Network-Assisted MPTCP

IETF#98, Chicago, March 2017

M. Boucadair (Orange)
C. Jacquenet (Orange)
O. Bonaventure (Tessares)
W. Henderickx (ALU/Nokia)
R. Skog (Ericsson)
D. Behaghel (OneAccess)
S. Secci (Universite Pierre et Marie Curie)
S. Vinapamula (Juniper)
S. Seo (Korea Telecom)
W. Cloetens (SoftAtHome)
U. Meyer (Vodafone)
LM. Contreras (Telefonica)
B. Peirens (Proximus)

Documents Structure

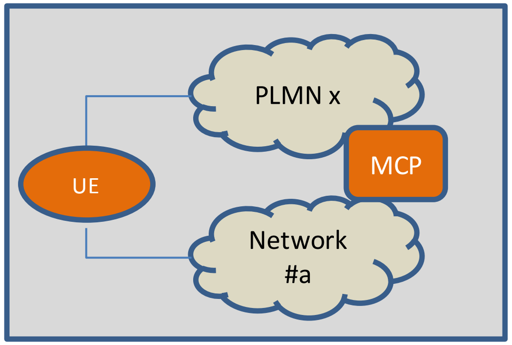
- Deployment considerations
  - draft-nam-mptcp-deployment-considerations
- Core specification
  - draft-boucadair-mptcp-plain-mode
- Provisioning
  - draft-boucadair-mptcp-dhc (customer side)
  - draft-boucadair-mptcp-radius (network side)

Recall the Motivation

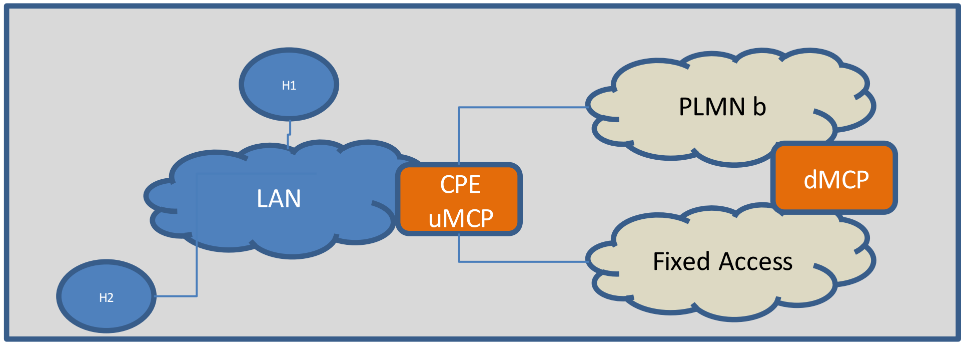
- Operators that own both cellular and fixed networks want to offer converged services
- Operators want to enhance Quality of Experience for their customers by boosting some access lines
  - Grab more capacity by means of link aggregation
  - Increase serviceability during network attachment failures
- Applies for both fixed and cellular networks

Network-Assisted MPTCP: Rationale

- Given
  - The MPTCP penetration rate is close to null at the server side, and
  - Network Providers do not control customers' terminals
- A network-assisted MPTCP model is needed



Single Proxy



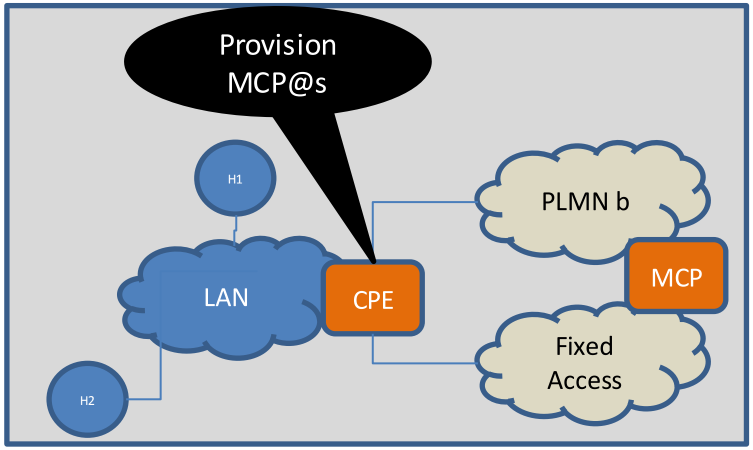
Dual Proxy

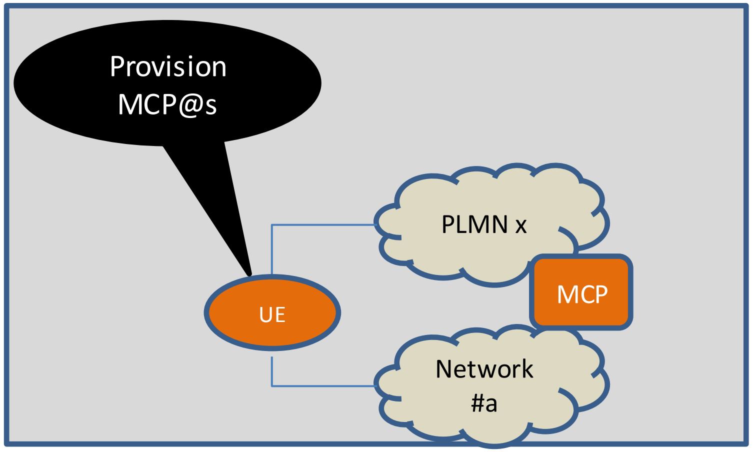
MCP Design Goals

- **0-RTT proxy**
- No overhead: Avoid the use of tunnels/encapsulation
- Accommodate various deployments
  - Be compatible with IPv4/IPv6
  - Do not assume the MCP is located on a default forwarding path
  - Support both single and dual proxy deployments
- **Avoid interfering with native MPTCP connections**
  - … and encourage MPTCP when the remote peer supports it for the sake of path diversity

How MCPs are inserted in an outbound connection?

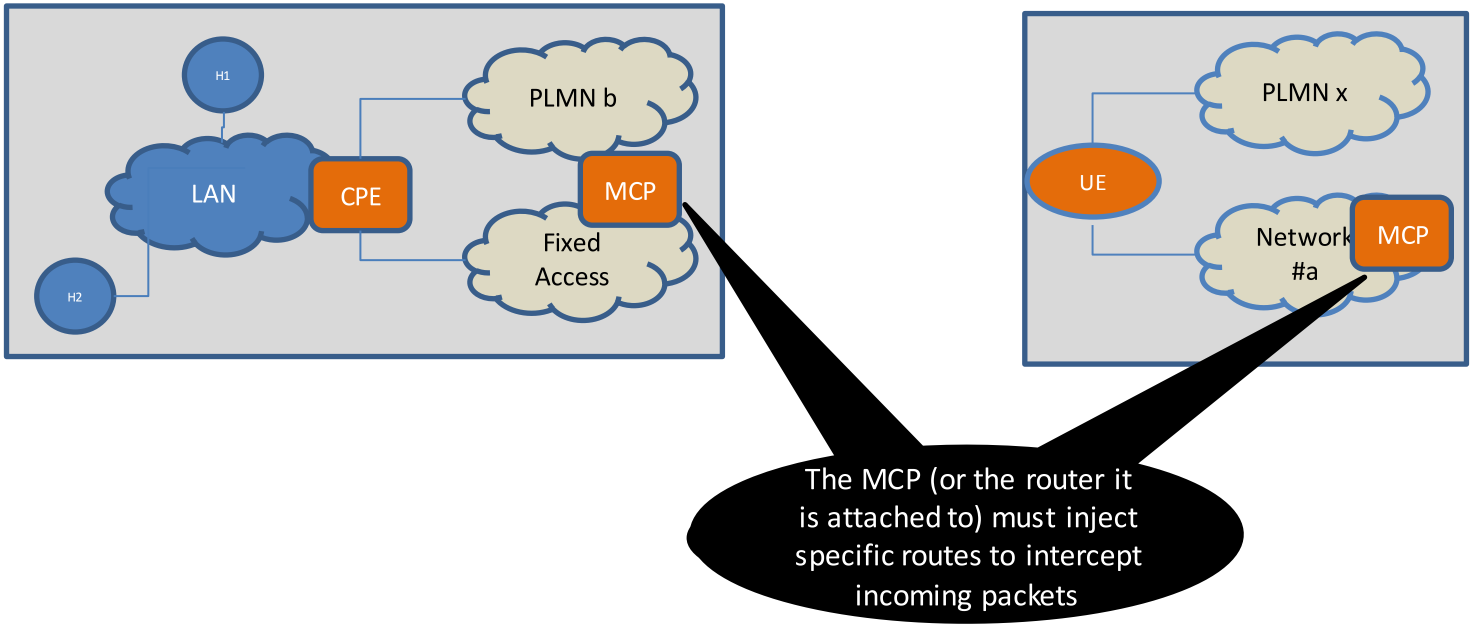
- ***Explicit Mode***: MPTCP data are sent explicitly to an MCP’s IP address
  - No need for traffic inspection
  - No adherence to the underlying routing and forwarding policies
    - The MCP can be located anywhere in the network
- The initial subflow may be placed via any of the available network attachments
- Allows also for backup service





How MCPs are inserted in an inbound connection?

- Specific routes must be injected to intercept incoming traffic
  - Achieved by the MCP or a router to which it is attached to
  - The prefix/address aggregates to be announced are deployment-specific
- The address/port to use to place an incoming connection is retrieved by the remote peer using out of band mechanism (e.g., DNS)



The MCP (or the router it is attached to) must inject specific routes to intercept incoming packets

How 0-RTT proxying is possible? Explicit Mode

- ***Supply (forwarding) data during the 3WHS of the initial subflow***
  - Supply at least the ultimate destination IP address [and port] by means of MP_CONVERT elements
  - No overhead for subsequent MPTCP messages
- **Which channel to use to supply data during the 3WHS?**
  - The payload of the SYN of the initial subflow
- **What if data is present in the original SYN?**
  - That data must be placed right after the MP_CONVERT IEs when the MCP creates the initial SYN of the MPTCP leg
  - MP_CONVERT IEs will be striped by the downstream MCP
- **How to distinguish MP_CONVERT elements from application supplied data?**
  - Use a 32-bit magic number to unambiguously determine this is about supplied proxy data: 0xFAA8 0xFAA8

How 0-RTT proxying is possible? Explicit Mode

- **How supplied data is structured?**
  - TLV format
  - Does not consume any MPTCP code point
  - Multiple elements can be supplied

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +---------------------------------------------------------------+
   |                 Magic Number("0xFAA8 0xFAA8")                 |
   +---------------+---------------+-----------------------------+-+-+
   |     Type      |     Length    |           Reserved          |D|M|
   +---------------+---------------+-----------------------------+-+-+
   |          Address (IPv4 - 4 octets / IPv6 - 16 octets)         |
   +-------------------------------+-------------------------------+
   |   Port (2 octets, optional)   |
   +-------------------------------+
```

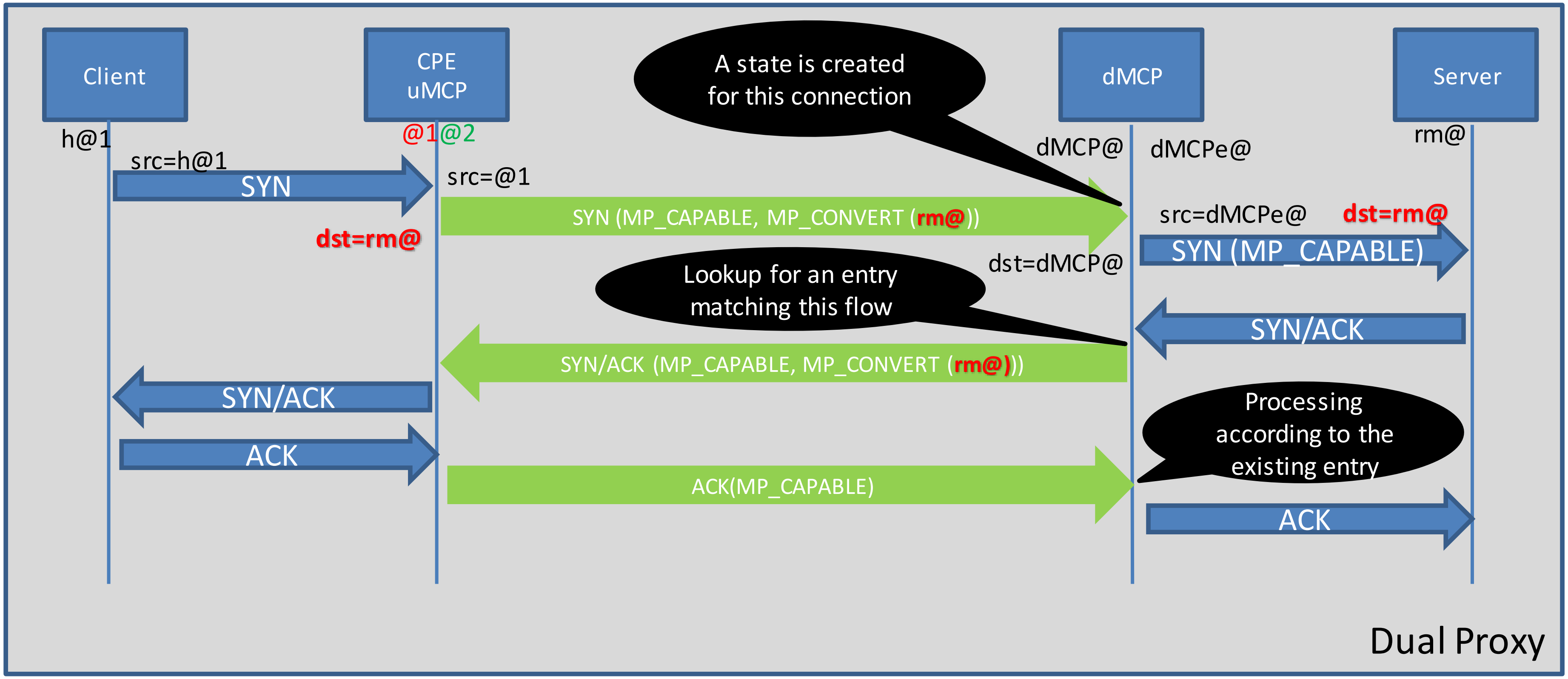
Type 0 is defined. New types can be defined in the future, if needed.

source/destination IP address/port

More bit. Must be set for the last MP_CONVERT IE

How 0-RTT proxying is possible? Explicit Mode

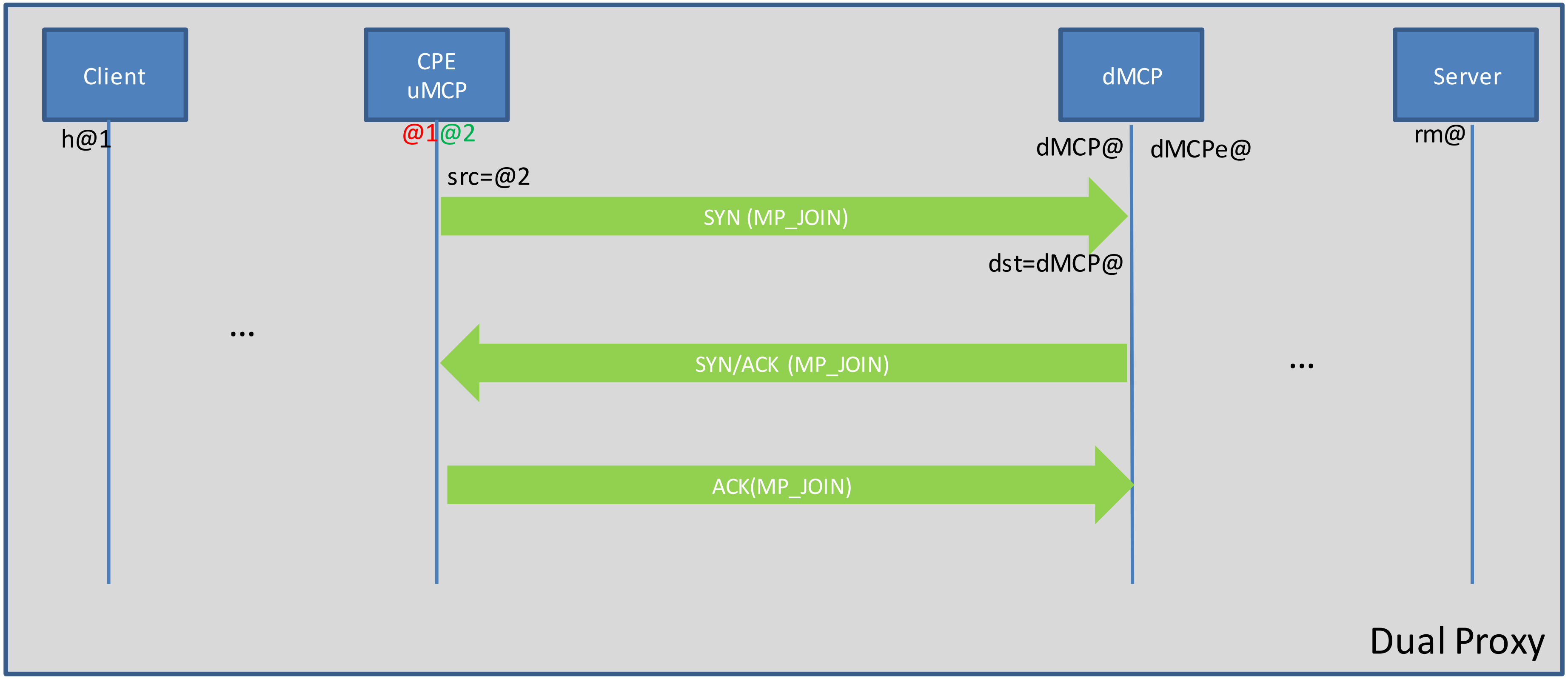
- ***Initial subflow***



Client
CPE uMCP
A state is created for this connection
dMCP
Server
h@1
@1@2
dMCP@
dMCPe@
rm@
src=h@1
SYN
src=@1
SYN (MP_CAPABLE, MP_CONVERT (rm@))
src=dMCPe@
dst=rm@
dst=rm@
SYN (MP_CAPABLE)
dst=dMCP@
Lookup for an entry matching this flow
SYN/ACK
SYN/ACK (MP_CAPABLE, MP_CONVERT (rm@))
SYN/ACK
Processing according to the existing entry
ACK
ACK(MP_CAPABLE)
ACK
Dual Proxy

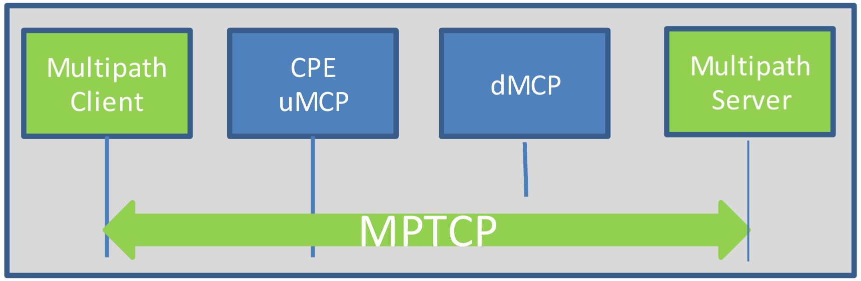
How 0-RTT proxying is possible? Explicit Mode

- *Subsequent subflows:* ***Normal MPTCP behavior is followed***

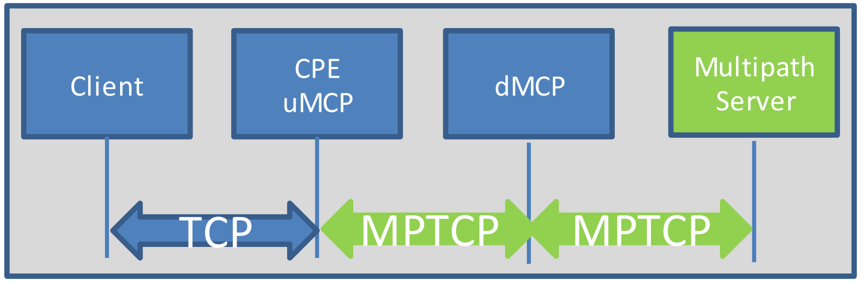


Encourage End-to-End MPTCP Connections

- A policy can be provisioned on the CPE so that native MPTCP connections ***are not proxyied***
  - Deployment-specific



- The downstream ***MCP must not strip MP_CAPABLE*** from the SYN segments it forwards to the server

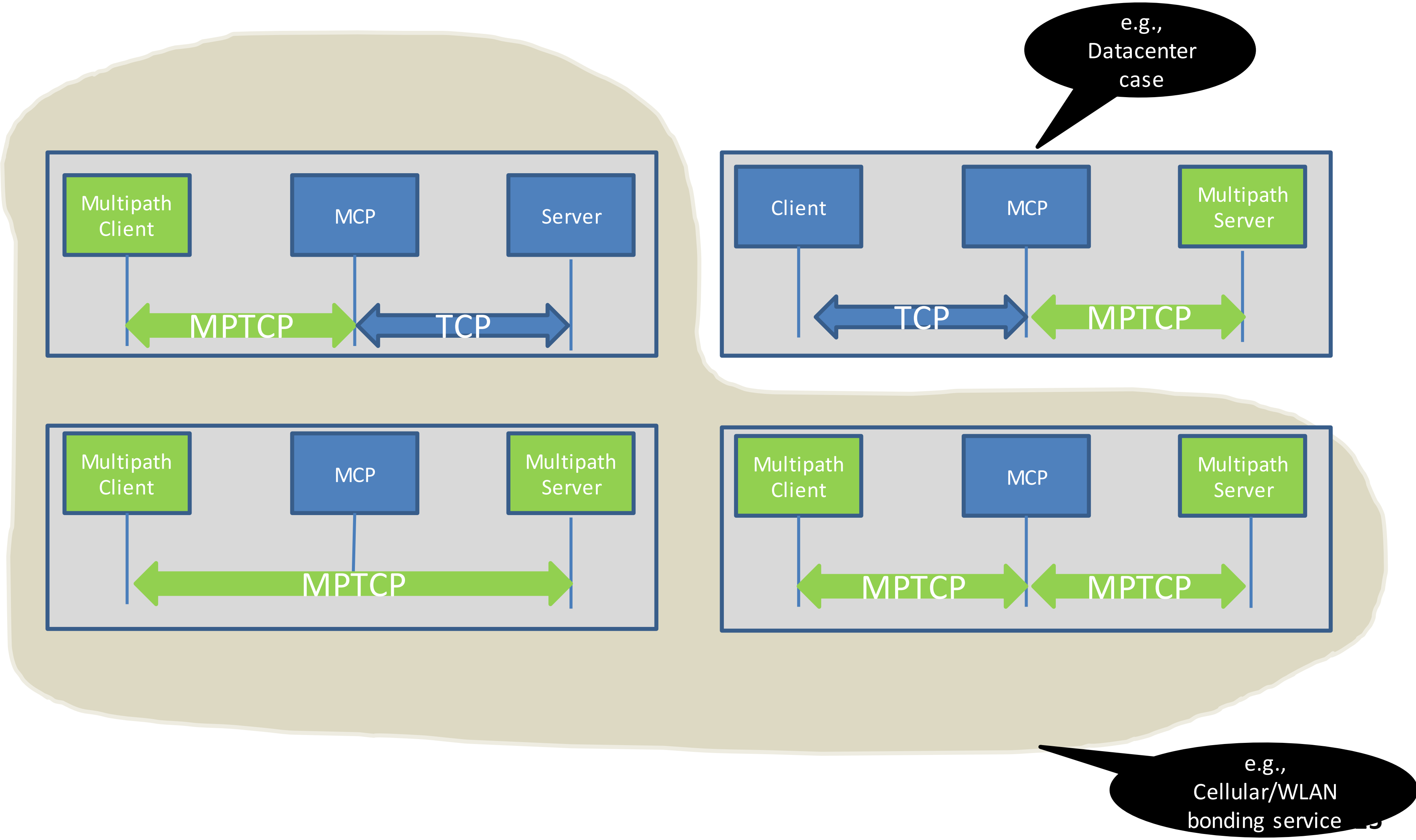


Recap

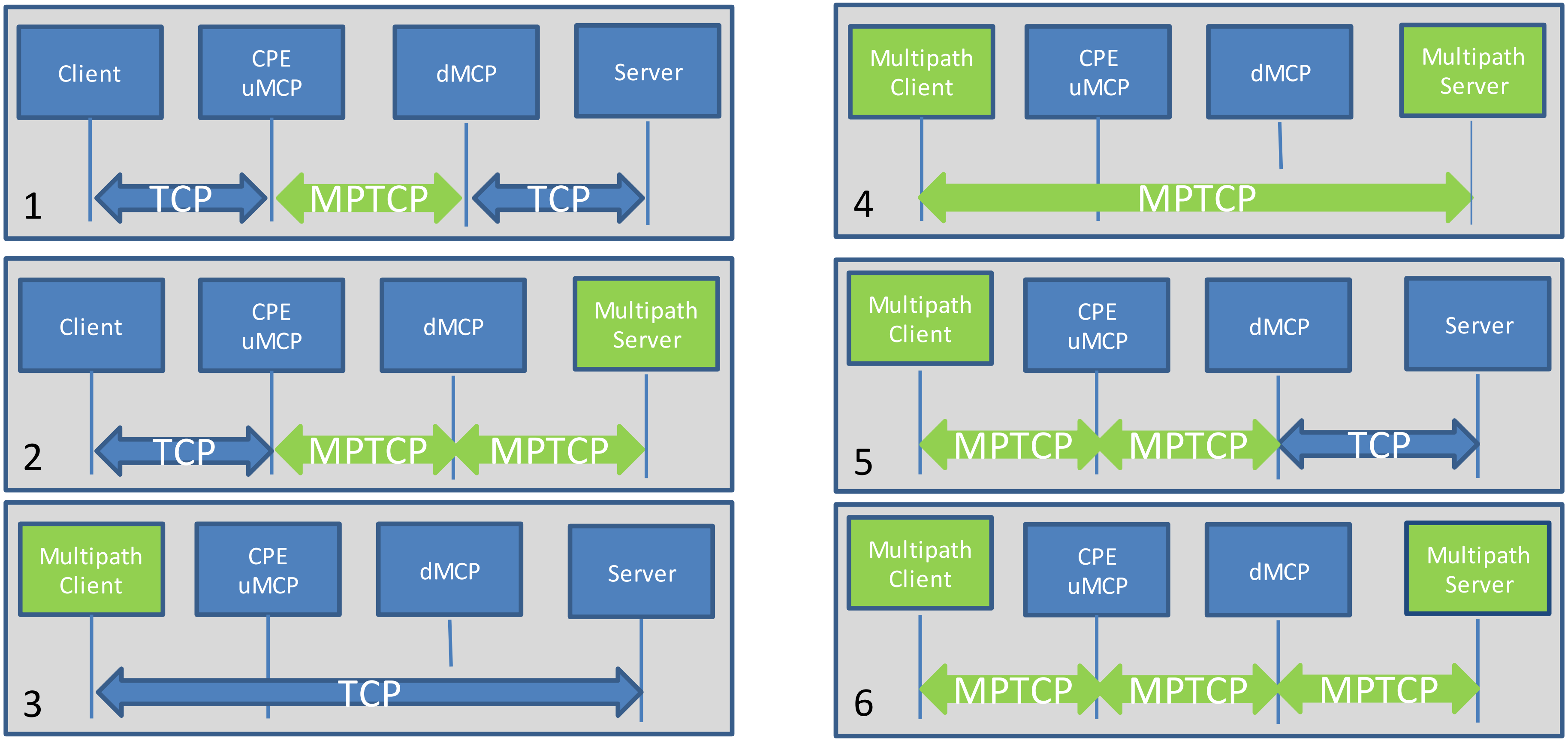
- 0-RTT
- No tunnels, no encapsulation
- No change to the base MPTCP specification
- Provides resource pooling and resilience
- Accommodates various deployment schemes
- Builds on security BCPs: ingress filtering, mitigation against SYN flood attacks, rate-limit flows/state creation, etc.
- Preserves privacy: no sensitive information is leaked
- Encourages end-to-end MPTCP
  - Supports MCP exit strategy
  - MP_PREFER_PROXY allows clients to indicate whether a connection is to be proxyied or not
- Extensible

Appendix

Target Communication Segments: Single Proxy



(Some) Target Communication Segments: Dual Proxy



1
Client
CPE uMCP
dMCP
Server
TCP
MPTCP
TCP

2
Client
CPE uMCP
dMCP
Multipath Server
TCP
MPTCP
MPTCP

3
Multipath Client
CPE uMCP
dMCP
Server
TCP

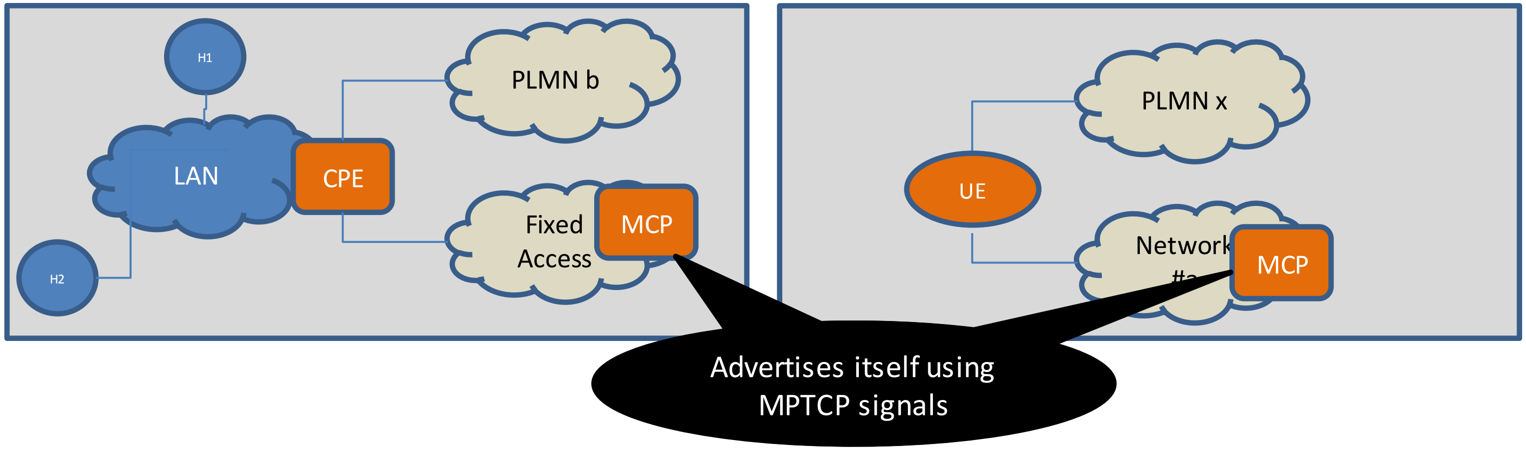
4
Multipath Client
CPE uMCP
dMCP
Multipath Server
MPTCP

5
Multipath Client
CPE uMCP
dMCP
Server
MPTCP
MPTCP
TCP

6
Multipath Client
CPE uMCP
dMCP
Multipath Server
MPTCP
MPTCP
MPTCP

How MCPs are inserted in an outbound connection?

- ***Implicit Mode***: an MCP is positioned on a default forwarding path
- The initial subflow must be placed over that path
- Inspects all TCP traffic to determine MPTCP connections
- Then, it advertises itself to a peer by means of MP_JOIN or ADD_ADDR

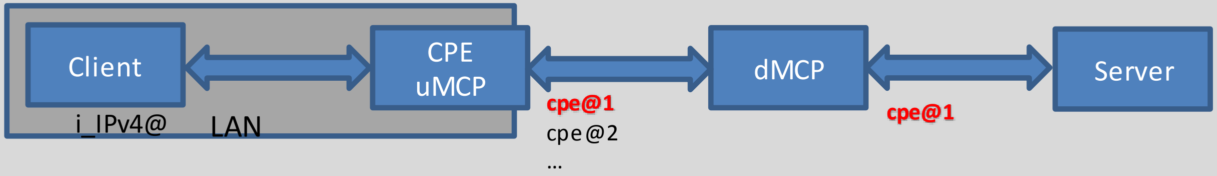


Advertises itself using MPTCP signals

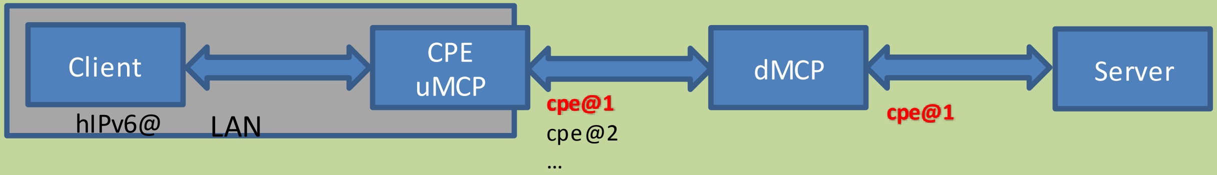
Transparent MCPs

- Preserves the source IP address/prefix of the CPE/UE
  - That is, packets sent by the MCP are sourced with an IP address/prefix that belongs to the CPE/UE
  - **Applies for both Implicit and Explicit modes**
- Various configurations are supported

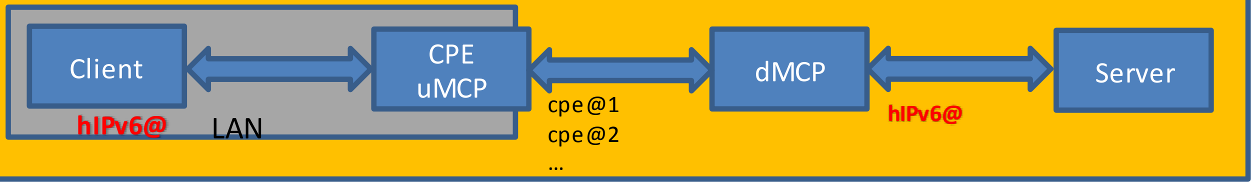
IPv4 source address preservation



IPv6 source prefix preservation



IPv6 source address preservation

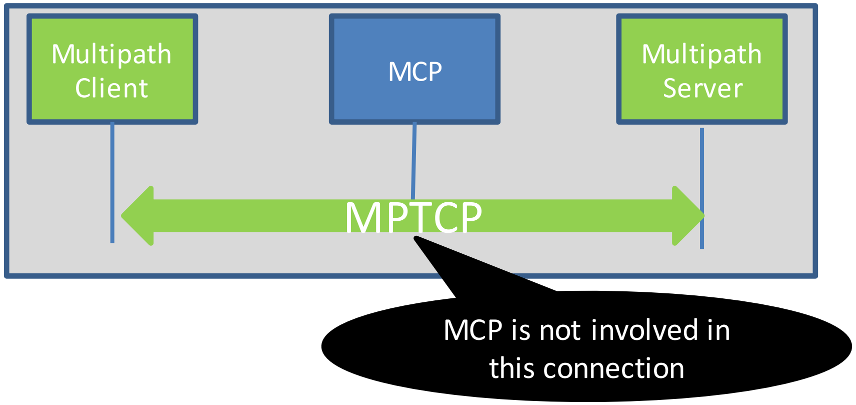


Non-transparent MCPs

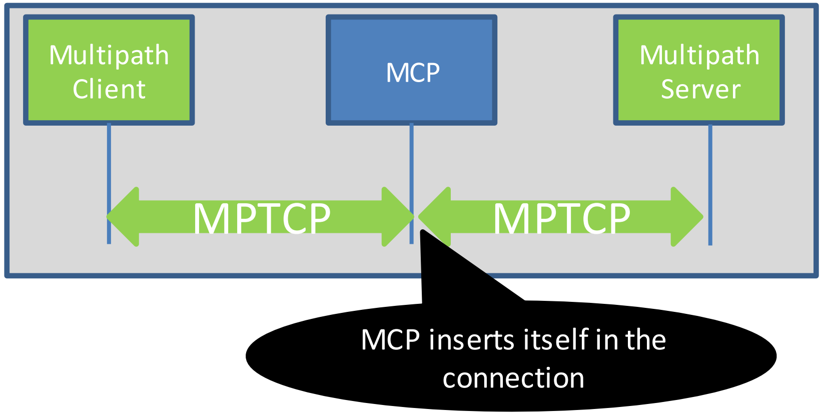
- Requires IP address pool(s) to be provisioned to the MCP
  - Packets sent to the Internet are sourced with an IP address from this pool
  - Both IPv4 and IPv6 pools may be configured
- Several configurations can be supported
  - IPv4 address sharing (N:1)
  - 1:1 address translation
  - IPv6 Network Prefix Translation (NPTv6)
- Straightforward for an MCP to intercept incoming packets
- **Applies only for the explicit mode**

Encourage End-to-End MPTCP Connections

- The MCP must not strip MP_CAPABLE from the SYN segments it forwards to the server
- Whether an MCP must be maintained in the processing of an MPTCP connection that involve MPTCP-capable client and server is a configurable parameter
  - **PROPOSED DEFAULT**: Maintain the MCP in the communication



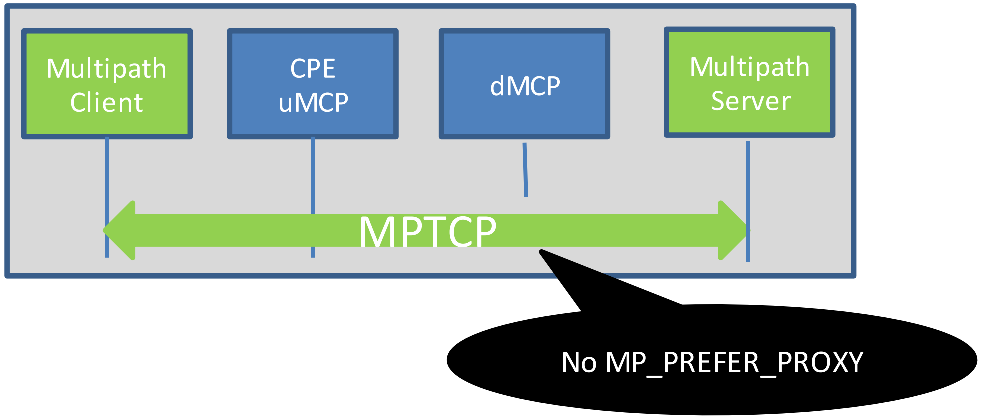
MCP is not involved in this connection



MCP inserts itself in the connection

Encourage End-to-End MPTCP Connections

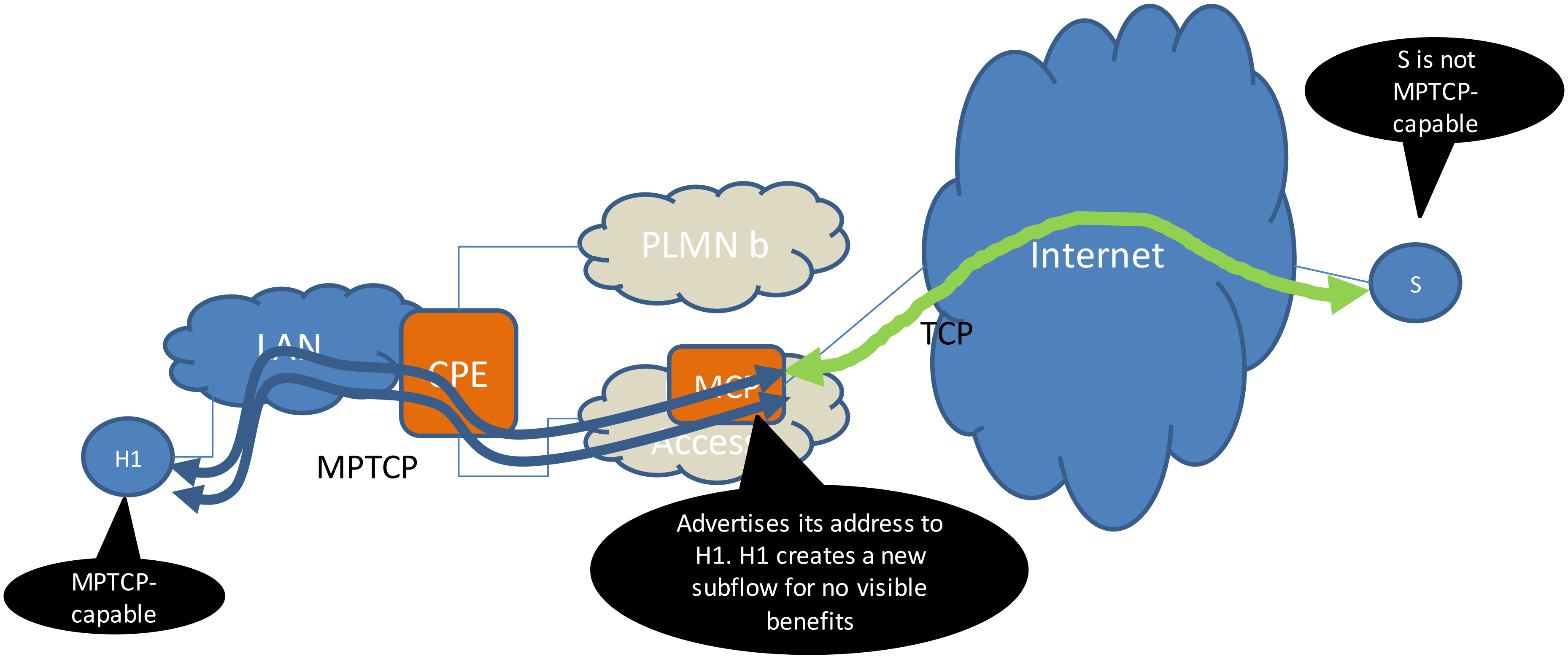
- **IMPLICIT Mode:** An MCP does only intervene in MPTCP connections that include MP_PREFER_PROXY signal
  - This signal may be set by the UE or by an MCP
  - MP_PREFER_PROXY is included in the initial SYN (MP_CAPABLE)



- Operators want to reserve MCP resources to proxyied connections

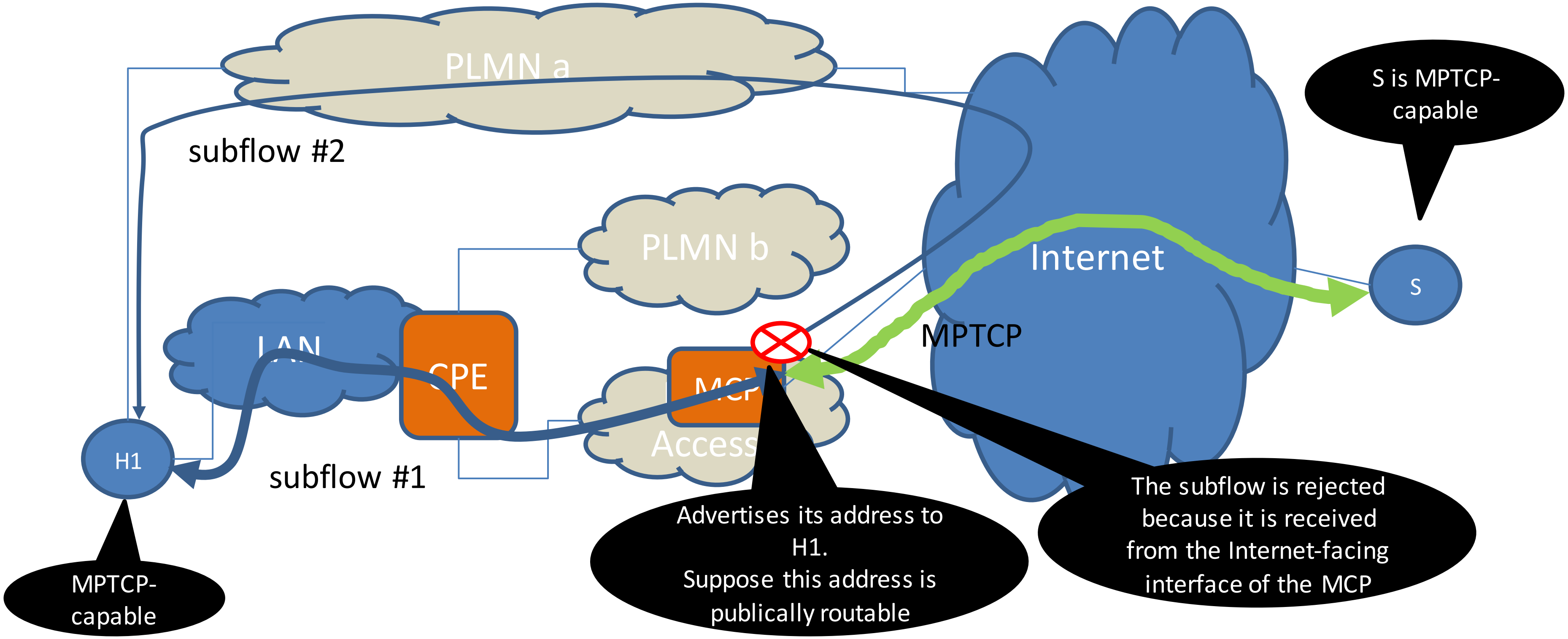
Identify Native Connections

- What if MP_PREFER_PROXY is not supported?


S is not MPTCP-capable
PLMN b
Internet
S
LAN
CPE
MCP
TCP
Access
H1
MPTCP
Advertises its address to H1. H1 creates a new subflow for no visible benefits
MPTCP-capable

Identify Native Connections

- What if MP_PREFER_PROXY is not supported?

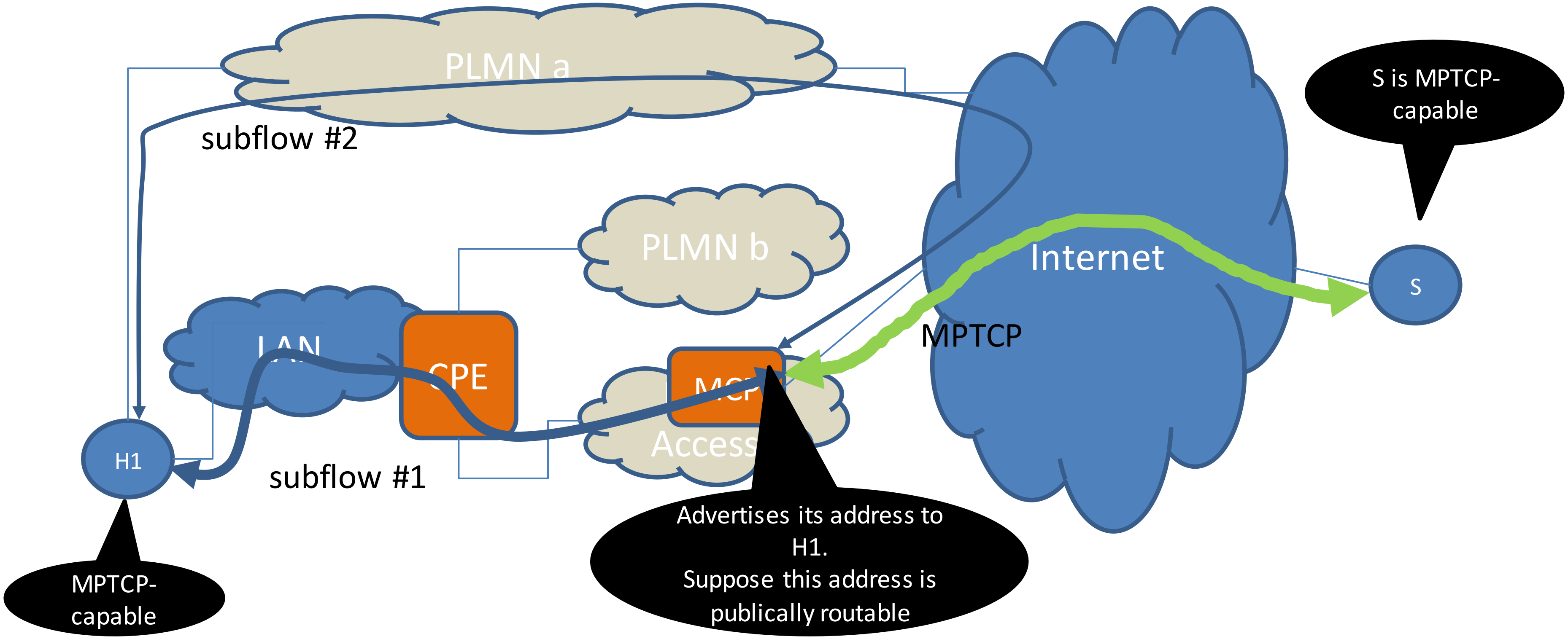


Advertises its address to H1.
Suppose this address is publically routable

The subflow is rejected because it is received from the Internet-facing interface of the MCP

Identify Native Connections

- What if MP_PREFER_PROXY is not supported?



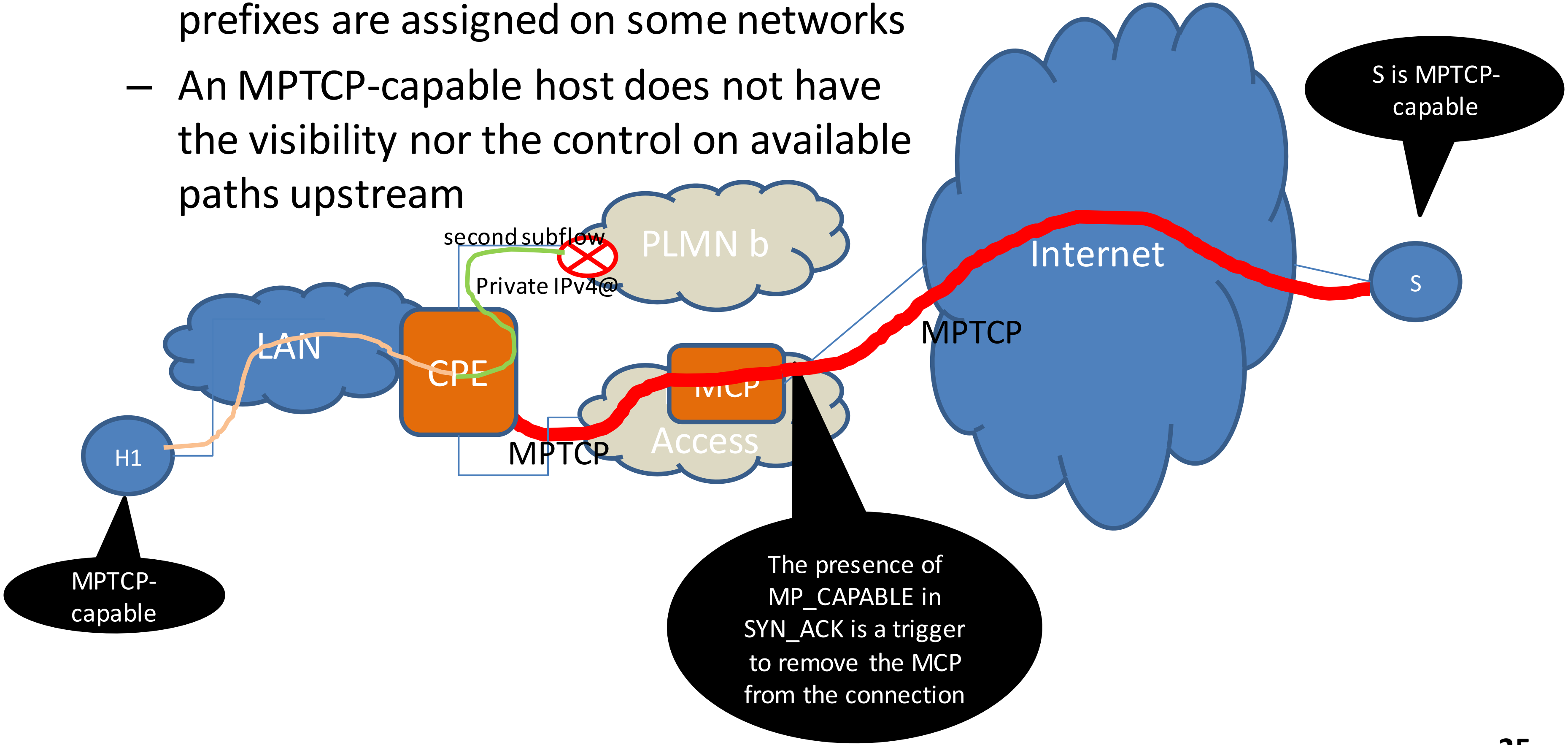
S is MPTCP-capable

MPTCP-capable

Advertises its address to H1.
Suppose this address is publically routable

Encourage End-to-End MPTCP Connections

- Blindly removing the MCP may be problematic
  - Issues with addressing: private IPv4, IPv4-only server while IPv6-only prefixes are assigned on some networks
  - An MPTCP-capable host does not have the visibility nor the control on available paths upstream



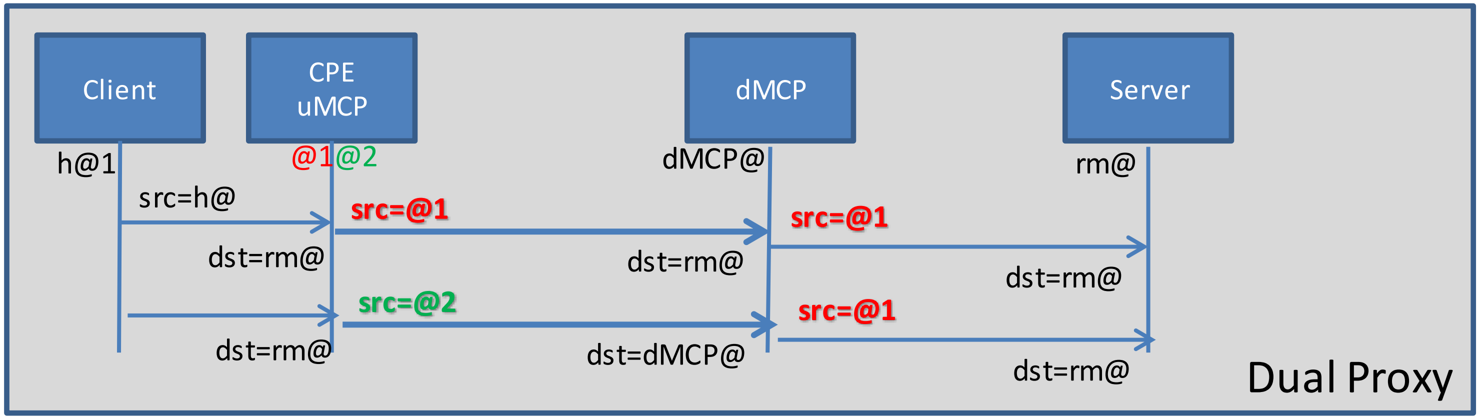
S is MPTCP-capable

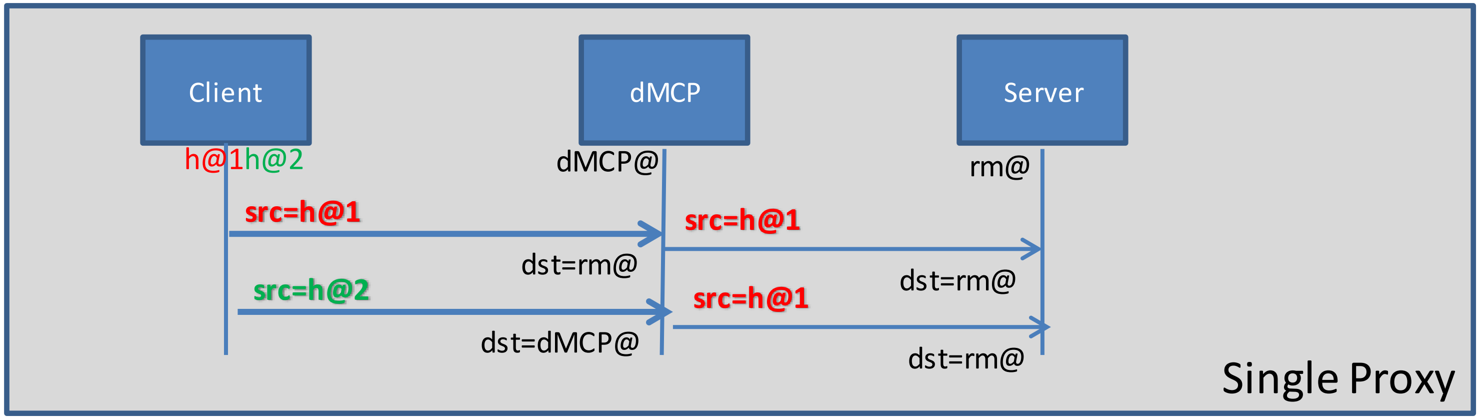
MPTCP-capable

The presence of MP_CAPABLE in SYN_ACK is a trigger to remove the MCP from the connection

How 0-RTT proxying is possible? Implicit Mode

- ***Intrinsic*** to the implicit mode



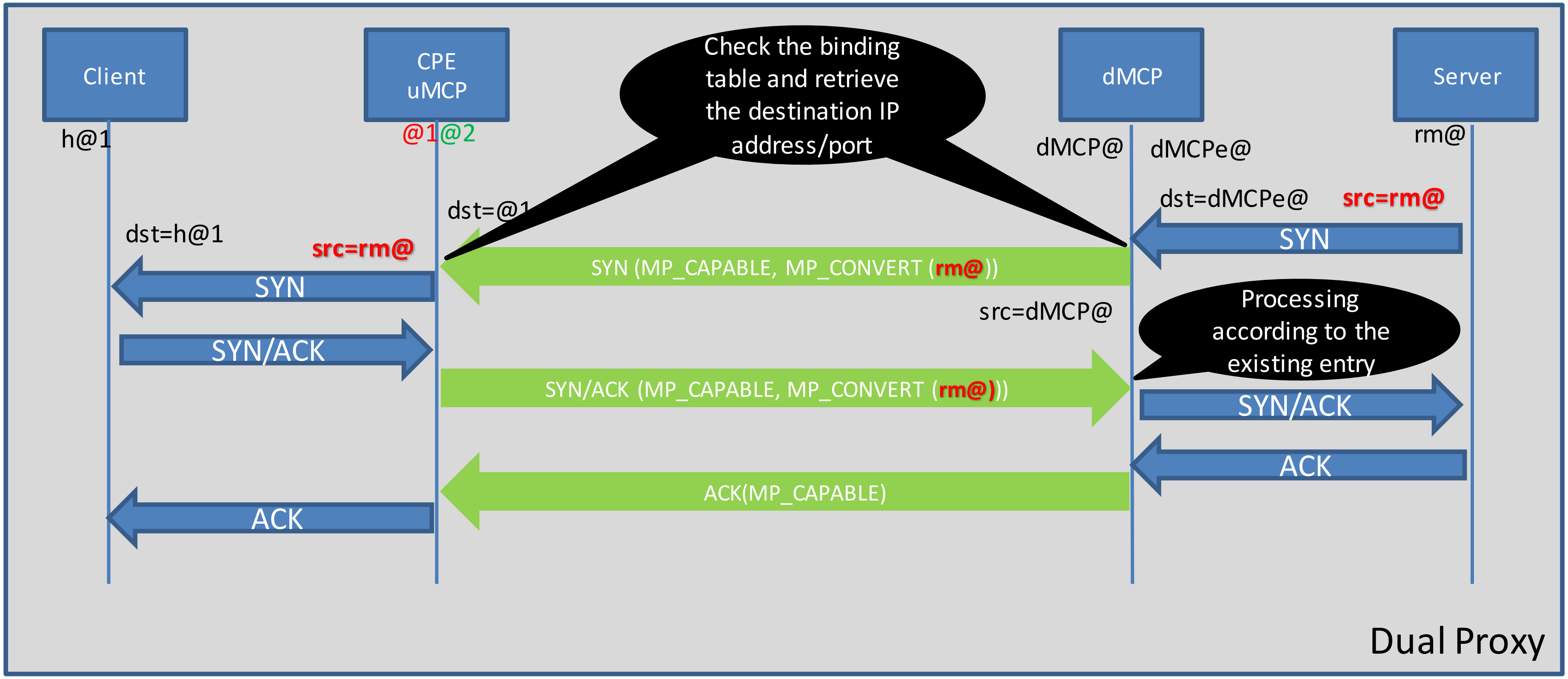


How 0-RTT proxying is possible? Explicit Mode

- **How to prevent leaking data to MP_CONVERT-unaware servers?**
  - It is likely that the provisioned MCP is compatible with the service design… but misconfiguration may happen
  - MCP must strip MP_CONVERT when forwarding upstream
  - To strengthen the procedure with means to detect misconfiguration, the behavior is as follows
    - Insert MP_CONVERT IEs (type 0) in a SYN
    - If the remote MCP supports the procedure, it MUST echo MP_CONVERT IE MP_CONVERT IEs (type 0) in the SYN/ACK
    - If no MP_CONVERT IEs (type 0) is echoed, that MCP MUST NOT be used for subsequent MPTCP assisted connections, till a new network attachment is detected, the device gets a new IP address/prefix, TTL expired, …

How 0-RTT proxying is possible? Explicit Mode

- ***Initial subflow***


Client
CPE
uMCP
dMCP
Server
h@1
@1@2
dMCP@
dMCPe@
rm@
Check the binding table and retrieve the destination IP address/port
dst=dMCPe@
src=rm@
SYN
dst=@1
dst=h@1
src=rm@
SYN (MP_CAPABLE, MP_CONVERT (rm@))
SYN
src=dMCP@
Processing according to the existing entry
SYN/ACK
SYN/ACK (MP_CAPABLE, MP_CONVERT (rm@)))
SYN/ACK
ACK
ACK(MP_CAPABLE)
ACK
Dual Proxy

Authorization

- Deployment-specific
- Some samples are (non-exhaustive list):
  - Use a **dedicated APN + filter based on the IMSI**. This method does not require any interaction with the MCP
  - **Access Control Lists (ACLs)**, e.g., at a Broadband Network Gateway (BNG) to control authorized subscribers. These ACLs may be installed as a result of **RADIUS** exchanges for instance ([I-D.boucadair-mptcp-radius]). This method does not require any interaction with the MCP
  - The device that embeds the **MCP may also host a RADIUS client** that will solicit an AAA server to check whether connections received from a given source IP address are authorized or not ([I-D.boucadair-mptcp-radius])
- Future MP_CONVERT types may be defined in the future for authorization purposes

Further Considerations

- Fragmentation
  - Unlikely; only the initial SYN packet is augmented (explicit mode)
  - MSS clamping can be used if needed
- Flows eligible to network-MPTCP assisted service
  - Deployment and policy-based
- DSCP preservation is supported
- Exhausted TCP option space in the original SYN
  - **DEFAULT**: No MPTCP options are inserted

MCP for DCs

- When an MCP is inserted, the original source IP address/port may be lost
  - The source IP address may be used for abuse, logging, policy enforcement, etc.
- HOST_ID TCP option (RFC7974) can be used to solve that problem
- No further extension is required

Why not MPTCP+SOCKS?

- Too chatty
- Extra delay to setup subflows
  - several tens of ms
- Need for UPnP IGD-SOCKS interworking

```
(MP Client) ->    TCP SYN    ->  (MCP)
            <- TCP SYN/ACK <-
            ->    TCP ACK    ->
            -> SOCKS Method Request (1)(a) ->
            <-    TCP ACK (b)   <-
            <- SOCKS Method Response (2)(c)  <-
            ->   TCP ACK (d)   ->
            -> SOCKS Authentication Request  (3)(e) ->
            <-    TCP ACK (f)    <-
            <- SOCKS Auth. Response (4)(g) <-
            ->   TCP ACK (h)   ->
            -> SOCKS Connection Request (5)(i)  -> (MCP)
            <-   TCP ACK (j)                    <- (MCP)
                                                   (MCP)  -> TCP SYN (k) -> (Server)
                                                   (MCP)  <- SYN/ACK (l) <- (Server)
            <- SOCKS Connection Response (n) (6) <-(MCP) -> TCP ACK (m) -> (Server)
            ->   TCP ACK (o)  ->
```