Practical VoIP Peering

Klaus Darilion
enum.at
klaus.darilion@enum.at

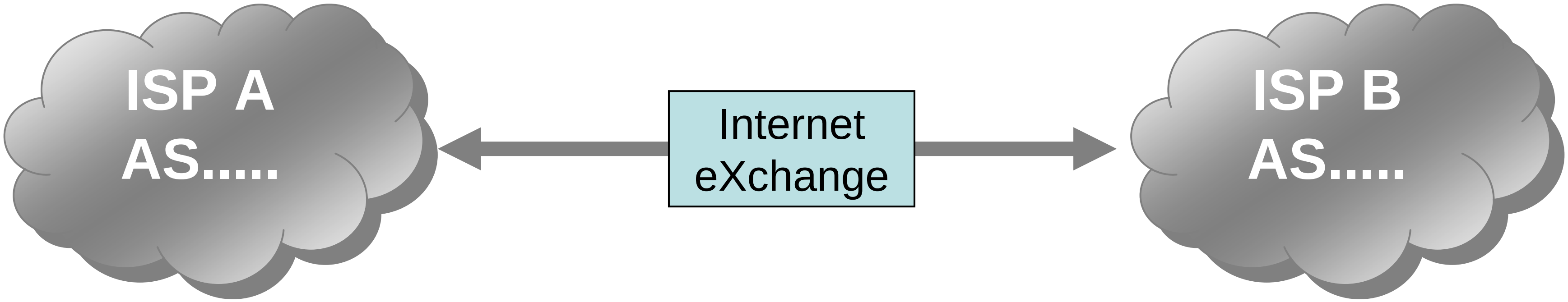
Peering*

- Peering: negotiation of reciprocal interconnection arrangements between service providers
  - Layer 3 peering
  - Layer 5 peering

*definitions from draft-ietf-speermint-terminology-06.txt

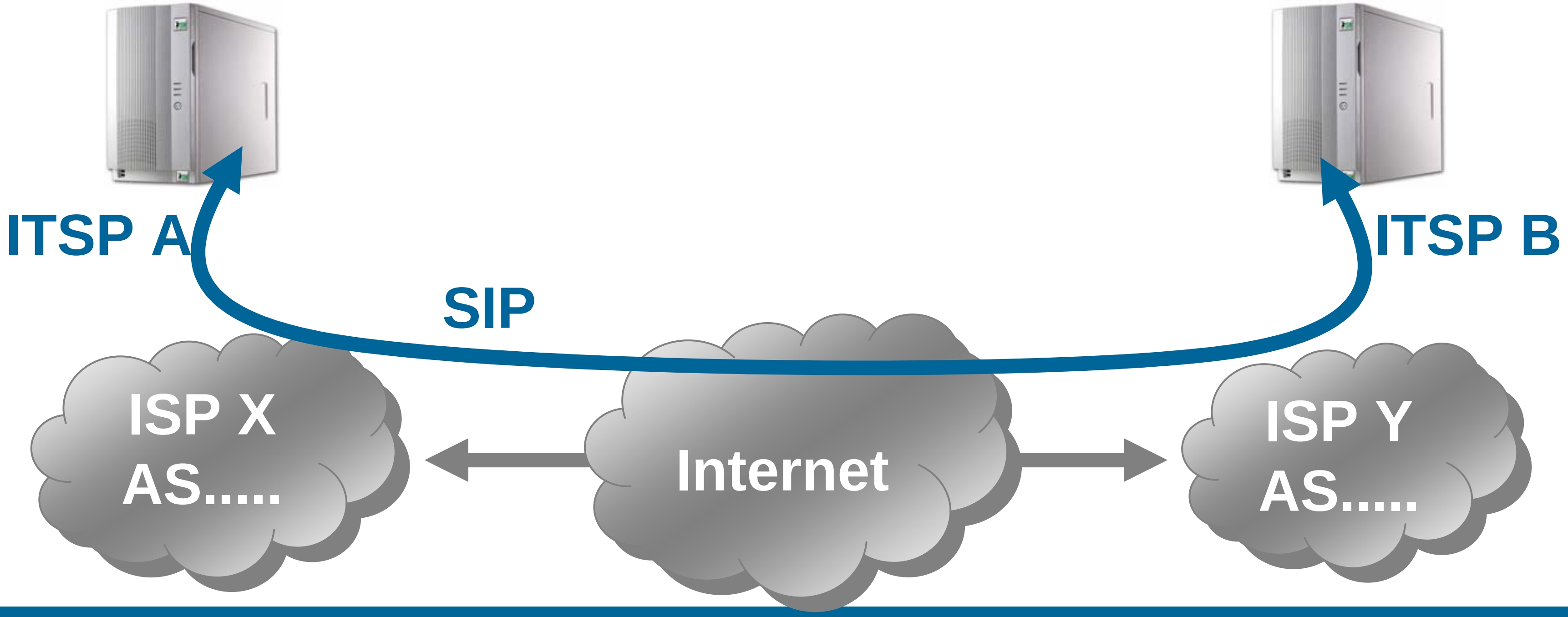
Layer 3 Peering

- interconnection of two service providers' networks for the purposes of exchanging IP packets which destined for one (or both) of the peer's networks



Layer 5 Peering = VoIP Peering

- interconnection of two service providers for the purposes of routing SIP signaling
- this presentation is about L5 peering



Why is L5-peering needed?

- SIP like Email/SMTP → no explicit peering needed
  - requires an “open” SIP proxy:
    - allow incoming SIP requests (from non-local domains)
    - allow outgoing SIP requests (to non-local domains)
  - examples: iptel.org, freeworlddialup, gizmoproject

Why is L5-peering needed?

- an “open” SIP proxy raises issues, e.g.:
  - SPIT (VoIP SPAM)
  - QoS
  - billing (interconnect fees, transit fees)
  - security (authentication, DoS, ...)

Peering Terminology*

- “open” connectivity
  - SMTP-style
- static peering
  - pre-defined peering partners
- dynamic peering
  - peering partners not known in advance
- bilateral peering vs. federation peering

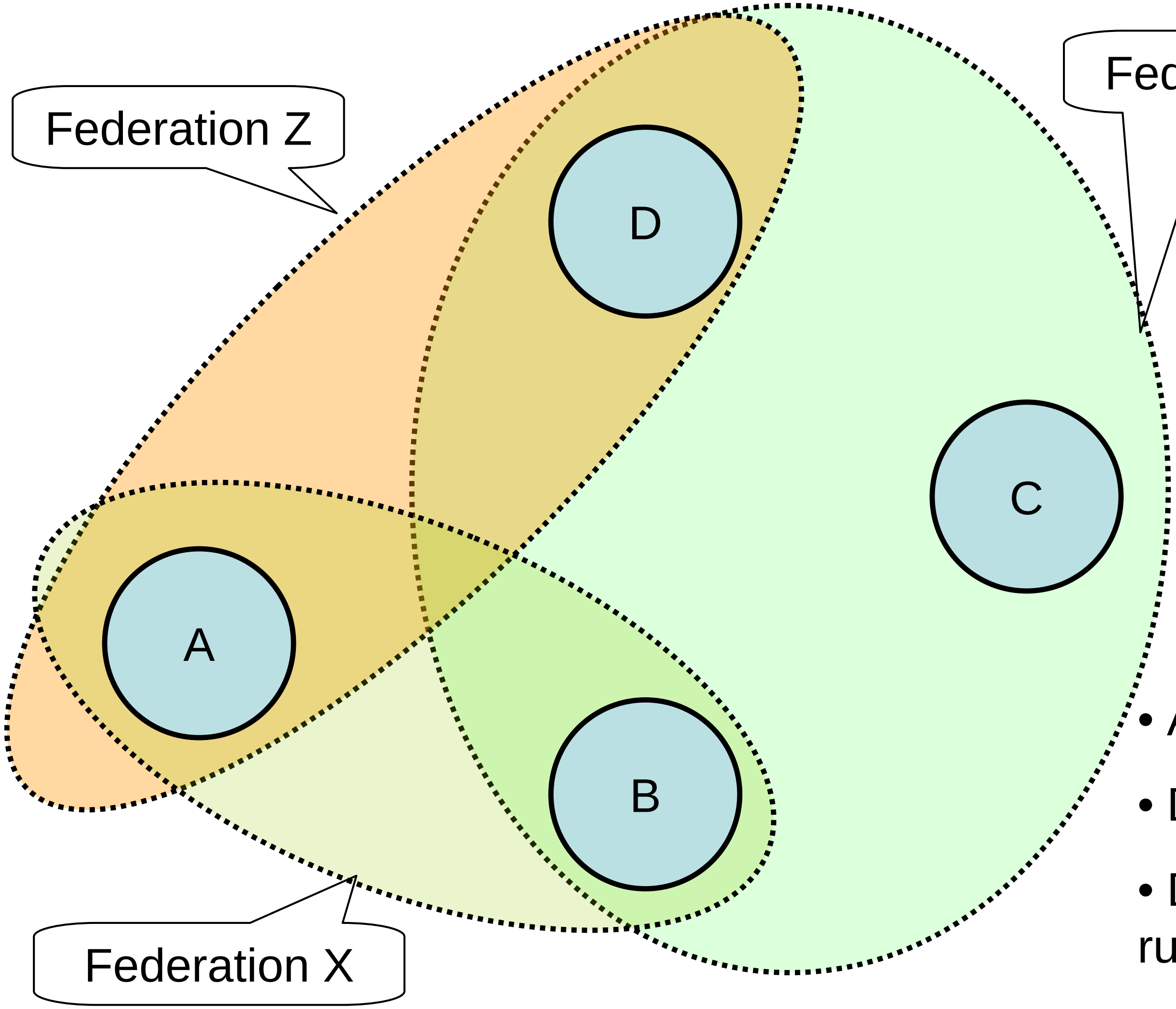
*my definition

Federation*

- A group of ITSPs agree to receive calls from each other via SIP
  - agree on administrative rules (settlement, abuse-handling, ...)
  - agree on technical details of the interconnection
- an ITSP can be a member of
  - no federation
  - a single federation
  - multiple federations
  - can have any combination of bi-lateral and multi-lateral (i.e., federated) interconnections.

*definition from draft-ietf-speermint-terminology-06.txt

Federations



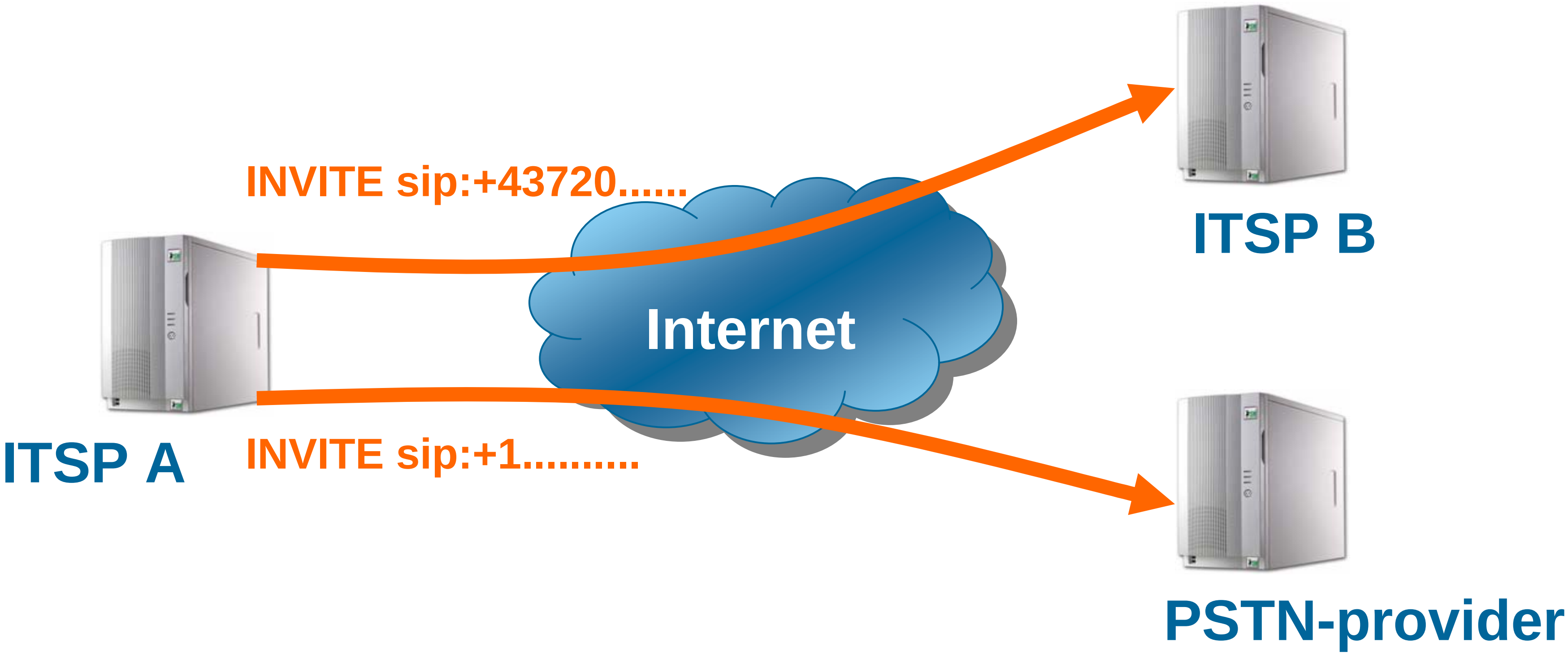
Required:
federation discovery and matching

Call setup details are intra-federation.

- A ←→ B by rules of X
- D ←→ A by rules of Z
- D,B,C to each other by rules of Y
- No federated path from A to C

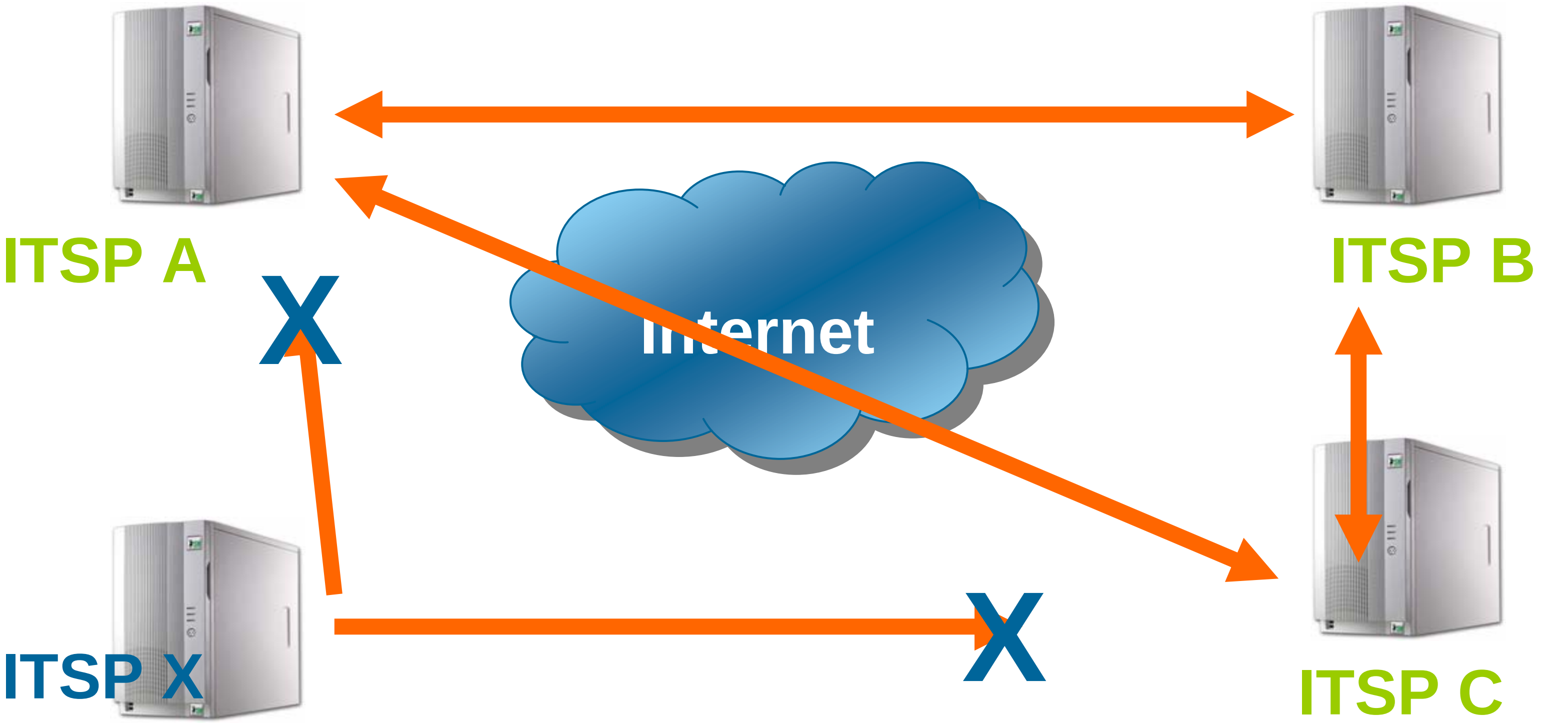
Static Peering

- peering partners known in advance
- typically block routing (phone numbers)



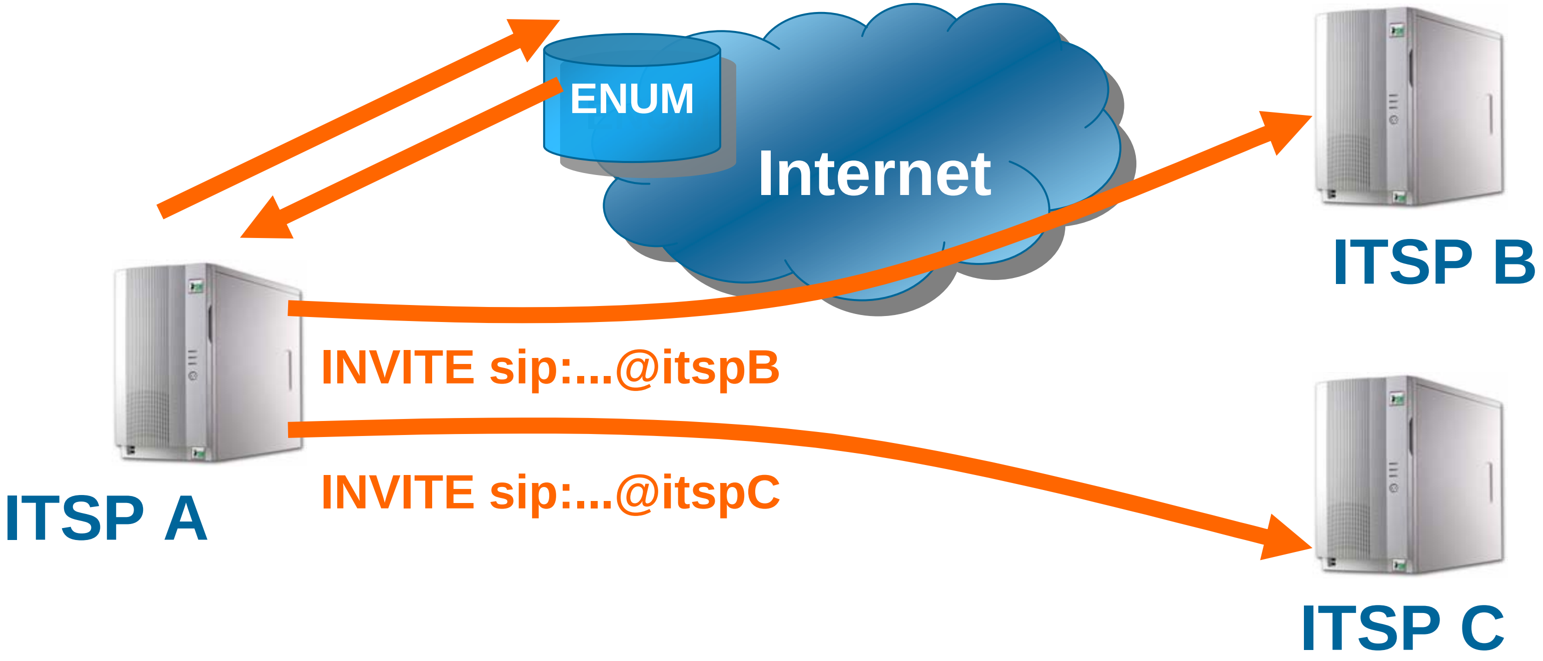
Static Peering

- only traffic between known peers



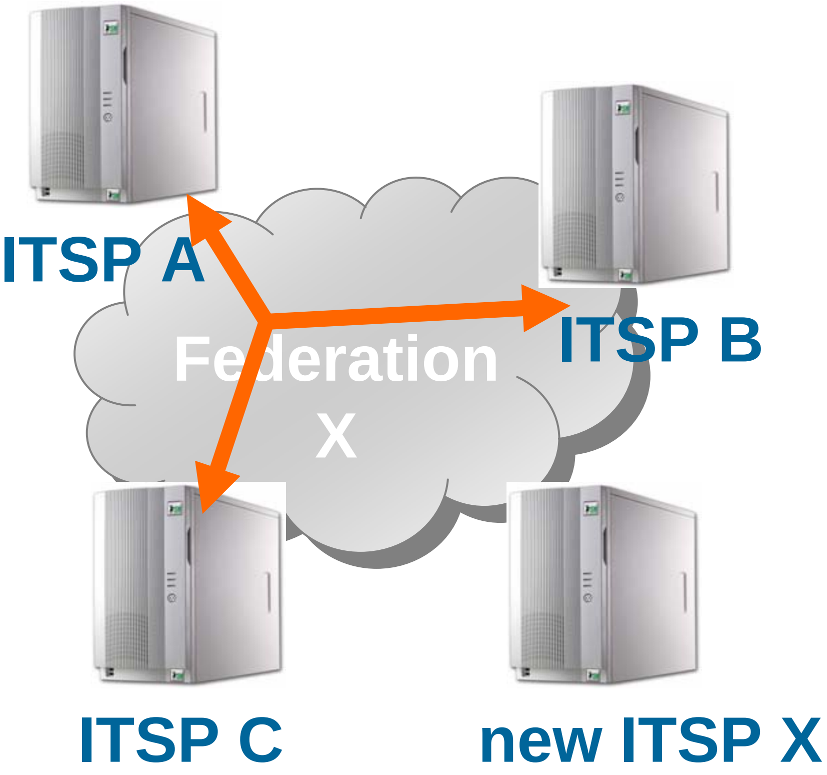
Dynamic Peering

- peering partners **NOT** known in advance
- usually an E.164-URI mapping (ENUM)



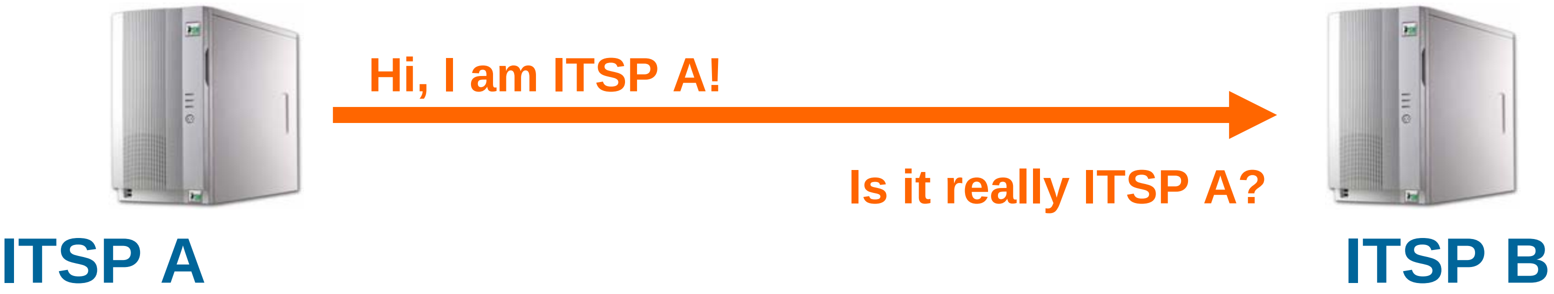
Static vs. Dynamic Peering within Federations

- new ITSP joins federation
- static peering
  - A, B and C have to configure peer X
- dynamic peering
  - ITSP X announces federation membership
  - at A, B and C no configuration needed



Peering Requirement

- **authentication**, authorization, accounting



Hi, I am ITSP A!

Is it really ITSP A?

ITSP A

ITSP B

- authentication is essential for peering
  - layer 1/2: dedicated links
  - layer 3: IP based (TCP or UDP+IPSEC)
  - layer 5: TLS, cookie/token, SIP Identity ...

Components

- flexible SIP proxy
- ENUM lookup
- TLS
- domainpolicy module

➔

Peering with Openser

- config snippets
  - static peering, IP based authentication
  - static peering, TLS based authentication
  - dynamic peering with TLS

Static Peering - IP

- outgoing: block based routing (one "if" for each peer)

```
if (uri =~ "^sip:\+1") { # USA
  sethostport(“1.2.3.4:6060;transport=tcp");
} else if (uri =~ "^sip:\+4359966") {
  # austrian ITSP xyz
  sethostport(“10.10.0.4"); # private VLAN
} else if (uri=~"^sip:\+491234") {
  # german ITSP foobar
```

or using openser’s LCR module

`load_gws(), next_gw()`

Static Peering - IP

- outgoing: domain based routing

```
if (uri =~ "^sip:*.@itspA") {
  sethostport("peer.itspA;transport=tcp");
} else if (uri =~ "^sip:*.@itspB") {
  # do nothing, current R-URI is fine
} else {
  sl_send_reply("403","untrusted peer");
  ...
```

Static Peering - IP

- incoming: authentication based on IP address (one "if" for each peer)

```
if ((src_ip==1.1.1.1)&&(proto==TCP)) {
  # from ITSP foobar
  route(10);
} else {
  # unknown peer
  sl_send_reply("403","stay away");
```

or using openser’s LCR module

```
from_gw()
```

Static Peering - TLS

- authentication based on TLS: TLS config (one pair for each peer)

```
# socket based TLS server domain, used by itspB
tls_server_domain[local_ip:port] {
    # show the following cert to incoming peer
    tls_certificate = "/certs/signedByItspB/mycert.pem"
    tls_private_key = "/certs/signedByItspB/myprivkey.pem"
    # validate presented certificate against this CA
    tls_ca_list     = "/certs/myself/myCa"
    tls_verify_client = 1
    tls_require_client_certificate = 1
}

# socket based TLS client domain for peering with peerX
tls_client_domain[remote_ip:port] {
    # show the following cert to peer
    tls_certificate = "/certs/signedByItspB/mycert.pem"
    tls_private_key = "/certs/signedByItspB/myprivkey.pem"
    # validate presented certificate against this CA
    tls_ca_list     = "/certs/myself/myCa"
    tls_verify_server = 1
}
```

Static Peering - TLS

- incoming routing: authentication based on TLS

```
if (proto==TLS) {
   # already authenticated by TLS stack
   route(10);
} else {
   # unknown peer
   sl_send_reply("403","use TLS");
```

- outgoing routing: TLS is transparent

```
# request/destination URI contains transport=TLS
t_relay();
```

Static Peering - Conclusion

- requires manual configuration
  - outgoing
  - incoming
- does not scale
  - either complex IP address management or
  - complex certificate configuration
- dynamic peering not possible

Solution: Domain Policy

- domain based policy announcing (draft-lendl-domain-policy-ddds)
  - callee domain (ITSP) announces peering policy in DNS
    - technical
    - federation
  - caller applies policy
- implemented in openser’s domainpolicy module

Domain Policy Example

```
$ORIGIN itspB.
IN NAPTR 10 10 ("U" "D2P+SIP:fed"
      "!^.*$!http://sipxconnect.example.org/!" . )
IN NAPTR 20 10 ("U" "D2P+SIP:fed"
      "!^.*$!http://myfederation.foobar/!" . )
IN NAPTR 30 10 ("U" "D2P+SIP:std"
      "!^.*$!urn:ietf:rfc:4474!" .)
```

- itspB accepts calls from:
  - members of the federations
  - peers identified by RFC4474 (Authenticated Identity Management)

Openser Domainpolicy Howto

1. configure domainpolicy table with federation policy
2. configure TLS (preferred authentication method)
3. announce domainpolicy (federation membership) in DNS
4. query and apply domainpolicy

1. Configure Federation Policy

- sample federation policy
  - federation identifier: **http://fedx/**
  - TLS (federation signs certificates)
  - prefix peer's URI with "fedx" to find ingress proxy
- openser's domainpolicy table

```
+----+--------------+------+---------------------+------+
| id | rule         | type | att (avp name)      | val  |
+----+--------------+------+---------------------+------|
| 1  | http://fedX/ | fed  | s:domainprefix      | fedx |
| 2  | http://fedX/ | fed  | s:transportoverride | tls  |
| 3  | http://fedX/ | fed  | i:400               | fedx |
+----+--------------+------+---------------------+------+
```

2. Configure TLS

```
...
tls_client_domain_avp=400
...
# socket based TLS server domain, used for ingress of federationX
tls_server_domain[local_ip:6061] {
    # show the following cert to incoming peer
    tls_certificate = "/certs/fedX/mycert.pem"
    tls_private_key = "/certs/fedX/myprivkey.pem"
    # validate presented certificate against this CA
    tls_ca_list     = "/certs/fedX/ca"
    tls_verify_client = 1
    tls_require_client_certificate = 1
}
# name based TLS client domain for egress peering with federationX
tls_client_domain["fedx"] {
    # show the following cert to peer
    tls_certificate = "/certs/fedX/mycert.pem"
    tls_private_key = "/certs/fedX/myprivkey.pem"
    # validate presented certificate against this CA
    tls_ca_list     = "/certs/fedX/ca"
    tls_verify_server = 1
}
```

3. Announce Domainpolicy in DNS

```
$ORIGIN itspA.

; announce federation memberships

IN NAPTR 10 10 "U" "D2P+SIP:fed""!^.*$!http://fedX/!" .

;SIP domains

_sips._tcp.fedx  IN SRV 0 0 6061 ingress.itspA.
ingress          IN A            1.2.3.4
```

4. Query and Apply Domainpolicy

```
route[] {
   ...
   # map TN to URI with ENUM
   if ( i_enum_query() ) {
        # check the domainpolicy of the destination
        if (dp_can_connect()) {
               xlog("L_INFO","dp_can_connect succeeded:\n");
               # apply domain policy
               if (dp_apply_policy()) {
                       xlog("L_INFO","  new d-URI = $du\n");
                       route(4);
                       exit;
               }
               xlog("L_INFO","dp_apply_policy failed\n");
        } else {
               xlog("L_INFO","dp_can_connect failed\n");
        }
   } else {
```

Domainpolicy Conclusion

1. only one TLS config-pair per federation
2. no configuration changes needed for new federation members
3. no routing changes needed for new federations
4. also simplifies static peering (DB based)
5. a scaleable solution

Summary

- "open" SIP connectivity unusual – peerings are preferred
- there will be lots of federations (peering fabrics)
- static configuration does not scale
- domainpolicy allows dynamic peering
- code for infrastructure ENUM and domainpolicy in openser CVS since 2006-11-02

enum.at
net.communications

Klaus Darilion
phone: +43 1 5056416 36, fax: +43 1 5056416 39
klaus.darilion@enum.at, www.enum.at

References

- Speermint Working Group
  http://ietf.org/html.charters/speermint-charter.html
- draft-lendl-domain-policy-ddds
  http://www.ietf.org/internet-drafts/draft-lendl-domain-policy-ddds-02.txt
- draft-lendl-speermint-federations
  http://www.ietf.org/internet-drafts/draft-lendl-speermint-federations-03.txt
- draft-lendl-speermint-technical-policy
  http://www.ietf.org/internet-drafts/draft-lendl-speermint-technical-policy-00.txt
- draft-haberler-carrier-enum
  http://www.ietf.org/internet-drafts/draft-haberler-carrier-enum-03.txt
- domainpolicy documentation
  - module README
  - Tutorials at enum.at homepage: http://www.enum.at/index.php?id=dokumente

Appendix

enum.at
net.communications

Static Peering with domainpolicy module

- type = "dom" (domain)

```
if (uri =~ "^sip:*.@itspA.foo.bar") {
   sethostport("peer.itspA.foo.bar;transport=tcp");
} else if (uri =~ "^sip:*.@itspB") {
   # do nothing, current R-URI is fine
} else {
   sl_send_reply("403","untrusted peer");
   ...
```

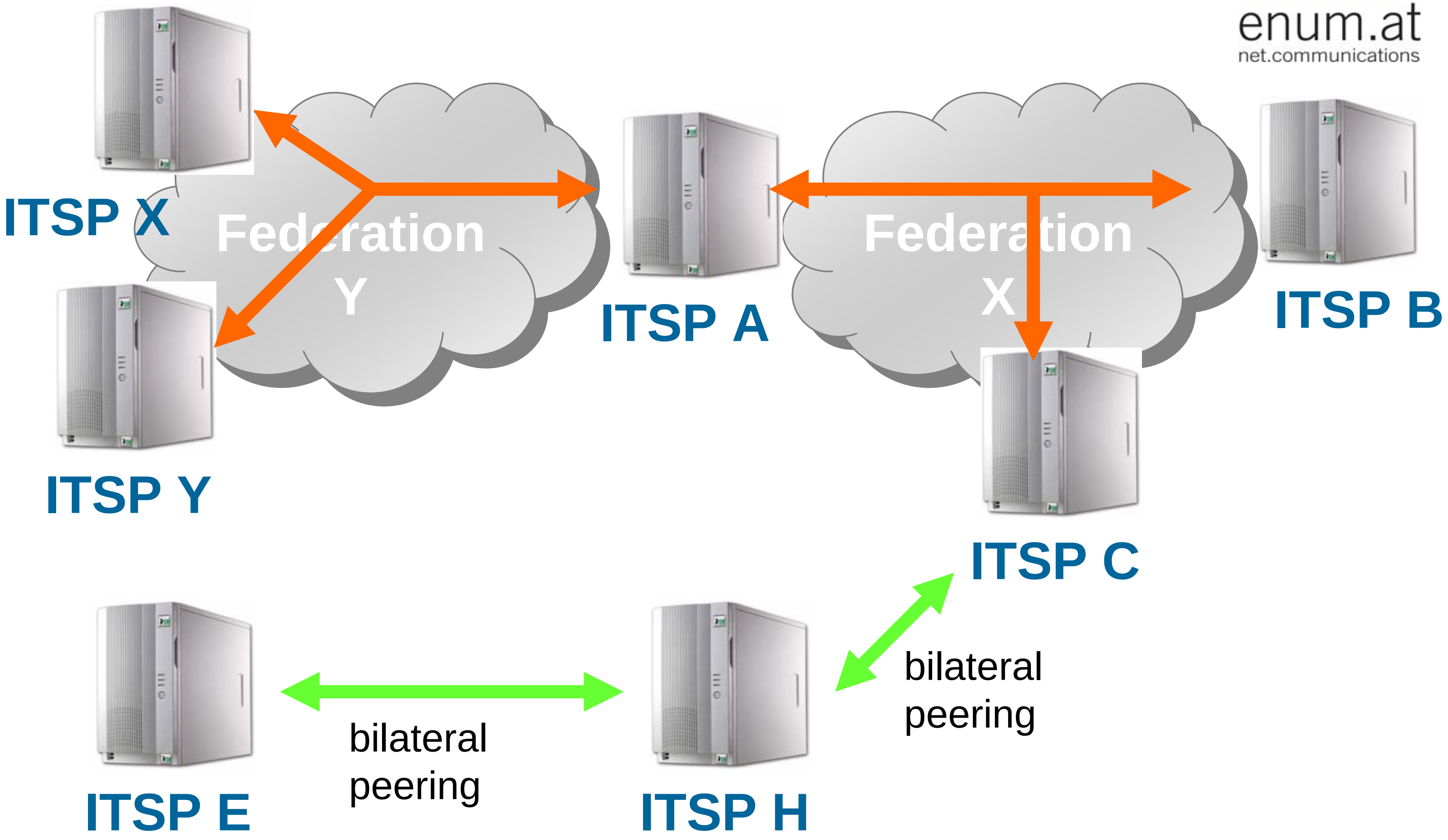
➔

```
+----+--------------+------+---------------------+--------------------+
| id | rule         | type | att (avp name)      | val                |
+----+--------------+------+---------------------+--------------------|
| 1  | itspa.foo.bar| dom  | s:domainreplacement | peer.itspA.foo.bar |
| 2  | itspa.foo.bar| dom  | s:transportoverride | tcp                |
+----+--------------+------+---------------------+--------------------|
```

practical peering tips

- no NAT traversal for other peers
- P-Asserted-Identity: use tel URI for phone numbers, not SIP URI
- use dedicated peering proxy
- do not use UDP
- allow ICMP or SIP error messages

Federations


ITSP X
Federation Y
ITSP A
Federation X
ITSP B
ITSP Y
ITSP C
bilateral peering
ITSP E
bilateral peering
ITSP H