This project has received funding from the European Unions Horizon 2020
research and innovation programme under grant agreement No 688421.

# Measurement and Architecture for a Middleboxed Internet

**H2020-ICT-688421**

# Use Cases and Requirements


| | | |
|---|---|---|
| **Author(s):** | ETH | Mirja Kühlewind (ed.) and Brian Trammell |
| | TID | Matteo Varvello |
| | UNIABDN | Gorry Fairhurst |
| | ZHAW | Stephan Neuhaus |
| | ALCATEL | Thomas Fossati |



| | |
|---|---|
| **Document Number:** | D3.1 |
| **Internal Reviewer:** | David Ros |
| **Due Date of Delivery:** | 30 June 2016 |
| **Actual Date of Delivery:** | 30 June 2016 |
| **Dissemination Level:** | Public |

# Disclaimer

*The information, documentation and figures available in this deliverable are written by the MAMI consortium partners under EC co-financing (project H2020-ICT-688421) and does not necessarily reflect the view of the European Commission.*

*The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The user uses the information at its sole risk and liability.*

European Commission | Horizon 2020 European Union funding for Research & Innovation

# Contents

# Executive Summary

This document provides a use case analysis to derive requirements for the protocol design of the Middlebox Cooperation Protocol (MCP) as well as other protocol extensions needed to support the deployment of MCP. As a starting point we perform a detailed analysis of the technical mechanisms and information flows for the following use cases:

1. Low Latency Support in Mobile Access Networks.
2. Throughput Guidance for Congestion Management in Mobile Networks.
3. Web Identity Translation (WIT) as a Network Service.
4. Multipath Bonding of Mobile and Fixed Network Capacity.

From this use case analysis, and a set of first principles, we derive a set of requirements for the design of MCP. The requirements specified in this document provide initial guidance for the development of MCP. However, they may be refined during the implementation of use cases, and based on discussions with the External Advisory Board (EAB), during the course of the project. Refinements will be presented in future deliverables D3.2 “Middlebox Cooperation Protocol Specification” and D3.3 “MCP and Flexible Stack Implementation Report”.

This document is related to requirements and use case analysis for a middlebox cooperation protocol called SPUD developed by MAMI partners within the IETF [22, 17], focusing on the use cases to be explored within the MAMI project.

# 1 Introduction

Given the increasing number of middleboxes in the Internet, it has become harder and harder to deploy new protocols or protocol extensions at Internet scale. This ossification has affected new transport protocols such as SCTP, which has limited deployment despite being standardized a decade and a half ago; and QUIC, which can only be deployed using UDP as an encapsulation protocol. Even extensions to TCP might be stripped at any point during the connection. While UDP encapsulation provides the port information that is necessary for traversing Network Address Translator (NAT) gateways, stateful network elements, including many firewalls, might require more information about the flow semantics, and therefore are often configured to block all traffic other than TCP.

Encryption of the transport layer headers inside an encapsulation layer would solve the problem of middlebox mangling. However, this would also break a large number of deployed functions in the Internet, e.g., network address translators (NAT) [4] and firewalls that rely on TCP's SYN/SYNACK semantics to set up state [13], or middleboxes that observe packets to associate forwarding the traffic with Quality of Service (QoS) policies [9].

In the Measurement and Architecture for a Middleboxed Internet (MAMI) project we develop a new common approach for pulling information of universal interest to middleboxes into a shim layer providing a the Middlebox Cooperation Protocol (MCP), which allows transport and application protocols to selectively and explicitly expose semantic information to middleboxes while maintaining protocol level details inside an encrypted encapsulation protocol such as the Datagram Transport layer Security (DTLS) [3]. This arrangement is shown in Figure 1: the MCP provides and end-to-end facility for communicating with the path, encapsulating and encrypting the transport and application layer headers and payload to prevent unauthorized meddling.

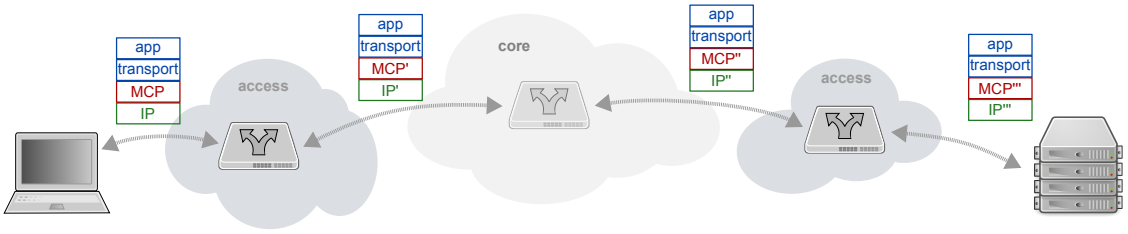


Figure 1: Overview of the MCP.

This document describes the use cases we will focus on in the development of the MCP in section 2 and functional requirements derived therefrom in section 3. The design of the MCP is additionally based on a few first principles, described in the subsections below.

Further this document provides the results of a preliminary security analysis also based on the described use cases and in consideration of the first principles described below. These insights described in section 4 on assumptions about the potential attacker model we make as well as possible different trust approaches that would be applicable will be considered during the development of MCP as well as used as a basis for a more extensive security analysis that will be provided in deliverable D3.3 towards the end of the project and after the development phase has been concluded. Further, MCP must therefore be designed such that the security and privacy properties of transports running over it are no worse than if they were to run without MCP.

Note that this document is related to two Internet-Drafts currently under consideration in the Internet Engineering Task Force (IETF): a use cases document [17] and a requirements document [22]. Given that the MAMI project envisions an evolutionary change to the Internet architecture with the MCP, standardization is a key dissemination channel to increase the impact of the project. The requirements in the Internet-Draft [22] are completely compatible with those in this document, expanding on technical requirements. This is because the use cases in the Internet-Draft are somewhat more expansive than those in this document, reflecting the fact that the MAMI project is focused on a subset of the IETF community's interest in explicit cooperation between endpoints and path devices, particularly those relevant to mobile access networks.

## 1.1 First Principles

### 1.1.1 Explicit Cooperation

The most important element of the architecture we envision in MAMI is *explicit cooperation*: MCP provides a method by which endpoints can explicitly communicate their intent to cooperate with middleboxes on the path, along with information that facilitates that cooperation, and by which middleboxes can explicitly communicate information about their properties to the endpoints. This replaces the present regime of *implicit cooperation*, where middleboxes are built on assumptions about the protocols the endpoints are speaking to derive information about the traffic (e.g. through Deep Packet Inspection (DPI)), which impedes the deployment of new protocols and leads to difficult tussles in the face of the rise of ubiquitous encryption.

Explicit cooperation has the added benefit that applications (and thereby users) have control over the information exposed to the network, providing a means by which we can re-balance the current tussle between user privacy and in-network functionality.

### 1.1.2 Declarative Signaling

Related to explicit cooperation is *declarative signaling*. There are many existing attempts to build in-band signaling for some of the use cases we consider here (those with a focus on QoS). However, these have largely failed to deploy at Internet scale, in part because the imperative nature of the signaling: a given signal is essentially a command to the on-path device to handle the packet in a certain, predetermined, predictable way. This leads to incentives for endpoints to lie about their traffic to gain advantage over traffic from other endpoints.

MCP bases all of its signals on a declarative vocabulary, in which the endpoints reveal information about intended properties of traffic and hints about which tradeoffs the traffic is willing to accept, and the path reveals information about intended traffic handling. However, there are no guarantees to receive a certain treatment from the network. In this way the MCP will reduce the incentive to lie, even in entirely untrusted environments.

### 1.1.3 Property Binding to Packet Groups

All transport semantics and other application or communication properties that an endpoint may wish to expose to middleboxes are bound to flows, or a group of packets within a flow. In MCP we therefore introduce a new term for associating packets together - a “tube”. Notionally, a tube consists of a set of packets with a set of common properties, that should therefore receive equivalent treatment from the network; these tubes may or may not be related to separate semantic entities in the upper layer protocol (e.g., a tube could be an SCTP stream, or an SCTP association). The information provided by MCP from an endpoint to a middlebox or vice versa is associated to a tube.

The 5-tuple of source and destination IP addresses, the source and destination ports, and IP protocol identifier (17 for UDP) is used in the Internet as an existing flow identifier. Due to the widespread deployment of network address and port translation, an MCP tube should be scoped to this 5-tuple: this mean while one 5-tuple flow can have multiple tubes, different 5-tuples cannot have the same tube ID.

Further this means that while there may be more than one tube active per 5-tuple, a given tube is associated with only one 5-tuple. While globally unique tube identifiers could be used to support mobility, this would introduce significant, uncontrollable risk to privacy through user and node tracking, and would require a larger tube ID space to reduce the probability of collision, as a tube ID would really be a global active sub-flow identifier. If it is necessary to expose to the path, which is not the case for any of our envisioned use cases, this functionality can be supported separately from the tube identification mechanism, by using a generic tube-grouping signaling bound to the tube.

### 1.1.4 Internet Deployability

Declarative signaling (based on grouping) supports one aspect of a larger principle, that of *Internet deployability*. The MCP will not be of any real use to evolving the transport layer of the Internet protocol stack if it does not itself deploy widely in the Internet. This has a specific technical consequence: supporting NAT and firewall traversal are fundamental requirements that any new protocol has to support. More specifically, MCP needs to provide a binding of limited related semantics (start, acknowledgment, and stop) to packets of a flow, or a tube, that are semantically related in terms of the application or higher layer protocol. By explicitly signaling start and stop semantics, a flow allows middleboxes to use those signals for setting up and tearing down their relevant state (NAT bindings, firewall pinholes), rather than requiring the middlebox to infer this state from continued traffic (such as STUN keep alive packets [14]).

Related to Internet deployability, any realization of the MCP which does not preserve the security properties of applications with respect to their operation without MCP (see section 4), or which presents a threat to end-user privacy with respect to operation without MCP, is not likely to be deployed.

### 1.1.5 Mobile Access Network Deployability

Given our focus on use cases relevant to mobile access networks, MCP must also deploy within present and future architectures and standards for mobile access. Here we envision

leveraging network function virtualization (NFV) and service function chaining (SFC) to rapidly deploy MCP-aware middleboxes within these networks.

SFC[1] is a standard currently developed by the IETF to support transport-independent packet processing chains. SFC can be applied to provide a virtualized, adaptive, and multi-functional middlebox. In essence, an SFC is an abstracted view of the required service functions (e.g., firewall, load balancer, video optimizer, spam filter, etc.) and the order in which they are applied to a given traffic flow. The integration between SFC and the mobile network is realized by interconnecting the SFC classifier with the relevant 3GPP control plane functions. In practice, the SFC steering policy is directly derived from a parent Policy Charging and Control (PCC) rule [2].

The SFC data plane is realized by the Network Service Header (NSH) protocol [21] which provides two key facilities: a service overlay network decoupled from the actual network topology to allow the dynamic insertion and removal of services, and a standard mechanism to share metadata about the flows within a given chain. The latter makes NSH similar to MCP, with the significant difference that NSH's scope is completely local to the network segments where SFC is applied (the (S)Gi-LAN, the datacenter), whereas MCP is end-to-end. When used with a service function chain, the MCP can provide information to the SFC classifier to help it identify the service chain that has to be applied to the present flow. MCP-supplied information could vary from a more or less precise hint about the transported content to the exact service chain identifier to be applied (a 24-bits identifier seems a sensible choice for this).

### 1.1.6 Failure Transparency

Troubleshooting of network-related problems with transport protocols is key to the smooth operation of a network; this is even more the case when transport protocol headers are encrypted to prevent middlebox meddling not explicitly authorized by the endpoints. Today troubleshooting is realized by additional mechanisms that may or may not work, e.g., Internet Control Message Protocol (ICMP) [20] that might be blocked. Given that MCP is designed for explicit middlebox cooperation, we see it as essential to integrate facilities for supporting troubleshooting into the protocol design. This will allow endpoints to bind troubleshooting information to traffic flows, as well as error reports from MCP-aware middleboxes directly back to sending endpoints.

[1] https://datatracker.ietf.org/group/sfc/charter/

# 2 Use Cases

Most use cases described in this section are focused on deployment scenarios in a mobile network. Similar mechanisms may also be used in a fixed network, however, resources in a mobile context are often more constrained and therefore more consideration is required of traffic handling needs. This leads us to focus on mobile scenarios. Also given that we will use the MONROE platform for experimentation and deployment tests, the large number of mobile nodes they plan to deploy will help us to investigate these use cases appropriately.

## 2.1 Low Latency Support in Mobile Access Networks

Mobile access has to handle traffic with varying characteristics: voice, web browsing, instant messaging, streaming video, email and lately more and more high-volume interactive applications, such as videoconferencing using WebRTC (`https://webrtc.org/`). These different applications do have very different requirements. While more traditional services such as email or web browsing are often impacted by high loss rates, interactive applications can usually tolerate higher loss rates, while having hard latency requirements. Further, for example in the case of WebRTC, the application has various streams with different characteristics and requirements that originate from (and are terminated on) the same end user device, possibly at the same time. This heterogeneity is likely to be exacerbated by the increase of machine-to-human and human-to-machine communication in the years to come.

3GPP networks have been designed to classify traffic so that they may assign an appropriate bearer to each upstream and downstream flow. A bearer is essentially an overlay network that spans the access radio and internal packet core segments of 3GPP networks. Each bearer has an associated QoS class. This bearer is in turn related to the resource allocation with the Radio Access Network (RAN).

An over-assignment could result in an inefficient use of the radio spectrum. Such wastage is undesirable, especially over a network that incurs cost to an operator or user — over-allocation can prevent resources from being available to other users of the RAN. An under-assignment could fail to deliver acceptable quality of experience to the user.

More and more (multiplexed) transmissions encrypt their content. Motivations include increased concern for privacy in the face of network monitoring [10], desire to avoid network manipulation of the transport protocol, and concerns about unwanted middlebox enhancement of content. As an example HTTP/2 [7, 18] breaks a basic assumption that is made by the 3GPP architecture: the notion that a 5-tuple represents a single flow with a given set of stable QoS attributes. Instead, a HTTP/2 session can carry multiple sub-streams over the same 5-tuple, all at the same time, all potentially different from the point of view of their required QoS treatment. Further, there is a trend that transport protocols will encrypt their payload opportunistically, e.g. see the TCP increased security (tcpinc) working group in the IETF (`https://datatracker.ietf.org/group/tcpinc`).

Unfortunately, increasing use of end-to-end and opportunistic encryption does not provide a means for a proper bearer identification in 3GPP networks and therefore does not allow for any differentiated support for certain traffic or services classes. Especially with the increasing bandwidth demand new services impose on the mobile network, a lack of information to perform traffic classification translates into a degradation of mobile network stability and a poorer service

to users, that has only been partially alleviated by the application of additional middleboxes such as Performance-Enhancing Proxies (PEPs).

## Information Exposed and Information Flow

Today, middlebox methods are often used to adapt network protocols to transmission network characteristics, or to derive information about the QoS expectations of the network flows. These methods vary in complexity from PEPs [8] that can split or augment the protocols to adapt to a radio network, e.g. by providing methods that manipulate the protocol acknowledgement channel [5], transparently intercept and proxy web flows, or implementing DPI that gathers metadata about the flow to assign traffic to service classes, bearers, QoS class, etc.

While encryption would prohibit these methods, in general a 3GPP network is completely uninterested in the transported content. To function properly, the network only needs to understand how the transported streams need to be treated in terms of their latency budget, sensitivity to loss, etc., so as to map a packet to an appropriate bearer and ultimately schedule its transmission in the most efficient way.

A simple signaling mechanism that addresses a large part of the problem mentioned above is to separate loss-sensitive from latency-sensitive traffic. Such a signaling has been proposed by participants of the MAMI project for DSCP codepoints in [23]. DSCP codepoints are often used within a network for traffic engineering. These codepoints may be mapped to lower-layer transmission networks (e.g., [15]), although many operators currently clear (bleach) the codepoints at peering points. Recent proposals recommend a restricted set of code points are exposed [12], but this approach still prevents introduction of new codepoints. Introducing new methods requires a transport-independent mechanism for signaling traffic characteristics from endpoints to middleboxes to carry this kind of metadata (see section 3.3).

An indication of latency-sensitivity vs. loss-sensitivity does not imply a need to prioritize one kind of traffic over another: while loss-sensitive traffic might face larger buffering delay but lower loss rate, latency-sensitive traffic has to make exactly the opposite tradeoff. In the 3GPP network this simple loss/latency indicator could be mapped to an appropriate QoS Class Identifier (QCI) [1] value (e.g., 7 for low-latency and 6 or 8 for low-loss) and in turn to the corresponding bearer, or the indicator could be interpreted directly by devices in the mobile core.

Signaling traffic characteristics (instead of application classes that would need to be mapped to a certain network mechanism) does not provide guaranteed network treatment, which is in line with the assumption that all signals provided by MCP are declarative, while it effectively also avoids any net-neutrality concerns. Further using trade-off-based signals at the same time also removes any incentive to lie by the endpoints.

In addition an endpoint could also indicate a maximum acceptable single-hop queueing delay per tube, expressed in milliseconds. While such a signal does not guarantee that sent packets will experience less than the requested delay due to queueing delay, it can significantly reduce the amount of traffic uselessly sitting in queues, since at any given instant only a small number of queues along a path (usually only zero or one) will be full.

## 2.2 Throughput Guidance for Congestion Management in Mobile Networks

Many flows are application-limited, where the application itself adapts the peak rate to changing traffic conditions or path characteristics, such as with unicast adaptive bitrate streaming video. This adaptation can be difficult, especially important when an application seeks to increase its rate, since TCP cross-traffic will often probe for available bandwidth more aggressively than the application's control loop. Further frequent rate adaptation may have a negative effect on the user's quality of experience, and should therefore be done infrequently.

When the access is provided by a mobile network, the situation is complicated by capacity variations in the RANs that can be unpredictable (e.g., because of multipath or shadow fading, interference, movement of the device, etc.). On the one hand, adaptation solely based on the transport control loop can therefore be inefficient: An overshoot of the allocated rate degrades application performance while an undershoot of the rate degrades link/network performance. On the other hand, at any point in time the mobile network knows how much RAN bandwidth (and hence capacity) is available and therefore can predict what will be available to any user's mobile device. This value is a function of the radio link quality reported by the mobile device, and anticipated interference. For paths comprising a single link, this feedback could be immediately fed back to the end host to inform the server's choice of the optimal sending rate (or cwnd in TCP). However, there are four key challenges that prevent such methods easily scaling to Internet flows:

- A robust signalling method is needed to ensure the path capacity signals do indeed relate to the traffic flows they control;
- methods need to be robust to equipment/protocol failure and the multiplexing of traffic;
- it needs to be possible to predict the total traffic and capacity; and
- the methods need to be deployable within the Internet.

One or more of these challenges have prevented usage of methods such as RSVP, QuickStart, RCP, etc. Nonetheless, there is renewed interest in such solutions for next generation mobile networks, as seen by recent work in the IETF [16], and IAB workshops and IETF BoFs on the topic (MaRNEW, see `https://www.iab.org/activities/workshops/marnew/`; ACCORD, see `https://www.ietf.org/proceedings/95/accord.html`).

### Information Exposed and Information Flow

Using signaling mechanisms provided in-band by MCP, middleboxes could note the maximum capacity available to a tube. Receiving this signal on a feedback channel could allow a sender to more quickly adapt its sending rate. This rate limit information might be derived from local per-flow rate limit policies, as well as from current information about load at the router or in the RAN.

This mechanism could be implemented similar to QuickStart [11]: The endpoint requests this information — in the first packet of the tube and/or periodically allowing the network to remark

a reduction of the rate — and a middlebox enters the requested information into a given header field if its own maximum value is lower than the current value in this field that was set by other middleboxes previously on the path.

Further, an endpoint thats sends application-limited traffic can provide an explicit per-tube indication of the maximum intended data rate needed by the current encoding or data source. If the bottleneck device is MCP-aware, it can use this information to decide how to correctly treat the tube, e.g., setting a rate limit or scheduling weight if served from its own queue. These signals can be sent throughout the lifetime of the tube, to help adapt to changing application demands and/or network conditions.

Note again that the provided information is declarative-only, meaning that neither a maximum capacity indication from the network nor the maximum demand estimate from the endpoint provide any guarantees that the indicated capacity can be used by the flow. However, indications from the network about absolute limits help the endpoint to make the decision in its local adaptation mechanism (e.g., congestion control, or codec rate selection). Demand indication from the endpoint can also be used for capacity planning where the actual sending rate can be monitored and policed if needed.

## 2.3 Web Identity Translation (WIT) as a Network Service

The massive growth of the web has been funded almost entirely via advertisements shown to users. To serve the most relevant ads, web advertisement agencies rely on mechanisms to *uniquely identify* and *track* user behavior over time. Known as *trackers*, these systems are able to uniquely identify a user via a variety of methods (e.g., persistent cookies or browser fingerprinting) and over time can build up enough information to be able to place targeted ads.

While ad agencies' use of trackers has enabled the free-to-use model of the web, it also raises invasion of privacy concerns. These concerns have led to the creation of client side applications that block trackers and ads, for example AdBlock. While AdBlock has been quite successful in mitigating users' exposure to trackers, *by definition* it prevents the display of ads, and thus hurts web services' revenue streams.

The *Web Identity Translation* (WIT) service [19] aims at balancing the needs of users for privacy and the needs of advertisers for information to drive Online Behavioral Advertising (OBA). WIT is an active service running on a proxy between users and web-sites that host tracking cookies and OBA code, configured to intercept encrypted traffic. WIT is transparent to trackers and does not require any change in the infrastructure of the ad ecosystem. Like Network Address Translation (NAT), it introduces a mapping between *private* and *public* third-party tracking cookies. When a particular user's browsing habits start making her uniquely identifiable, WIT intervenes via the private-to-public cookie mappings using one of several policies aimed at restoring user anonymity within the context of the OBA ecosystem.

When WIT receives a request from a new user, it places him in the quarantine phase, where all tracking is blocked until a *browsing signature*[1] for the user is collected. This signature is stored as a history vector and will be used to make intervention decisions later on, based on the ranking of WIT's private cookies with respect to this signature. Once there is sufficient data, the user enters the triage phase. This is an active monitoring phase, which examines the effect

[1] It is a vector of domain and visit frequency tuples.

each request has on potentially assisting the de-anonymization of a user. WIT monitors users during the triage phase, and intervenes only if 1) a new request makes the user more similar to their signature and 2) their current history ranks among the top k histories with respect to their signature. During the triage phase, multiple history vectors can be associated with a user. These history vectors correspond to a public cookie, and represent what information has been allowed to pass through to the tracker. The triage phase is responsible for guaranteeing that none of these history vectors can be linked with high probability to the signature vector of the user that is associated with them.

The intervention action works as follows. As requests arrive, if the current history of a user is determined to be ranked below a threshold $k$, WIT searches through the history of the $(k+1)$th ranked user for a request that, 1) matches the topic of the original request, and 2) when added to the user's browsing history minimizes the distance to its original signature.

### Information Exposed and Information Flow

Based on the above, in order to properly work WIT needs access to the following information:

- Visited domains: this data allows building user history vectors. Anonymization is possible in the form of a domain description composed by a unique identifier and a list of associated topics, instead of the domain visited itself.
- Cookies: WIT requires cookie access to strip them off during quarantine and manipulate them to allow intervention. Since this requirement can decrease user privacy, an alternative design allows WIT to inform the client on which strategy has to be applied, e.g., strip cookies or replace with some specific ones.

While this information could be provided out-of-band, such an approach would require an own (proprietary) protocol which is difficult to deploy. Using MCP to provide the needed background information or to signal user policies as an application-independent mechanism would allow wider support and thereby enhance the privacy-properties of the provided service.

## 2.4 Multipath Bonding of Mobile and Fixed Network Capacity

Recent work has applied Multipath TCP proxies to the problem of bonding a customer's multiple access interfaces to the Internet, to augment available capacity, especially in areas with marginal fixed connectivity. However, such proxies only apply to TCP traffic, and while UDP-based media streams can be tunneled through bonded TCP connections, this would lose the advantages of loss-tolerant media-oriented transports. Solutions that apply bonding at layer 3 [6] not only need to implement a scheduling algorithm to shift traffic between fixed and mobile lines, but also a reordering buffer at tunnel egress as many flows today are re-ordering sensitive. This is especially true for TCP, because TCP's fast recovery mechanism will interpret three duplicated acknowledgements as a signal of congestion loss, a case that can easily arise due to re-ordering, resulting in a reduction of sending rate.

Even though the majority of traffic in the Internet is (still) TCP, it is likely that new protocols will be designed such that they are (more) robust to reordering. Further, with an increasing deployment of ECN, even TCP's congestion control reaction based on duplicated acknowledgements could be relaxed (e.g., by reducing the sending rate gradually depending on the number of lost packets). If this information would be known by the tunnel egress, the bonding service could abstain from using a re-ordering buffer for traffic that is not re-ordering-sensitive in favor of reduced latency.

## Information Exposed and Information Flow

Reordering sensitivity is a per-tube signal (as reordering can only happen with a flow of multiple packets). However, to avoid state in middleboxes, it would be beneficial to have a reordering-sensitive flag in each packet.

A transport should set the flag if it is not sensitive to reordering, e.g., if it uses a more advanced mechanism than duplicated acknowledgements for loss detection, if the congestion control reaction to this signal imposes only a small performance penalty, if congestion control is not window-based, and/or if the flow is short enough that it will not impact its performance.

In addition MCP could be used to provide further policy indications to the scheduler about which channel is preferred for which tube or packet.

# 3 Requirements

In this section, we list functional requirements derived from the use cases and principles laid out in this document. Additional technical requirements appear in [22], which presents a generic superset of the requirements here, and will be discussed in detail along with other deployment considerations in future deliverable D3.2.

See table 1 for an overview of the link between each of these requirements and the first principle and/or use case from which they are derived.

| Principle | requirements (sec. ref.) | |
|---|---|---|
| Explicit Cooperation (§1.1.1) | 3.1, 3.2, 3.3, 3.4, 3.11, 3.13 | |
| Declarative Signaling (§1.1.2) | 3.8, 3.9 | |
| Internet Deployability (§1.1.4) | 3.1, 3.6, 3.11, 3.12, 3.13 | |
| Mobile Deployability (§1.1.5) | 3.1, 3.6 | |
| Property Binding (§1.1.3) | 3.1, 3.2 | |
| Failure Transparency (§1.1.6) | 3.5 | |
| Use Case | requirements (sec. ref.) | MAMI partner(s) involved |
| Low Latency Support (§2.1) | 3.1, 3.2, 3.7, 3.8 | ETH, UNIABDN, NOKIA |
| Throughput Guidance (§2.2) | 3.1, 3.2, 3.3, 3.4, 3.8, 3.12, 3.13 | NOKIA, TID |
| Web Identity Translation (§2.3) | 3.1, 3.2, 3.8, 3.10, 3.11 | TID, NOKIA |
| Multipath Bonding (§2.4) | 3.1, 3.2, 3.7, 3.8 | ETH |

Table 1: Overview of principles, use cases and requirements.

## 3.1 Grouping of Packets and Bidirectionality

MCP must provide a basic facility for associating packets together into a tube. Each packet in an MCP “flow” (determined by 5-tuple) can only belong to exactly one tube.

The simplest mechanisms for association is the addition of an identifier to each packet in a tube. For the purposes of this requirement we consider this identifier as being a simple vector of N bits. The properties of the tube identifier are subject to tradeoffs on the requirements for privacy, security, ease of implementation, and header overhead efficiency (see further below).

In determining the optimal size and scope for this tube identifier, we first assume that the 5-tuple of source and destination IP address, UDP ports, and IP protocol identifier (17 for UDP) is used in the Internet as an existing flow identifier, due to the widespread deployment of network address and port translation. We conclude that MCP tube IDs should be scoped to this 5-tuple[1].

N must still be sufficiently large, and the bits in the identifier sufficiently random, that possession of a valid tube ID implies that a node can observe packets belonging to the tube. This reduces the chances of success of blind packet injection attacks of packets with guessed valid tube IDs.

When scoped to 5-tuples, the forward and backward directions of a bidirectional connection will have different tube IDs, since these will necessarily take different paths and may interact with a

[1] Or 6-tuple, including the Differentiated Services Code Point (DSCP), which is often used for per-hop routing decisions, though it is not relevant to addressing contexts at NATs, and only variably related to per-flow firewall state.

different set of middleboxes due to asymmetric routing. MCP will therefore require some facility to note that one tube is the “reverse” direction of another, a general case of the tube grouping signal above.

This requirement directly addresses principle 1.1.3, and is a basis for all signaling other than per-packet signaling addressed in these requirements.

## 3.2 Signaling of Per-Tube Properties

MCP must be able to provide information scoped to a tube from the endpoint(s) to all MCP-aware nodes on the path about the packets in that tube, as described in the use cases in sections 2.1, 2.2, and 2.3.

We note that in-band signaling would meet this requirement.

## 3.3 Path to Receiver Signaling under Sender Control

MCP must be able to provide information from a MCP-aware middlebox to the receiving endpoint. This information is associated with a tube, in terms of “the properties of the path(s) the packets in this tube will traverse”. This signaling must happen only with explicit sender permission and be sent to the receiver of packets in the tube.

This requirement addresses the use case in section 2.2.

We note that in-band signaling would meet this requirement, if the sender created a “placeholder” in-band that could be filled in by the middlebox(es) on path. In-band signaling has the advantage that it does not require foreknowledge of the identity and addresses of devices along the path by endpoints and vice versa, but does add complexity to the signaling protocol.

## 3.4 Receiver to Sender Feedback

Since path to receiver signaling does not allow the path to send information to a sender, e.g. in order to influence the properties of the traffic sent, MCP must be able to send information collected from MCP-aware middleboxes along the path from receiver back to the sender.

This requirement addresses the use cases in section 2.2.

## 3.5 Direct Path to Sender Signaling

As noted in Section 1.1.6, MCP must provide a facility for a middlebox to send a packet directly in response to a sending endpoint, primarily to signal error conditions (e.g., “packet administratively prohibited” or “no route to destination”, as in present ICMP). We note that this direct return packet generated by the middlebox should use the reversed end-to-end 5-tuple in order to receive equivalent NAT treatment, though the reverse path might not be the same as the forward path.

To preserve endpoint control over this feature, and to reduce the surface for amplification at-

tacks, an MCP-aware middlebox must not emit a return packet unless it is a direct response to a packet from a sending endpoint, and must not forward a packet for which it has sent a direct return packet.

## 3.6 Tube Start and End Signaling

As noted in section 1.1.4, MCP must provide a facility for endpoints to signal that a tube has started from the endpoint perspective, that the start of the tube has been acknowledged and accepted by the remote endpoint(s), and that a tube has ended and its state can be forgotten by the path. Given unreliable signaling, both endpoints and devices on the path must be resilient to the loss of any of these signals. Specifically, timeouts at endpoints and middleboxes are still necessary to clean up stale per-tube and per-flow state. Tube start and tube end processing on on-path devices must be resilient to path changes.

## 3.7 Additional Per-Packet Signaling

MCP may provide a facility for signaling semantically simple information (similar to tube start and end) on a per-packet as opposed to a per-tube basis, such as a single bit for the loss/latency tradeoff as described in section 2.1, or for reordering tolerance as in 2.4.

Properties signaled per packet reduce state requirements at middleboxes, but also increase per-packet overhead. Small signal size (in bits of entropy) and encoding efficiency (in bits on the wire) is therefore more important for per-packet signaling than per-tube signaling. If per-packet signals need to be used by multiple hops along a path, these will need to be encoded in an efficiently-implementable way (i.e., using fixed-length, constant-offset data structures).

Given these constraints, per-packet signaling is necessary for certain use cases, it is likely that MCP will provide a very limited set of per-packet signals using flags in an MCP header, and require all more complex properties to be bound per-tube.

## 3.8 Declarative signaling

As noted in section 1.1.2, all use cases described above assume that information signaled via MCP is defined to be declarative (as opposed to imperative).

That also means, an MCP endpoint must function correctly even if no middlebox along the path understands the signals it sends, or if it is sent signals from middleboxes it does not understand. It must also function correctly if the path (and thereby the set of middleboxes traversed) changes during the lifetime of a tube; endpoints cannot rely on the creation or maintenance of state even on cooperative middleboxes. Likewise, an MCP-aware middlebox must function correctly if it is sent signals from endpoints it does not understand, or in the absence of expected signals from endpoints.

The declarative nature of this signaling removes any requirement that MCP must provide reliability for its signals.

## 3.9 Extensibility and Common Vocabulary

The MAMI project targets a set of use cases chosen to inform the design and evaluation of MCP. We expect MCP to be deployed in a wide range of other scenarios – including scenarios beyond what we can anticipate at this time, MCP must enable multiple new transport semantics and application/path declarations without requiring updates to MCP implementations in middleboxes.

Consequently, for the interoperability of MCP endpoints and middleboxes with each other, the use of MCP for standard signaling must use a common vocabulary with registered codepoints allocated under relatively restrictive policy. This restrictive policy serves primarily security and privacy goals (i.e., reducing the risk of misuse of the extensibility provided by the protocol).

This requirement addresses the principle in section 1.1.2.

## 3.10 Privacy

MCP must allow endpoints to control the amount of information exposed to middleboxes, with the default being the minimum necessary for correct functioning. This includes the cryptographic protection of transport layer headers from inspection by devices on path, in order to prevent ossification of these headers.

This requirement derives from the principle in section 1.1.4 and is reflected in the security analysis in section 4.

## 3.11 Authentication

MCP must not require any authentication or a priori trust relationship between endpoints and middleboxes to function. However, MCP should interoperate with the presentation/exchange of authentication information in environments where a trust relationship already exists, or can be easily established, either in-band or out-of-band, and use this information where possible and appropriate such as required in the use case in section 2.3.

Given the advisory nature of MCP signaling, MCP may also support eventual authentication: authentication of a signal after the reception of a packet after the one containing the signal.

This requirement follows the principles in sections 1.1.4 and 1.1.4, and the security analysis in section 4.

## 3.12 Integrity

MCP must be able to provide integrity protection of information exposed by endpoints in MCP-encapsulated packets. This implies protection of the content for all information exposed by a sending endpoint, as well as the protection of the presence (but not the content) of information exposed by on-path devices. The receiving endpoint should be able to detect changes to headers that MCP uses for its own signaling (whether due to error, accidental modification, or malicious modification), as well as the injection of packets into an MCP flow (defined by 5-tuple)

or tube by nodes other than the remote endpoints. Errors and accidental modifications can be detected using a simple checksum over the MCP header, while detecting malicious modifications requires cryptographic integrity protection, bootstrapped off the upper layer transport protocol.

In any case, integrity protection of the upper layer transport protocol is left up to that protocol.

This requirement follows the principles in sections 1.1.1 and 1.1.4, and the security analysis in section 4.

## 3.13 Encrypted Feedback

MCP must provide a feedback channel from the receiver to the sender to, e.g., reflect signal information on the forward path from middleboxes back to the sender. As feedback from a receiver to a sender (see Section 3.4) does not need to be exposed to the path, this feedback channel should be encrypted for confidentiality and authenticity, when available. This facility will rely on cooperation with the upper-layer transport protocol or some other out-of-band mechanism to provide these guarantees.

This requirement follows the principle in section 1.1.4, and the security analysis in section 4.

# 4 Security Analysis

This section discusses different trust models that the design of MCP can be based on. Further, we introduce the basic attacker model that we will use in a future deliverable, mainly D3.3, to evaluate the security of the proposed solution in the sense of which skills and access to information the attacker is assumed to have.

## 4.1 Trust Model

A *trust model* describes which actors in a system need to trust each other in order for the system to function correctly. Entities that trust each other need to verify that messages they receive are indeed from the trusted entity. In other words, trust requires *authentication*.

The more system components need to trust one another, the more difficult the system is to configure and run securely. Conversely, the fewer components need to trust one another, the easier it is to operate. The best case is thus that components do not need to trust one another. It is possible to implement “no trust” either by carefully vetting messages for plausibility or by treating messages as advisory only which is discussed in the next section.

Further we also discuss the alternative ways of realising MCP with regard to trust where bi-directional trust relationships exists.

### 4.1.1 Zero Trust Model

The first way requires no trust between participants. More precisely, the system does not employ authentication and thus no technical means by which trust could be established. All data exchanged through MCP are advisory only; while endpoints can authenticate each other and verify the data has not been tampered with (see requirement in section 3.12), middleboxes can’t authenticate endpoints and vice-versa, so information may have been manipulated or injected by an attacker.

This would mean that MCP-compliant middleboxes are easy to fool by any on-path component (or any component that can inject packets through an on-path component). Consider for example the case where one endpoint initiates a bulk data download from another endpoint. A malicious on-path component sees this and could then inject MCP packets that wrongly instruct other on-path middleboxes to shape the traffic for minimum latency instead of maximum throughput.

Of course, for someone to fool a middlebox, they have to either be on-path or at least collude with an on-path component. In that case, attackers could just as well shape traffic themselves. The only difference would be that attribution would be a lot easier if the attacker-controlled on-path middlebox were to do the shaping instead of a fooled but otherwise benign middlebox run by another organisation.

Advantages ($+$) and disadvantages ($-$) of this solution are thus as follows:

- $+$ No authentication, no key management, and thus no key management issues.
- $+$ Attackers do not have more opportunities for shaping traffic than they had before.

+ Data transported over MCP can (with correct MCP implementations) at worst cause QoS degradation, not a total disconnect.

– No authentication, thus all data transported via MCP is advisory only.

– Attackers may be able to cause other on-path devices to process packets differently.

– Attackers may be able to cause endpoints to behave differently by lying about path conditions.

### 4.1.2 Middlebox Authentication Model

The second way of realising MCP with regard to trust is to establish trust relationships through authentication with (potentially) every on-path middlebox. This makes it impossible for an attacker to inject MCP messages that are not detected and discarded, and it enables endpoints and on-path middleboxes to employ what amounts to write-protection of those parts of MCP messages that they authored.

Once an authenticated channel has been established, it is a comparatively simple additional step to agree on encryption keys. This would enable any on-path component to selectively send MCP messages only to selected on-path components, where other on-path components could not read the contents of these MCP messages. On the other hand, MCP messages are not supposed to contain privacy-relevant data in any case, so this feature may well go unused.

MCP is supposed to work largely autonomously, so autonomous systems for authenticity would have to be used for the MCP. This in all probability means some form of Public Key Infrastructure (PKI), with all its attendant possibilities, but also its problems. Issues like key expiration and revocation are as tricky now as they were when PKI for the web was first proposed and introduced.

However, while authenticity is a necessary condition for trust in our model, authenticity and trust are not the same: if Alice tells Bob to his face that “the moon is made of cheese”, this statement is perfectly authentic, but Bob would be well advised not to trust it. In other words, trust may be misplaced and betrayed.

Advantages (+) and disadvantages (−) of this solution are thus as follows:

+ Data can be exposed to selected middleboxes.

– Key management, and thus key management issues through PKI.

– Attackers can still lie.

– MCP data is supposed not to be personally identifiable anyway.

Trust is normally tied to identity: Alice may trust Bob, but she may not trust Robby, Bob’s twin brother. Nevertheless, we find it convenient to lump together hosts or middleboxes that have similar characteristics. These are sets of hosts or middleboxes where we either trust all or none of the members. We do not yet have a convenient name for such sets; the word “domain” would be good, but already has a specific meaning in networking circles. We will use the capitalised word “group” for this as yet unnamed property of a group of trusted middleboxes.

One way to determine membership in groups is through certificates. If a middlebox can show a certificate that is signed by a Certificate Authority (CA) that we believe will only issue valid certificates for a given group, and if that host can prove that it possesses the corresponding private key, then we will believe the hosts' claim to membership in a given group. Based on that membership, we make trust decisions: we may trust middleboxes in certain groups but not in others.

Finally, even when all parties are honest, adding more entities to a session necessarily increases the attack surface: a bug or misconfiguration on any one could compromise the session. This risk is inherent in the problem, not any particular solution.

## 4.2 Attacker model

For future evaluation of the security of our proposed protocol, we make the following assumptions about an attacker:

1. Attackers can sniff any packet from any source and can combine different flows from different sources to learn what they can from such collections.
2. Attackers can arbitrarily inject traffic.
3. Attackers do not have the ability to subvert any authentication scheme that is being used by MCP.

Users of MCP must be safe from such attackers.

To address assumptions (1) and (2), we note that the size N of the tube ID must still be sufficiently large, and the bits in the identifier sufficiently random, that possession of a valid tube ID implies that a node can observe packets belonging to the tube. This reduces the chances of success of blind packet injection attacks with packets with guessed valid tube IDs.

However, if the attacker is on-path, it still knows the tube ID and might be able to inject valid MCP packets to drive a Denial of Service (DoS) attack. Therefore, MCP must at least provide minimal protection against this trivial abuse including flooding and state exhaustion attacks, as well as reflection and amplification attacks.

It would probably even be possible, with appropriate key management, to control what data is exposed on a per-middlebox basis. But this would still not protect from several middleboxes *colluding* in order to undermine the protection mechanisms of MCP. There are two different situations, depending on what protection MCP is aiming for.

In the first situation MCP would provide (different) information to a selected set of middleboxes, even if they are on the same path. For example, if only one middlebox on the path between two endpoints does video transcoding, only this middlebox needs access to the relevant metadata. Another example is when part of the path goes through a country where data protection is not a high concern, especially for transit traffic; in this case, we may want to preclude middleboxes on that part of the path from seeing any plaintext metadata. Supporting this model would entail comparatively complicated and unwieldy key management, since different keys would have to be made available to different middleboxes on the same path.

In the second situation, where MCP does not impose per-middlebox key management as a requirement, we assume that all middleboxes on a path ought to have access to the same

information. In this case we still have to consider additional threats by middleboxes on different paths that cooperate to gain additional information about a transmission or endpoint.

The main problem is that there is really no way to prevent middleboxes from colluding: once different pieces of data are exposed to different middleboxes, we don't see a way to prevent those middleboxes from colluding to gain access to the union of this exposed data. Whether this is an actual threat is an question that MAMI is ideally suited to answering experimentally.

Given that any information exposure should be under endpoint control, an end host can decide that the protection of certain data has higher priority over middlebox support, even if that comes at the expense of user experience. For example, if an endpoint configuration explicitly specifies that information about this endpoint can not be shared, then a middlebox may not be able to transcode a video to fit on the device's screen. That may mean that the video takes longer to arrive at the endpoint, that it renders sub-optimally, or even that it does not arrive at all, but that is the endpoint's (or user's) choice.

We note that this is not the liberating principle that it appears to be: a service is free to provide that service only if certain data is surrendered, much like some services today that provide their service only in exchange for running scripts and setting cookies on the user's machine.

Further we note that the use of MCP must not weaken the essential security properties of any of the higher layer protocols. If the payload is encrypted for confidentiality, for example, the use of MCP must not allow deep packet inspection systems to have access to the plaintext. Likewise, the use of MCP must not create additional opportunities for linkability not already existing.

# 5 Conclusion

This document has described use cases that the MAMI project will focus on driven by industry needs as reported by the project partners, the External Advisory Board (EAB) or external industry cooperations as well as existing and up-coming challenges in mobile networks. Especially for mobile networks, where complexity continues to grow and resources remain constrained, the increasing use of encryption presents an enormous challenge. Explicit cooperation that provides signals as input for low latency support, throughput guidance, content translation, or re-ordering assistance as identified by the use cases in this document is necessary to keep mobile network management viable.

These use cases, on the background of a set of first principles adopted by the project, and subject to a security analysis of the space of explicit cooperation protocols given in this document, motivate the requirements listed in this deliverable as a basis for the design of MCP. These functional requirements describe the basic functionality a middlebox cooperation protocol needs to provide. We have not identified any conflicts among these requirements, and therefore the protocol design of MCP must address them all. We will evaluate the protocol specification of MCP in the future deliverable D3.2 against these requirements, by showing that the proposed design matches all requirements listed here.

Further, D3.2 will discuss additional technical requirements that need to be considered for wide-spread and incremental deployability, especially in mobile networks. These deployment considerations build the basis for a successful deployment in the current Internet for any new protocol above the networking layer. We will use the MONROE testbed for experimental evaluation of the proposed use cases and the applicability of MCP to the current network situation as it can be found in the Internet as well as in mobile networks. This evaluation and testbed-based validation of our implementation will be described and discussed in D2.2 specifying and applying evaluation criteria aligned to our main use cases as described in this deliverable. Further, D3.3 will provide a final thread and trust analysis, based on the initial security analysis provided in this document as well as further work that will be undertaken along with the development of MCP.

# References

[1] 3GPP. Policy and charging control architecture. TS 23.203, 3rd Generation Partnership Project (3GPP).

[2] 3GPP. Policy and charging control over Gx reference point. TS 29.212, 3rd Generation Partnership Project (3GPP).

[3] F. Andreasen and D. Hancock. Media gateway control protocol fax package. RFC 5347, RFC Editor, October 2008.

[4] F. Audet and C. Jennings. Network address translation (nat) behavioral requirements for unicast udp. BCP 127, RFC Editor, January 2007. http://www.rfc-editor.org/rfc/rfc4787.txt.

[5] H. Balakrishnan, V. Padmanabhan, G. Fairhurst, and M. Sooriyabandara. Tcp performance implications of network path asymmetry. BCP 69, RFC Editor, December 2002.

[6] M. Bednarek, G. B. Kobas, M. Kühlewind, and B. Trammell. Multipath bonding at layer 3. In *Proceedings of the First ACM/IRTF Applied Network Research Workshop*, Berlin, Germany, July 2016.

[7] M. Belshe, R. Peon, and M. Thomson. Hypertext transfer protocol version 2 (http/2). RFC 7540, RFC Editor, May 2015. http://www.rfc-editor.org/rfc/rfc7540.txt.

[8] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. Performance enhancing proxies intended to mitigate link-related degradations. RFC 3135, RFC Editor, June 2001.

[9] V. Fajardo, J. Arkko, J. Loughney, and G. Zorn. Diameter base protocol. RFC 6733, RFC Editor, October 2012. http://www.rfc-editor.org/rfc/rfc6733.txt.

[10] S. Farrell and H. Tschofenig. Pervasive monitoring is an attack. BCP 188, RFC Editor, May 2014. http://www.rfc-editor.org/rfc/rfc7258.txt.

[11] S. Floyd, M. Allman, A. Jain, and P. Sarolahti. Quick-start for tcp and ip. RFC 4782, RFC Editor, January 2007.

[12] R. Geib and D. Black. Diffserv-interconnection classes and practice. Internet-Draft draft-ietf-tsvwg-diffserv-intercon-06, IETF Secretariat, June 2016. http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-diffserv-intercon-06.txt.

[13] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. Nat behavioral requirements for tcp. BCP 142, RFC Editor, October 2008.

[14] C. Holmberg. Indication of support for keep-alive. RFC 6223, RFC Editor, April 2011.

[15] ITU-T. Quality of service mapping and interconnection between ethernet, ip, and mpls networks. Recommendation Y.1566, International Telecommunication Union, Geneva, July 2012.

[16] A. Jain, A. Terzis, H. Flinck, N. Sprecher, Swaminathan, and K. Smith. Mobile throughput guidance inband signaling protocol. Internet-Draft draft-flinck-mobile-throughput-guidance-02, IETF Secretariat, March 2015. http://www.ietf.org/internet-drafts/draft-flinck-mobile-throughput-guidance-02.txt.

[17] M. Kühlewind and B. Trammell. Use cases for a substrate protocol for user datagrams (spud). Internet-Draft draft-kuehlewind-spud-use-cases-01, IETF Secretariat, March 2016. http://www.ietf.org/internet-drafts/draft-kuehlewind-spud-use-cases-01.txt.

[18] M. Nottingham and M. Thomson. Opportunistic security for http. Internet-Draft draft-ietf-httpbis-http2-encryption-06, IETF Secretariat, June 2016. http://www.ietf.org/internet-drafts/draft-ietf-httpbis-http2-encryption-06.txt.

[19] F. Papaodyssefs, C. Iordanou, J. Blackburn, N. Laoutaris, and K. Papagiannaki. Web identity translator: behavioral advertising and identity privacy with wit. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, page 3. ACM, 2015.

[20] J. Postel. Internet control message protocol. STD 5, RFC Editor, September 1981. http://www.rfc-editor.org/rfc/rfc792.txt.

[21] P. Quinn and U. Elzur. Network service header. Internet-Draft draft-ietf-sfc-nsh-05, IETF Secretariat, May 2016. http://www.ietf.org/internet-drafts/draft-ietf-sfc-nsh-05.txt.

[22] B. Trammell and M. Kühlewind. Requirements for the design of a substrate protocol for user datagrams (spud). Internet-Draft draft-trammell-spud-req-04, IETF Secretariat, May 2016. http://www.ietf.org/internet-drafts/draft-trammell-spud-req-04.txt.

[23] J. You, M. Welzl, B. Trammell, M. Kühlewind, and K. Smith. Latency loss tradeoff phb group. Internet-Draft draft-you-tsvwg-latency-loss-tradeoff-00, IETF Secretariat, March 2016. http://www.ietf.org/internet-drafts/draft-you-tsvwg-latency-loss-tradeoff-00.txt.