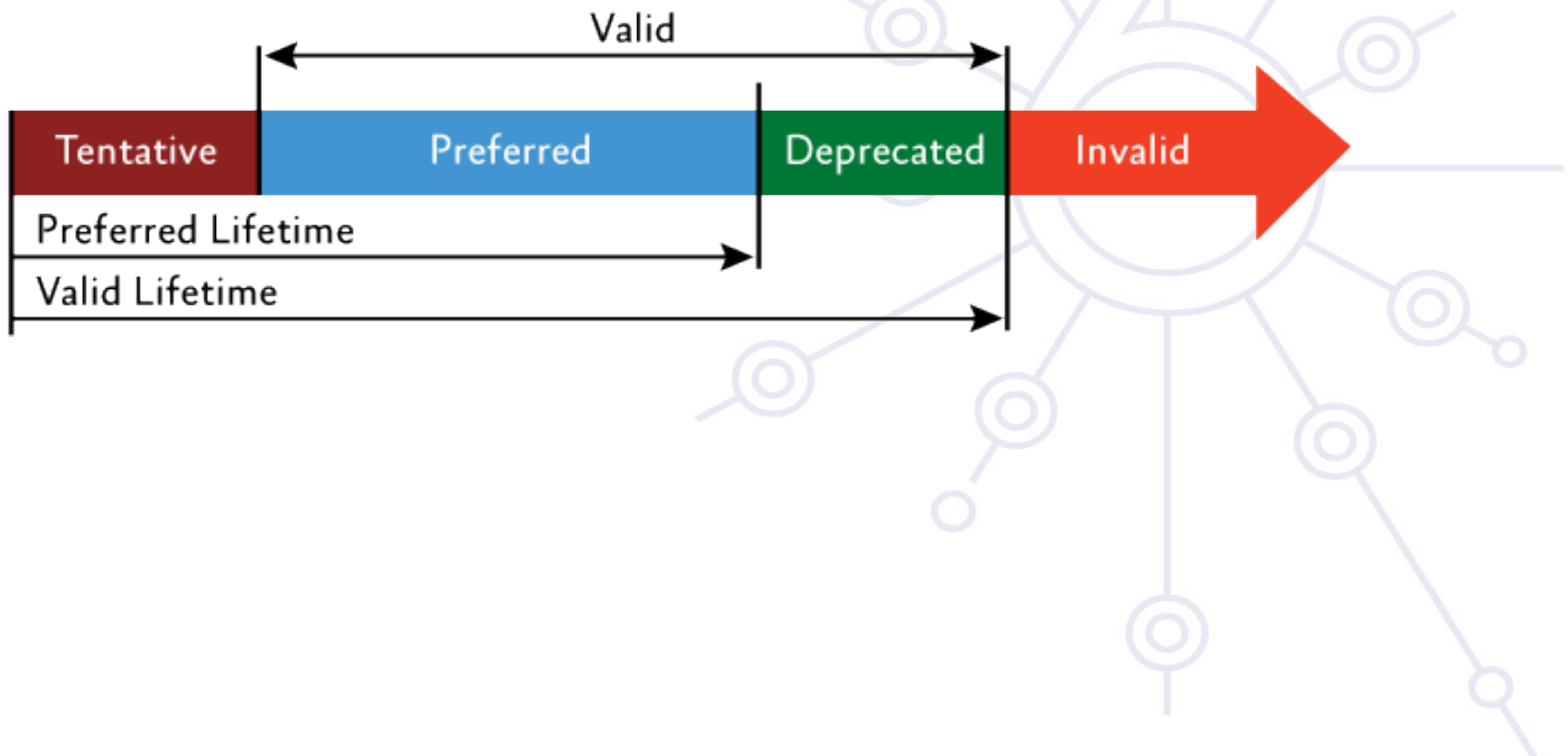
Campus services

Service provider deployment considerations



Discussion about address lifetimes

Each address has a lifecycle:

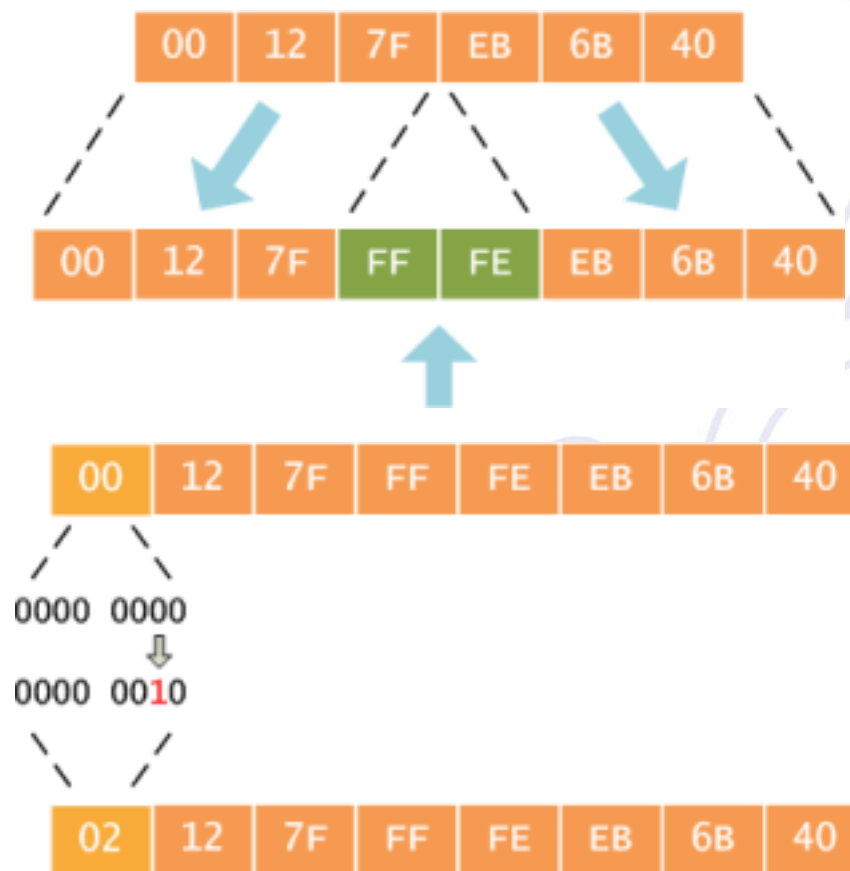


Campus Addressing - address assignment

- Which address assignment to use?
 - Autoconfiguration - IEEE provides uniqueness
 - DHCPv6 - central management provides uniqueness
 - Manual - 7th bit of IID should be 0
- Which one to use at host side – can be hinted at router – in RA messages
 - M – “Managed address configuration” flag. - use DHCPv6
 - O – “Other configuration” flag. - other configuration information is available via DHCPv6 (DNS et al) – stateless DHCPv6
 - Both clear - use SLAAC

New Things to Think About /4

Recap from EUI-64:



- The motivation for inverting the '**u**' bit when forming the interface identifier is to make it easy for system administrators to hand configure local scope identifiers. This is expected to be case for serial links, tunnel end-points and servers, etc. simply ::1, ::2, etc

Campus Addressing - address assignment

- Which address assignment to use?
 - Autoconfiguration - IEEE provides uniqueness
 - DHCPv6 - central management provides uniqueness
 - Manual - 7th bit of IID should be 0

Methods to manually assign addresses:

IID part	Description
0000::<smallnumber>	Easy to remember allocations
0080:vvww:yyzz:XXXX/112	Automatically assigned to vv.ww.yy.zz IPv4 address: /112 belongs to a IPv4 host - good for service virtualisation

Stateless address autoconfiguration [RFC4862]

- Additional option next to manual and DHCP assignment
- Just works ;-)
- Do not use autoconfigured addresses for stable services (e.g. mail, DNS, web) - servers can change overtime (network interface card change, complete server box change etc.) -> autoconfigured address changes
- DNS server address must be supplemented via DHCP(v6) or use RDNSS [RFC 5006] option:

- Cisco router configuration snippets:

```
ipv6 dhcp pool dhcp6dns
  dns-server 2001:db8:0::2
  domain-name example.hu
```

- and on the interface configuration:

```
ipv6 nd other-config-flag
ipv6 dhcp server dhcp6dns
```

Problems with SLAAC

Rogue RAs – as documented in [RFC 6104]

Possible solutions:

1. RA snooping - RA Guard - as defined [RFC 6105]
2. ACL on switches
3. Usage of SEND
4. Using RA router preference – use high
5. Layer 2 admission control – like 802.1X
6. Host based filtering - unwanted Ras
7. Deprecation tools:
 1. rafxid:
<http://www.kame.net/dev/cvsweb2.cgi/kame/kame/kame/rafixd/>
 2. ramond: <http://ramond.sourceforge.net/>
8. Using DHCPv6 with prefix and default gateway option

Privacy enhanced SLAAC [RFC4949]

prevents device/user tracking for 3rd parties
makes accountability harder

In a strict environment disable it

Windows clients: `netsh interface ipv6 set privacy=disabled`

Cryptographically Generated IPv6 Addresses (CGA)

Basic idea: Interface Id = hash (Public Key)

The public key is used to authenticate messages sent from the CGA address.

Proof of address ownership without security infrastructure

Not widely implemented and available

CGA: [RFC3972], HBA:[RFC5535]

DHCPv6 /1

IPv6 has stateless address autoconfiguration but DHCPv6 (RFC 3315) is available too

DHCPv6 can be used both for assigning addresses and providing other information like nameserver, ntpserver etc

If DHCPv6 is not used for address allocation, no state is required on server side and only part of the protocol is needed.

This is called *Stateless DHCPv6* (RFC 3736)

Some server and client implementations only do Stateless DHCPv6 while others do the full DHCP protocol

- Some vendors don't implement yet a DHCPv6 client (MacOS X, ...)

The two main approaches are

- Stateless address autoconfiguration with stateless DHCPv6 for other information
- Using DHCPv6 for both addresses and other information to obtain better control of address assignment

Statefull Autoconfiguration

DHCPv6 /2

DHCPv6 works in a client / server model

- **Server**
 - Responds to requests from clients
 - Optionally provides the client with:
 - IPv6 addresses
 - Other configuration parameters (DNS servers...)
 - Listens on the following multicast addresses:
 - All_DHCP_Relay_Agents_and_Servers (FF02::1:2)
 - All_DHCP_Servers (FF05::1:3)
 - Provides means for securing access control to network resources
 - Usually storing client's state, though 'stateless operation' is also possible (the usual method used for IPv4 today)

Statefull Autoconfiguration

DHCPv6 /3

- **Client**
 - Initiates requests on a link to obtain configuration parameters
 - Uses its link local address to connect the server
 - Sends requests to FF02::1:2 multicast address (All_DHCP_Relay_Agents_and_Servers)
- **Relay agent**
 - A node that acts as an intermediary to deliver DHCP messages between clients and servers
 - On the same link as the client
 - Listens on multicast address:
 - All_DHCP_Relay_Agents_and_Servers (FF02::1:2)

DHCPv6 considerations and implementations

One possible problem for DHCP is that DHCPv4 only provides IPv4 information (addresses for servers etc) while DHCPv6 only provides IPv6 information. Should a dual-stack host run both or only one (which one)?

Several vendors working on DHCP integrations - several implementations available at the moment

- DHCPv6 <http://dhcpv6.sourceforge.net/> - discontinued?
- dibbler <http://klub.com.pl/dhcpv6/>
- KAME-WIDE DHCPv6 <http://sourceforge.net/projects/wide-dhcpv6/>
- ISC DHCPv6 <https://www.isc.org/software/dhcp>
- Cisco routers have a built-in DHCPv6 server that can work as stateless or statefull server.
- **Beware:** DHCPv6 software is not installed as standard by most Linux and BSD distributions.

DHCP can also be used between routers for prefix delegation (RFC 3633). There are several implementations. E.g. Cisco routers can act as both client and server

DHCPv6 some more information

- BootP – client identification via MAC address
- DHCP – client identification via MAC address or client ID
- DHCPv6 –client identification via DUID (DHCP unique ID)
 - DUID is opaque in the communication

DUID versions:

1. DUID-LLT – Link-Layer Address + time

Type:1	Hardware Type: (Ethernet=6)
Time (time()) since 1 Jan 2000)	
Link-Layer Address (variable)	

2. DUID-EN - Vendor-Assigned Based on Enterprise Number

DHCPv6 some more information /2

DUID versions:

3. DUID-LL – Link-Layer Address

Type:3	Hardware Type: (Ethernet=6)
Link-Layer Address (variable)	

Some important terminologies:

IA – “identity-association” is a construct server and a client can identify and manage a set of related IPv6 addresses (set of addresses assigned to a client) – similar timing as SLAAC

IAID, IA_TA, IA_NA

DHCPv6 software capabilities

Dibbler

- Windows and Linux
- Flexible – number of options, RFCs, and drafts (e.g. DS-lite) supported
- Sometime complex to configure

WIDE-DHCPv6

- Linux, *BSD, UNIX
- No IA_TA support, only DUID_LLTD support in the client
- Can run on as server and client in the same machine

Windows (Vista, Win7)

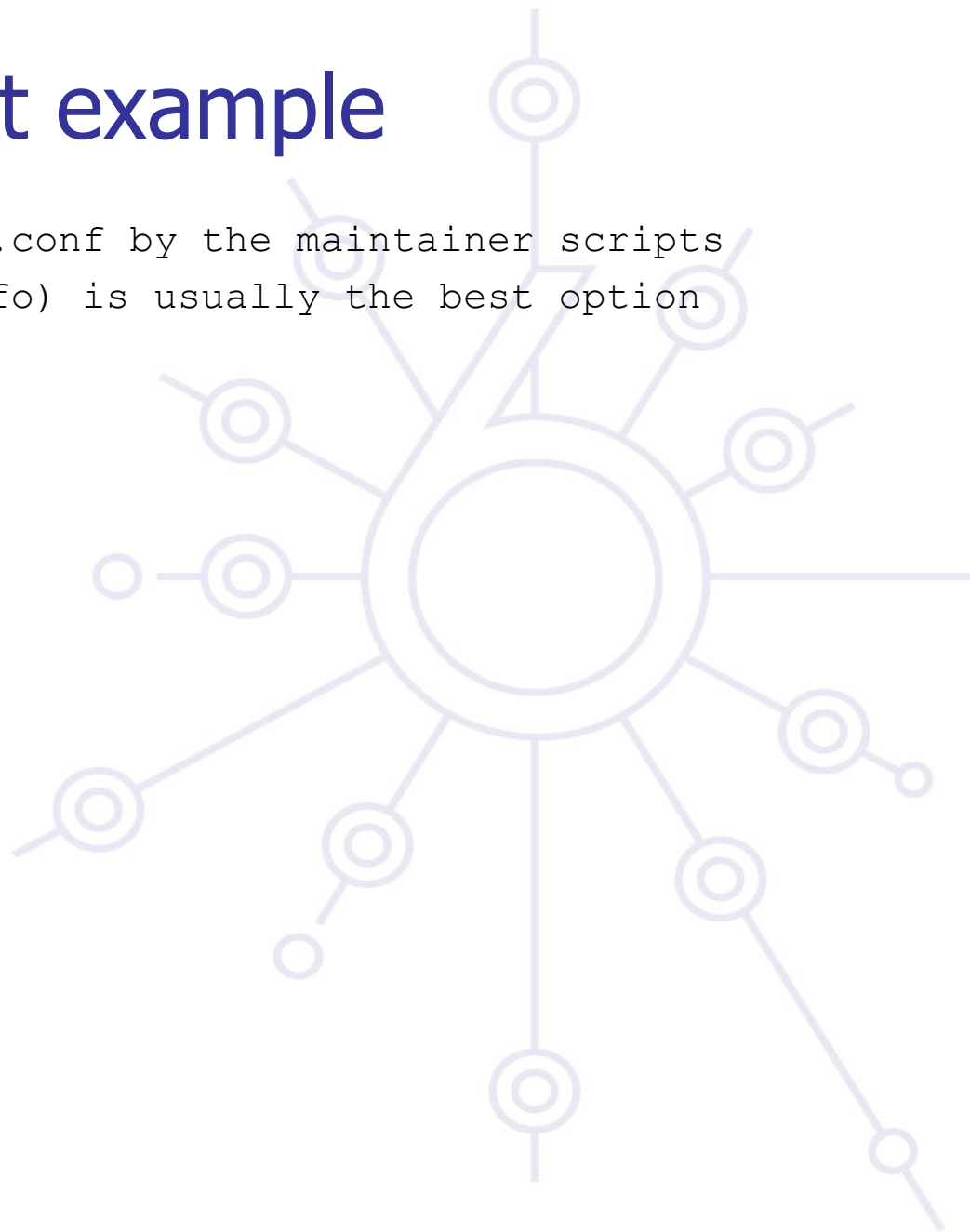
- No IA_TA support

Dibbler client example – address assignment

```
# installed at /etc/dibbler/client.conf by the maintainer scripts
# 8 (Debug) is most verbose. 7 (Info) is usually the best option
log-level 7
# uncomment only ONE of the lines below: duid-llt is the default
#duid-type duid-llt
#duid-type duid-en 1234 0x56789abcde
#duid-type duid-ll
iface eth0 {
    # ask for address
    ia
    # ask for options
    option dns-server
}
```

Dibbler stateless client example

```
# installed at /etc/dibbler/client.conf by the maintainer scripts
# 8 (Debug) is most verbose. 7 (Info) is usually the best option
log-level 7
stateless
iface eth0 {
    # ask for options
    option dns-server
    option domain
    option ntp-server
}
```



Dibbler server example - pool

```
# server.conf
iface eth0
{
    #renew lease every 10 minutes
    T1 600
    #in case of failure ask other servers in 15 minutes
    T2 900
    # preferred lifetime and valid lifetime option
    preferred-lifetime 3600
    valid-lifetime 86400
    class
    {
        pool 2001:db8::100/80
    }
    option dns-server 2001:db8::1234
    # lifetime 2h
    option lifetime 7200
}
```

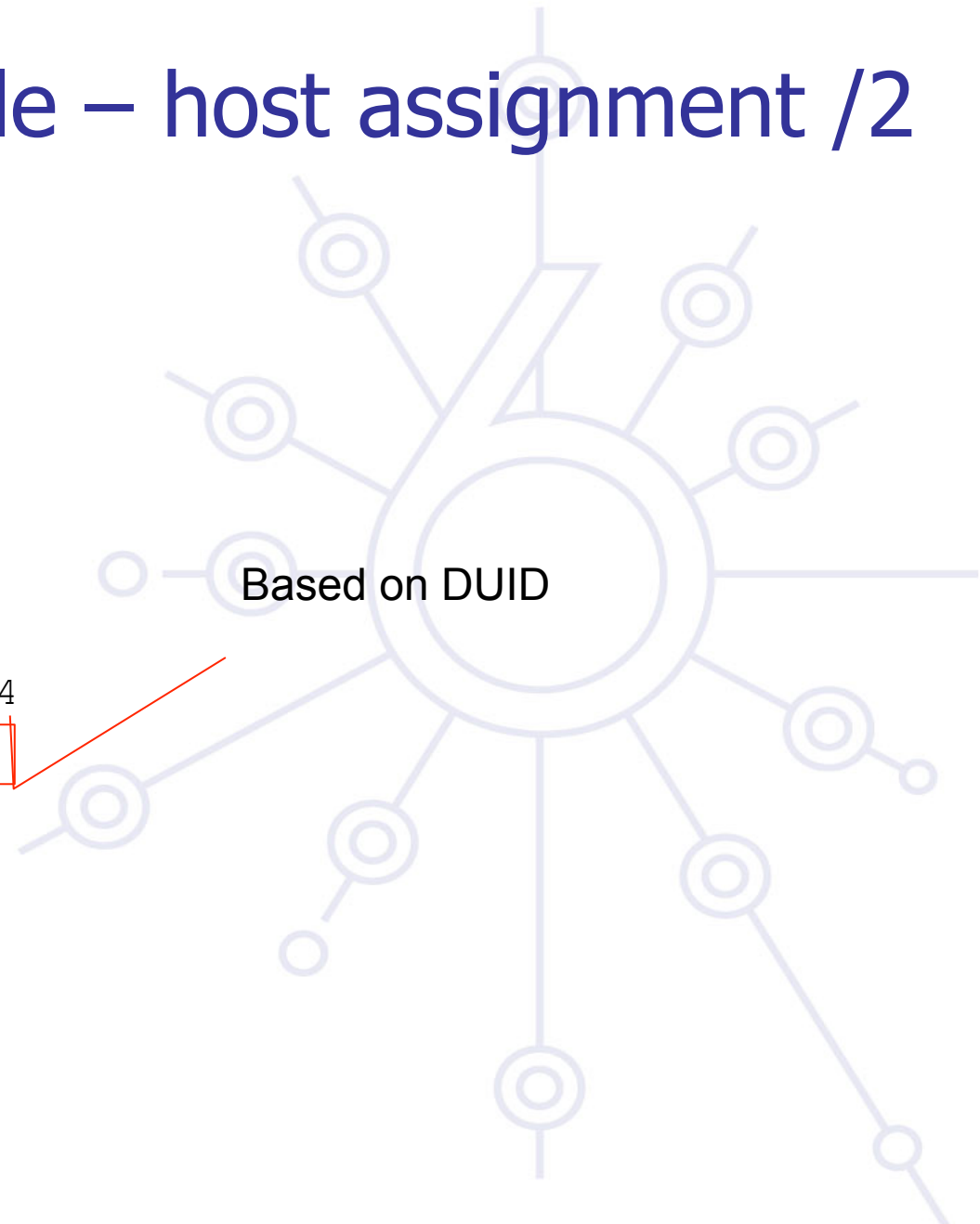
Dibbler server example – host assignment /1

```
# server.conf
iface eth0
{
    # preferred lifetime and valid lifetime option
    preferred-lifetime 3600
    valid-lifetime 86400
    class {
        class-max-lease 1
        #
        host: example1
        accept-only FE80::207:E9FF:FE6C:44C9
        pool 2001:db8::2
    }
    option dns-server 2001:db8::1234
    # lifetime 2h
    option lifetime 7200
}
```

Based on link-
local address

Dibbler server example – host assignment /2

```
# server.conf
iface eth0
{
    preferred-lifetime 3600
    valid-lifetime 86400
    class {
        pool 2001:db8::1/64
    }
    option dns-server 2001:db8::1234
    client duid 0x000102030406
    {
        address 2001:db8::123
    }
}
```



Based on DUID

Dibbler server example – prefix length assignment

```
# server.conf
# Warning: This feature is non-standard and is not described by any
# standards or drafts.
log-level 8
# allow experimental stuff (e.g. addr-params)
experimental
iface eth0 {
    preferred-lifetime 120
    valid-lifetime 180
class {
    addr-params 80 // addresses will be assigned with /80 prefix
    pool 2001:db8:ff01:ff03::/80
}
# provide DNS server location to the clients
option dns-server 2001:db8:ffff:ffff::53
}
```

**Prefix length – client
side must be
configured also!**

WIDE client example – address assignment

```
# installed at /etc/wide-dhcpv6/dhcp6c.conf
#
interface eth0 {
    request domain-name-servers;
    request domain-name;
    send rapid-commit;
    send ia-na 1;
    script "/etc/wide-dhcpv6/dhcp6c-script";
};

id-assoc na 1 { # empty but do not remove
};
```


WIDE stateless client example

```
# installed at /etc/wide-dhcpv6/dhcp6c.conf
#
interface eth0 {
    request domain-name-servers;
    request domain-name;
    send rapid-commit;
};
```



WIDE client with DUID LL

- **Why?**

- Admin don't know what the value is of the automatically created DUID -> create a new DUID with know values
- Timestamp can be good for uniqueness, but in the campus admins wants control

- **Generate new duid**

- wide_mkduid.pl Perl script available from Jeffrey F. Blank of Michigan Technological University:

http://www.ipv6.mtu.edu/wide_mkduid.pl

- Option to create LLT and LL DUID:

```
wide_mkduid.pl [ -t <time> ] { -m <macaddr> | <ifname> }
```

if specified, <macaddr> must be 6 colon-separated hex values

if specified, <time> must be an integer or 'now'

- Then put in the client config file locations (`/var/lib/dhcpv6/dhcp6c_duid` or `/var/db/dhcp6c_duid`)

WIDE server example – host assignment

```
# DNS server search list, v6 addresses only
option domain-name-servers 2001:db8::1e;

# DNS suffix search list
option domain-name "example.ac.hu";

interface bge0 {
    # interface parameters go here
};

host some-pc {
    # the contents of Dibbler's client-duid file
    # (or any other client DUID)
    duid 00:01:00:06:46:e2:f8:c2:00:08:74:da:ab:64;

    # host's address with preferred and valid lifetimes in seconds
    address 2001:db8:0:2::1:c8 1800 7200;
};
```

WIDE server example - pool

```
# DNS server search list, v6 addresses only
option domain-name-servers 2001:db8::1e;

# DNS suffix search list
option domain-name "example.ac.hu";

interface bge0 {
    # pool with preferred and valid lifetimes in seconds
    address-pool mysubnet 1800 2700
};

pool mysubnet {
    range 2001:db8:0:2::100 to 2001:db8:0:2::1ff;
};
```

ISC DHCP stateless server example

```
authoritative;
```

```
#address lease times  
default-lease-time 3600;  
max-lease-time 86400;
```

```
subnet6 2001:db8:0:2::/64 {  
    option dhcp6.name-servers  
  
}
```

```
2001:db8::da44,2001:db8:1::2;
```

Integration of stateless WIDE DHCPv6 into Mac OS X

Install WIDE DHCPv6

Hack DHCPv6 DNS answers into resolving DNS entries
with `scutil` with running **WIDE DHCPv6** `dhcp6c-script.sh`

How to at:

<http://wouter.horre.be/doc/stateless-dhcpv6-on-mac-os-x>

Problems

1. IPv6 addresses – put in several databases

There is a need to put hosts in the DNS – manually it is troublesome due to length of the addresses

More controlled environment would like to use DHCP also

2. IPv6 Address and MAC address binding

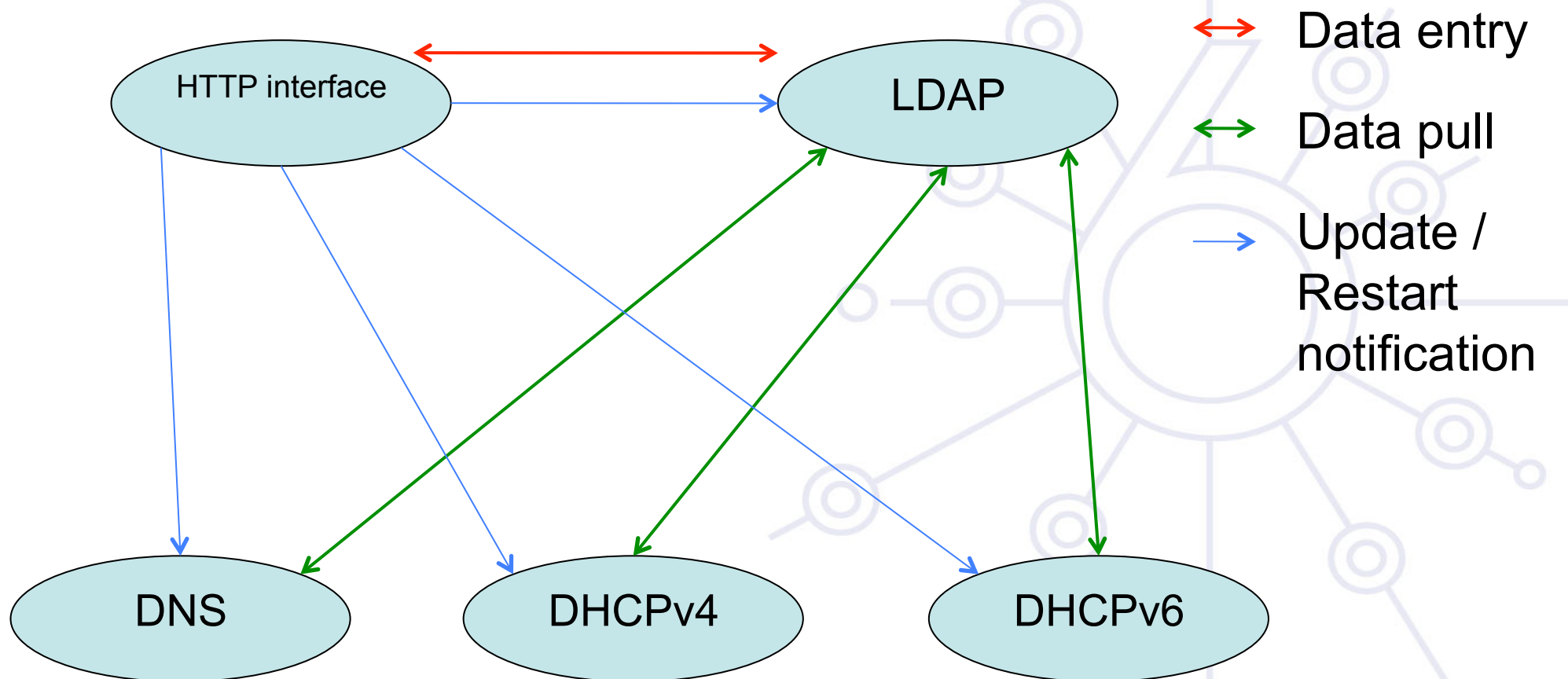
To monitor the campus environment you should have real-time information about that – for example for later incident coordination

Particularly important if someone is using privacy enhanced addresses

Problem 1 – solution 1: L2D2 /1

- **Store the data in database**
 - LDAP
- **The user interface should be platform neutral, easy to access**
 - HTTP és CGI
- **Flexible**
 - Distributed: HTTP, LDAP, DNS, DHCP (IPv4), DHCP (IPv6)
- **Robust**
 - DNS and DHCP servers are using configuration files
- **Secure**
 - Use mostly non-harmful operations
- **L2D2 available:**
 - <http://www.kfki.hu/cnc/projekt/l2d2>

Problem 1 – solution 1: L2D2 /2





Problem

I2d2 admin

File Edit View

Home

Admin: rootadmin

>> Search and

>> Create a r

>> Create a r

>> Create a r

>> Create a r

>> Server up

>> Change a

>> Change to

>> Logout...

Done

I2d2 adminstartion page - Mozilla

File Edit View Go Bookmarks Tools Window Help

http://localhost/cgi-bin/I2d2/main.cgi

Search

Home Bookmarks SUSE mozilla.org

example.org

Admin: rootadmin IPv4 subnet: 192.168.0.0/16 IPv6 subnet: 2001:db8::/32

Name and IP addresses		DHCP parameters	
Hostname:	<input type="text" value="example1"/>	Hardware address:	<input type="text"/>
IPv4 addresses:	<input type="text" value="192.168.0.1"/>	DHCPv4 parameters	
IPv6 addresses:	<input type="text" value="2001:db8::1"/>	IPv4 default route:	<input type="text"/>
		IPv4 default lease time:	<input type="text"/>
		IPv4 maximum lease time:	<input type="text"/>
		IPv4 name server:	<input type="text"/>
		Other IPv4 DHCP options:	<input type="text"/>
		DHCPv6 parameters	
		IPv6 preferred lifetime:	<input type="text"/>
		IPv6 valid lifetime:	<input type="text"/>
		IPv6 T1 timer:	<input type="text"/>
		IPv6 T2 timer:	<input type="text"/>

DNS data	
Host TTL:	<input type="text"/>
DNS CNAME:	<input type="text"/>
Mail handler (MX):	<input type="text"/>
Host information:	<input type="text"/>
DNS SRV:	<input type="text"/>
Text:	<input type="text"/>
CERT:	<input type="text"/>

>> Don't check IP addresses! Delete Back<<

Problem 1 – solution 2: use nsupdate

- **Can be scripted**
- **Some DHCP server has support for it:**
 - Dnsmasq
 - Dnsmasq
 - ISC DHCP



Problem 2 – solutions

Have your IPv6 neighbor cache logged!

1. Collect IPv6 neighbor cache from your router

Beta version of netdisco can discover routers ipv6 neighbor cache (<http://www.netdisco.org>)

Beware you need development version of NET::SNMP::INFO::IPv6 perl module

2. Monitor your network segment

Sniff your segment about ND and RA traffic: ndpmon developed at LORIA (<http://ndpmon.sourceforge.net/>)

Reports: wrong couple MAC/IP, wrong router MAC, wrong router IP, wrong prefix, wrong router redirect, router flag in Neighbor Advertisement, DAD DOS, flip flop, reused old ethernet address

Outline

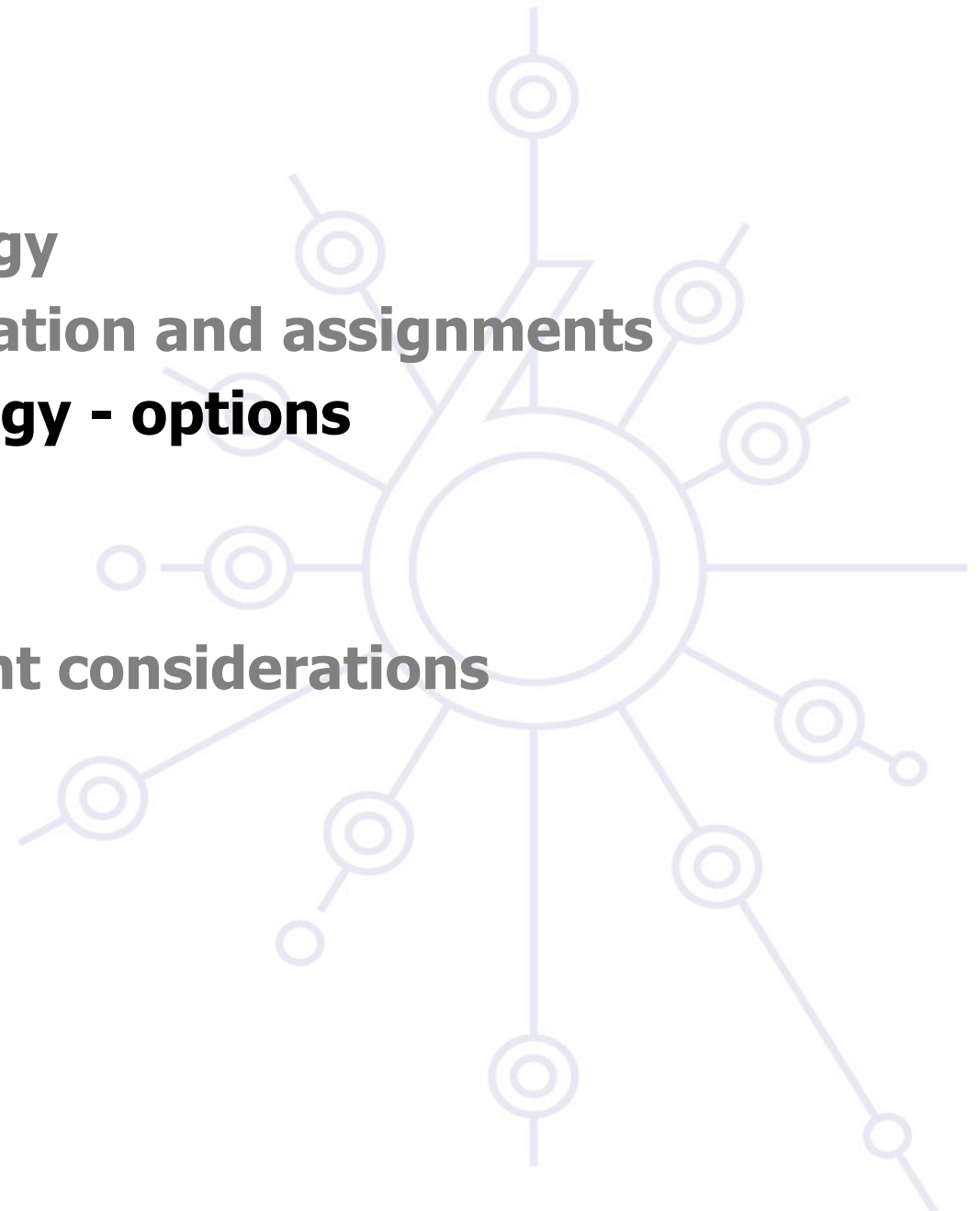
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

Campus services

Service provider deployment considerations



IPv6 deployment options

The simplest

- deploy dual stack network environment

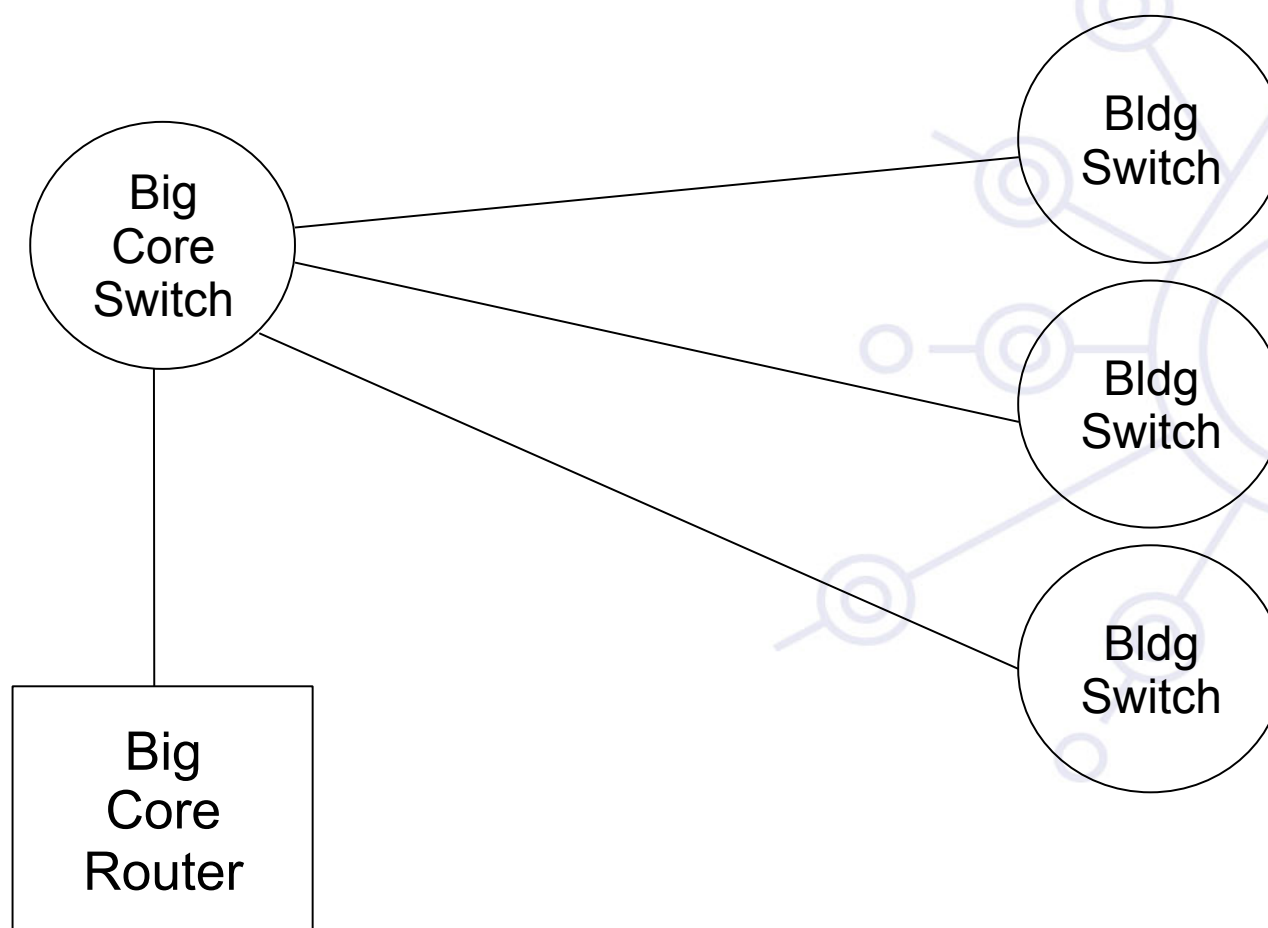
If the hosts/services are not dual stack enabled

- It does not break anything
- this tends to be a false assumption (Windows Vista, Mac OS X shipped with IPv6 enabled)

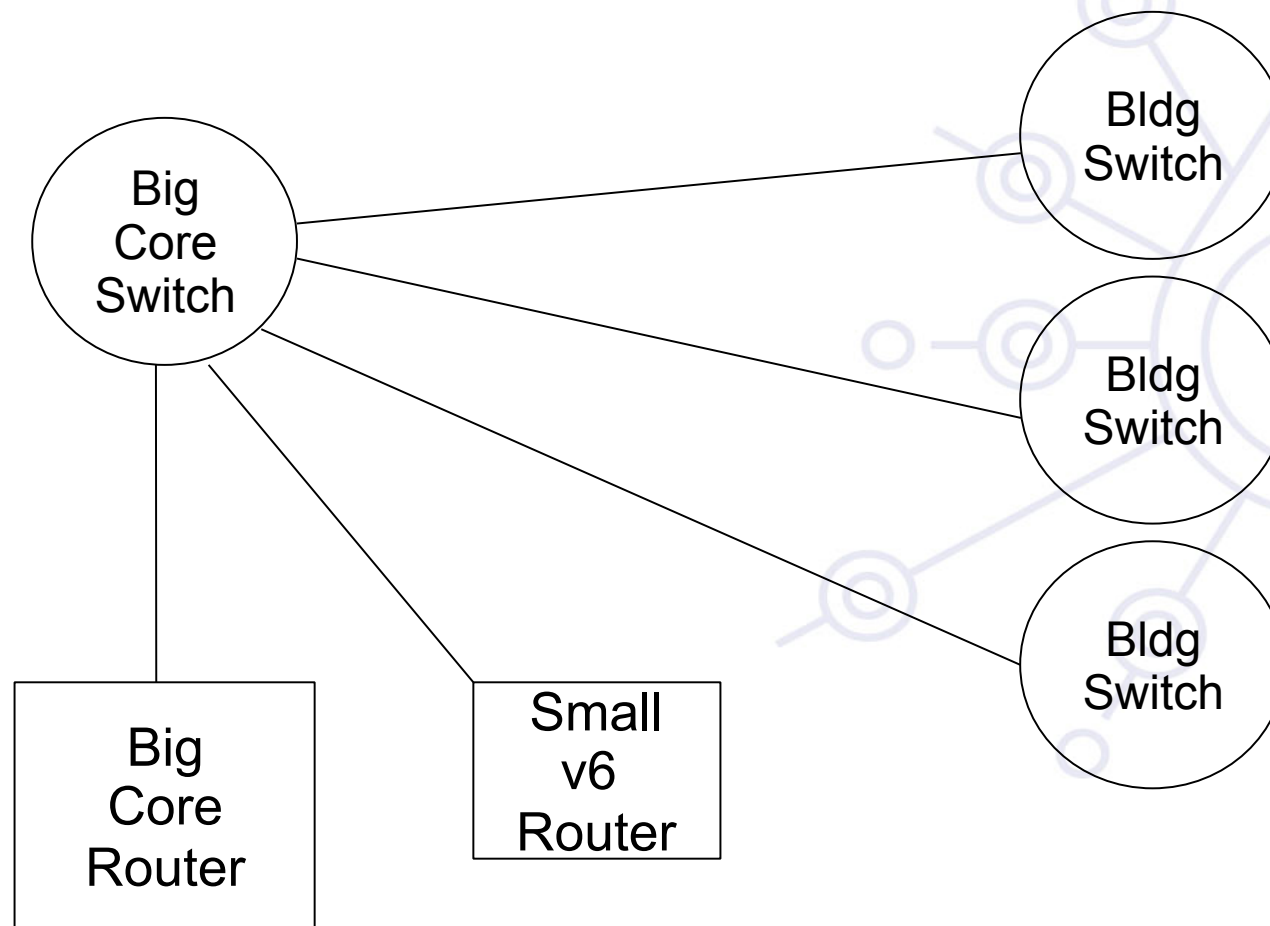
If the L3 devices cannot cope with IPv6 or administrators are not in favor of upgrading the router

- Add additional IPv6 capable L3 device(s)
- Investment money is usually a problem, but you can do some engineering with simple (low cost) PCs.

Layer-2 Campus - 1 Switch

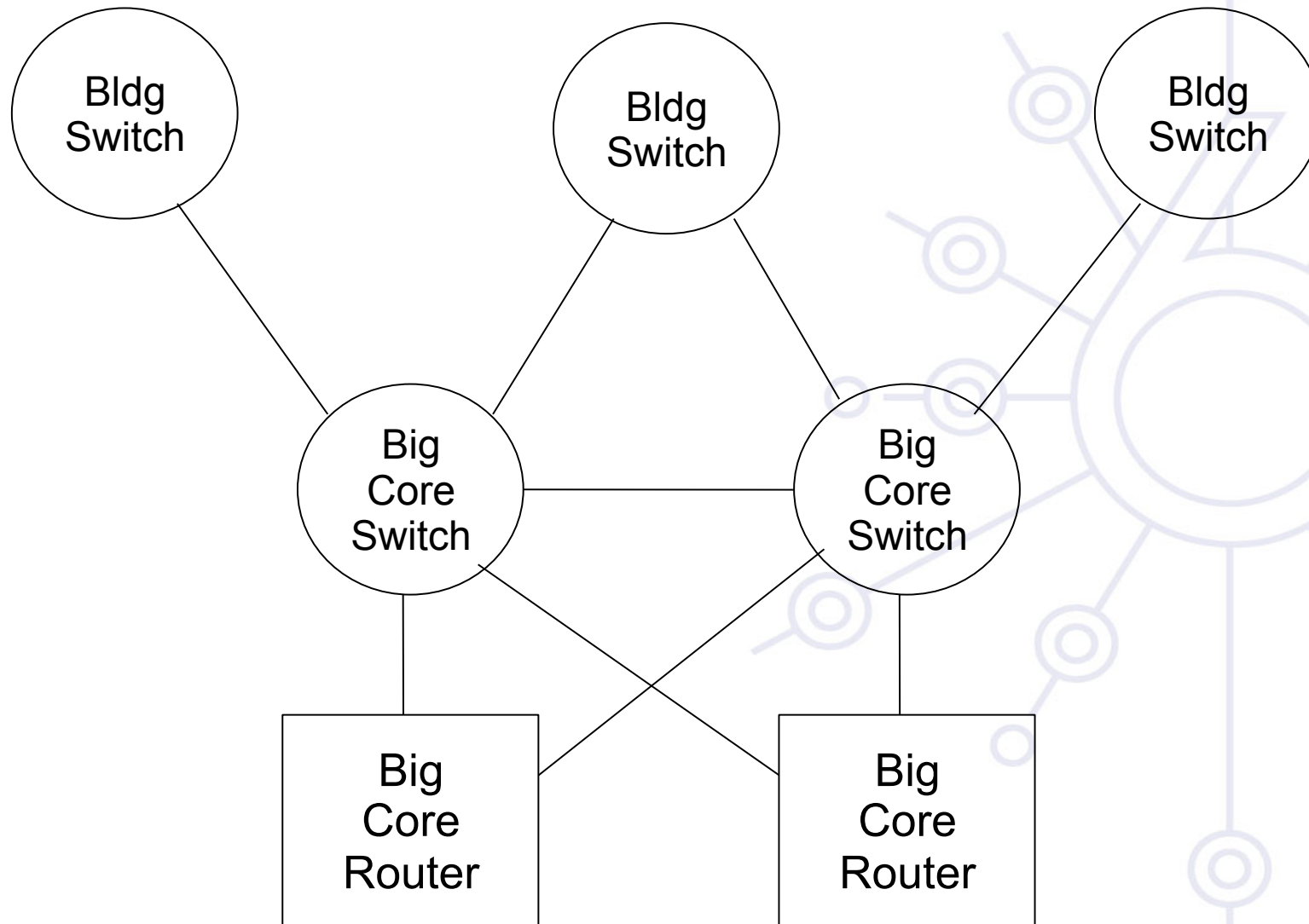


Layer-2 Campus - 1 Switch

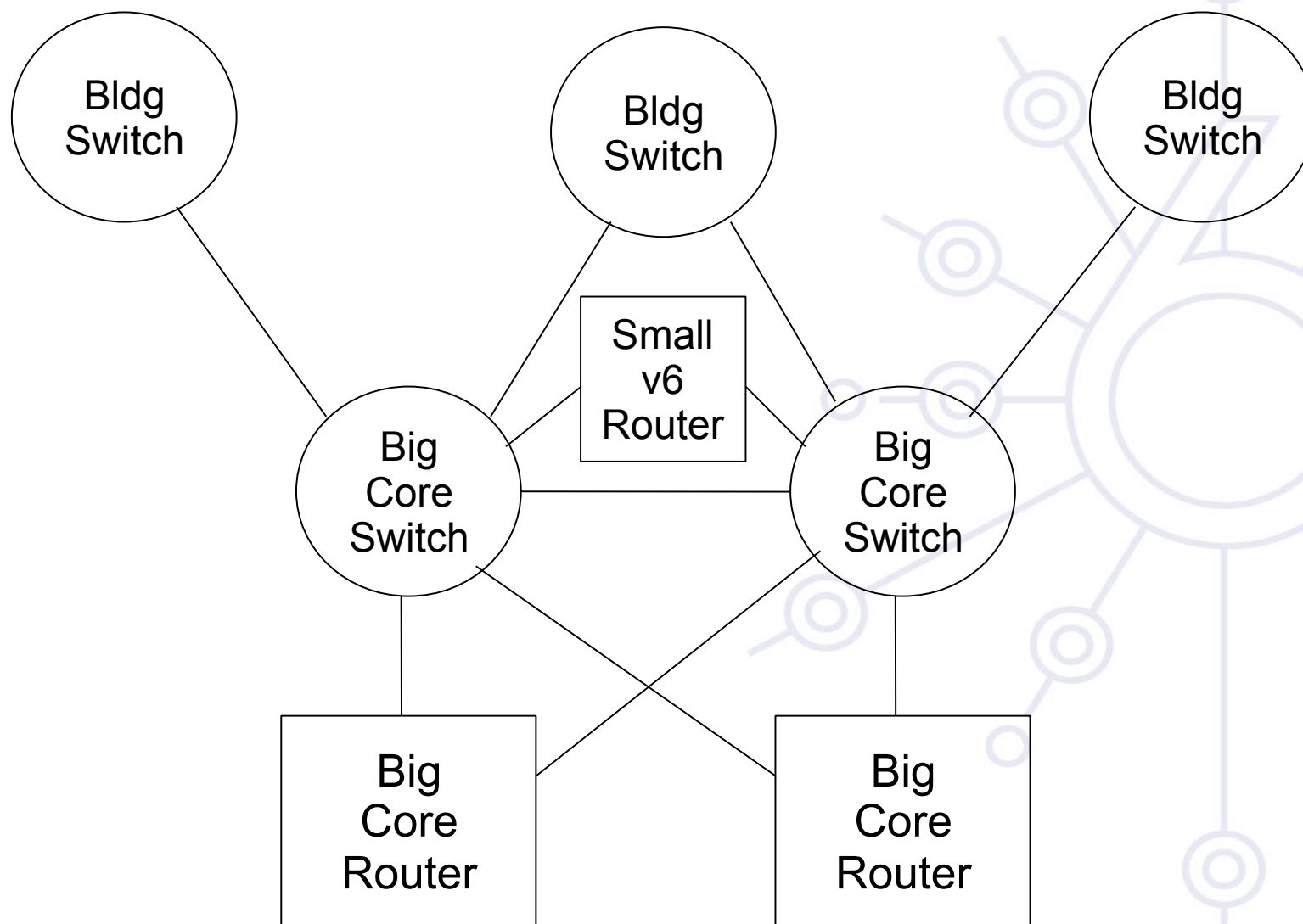


Layer-2 Campus

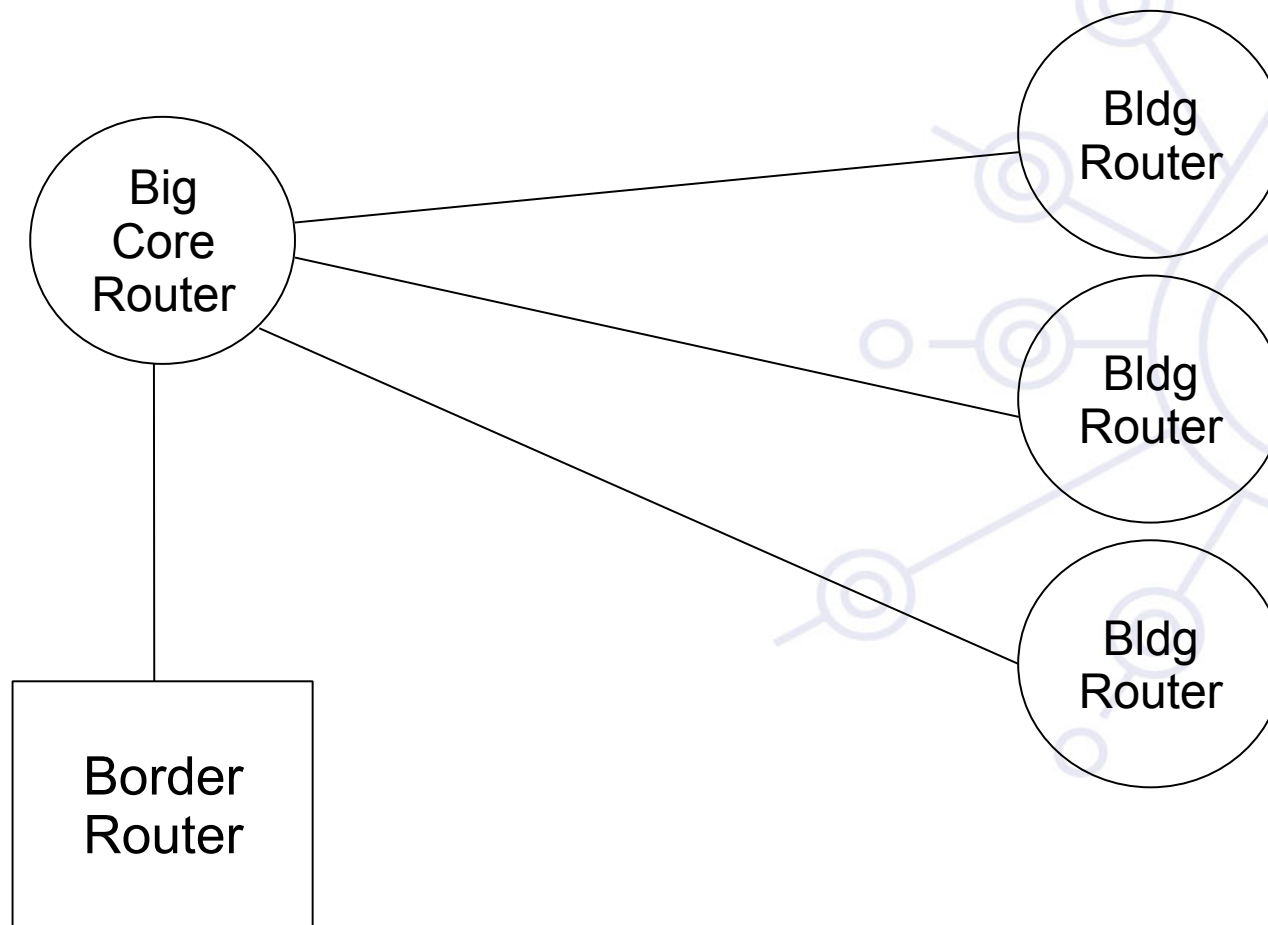
- Redundant Switches



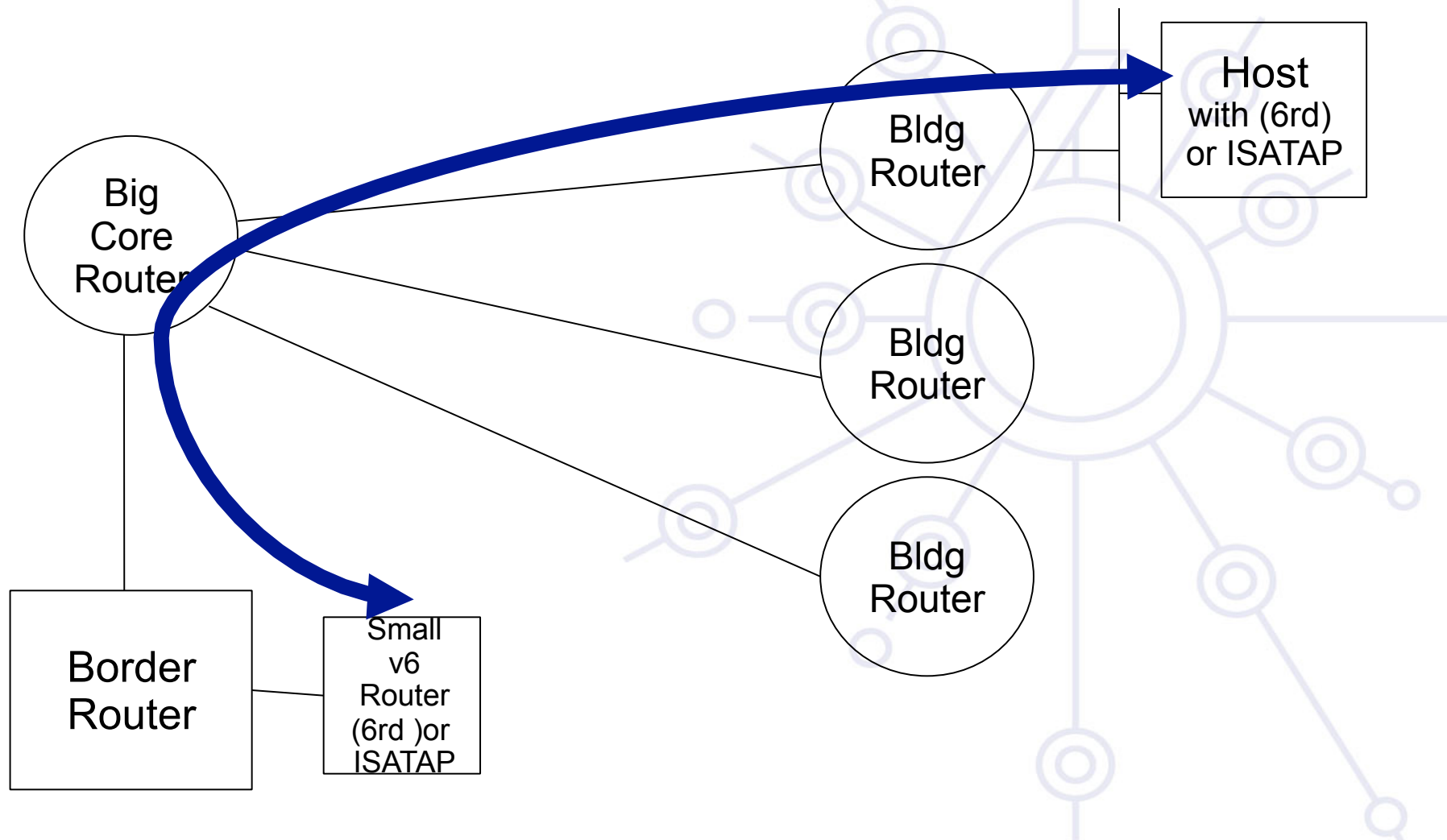
Layer-2 Campus Redundant Switches



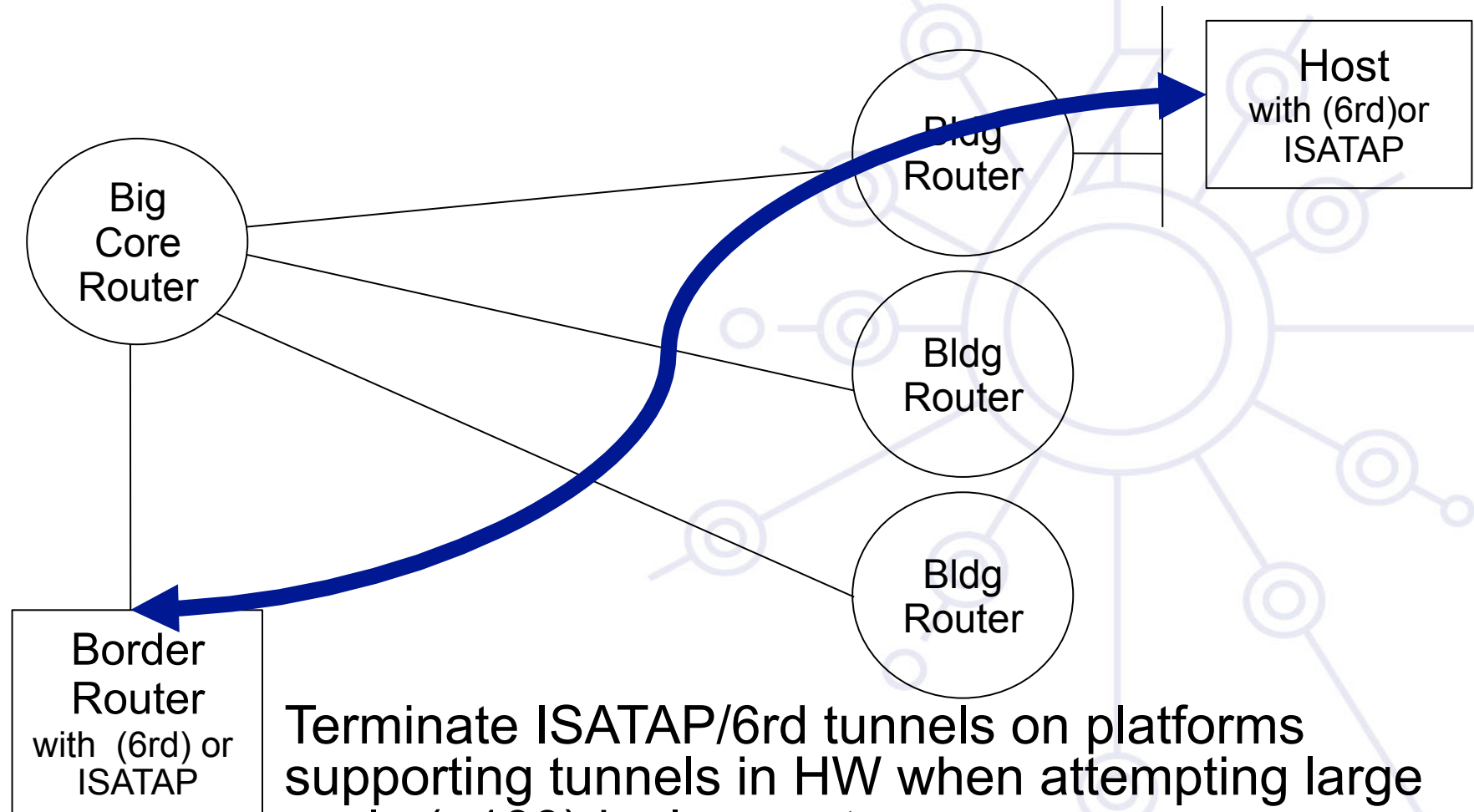
Layer-3 Campus



Layer-3 Campus – solution 1

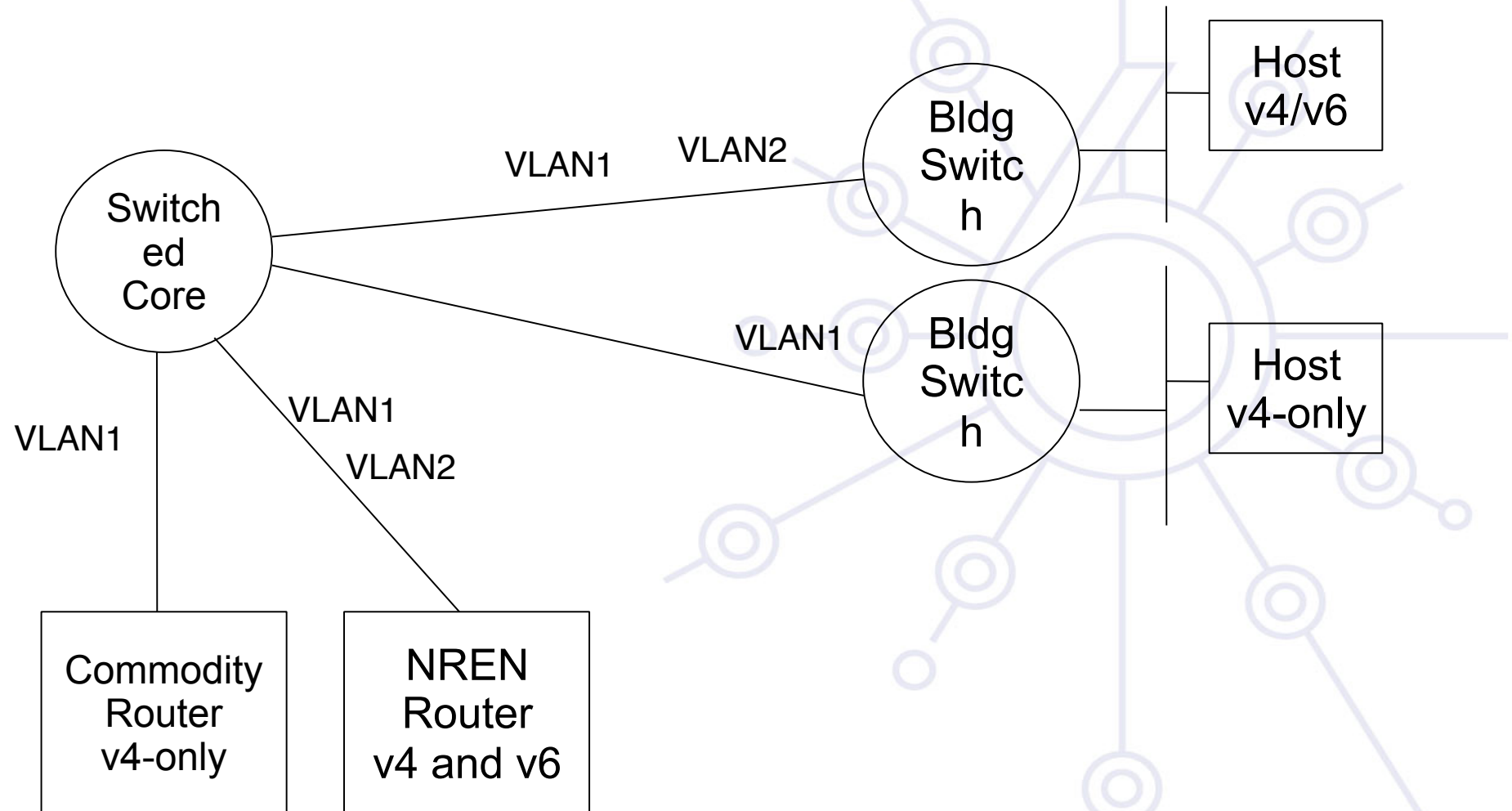


Layer-3 Campus - solution 2



Terminate ISATAP/6rd tunnels on platforms supporting tunnels in HW when attempting large scale (>100) deployments

Edge Router Options



Routing Protocols

iBGP and IGP (IS-IS/OSPFv3)

- IPv6 iBGP sessions in parallel with IPv4
- You need a 32 bit router-id for IPv6 BGP peering configuration

Static Routing

- all the obvious scaling problems, but works OK to get started, especially using a trunked v6 VLAN.

OSPFv3 might be good

- It will run in a ships-in-the-night mode relative to OSPFv2 for IPV4 - neither will know about the other.

Use the same (type) of protocol you used in IPv4.

See more in routing module

Outline

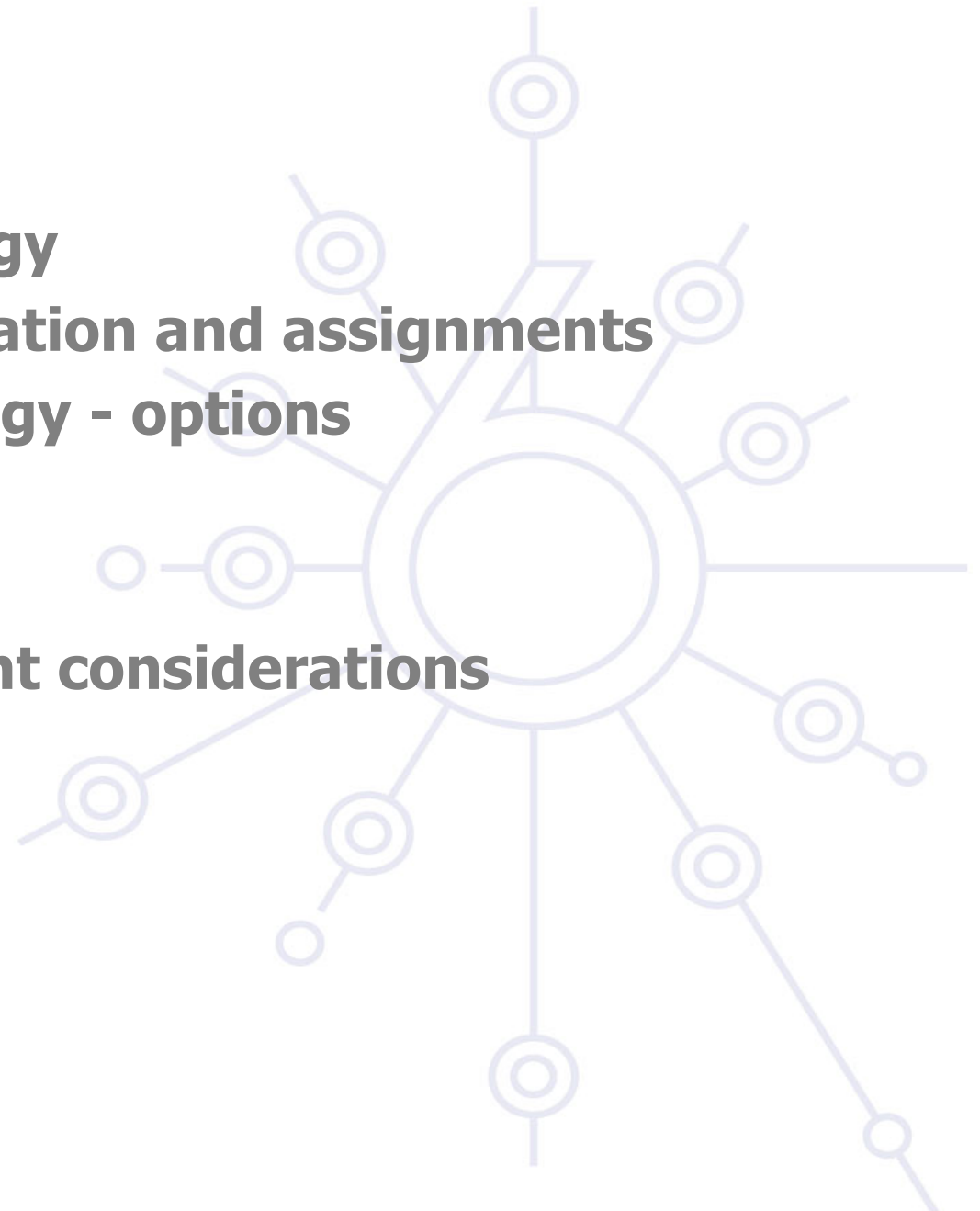
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

Campus services

Service provider deployment considerations



Campus services –Road Map

- **Name service - see DNS module**
- **Security policy - see security module**
- **Routing - see routing module**
- **(Mail) not considered here - see application module**
- **Proxying**
- **Remote access**
- **Monitoring the network and the services - see monitoring module**

=> For most of these services, refer to the ad hoc modules on <http://www.6deploy.org>

How to enable IPv6 services where you might expect some problems?

You should consider this method only for more complex services:

- Add v6 testing service for different name first:
 - service.v6.fqdn or service6.fqdn with AAAA + reverse PTR entry.
 - Test it
- Add v6 service under the same name:
 - service.fqdn with A +AAAA and two PTR.

How to enable IPv6 services if you don't have an IPv6 capable server?

Use proxy (more exactly reverse-proxy) server

- Apache2.x proxy is a very good one

Use netcat

- Kind of hack ☺

Other proxies

Proxy solutions

Proxy

- Squid (<http://devel.squid-cache.org/projects.html>)

Web Cache

- NetCache C1300, C2300, C3300. BlueCoat SG
- WCCP does not have IPv6 support in CISCO yet

Apache2 reverse proxy

Configuration is very easy:

```
ProxyRequests Off  
ProxyPass / http://ipv4address  
ProxyPassReverse / http://ipv4address  
ProxyPreserveHost On
```



Reverse proxy pros & cons

Advantage:

- Fast implementation, instantly provide web service over IPv6
- No modifications required in a production web server environment
- Allow for timely upgrading of systems
- Scalable mechanism: a central proxy can support many web sites

Disadvantage:

- Significant administrative overhead for large scale deployment
- May break advanced authentication and access control schemes
- Breaks statistics: all IPv6 requests seem to be coming from the same address
 - may be fixed with filtering and concatenation of logs or specialised module on proxy
- Not a long term solution overall, native IPv6 support is readily available in related applications and should be preferred whenever possible

Management and monitoring

- Device configuration and monitoring -SNMP
- Statistical monitoring e.g. Cricket/MRTG
- Service monitoring - Nagios
- Intrusion detection (IDS)
- More information
 - Module #060 : IPv6 Networks management
 - <http://www.6deploy.org>

Introduction

IPv6 networks deployed:

- Most are dual stack
 - LANs (campuses, companies, ...)
 - MANs
 - WANs - ISPs (Géant, NRENs, IIX, NTT/Verio, Abilene, ...)
 - IXs

Testbed, pilot networks, production networks

- Management tools/procedures are needed

What applications are available for managing these networks ?

- Equipment, configurations, ...
- **IP services** (servers : DNS, FTP, HTTP, ...)

Introduction

Different types of networks

- Dual stack IPv6 & IPv4 networks
- IPv6 only networks (few of them)

Important to keep in mind

- Dual stack is not forever
- One IP stack should be removed... one day
- No reasons for network admins to face twice the amount of work

Dual Stack IP networks

Part of the monitoring via IPv4

- Connectivity to the equipment
- Tools to manage it (inventory, configurations, «counters», routing info, ...)

Remaining Part needs IPv6

- MIBs IPv6 support
- NetFlow (v9)



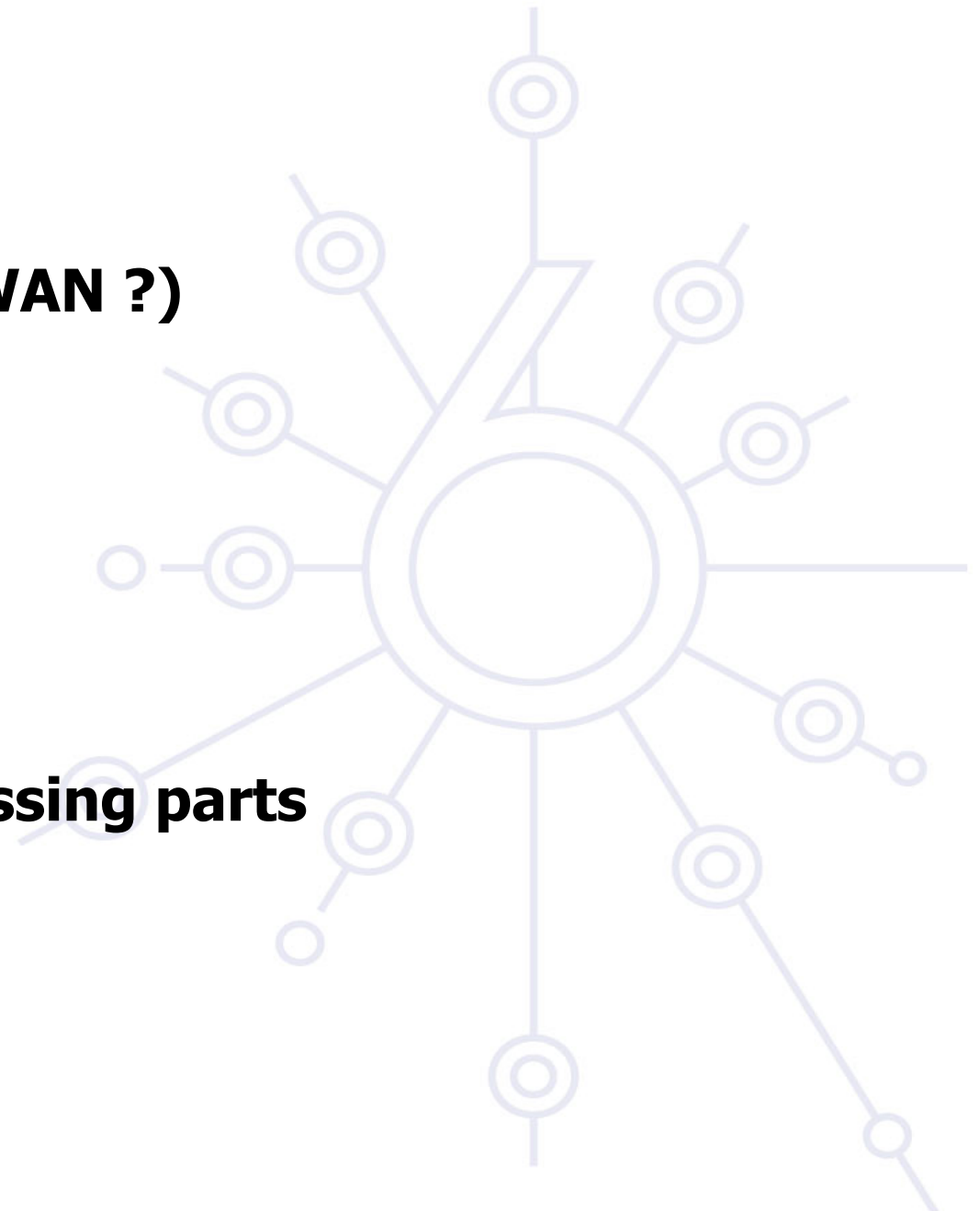
IPv6 only networks

Topology discovery (LAN, WAN ?)

IPv6 SNMP agent

SNMP over IPv6 transport

=> Need to identify the missing parts



Accessing -SSH/TELNET/TFTP...

All routers support IPv6 connections (SSH, TELNET)

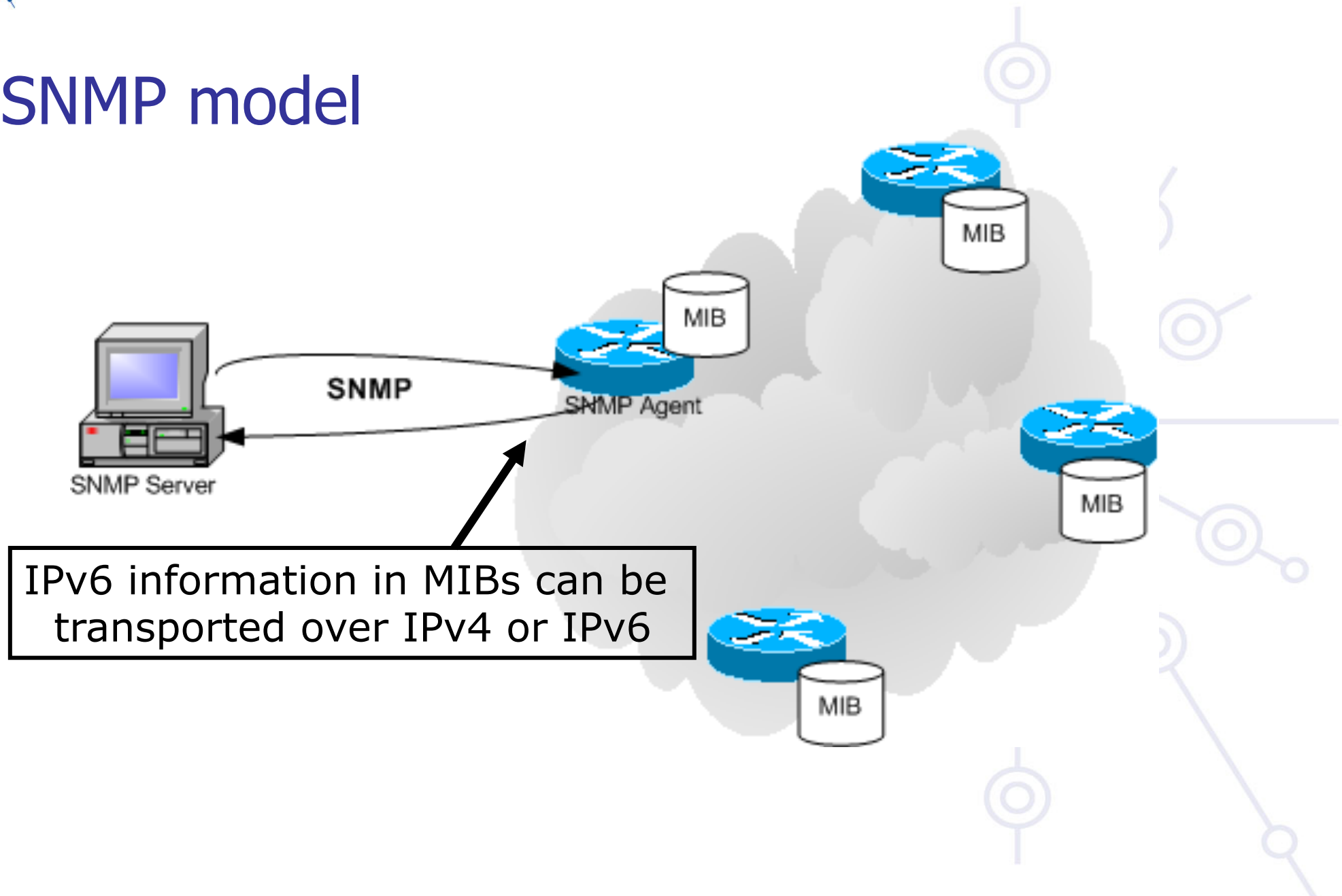
- Periodic scripts can retrieve information from the routers over IPv6

TFTP/IPv6 is also supported on all equipment

- Images can be downloaded over IPv6

FTP/IPv6 should be also supported on all equipment

SNMP model



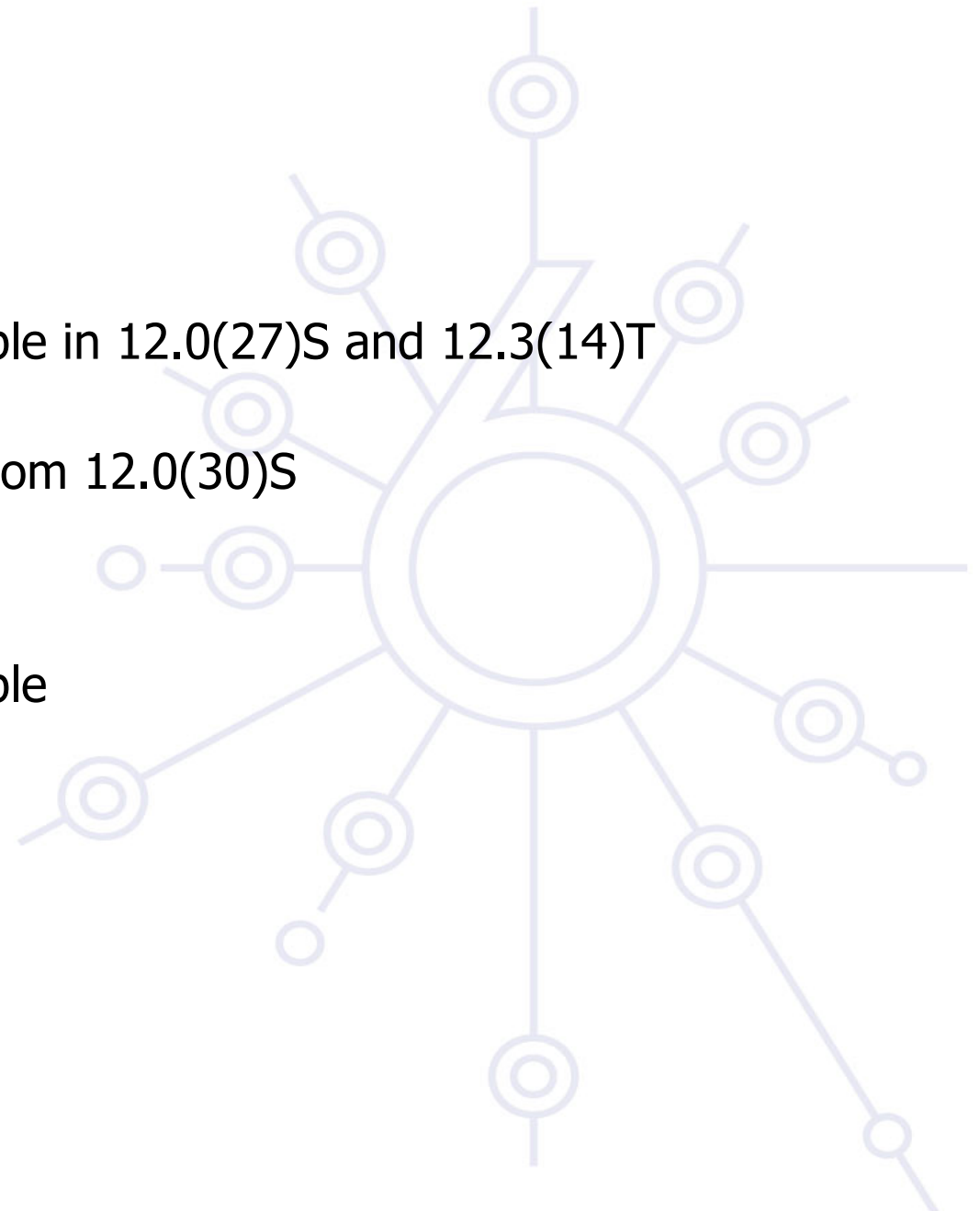
SNMP over IPv6

Cisco:

- SNMP over IPv6 is available in 12.0(27)S and 12.3(14)T
- IOS 12.4 & 12.4T too
- More features available from 12.0(30)S

Juniper, Hitachi, 6wind:

- SNMP over IPv6 is available



IPv6 MIBs status / 1

MIBs are essential for the network management

SNMP-based applications are widely used but others exist too (NetFlow, XML, ...)

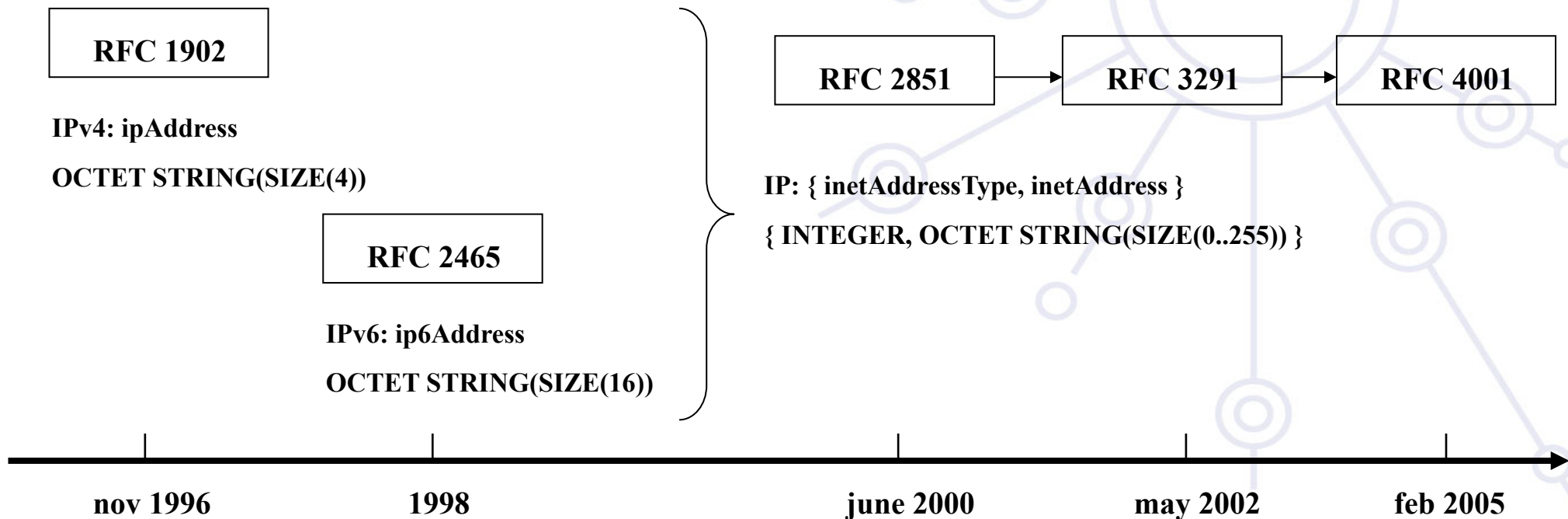
SNMP rely upon MIBs

=> Need to have MIBs to collect IPv6 information as well as get MIBs reachable from an IPv6 address family

IPv6 MIBs /2

Standardization status at IETF:

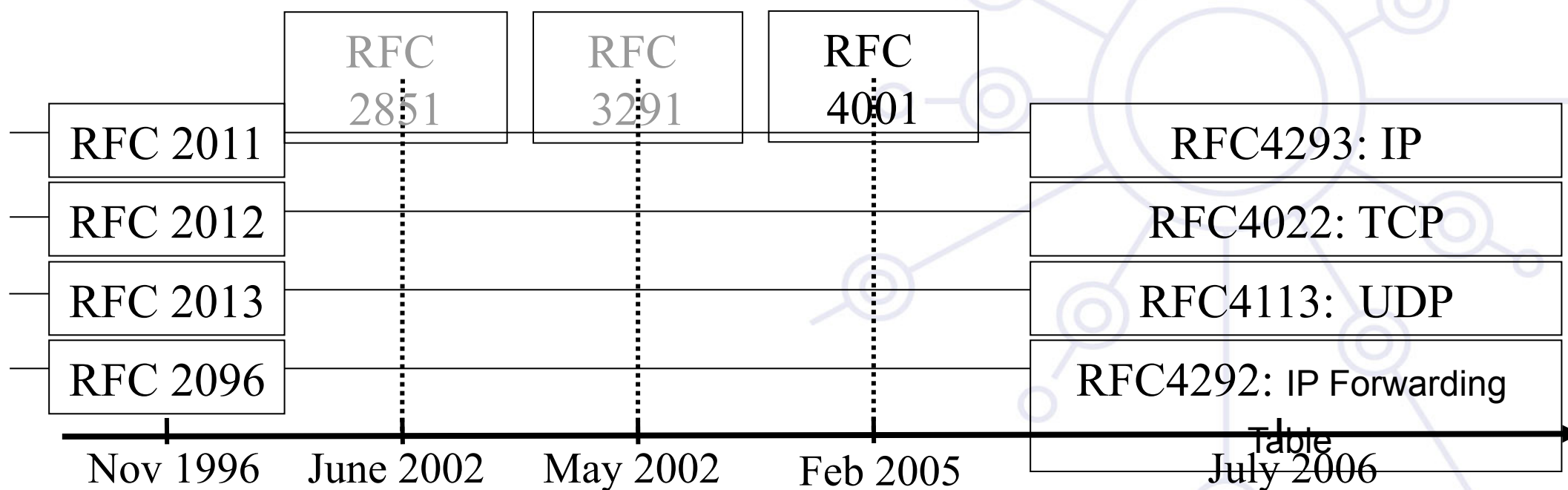
- At the beginning:
 - IPv4 and IPv6 MIBs were **disassociated**
- Currently, IPv4 and IPv6 use unified MIBs



IPv6 MIBs /3

Standardization status at IETF

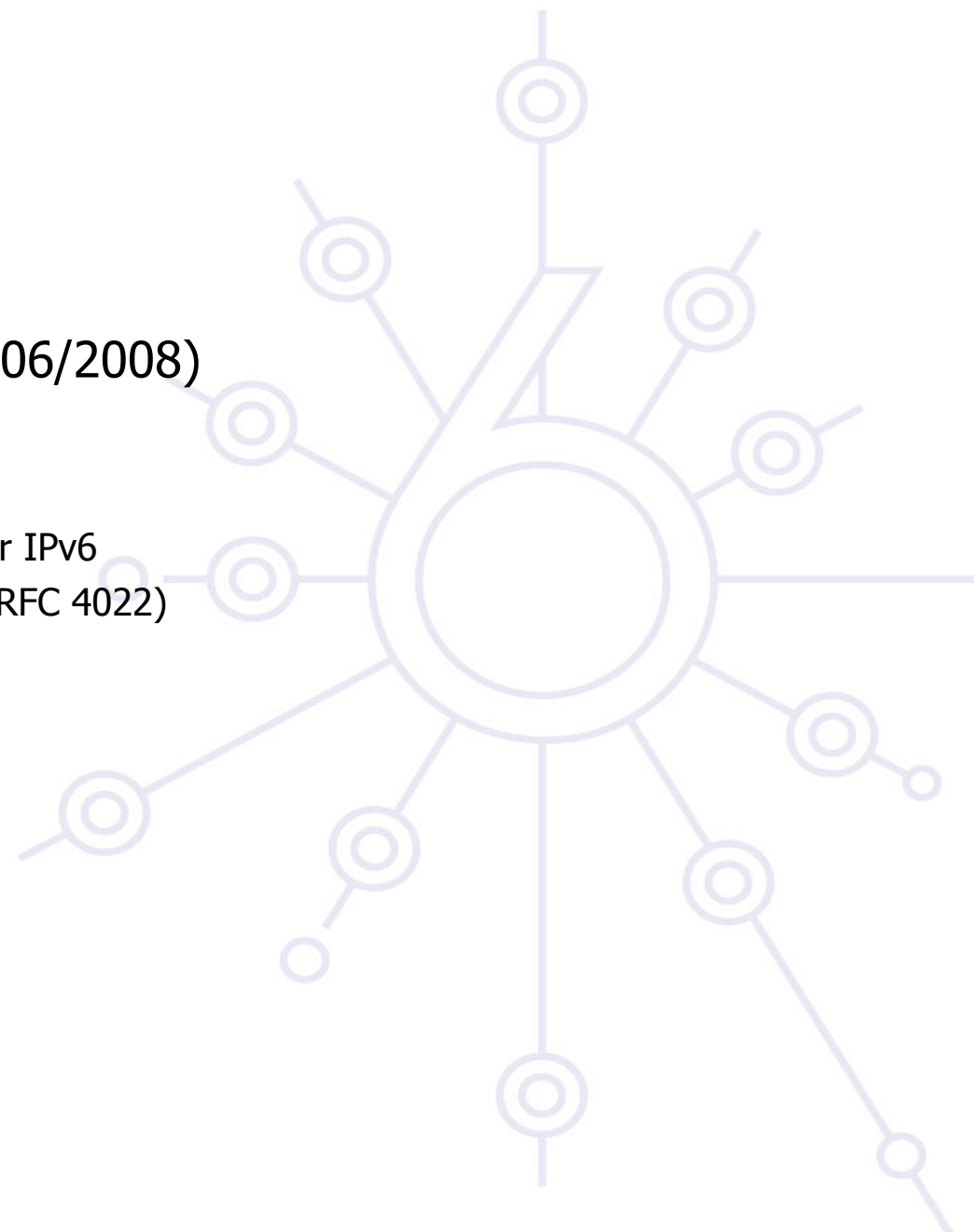
Today : **unified MIBs** are on standard track.



IETF MIB Status /4

BGP MIB v6:

- draft-ietf-idr-bgp4-mibv2-07 (06/2008)
 - Expires in Dec 2008
 - Includes IPv6
 - reference to RFC2545: BGP4 for IPv6
 - Reference to unified TCP MIB (RFC 4022)



IPv6 MIBs implementation/1

Cisco

- Private Cisco MIBs implement RFC 2011 (IP) & 2096 (Forwarding) updated drafts
- Work on implementing the new standards: **Private MIBs based on standards: traffic counters available (packets and bits) on 12.0(33)S. Available also on C7600:**
 - CISCO-IETF-IP-MIB
 - CISCO-IETF-IP-FORWARD-MIB
- Also, information available from CLI (if private MIBs not available)
 - `show interface accounting`
 - ...

Cisco: IPv6 CLI

"show interface accounting"

Differentiate IPv4/IPv6 counters at the interface level for all Cisco routers, except for:

- Catalyst **6500** / Cisco **7600** supervisor engine 720:
Counts only for packets that are software switched, not the hardware switched packets
- GSR:
 - **'show interface counters'** correctly counts IPv6 traffic and separates ingress and egress traffic
 - **Engine 3:**
 - * OUTPUT IPv6 traffic is counted under IPv6 (correct)
 - * INPUT IPv6 traffic is counted under IP (will get corrected)

IPv6 MIBs implementation/2

Juniper

- MIB based on (old) RFC 2465
 - with different counters for IPv4 and IPv6 traffic
- Or based on filters to collect IPv6 traffic:
 - Eg: Geant monitoring

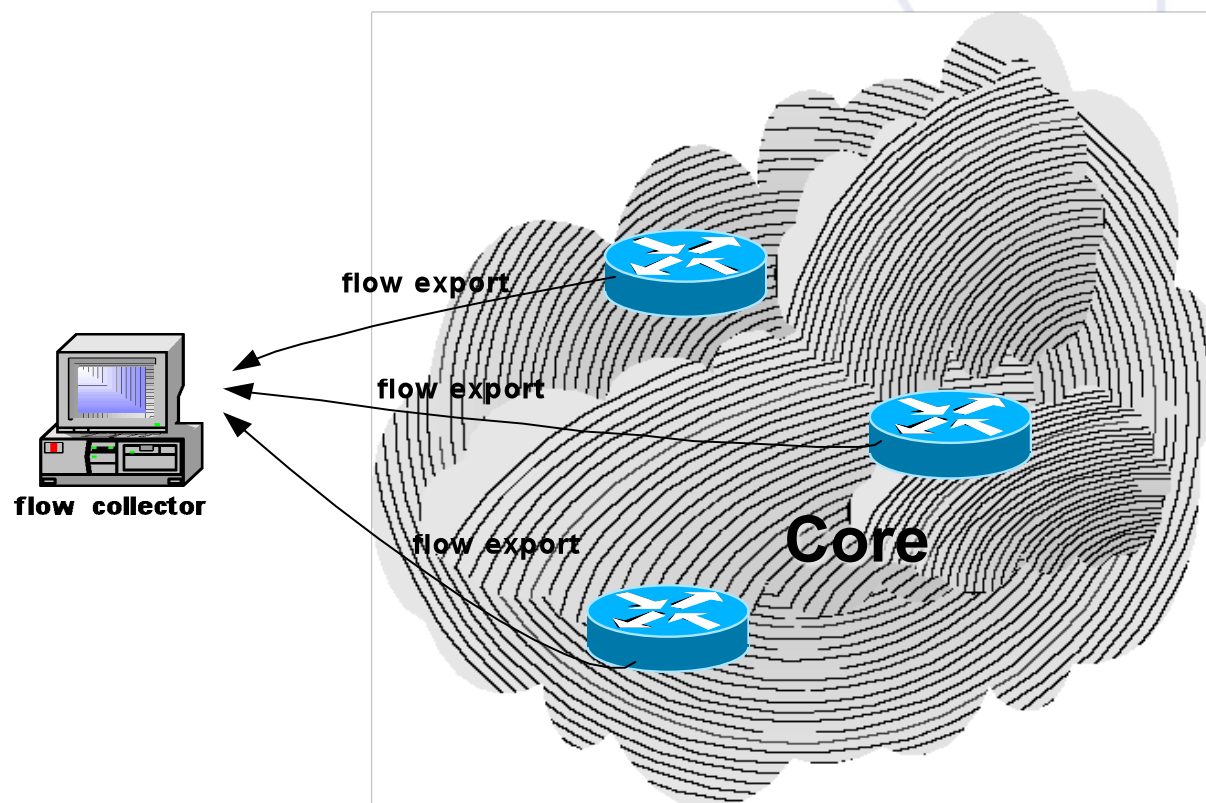
=> Expected : unified MIBs implementation

IPv6 MIBs implementation/4

Net-SNMP (Carnegie Mellon Univ)

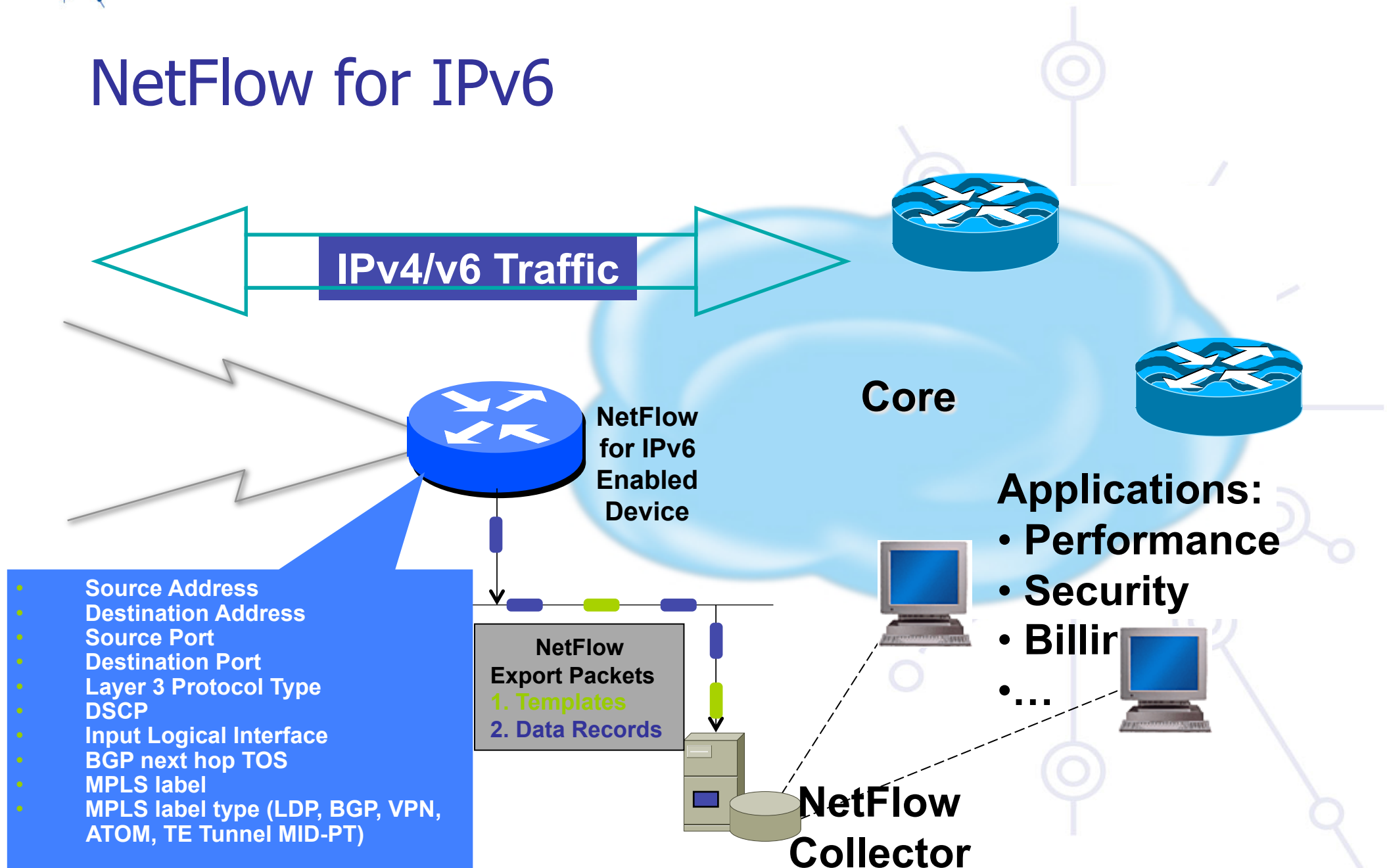
- <http://net-snmp.sourceforge.net/>
- IPv6 support from version 5.0
- RFC 2452: TCP/IPv6
- RFC 2454: UDP/IPv6
- RFC 2465: IPv6
- RFC 2466: ICMPv6
- RFC 3291: (new) textual convention for representing Internet Addresses

Netflow & IPFIX model



Flow= set of packets belonging to the same application between a Source/Destination couple

NetFlow for IPv6

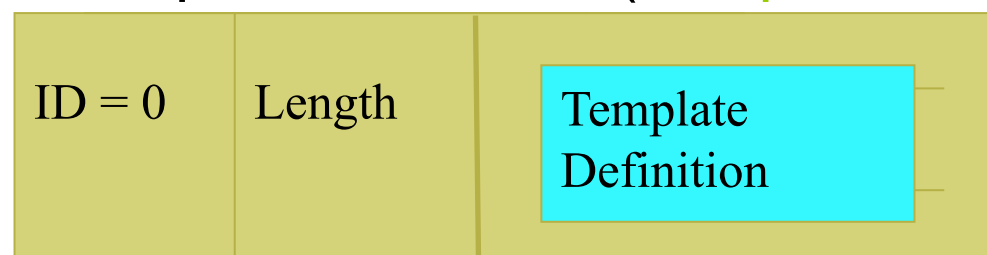


NetFlow for IPv6

Packet



Template Definition (Template FlowSet)



Flow Records (Data FlowSet)



Record



Remote access via IPv6

Use native connectivity when available

- Rather easy if you are operating dial-in pool or you are an ADSL service provider
- ... and even more easy if your home ISP provides IPv6 connectivity
 - Like Free and Nerim in France

Use (Open)VPN / IPSec VPN

Use tunnel broker service – rather suboptimal ?

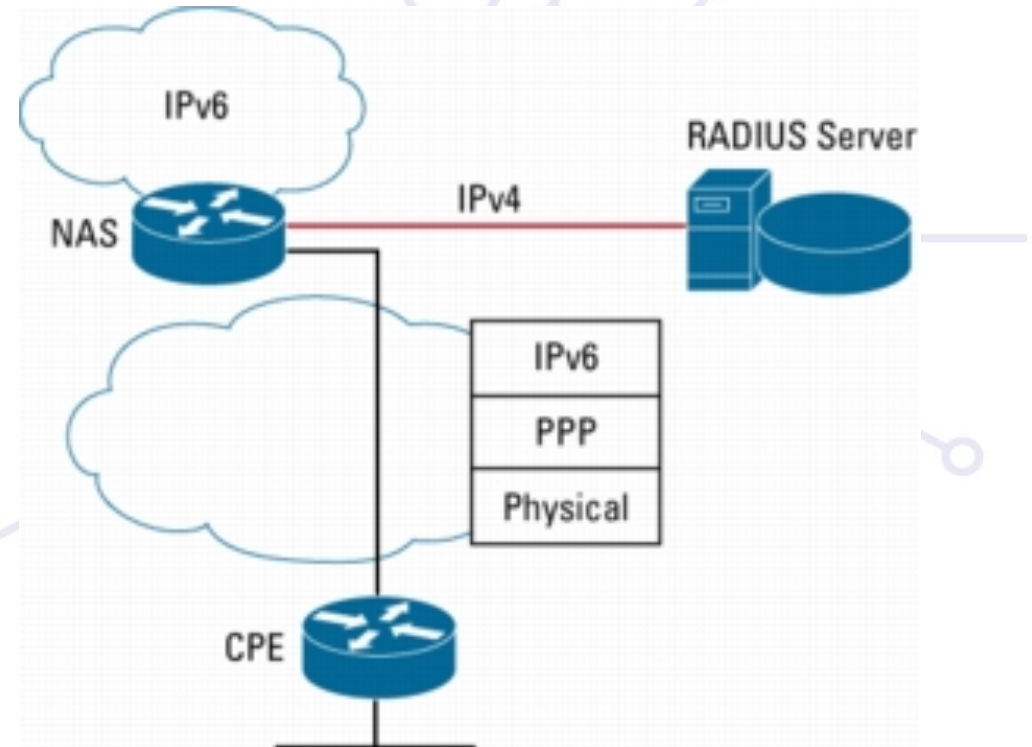
Use 6to4/6rd if you have global IPv4 address

- Good 6to4 relay connectivity is a must – beware of asymmetric routing!

Use Teredo/software if you have NAT or multiple level of NATs

Remote access via IPv6 - PPP

- **The dial-up connection uses a modem and the PSTN service in order to get connection to remote devices.**
 - Most cases use PPP (Point-to-Point Protocol), which gives a standard method to transport the datagrams of several protocols over point-to-point links (RFC1661, 2153, 5342) - PPP has been updated to support the transport of IPv6 datagrams (RFC5072)



PPP and IPv6

PPP protocol has three main parts

- Definition of the encapsulation method of the IPv6 datagrams over the point- to-point link (IP6CP)
- LCP (Link Control Protocol) used to establish, configure and test the connection at link layer
- NCP (Network Control Protocol) used to establish and configure the connection at network layer

IPv6 operation:

- negotiates one link local address (fe80::/64) between the end points or peers
- Could negotiate datagram compression via IP6CP (IPv6 Control Protocol)
- PPP does not give global IPv6 addresses but link local - The global IPv6 addresses must be configured by other means
 - Manual configuration
 - Autoconfiguration (RA)
 - DHCPv6

PPP and IPv6 - implementations

Routers:

- Cisco
- Juniper

Hosts:

- Windows Vista and Microsoft Windows Server 2008
 - Windows XP: Cfos IPv6 link http://www.cfos.de/ipv6_link/ipv6_link_e.htm
- Linux, *BSD (including Mac OS X), Solaris

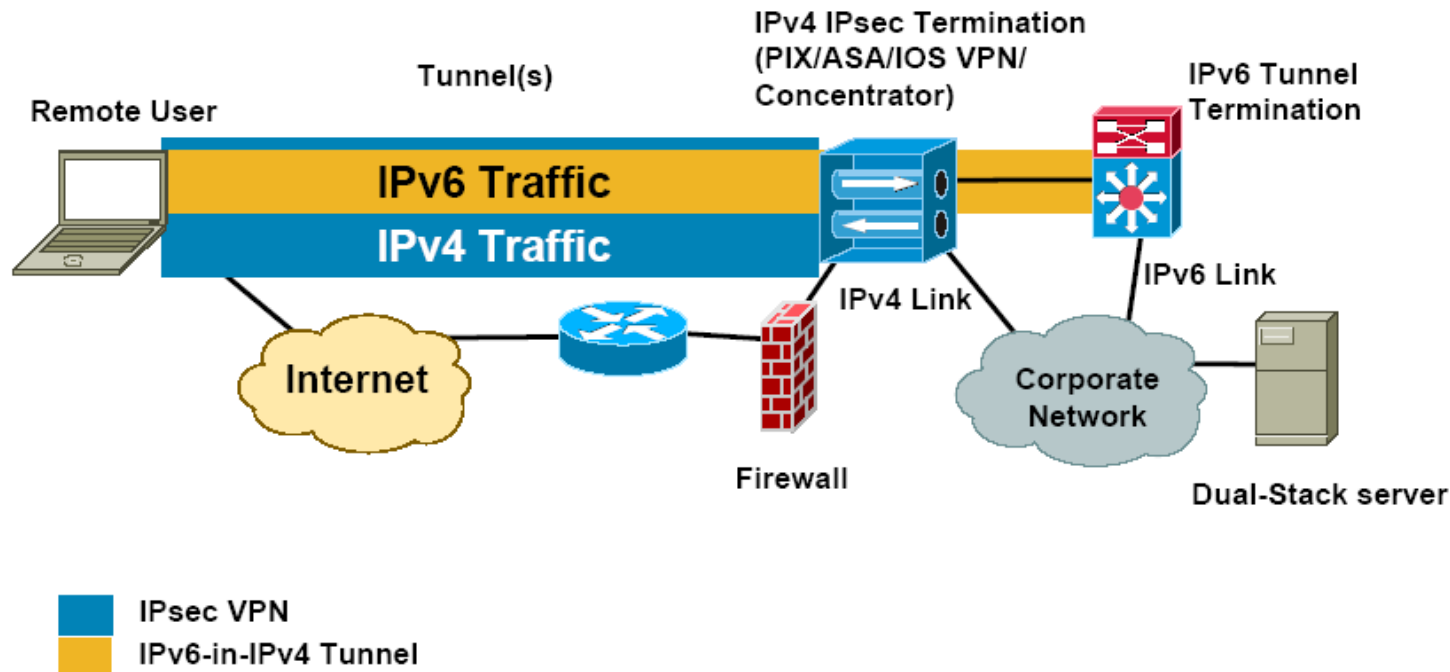
Opensource:

<http://sourceforge.net/projects/pppcbcp>

<http://freshmeat.net/projects/pppd>

Remote Access with IPSEC – or other VPNs

IPv6-in-IPv4 Tunnel Example



IPv6 load balancing

- Server clusters
 - Opensource solution: *BSD pf (<http://www.openbsd.org/faq/pf/>), Linux LVS after 2.6.28 (http://kb.linuxvirtualserver.org/wiki/IPv6_load_balancing)
 - Commercial platforms: Veritas Cluster Server, BigIron F5, Windows Server 2008 - Network Load Balancer, Citrix Netscaler
- First-Hop Redundancy:
 - HSRPv6 (Cisco only)
 - VRRPv6 - standardisation at IETF
 - GLBP – (Cisco only) with redundancy with loadbalancing
 - NUD (Neighbor Unreachability detection)- see next slide
- Traffic loadbalancing
 - Multilink PPP - supported if multilink PPP supported
 - Equal-Cost Multi-Path routing - if IPv6 routing supported...
 - Ethernet Link Aggregations - L2 solution

Implementing default gateway redundancy

**If HSRP, GLBP or VRRP for IPv6 are not available
 NUD can be used for a good HA at the first-hop
 (today this only applies to the Campus/
 Datacenters ... HSRP is available on routers)**

- (config-if)#ipv6 nd reachable-time 5000

**Hosts use NUD "reachable time" to cycle to next
 known default gateway (30 seconds by default)**

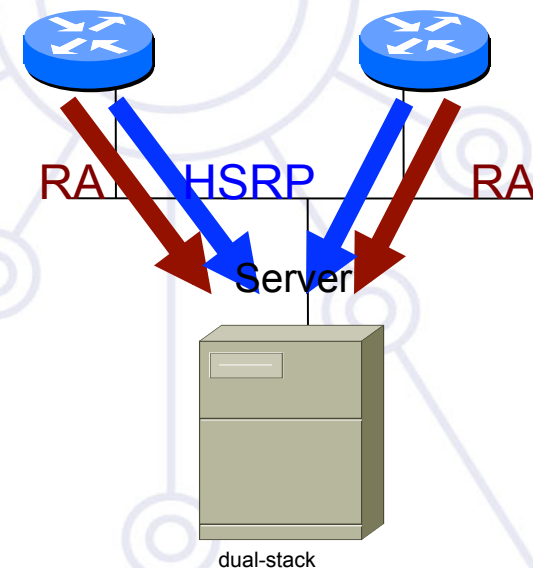
Default Gateway : 10.121.10.1

fe80::211:bcff:fec0:d000%4

fe80::211:bcff:fec0:c800%4

Reachable Time : 6s

Base Reachable Time : 5s



Outline

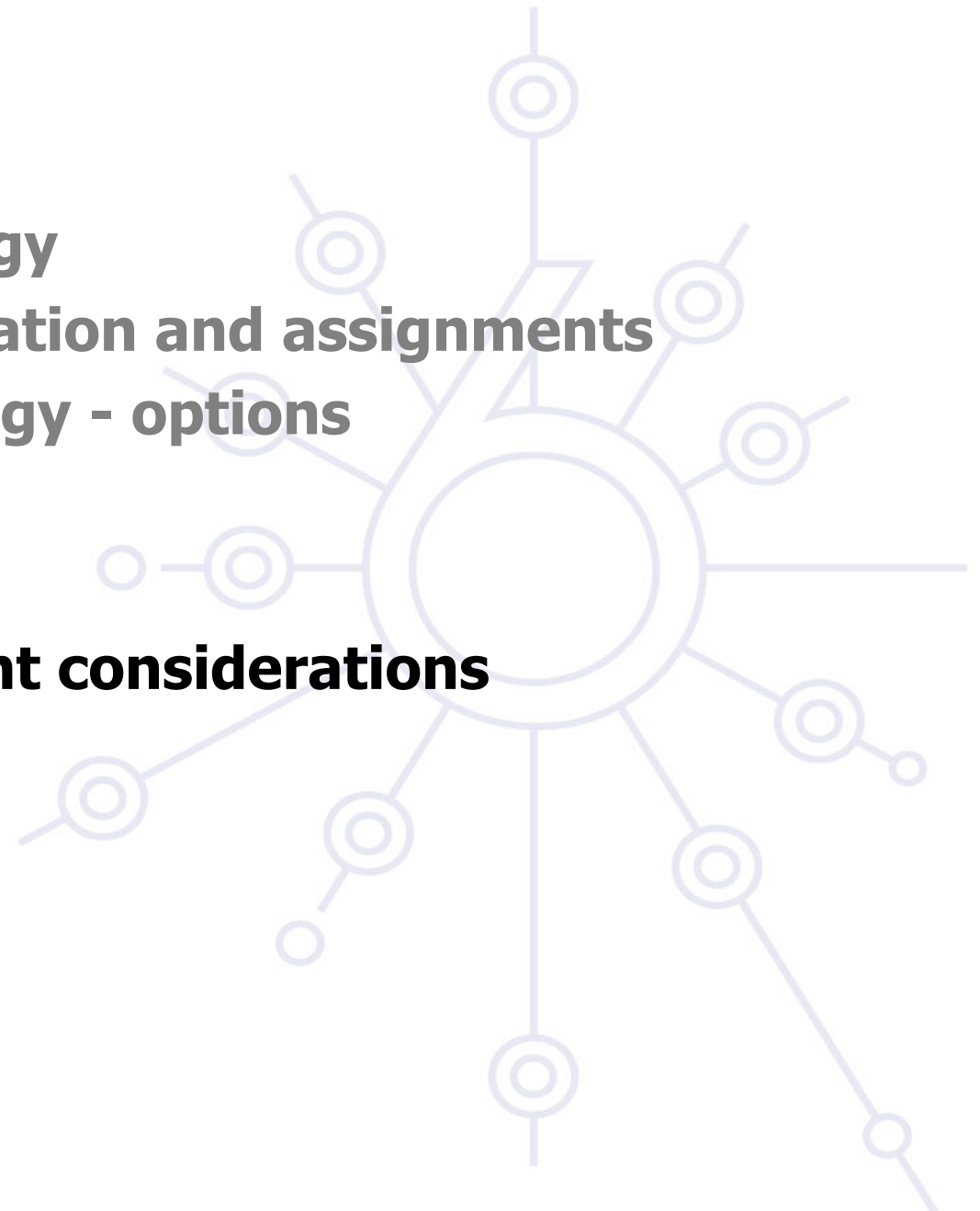
Campus deployment strategy

Campus IPv6 address allocation and assignments

Campus deployment topology - options

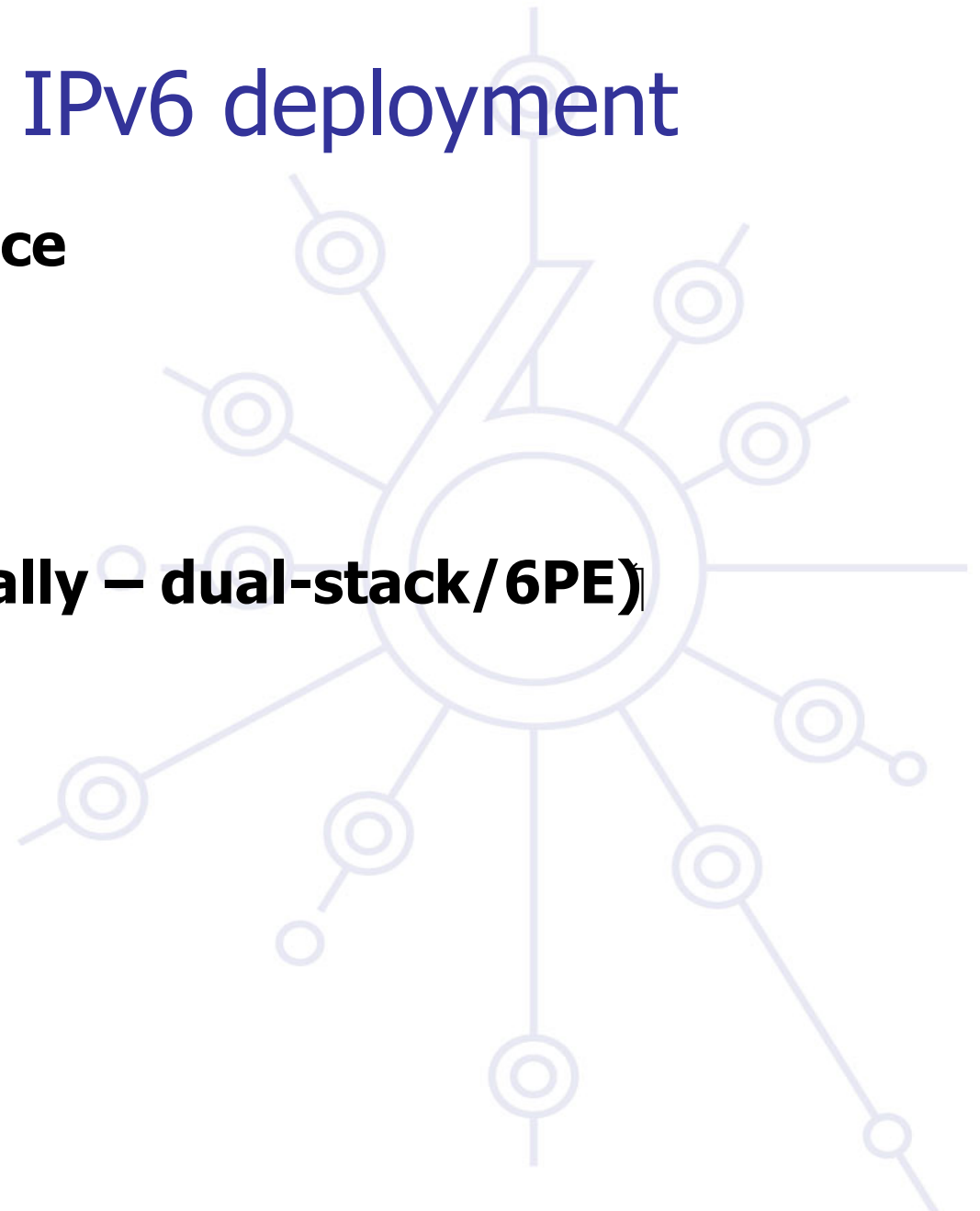
Campus services

Service provider deployment considerations



Outline of NRENs/ISP IPv6 deployment

- 1. Obtain IPv6 address space**
- 2. Plan the addressing**
- 3. Plan the routing**
- 4. Test in a small case**
- 5. Deploy IPv6 (incrementally – dual-stack/6PE)**
- 6. Enable IPv6 services**



Getting IPv6 prefix for LIRs/ISPs

Global IPv6 RIR rules

- <http://www.ripe.net/ripe/docs/ipv6.html>
- simple rules for LIRs
- IPv6 service should be provided
- detailed plan
- Usually /32 allocation

Establishing global rules was not easy

- Different structure in different RIR regions: ISP, NIRs/LIRs, LIRs

What about IX? – slightly different rules

- Infrastructure addresses
- Routable /48 address

You can have PI IPv6 also -> direct contractual agreement with RIRs

- be multihomed!
- Routable, shorter than /48

RIPE entries /1

```
whois -h whois.ripe.net 2001:0738::
```

```
inet6num:      2001:0738::/32
netname:       HU-HUNGARNET-20010717
descr:        Hungarnet IPv6 address block
               Hungarian Research & Educational Network
               Budapest, Hungary
country:      HU
mnt-by:       RIPE-NCC-HM-MNT ←New mandatory
mnt-lower:    NIIF6-MNT ←New mandatory
status:       ALLOCATED-BY-RIR ←New
```

RIPE entries /2

possible values of STATUS field

- ALLOCATED-BY-RIR – Allocated address space by RIR to LIR.
- ALLOCATED-BY-LIR – Allocated address space by LIR to smaller registries/institutions
- ASSIGNED – Assigned to end-users

RPSLng is in production (at least in RIPE region)

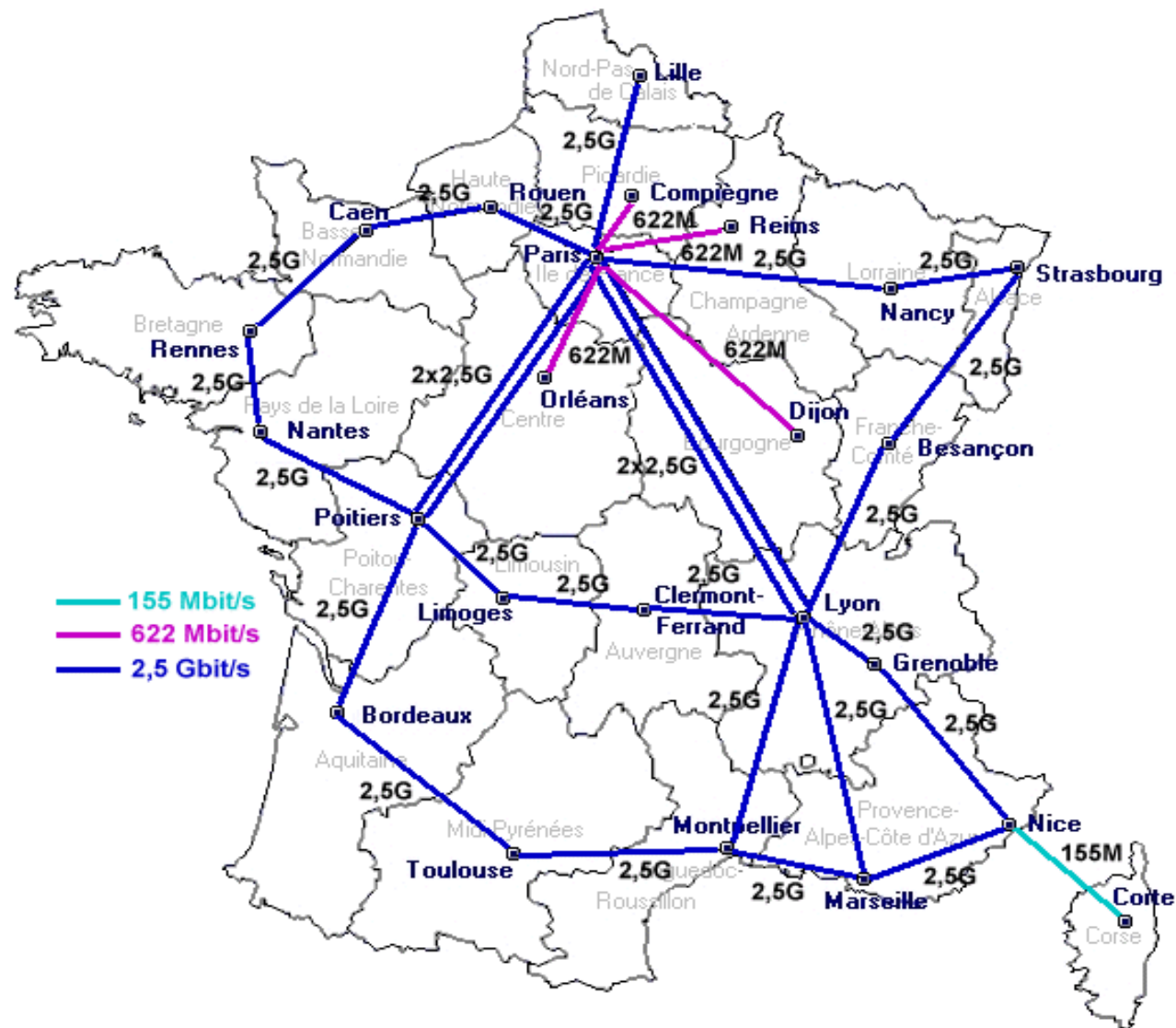
Reverse delegation is strongly recommended



deploy

RENATER IPv6 numbering

Renater-3: national backbone



Addressing

Hierarchical addressing

Renater

- Prefix = 2001:0660::/32
- Allocated by the RIR (RIPE NCC)

Regional Nodes

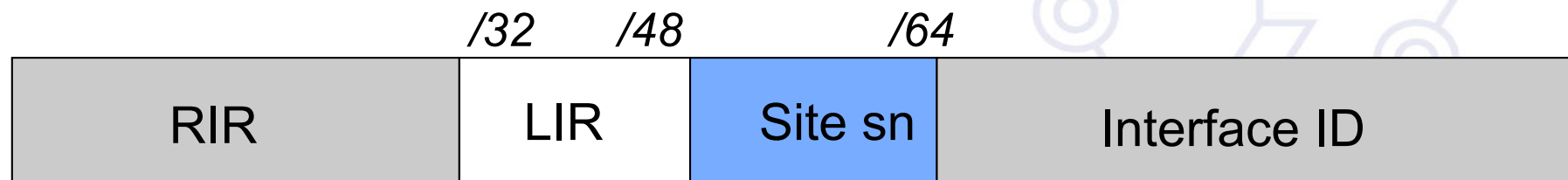
- POP-ID = 2001:0660:xy::/40

Site

- Site-ID : a /48
 - from RN's prefix (/40) it's connected to
- Site-IDs allocated by Renater (LIR)
- 16 bits are reserved for the site topology



Addressing



2001:0660:



2001:0660:3000:/40 Paris NRI
 2001:0660:3300:/40 Paris Jussieu RI
 2001:0660:4400:/40 Lille RI
 2001:0660:5400:/40 Marseille RI
 (...)

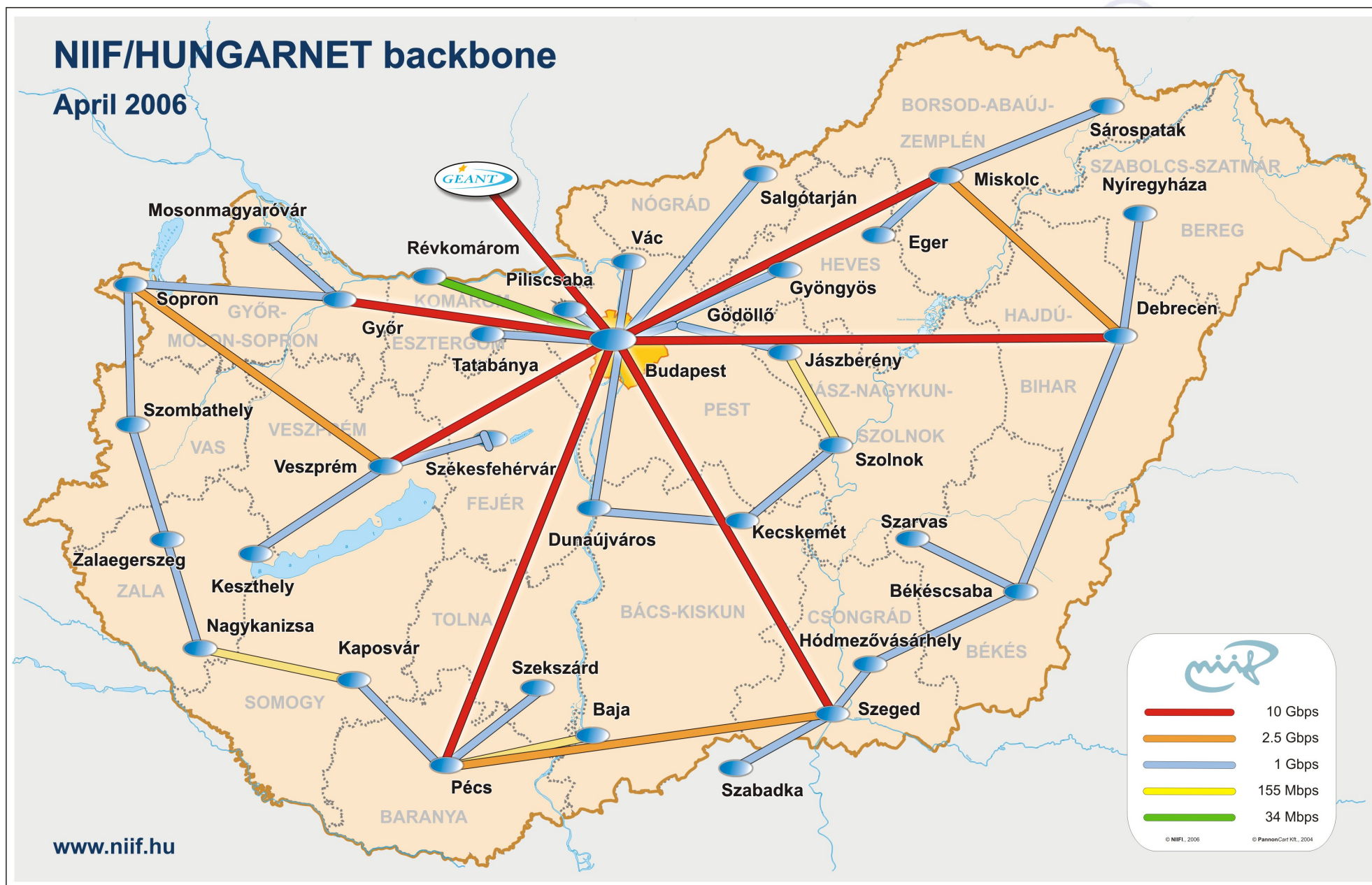
2001:0660:300x:/48



deploy

NIFF/HUNGARNET IPv6 numbering

NIIF/HUNGARNET network



IPv6 deployment at NIIF/Hungarnet

Initial IPv6 deployment:

- MPLS based backbone: 6PE with additional dual stack routers + sometimes tunnels at connected institutions

Second phase (2004):

- Router upgrade for HW based IPv6 forwarding
- Used features
 - Routing: IPv4 (unicast, multicast), IPv6 (unicast only), OSPFv2, OSPFv3, BGP, MPLS VPNs
 - Netflow, minimal QoS
 - IPv6 multicast with additional dual stack routers with tunnels

Third phase (2008):

- Software upgrade for IPv6 multicast support
- Netflow v9 support

IPv6 address space – based on flexible address allocation RFC3531

Location	IPv6 POP addressing:
CNTRL (Central)	2001:0738:0::/36
Gödöllő (Szent István University)	2001:0738:58::/44
BME (Budapest University of Technology and Economics)	2001:0738:2000::/44
KFKI (Research Institute on Physics)	2001:0738:5000::/44
SZEGED (University of Szeged)	2001:0738:7000::/44
MISKOLC (University of Miskolc)	2001:0738:6000::/44
PECS (University of Pécs)	2001:0738:7800::/44

Site addressing

Each site (including site infrastructure) gets /48:

- each NIIF managed site the 16 bit SLA is allocated based on the following convention: <SLA> = Address segmentation within the POP
- Where for <SLA>:
 - Range: 0000 till 00FF: Loopback addresses
 - Range: 0100 till 01FF: Intra-pop point-to-points (if it necessary to number it)
 - Range: 0200 till 02FF: connections to HUNGARNET member of institution
 - Range: 0300 till 03FF: external IPv6 connectivity (e.g. local IPv6 peering)
 - Range: 0400 till 04FF: POP Local Ethernets

IPv6 loopback addresses

loopback address will also be used for operational and management actions on the equipment, and for routing protocols like iBGP, which will use these addresses for terminating the peering-sessions.

Loopback addresses have typically a prefix mask of /128. This will avoid unnecessary unused addresses although address conservation is not really an issue in IPv6.

Link IPv6 addresses?

Not necessary!

- OSPFv3 is working with link-local
- IS-IS not necessary

IGP table can be quite small!

- Reduces the convergence time

Customer network is propagated into BGP (even if static routes are used)

- not with redistribute
- with network statement

Drawback:

- Traceroute can pick up arbitrary IPv6 address as a reply source -
- Avoid - configure on each point-to-point links:
 - `ipv6 unnumbered loopback0`

Link IPv6 addresses -other options

/127: not a good idea

- the all-zeros address is supposed to be the any router anycast address although this is not widely implemented today - see more RFC 3627

/126: works

- although the top 128 addresses are reserved for anycast stuff

/120: no clashes with top 128 anycast addresses

/112: alignment is on a nice colon boundary

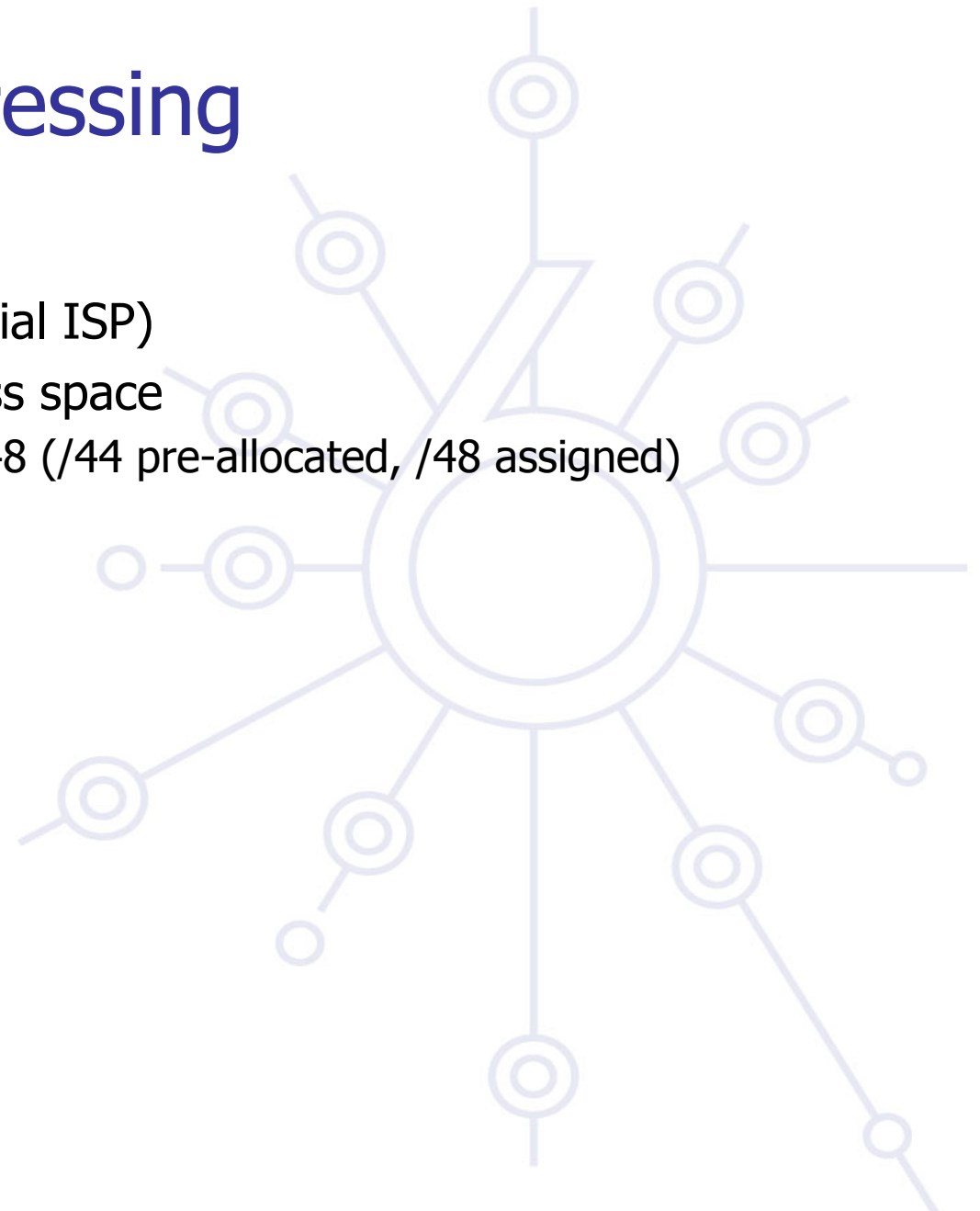
/64: based on RFC 3513

- Allows to use EUI-64 addressing
- advisable for point-multipoint and broadcast link scenarios

Customers' Nets Addressing

Two possibilities

- Uses its own prefix (Commercial ISP)
- Uses NIIF/Hungarnet's address space
 - 2001:0738:<customer id>::/48 (/44 pre-allocated, /48 assigned)



Conclusion

Preparing an IPv6 addressing plan is a bit complex

Plan it in advance ...

- Not forgetting your PoPs equipment (loopbacks, admin LANs, interconnects ...)

Draw benefit from aggregation

- Smaller routing tables to manage (even in the core)
- Less prefixes to advertise to BGP peers

Lot of people have an experience yet ...

- Not necessary to reinvent the wheel ;)

Summary

Campus deployment strategy

- Coexistence mechanism ?
- Getting an IPv6 prefix
- ... and external IPv6 connectivity
- Decide a security policy for IPv6 traffic

Campus IPv6 address allocation and usage

- Work out an addressing plan
- Decide which address allocation mechanism will be used

Campus deployment topology - options

- Start IPv6 deployment
- How to remote access the campus ?

Campus services

- Enable services for IPv6
 - Starting with the DNS
- Enable management and monitoring tools
- Enable IPv6 on hosts



deploy

Questions ...