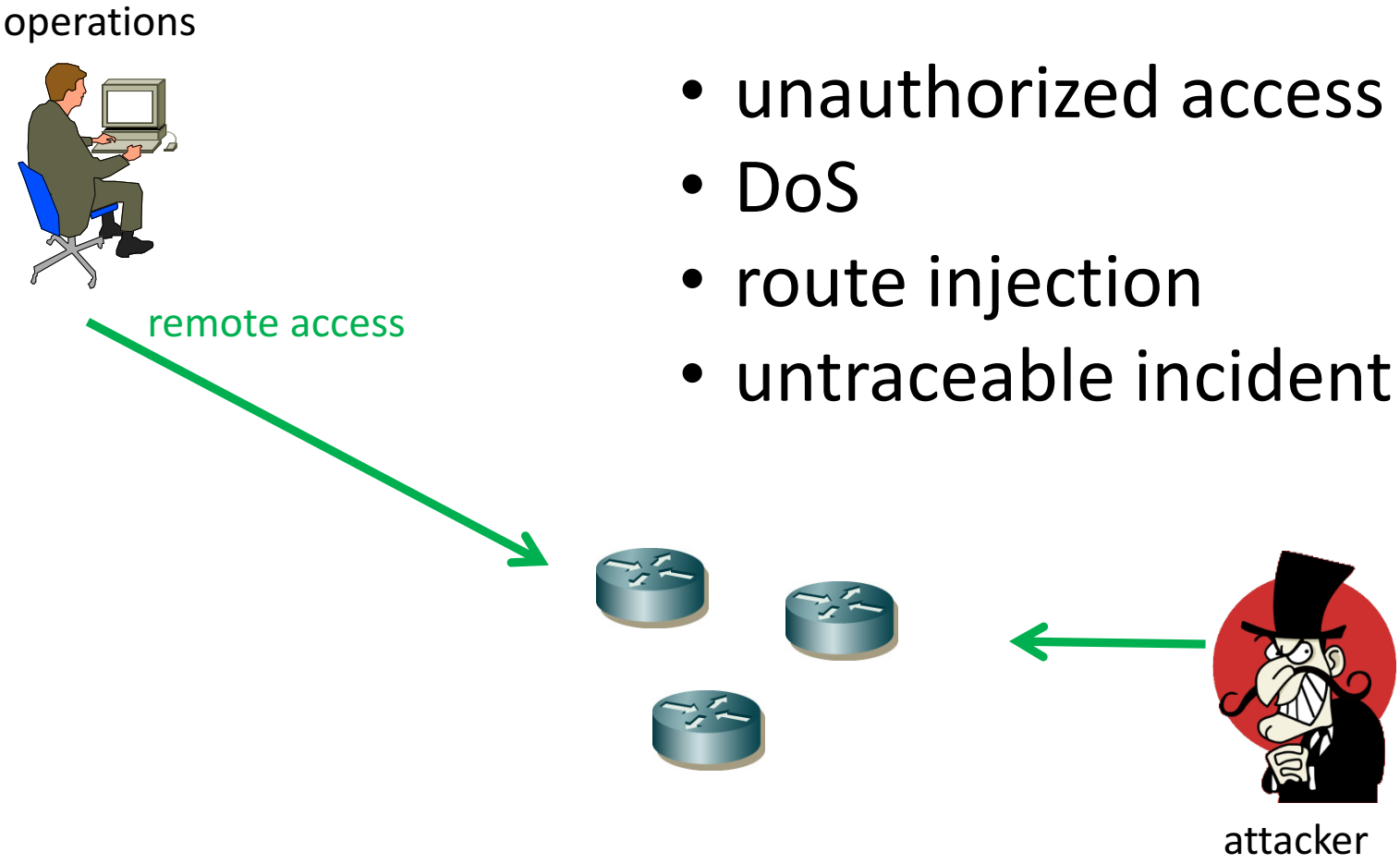
Securing network infrastructure

Matsuzaki ‘maz’ Yoshinobu
<maz@iij.ad.jp>

Our Goals

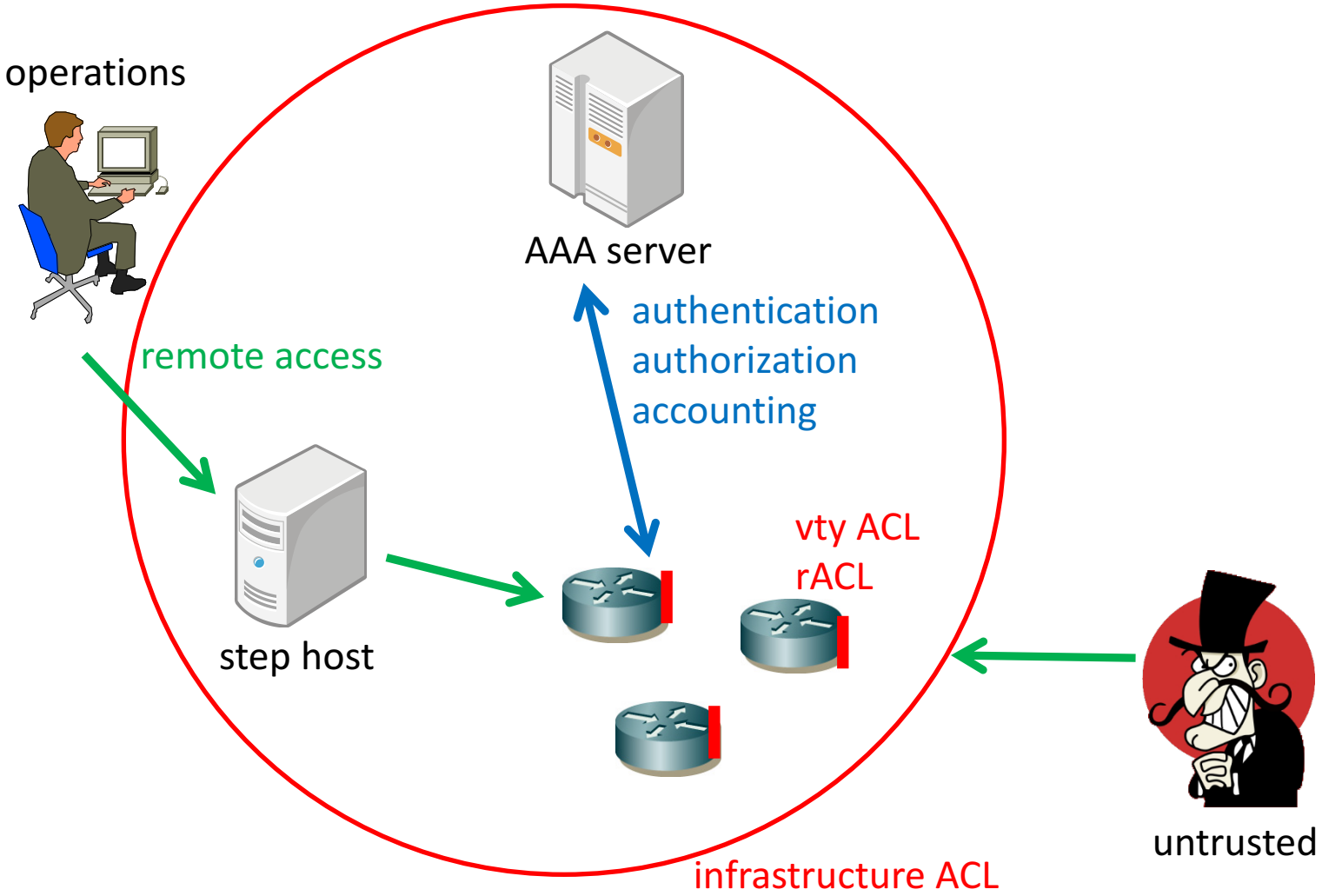
- Ensuring Network Availability
- Controlling Routing Policy
- Protecting Information
- Preventing Misuse
- Mitigating Attacks
- Responding to Incidents
- etc.

Risks



- unauthorized access
- DoS
- route injection
- untraceable incident

protecting devices



AAA server and remote access

- Authentication, Authorization, Accounting
  - tacacs, radius
- each operators has own login account
  - You can set privileges per tasks of the operator
- logging at AAA servers
  - where (device)
  - who (login account)
  - what (command)

Remote Access to Devices

- in-band access
  - vty, snmp, ntp, etc...
  - IP reachability is required
  - useful for daily operations
- out-of-band access
  - serial console
  - workable without IP reachability
  - useful for restoration

Access Control for in-band access

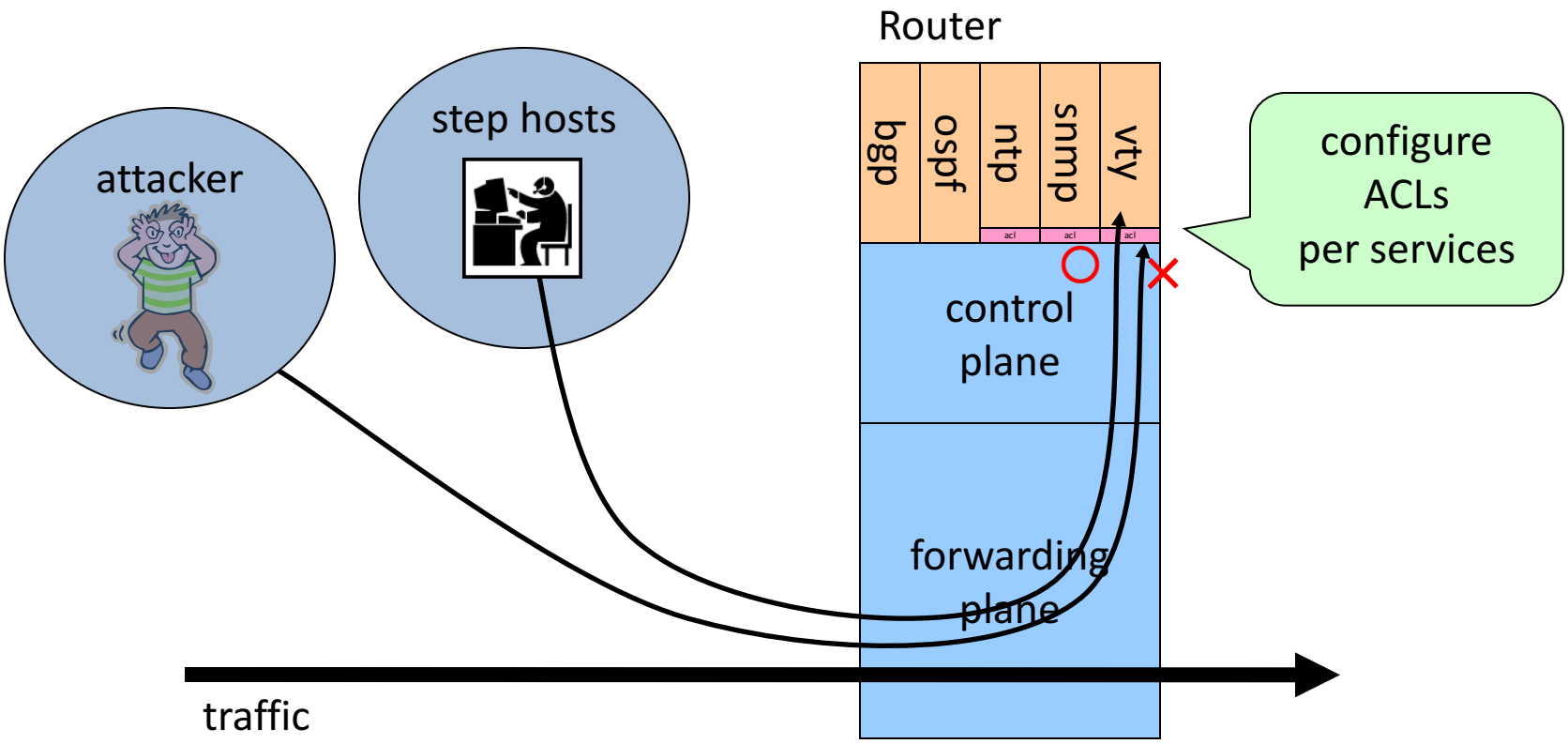
- operations need to access remote devices
  - to manage the devices
- packet filtering on vty, snmp and etc
  - to protect devices from unauthorized access
  - allow access from trusted network only
    - source IP address based filtering

step hosts

- are placed on a trusted network
- useful to enforce more restricted control
- each operations has own login account

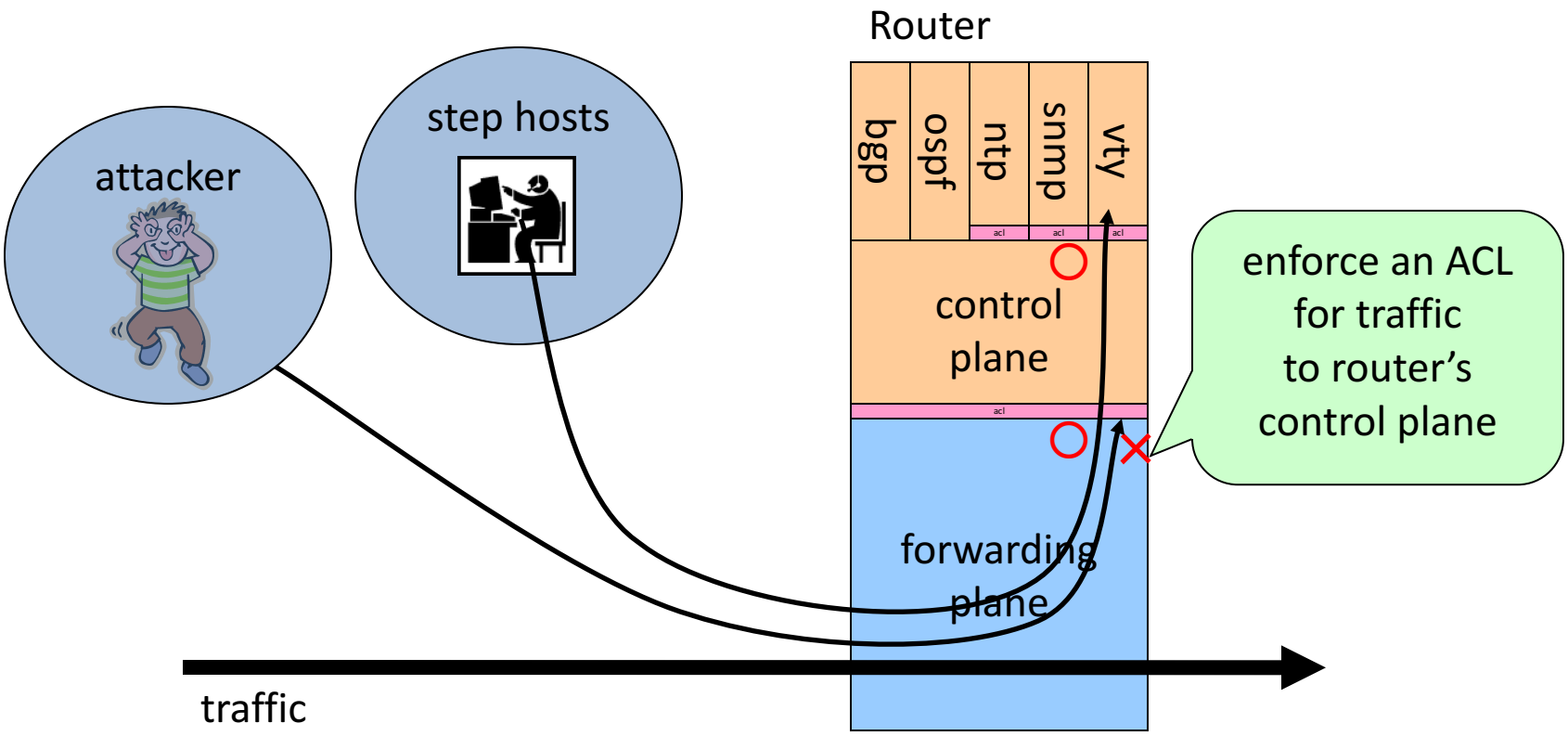
- logging on step hosts
  - typescript of a VTY session
  - login/logout

access control per services



Received/Router ACL (rACL)

access control against control plane

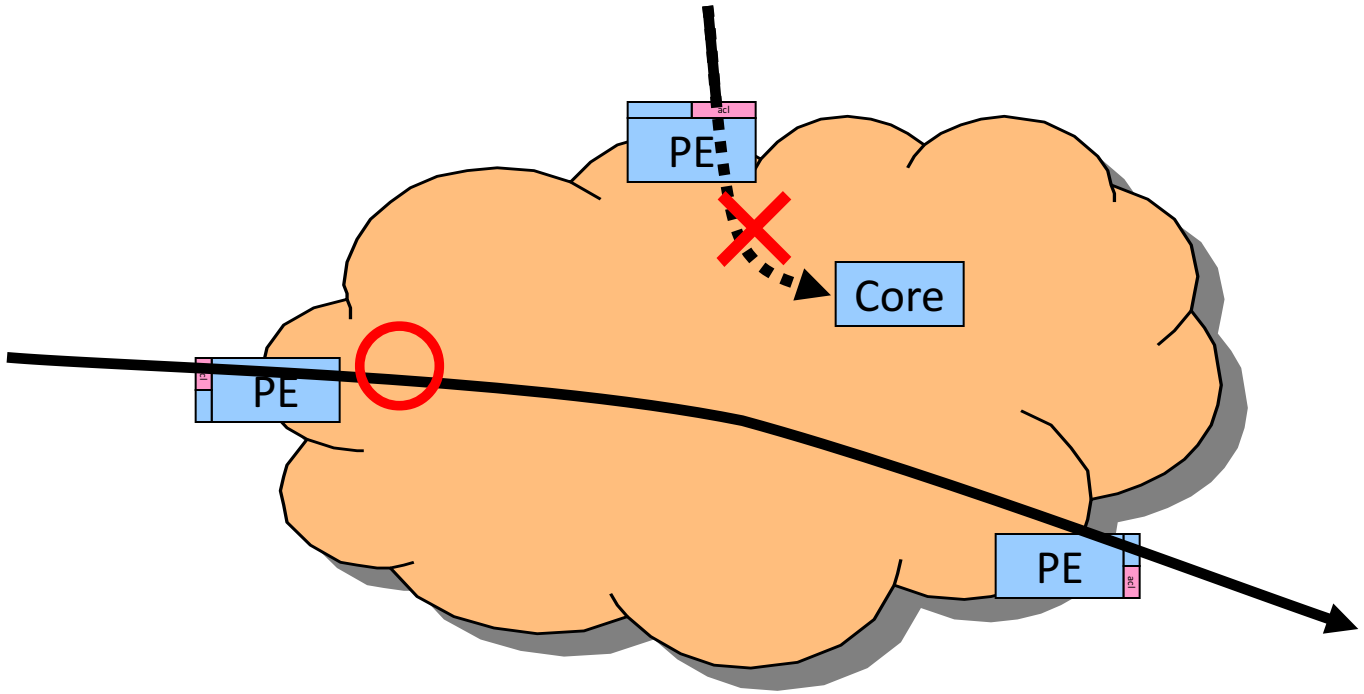


infrastructure ACL

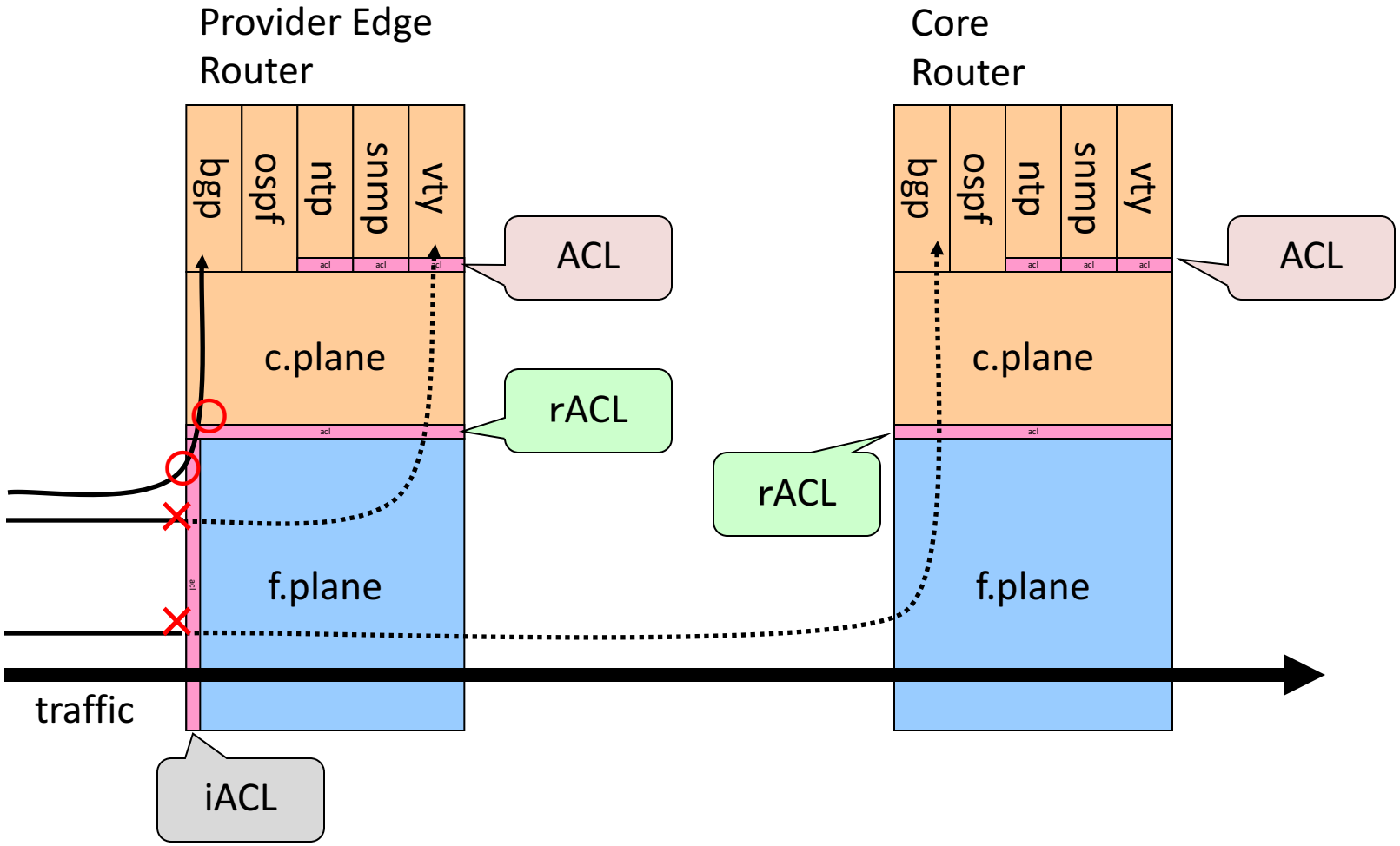
- to protect our management traffic
  - not too much
  - ping, traceroute to our devices should be workable
- deny packets from INFRA to INFRA on edge
  - INFRA: routers, step hosts and so on
    - these ip range should be stayed inside

Infrastructure ACL (iACL)

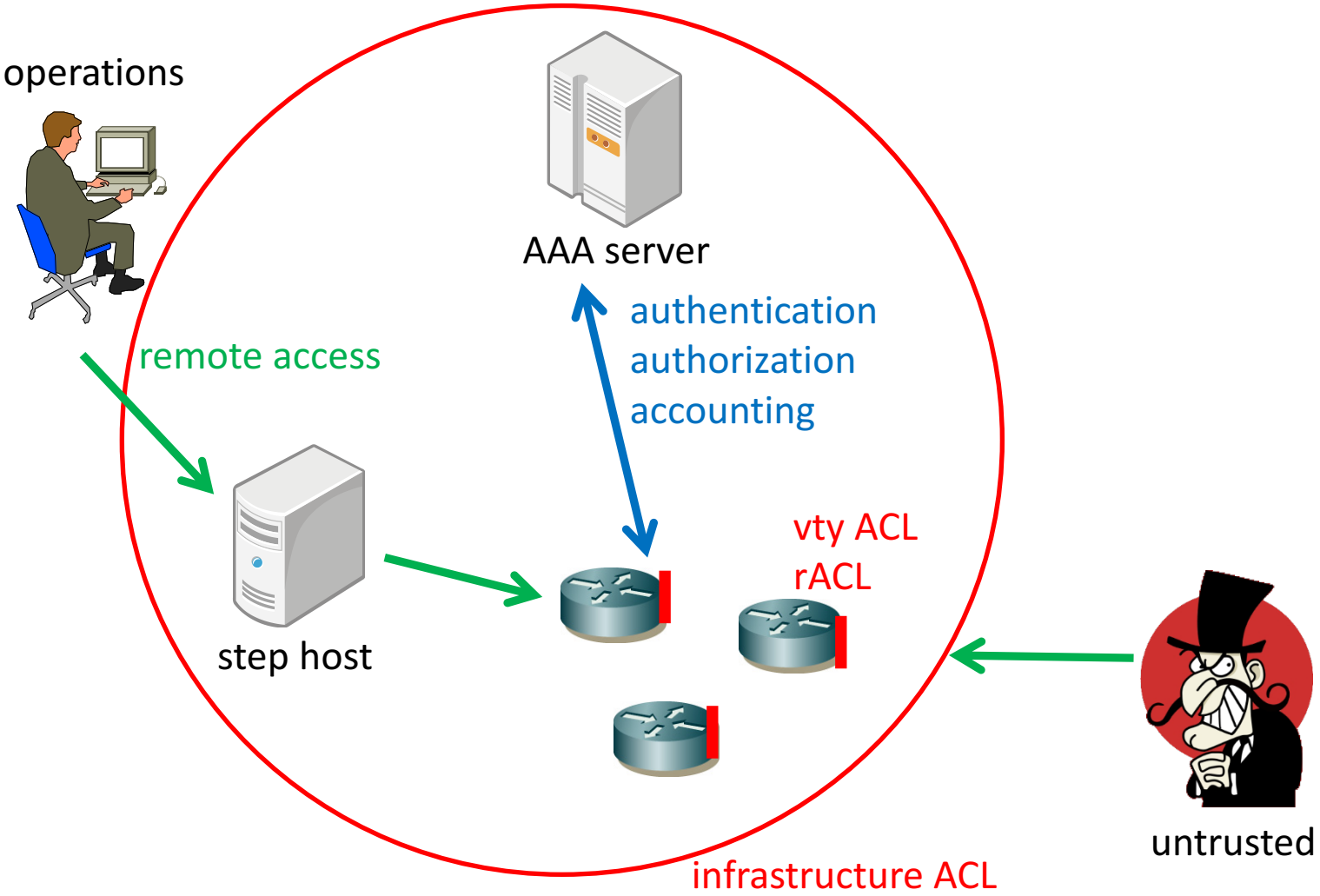
- enforce a policy on the network edge



multiple ACLs to protect Devices



protecting devices



config audit

- configuration files are periodically gathered
  - by in-house automated tool
- sanity check
  - filtering rules
  - routing configuration
  - and so on

monitoring

- what's happened in the past
- syslog
  - to record messages from devices/softwares
- snmp
  - to monitor resources
- netflow
  - to monitor packet flows

syslog messages

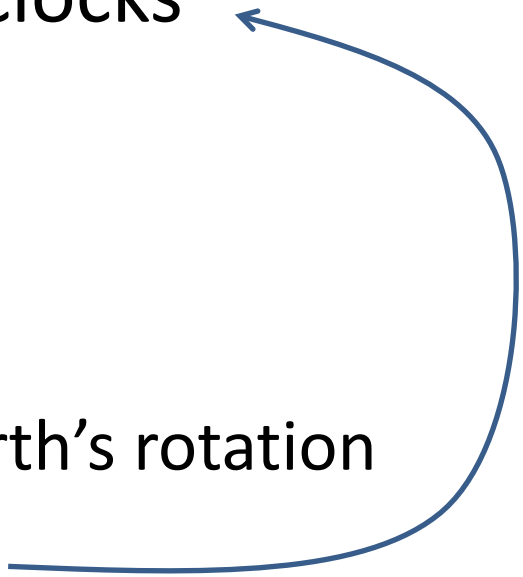
- Nov 9 15:19:14.390 UTC: config[65775]: %MGBL-SYS-5-CONFIG_I : Configured from console by maz on vty0 (2001:db8:120:100:e1dd:97f3:fd98:a51f)
- Nov 12 13:53:38 maz sudo: maz : user NOT in sudoers ; TTY=pts/3 ; PWD=/home/maz ; USER=root ; COMMAND=/bin/bash

synced timestamp

- makes log messages useful
  - to compare incidents among devices
  - to compare time-related events
- Use ntp to sync clocks
  - choose a proper clock source
    - national ntp server
    - stable clocks
      - ATOM, GPS

clock = oscillation + counter

- TAI = weighted average of atom clocks
  - TAI: International Atomic Time
- UTC = TAI + leap seconds
  - UTC: Coordinated Universal Time
  - leap seconds: to adjust clock to Earth's rotation
- atom clocks are adjusted to TAI
- localtime = UTC + timezone (+ summer time)

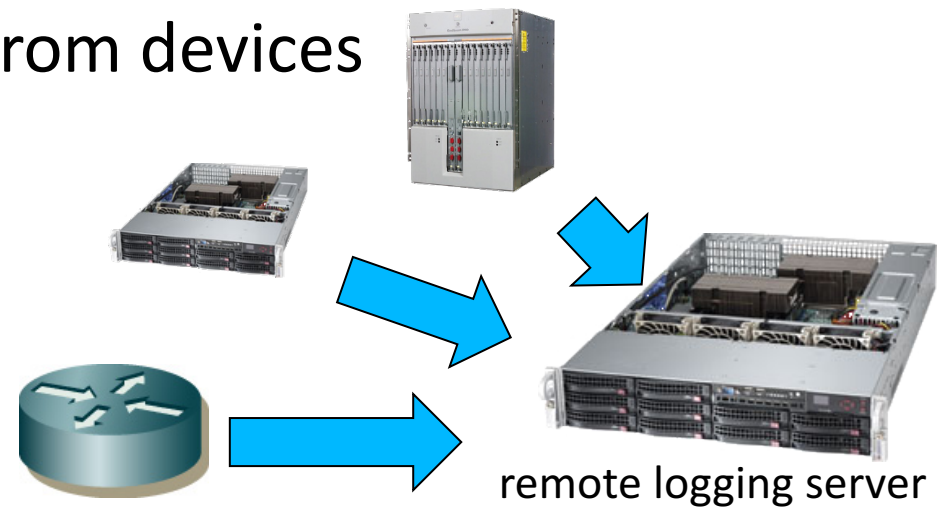


leap second

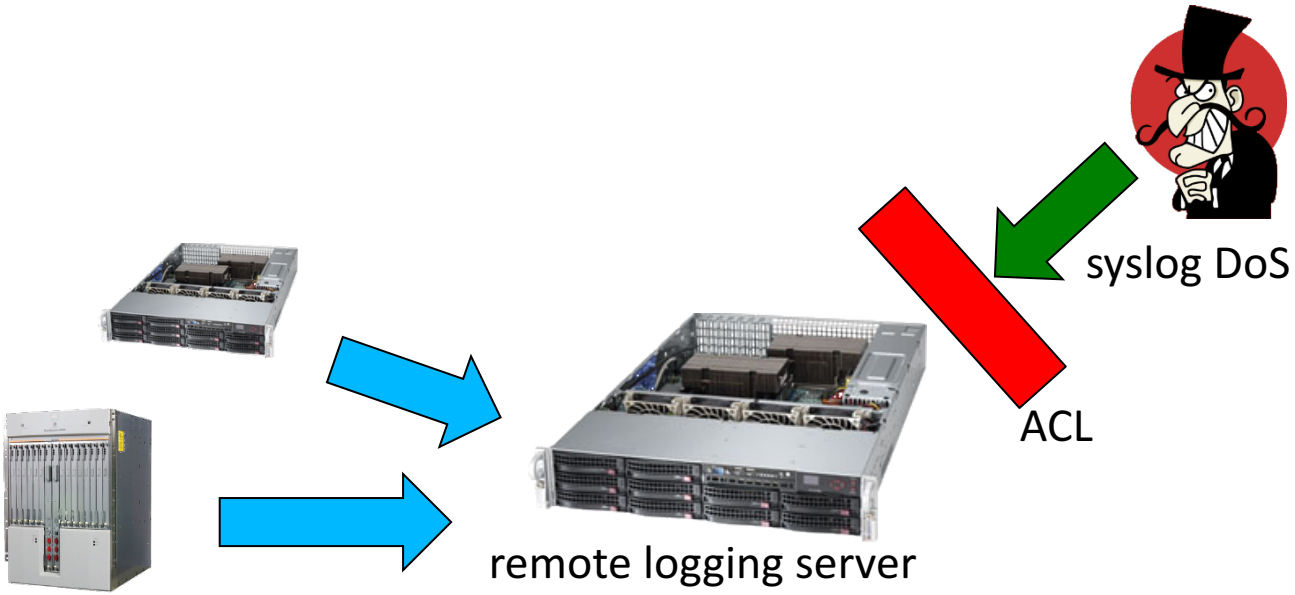
- The next leap second will be introduced on 30 June 2015 23:59:60 UTC
- make sure your applications works as usual even the leap second introduced
  - https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/commit/?id=6b43ae8a619d17c4935c3320d2ef9e92bdeed05d

remote logging

- log messages could be modified/deleted
  - if the system is compromised
  - limited memory buffered log messages
- remote logging server
  - receive log messages from devices
  - syslog-ng
  - enough storage there



protecting syslog

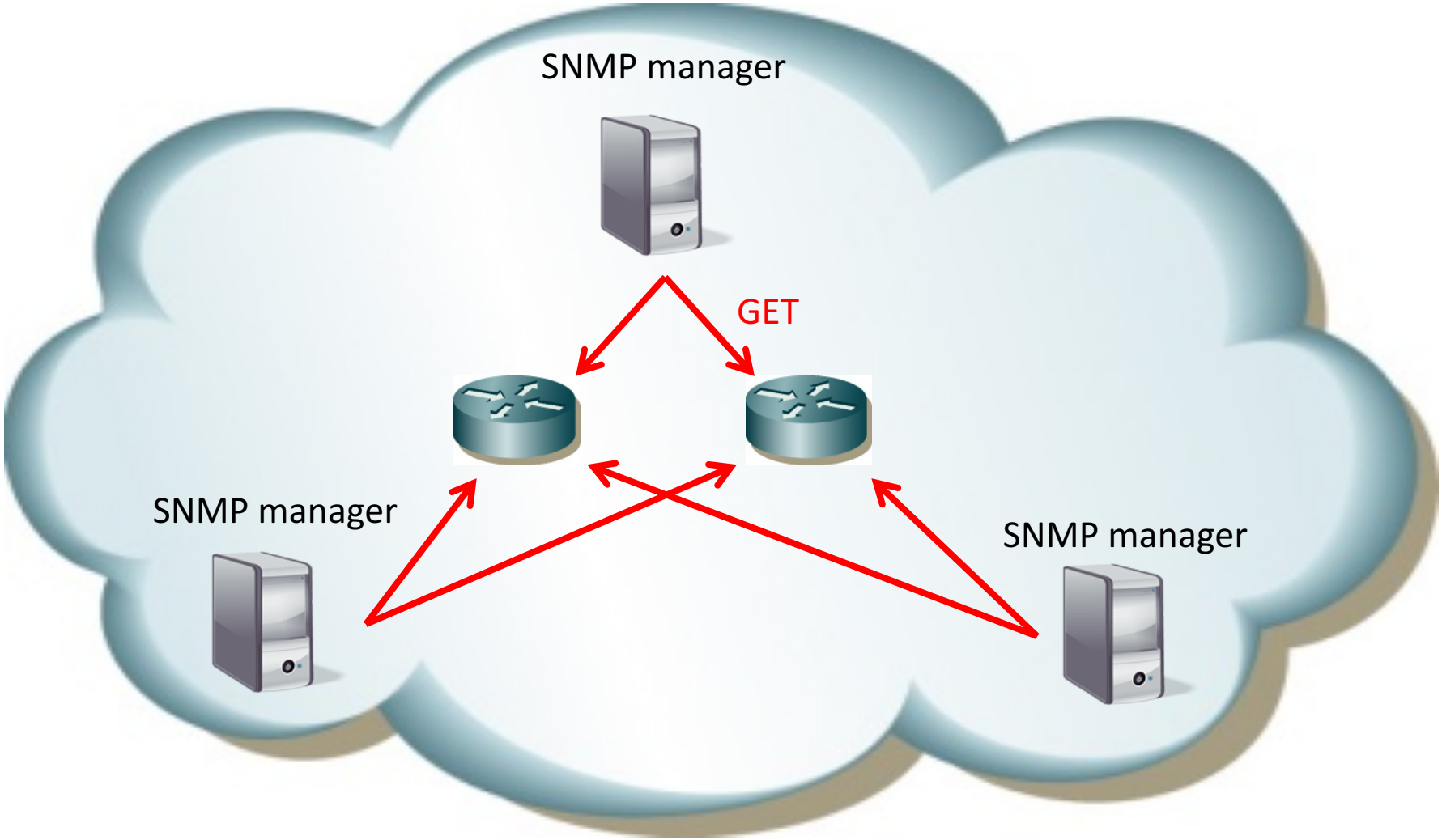


ensure the correctness of log entries

snmp

- can read/write information and send a trap
  - use version 3, and set password
  - prevent ‘write’ function, or just disable it on agents
  - put ACL to prevent unauthorized access
- require a little disk space on snmp manager
  - useful to check long-term trend

snmp monitoring system



snmp MIB

- Management information base
  - MIB-2, IF-MIB, vender-specific MIB
  - you can get information if an agent supports the MIB you want
- you can specify the information by OIDs
  - ifHCinOctets = .1.3.6.1.2.1.31.1.1.1.6
  - ifHCOutOctets = .1.3.6.1.2.1.31.1.1.1.10

snmp counters

- frequency of updating counters
  - depends on agents (0-30sec)
  - 5min is widely used as snmp polling time
- counter overflow
  - 32bit counters(ifIn/OutOctets) could wrap in 5.7min at 100Mbps
  - consider 64bit counters(ifHCInOctets) for 1Gbps or more interfaces

useful information via SNMP MIBs

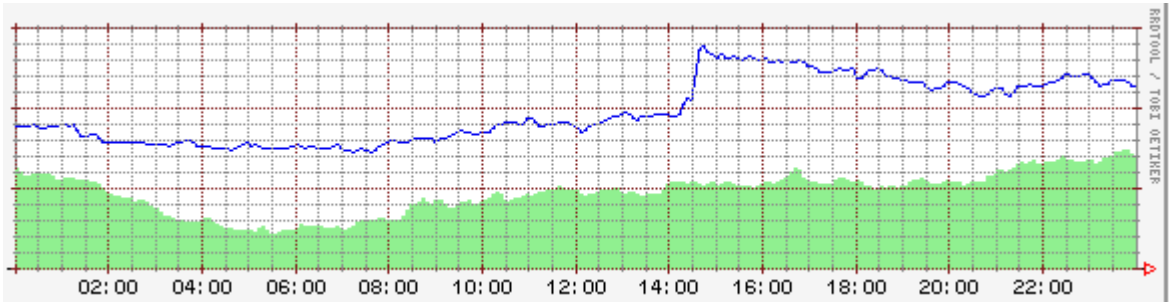
- interface
  - bytes, packets, errors
- system
  - cpu load
  - memory usage
  - temperature
  - icmp, udp
  - ntp

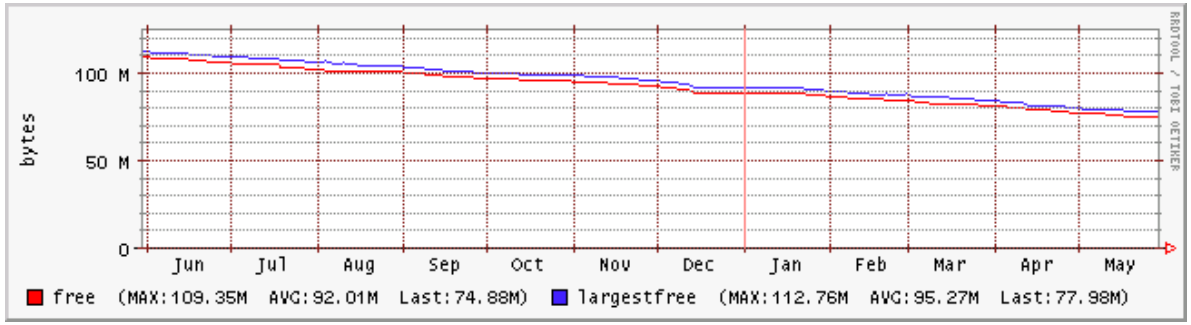
snmp use case

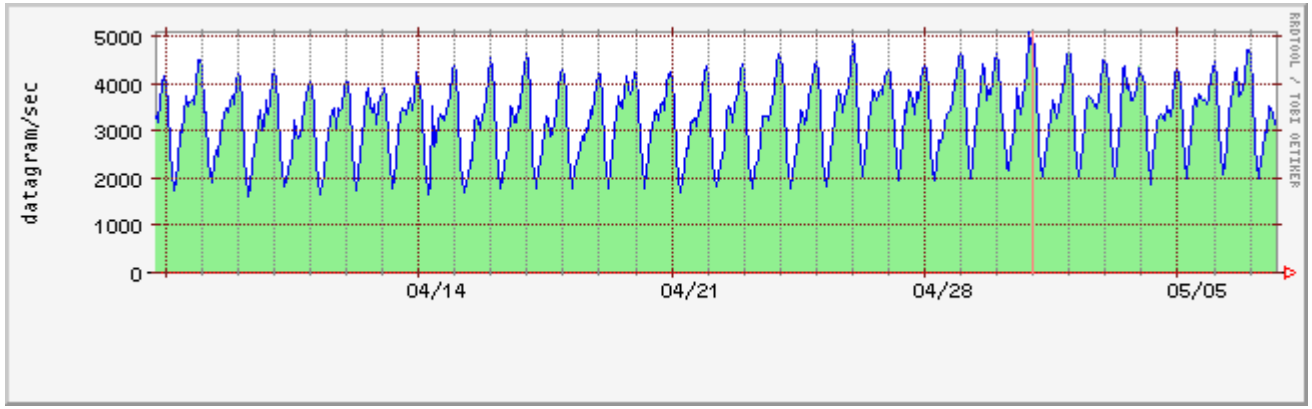
- usage monitoring
  - bandwidth and traffic volume
- visualize
  - stackable graph
    - useful for multiple links between POPs
  - grouping
    - international links
    - IX

visualize

- RRDtools







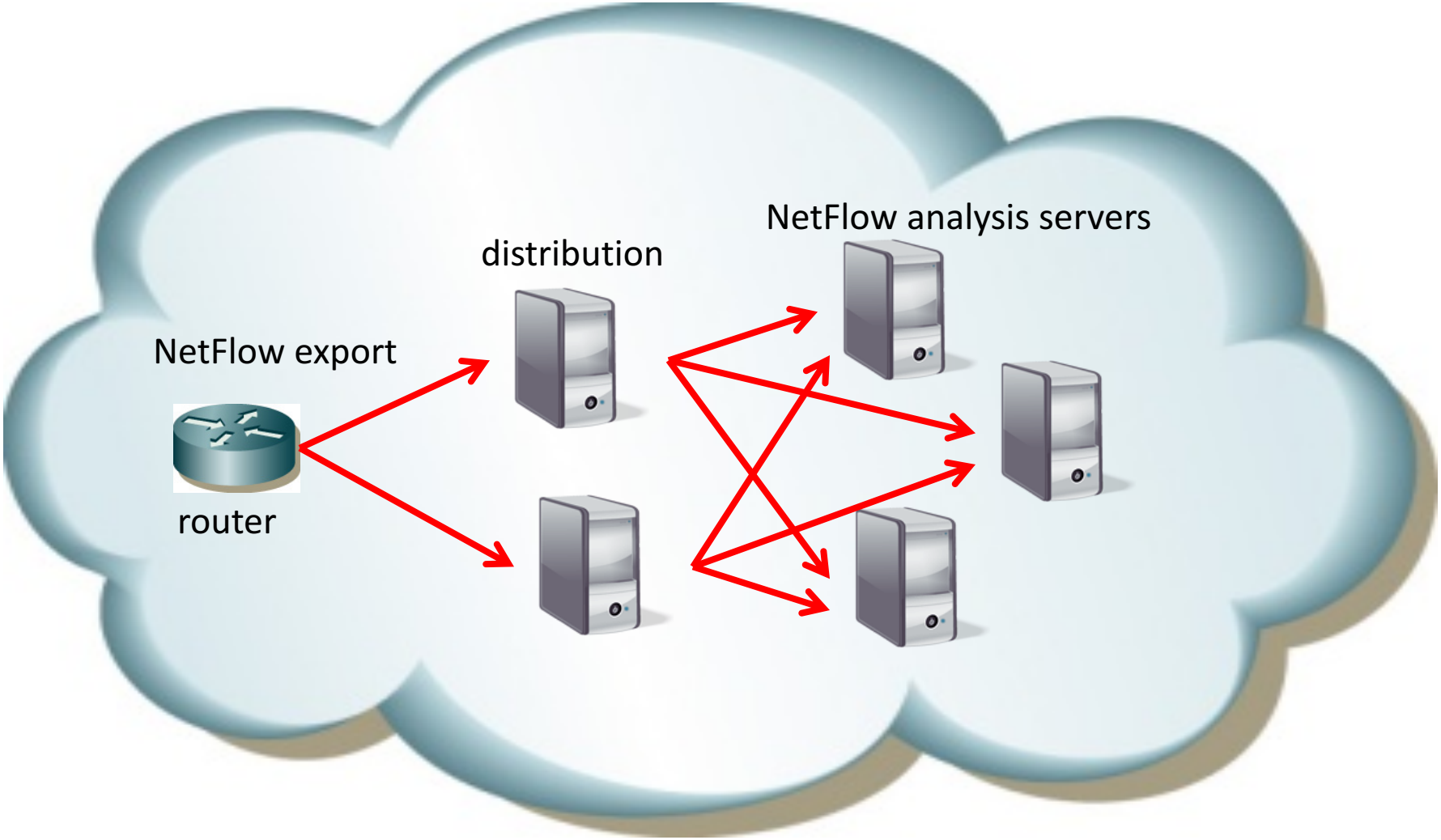
netflow

- to monitor flow information
  - packet header
  - most routers support it
- require more storage
  - even with sampling, still need to expect huge data
  - not for long term monitoring
- useful for analysis and anomaly detection

netflow and sampling

- sampled netflow is widely used
  - just to know trend
  - to reduce data
- margin of error
  - sampled netflow and actual traffic
  - depends on routers
  - worst case: 20%
- IIJ uses magic number as sampling rate
  - 1/16382

netflow monitoring system

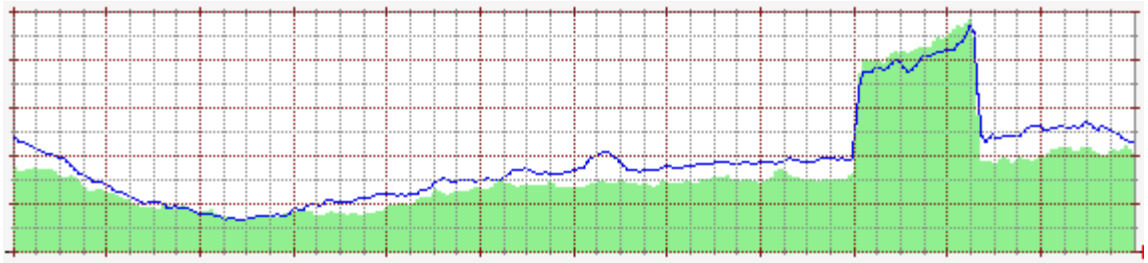


netflow analysis

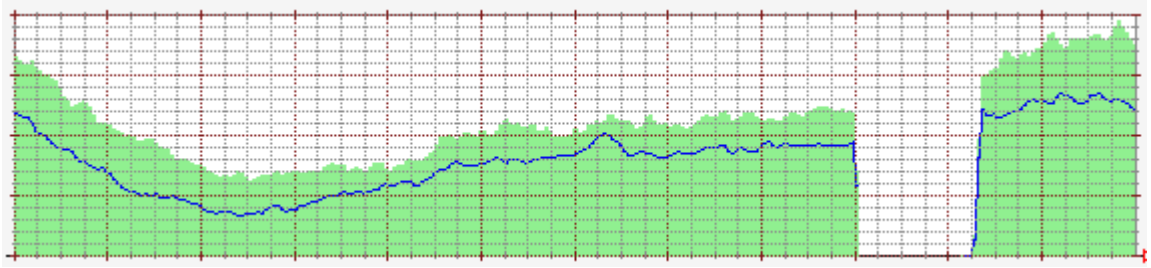
- combination of parameters
  - AS, IP address, protocol, port number
  - too many patterns to pre-generate every graphs
- Graphs
  - pre-defined graphs
  - dynamic graph system

case 1: bps

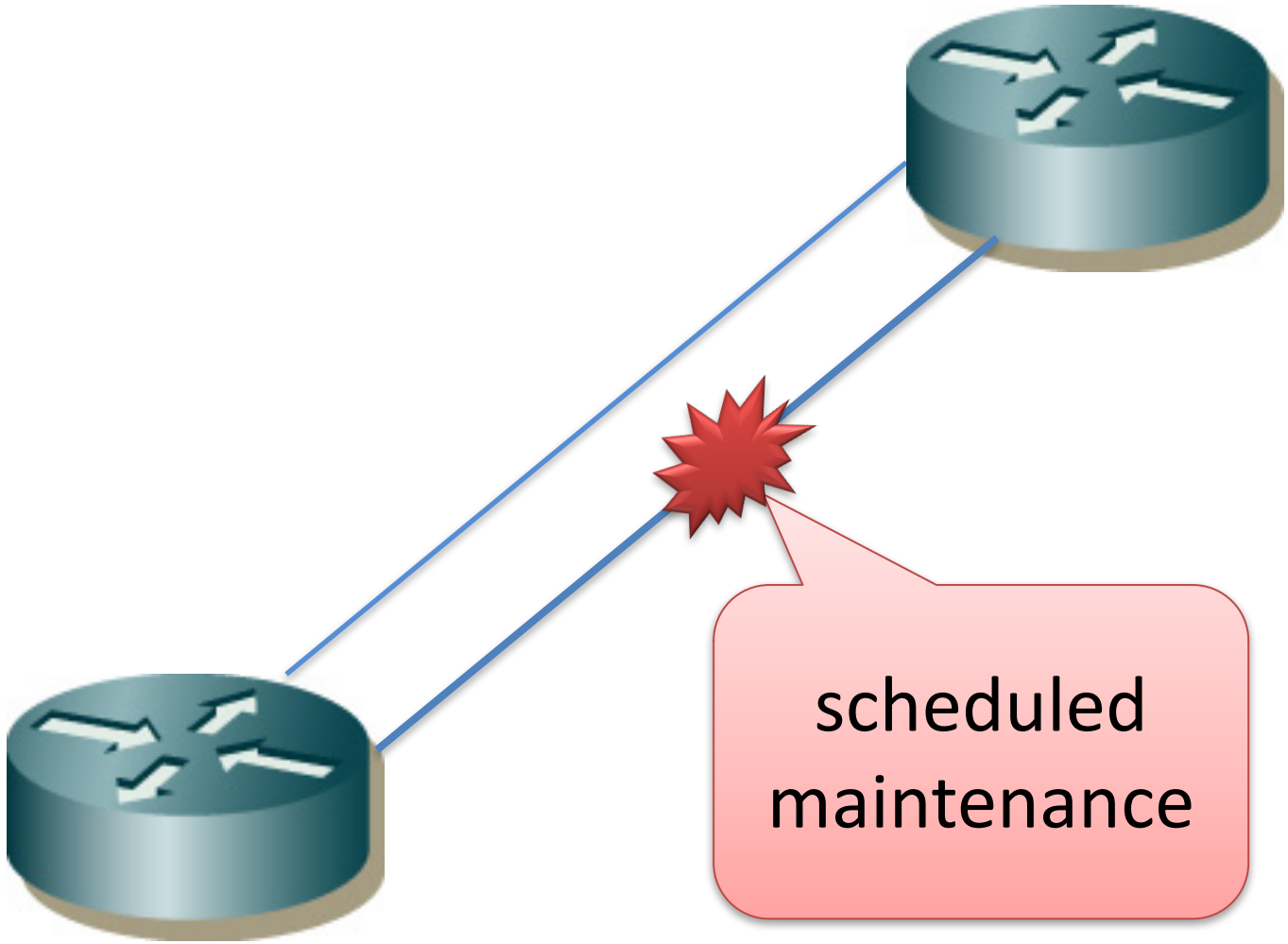
- traffic was suddenly doubled on a link



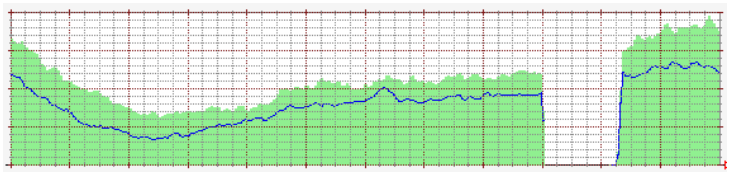
- also found a missing traffic

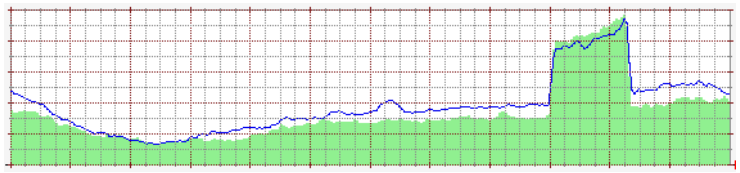


case 1: 2 links between routers

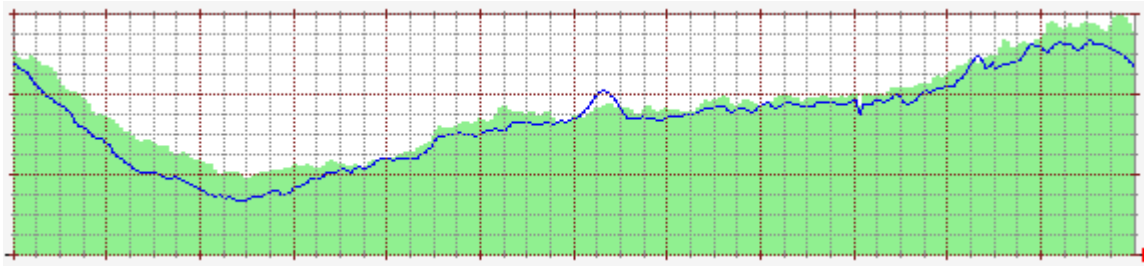


case 1: total traffic: bps



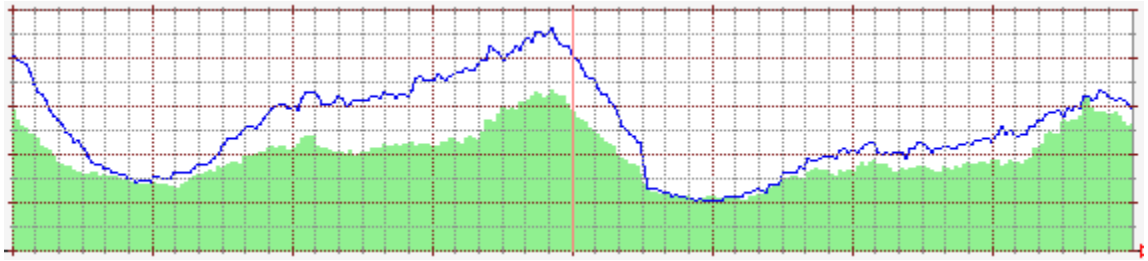


merge

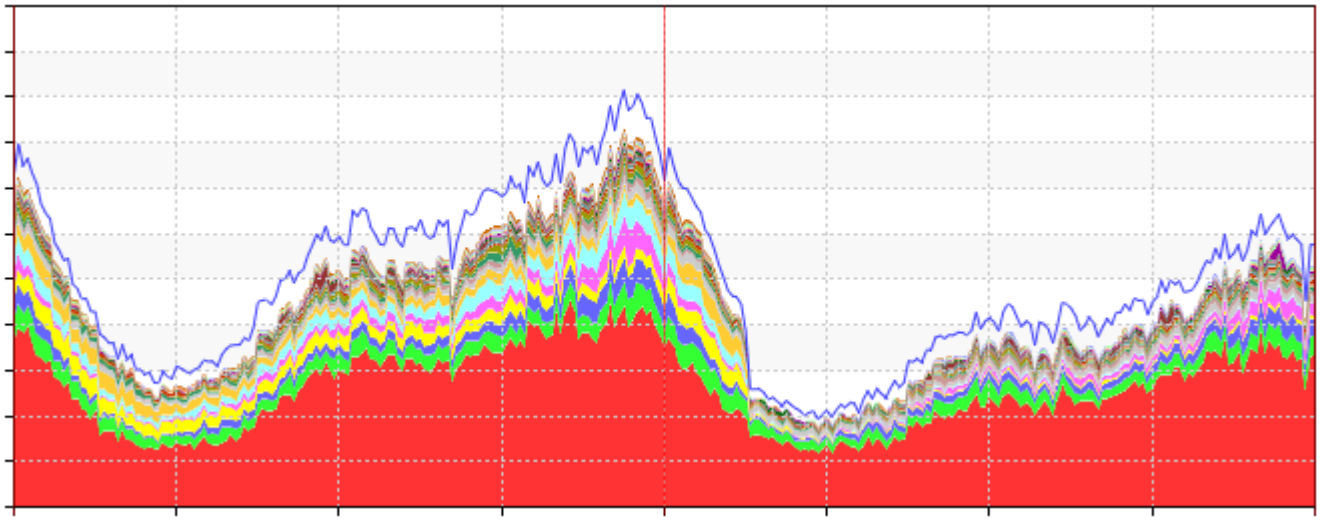


case 2: bps

- traffic decreased
- There is no routing change in the network

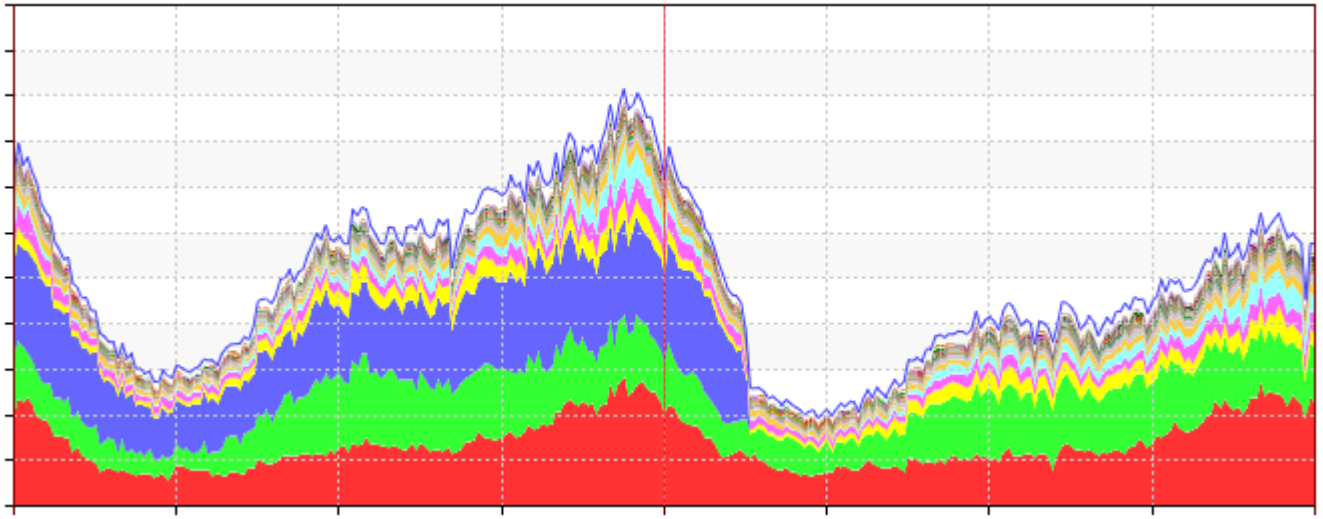


case 2: netflow graph(dst AS)



- the dst AS based graph shows
  - missing traffic to several ASes
  - traffic to the other ASes also a bit decreased

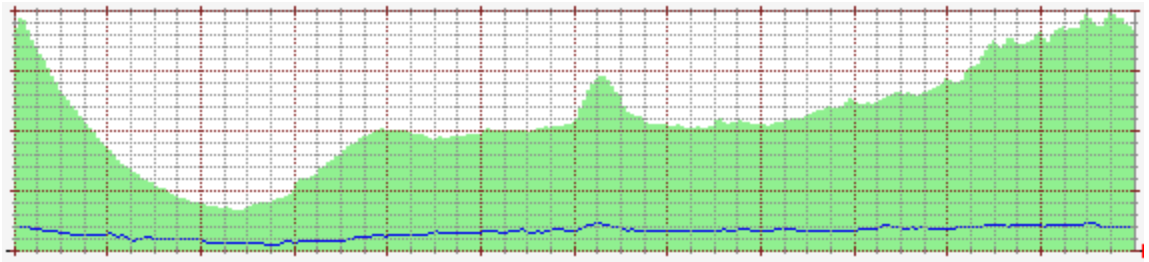
case 2: netflow graph(src AS)



- traffic from a particular AS(blue) was gone
- probably something was happened on the AS(blue)
  - trouble or route change

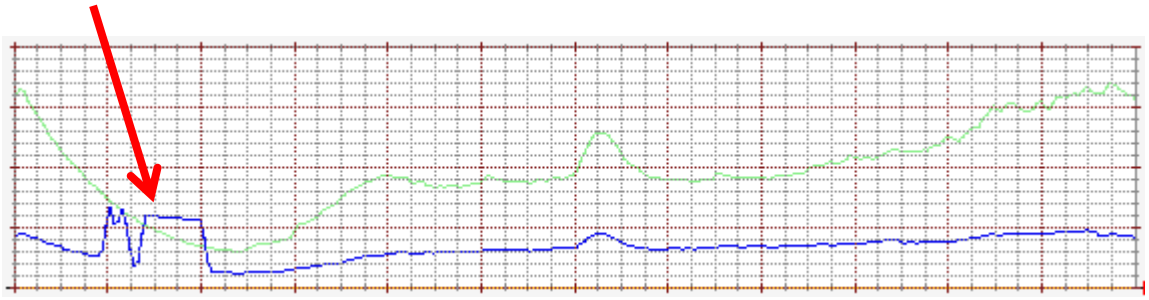
case 3: bps

- traffic looks stable



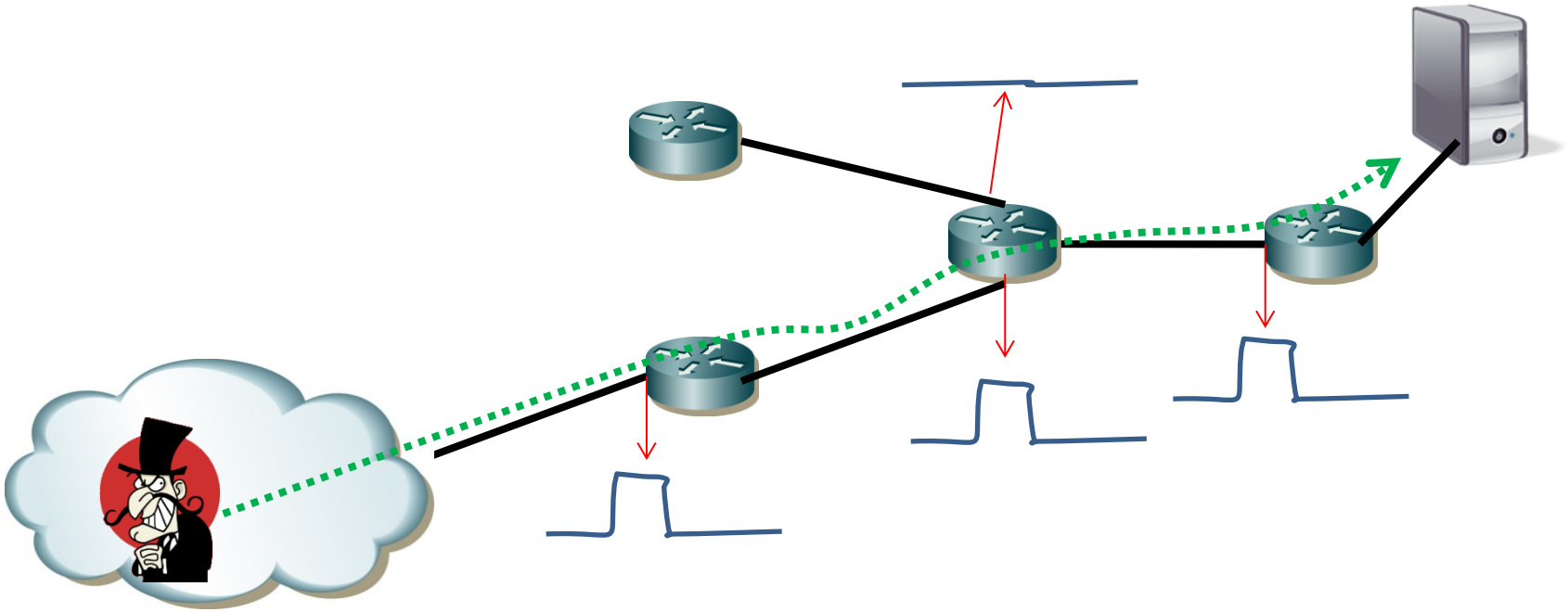
case 3: pps

- pps(packets/sec) graph shows something anomaly

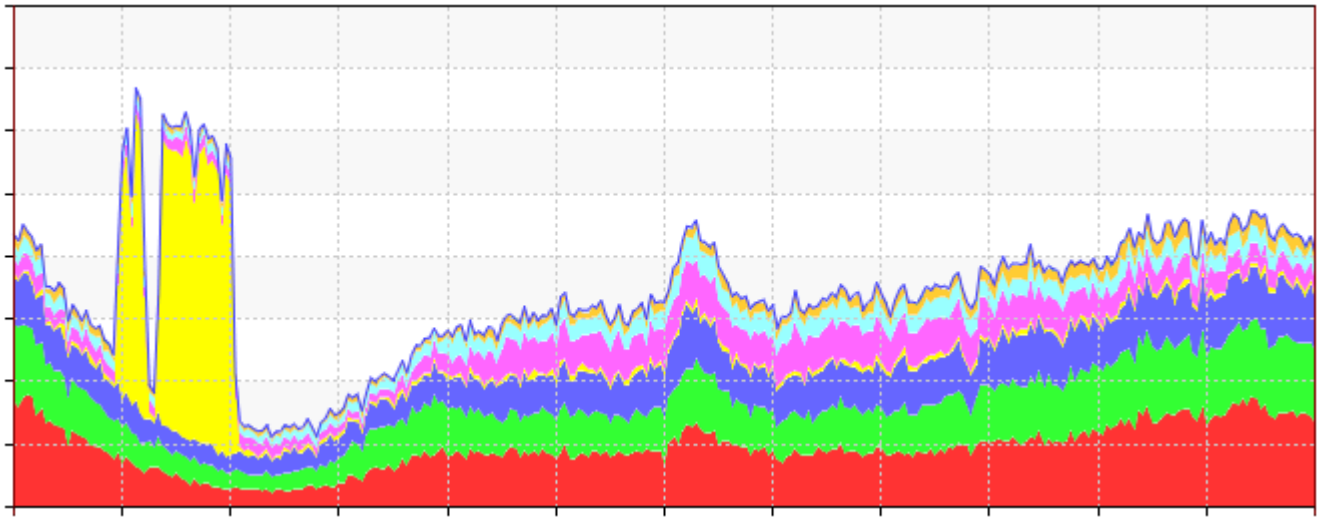


traceback by a shape

- if the traffic pattern is enough characteristic, you can traceback to the inbound interface

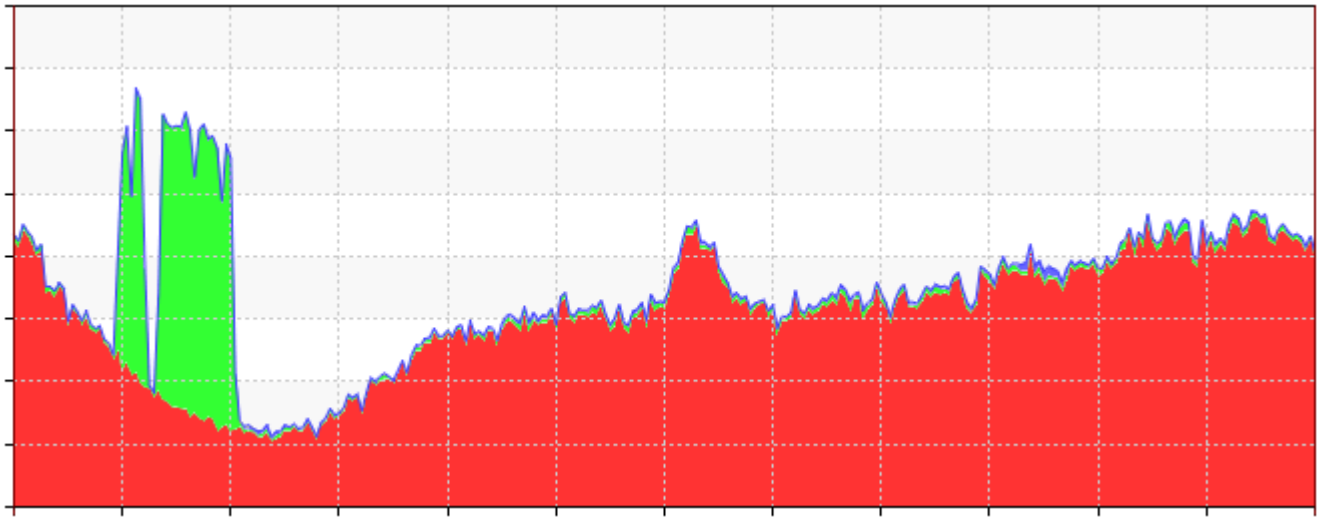


case 3: netflow graph(dst AS, pps)



- according to dst AS based graph, the anomaly traffic was directed to a particular AS(yellow)

case 3: netflow graph(protocol, pps)

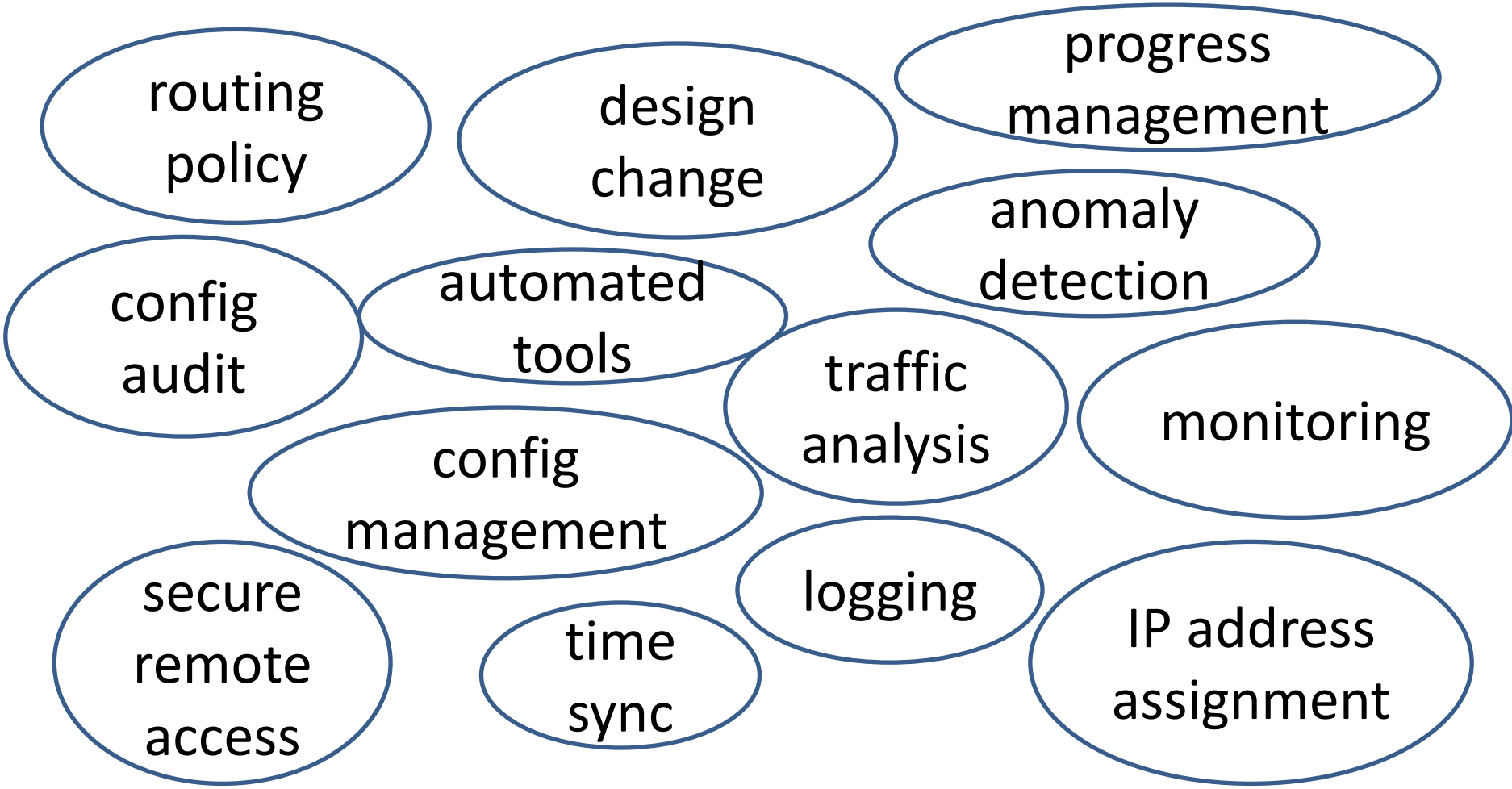


- the traffic profile was mostly UDP

monitoring and detection

- snmp is useful to check
  - trend
  - threshold
- netflow is useful to analysis
  - anomaly
  - change

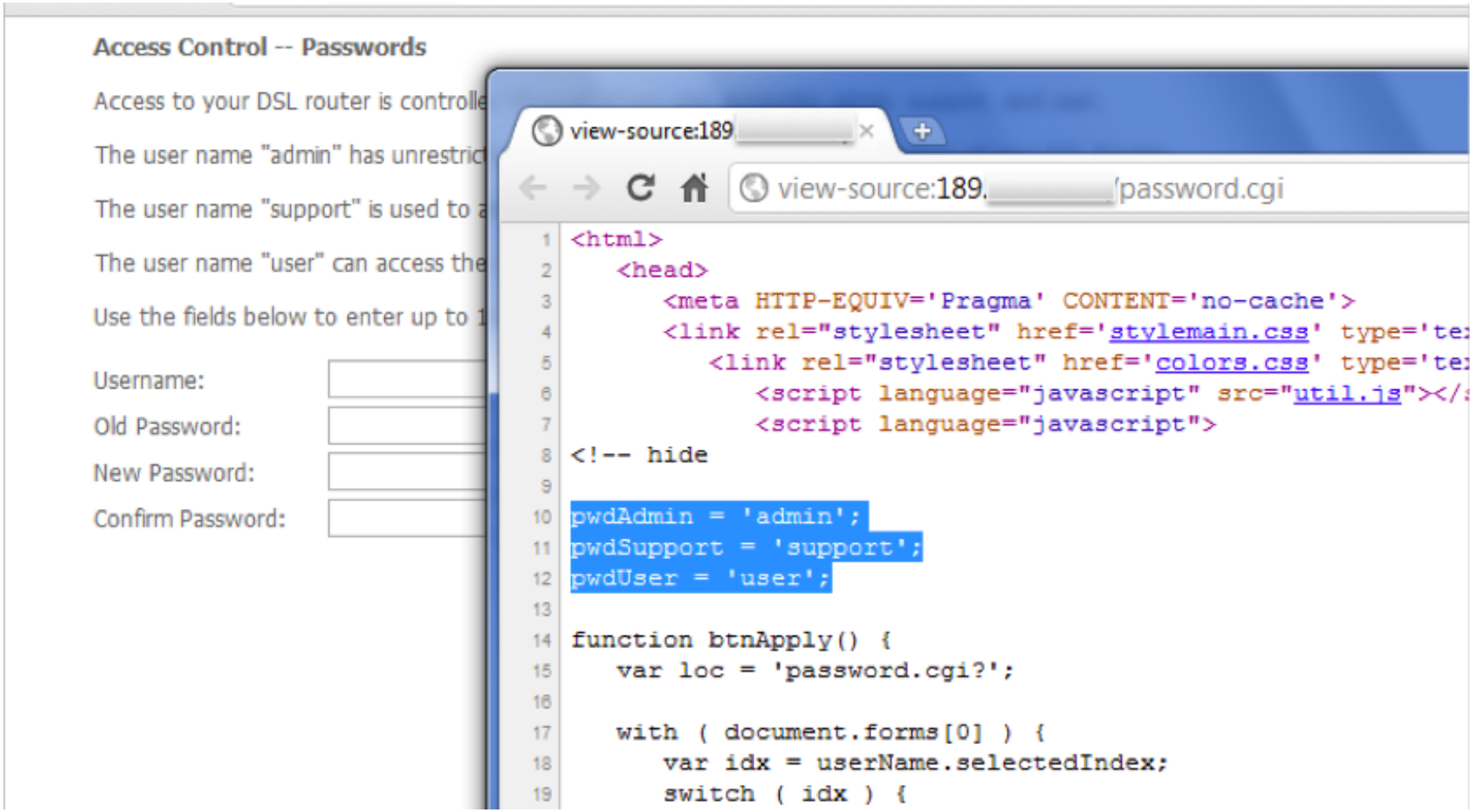
Operational Design



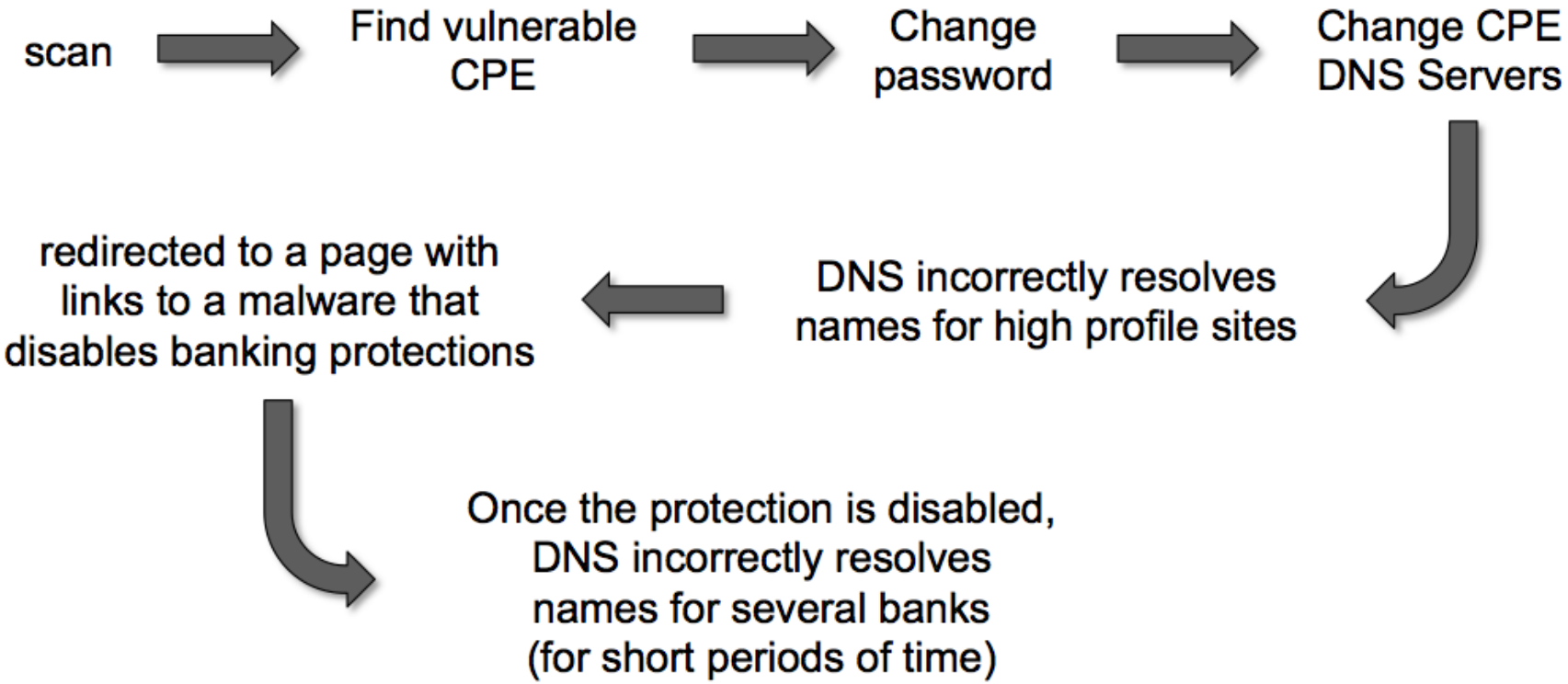
Think of All Devices

- The following problem was recently reported and affects low-end CPEs (ADSL connections only)
  - Admin password exposed via web interface
  - Allow WAN management (this means anyone on Internet)
  - Bug fixed and reintroduced depending on the firmware version
- The bug is quite a number of years old

Password Visible via the Web UI



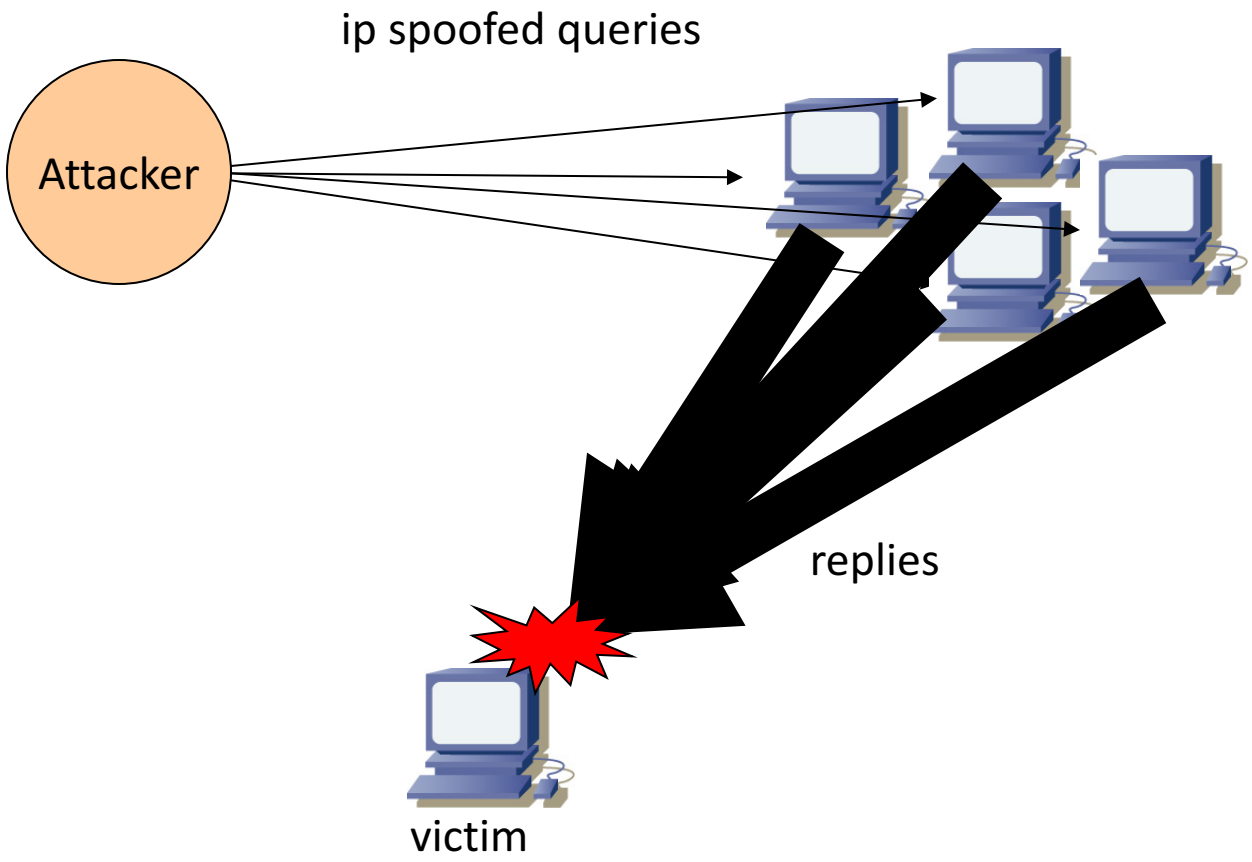
The senario


scan
Find vulnerable CPE
Change password
Change CPE DNS Servers
DNS incorrectly resolves names for high profile sites
redirected to a page with links to a malware that disables banking protections
Once the protection is disabled, DNS incorrectly resolves names for several banks (for short periods of time)

Numbers

- 4.5 Million CPEs (ADSL Modems) using a unique malicious DNS
- In early 2012 more than 300,000 CPEs still infected
- 40 malicious DNS servers found

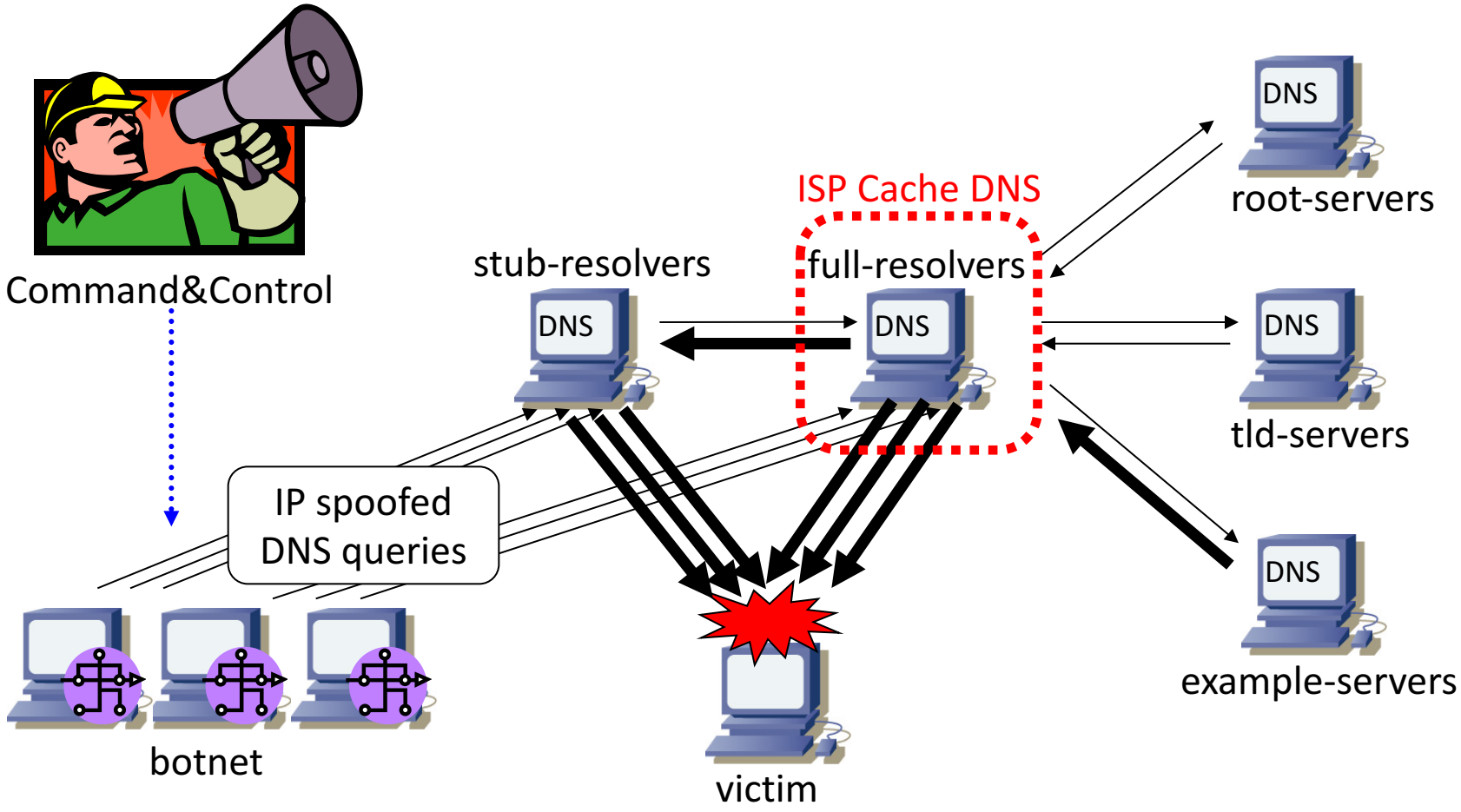
reflection attacks



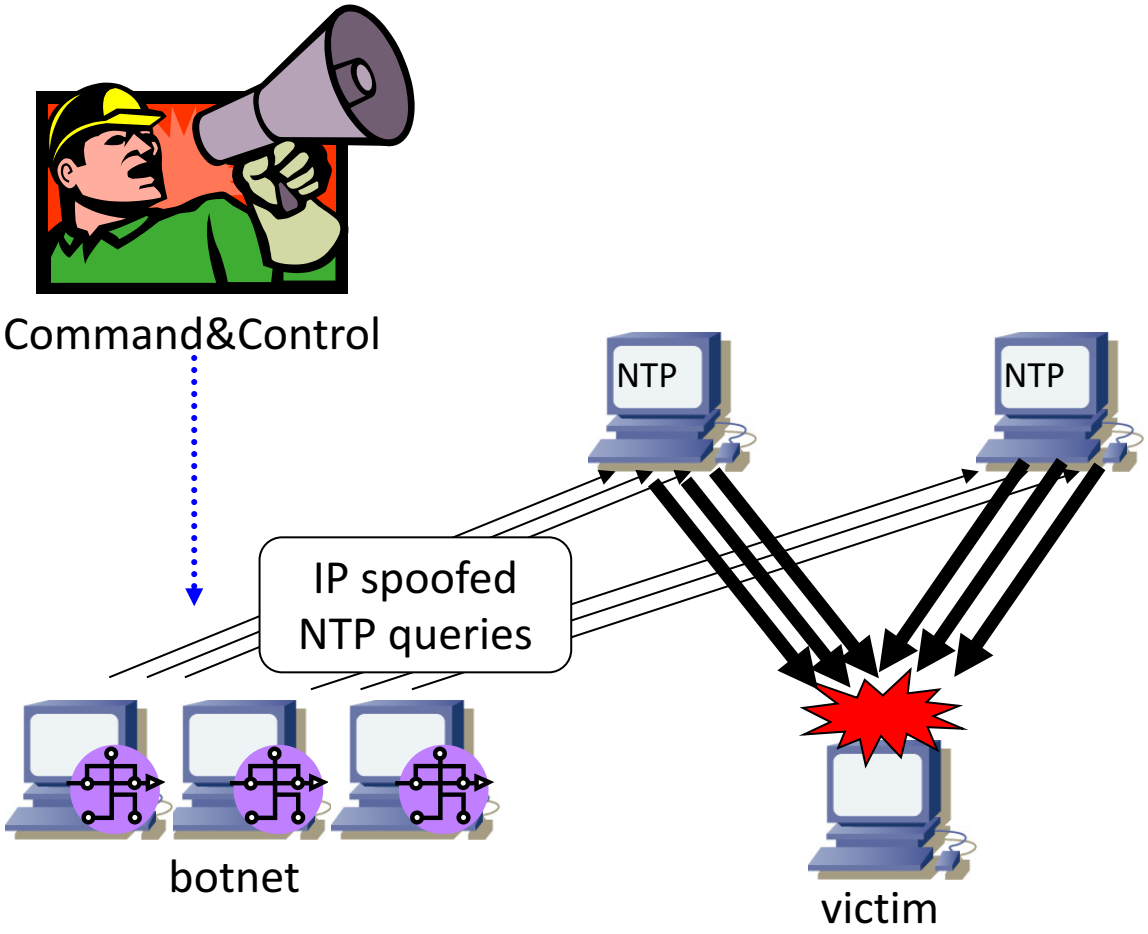
amplifiers

- smurf attack
  - directed broadcast
  - amplification ratio: ~100
- dns amplification attack
  - a huge size record
  - amplification ratio: ~60
- ntp amplification attack
  - monlist query
  - amplification ratio: ~200

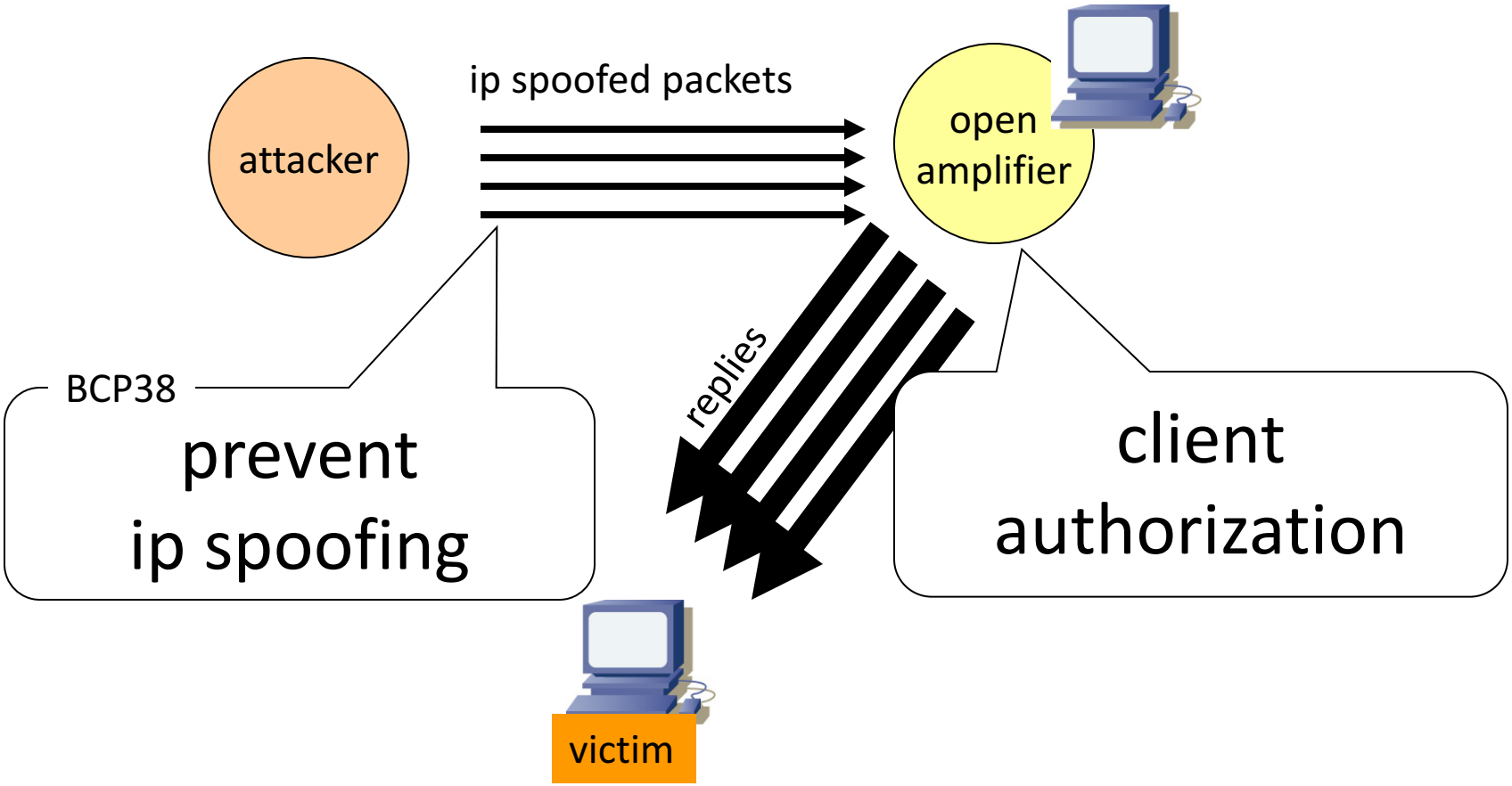
dns amp attack



ntp amp attack



solutions against ip reflection attacks



client authorization

- Incoming interface base
  - useful for home users and enterprises
  - allow from inside, deny from outside
- source IP address base
  - useful for service providers
  - allow from customer network

- you can simply disable the service if it's not necessary

BCP38

- A “Best Current Practice” document of the IETF. BCP38(RFC2827) is intended to limit the impact of DDoS attacks by:
  - Denying traffic with spoofed source address
  - Helping to ensure that traffic is traceable to its correct source network

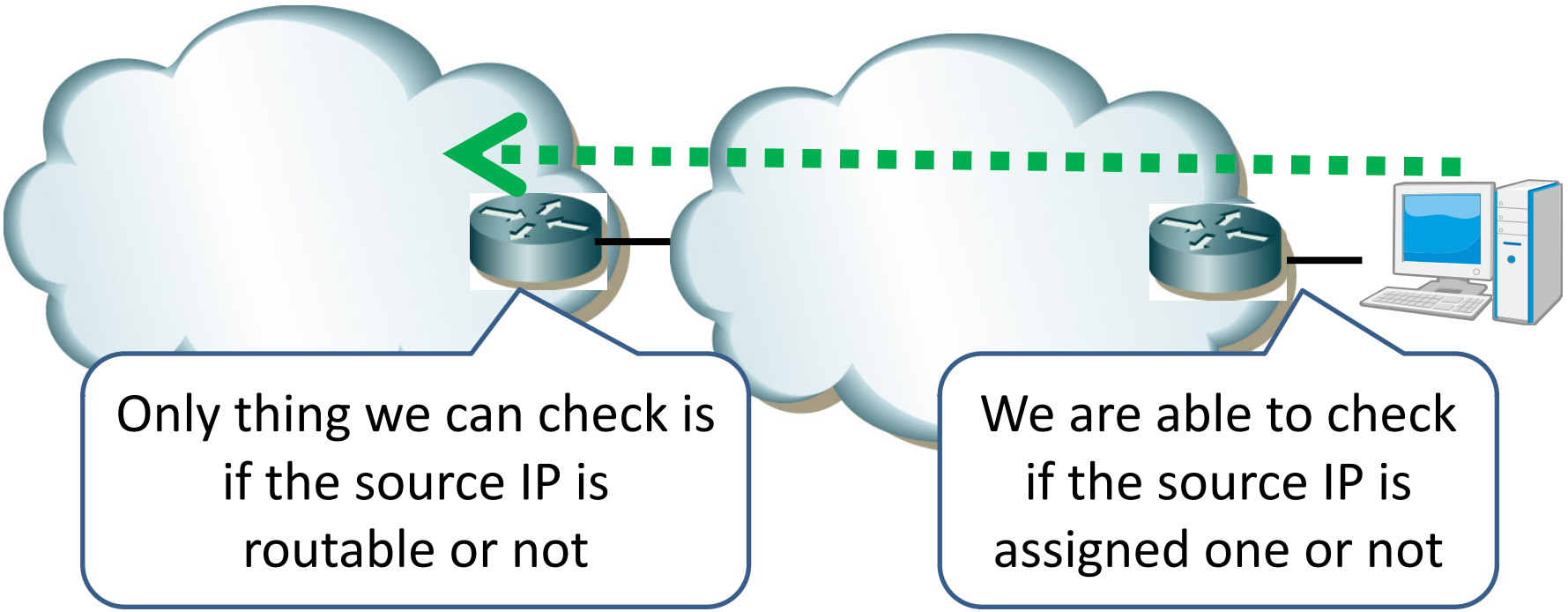
Addressing and Users

- ISP/network administrator assigns IP prefix(es) to their users
  - dynamic or static
  - DHCP, PPP, RA
- Users should use these assigned IP prefixes as their source IP address

BCP38 implementation

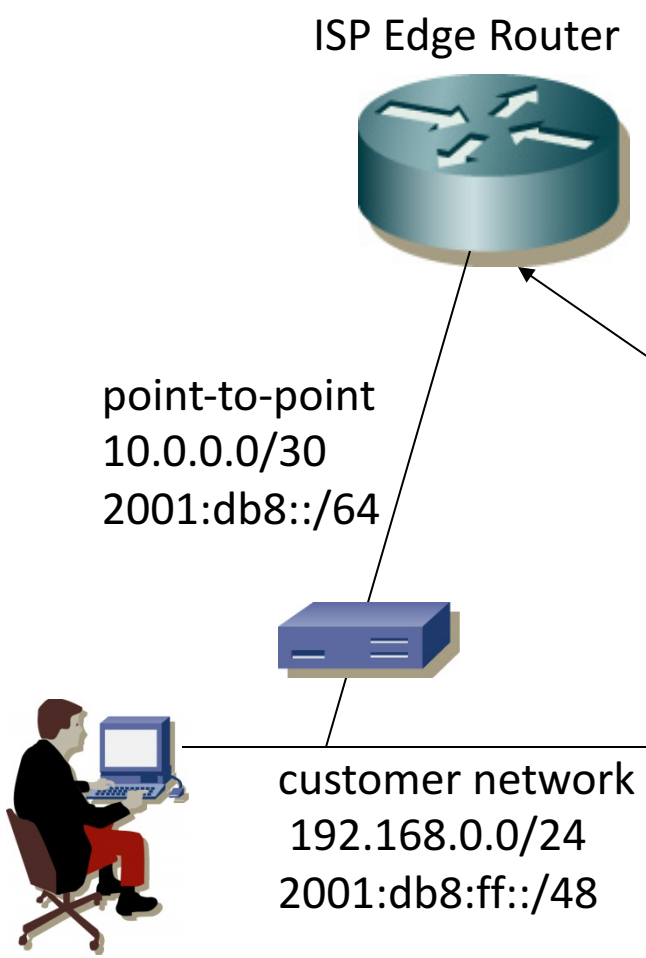
- ACL
  - packet filter
  - permit valid-source, then drop any
- uRPF check
  - checks incoming packets using ‘routing table’
  - look-up a return path for the source IP address
  - loose mode can’t stop most misuse
    - use strict mode

deployment point



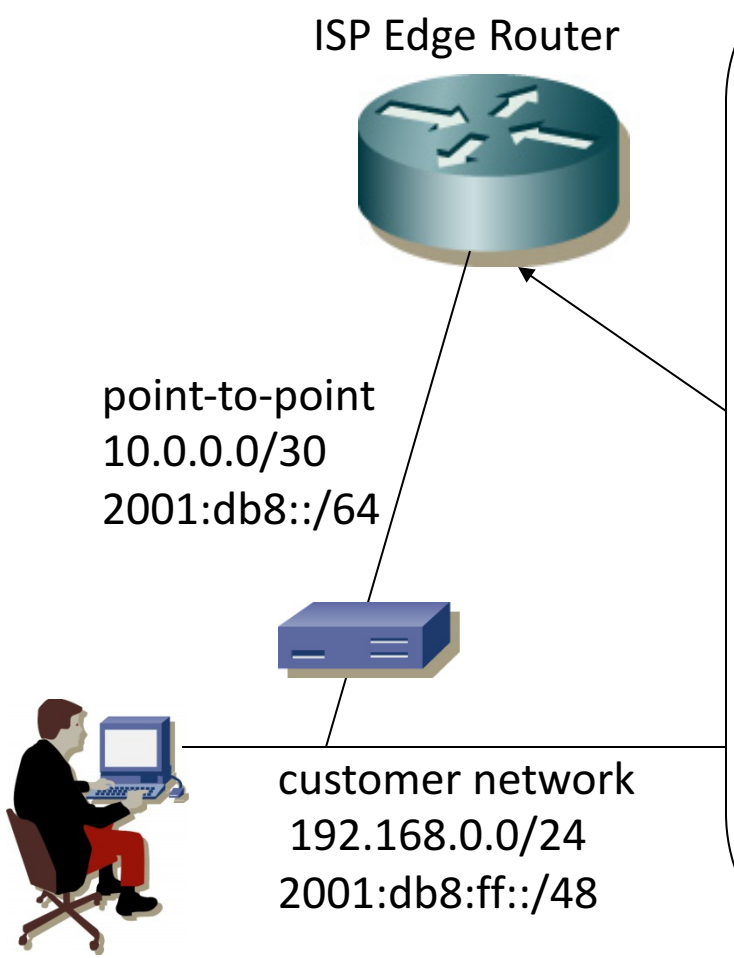
- ISP Edge (customer aggregation) router
  - close to packet source as possible

cisco ACL example



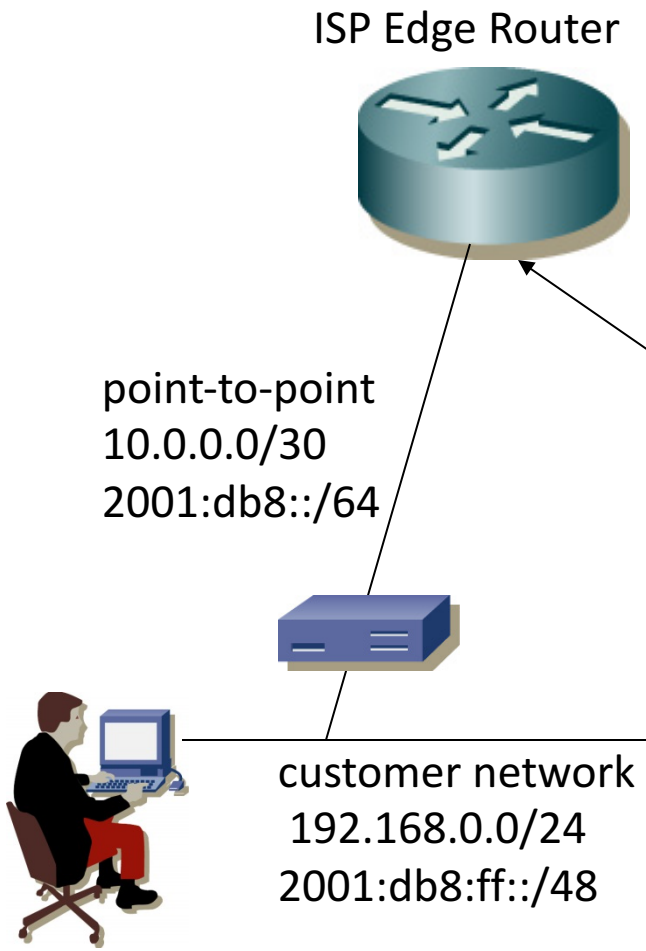

```
ip access-list extended fromCUSTMER4
 permit ip 192.168.0.0 0.0.255.255 any
 permit ip 10.0.0.0 0.0.0.3 any
 deny   ip  any any
!
IPv6 access-list fromCUSTMER6
 permit ipv6 2001:db8::/64 any
 permit ipv6 any 2001:db8::/64 any
 permit ipv6 2001:db8:ff::/48 any
 permit ipv6 fe80::/10 fe80::/10
 permit ipv6 fe80::/10 ff02::/16
 deny ipv6 any any
!
interface Gigabitethernet0/0
 ip access-group fromCUSTOMER4 in
 ipv6 traffic-filter fromCUSTOMER6 in
```

juniper IPv4 ACL example



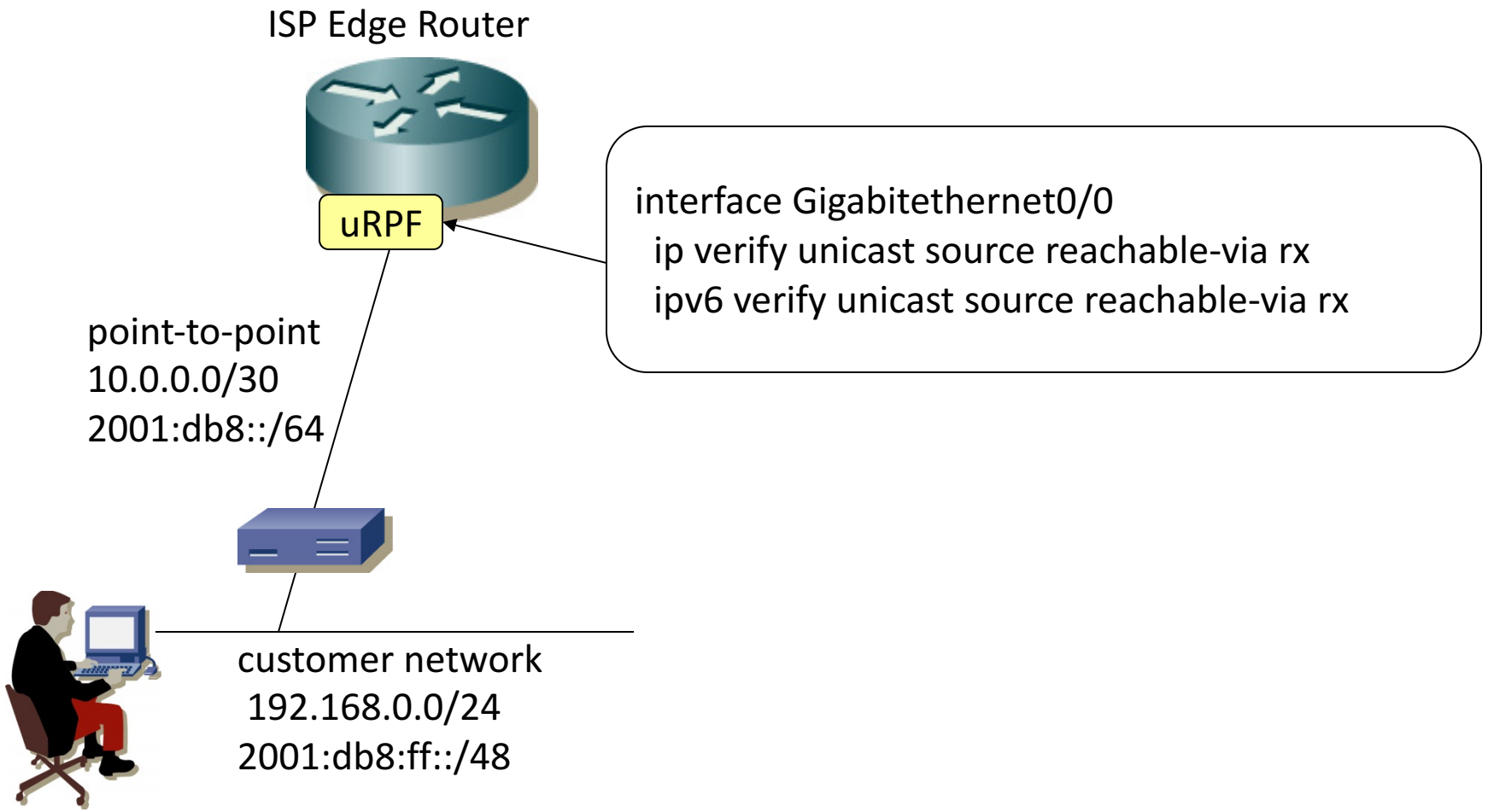

```
firewall family inet {
 filter fromCUSTOMER4 {
  term CUSTOMER4 { from
   source-address {
    192.168.0.0/16;
    10.0.0.0/30;
   }
   then accept;
  }
  term Default {
   then discard;
}}}
[edit interface ge-0/0/0 unit 0 family inet]
filter {
 input fromCUSTOMER;
}
```

juniper IPv6 ACL example

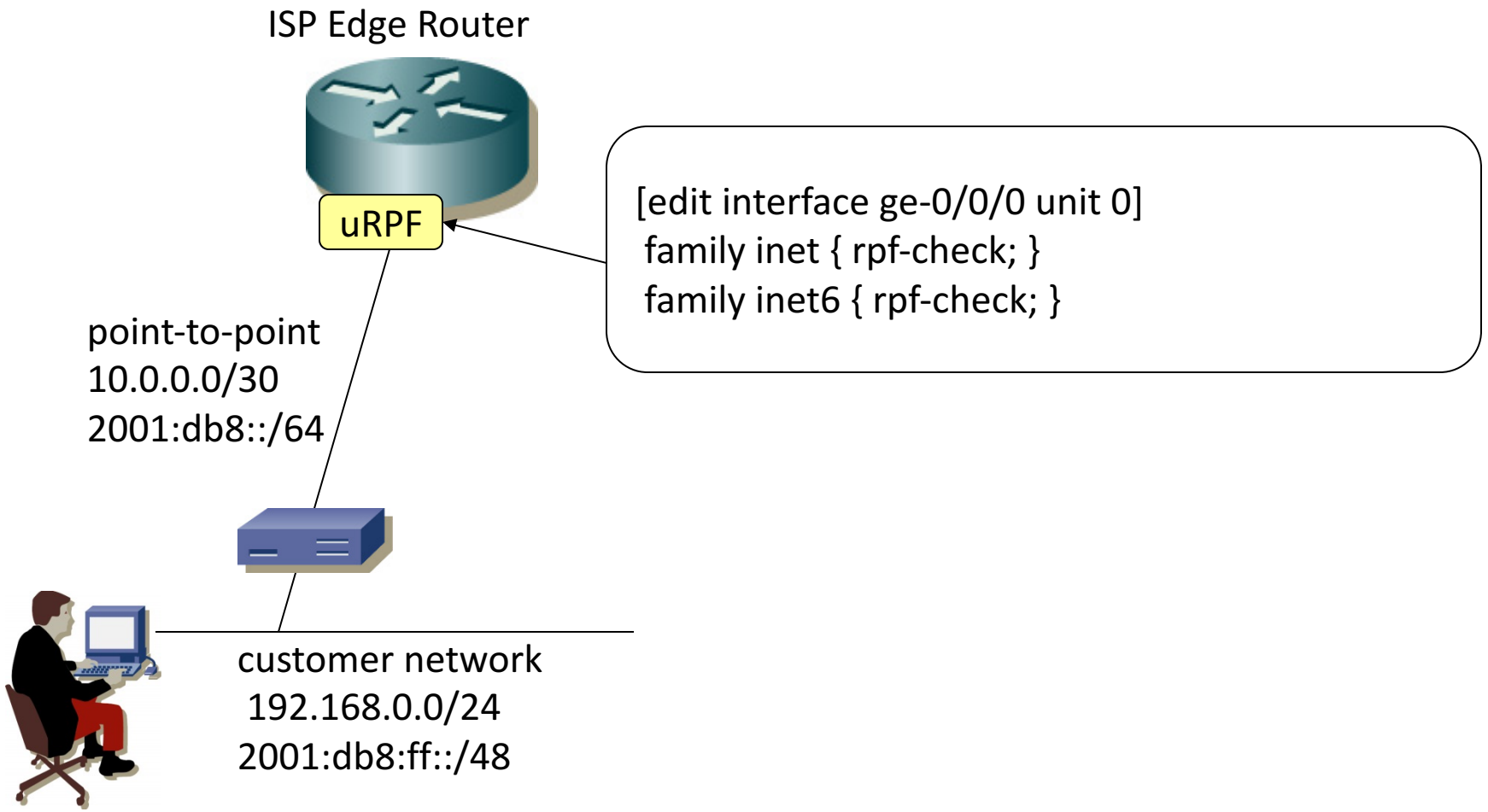



```
firewall family inet6 {
 filter fromCUSTOMER6 {
  term CUSTOMER6 { from
   source-address {
    2001:db8::/64;
    2001:db8:ff::/48;
   }
   then accept;
  }
  term LINKLOCAL { from
    source-address {
      fe80::/10;
    } destination-address {
      fe80::/10;
      ff02::/16;
    }
    then accept;
  }
  term Default {
   then discard;
}}}
[edit interface ge-0/0/0 unit 0 family inet6]
filter {
 input fromCUSTOMER6;
}
```

cisco uRPF example

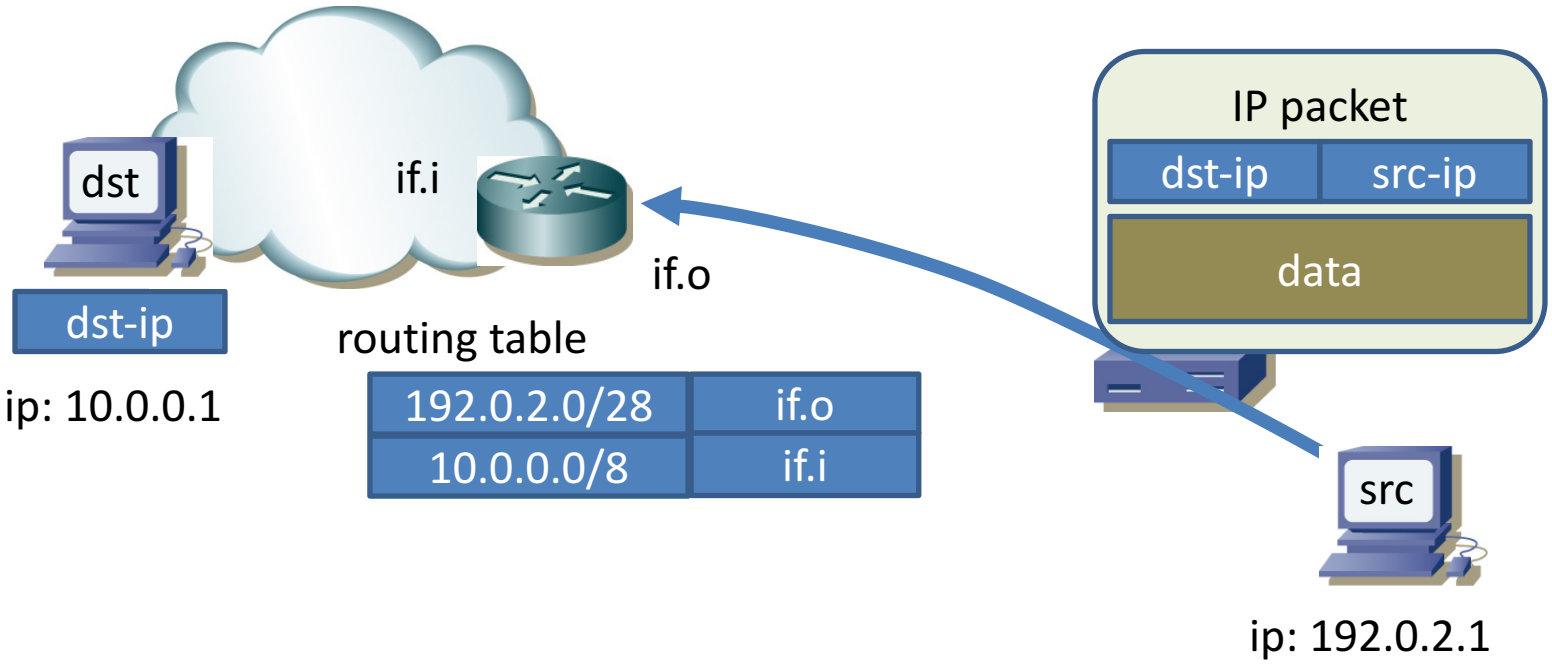


juniper uRPF example



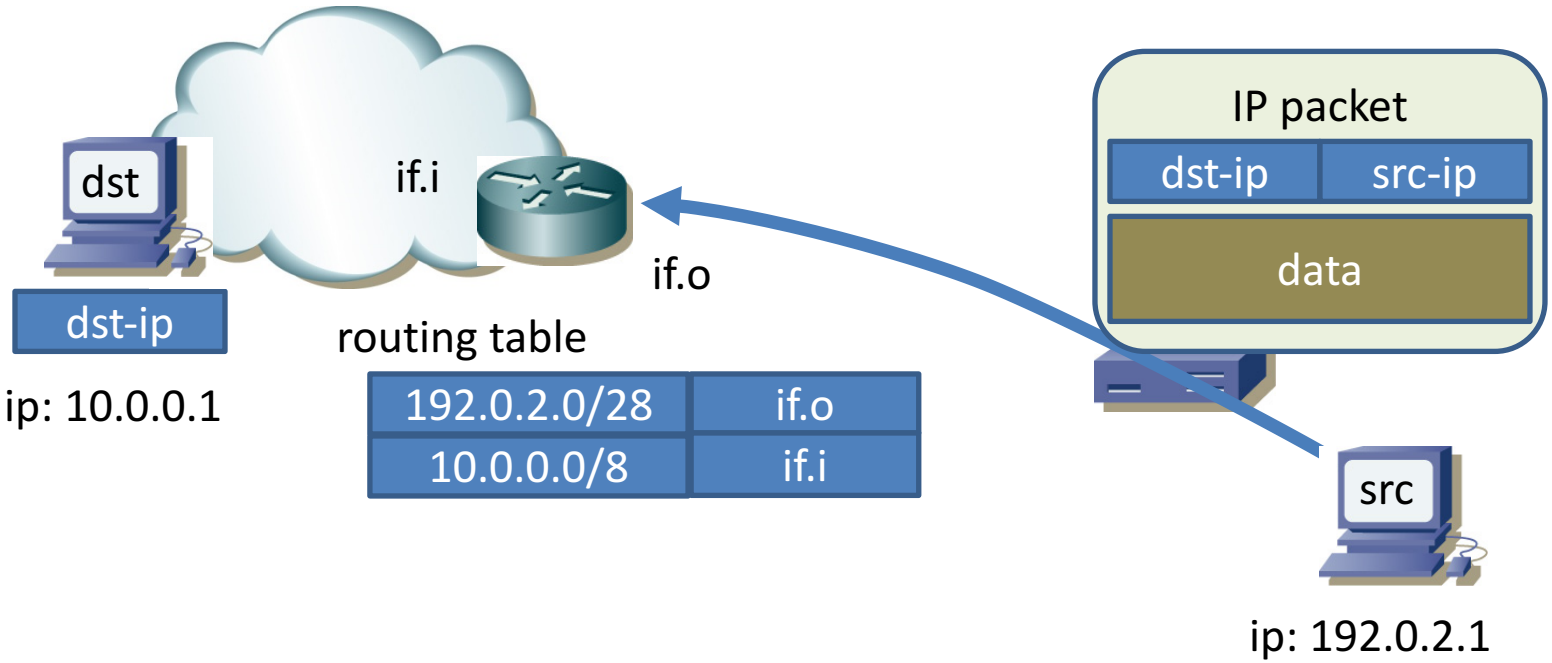
packet forwarding – dst-ip based

- routing_table(dst-ip) => outgoing interface
  - lookup by 10.0.0.1 => if.i
  - then router forwards the packet

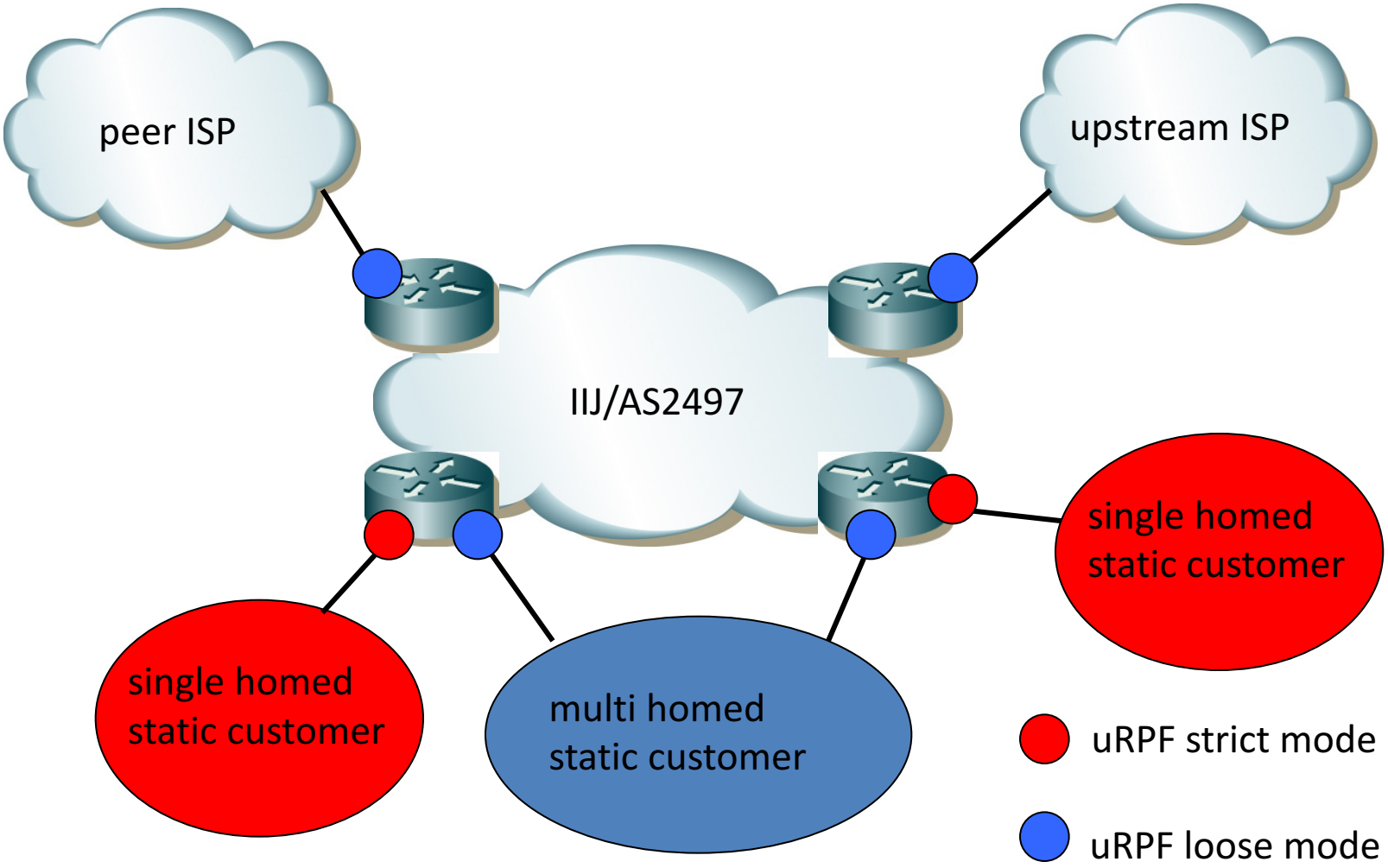


uRPF check – lookup by the src-ip

- routing_table(src-ip) => interface
  - lookup by 192.0.2.1 => if.o
  - The result MUST match the incoming interface



IIJ's policy

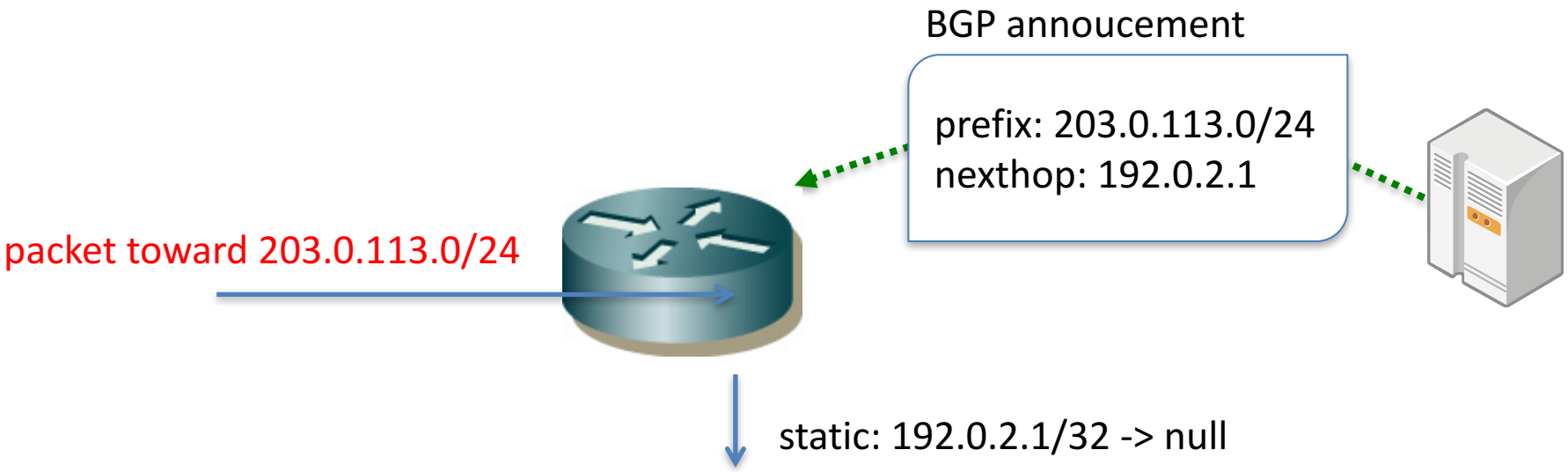

peer ISP
upstream ISP
IIJ/AS2497
single homed
static customer
single homed
static customer
multi homed
static customer
uRPF strict mode
uRPF loose mode

blackhole routing

- routers are good at forwarding
  - not packet filtering
- use the forwarding function to discard packets
  - null routing

RTBH

- Remote Triggered Black Hole

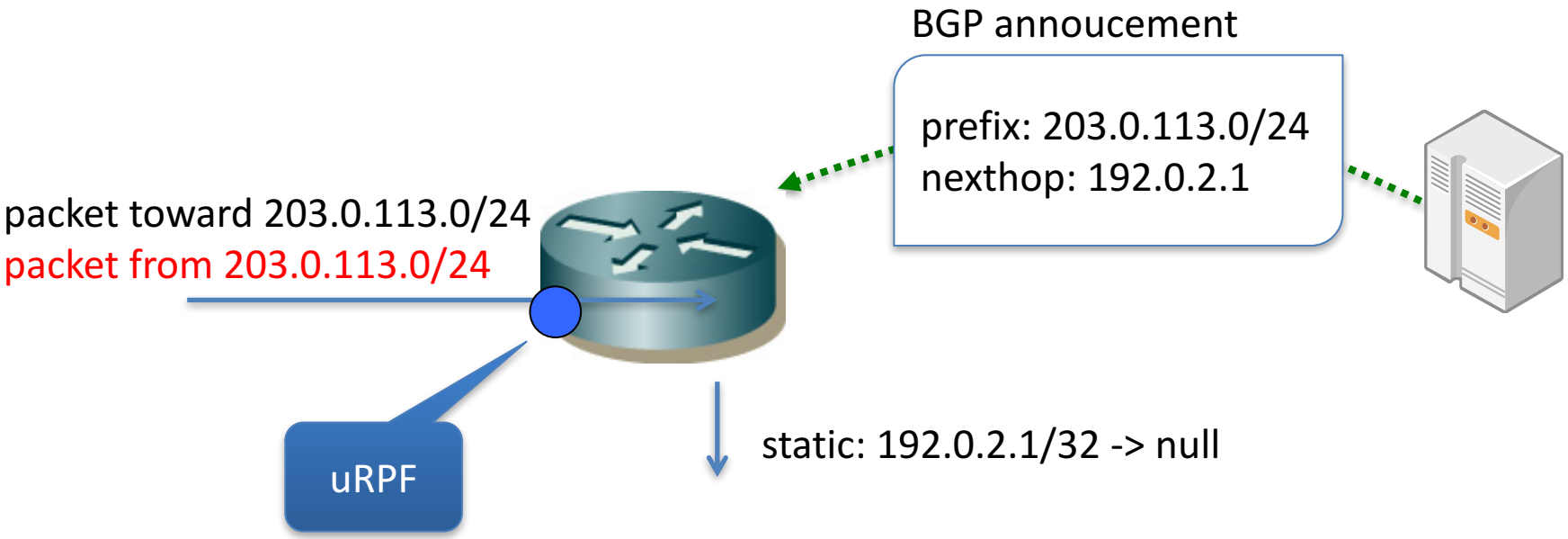


uRPF and blackhole routing

- you can drop a packet that has source ip matches those blackhole route
  - cisco and juniper(>junos12.1)
- source IP address based filtering

RTBH w/ uRPF

- Remote Triggered Black Hole



packet filtering for transit traffic

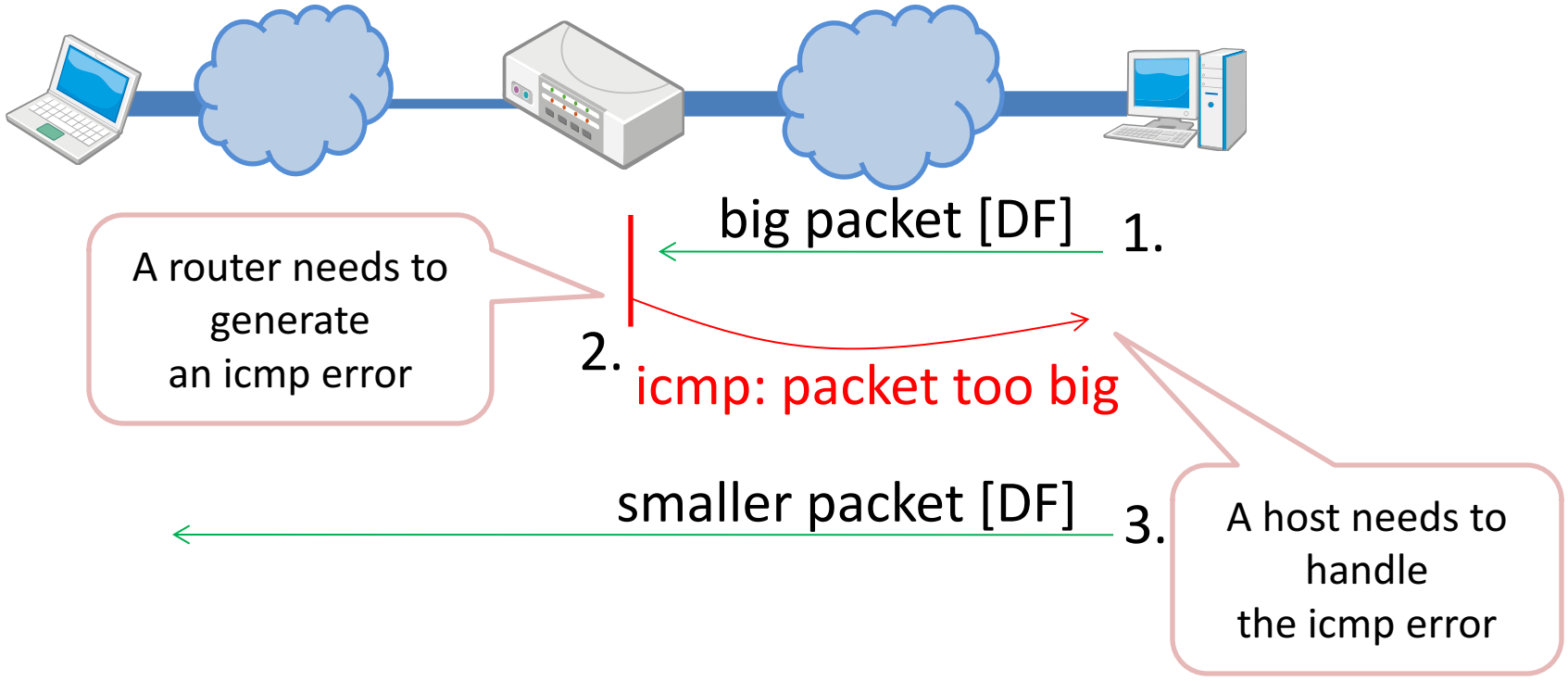
- IP is not that simple
  - IP fragments
  - path MTU discovery
- IPv6, DNSSEC and so on

Path MTU Discovery

- Path MTU discovery [RFC1191]
- Path MTU discovery for IPv6 [RFC1981]

- IPv4 minimum link MTU [RFC791] == 68
  - 576 is widely accepted though
- IPv6 minimum link MTU [RFC2460] == 1280

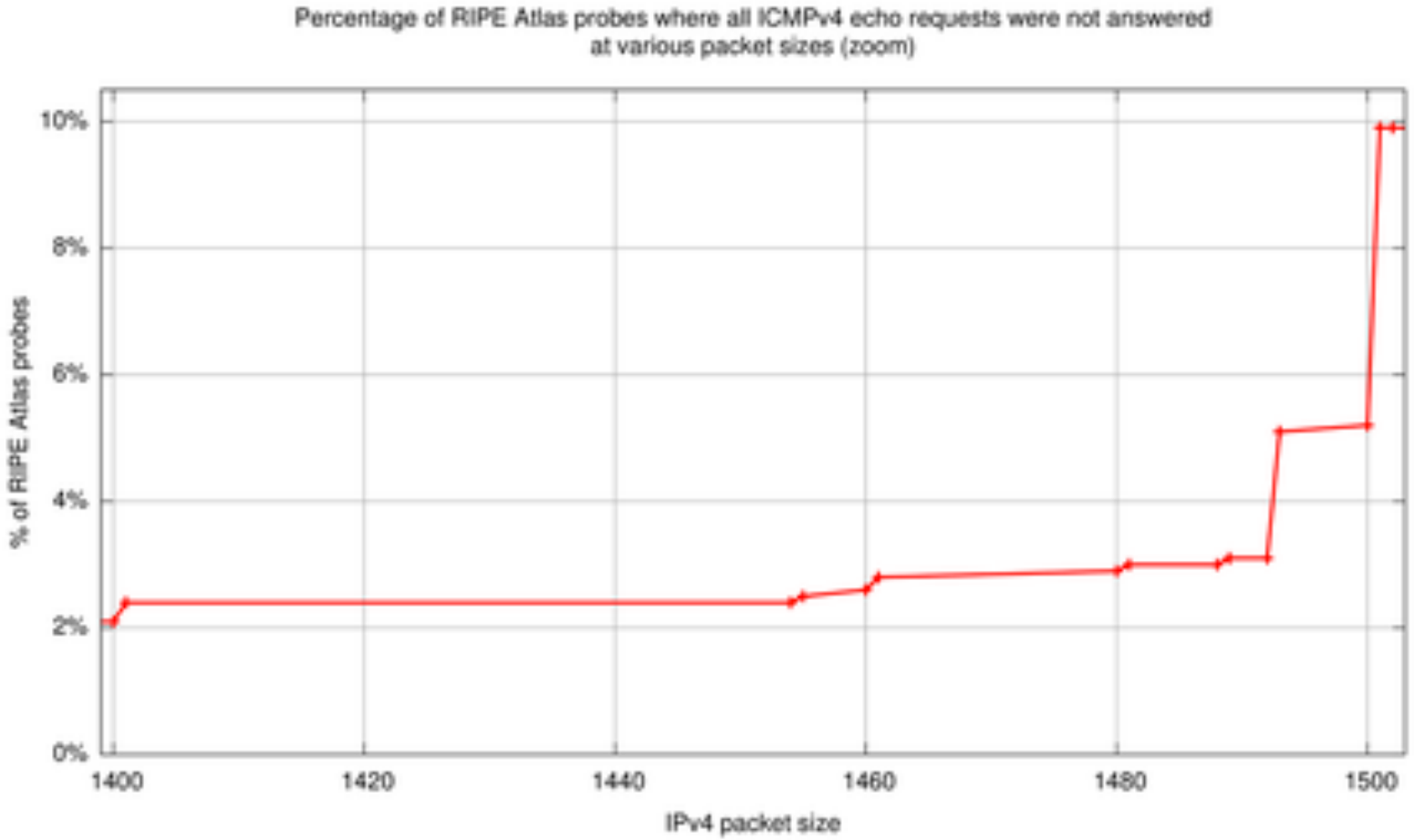
path MTU discovery scenario



icmp originating-limit

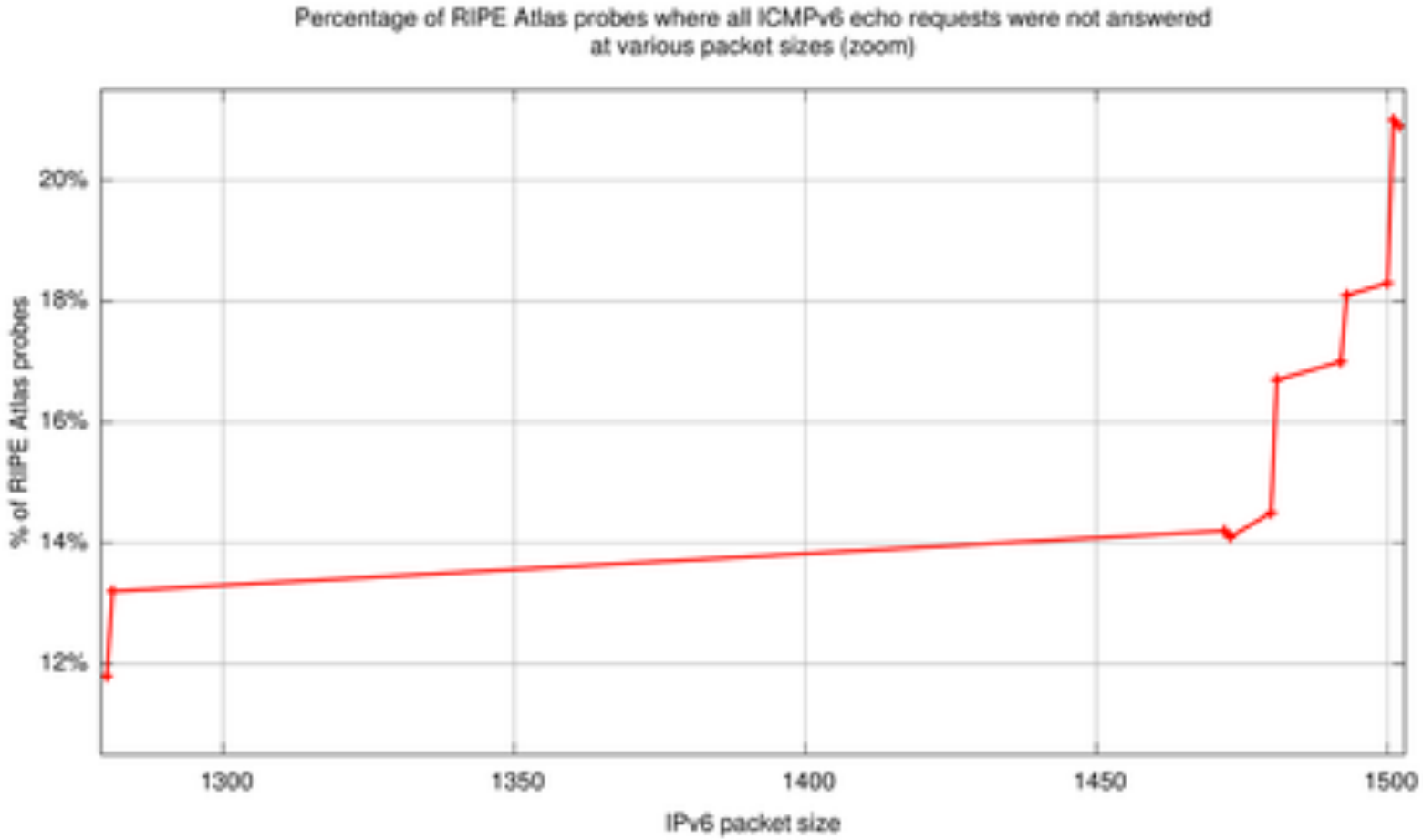
- cisco ios
  - ip icmp rate-limit unreachable 500
    - icmp errors are limited to one every 500msec
  - ipv6 icmp error-interval 100
    - icmp errors are limited to one every 100msec
- juniper junos
  - icmpv4-rate-limit {packet-rate 1000;};
    - up to 1000pps icmp packets to/from RE
  - icmpv6-rate-limit {packet-rate 1000;};
    - up to 1000pps icmp packets to/from RE

IPv4 pMTUd fails



https://labs.ripe.net/Members/emileaben/ripe-atlas-packet-size-matters

IPv6 as well

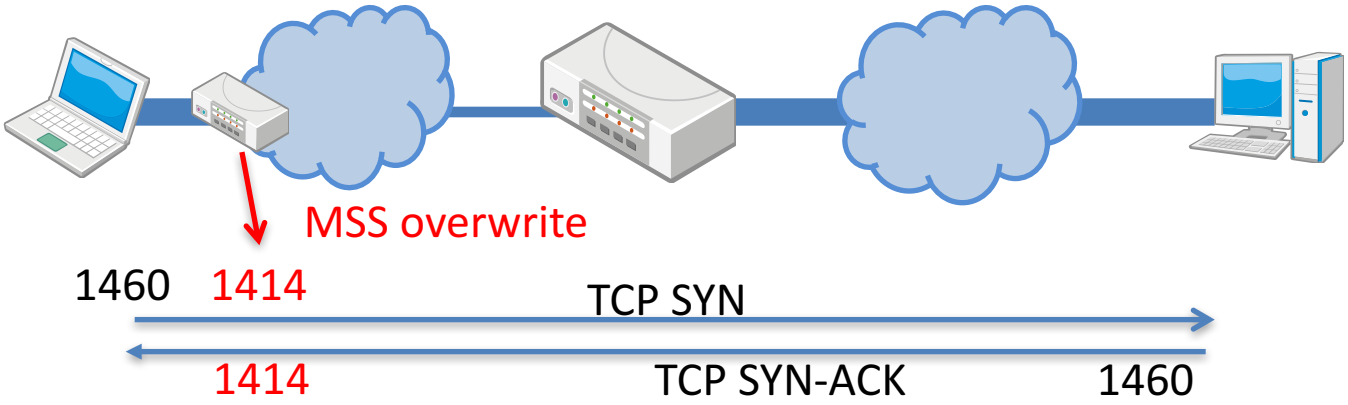


https://labs.ripe.net/Members/emileaben/ripe-atlas-packet-size-matters

learning from IPv4

- Almost of all broadband routers have a TCP MSS hack capability
- It chokes TCP MSS on a tunnel link
  - PPPoE, or whatever the link MTU is less than 1500
  - to avoid unnecessary fallbacks
- The TCP MSS hack works fine
  - No complaint from customers

TCP MSS hack



- both ends agree to use 1414 as MSS size

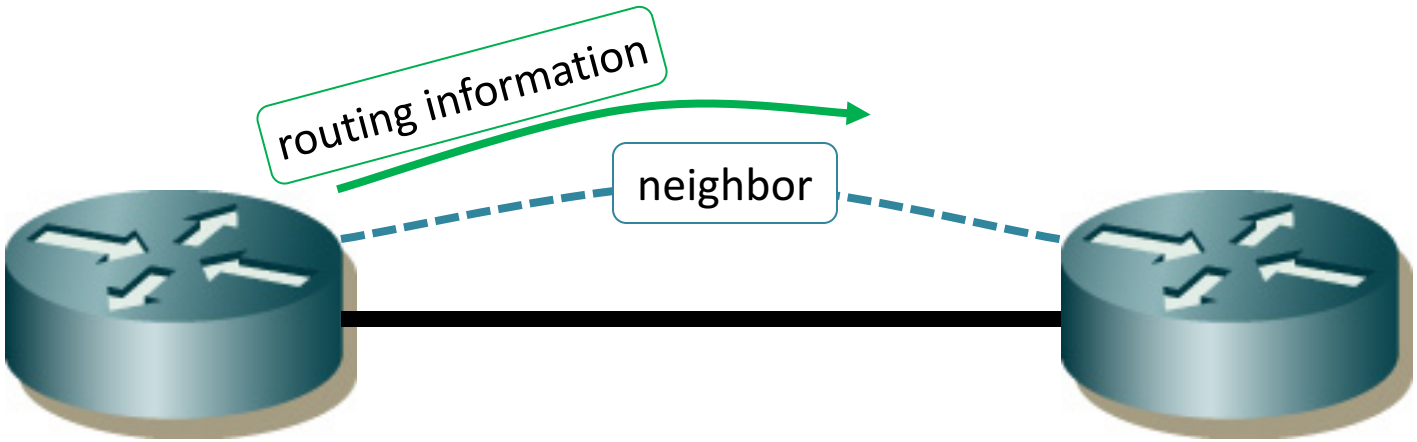
still we need pMTUd

- MSS hack work only for TCP
  - UDP, and any other protocols
- do not filter ICMP error messages without consideration

Protecting Routing

- To keep your network working
  - as you designed
  - as you configured
- Static Routing
  - mostly depends on design
- Dynamic Routing
  - possibility of remote attacks

Routing Protocol



- Routers exchange routing information over a neighboring relationship.

Threat Model for Routing

- Neighboring Relationship
  - Unexpected Neighboring
  - Shutdown by Someone else
  - Spoofed Neighbor
- Routing Information
  - Propagation of Wrong Information
  - Unintended Routing Policy
  - Hit a Hardware Limitation

OSPF Neighbors

- Establishing a relationship among trusted neighbors only
- Disabled by default
  - Especially on a link to other parties (IX,customer)
    - to avoid unexpected neighbors
    - if you have to enable on these links, use ‘passive’ feature
  - Enabled where it is needed like backbone
- Authentication
  - MD5 authentication (OSPFv2, RFC2328)

OSPF md5 configuration

cisco

```
interface <interface_name>
 ip ospf authentication message-digest
 ip ospf message-digest-key <keyid#> md5 <md5_key>
```

juniper

```
protocols ospf  {
 area <area#> {
  interface <interface_name> {
     authentication {
          md5  <keyid#> key “<md5_key>”;
     }
  }
}}
```

BGP4 Neighbors

- Protecting TCP sessions
  - md5 authentication
- Peering with other parties
  - possibility of injection
  - needs more attention about routing information

BGP md5 configuration

cisco

```
router  bgp <as#>
 neighbor <neighbor_ip> password <md5_key>
```

juniper

```
protocols bgp {
 neighbor <neighbor_ip>  {
    authentication-key “<md5_key>”;
 }
}
```

Protecting routing information

- OSPF
  - mostly relies on neighboring
  - IGP should be used for internal purpose
    - should not be used to share routing information with your customers
- BGP
  - routing information is more problematic

critical routing information inside AS

- iBGP neighbor
  - usually loopback interface
  - /32 announcement by IGP
    - the most preferred
- BGP nexthop
  - typical BGP nexthop
    - IX segment
    - peering link
    - customer link
  - route filtering on eBGP sessions
    - needs care about more-specifics