DDoS made easy –

IP reflection attacks for fun and profit

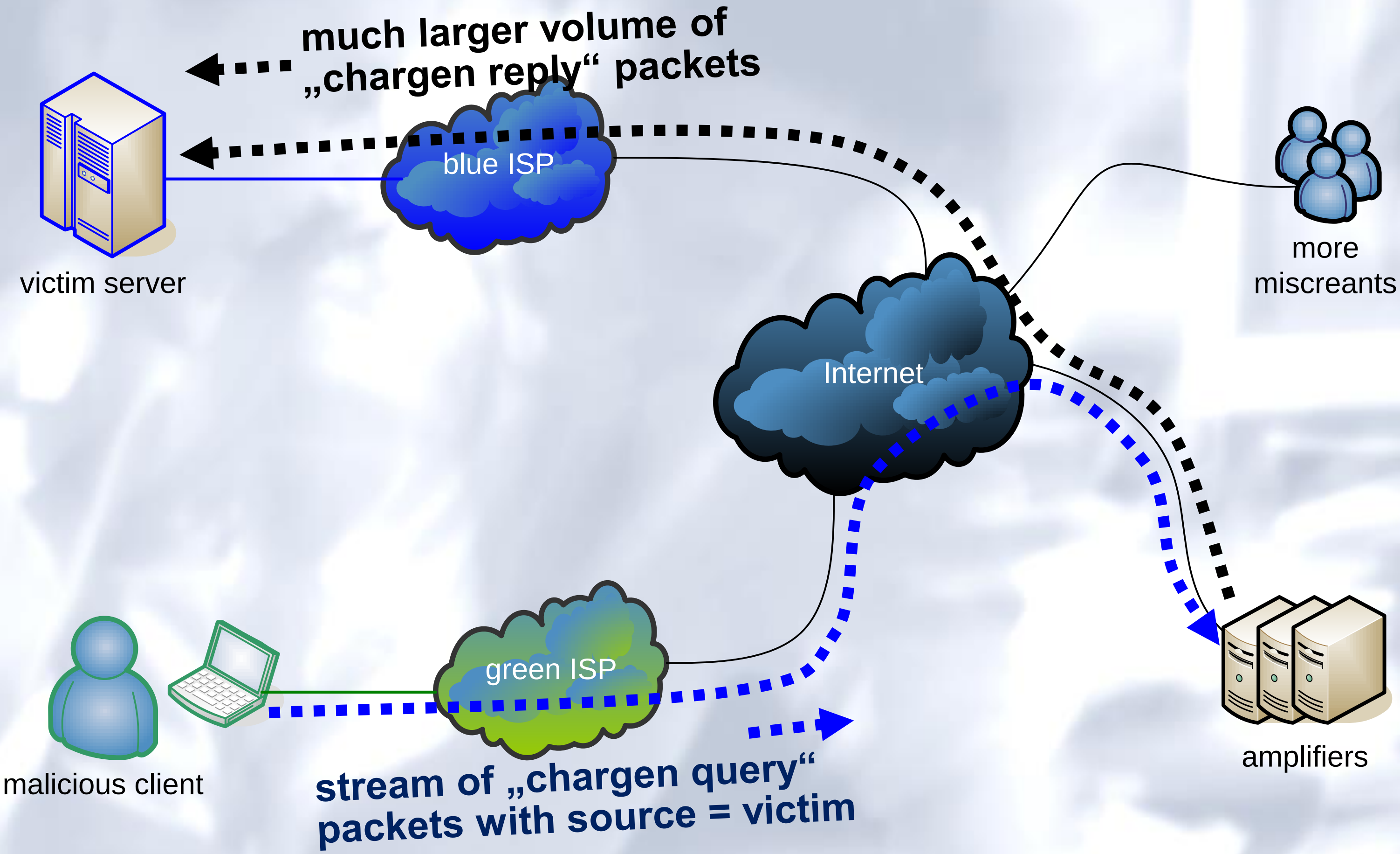
Gert Döring, SpaceNet AG, München

DECIX/ECO security event, 04.12.14, Frankfurt

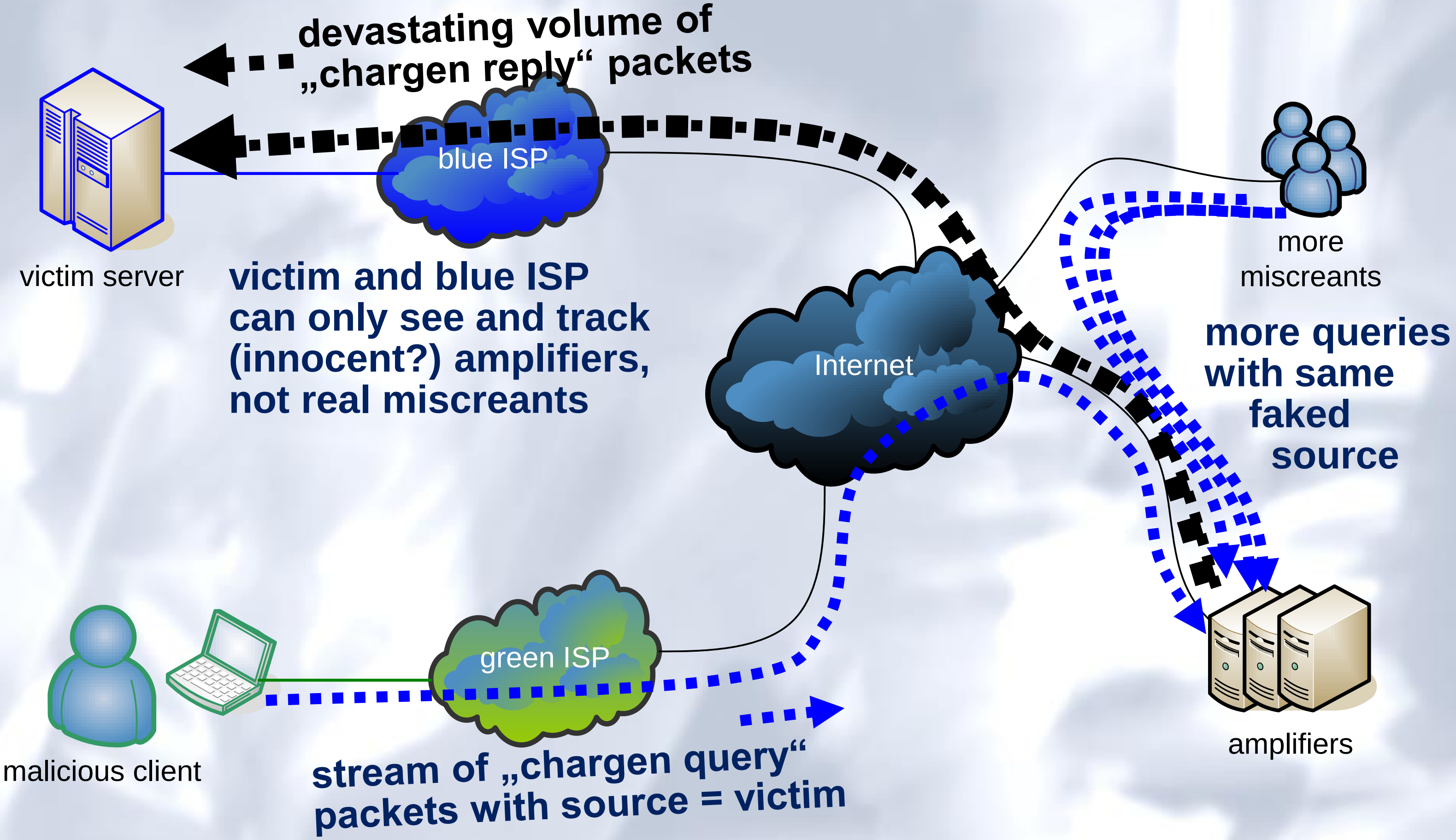
Agenda

- what are IP reflection attacks?
- why are they so effective (= fun to use)?
- countermeasures:
    - abandoning all reflection-prone IP protocols
    - uRPF at the edge
    - bgpq-generated packet filters for BGP customers
    - egress filters, if unavoidable
- discussion

reflective DoS explained



reflective DoS explained



devastating volume of „chargen reply“ packets

victim and blue ISP can only see and track (innocent?) amplifiers, not real miscreants

more queries with same faked source

stream of „chargen query“ packets with source = victim

So, let‘s shut down these amplifiers!

- Nobody needs open NetBios, Echo or Chargen ports facing the Internet! Banish the Evil Protocols!
- Nobody needs open NTP servers on the Internet anyway
- Nobody needs open DNS recursors (recursive DNS servers) on the Internet anyway
- Nobody needs authoritative DNS servers with DNSSEC.
  - Wait, what? Uh, ok, let´s mandate rate limiting!!
- This TCP thing is really bad, can be used to amplify small-packet rate – 1x SYN → 6-10x SYN/ACK.
  - So, let´s rework the whole TCP layer! … wait, what?

Not A Good Idea

what is the real problem here?

- real problem is not „servers that answer queries“ but „source IP spoofing“:
- sending IPv4 or IPv6 packets with a source address that the sender has no authority over, to other parties outside the sender‘s authority
  - „not your source“ and
  - „not your destination host“
- could be „in the LAN“, to attack hosts in the same LAN segment (hiding / stealing identity)
- focus here: WAN, aka „the Internet“

what is „the IP spoofing problem“?

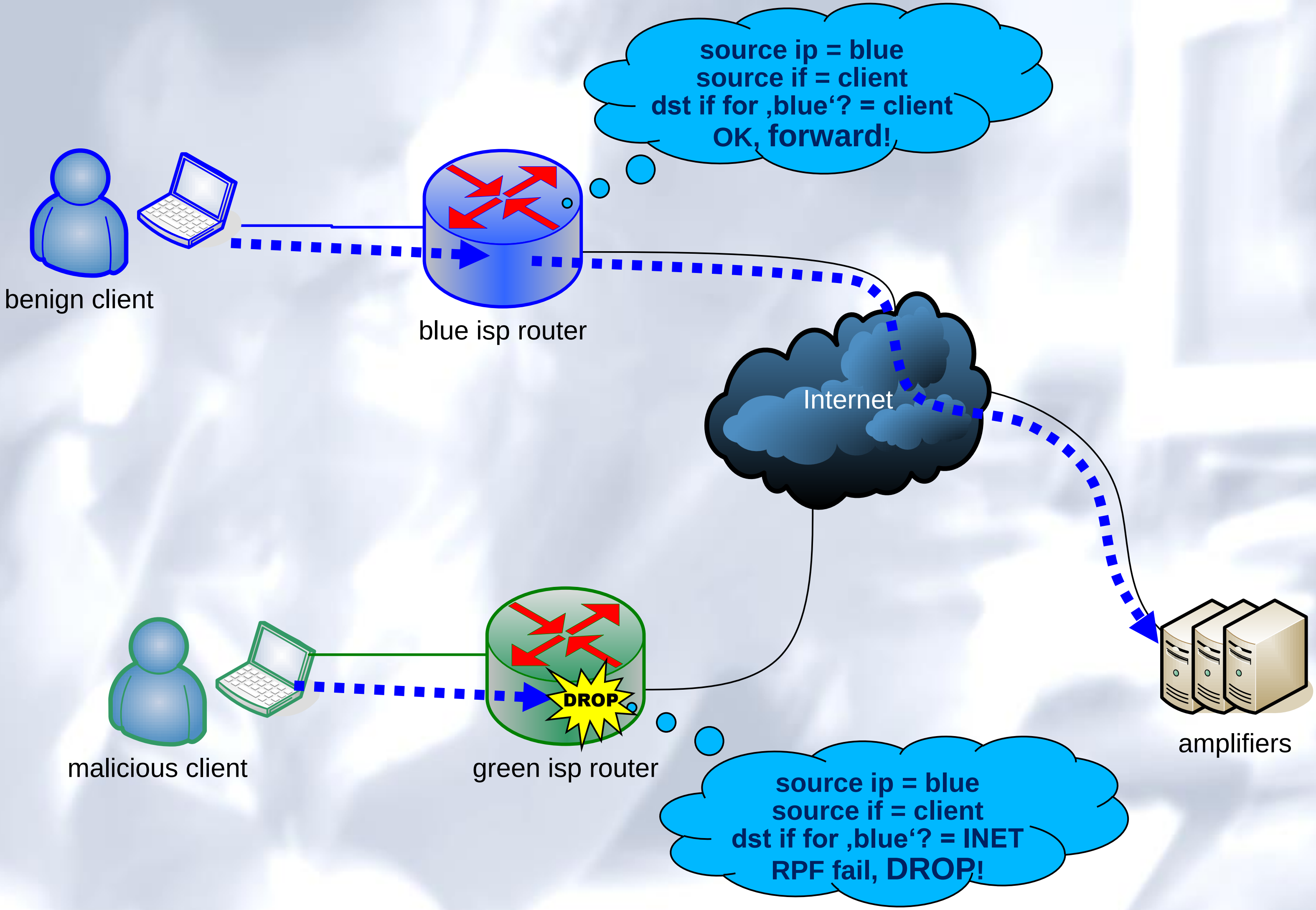
- For two-way IP communication, both parties need to send packets with „their own“ source address, that is, an address that is routed back to that party
- Under normal circumstances, there is no need to ever send packets from a source address that would not be routed back to you
- But it can be nicely **used for attacks** on others:
  - reflective DoS attacks
  - TCP stream interference (data injection, resets)
  - gaining unauthorized access (the 15+ year old rsh attack)

new approach, fix problem at source: uRPF

- „unicast reverse path filtering“, uRPF
- teach routers to *verify source address* on ingress
    - take incoming packet‘s source address
    - do a route lookup for the source address
    - if the result of the route lookup („where would a packet with that address be sent to?“) does not point to the interface where it came in: **drop packet**.
    - if verification succeeds, forward normally
- described 14 years ago in RFC2827 / BCP38
- implemented by most vendors

- (nitpick: this is „strict mode“ uRPF. „loose mode“ uRPF = „any route is OK“)

uRPF visualized



source ip = blue
source if = client
dst if for ‚blue'? = client
OK, forward!

source ip = blue
source if = client
dst if for ‚blue'? = INET
RPF fail, DROP!

uRPF examples

- Cisco:

```
interface GigEth 3/8
  ip verify unicast reverse
```

- Juniper:

```
edit interface ge-0/3/0 unit 0 family inet
  set rpf-check;
```

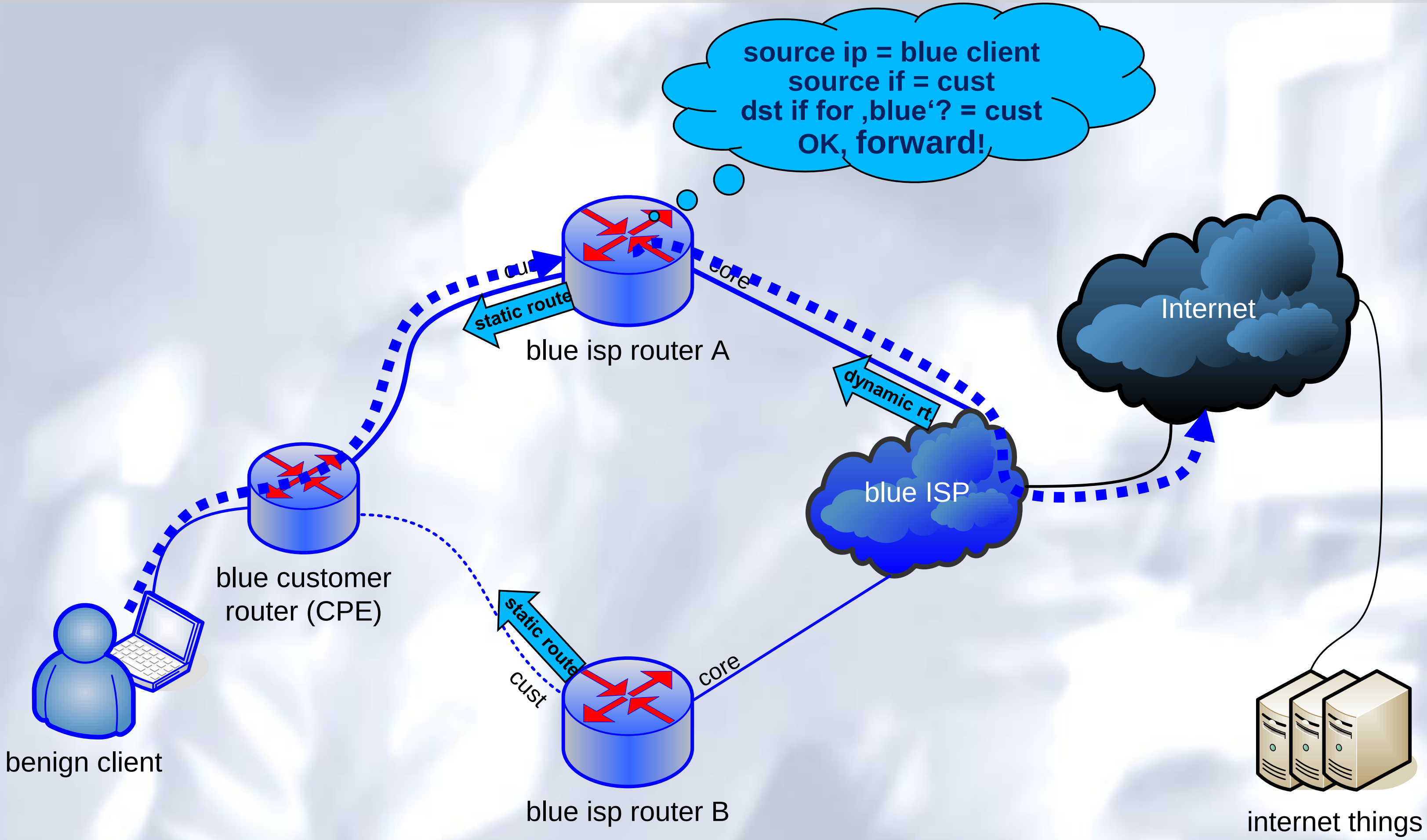
- Bintec:

```
[WAN][EDIT][IP][Advanced]: Advanced Settings
    Back Route Verify       on
```

so, why is uRPF not universally deployed?

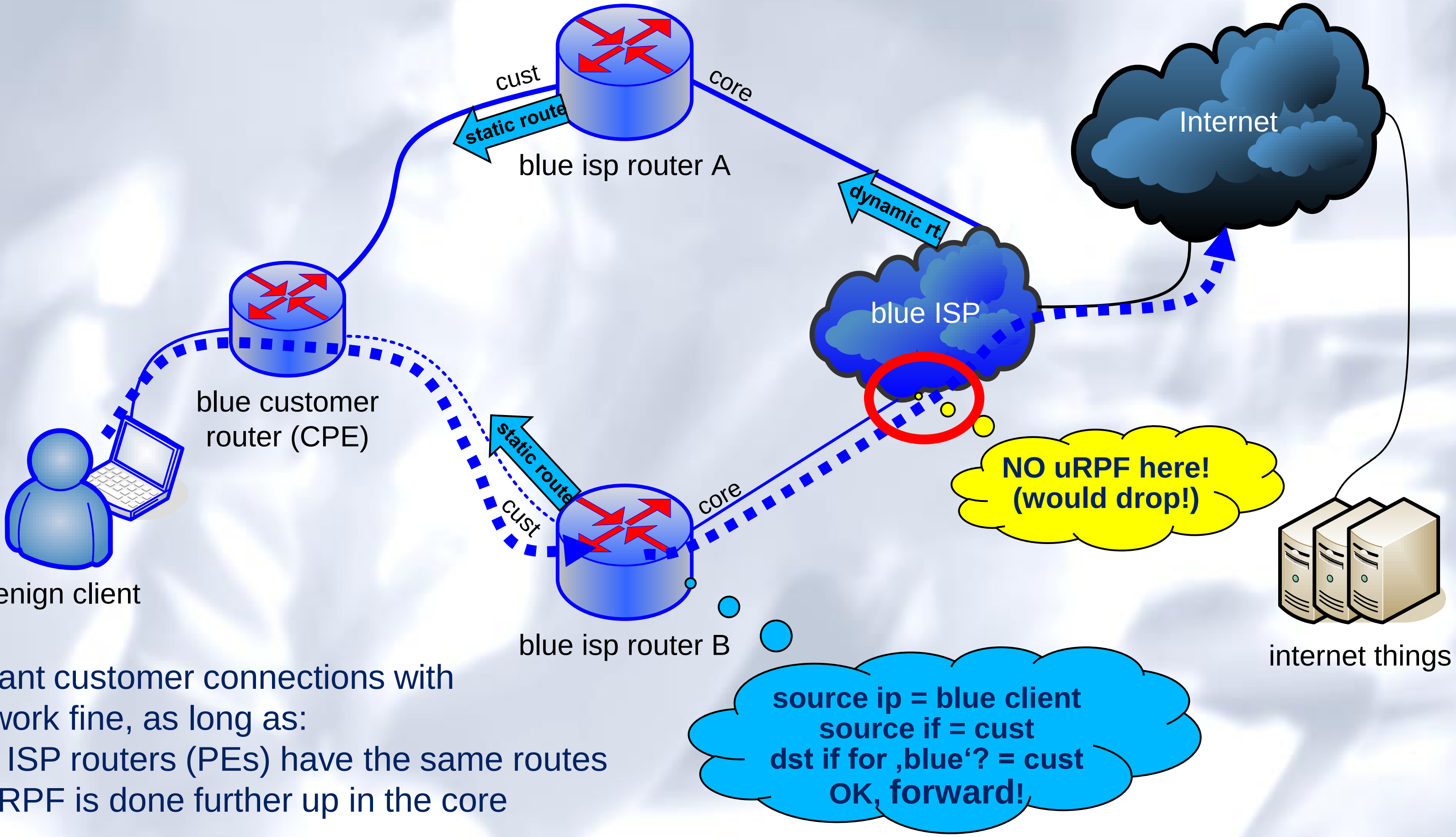
- It perfectly solves the spoofing problem…
- … for everyone *else*: you filter, nobody else is attacked by your customers – you pay, everybody else benefits. So the commercial incentive is negative.
  - peer pressure could help here...
- plus, there are corner cases where it indeed gets in the way, causing issues for legitimate traffic – quite obviously for asymmetric traffic
- plus, there can be vendor (hardware) limitations

uRPF problem spot 1: redundant links



source ip = blue client
source if = cust
dst if for ‚blue‘? = cust
OK, forward!

uRPF problem spot 1: redundant links



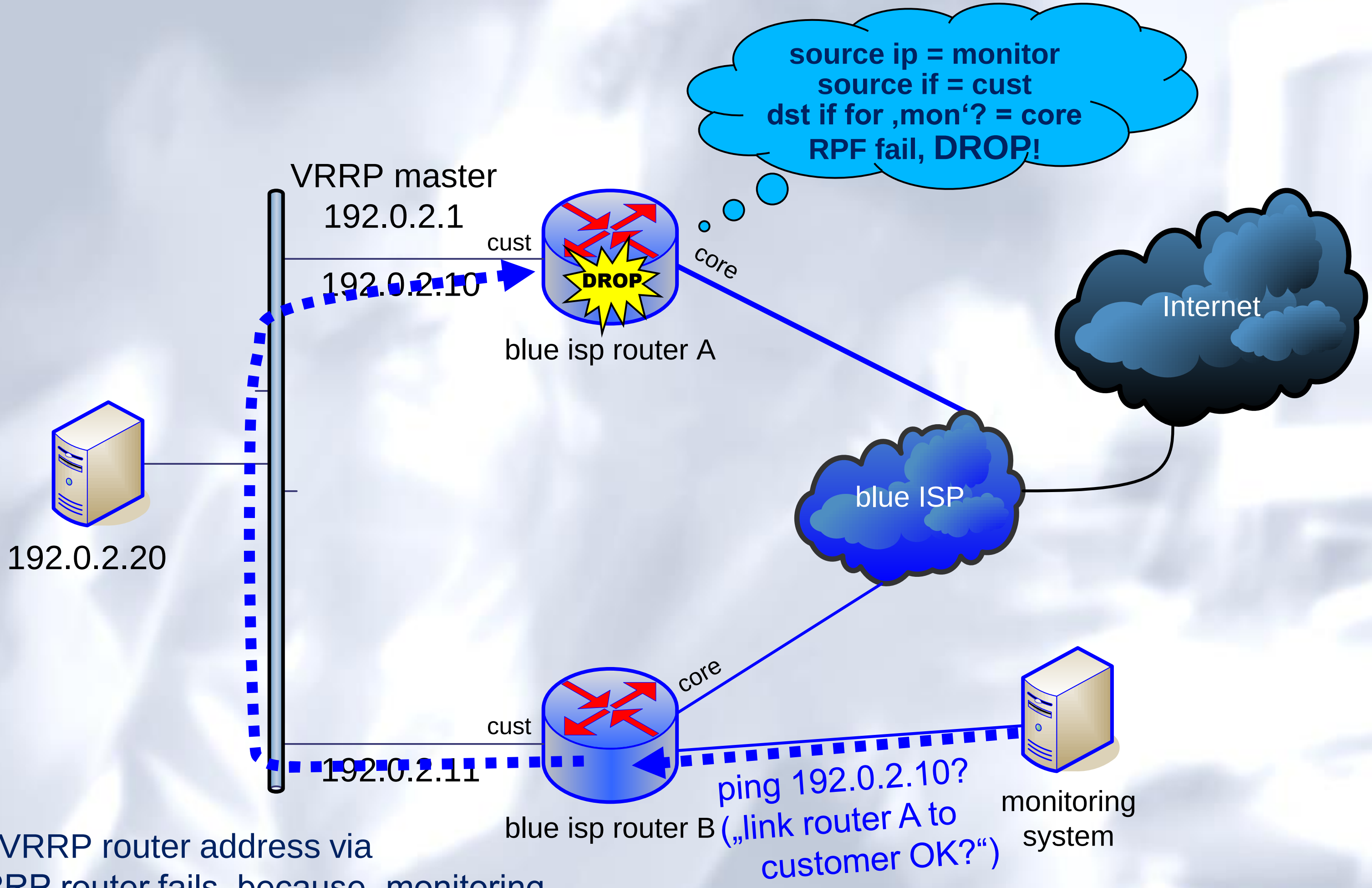
NO uRPF here!
(would drop!)

source ip = blue client
source if = cust
dst if for ‚blue'? = cust
OK, forward!

redundant customer connections with uRPF work fine, as long as:

- both ISP routers (PEs) have the same routes
- no uRPF is done further up in the core

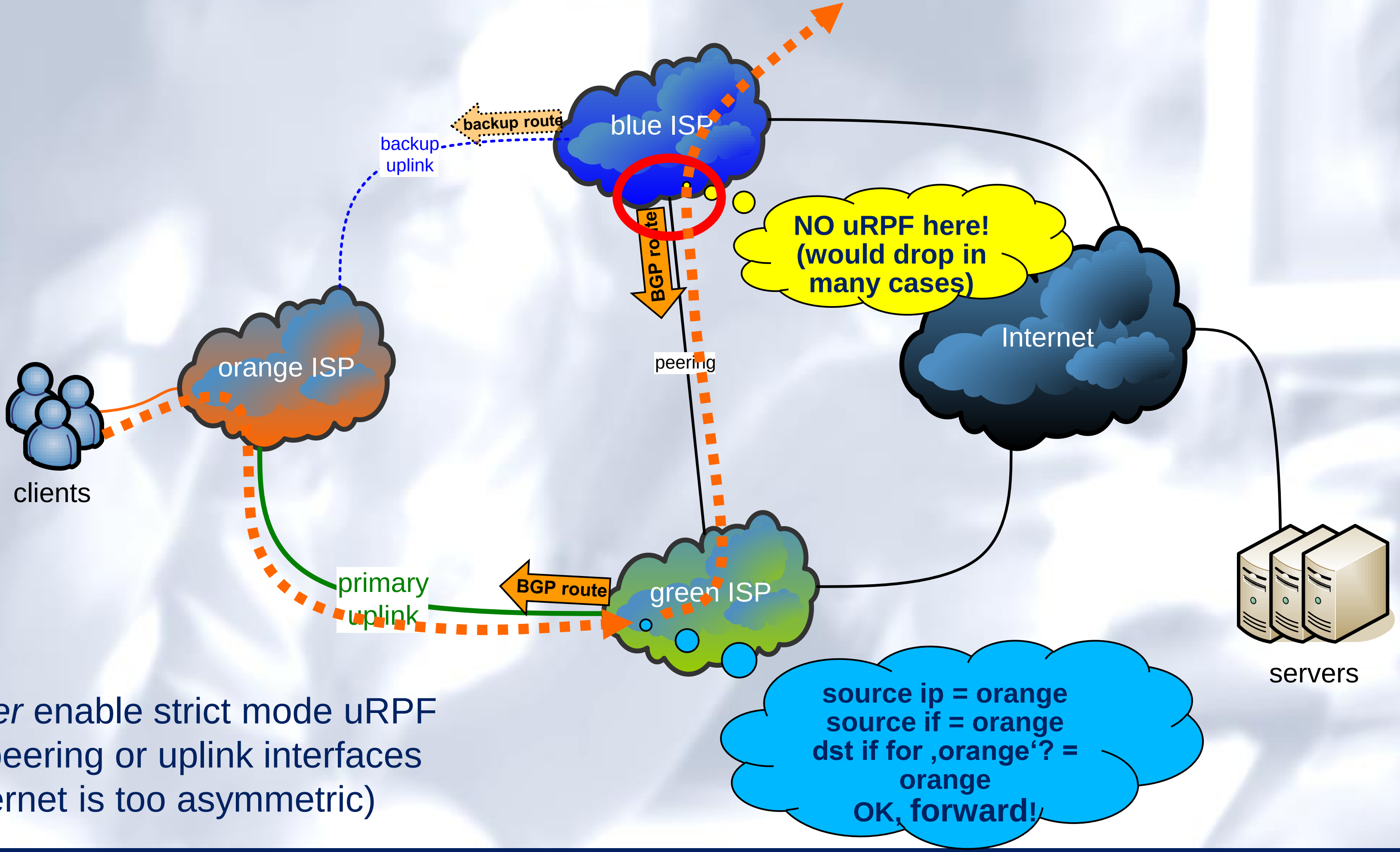
uRPF problem spot 2: dual-routers (vrrp)



ping 192.0.2.10?
(„link router A to customer OK?“)

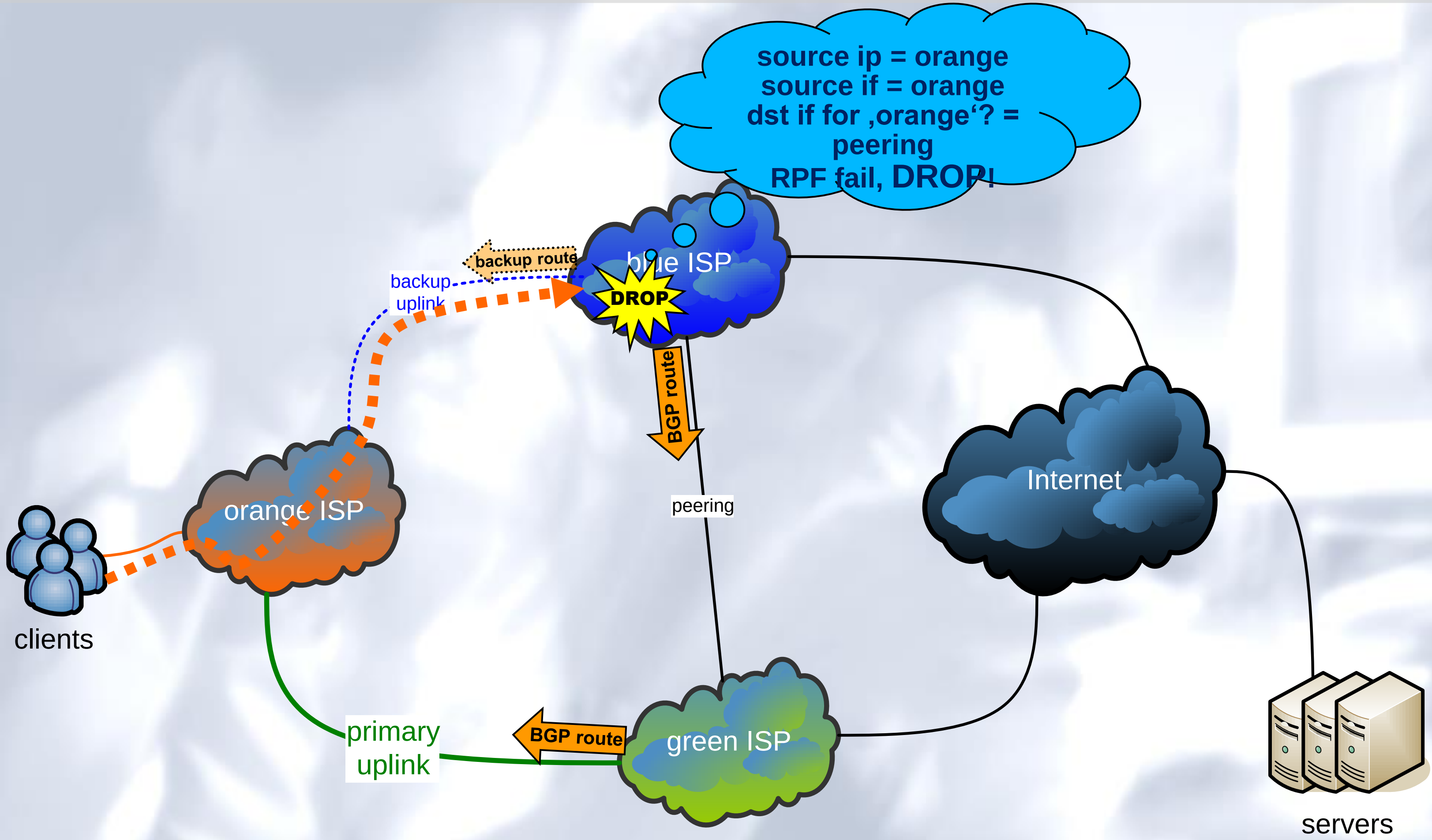
pinging VRRP router address via peer VRRP router fails, because „monitoring system“ IP is known to be „outside“.
Workaround: exception ACL / check differently

uRPF problem spot 3: BGP customers



never enable strict mode uRPF on peering or uplink interfaces (Internet is too asymmetric)

uRPF problem spot 3: BGP customers



uRPF on BGP customer links **will cause problems** in asymmetric routing scenario (which is quite common) → use ACLs instead

uRPF problem spot 3: BGP customers

- commercial fix:
  - require by contract that the customer deploys uRPF, and monitor incoming traffic for violations (netflow vs. BGP)
  - if violations detected, apply pain by invoice

- technical fix:
  - instead of deploying „automatic uRPF“, deploy source address verification by ACL-filtering ingress packets
  - generate ACL by same toolset that generates downstream BGP filters from RIPE DB (etc.)
  - „if he´s not permitted to send BGP announcements for a prefix, he shouldn‘t source packets from there either“

uRPF problem spot 3: BGP customers

- build prefix list for BGP:

```
$ bgpq -P -l in-prefix-8481 AS8481
 no ip prefix-list in-prefix-8481
 ip prefix-list in-prefix-8481 permit 82.118.32.0/19
 ip prefix-list in-prefix-8481 permit 195.24.96.0/19
```

- build ACL for source address verification (s.a.v.):

```
$ bgpq -A -l in-sav-8481 -i AS8481
 no ip access-list extended in-sav-8481
 ip access-list extended in-sav-8481
   permit ip 82.118.32.0 0.0.31.255 any
   permit ip 195.24.96.0 0.0.31.255 any
   deny ip any any
```

- apply to BGP peer and ingress interface(s)
- update regularily, and let your customer know(!)

uRPF problem spot 4: dumb routers

- in Cisco 6500/Sup2, enabling uRPF reduces FIB table size by 1/2
- Cisco 6500/Sup720 cannot do uRPF for IPv6 in Hardware (= Software forwarding, sloowww)
- check what *your* vendor can and can not do
- if uRPF is not workable, find alternatives, like:
  - ingress ACLs on customer interfaces (automatic generation from your provisioning system / radius?)
  - ingress ACLs at aggregation points
  - egress ACLs at peering/upstream links
    („last resort" only, needs updating if customer net blocks change, and will not tell you *which* customer sent spoofed traffic)

Summary

- everyone needs to apply source-address verification on their networks, to ensure long-term sustainability of the Internet
  - best applied at customer ingress ports
  - but can be applied at aggregation or egress as well, if ingress cannot be done
  - S.A.V.E. = Source Address Verification Everywhere
- read: RFC2827 and http://bcp38.info/
- http://www.cymru.com/Documents/secure-ios-template.html
- questions or comments: gert@space.net

Discussion / S.A.V.E!

- Why are *you* not deploying uRPF (or some other way of source address verification)?
- Are you deploying uRPF for IPv6?

- How can we motivate „all the others“ to deploy source address verification?
- If we fail, how can we fix the Internet?