ENABLERS FOR TRANSPORT LAYER PROTOCOL EVOLUTION

draft-mihaly-enablers-for-tlp-evolution-00

Attila Mihály
Szilveszter Nádas (presenting)

Ericsson Research

TP AND TP FRAMEWORK EVOLUTION IS SPEEDING UP

- Experimenting with Transport Protocols using a user space implementation
  - App-speed evolution, fast deployment, less standardization
  - Mainly over UDP
  - E.g. QUIC,
  - E.g. SPUD enables similar solutions
- Addressing middlebox issues
  - Assuming TCP wire format and given app protocols – ossification
  - E2E encryption and some applications (e.g. gaming) already enforcing them to let UDP pass

- (taps, spud, IAB Stack Evolution Program, tcpm, QUIC, …)

SCOPE – TRANSPORT PROTOCOL FRAMEWORK

- Put requirements on TP framework to achieve
  - a healthy eco-system
  - fast TP evolution
- Investigate the effect of accelerated TP evolution
  - E.g. what happens if many app developers implement their own TP?
  - How is it possible to keep the stability of Internet in this case
- Ideas to meet these requirements

- Not in scope: features of the TPs.

REQUIREMENTS – CONTROL

- Enforce expected TP behavior (2.1)
  - Implementations might be buggy or malicious on purpose (e.g. CC aggressiveness)
  - Protect other flows of the same user
  - Protect other users
  - Example behavior to be enforced: congestion control, MTU, packet pacing
- Allow the path influencing TP selection (2.4)
  - The path may offer enhancement/cooperation/blocking of some TPs
- Ensure user/OS control (2.9)
  - What TP is selected (for an app)
  - Preferred resource sharing (between apps and app streams)
  - Communication to middleboxes (at lease the ones the user has agreement with)

REQUIREMENTS – ACCESSIBILITY

- Apps shall be able to access available TPs (2.2)
  - Shall be possible to select by apps
  - Shall be possible to insert a new TP into transport protocol selection frameworks
- Allow consistent TP selection (2.3)
  - The selected TP shall be supported by both endpoints and the path
    - (support by path: the packets of the selected TP shall be able to arrive to the other end)

REQUIREMENTS – PRIVACY/ SECURITY

- Ensure confidentiality of end-to-end communications (2.7)
  - If middlebox accesses or modifies the TP then the content shall be protected separately
- Ensure security of end-to-end communications (2.8)
  - Take reasonable effort to avoid 3rd parties exploiting implementation flaws in TP
  - Encryption/ authentication of TP fields is a solution, though that makes it hard for friendly middleboxes to access/modify information

REQUIREMENTS - MIDDLEBOX COOPERATION (2.6)

- Ensure that the access providers can be part of the value chain
  - By either
    - selection between different tradeoffs in local domain QoS/policing most fit for the TP/app
      - (e.g. lower latency vs. higher utilization; higher throughput vs. more stable throughput)
    - further QoE improvement by increasing resource share of critical apps
      - may be fair in the longer run (needs incentives and further consequences)
      - details in *draft-mihaly-spud-mb-communication*
  - These shall be explicit, cooperative, extensible middlebox functions which improve performance, but might have consequences (e.g. economic)
  - It shall be possible for the end-hosts to opt out (and get a reasonable default handling)
  - Different levels of trust shall be possible → different solutions (from hiding everything to accessing content)

REQUIREMENTS – PERFORMANCE (2.5)

- The framework should not result in (significant) degradation of performance characteristics when achieving other requirements
  - E.g. low setup latency, throughput
  - Especially long signaling conversation shall be avoided
- Valid for the common case, some exceptional cases are possible
  - E.g. downloading and storing a TP before the first session

IDEAS – COVERED BY SPUD INITIATIVE (OUR UNDERSTANDING)

- Substrate Protocol for User Datagrams (SPUD)
  - In-band channel/protocol for Middlebox communication
  - Explicit communication and behavior
  - Potentially authenticated and/or encrypted messages to middleboxes
    - This encryption is not the same as the E2E TP or object encryption

- We think that the SPUD initiative is a very important piece of the puzzle to achieve a healthy ecosystem

IDEAS – TRUST AND ENFORCEMENT ISSUES

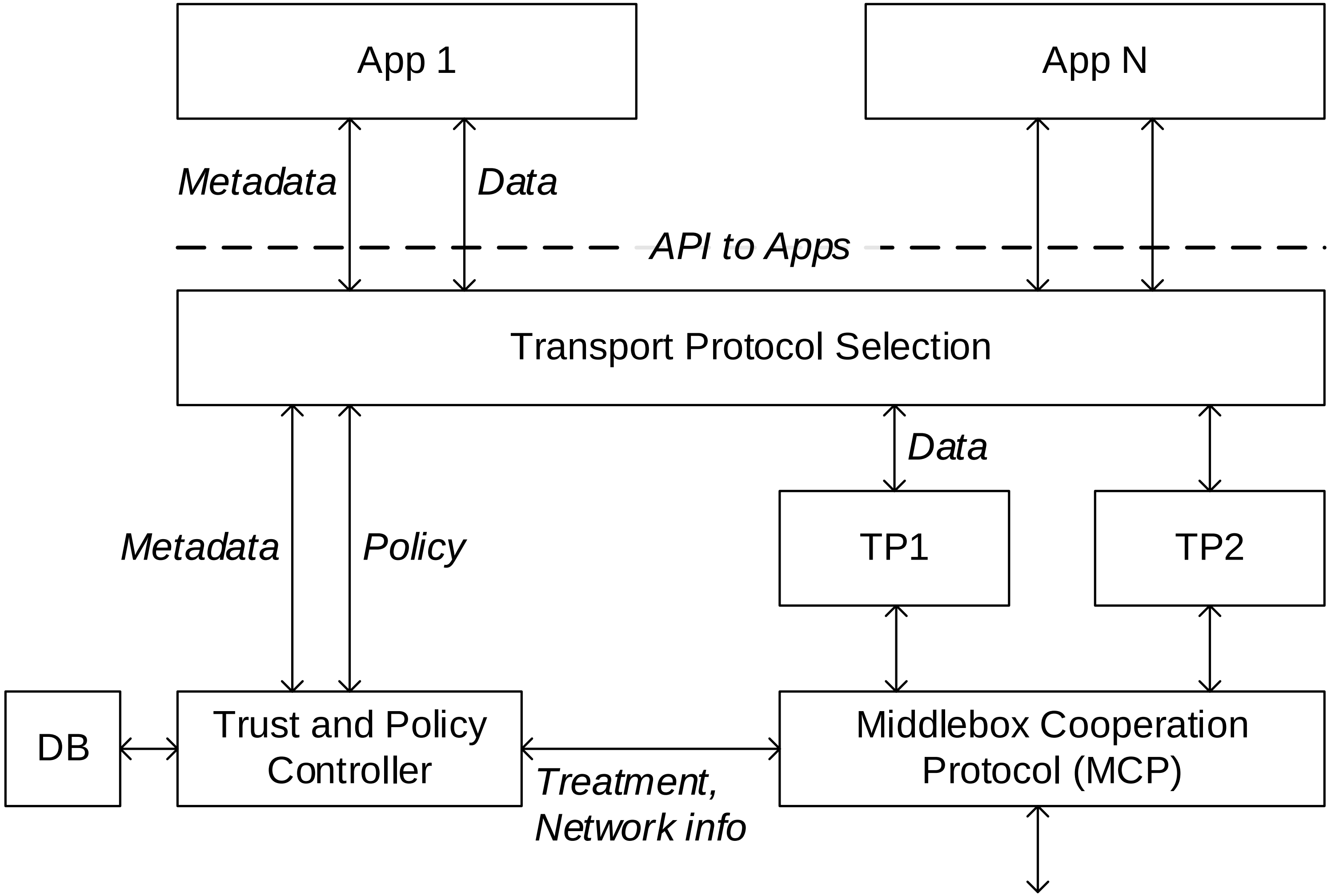
- Within the device of the end user
  - Controlling resource sharing and CC aggressiveness
    - Might require that congestion detection is visible for control functions
    - Might require policing solutions in end-host
    - Might communicate the CC flavor used
  - Middlebox communication
    - what can be communicated to a MB, with what authentication keys?
- Between end-hosts and Middleboxes
  - What authentication keys can be used for a given communication?
  - Who can decode different parts of the communication?
    - e.g. metadata, content, TP header
  - What is the possible consequence of a middlebox communication?

TRUST AND ENFORCEMENT (CONTINUED)

- Who shall control these
  - OS/App store?
  - Network vendor?
  - User?
  - Community database
  - Etc?
- All have reasons to control, see some examples in following slides

TRUST AND POLICY CONTROLLER AND MIDDLEBOX COOPERATION

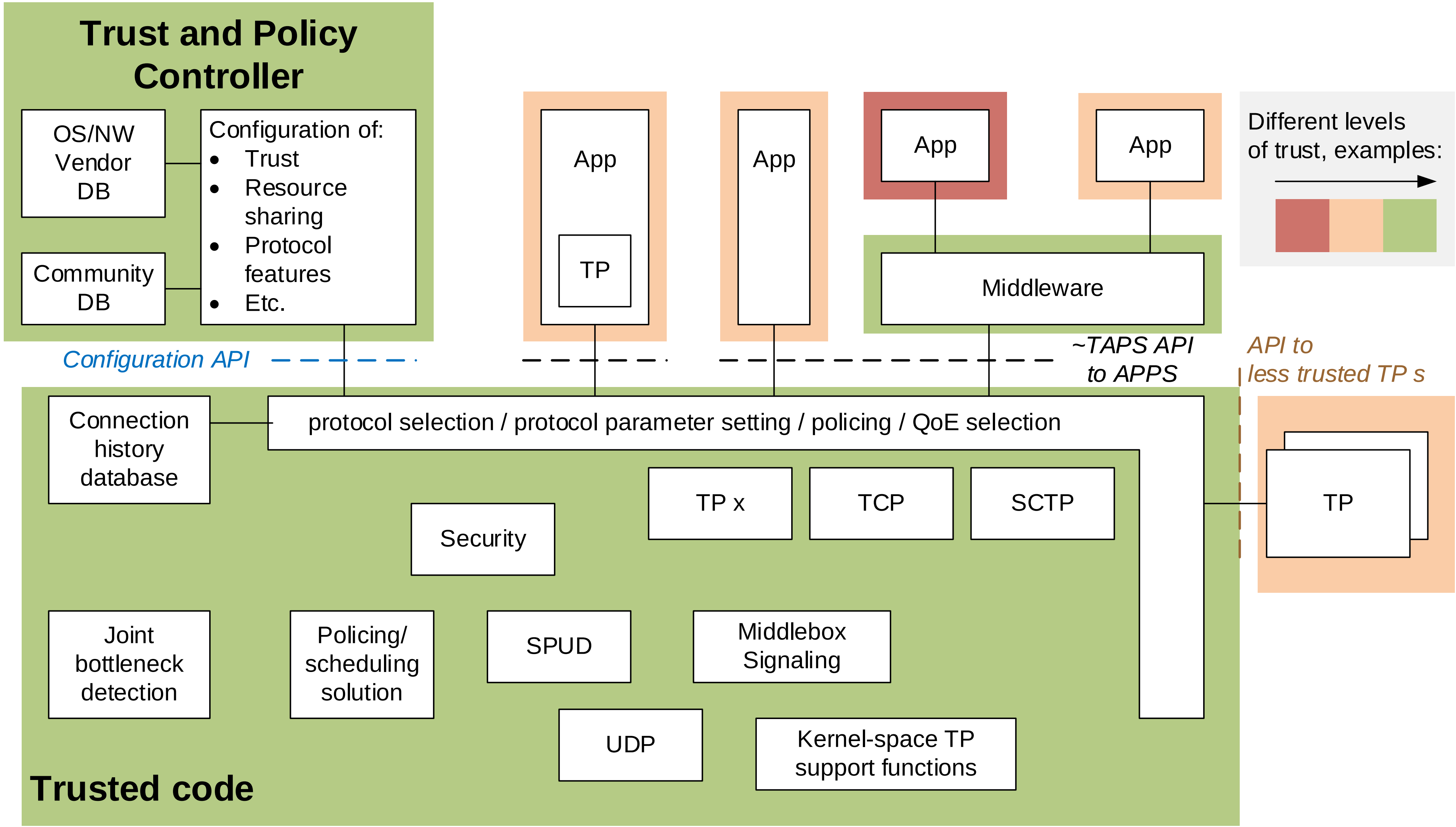
EXAMPLE


App 1
App N
Metadata
Data
API to Apps
Transport Protocol Selection
Data
TP1
TP2
Metadata
Policy
DB
Trust and Policy Controller
Middlebox Cooperation Protocol (MCP)
Treatment, Network info

The Trust and Policy Controller

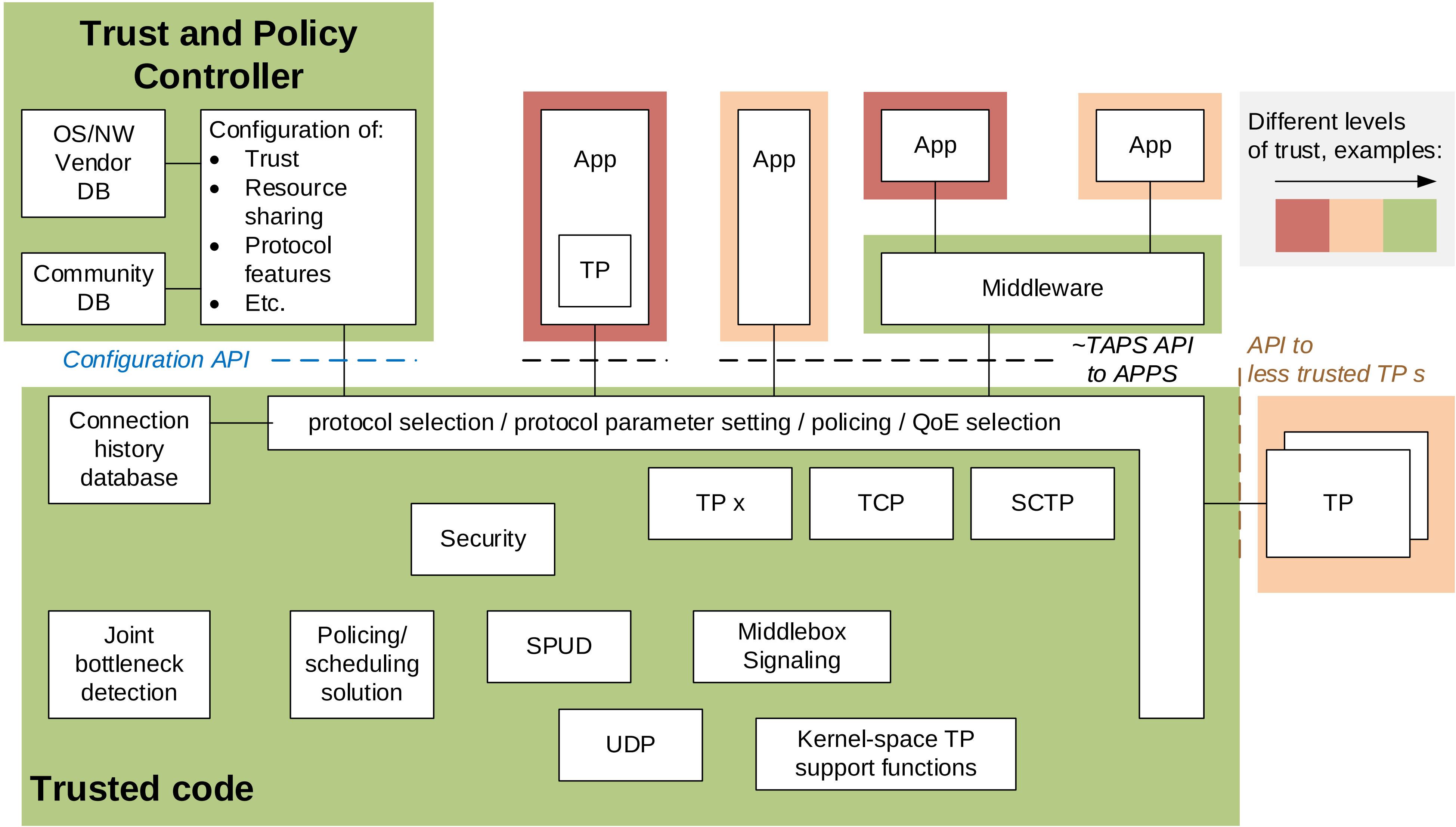
- May receive rich metadata
- Removes privacy sensitive parts
- Determines preferred treatment and other metadata to communicate through MCP using
  - Database
  - User configuration
- May also influence TP selection

TP FUNCTIONS AND APIS IN DEVICE
EXAMPLE

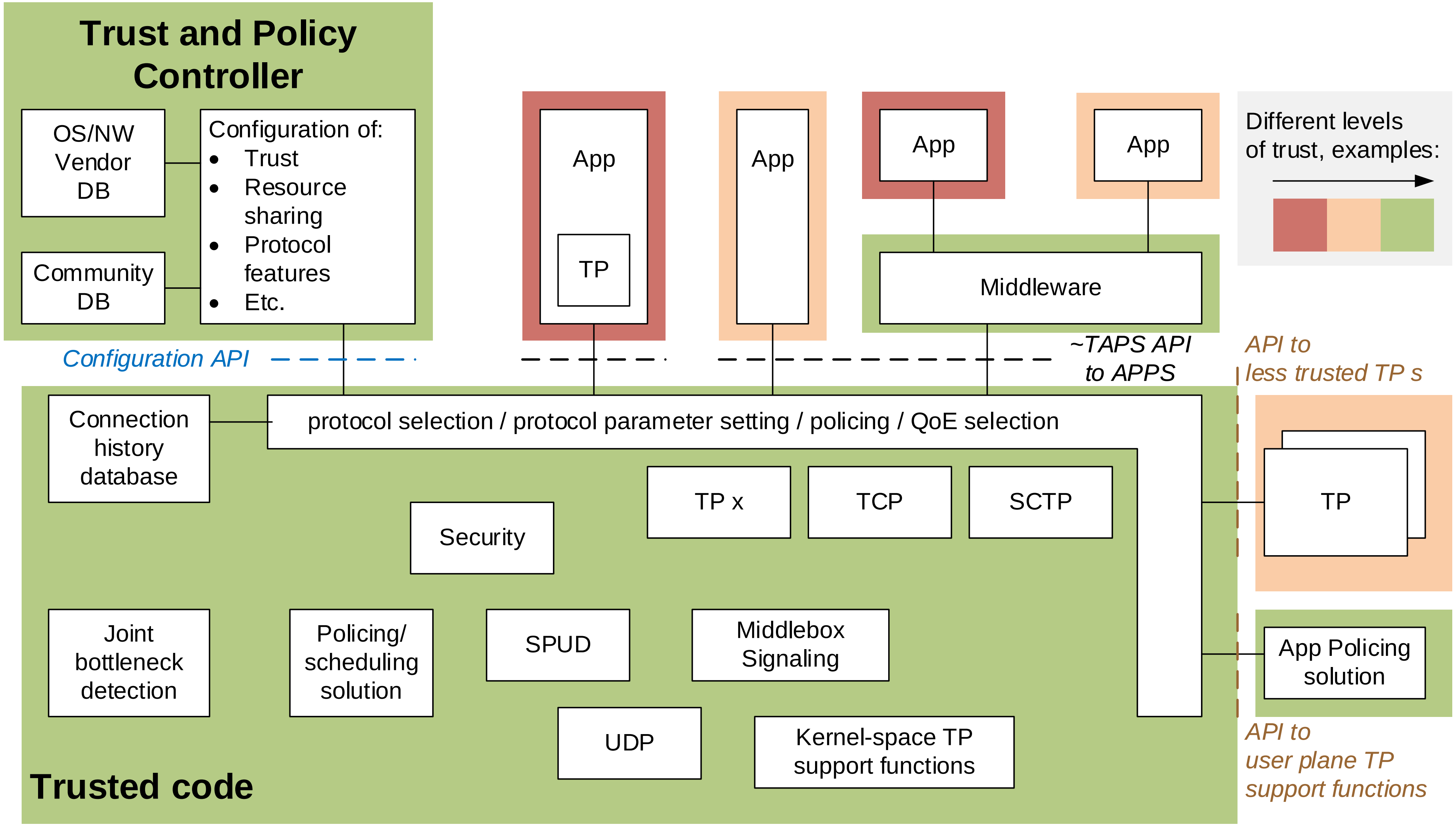


Trust and Policy Controller
OS/NW Vendor DB
Community DB
Configuration of:
- Trust
- Resource sharing
- Protocol features
- Etc.
Configuration API
App
TP
App
App
App
Middleware
Different levels of trust, examples:
🟥 🟧 🟩
~TAPS API to APPS
API to less trusted TP s
Connection history database
protocol selection / protocol parameter setting / policing / QoE selection
TP x
TCP
SCTP
TP
Security
Joint bottleneck detection
Policing/ scheduling solution
SPUD
Middlebox Signaling
UDP
Kernel-space TP support functions
Trusted code

TP FUNCTIONS AND APIS IN DEVICE
EXAMPLE


Trust and Policy Controller
OS/NW Vendor DB
Community DB
Configuration of:
• Trust
• Resource sharing
• Protocol features
• Etc.
Configuration API
App
TP
App
App
App
Middleware
Different levels of trust, examples:
~TAPS API to APPS
API to less trusted TP s
Connection history database
protocol selection / protocol parameter setting / policing / QoE selection
TP x
TCP
SCTP
Security
TP
Joint bottleneck detection
Policing/ scheduling solution
SPUD
Middlebox Signaling
UDP
Kernel-space TP support functions
Trusted code

TP FUNCTIONS AND APIS IN DEVICE
EXAMPLE



Trust and Policy Controller

OS/NW Vendor DB

Community DB

Configuration of:
- Trust
- Resource sharing
- Protocol features
- Etc.

Configuration API

App

TP

App

App

App

Middleware

~TAPS API to APPS

Different levels of trust, examples:

🟥 🟧 🟩

API to less trusted TP s

protocol selection / protocol parameter setting / policing / QoE selection

Connection history database

Security

TP x

TCP

SCTP

TP

Joint bottleneck detection

Policing/ scheduling solution

SPUD

Middlebox Signaling

App Policing solution

UDP

Kernel-space TP support functions

API to user plane TP support functions

Trusted code

TRUST AND ENFORCEMENT

- Trust has to be handled even within the device.
- User control shall be “almost invisible” to the end-user during using the applications
- We propose trust and policy controller functions which can do all this on behalf of the end-user, OS vendor and Network operator

SUMMARY

- We put requirements on TP framework to achieve
  - a healthy eco-system
  - fast TP evolution
- We proposed solutions to meet these requirements
  - We think that the SPUD initiative is a very important piece of the puzzle
  - Trust and enforcement issues have to be handled, we presented some ideas for this
- Several open questions, especially in the area of “trust and enforcement”
  - What is the task of IETF here?
  - What is next? What is missing?
  - Do the potential gains justify this complexity? Can we have something similar and good enough?

ERICSSON