

MIF WG
Internet-Draft
Intended status: Informational
Expires: February 15, 2014

P. Seite
Orange
JC. Zuniga
InterDigital Communications, LLC
August 14, 2013

MIF API for Connection Management
draft-seite-mif-cm-02.txt

Abstract

There is currently a need to present a coherent connection management behaviour for different terminal platforms (e.g. mobile phones, PCs, tablets, etc.). This document discusses how a connection manager can use the MIF API to provide this coherent behaviour and enhance the end user's experience when a terminal is able to connect to multiple interfaces. The goal of this document is not to define a connection manager specification, but to focus on the interaction with the MIF API and suggest relevant generic messages for the interface.

This document is for discussion and its intention is to help clarifying the utilization of the MIF API in a connection management context and propose some relevant considerations.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 15, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
2.	Architecture of a MIF terminal	2
3.	Use-case	3
4.	Functions of the connection manager	5
5.	Security Considerations	9
6.	IANA Considerations	9
7.	Acknowledgements	9
8.	References	9
8.1.	Normative References	9
8.2.	Informative References	9
	Authors' Addresses	10

[1.](#) Introduction

[I-D.ietf-mif-api-extension] describes an abstract API that provides commands and services for applications and higher layer APIs running on a terminal with more than one interface. There is currently a need to present a coherent connection management behaviour for different terminal platforms (e.g. mobile phones, PCs, tablets, etc.), as users often experience a very different behaviour when connecting with various platforms to the same networks and for the same purposes (e.g. web browsing, email access, dedicated applications, etc.). This document builds on top of the MIF API and aims to discuss how connection managers can use the MIF API to provide a coherent and constant behaviour to the users. The goal of this document is not to define a connection manager specification, but to focus on the interaction with the MIF API and suggest relevant generic messages for the interface.

This document is only for discussion; its intention is to help clarifying the utilization of the MIF API in a connection management

context.

2. Architecture of a MIF terminal

The terminal's MIF API based architecture is an instantiation of the MIF API model described in [[I-D.ietf-mif-api-extension](#)]; main functions and APIs are described below:

- o MIF API: it provides information as per [[I-D.ietf-mif-api-extension](#)].
- o Connection Manager: it is an application relying on the MIF API; this application acts on behalf of other running applications. The connection manager decides about the mapping between IP flows and interfaces, then configures the IP stack, and lower layers, accordingly, e.g. configuration of the routing table. The connection manager relies on information provided either by the MIF API or the OS API. Only interface with the MIF API is in the scope of this document.
- o OS API: Provides the interface to manipulate IP object configuration, e.g. routing table.

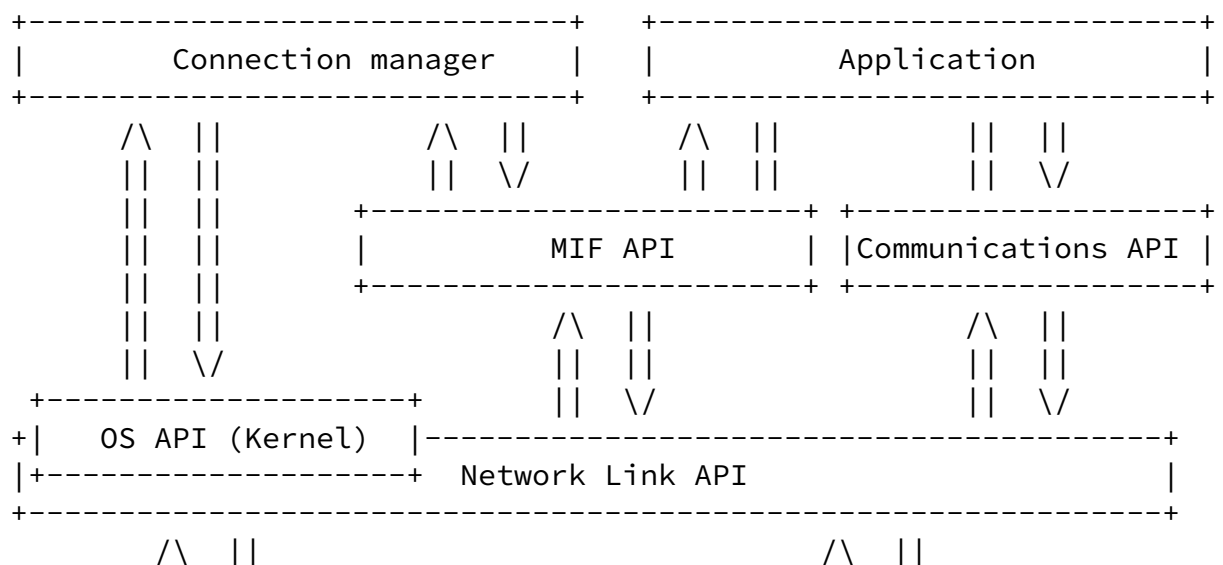




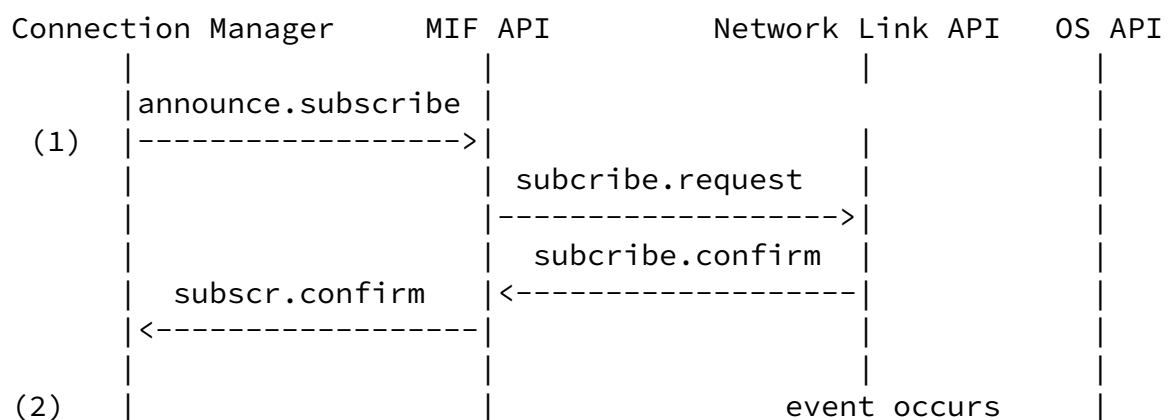
Figure 1: MIF API framework

3. Use-case

The presented use-case aims to illustrate the behaviour of the MIF API in a concrete situation. The use-case is as follows:

1. Multiple IP communications are running simultaneously; each can be mapped to different interfaces/provisioning domains.
2. The connection manager selects the appropriate interface/provisioning domain for the application; making a decision according to various criteria (information provided by the MIF and lower layers APIs, user preferences, and so on).

The interaction between the different APIs is depicted in Figure 2. It is assumed that at least one IP communication is running. Then, an interface event occurs and the connection manager decides to move the communication to a different interface.



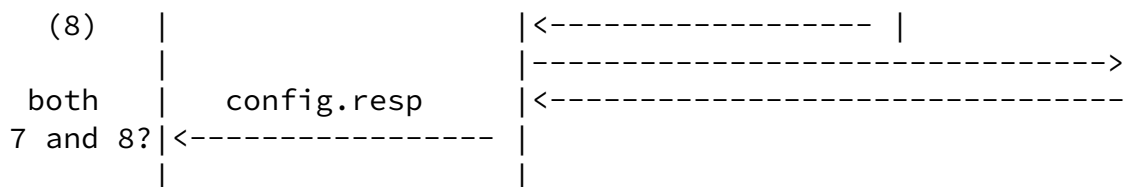


Figure 2: APIs interaction

Operations are as follows:

1. The connection manager subscribes to the MIF API notifications [[I-D.ietf-mif-api-extension](#)].
2. An event, to which the connection manager has subscribed, occurs; e.g. a new interface becomes available or a low radio signal level is crossed.
3. The connection manager is notified about the event.

4. In order to take its decision, the connection manager gets some configuration information from the MIF API.
5. The connection manager fetches additional information from the OS API
6. The connection manager decides to move the ongoing IP communication to another interface.
7. The connection manager requests the OS API to reconfigure one or multiple interfaces according to the decision; for example, the connection manager could request reconfiguration of the routing table or trigger a MIP operation.

4. Functions of the connection manager

This section focuses on the interactions between the connection manager and the MIF API and OS API. The interactions between the connection manager and other complementary APIs, like user preferences and/or ANDSF network operator policies are out of the scope of this document.

A connection manager may also rely on different abstraction layers together with the MIF API. The IEEE 802.21 MIH SAP [[IEEE802.21](#)] is an example of such an abstraction layer, which can be seen as a partial instantiation of the MIF API. A companion document

[I-D.zuniga-mif-802-21-overview] addresses interaction between IEEE 802.21 and MIF API.

Generic connection manager functions and their relation to the MIF API are described below. The following assumes the MIF API is the unique API for any manipulation of IP objects. However, current MIF API allows to gather information from the IP stack but not to configure it. As a consequence three functions are added to the MIF API: GetIPType(), ConfigFlowRouting() and SetSourceAddress().

Subscribe(eventID)

Description: register for a MIF API event notification, e.g. WLAN scan results ready, WLAN connected, WLAN disconnected, interface is going to be disconnected detected (e.g. because of low radio

signal level detected), Cellular connected, Cellular disconnected, etc.

Input: identifier of the event to be notified. Some events are defined in [[I-D.ietf-mif-api-extension](#)]

API: MIF API

UnSubscribe(eventID)

Description: unregister to a MIF API notification.

Input: identifier of event. Some events are defined in [[I-D.ietf-mif-api-extension](#)]

API: MIF API

ListInterfaces()

Description: return the list of available interfaces with their characteristics. Interfaces may have different access technologies.

Input: n/a

API: MIF API

ListProvisioningDomains()

Description: return the list of available provisioning domains with their characteristics.

Input: n/a

API: MIF API

GetStatus(IID)

Description: provide the status of the interface, e.g. enabled/disabled, active, idle, connection failed, connecting, disconnecting, scanning, unknown state, etc.

Input:Interface Identifier

API: MIF API

IPconnectivityCheck(PID, IP[])

Description: check IP connectivity to the intranet/Internet: the interface may have a valid IP address but no IP connectivity to data networks (e.g. web based authentication through a captive portal).

Input: Provisioning domain Identifier, IP addresses to be tested

API: MIF API

GetConfiguration(IID)

Description: retrieve layer 2 configuration information for a given interface.

Input: Interface Identifier

API: OS API

SetConfiguration(IID)

Description: configures an interface, e.g. enable/disable, scan, etc.

Input: Interface Identifier

API: OS API

GetConfiguration(PID)

Description: retrieve configuration information for a given provisioning domain(IP address(es), DNS, default gateway, authentication method, associated interface(s))

Input: Provisioning domain Identifier

API: MIF API

SetConfiguration(PID)

Description: configure provisioning domain information (IP adresse(s), default gateway, authentication method, associated interface, routing table, etc.)

Input: Provisioning domain Identifier

API: MIF API

GetTheoriticalQoS(IID)

Description: provide information on the theoretical interface capabilities (e.g. upload/download speed)

Input: Interface Identifier

API: MIF API

GetAvailableQoS(IID)

Description: provide information on the quality of communication (Jitter, delay, average upload data rate, average Download data rate, signal strength, etc.)

Input: Interface Identifier

API: MIF API

GetIPtype(IP address)

Description: return the type of address and properties (e.g. local, remote, mobile IP anchored, etc.). This function is to be added to the MIF API as per [[I-D.ietf-mif-api-extension](#)]. [[I-D.korhonen-dmm-prefix-properties](#)].

Input: IP address

API: updated MIF API

ConfigFlowRouting(ROUTE, FlowID)

Description: associate a route, ROUTE, to the IP flow identified by FlowID, e.g. as defined in [[RFC6088](#)]. This function is to be added to the MIF API as per [[I-D.ietf-mif-api-extension](#)].

Input: routing table identifier, flow identifier

API: updated MIF API

SetSourceAddress(IP, FlowID)

Description: influence source address selection for a given IP flow. This function is to be added to the MIF API as per [\[I-D.ietf-mif-api-extension\]](#).

Input: IP source address, flow identifier

API: updated MIF API

[5.](#) Security Considerations

TBD.

[6.](#) IANA Considerations

This document has no actions for IANA.

[7.](#) Acknowledgements

The authors would like to express their gratitude to Ralph Droms, Ted Lemon and Dave Thaler for the fruitful discussions regarding MIF API and connection managers.

[8.](#) References

[8.1.](#) Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC6088] Tsirtsis, G., Giarreta, G., Soliman, H., and N. Montavont, "Traffic Selectors for Flow Bindings", [RFC 6088](#), January 2011.
- [RFC6089] Tsirtsis, G., Soliman, H., Montavont, N., Giaretta, G., and K. Kuladinithi, "Flow Bindings in Mobile IPv6 and Network Mobility (NEMO) Basic Support", [RFC 6089](#), January 2011.

[8.2.](#) Informative References

Internet-Draft

MIF API for Connection Management

August 2013

Deng, H., Krishnan, S., Lemon, T., and M. Wasserman,
"Guide for application developers on session continuity by
using MIF API", [draft-deng-mif-api-session-continuity-
guide-03](#) (work in progress), October 2012.

[I-D.ietf-mif-api-extension]

Liu, D., Lemon, T., and Z. Cao, "MIF API consideration",
[draft-ietf-mif-api-extension-03](#) (work in progress),
November 2012.

[I-D.ietf-netext-logical-interface-support]

Melia, T. and S. Gundavelli, "Logical Interface Support
for multi-mode IP Hosts", [draft-ietf-netext-logical-
interface-support-07](#) (work in progress), April 2013.

[I-D.korhonen-dmm-prefix-properties]

Korhonen, J., Patil, B., Gundavelli, S., Seite, P., and D.
Liu, "IPv6 Prefix Mobility Management Properties", [draft-
korhonen-dmm-prefix-properties-03](#) (work in progress),
October 2012.

[I-D.zuniga-mif-802-21-overview]

Zuniga, J. and P. Seite, "IEEE 802.21 Overview", [draft-
zuniga-mif-802-21-overview-00](#) (work in progress), February
2013.

[IEEE802.21]

IEEE, "IEEE Standard for Local and Metropolitan Area
Networks - Part 21: Media Independent Handover Services",
IEEE LAN/MAN Std 802.21-2008, January 2009.", 2009, <
[http://www.ieee802.org/21/private/Published%20Spec/
802.21-2008.pdf](http://www.ieee802.org/21/private/Published%20Spec/802.21-2008.pdf)>.

Authors' Addresses

Pierrick Seite
Orange
4, rue du Clos Courtel, BP 91226
Cesson-Sevigne 35512

France

Email: pierrick.seite@orange.com

Seite & Zuniga

Expires February 15, 2014

[Page 10]

Internet-Draft

MIF API for Connection Management

August 2013

Juan Carlos Zuniga
InterDigital Communications, LLC
1000 Sherbrooke Street West, 10th floor
Montreal, Quebec H3A 3G4
Canada

Email: JuanCarlos.Zuniga@InterDigital.com

