# Directions for Signaling (for traffic) between Application and Network

draft-eckert-int-flow-metadata-framework-<latest>
draft-choukir-tsv-flow-metadata-encoding-<latest>
draft-zamfir-tsvts-flow-metadata-rsvp-0<latest>
draft-martinsen-mmusic-malice-00
draft-wing-pcp-flowdata-<latest>

Amine Choukir, Charles Eckel, Toerless Eckert, Reinaldo Penno, Pal Martinsen, Anca Zamfir, ..

IETF87, July/Augus 2013

# Agenda

- Motivation

  What: Use-cases

  How (today): Use cases via ACL…DPI (the problem)

- Proposed Solution Framework

  Metadata Signaling: Concept

  Example/Tentative Attributes

  Loose coupling options to enable services

  Support/leverage variety of transport protocols – no “one-protocol-fits-all”

- Proposed (initial) IETF goals

  Propose to start with three important ones (RSVP, ICE, PCP)

  IETF procedure to define/register attributes

  Common encoding proposal

  Open (but not considered) to include other elements of workflow (policy rules etc..)

# Reality

### Applications

- Best-Effort experience often far from “best”.
- Getting value added services from network is difficult and overall seldom adopted – variety of protocols/mechanisms/market-segment differences.

### Operator/User

- Difficult and complex to gain visibility into traffic
  what uses the network and what it needs.
- No easy and ubiquitous mechanisms to provide differentiated experiences for traffic.
- Wide range of applications requiring it:
  - Pervasive Video/Collaboration
  - Applications with extensive use of rich media
  - Business critical application

# Use-cases

- Enterprise / Industrial / SMB:

  Operational Simplicity! "zero touch benefits"

  Many Applications: Video (Skype, UC, Webex), Business-specivi (DB, …) scavenger (social networking,..)

  Visibility: Analysis, Planning

  Many Actions: QoS / CAC, Routing: 3G/4G, Managed (L3VPN), OTT (IPsec), Monitoring/Performance

- SP: enable additional revenue services … competitive/differentiated service
  - Managed Services Edge (to enterprises) – PE, (managed) CE

    Everything the enterprise is asking/paying for, Bandwidth on demand, load-balancing

    Same/better as what the Enterprise would do on CE/PE - Autoconfiguration of QoS
  - "More than flat broadband access pipe" (DSL, Cable, 3/4G)

    Prioritize Apps in 3G/4G, 3G-to-WiFi- bypass for specific applications, Hotspot service differentiation

    Bandwidth on-demand for specific sessions

    Low delay for gaming,

    Differentiated assured bandwidth for TV streaming from SP or OTT

# TODAY

- Toolset: ACLs/DPI

  Application/Device-User-Group visibility and control

- ACL:

  IP-address,/”Port-range”/ACL management, coarseness

- DPI:

  Encryption, Authentication

  Dynamic / abesent information

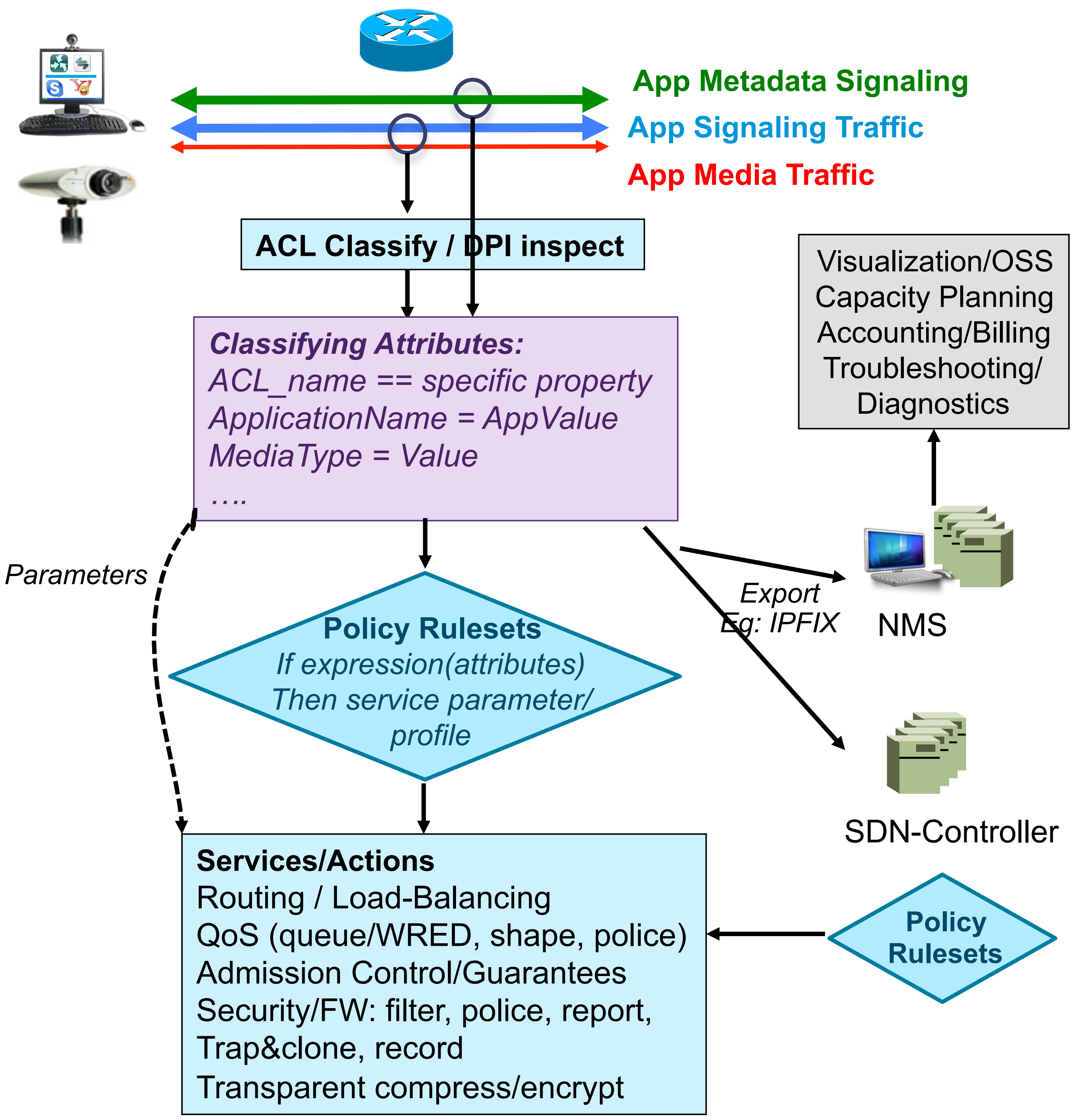
  Agility of media/signaling format

  Incongruent paths for signaling and media

  Unreliability due to heuristics

- Proposal: explicit signaling of attributes

  Business-relevant == useful in policy rulesets
  and/or Visualization/OSS



App Metadata Signaling

App Signaling Traffic

App Media Traffic

**ACL Classify / DPI inspect**

***Classifying Attributes:***
*ACL_name == specific property*
*ApplicationName = AppValue*
*MediaType = Value*
*….*

*Parameters*

**Policy Rulesets**
*If expression(attributes)*
*Then service parameter/ profile*

*Export*
*Eg: IPFIX*

NMS

Visualization/OSS
Capacity Planning
Accounting/Billing
Troubleshooting/
Diagnostics

SDN-Controller

**Services/Actions**
Routing / Load-Balancing
QoS (queue/WRED, shape, police)
Admission Control/Guarantees
Security/FW: filter, police, report,
Trap&clone, record
Transparent compress/encrypt

**Policy Rulesets**

# Goal !

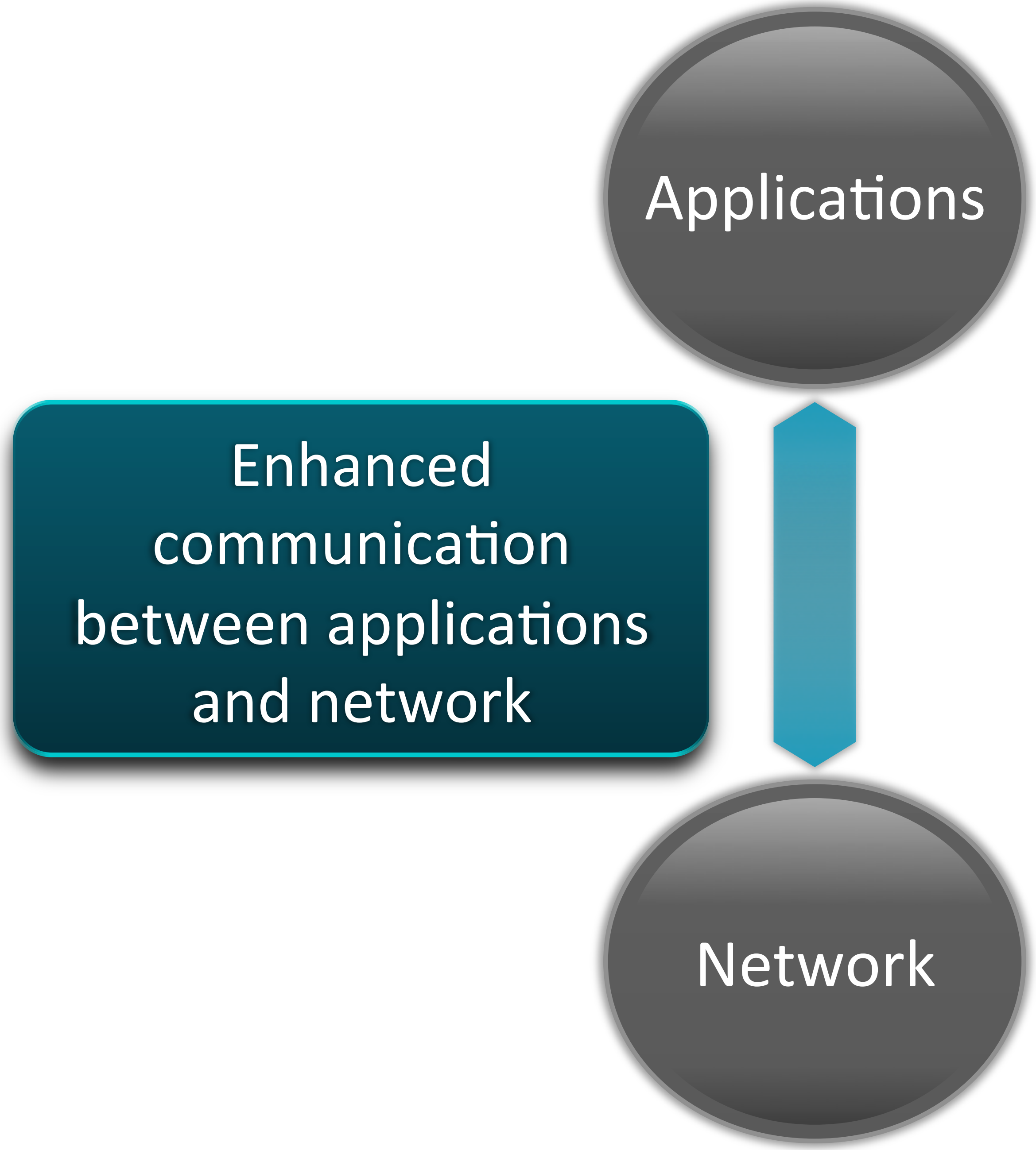
**Application**

- Get appropriate ("better") treatment from network by exposing characteristics of traffic.
- Use protocol independent common data model.
- Let "Operator" figure out what appropriate is.
- Request services explicitly if desired

**Network operator/User**

- Comprehensive visibility into traffic in the network. Presence, requirements, performance.
- Easy policies to differentiate application experience across services in the network:
- QoS/CAC, Routing, Monitoring, Security, …

# Metadata Signaling
## Overall concept



- Application signals
  - For traffic flows - initially 5-tuple
    (future: 4-tuple, tuple with flow-label, …)
  - Business/workflow relevant “classification” attributes (“metadata”)
    - attributeX=valueX, attributeY=valueY,…
    - Protocol independent semantic, well defined/registered
    - Encoding optional cross-protocoll
      (one for TLV, one for textual protocols ?)
- *Tentative features*

  *Signaling for sent/received flows*

  *Authentication (app to network)*

  *NAT/FW traversal*

  *Signaling for network feedback*

  *Support for wide range of transport protocols*

  *Proxy support: in-sender/in-network: home-gateway, CE/CPE/AN*

  *Add/change/delete attributes (eg: authentication, network specific service-request attributes).*

  *Enable Application not supporting signaling themselves (not ideal)*

# Example/Tentative attributes

- Bandwidth indications

  MinBandwidth, MaxBandwidth: Sustained (>> queueing time) bandwidth range for traffic flow.
  Inelastic flows MinBandwidth = MaxBandwidth.

  BandwidthPool: GUID for flows sharing same bandwidth, …

- Traffic Class "QoS" indications

  Rfc4594-dscp: "My app-developer thinks this traffic best matches this DSCP from rfc4594"

  TCL – Traffic Class Label: structured string - *category.application.{adjective{.adjective…}}*

- Acceptable path properties

  DelayTolerance, LossTolerance

- Application Identification *important! Known IETF rathole (DPI) – this is not DPI – application-self-assigned*

  "AppId" (RFC6759): Eg: L4-port or vendor (PEN) specific AppID (from AppVendor or MarketVendor)

  AppURI: <appdomain>.com/<appname

- …

- Subscriber-ID, (local-significant) User-ID, Device-Name/ID

- "Session-Detail-Record" metadata (caller/calling-#/URI), Codec-information ("media-type"), …

# Service instantiation
## through loose coupling

- Classical approach

  Per-service protocol/signaling,  Request/reply

  Adoption/Flexibility/Support issues

- Loose coupling can solve this problem

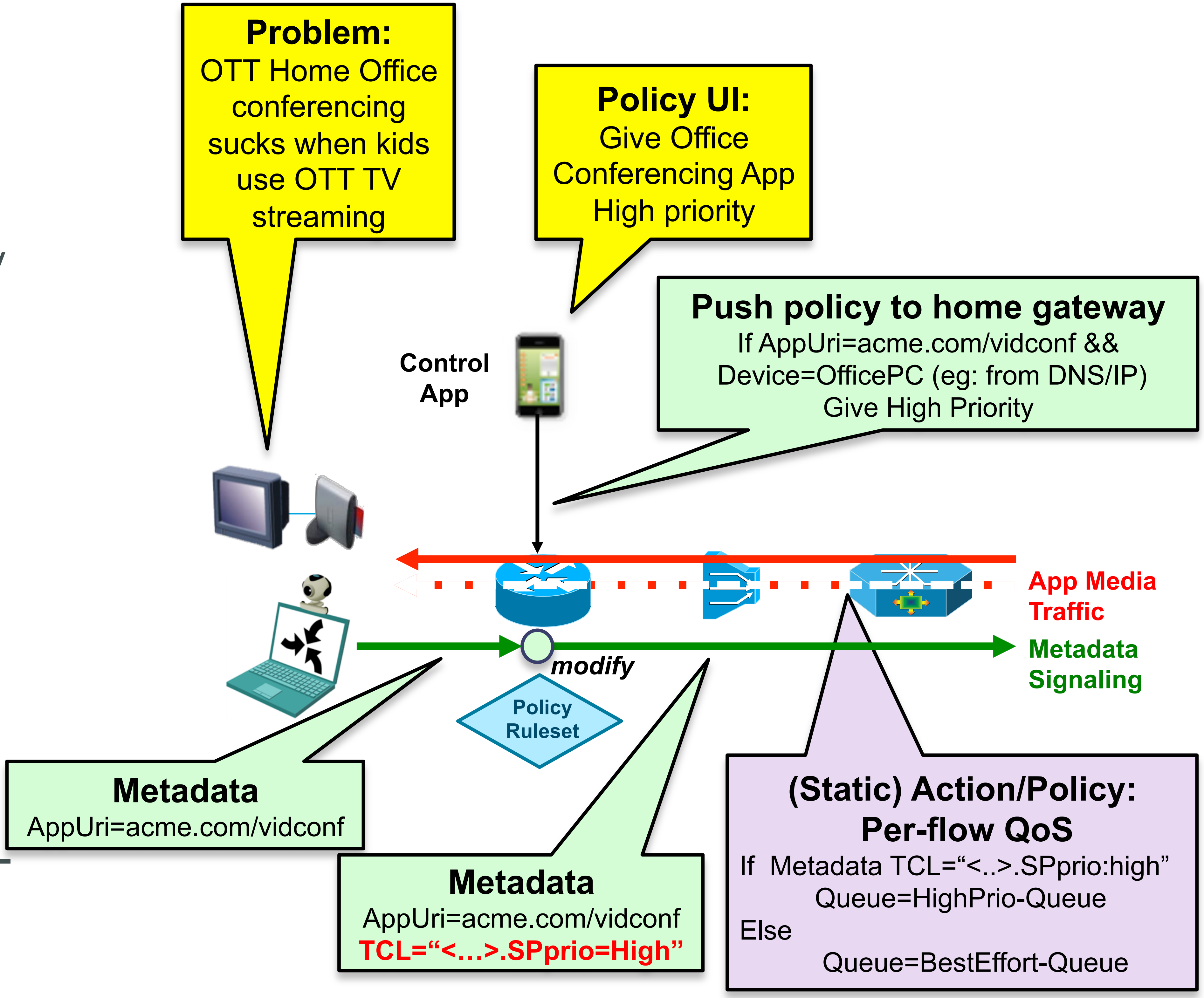
  Applications can not know about all possible network services. Should only worry about describing their traffic

  Different services on different networks

  Network Services still being explored (eg: bandwidth on demand). Standardization premature.

- Example how loose coupling via policy-rules can solve this problem

  Policy could be pushed into various places (Home Gateway, AN, …)



**Problem:**
OTT Home Office conferencing sucks when kids use OTT TV streaming

**Policy UI:**
Give Office Conferencing App High priority

**Push policy to home gateway**
If AppUri=acme.com/vidconf &&
Device=OfficePC (eg: from DNS/IP)
Give High Priority

**Metadata**
AppUri=acme.com/vidconf

**Metadata**
AppUri=acme.com/vidconf
**TCL="<…>.SPprio=High"**

**(Static) Action/Policy:**
**Per-flow QoS**
If  Metadata TCL="<..>.SPprio:high"
Queue=HighPrio-Queue
Else
Queue=BestEffort-Queue

# Target IETF goals

- Enable use-cases
- Support beneficial signaling protocols via metadata attribute signaling
  - Today: No one-size fits all: RSVP, STUN/ICE, PCP, … (more possible …NSIS, XML/JSON/HTTP/…)
  - Reduce protocol options in future ?!
- Evolve from protocol definition to data-model approach
  - Applications should only care about the data (attributes), not (transport) protocols
    - SDK, Middleware (eg: browser) can take care of the protocols!
- Offer cross-protocol common encoding of attributes (first round: for binary protocols)
- Establish rules to Define / Standardize / Register relevant attributes for traffic
- Support (ultimately) all attribute signaling options:
  - **Informative**: application to network
  - **Advisory**: network to application feedback
  - **Service-Request**: via common attributes

# Signaling Protocol diversity
## No “One Size fits all”

- “binary”: **RSVP,** NSIS, **PCP, STUN/ICE**, … PIM/IGMP, what else ?,
  “textual/encoding”: HTML/XML, XMPP, JSON, …
- How easy is it to send/receive for applications ?
  - Text better ? Binary more commonly used, “over TCP” most easy ? Over UDP necessary ? Raw-IP sucks ?
- How easy is it for the network to interact ?
  - Router alert is standard (but practice suxx ?), simple signature inspection easy ? direct/anycast addressing
- How lightweight, how high can it scale ?
- How can it pass NAT/Firewall ?
- Can it support TCP and UDP app traffic *(maybe even multicast ?)*
- How much can it directly signal to routers/switches “onpath” ?
- End-to-end vs. “edge-only” signaling ?

# Signaling Protocol diversity

## No "One Size fits all" – conclusions:

- Protocol choice determined by deployment situation:
  - RSVP "heavyweight" – scales to "video/media" flows but not "large" number of flows. Supports UDP/TCP,even multicast
    - Good in enterprise !?
  - STUN/ICE passes through 3rd party NAT/FW, could be implemented very lightweight in routers, supports end-to-end
    - General purpose "across internet" (b2c, b2b), more lightweight enterprise future option ?
    - Already relied on heavily for address selection (primary ICE use-case), Can amend end-to-end session-layer signaling
  - PCP supports explicit negotiations of services already, focusses on edge-signaling
    - Ideal starting point for residential sub-SP signaling cases ?

  *These protocols look like a good starting point!*

- Information to signal from/to network quite independent of transport protocol!
  - Same metadata attributes make sense across all protocols!

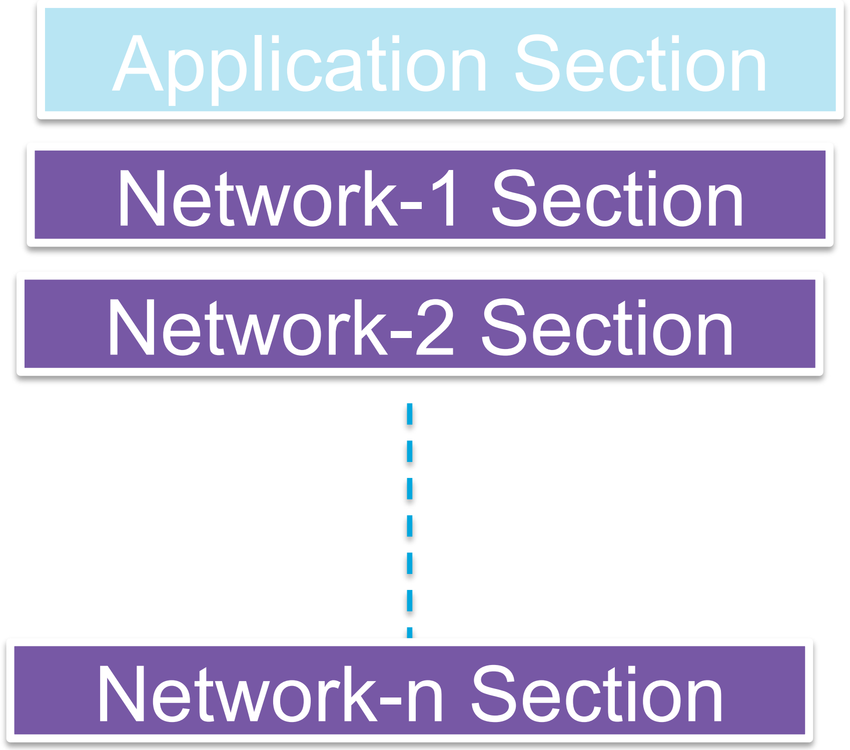
# Attribute registration / definition

- Registration: IPFIX (RFC5101, 5102/5102-bis)
  - Best IETF available registration mechanism !?
  - Supports IETF-process/ IANA registry option AND vendor specific (via PEN)
  - For IETF process defined attributes,
    - “draft-ietf-ipfix-ie-doctors” proposes a process/review rules for attribute definitions.
- Definition
  - Attributes can be defined by any working group.
    - Protocol independent working groups desirable ?
  - What details are necessary/sufficient to permit app-developers to provide attributes consistently ?

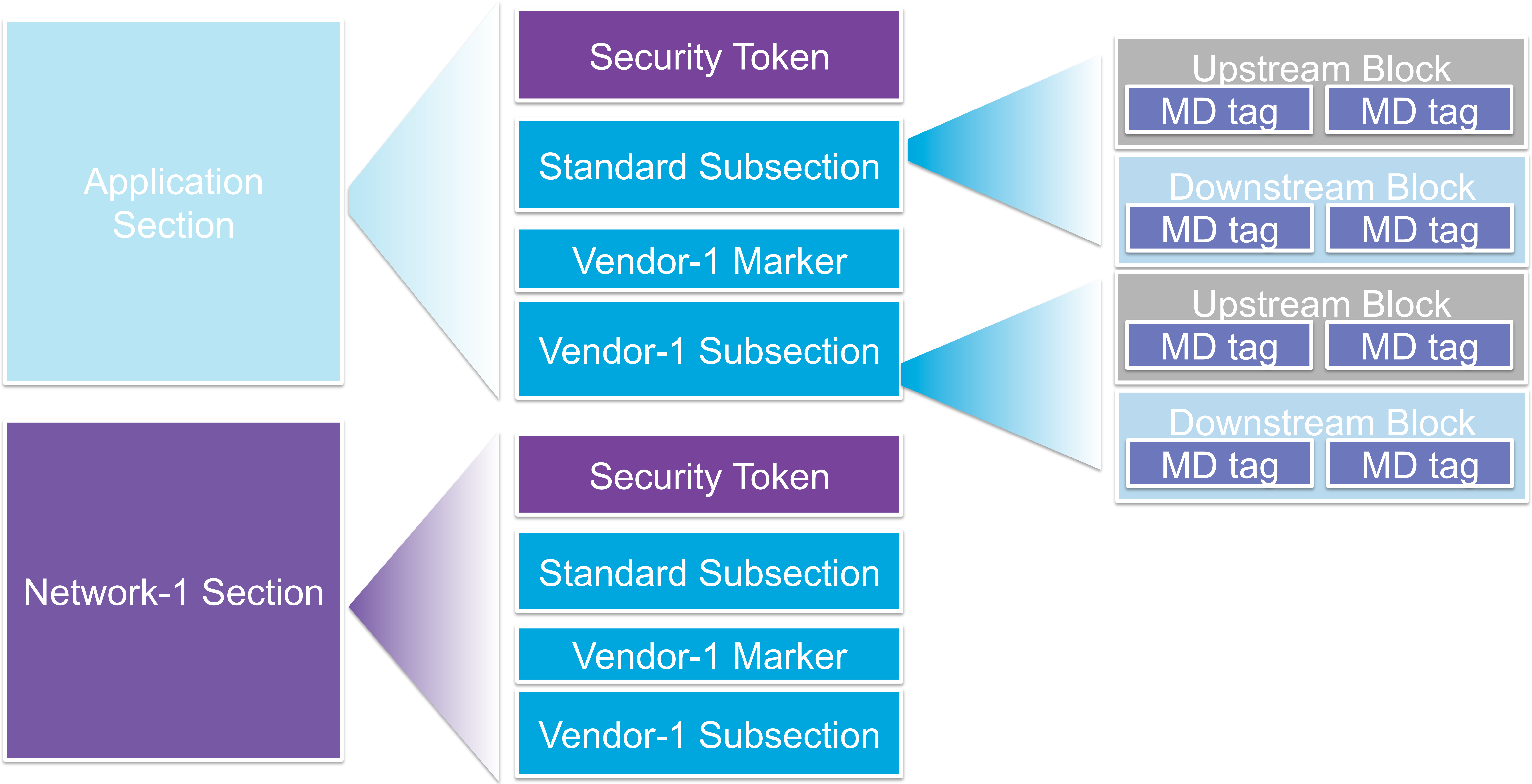
# Attribute Encoding Goals

## draft-choukir-tsv-flow-metadata-encoding-<latest>

- Protocol independent for “binary” protocols.
- TLV-encoding for IPFIX style attributes
  - Standard and vendor specific namespaces
  - Simplified: No templating (only useful for export, not signaling)
  - Compact: (eg: every PEN only sent once)
  - Upstream and downstream (optional) signaling
  - Extensible
  - Allow tags to be secured on a per producer basis
  - Encodes the producer precedence
- *Adoption of this encoding in targeted protocols in various stages (not fully embodied in latest PCP, MALICE drafts)*



# Attribute Encoding



Application Section

Security Token

Standard Subsection

Vendor-1 Marker

Vendor-1 Subsection

Upstream Block

MD tag MD tag

Downstream Block

MD tag MD tag

Upstream Block

MD tag MD tag

Downstream Block

MD tag MD tag

Network-1 Section

Security Token

Standard Subsection

Vendor-1 Marker

Vendor-1 Subsection

# The End