IPFIX Export at IXPs

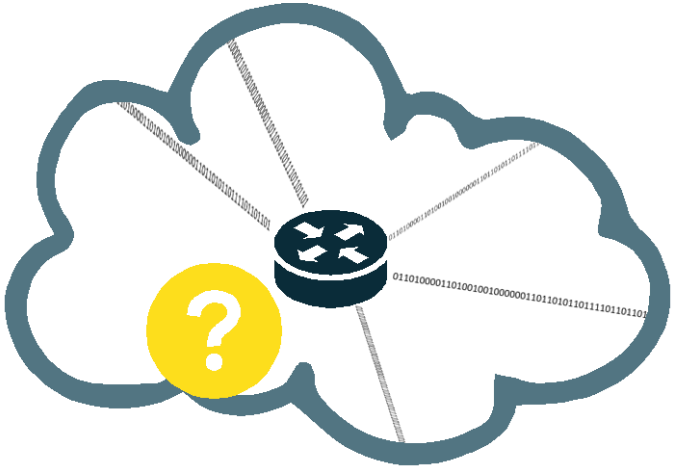
Insights into Your IXP

Thomas King, CTO, DE-CIX
Swinog #37

Motivation

- ➔ Insights in traffic statistics
- ➔ Beyond customer‘s rate limit / Access Port capacity
- ➔ No load on customer‘s router
- ➔ No router configuration needed

DE-CIX FRA



IPFIX Protocol

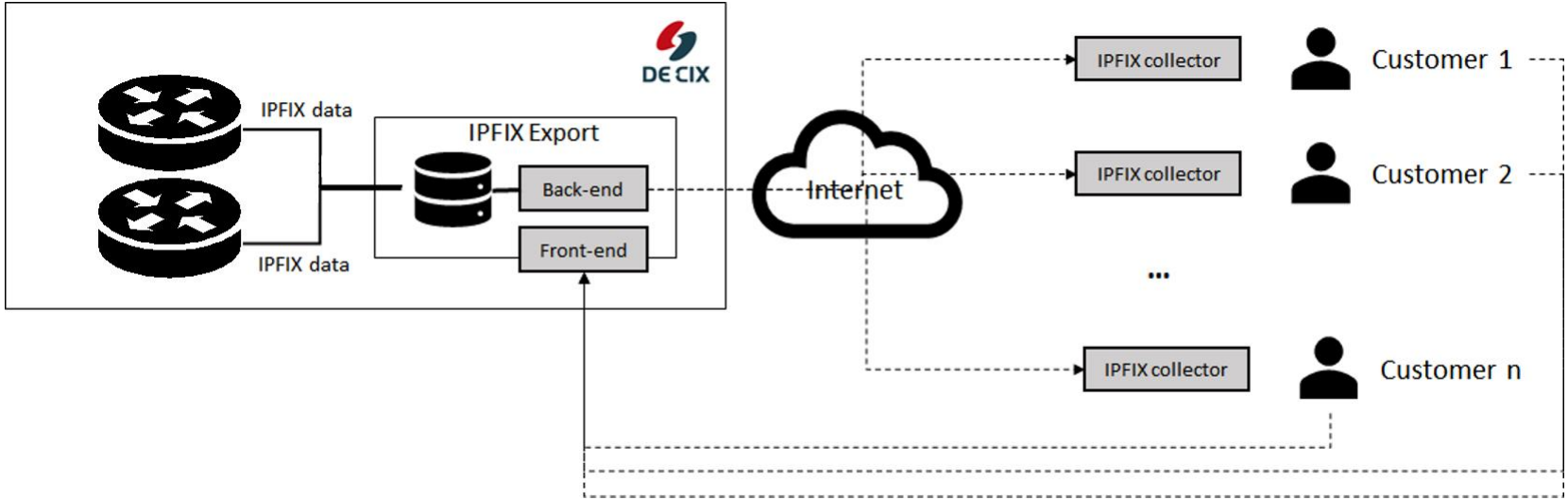
- RFC7011[1]
- Templates
- 491 data fields defined[2]
- Dead and alive timeout

```
▾ Cisco NetFlow/IPFIX
    Version: 10
    Length: 1337
  ▸ Timestamp: Mar  8, 2021 13:06:55.000000000 CET
    FlowSequence: 84
    Observation Domain Id: 0
  ▾ Set 1 [id=257] (14 flows)
      FlowSet Id: (Data) (257)
      FlowSet Length: 1306
      [Template Frame: 31 (received after this frame)]
    ▾ Flow 1
        Source Mac Address: [redacted] ([redacted])
        Destination Mac Address: [redacted] ([redacted])
        SrcAddr: [redacted]
        SrcMask: 0
        DstAddr: [redacted]
        DstMask: 0
        SrcAddr: ::
        DstAddr: ::
        Protocol: TCP (6)
        SrcPort: 55483 (55483)
        DstPort: 38164 (38164)
      ▸ [Duration: 0.000000000 seconds (milliseconds)]
        Octets: 68
        Packets: 1
      ▸ TCP Flags: 0x0000
    ▸ Flow 2
```


[1] https://tools.ietf.org/html/rfc7011
[2] http://www.iana.org/assignments/ipfix/ipfix.xhtml

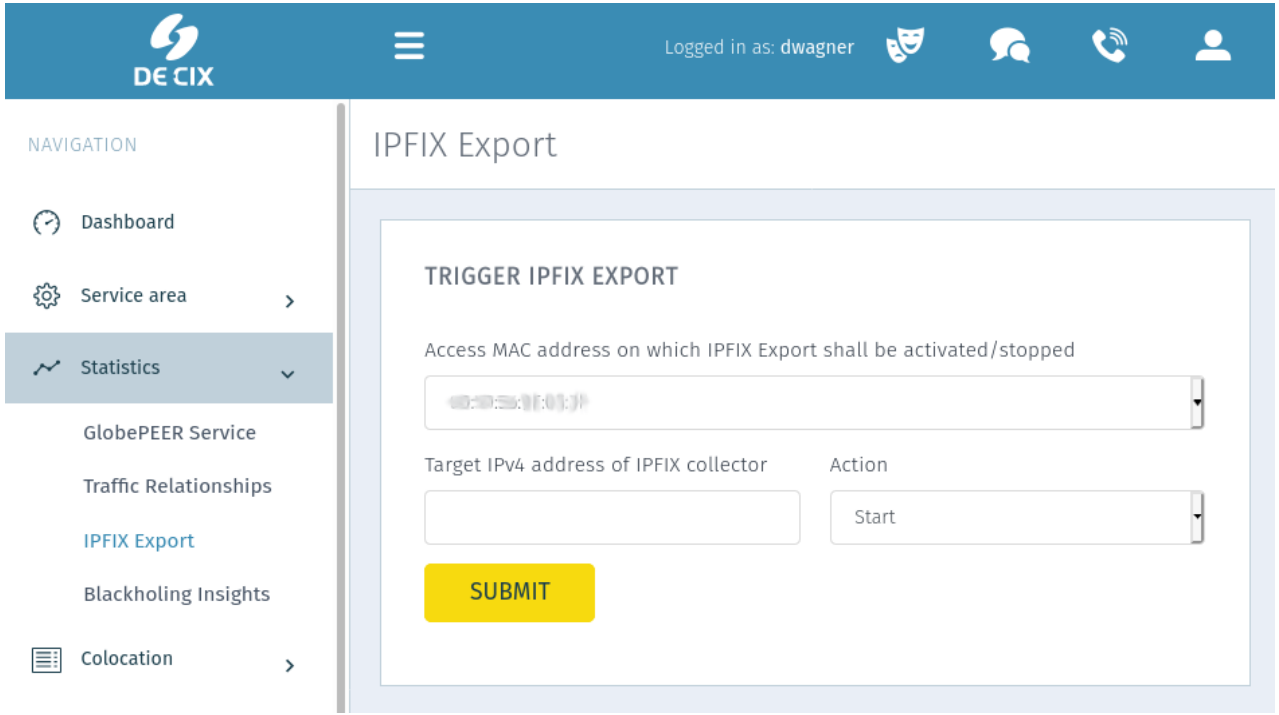
Architecture

- → Packet sampling rate 1:10k
- → Dead timeout: 15s, alive timeout 60s



Front-End[3]

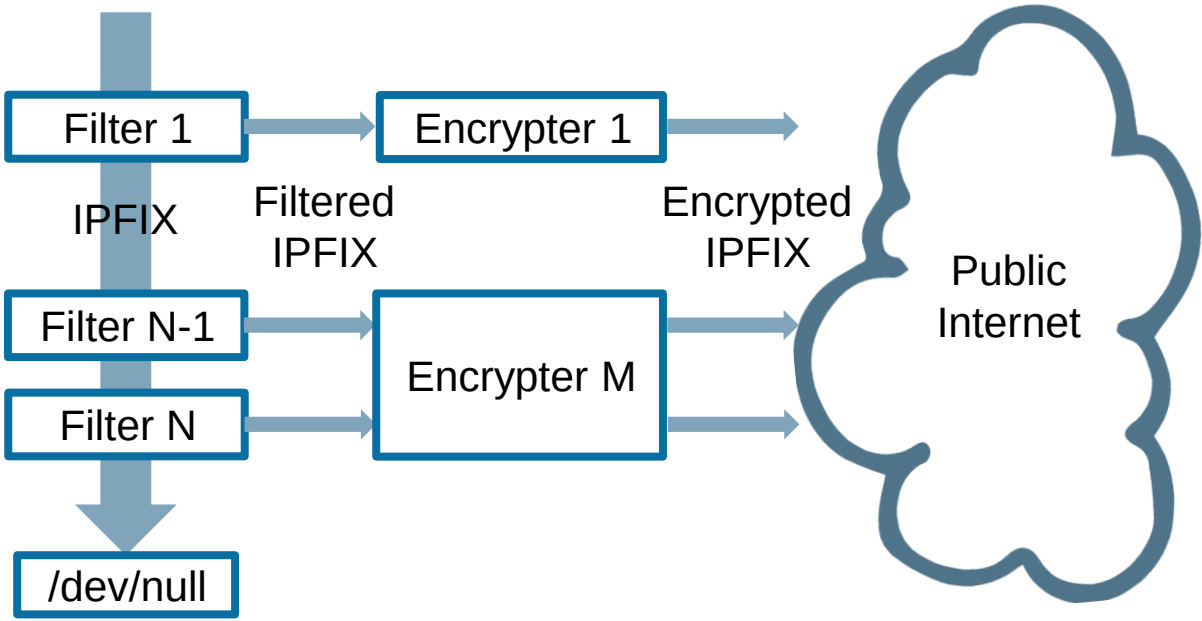
- Customers choose from their MAC addresses
- Enter any target IP
- Select start/stop



[3] https://portal-beta.de-cix.net/statistics/ipfix-export

Implementation Challenges

- ➔ Incoming:
  One large IPFIX stream
- ➔ Outgoing:
  N filtered IPFIX streams to M target IP addresses
- ➔ Need for new IPFIX stream creation



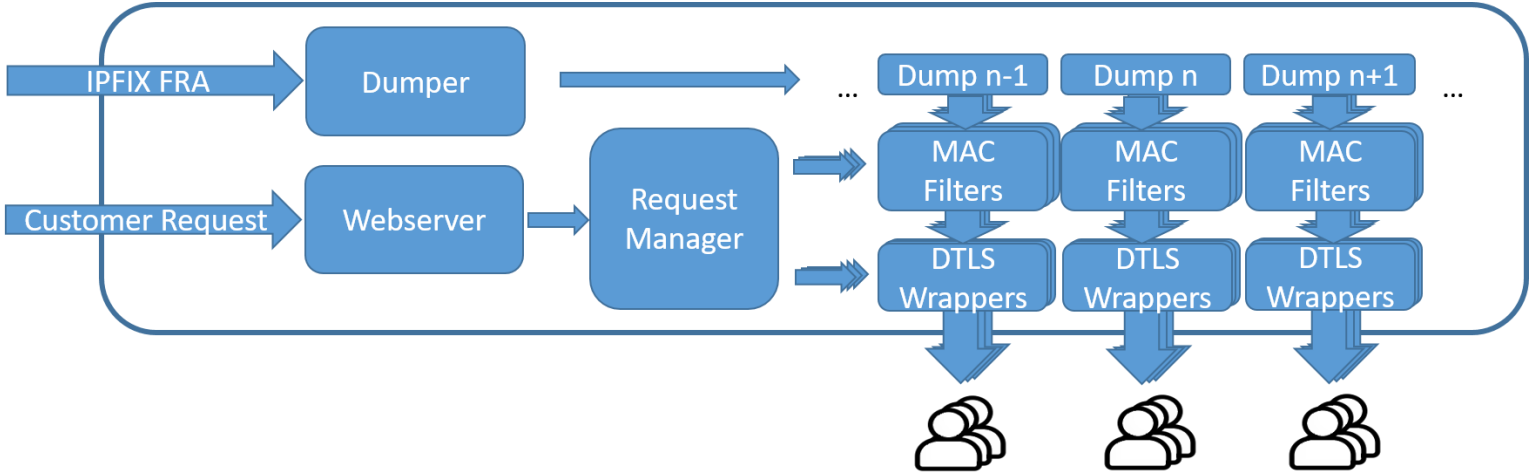
Design Space

- **1 Vermont[4] instance**
  - Config contains filters for every MAC address
  - Output redirected to encrypter on demand

[4] https://github.com/tumi8/vermont/

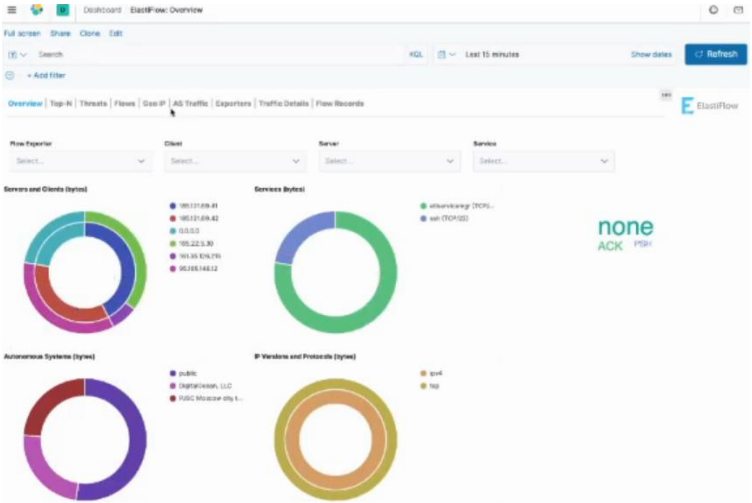
Back-End

- → Dumping + filtering: Vermont
- → No interruption upon request
- → Approx. 1 minute delay



Receiving Data

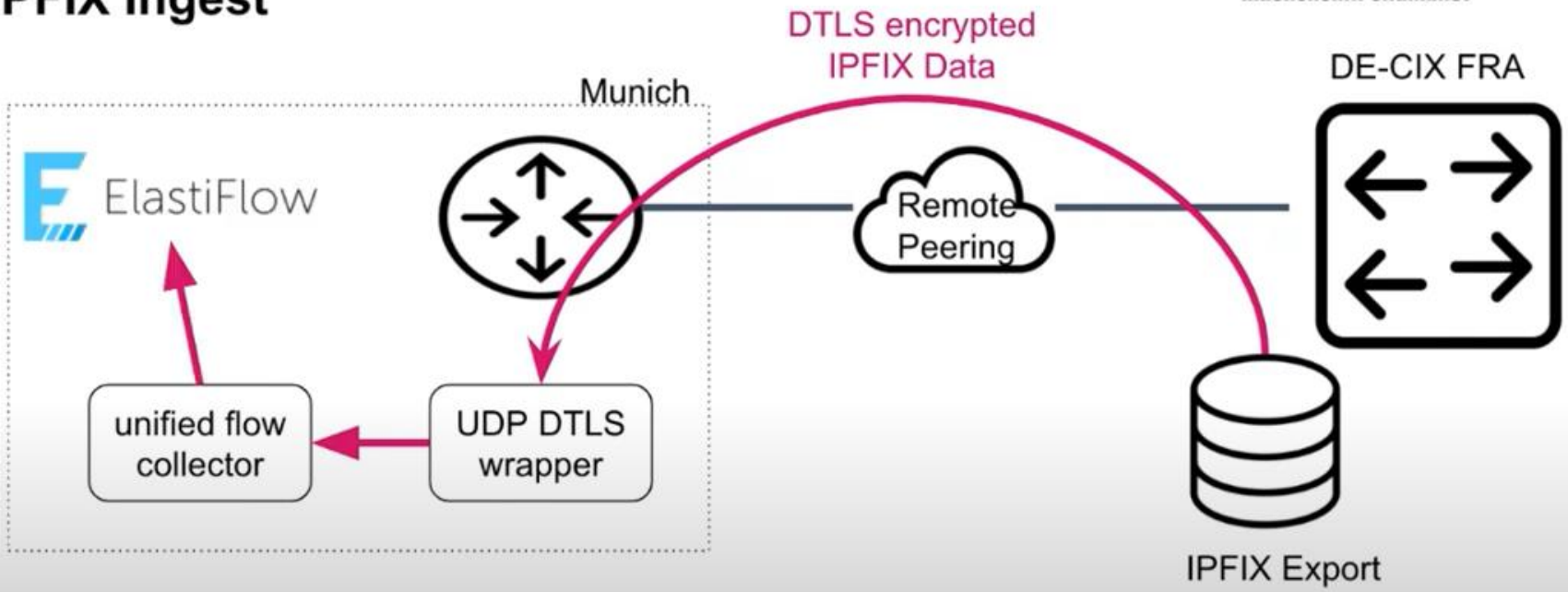
- → Open-source decrypter[5]
- → Pmacct[6]
- → FastNetMon[7]




```
[dwagner@[redacted] ~]$ ./dtls-decrypter --listen [redacted]:2055
--output 127.0.0.1:2055
Listening on [redacted]:2055 (UDP) for DTLS traffic.
Sending decrypted traffic to 127.0.0.1:2055 (UDP)
Packets received: 1368  Bytes received: 335847
```


[5] https://github.com/de-cix/udp-dtls-wrapper/
[6] http://www.pmacct.net/
[7] https://fastnetmon.com/

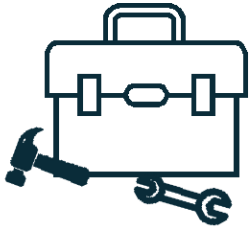
IPFIX Ingest



https://youtu.be/HS-PkYJhT0A

Planned Enhancements

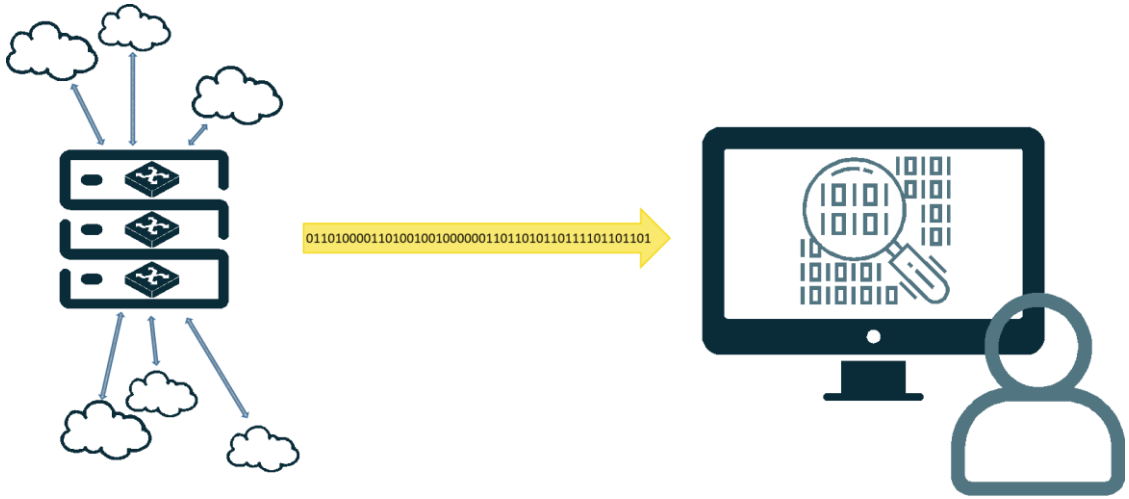
- ➔ Configure transport port
- ➔ Overview of running exports
- ➔ Export via IPv6
- ➔ Support other DE-CIX Locations (e.g. MUC, NYC)
- ➔ Webinar [8] – We already have that! ☺



[8] https://www.de-cix.net/de/about-de-cix/academy

Summary

- → Self-Managed IPFIX collection
- → Sensible data encrypted
- → Analysis with own tools
- → Free beta service



Thank you for your attention!

Any questions?