IBM
Shared Memory Communications:
Version 2

Fourth Edition

January 2024

CONTENTS

TABLE OF FIGURES

SMCV2 OVERVIEW FOURTH EDITION

The Fourth Edition of the *IBM SMCv2 Overview* adds the SMC Version 2.1 feature updates for SMC-Dv2 with Emulated-ISM specifications. The Fourth Edition was published in January of 2024.

SMCV2 OVERVIEW THIRD EDITION

The Third Edition of the *IBM SMCv2 Overview* adds the SMC Version 2.1 feature updates for SMC-Rv2 with RoCEv2 specifications. The Third Edition was published in July of 2023.

SMCV2 OVERVIEW SECOND EDITION

The Second Edition of the *IBM SMCv2 Overview* adds the SMC-Rv2 with RoCEv2 specifications. The Second Edition was published in September of 2021.

SMCV2 OVERVIEW FIRST EDITION

The First Edition of the *IBM SMCv2 Overview* introduces the next version of SMC, referred to as SMC Version 2 (SMCv2). SMCv2 provides Shared Memory Communications capability over multiple IP subnets. The SMCv2 specifications for SMC-Dv2 and ISMv2 are provided in the first edition.

ABSTRACT

This paper is provided for IBM®, IBM customers and vendors who have an interest in the IBM Shared Memory Communications (SMC) protocol and related solutions such as IBM z/OS® Shared Memory Communications, IBM Z® Internal Shared Memory, and IBM RoCE Express features.

The purpose of this paper is to:

1. Provide a brief high-level review of SMC in its original form. In this paper, SMC in its original form is referred to as SMC Version 1 (SMCv1)[1]. SMCv1 connectivity is limited to hosts that are directly attached to a common single IP subnet.

2. Introduce and provide the specifications of the next version of SMC, referred to as SMC Version 2 (SMCv2). SMCv2 provides Shared Memory Communications capability over multiple IP subnets.

3. Introduce and provide the specifications of the first version and release of SMCv2 defined as SMCv2.1. The purpose of SMC version releases is to allow the introduction of minor SMC feature updates to an existing SMC version that require protocol changes without disrupting compatibility.

SMC-R is an open protocol that was initially introduced in z/OS V2R1 on the IBM zEC12. SMC-R is defined in an informational RFC entitled *IBM's Shared Memory Communications over RDMA* (https://tools.ietf.org/html/rfc7609).

SMC-D is a variation of SMC-R. SMC-D is closely related to SMC-R but is based on the Internal Shared Memory (ISM) capabilities introduced with the IBM z13® (z13) hardware model.

Although this paper makes references to specific IBM products, readers must recognize that SMC is available in other operating systems such as Linux® and potentially available on other non-IBM platforms.

The primary purpose of this paper is to introduce concepts of and an overview of the changes in the SMC specifications and the SMC protocol to provide SMCv2 multiple IP subnet support. This includes any updates to the SMCv2 protocol, such as SMCv2.1

[1] This paper introduces the term SMC Version 1 (SMCv1) to describe the original version of SMC.

TOPICS COVERED

This paper is organized into the following topics:

1. **Review - Shared Memory Communications Overview**
   Provides a brief review of Shared Memory Communications (SMC-R and SMC-D) in its original single subnet form. This portion of the paper provides sufficient background to understand why the protocol was limited to a single IP subnet. If you are looking for a more detailed discussion of SMC itself, refer to the SMC reference materials at:
   https://www.ibm.com/docs/en/zos/3.1.0?topic=communications-shared-memory-reference-information

   For the base SMC specifications, refer to the Information RFC 7609 (https://tools.ietf.org/html/rfc7609).
2. **SMCv2 (Shared Memory Communications Version 2)**
   Introduces the key concepts of SMCv2, SMC over multiple IP subnets. Updates to the SMC messages required for SMCv2 are provided in the appendix. SMCv2 reuses the existing SMCv1 (CLC and LLC) messages. For more information about the SMC message (wire) flows, refer to RFC 7609 (https://tools.ietf.org/html/rfc7609).
3. **SMC-Dv2 (Shared Memory Communications - Direct Version 2)**
   Provides an overview of how SMCv2 applies to SMC-Dv2 and the IBM Z technology that enables the multiple IP subnet support for SMC-D.
4. **SMC-Rv2 (Shared Memory Communications - RDMA Version 2)**
   Provides an overview of how SMCv2 applies to SMC-Rv2 and RoCEv2 (also known as “Routable RoCE”) that enables the multiple IP subnet support for SMC-R.
5. **SMCv2.1 (Shared Memory Communications Version 2.1) Feature Update Summary**

   Provides a general overview of the SMCv2 dot release architecture. The initial dot release of SMC Version 2.1 is introduced along with the specific v2.1 features. A summary of the changes to the SMC Version 2.1 protocol (message updates) are provided.

In some cases, references to or examples of IBM z/OS operating system are included. This paper is not intended to describe how to define or deploy SMCv2 on z/OS or any other specific operating system. For more information about defining and deploying SMCv2, refer to the product documentation for each specific operating system. The reader is assumed to have a basic understanding of network communication protocols including Ethernet, IP, and TCP.

Version terminology notes:

1. The terms SMC V1 and SMCv1 along with the terms SMC V2 and SMCv2 are used interchangeably throughout this document. Both forms of versioning terminology are acceptable.

2. The third edition of this document introduces the version 2 dot release described as Version 2.1. The dot release term is also described as *Version.Release* and shown in short form as both V2.1 or v2.1. References to the dot release architecture and rules shown in both forms (V2.1 or v2.1) of SMC version dot release (Version.Release) terminology are acceptable.

REVIEW: SHARED MEMORY COMMUNICATIONS OVERVIEW

Shared Memory Communications (SMC) allows two SMC capable peers to communicate using memory that each peer allocates and manages for their partner's use. There are two forms of Shared Memory Communications:

- SMC over Remote Direct Memory Access (SMC-R)
- SMC - Direct Memory Access (SMC-D)

Both forms of SMC allow your TCP sockets applications to benefit from direct, high-speed, low-latency, memory-to-memory (peer-to-peer) communications transparently – no changes are required in your application programs.

SMC provides services that are designed for enterprise-class data center networks. Communicating peers (the z/OS TCP/IP stacks) dynamically learn about the shared memory capability by using traditional TCP/IP connection establishment flows. With this awareness, the TCP/IP stacks can switch from TCP network flows to more efficient direct memory access or RDMA flows, as appropriate. The application programs using TCP sockets are unaware of the transition to Shared Memory Communications.

Let's take a closer look at each form of SMC.

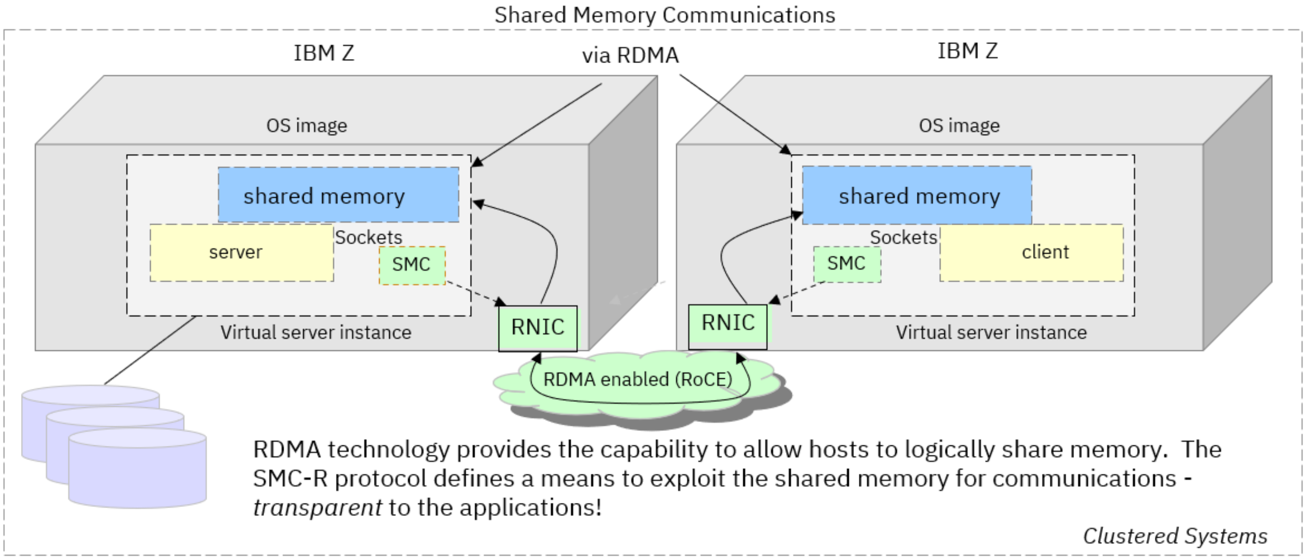

Shared Memory Communications
IBM Z
via RDMA
IBM Z
OS image
OS image
shared memory
shared memory
server
Sockets
SMC
Sockets
SMC
client
RNIC
RNIC
Virtual server instance
Virtual server instance
RDMA enabled (RoCE)
RDMA technology provides the capability to allow hosts to logically share memory. The SMC-R protocol defines a means to exploit the shared memory for communications - *transparent* to the applications!
Clustered Systems

Figure 1. SMC-R (SMC over RDMA / RoCE)

SMC-R is an open socket over RDMA protocol that provides transparent exploitation of RDMA (for TCP-based applications) while preserving key functions and qualities of service from the TCP/IP ecosystem that enterprise-level servers/network depend on! See RFC 7609 (https://tools.ietf.org/html/rfc7609) for reference.

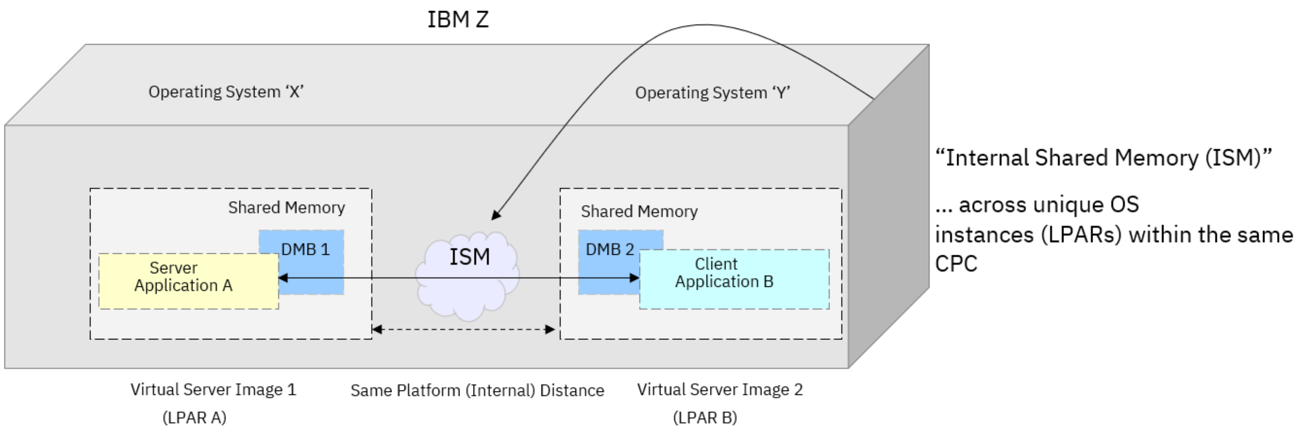


Figure 2. SMC-D (SMC Direct over ISM)

SMC-D (over ISM) extends the value of the Shared Memory Communications architecture by enabling SMC for direct LPAR to LPAR communications. SMC-D is similar to SMC-R (over RoCE) extending the benefits of SMC-R to the same CPC operating system instances without requiring physical resources (RoCE adapters, PCI bandwidth, NIC ports, I/O slots, network resources, 10 or 25 GbE switches, and so on.).

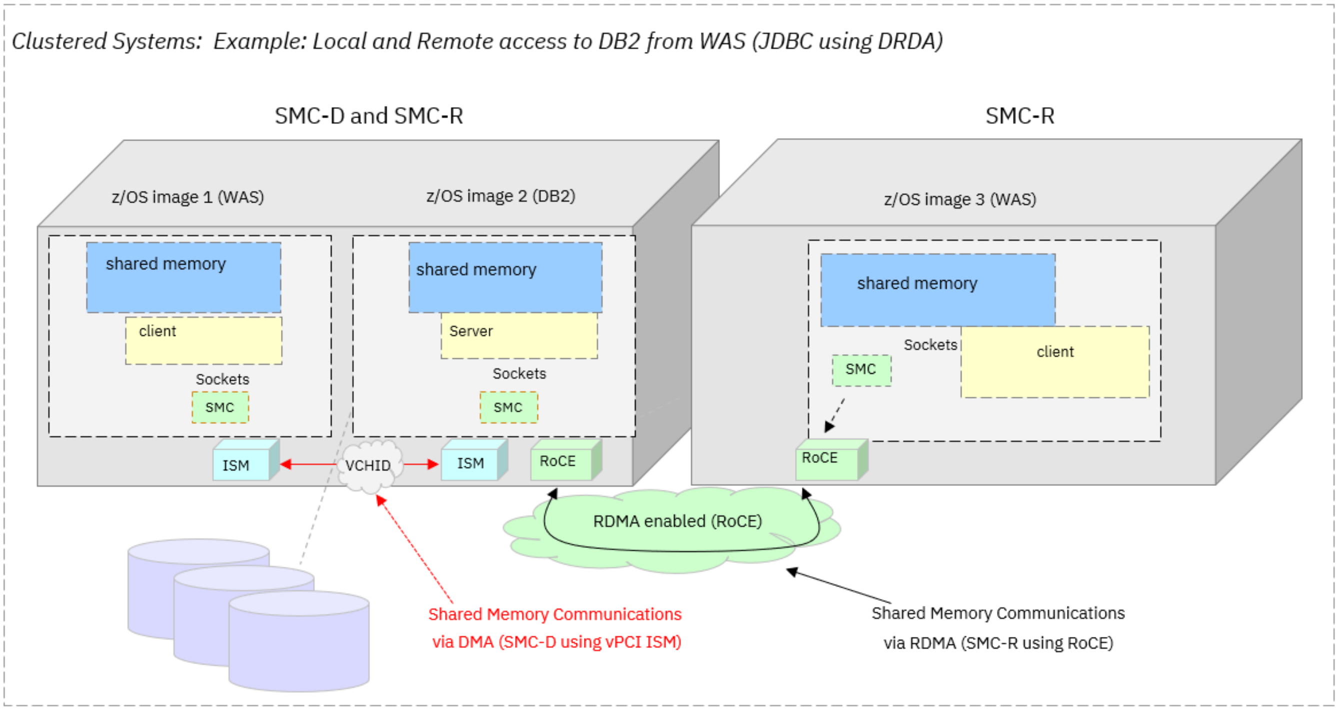
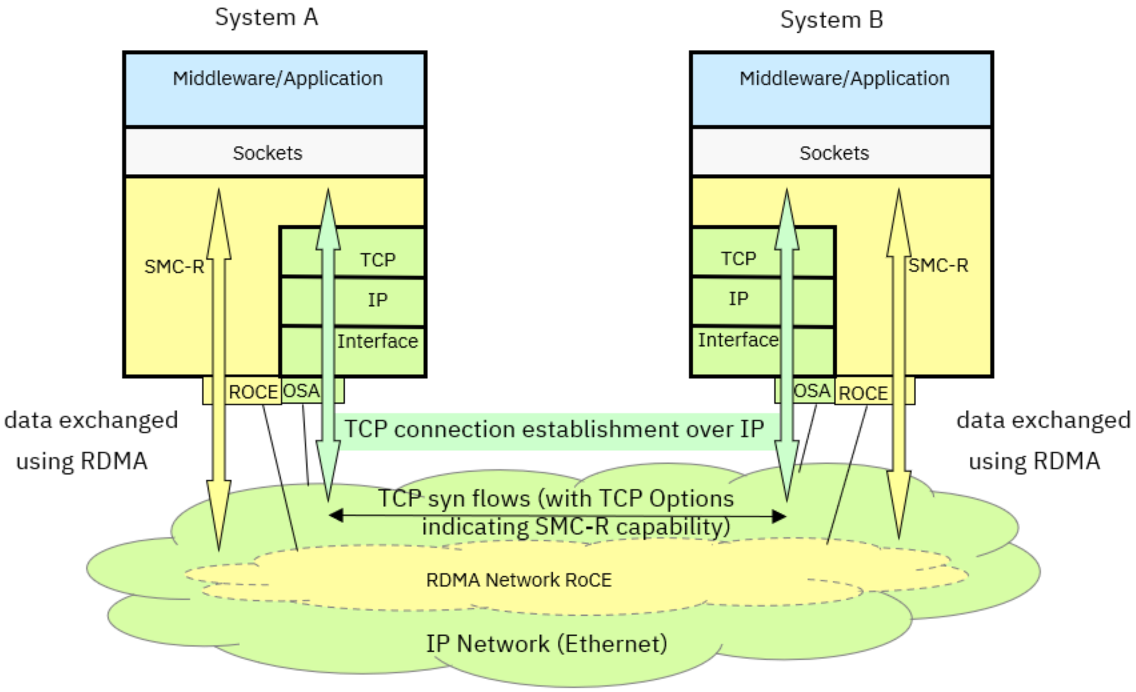

Clustered Systems: Example: Local and Remote access to DB2 from WAS (JDBC using DRDA)
SMC-D and SMC-R
SMC-R
z/OS image 1 (WAS)
z/OS image 2 (DB2)
z/OS image 3 (WAS)
shared memory
client
Sockets
SMC
shared memory
Server
Sockets
SMC
shared memory
Sockets
client
SMC
ISM
VCHID
ISM
RoCE
RoCE
RDMA enabled (RoCE)
Shared Memory Communications via DMA (SMC-D using vPCI ISM)
Shared Memory Communications via RDMA (SMC-R using RoCE)

Figure 3. Using Both Variations of SMC (SMC-R and SMC-D)

Both SMC variations can be used concurrently based on the hosts' configuration (same CPC versus different CPCs).



Dynamic (in-line) negotiation for SMC-R is initiated by presence of TCP Options

TCP connection transitions to SMC-R allowing application data to be exchanged using RDMA

Figure 4. Transition from TCP/IP to SMC-R

Eligible TCP/IP connections dynamically transition from TCP/IP to SMC (SMC-D or SMC-R). In this example, the transition is to SMC-R. Once the connection transitions, then user (socket) data is exchanged using SMC. The TCP connection remains active and idle. When the TCP connection terminates, the SMC connection also terminates.

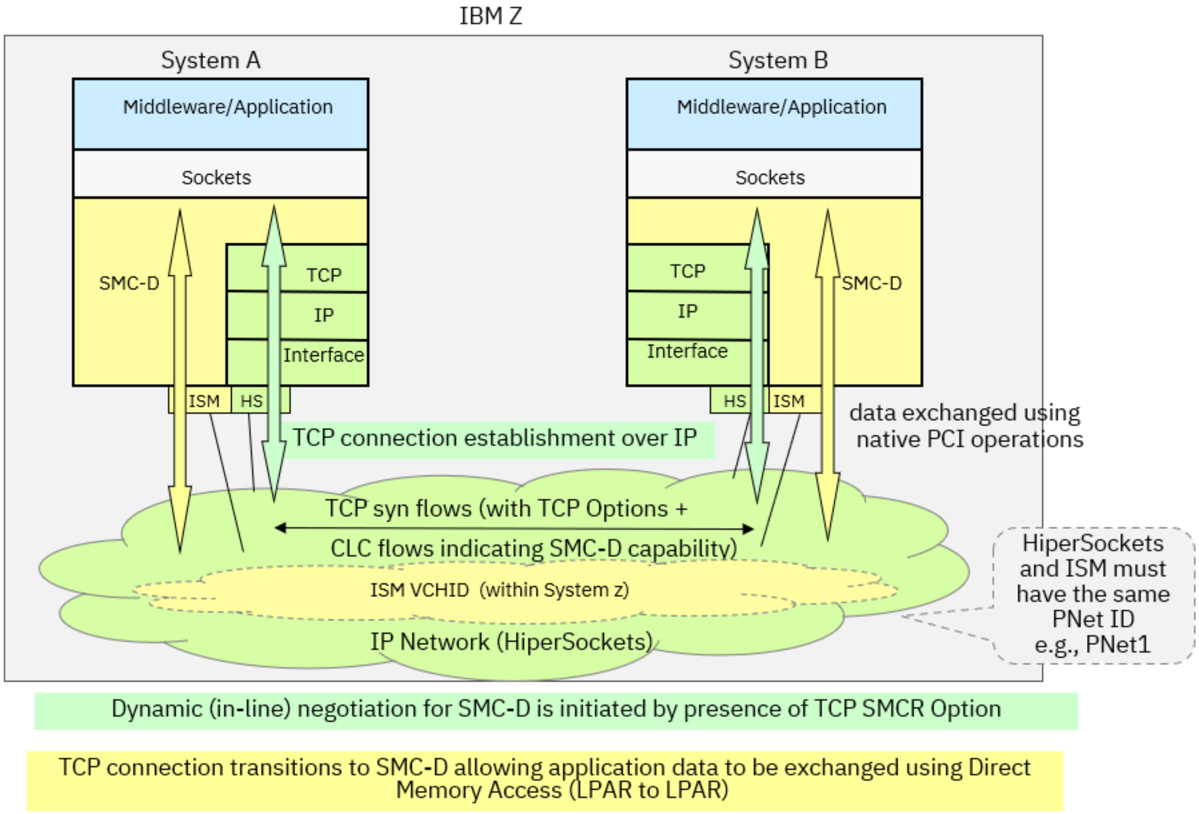

IBM Z
System A
Middleware/Application
Sockets
SMC-D
TCP
IP
Interface
ISM
HS
System B
Middleware/Application
Sockets
TCP
IP
Interface
SMC-D
HS
ISM
TCP connection establishment over IP
data exchanged using native PCI operations
TCP syn flows (with TCP Options + CLC flows indicating SMC-D capability)
ISM VCHID (within System z)
IP Network (HiperSockets)
HiperSockets and ISM must have the same PNet ID e.g., PNet1
Dynamic (in-line) negotiation for SMC-D is initiated by presence of TCP SMCR Option
TCP connection transitions to SMC-D allowing application data to be exchanged using Direct Memory Access (LPAR to LPAR)

Figure 5. Transition from TCP/IP to SMC-D

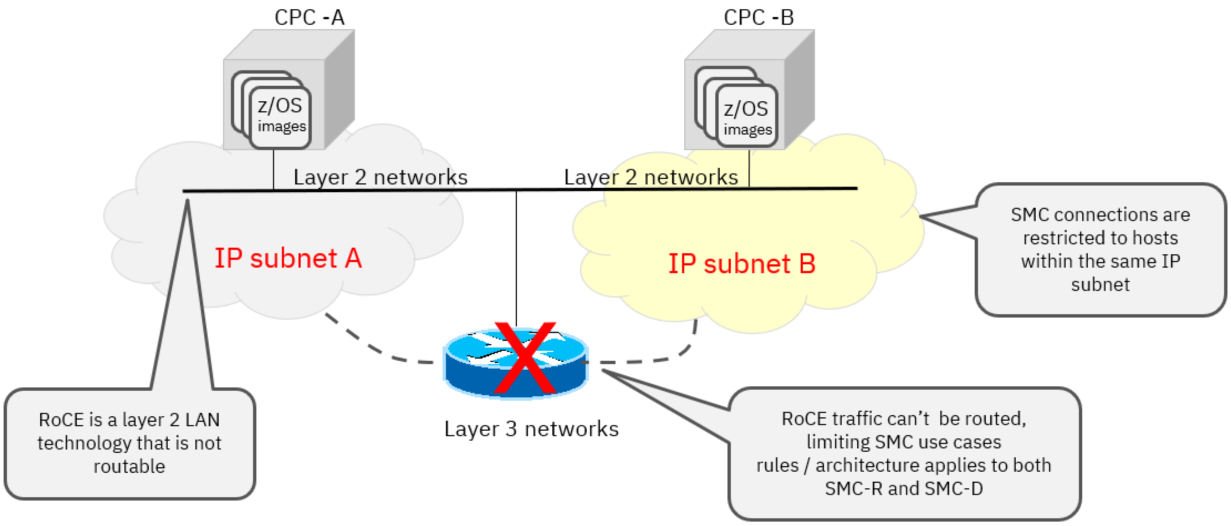
Similar to the SMC-R transition, eligible TCP/IP connections can also dynamically transition to SMC-D. The variation of SMC (SMC-R or SMC-D) is negotiated in the SMC 3-way CLC (Connection Layer Control) handshake. Peer hosts must be executing on the same IBM Z CPC and have access to the same ISM CHID[2]. If VLANs are defined on the IP network (in this example HiperSockets), the peers must also have access to the same VLAN. All SMC connections must be associated with a TCP/IP connection that is over a single IP subnet.

Once the connection transitions, then user (socket) data is exchanged using SMC. The TCP connection remains active and idle. When the TCP connection terminates, the SMC connection also terminates.

[2] For a definition of the ISM CHID or VCHID, refer to the glossary.

SMC CONNECTION ELIGIBILITY (SINGLE IP SUBNET)

SMCv1 is limited to layer 2 communications within a single IP subnet. This limitation was based on the original RoCE layer 2 specifications (which was not routable). Both SMC-R and SMC-D followed the same SMC base architecture and the same single subnet connection eligibility rule.



SMC connections are restricted to hosts within the same IP subnet

RoCE is a layer 2 LAN technology that is not routable

RoCE traffic can't be routed, limiting SMC use cases rules / architecture applies to both SMC-R and SMC-D

Figure 6. SMC Single IP Subnet Limitation

TCP connections are eligible for SMC when:

1. Both client and server have access to the same physical layer 2 network (LAN).
2. Both client and server have access to the same IP subnet (and VLAN when applicable).
3. The TCP connection does not require IPSec encryption.

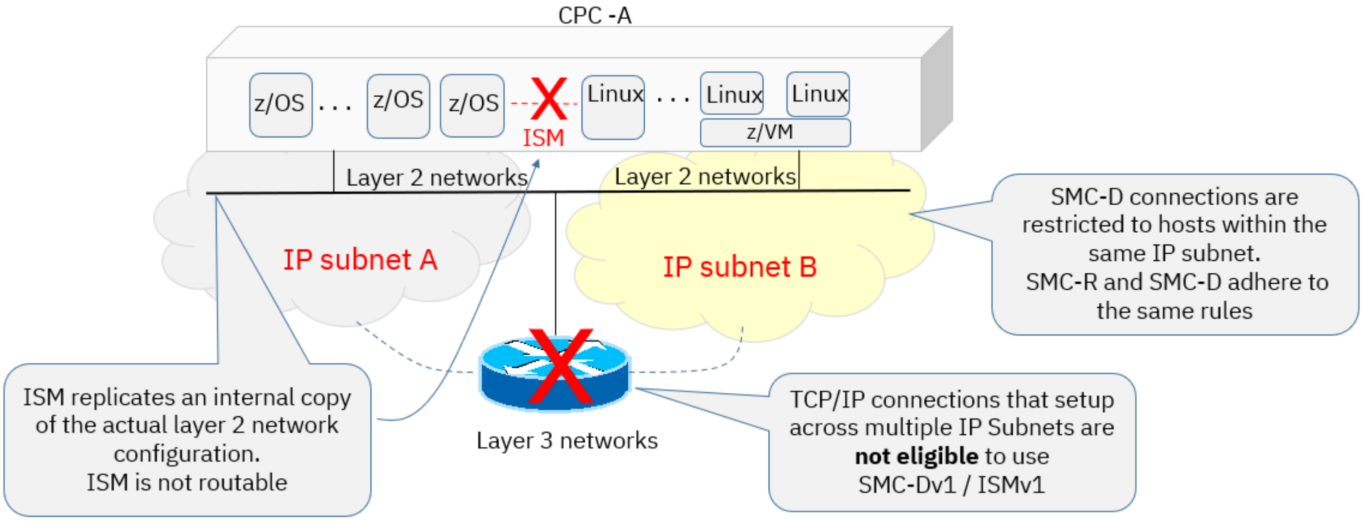

CPC -A
z/OS
. . .
z/OS
z/OS
ISM
Linux
. . .
Linux
Linux
z/VM
Layer 2 networks
Layer 2 networks
IP subnet A
IP subnet B
Layer 3 networks
SMC-D connections are restricted to hosts within the same IP subnet.
SMC-R and SMC-D adhere to the same rules
ISM replicates an internal copy of the actual layer 2 network configuration.
ISM is not routable
TCP/IP connections that setup across multiple IP Subnets are **not eligible** to use SMC-Dv1 / ISMv1

Figure 7. SMC within a single CPC - Single IP Subnet Rule Applies

In this example, the hosts are executing on the same physical IBM Z CPC, but the z/OS systems and the Linux® systems have access to unique IP subnets. The z/OS to Linux TCP/IP connections require Layer 3 IP routing (multiple IP subnets) and are not eligible for SMC (SMC-R or SMC-D).

SHARED MEMORY COMMUNICATIONS VERSION 2

SMC Version 2 (SMCv2) is introduced to provide Shared Memory Communications over multiple IP subnets.

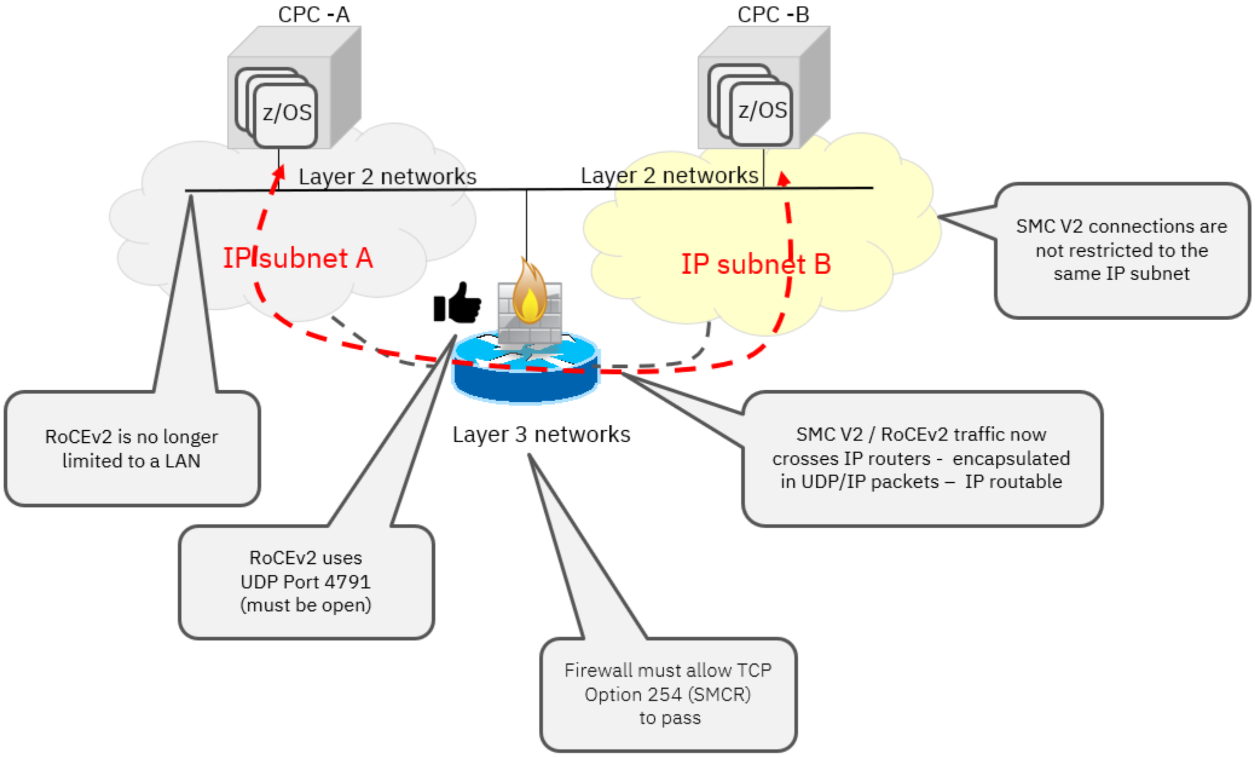

CPC -A
CPC -B
z/OS
z/OS
Layer 2 networks
Layer 2 networks
IP subnet A
IP subnet B
SMC V2 connections are not restricted to the same IP subnet
RoCEv2 is no longer limited to a LAN
Layer 3 networks
SMC V2 / RoCEv2 traffic now crosses IP routers - encapsulated in UDP/IP packets – IP routable
RoCEv2 uses UDP Port 4791 (must be open)
Firewall must allow TCP Option 254 (SMCR) to pass

Figure 8. SMC Version 2 - Multiple IP Subnet Support

SMC Version 2 (SMCv2) defines the specifications that enable multiple IP subnet capabilities for SMC. The multiple IP subnet capability is enabled by updates to the underlying networking specifications for RoCE (referred to as RoCEv2) and the IBM Z Internal Shared Memory (ISM) feature (referred to as ISMv2) along with updates to the related technologies. SMCv2 was announced in the IBM z/OS V2.4 3Q 2020 new functions and enhancement RFA.

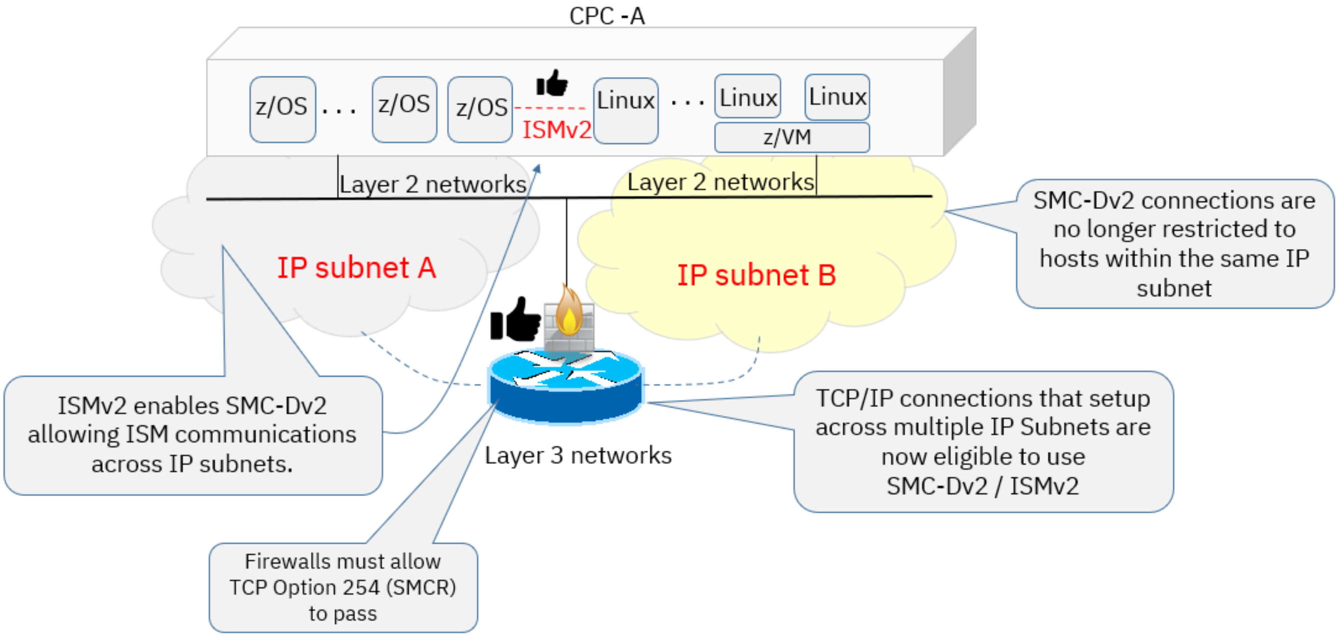

CPC -A
z/OS
. . .
z/OS
z/OS
ISMv2
Linux
. . .
Linux
Linux
z/VM
Layer 2 networks
Layer 2 networks
IP subnet A
IP subnet B
SMC-Dv2 connections are no longer restricted to hosts within the same IP subnet
ISMv2 enables SMC-Dv2 allowing ISM communications across IP subnets.
Layer 3 networks
TCP/IP connections that setup across multiple IP Subnets are now eligible to use SMC-Dv2 / ISMv2
Firewalls must allow TCP Option 254 (SMCR) to pass

Figure 9. SMC-Dv2 with ISMv2 Multiple IP Subnet Connectivity

The SMCv2 protocol also enables SMC-Dv2 with ISMv2 connections when operating systems are on the same IBM Z system but are not directly attached to the same IP subnet. In this example, the z/OS systems are attached to subnet A and the Linux systems are attached to subnet B. TCP/IP connections can be established over the external multiple hop IP network. If both systems are updated with SMC-Dv2, the TCP connections become eligible to connect using SMC-Dv2 with ISMv2.

The purpose of this paper is to introduce and describe the basic principles of SMCv2 describing how SMC is updated to provide multiple IP subnet support for both SMC-D and SMC-R. SMC-D is a variation of SMC-R that is an open protocol defined in RFC 7609.

This paper is in support of the IBM announced SMCv2 related solutions that became available on z/OS in the following order:

1. SMC-Dv2 with ISMv2 on the IBM z15 System in September of 2020

2. SMC-Rv2 with RoCEv2 on the IBM z15 System with the IBM RoCE Express2 feature in September of 2021

The information for IBM Linux support for SMCv2 can be found at the following URL:

https://linux-on-z.blogspot.com/p/smc-for-linux-on-ibm-z.html

SMCV2 ENTERPRISE ID

Given that SMCv1 is a layer 2 solution the “scope” of eligible peer hosts was self-defined. To be eligible to connect with SMC, peers must have direct access to the same IP subnet. SMC was designed to be an Enterprise Data Center solution, either within an IBM Z or within an Ethernet (RoCE) layer 2 network (LAN). VLANs could also be used to further isolate LAN traffic.

SMCv2 is also designed to be an Enterprise Data Center solution and is not intended for the WAN or “extended distances”. However, since SMCv2 connections are no longer bounded by a single subnet a new boundary or scope must be defined. The V2 protocol and solution must provide some concept of defining or declaring eligibility along with “locality”.

SMCv2 provides the ability for the administrator to define a “group of systems” that are allowed to use SMCv2 across unique IP subnets within (the concept of) a location. The “group” is intended to represent the group of systems within the same general location, which could be a data center, a campus, or centers spanning sites within close geographical locations[3]. The groups could be large or small. SMC groups can be logically divided by business lines, system types (production versus test or development systems) or other business needs. The intent is to define systems that communicate frequently within an enterprise data center (close proximity data centers).

To solve this need, SMCv2 defines the Enterprise ID (SMCv2 EID). The EID is a user-defined name (UEID) that represents a group of systems allowed to communicate using SMCv2. The administrator defines the EID once and then shares or configures the same EID in every system allowed to participate in SMCv2 within the group. For example, an EID might be defined as NORTHCAMPUS, EASTPLEX or WESTPLEX, etc. Each system enabled for SMCv2 defined with the same EID would be permitted to communicate using SMCv2. With SMCv2, the IP subnets are not relevant and are not evaluated.

Given that operating systems can be moved from site to site, the operating system should support the ability to initialize or configure the EID based on the current location. In z/OS this is accomplished using system symbolics (i.e., configure EID based on “locality”). This methodology should be followed by administrators (when provided by the operating system).

The next two figures provide examples of the considerations when configuring EID illustrating the basic concepts of EID for both close proximity sites and sites not within close proximity. The examples referenced here apply to RoCEv2 (versus a single CPC) but the basic concepts of EID (of SMC groups) apply to both forms of SMCv2.

[3] The SMCv2 protocol does not provide a specific definition of “close proximity or close geographical locations”. The objectives for defining “SMC groups” using EIDs within an Enterprise data center within close geographical locations” is described topic “*SMC-Rv2 and RoCEv2 at Distance*” later in this paper.

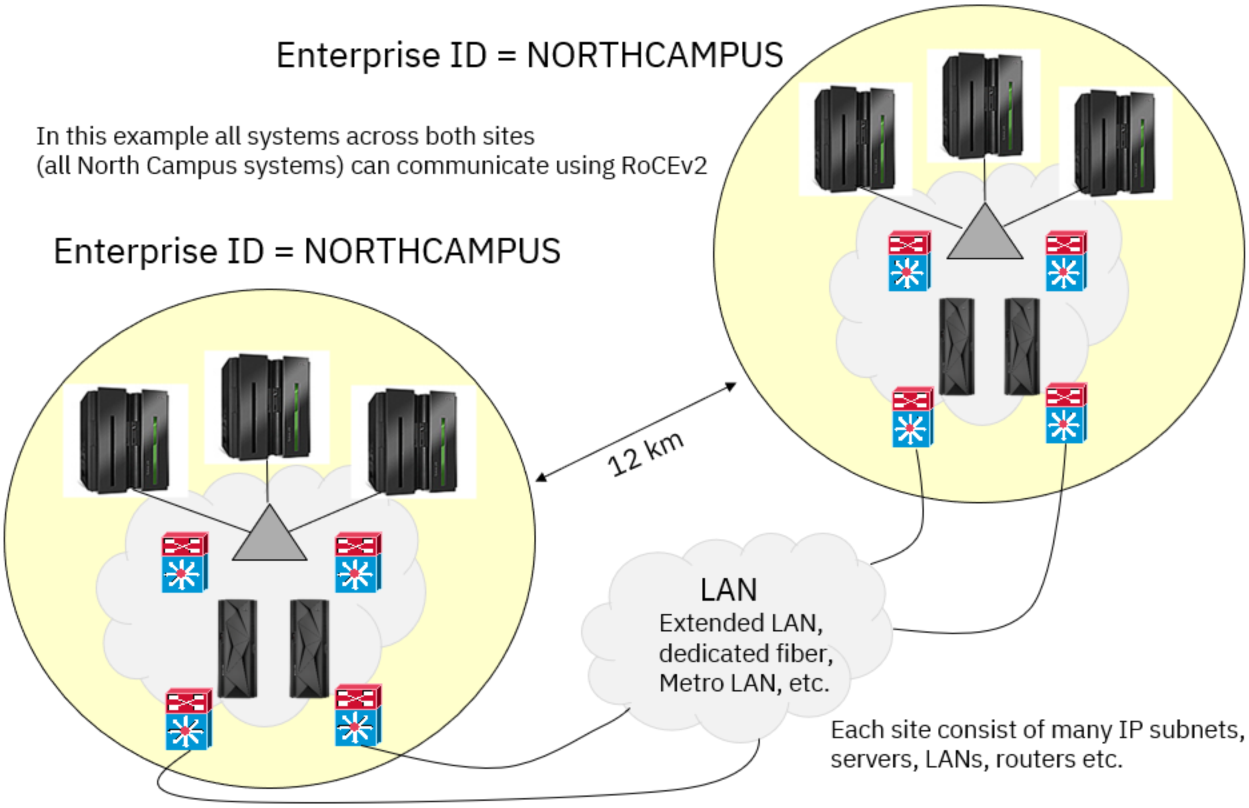

Enterprise ID = NORTHCAMPUS
In this example all systems across both sites
(all North Campus systems) can communicate using RoCEv2
Enterprise ID = NORTHCAMPUS
12 km
LAN
Extended LAN,
dedicated fiber,
Metro LAN, etc.
Each site consist of many IP subnets,
servers, LANs, routers etc.

Figure 10. Close Proximity Sites - Single EID

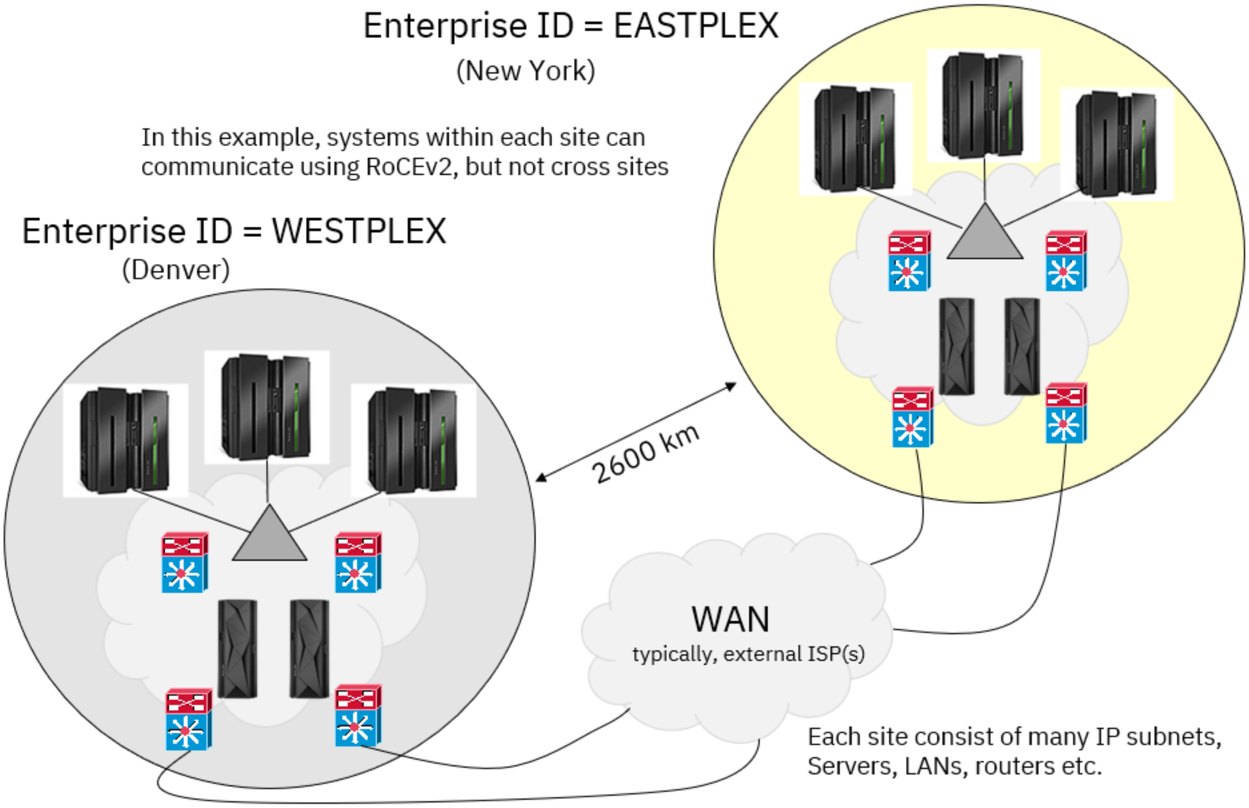

Enterprise ID = EASTPLEX
(New York)
In this example, systems within each site can communicate using RoCEv2, but not cross sites
Enterprise ID = WESTPLEX
(Denver)
2600 km
WAN
typically, external ISP(s)
Each site consist of many IP subnets, Servers, LANs, routers etc.

Figure 11. Distant Sites - Multiple EIDs

DYNAMIC EXCHANGE OF SMCV2 EIDS

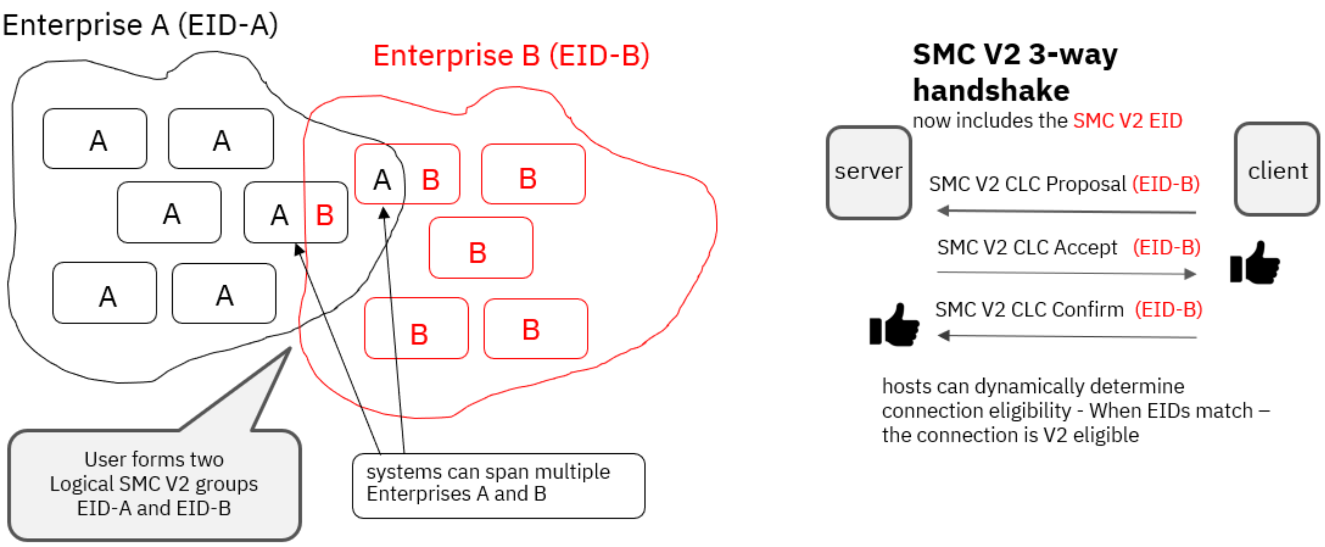

Enterprise A (EID-A)
Enterprise B (EID-B)
A
A
A
A B
A
A
A B
B
B
B
B
User forms two Logical SMC V2 groups EID-A and EID-B
systems can span multiple Enterprises A and B
SMC V2 3-way handshake
now includes the SMC V2 EID
server
client
SMC V2 CLC Proposal (EID-B)
SMC V2 CLC Accept (EID-B)
SMC V2 CLC Confirm (EID-B)
hosts can dynamically determine connection eligibility - When EIDs match – the connection is V2 eligible

Figure 12. Dynamic Exchange of EIDs

The SMCv2 protocol defines Version 2 updates to the SMC CLC messages and the related processing. The V2 SMC CLC 3-way handshake includes the EID. When the peers have a matching EID defined the peers are considered to be within the same SMCv2 group permitting SMCv2 communications. If the peers don't exchange a common EID (don't have a common EID defined), the connection can either fall back to SMCv1 (if same subnet) or to TCP/IP.

As this example illustrates, in some cases a host could reside in multiple SMC groups and require multiple EIDs to be defined. SMCv2 defines a maximum of 8 EIDs. z/OS will support up to four configured EIDs. The CLC handshake exchanges the defined EIDs. If at least one common EID is exchanged, the connection is eligible for SMCv2.

SMCV2 MIGRATION

The SMCv2 protocol allows for mixed-level systems. Operating systems that are not updated for SMCv2 will require an SMCv2 toleration software update[4].

If the client is up-level (supports V2) and enabled for V2, it could send the server an SMC Version 2 CLC Proposal Message (proposes V2). However, if the server is either down-level (supports V1 only) or is not enabled for V2, the connection will either fall back to SMCv1 (if same subnet) or use TCP/IP.

[4] For more information about the SMCv2 toleration software requirements, refer to the IBM SMC general information website: https://www.ibm.com/docs/en/zos/3.1.0?topic=communications-shared-memory-reference-information.

SMC-D VERSION 2 (SMC-DV2)

For many practical reasons, both SMC-R and SMC-D adhere to the same SMC base architecture and rules. This guideline is preserved with the updates for the SMCv2 protocol rules. In some cases, there will be slight variations or differences as needed.

Both protocols will exploit the user-defined EID. However, given that SMC-Dv2 is limited to communications within a single IBM Z CPC, SMC-Dv2 will also support the concept of a “System EID” (SEID). The SEID is a single auto-generated EID that is based on the IBM Z CPC itself. Depending on the level of granularity required within a CPC when defining SMC groups, users could elect to use the single SEID, separate operating systems using multiple UEIDs, or use a combination of both types of EIDs.

SMC-DV1 REVIEW

Recall that the original version of SMC was based on layer 2 communications (i.e., a LAN). This means that the scope of communications was defined by the physical network (LAN) and possibly a virtual LAN (when VLANs are defined). Since communications were limited to layer 2, the IP subnet was not applicable (i.e., the original RoCE did not use IP packets). Therefore, communications could not cross IP subnets (i.e., limited to a single subnet).

SMCv1 communications reused or inherited the layer 2 network configuration related to the host's IP interfaces. For SMC-D, this was the same physical and virtual network attributes associated with OSA or HiperSockets interfaces.

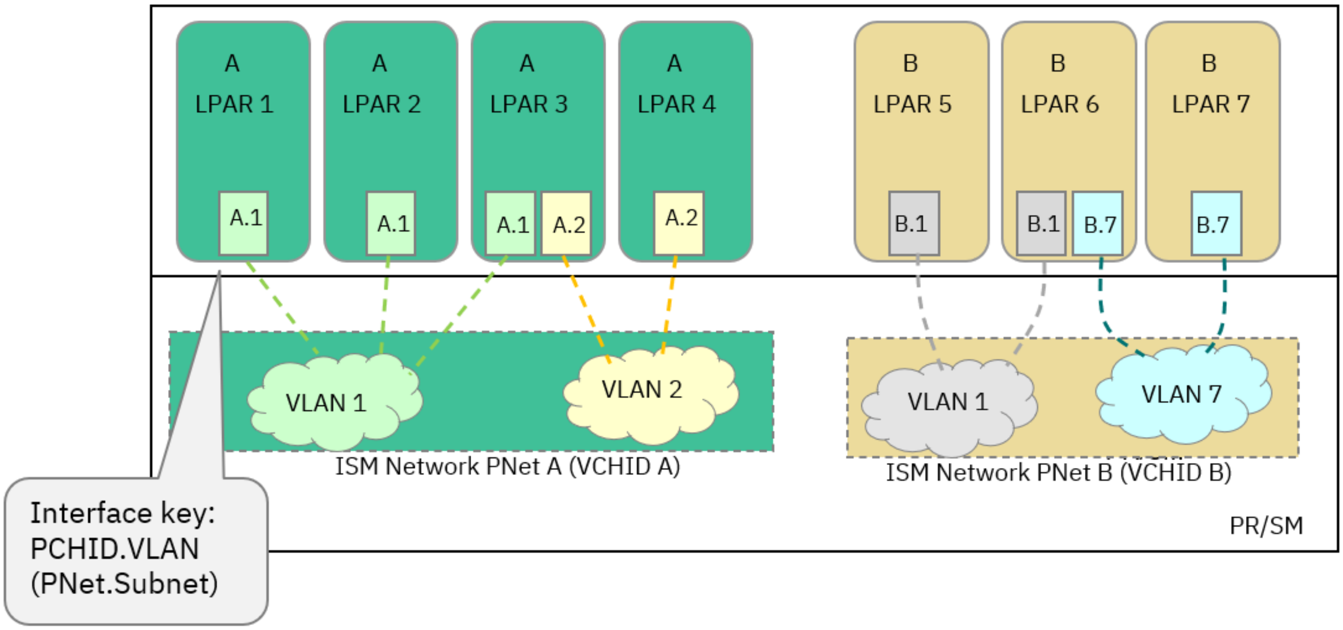


Figure 13. Review - IBM Z ISMv1 Configuration (Isolating Systems using SMC-Dv1)

This figure of the IBM Z Internal Shared Memory architecture illustrates two important SMC-Dv1 key concepts:

1. How ISM inherits the associated IP networks' layer 2 attributes (from OSA (Ethernet) or HiperSockets (Internal Virtual Network)) of the physical network (based on PNetIDs) and virtual network (based on the VLAN IDs). In the examples, you can also view each VLAN as a unique IP subnet.

2. How ISM communications can isolate a group of systems (i.e., group A and group B) using unique PNetIDs (ISM VCHIDs) or unique VLANs on the same network.

The key point of this figure is to establish an understanding of the base concepts of SMCv1 by noting how ISM adheres to the user's existing layer 2 topology attributes of the associated IP network configuration. In this case, the ISM interfaces are associated with an OSA or HiperSockets interface.

A TCP/IP connection is eligible to use SMC-Dv1 over this ISM network path when the network IP topology of the associated TCP/IP connection is within the same IP subnet (for example, LPARs 1 and 2). However, if the IP topology crossed IP subnets using an IP router (e.g., LPARs 1 and 4), this TCP/IP connection would not be eligible for SMC (using SMC-Dv1).

SMC-DV2 AND ISMV2 SOLUTION

To provide an ISM connection that is not restricted to the existing layer 2 topology and rules, a new form of "ISM Layer 3" internal connection was required. This type of ISM connection could not be associated with or bound to the existing Layer 2 physical or virtual networks (VLANs) shown in Figure 12.

With the IBM z15, IBM Z introduced updates to the ISM feature, referred to as ISMv2 that removes the existing PNetID and VLAN ID attributes and definitions. This solution is provided with the IBM z15[5].

When the IBM Z operating systems[6] that support SMC-D connecting to the ISM VCHID are updated to support the new ISMv2 connectivity feature, the multiple IP subnet solution of the SMC-Dv2 protocol can be exploited.

The SMC-Dv2 and ISMv2 multiple IP subnet eligibility requirements are summarized as follows, both the TCP client and server LPARs (guests) must:

1. Be updated (and enabled) to support SMC-Dv2
2. Execute on the same IBM Z CPC that supports ISMv2
3. Have access to a common ISM VCHID
4. Be defined with a common EID
5. **Not** be associated with a TCP/IP connection that requires IPSec encryption

Using SMC-Dv2 with ISMv2, the next figure illustrates the solution and the main concepts of how LPARs 1 and 4 can now communicate over multiple IP subnets.

[5] For IBM z15 T01, refer to the MCL number P46601.067 driver D41C. The ISMv2 support is in the base of the IBM z15 T02.

[6] For the z/OS V2R4 software requirements, refer to APARs/PTFs OA59152/UJ03768 and PH22695/71143.
For the Linux support, refer to the following URL: https://linux-on-z.blogspot.com/p/smc-for-linux-on-ibm-z.html

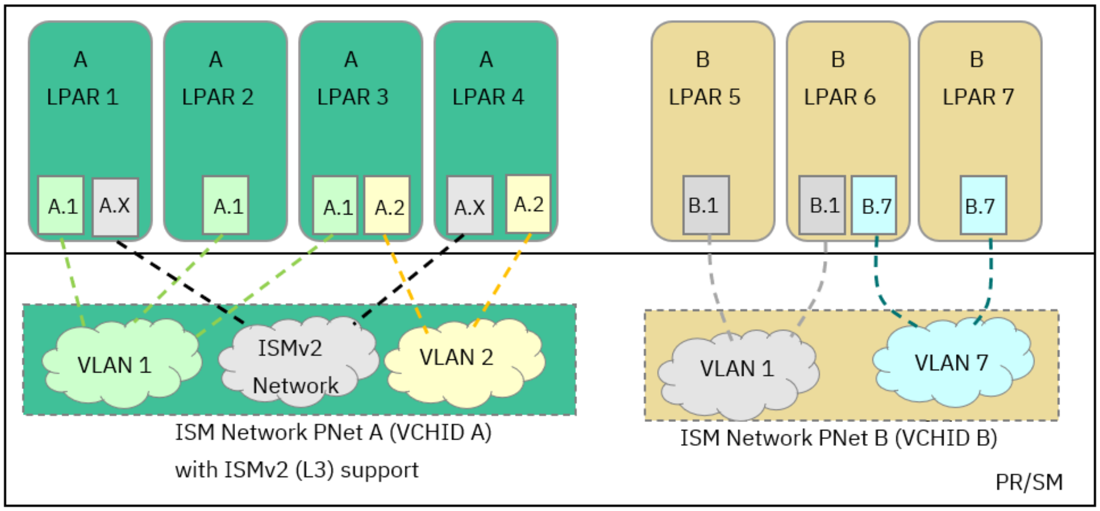


Figure 14. SMC-Dv2 with ISMv2 L3 Networks (ISMv2 example 1)

This figure illustrates some of the main concepts of SMC-Dv2 and ISMv2 as follows:

1. Both LPARs 1 and 4 have an internal Layer 3 connection to ISM VCHID A that is created for the SMC-Dv2 solution (shown here as A.X interfaces). LPARs 1 and 4 have software updates to support the SMC-Dv2 protocol and the new ISM L3 capability.

2. The ISM L3 connection to VCHID A does not use any L2 attributes (such as a VLAN ID).

3. As shown here in this example 1, VCHID A is associated with a PNetID A and VCHID B is associated with PNetID B. The ISM VCHIDs are considered to be “**associated ISM VCHIDs**” meaning the ISM VCHIDs:

    a. Are defined (in HCD) with a PNetID (the CHIDs are “associated with” a physical network) and this attribute allows the VCHIDs to

    b. Be associated with IP interfaces (such as OSA or HiperSockets) when they (OSA or HiperSockets) are defined with matching PNetIDs. This PNetID association is required for SMCv1 (single subnet layer 2 support).

4. However, ISMv2 does not require a PNetID definition on the ISM VCHID and therefore the L3 connection does not require an association with an IP interface (i.e., inheriting the L2 attributes of an OSA or HiperSockers IP interface is N/A to ISMv2).

5. This configuration permits L3 connectivity among up-level hosts (those that support SMC-Dv2 such as LPARs 1 and 4) and traditional ISM L2 connectivity to down-level hosts (support only layer 2 based SMC-Dv1, such as LPARs 2 and 3).

6. Using SMC-Dv2 when the TCP/IP connection spans multiple IP subnets means the SMC 3-way handshake occurs over the IP network. It is important that any firewalls in the TCP/IP connection path allows TCP option 254 pass.

MIGRATION CONSIDERATIONS (ALTERNATIVE ISMV2 CONFIGURATIONS)

Groups A and B might be required to maintain this isolated configuration for some time. Their existing isolated L2 topology with their isolated ISM connectivity using separate ISM VCHIDs A and B might be continued to be required. Examples for the reason for this separation could be due to:

1. Some systems are not scheduled for updating (SMC-Dv2) for a longer period (this example will make more sense when you see the next example configuration) or
2. The two groups must continue to preserve their existing L2 network separation. This L2 network separation requirement could be based on various reasons, such as when the two groups (A and B) are unique business units (must use unique LANs).

The next figure provides an example of an ISM VCHID without PNetID.

However, in this same configuration there could also be systems among groups A and B that now have a need to exploit ISMv2 to communicate across VCHIDs A and B. The user now wants to take advantage of the benefits of SMC-D and the increased (multiple IP subnet) reach of ISMv2. For example, in this configuration, how could LPARs 4 and 5 communicate using ISMv2?

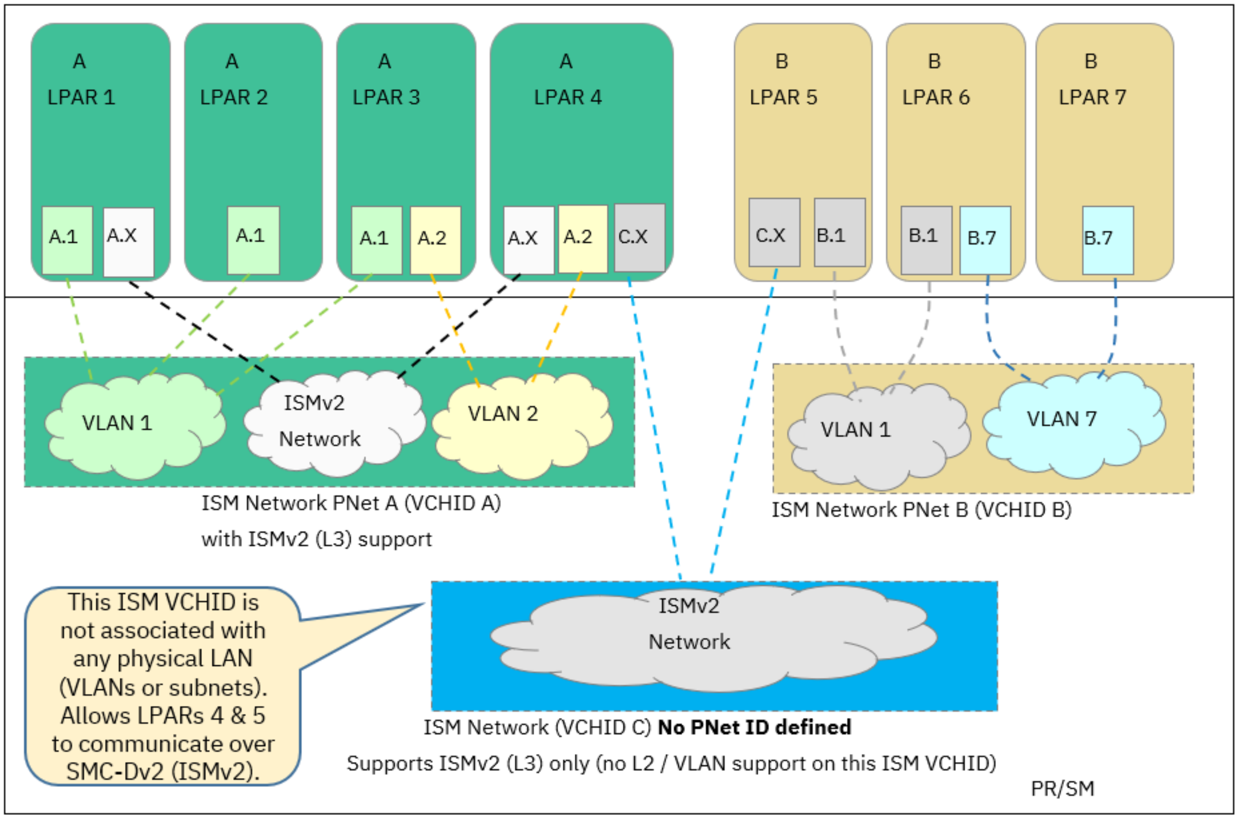

A
LPAR 1
A.1
A.X
A
LPAR 2
A.1
A
LPAR 3
A.1
A.2
A
LPAR 4
A.X
A.2
C.X
B
LPAR 5
C.X
B.1
B
LPAR 6
B.1
B.7
B
LPAR 7
B.7
VLAN 1
ISMv2
Network
VLAN 2
ISM Network PNet A (VCHID A)
with ISMv2 (L3) support
VLAN 1
VLAN 7
ISM Network PNet B (VCHID B)
This ISM VCHID is not associated with any physical LAN (VLANs or subnets). Allows LPARs 4 & 5 to communicate over SMC-Dv2 (ISMv2).
ISMv2
Network
ISM Network (VCHID C) **No PNet ID defined**
Supports ISMv2 (L3) only (no L2 / VLAN support on this ISM VCHID)
PR/SM

Figure 15. ISMv2 VCHID without PNetID (ISMv2 example 2)

In this example 2, VCHID C does not have a PNetID defined in HCD. VCHID C is considered to be an **unassociated ISM VCHID**. ISM VCHID C is not associated with any IP (OSA or HiperSockets) interfaces. Since ISM VCHID C does not inherit any L2 attributes, it can only be used for ISMv2 Layer 3 communications. SMCv2 allows up to 8 unassociated ISM VCHIDs. z/OS will support (activate and use) up to four unique unassociated ISM VCHIDs. LPARs 4 and 5 can now use SMC-Dv2 and ISMv2 over multiple IP subnets[7]. Refer to the example shown in Figure 6 TCP/IP connectivity with multiple IP subnets, which is now resolved by SMC-Dv2 and ISMv2.

[7] The ISMv2 L3 connection means that the client and server are using SMC-Dv2 and the IP subnets of the client and server are not evaluated (not relevant). Also, the IP route, number of IP hops or the overall IP topology of the associated TCP/IP connection is not relevant for SMC-Dv2 for the ISMv2 L3 connection.

ISMV2 ONLY CONFIGURATIONS

In an environment where down-level systems are not present where ISMv1 is not required, a user might elect to have a Z configuration where a minimal number of ISM VCHIDs are used. Here all operating systems are now up-level (all support SMC-Dv2). This could be a user rolling out SMC-D for the first time or an existing SMC-D user who has updated all operating system software. This user could simplify their ISM configuration by exploiting ISMv2 only with an unassociated ISM VCHID (no PNetID). In this environment, the ISM topology is greatly simplified.

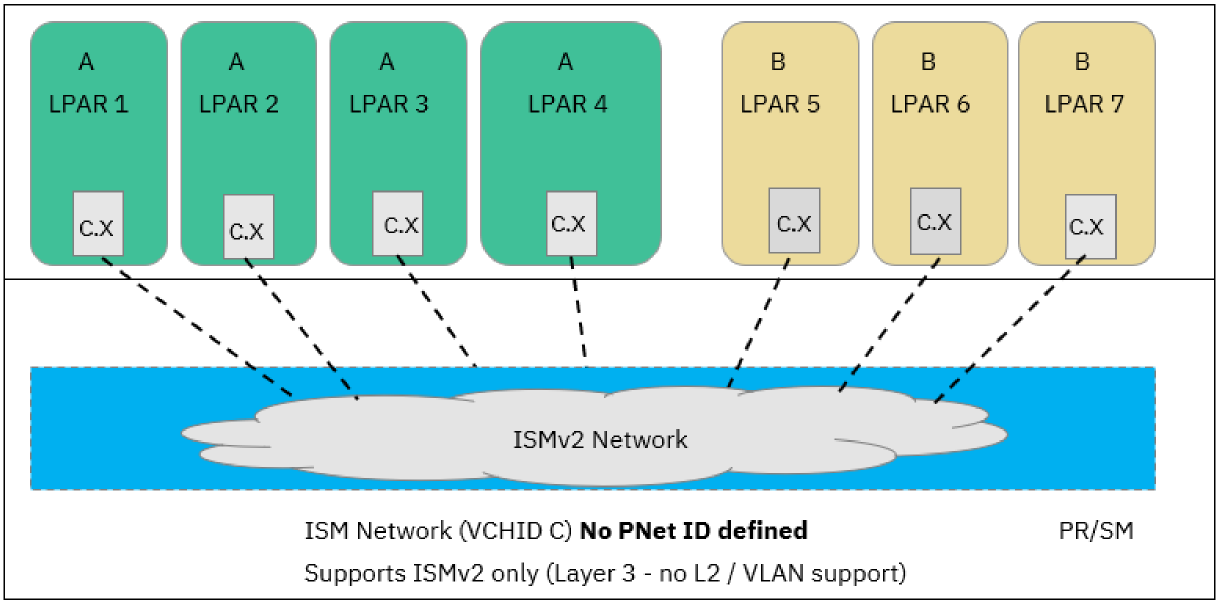

A
LPAR 1
C.X
A
LPAR 2
C.X
A
LPAR 3
C.X
A
LPAR 4
C.X
B
LPAR 5
C.X
B
LPAR 6
C.X
B
LPAR 7
C.X
ISMv2 Network
ISM Network (VCHID C) **No PNet ID defined**
Supports ISMv2 only (Layer 3 - no L2 / VLAN support)
PR/SM

Figure 16. ISMv2 Only Configuration

Note that in this configuration each host now has a single ISM interface that is not associated with any IP interfaces (no layer 2 attributes). The IP topology (single versus multiple IP subnets) of the associated TCP/IP connections becomes irrelevant for SMC-Dv2.

ISMV2.1 EMULATED-ISM DEVICE FEATURE

SMC-Dv2.1 introduces a new ISM feature where the ISM device support is provided completely within software, and contained within a single:

1. Operating System instance for socket application "loop-back" or
2. hypervisor instance for intra-hypervisor guest to guest connectivity.

The SMC-Dv2.1 ISM feature is referred to as the "*Emulated-ISM Device Feature"* that introduces the following SMC-Dv2.1 concepts, terminology, and protocol specifications.

EMULATED-ISM DEVICES, EMULATED-ISM CHIDS AND EXTENDED GIDS

1. SMC-Dv2.1 introduces a new type of ISM device called an Emulated-ISM device used for:
   a. loopback-ism and
   b. virtio-ism
2. Emulated-ISM devices are created and primarily supported by system (OS or hypervisor) software vs. platform firmware or hardware. There is also a potential for Emulated-ISM devices to be supported by a combination of software and platform compute resources (such as special processors, ASICs, firmware etc.). SMC-Dv2.1 Emulated-ISM device support is defined as ISMv2.1 support. SMC-Dv2.1 specifications do not define the OS or hypervisor Emulated-ISM device implementation. For example, another hypervisor might elect to virtualize ISM without using virtio.
3. Emulated-ISM device support is a supplemental feature of SMC-Dv2.1. When this feature is supported, it is indicated in the SMC-v2.1 Supplemental Feature Bit Mask passed in the CLC exchange. An OS (client or server) can support SMC-Dv2.1 and optionally support the Emulated-ISM Device feature.
4. ISMv2.1 devices support V2 only and are also "unassociated ISM devices" meaning the devices are not associated with a specific IP device and therefore do not require a PNET ID definition.
5. The scope or reach of the hypervisor virtio-ism device is not defined by the SMC-Dv2.1 specifications. For example, the virtio-ism support can be extended to include guest virtual machines, containers and communications in any supported combination.
6. Two types of ISM devices are now supported by SMC-Dv2.1.
   With the introduction of the Emulated-ISM feature there are now two types of ISM devices. To avoid ambiguity attention to specific ISM terminology becomes important:
   a. ISM device, references to an ISM device indicates the existing platform firmware ISM device support, or the specific reference is general to ISM indicating a distinction in a specific ISM device type is not necessary.
   b. Emulated-ISM device, references to an ISM device where the reference is unique to Emulated-ISM devices must use the Emulated-ISM terminology.
7. Emulated-ISM devices require a larger GID that is a 128-bit GID vs. the existing ISM 64-bit GID. The 128-bit GID is referred to as an ***Extended GID*** and applies only to Emulated-ISM devices.
8. Extended GIDs are created and managed by software, the OS or hypervisor, using a UUID as defined by the IETF in RFC 4122. Version 4 UUIDs are recommended.
9. *Emulated-ISM CHIDs* are introduced to support Emulated-ISM devices. Extended ISM GIDs are associated with Emulated-ISM CHIDs. Like ISM CHIDs, Emulated-ISM CHIDs represent an internal fabric. Emulated-ISM CHIDs are defined with a reserved CHID value or range of values defined as:
   a. Loopback-ism (xFFFF) and

   b. Virtio-ism (CHID range of xFF00 – xFFFE).

10. Reserved Emulated-ISM CHID values will not be used by (will not conflict with) platform ISM firmware.
11. Extended GIDs along with their associated Emulated-ISM CHIDs are passed in the SMC-Dv2.1 CLC 3-way handshake.
12. The SMC-Dv2.1 client sends Extended GID(s) in the SMC-Dv2.1 CLC Proposal message in the existing SMC-Dv2 Extension GID-CHID array. Extended GIDs consume two entries within the GID-CHID array with the following format:
    a. ISM Extended GID 8-byte part 1 (1st 8 bytes)
    b. ISM CHID (Emulated-ISM CHID, reserved CHID value)
    c. ISM Extended GID 8-byte part 2 (2nd 8 bytes)
    d. ISM CHID (same Emulated-ISM CHID repeated, reserved CHID value)
13. Usage notes:
    a. If the server does not support SMC-Dv2.1 Emulated-ISM devices, then reserved CHIDs passed by the client will be ignored by the server (i.e. will not match the server CHIDs).
    b. ISM devices are defined by platform native (virtual) PCI device architecture represented by a PFID and VF. Emulated-ISM devices are supported by software, and the software implementation is responsible for device specifications (such as PFID and VF applicability, definition, configuration, values, or replacement).
    c. If the client uses a reserved CHID value but fails to repeat the Extended GID values in two consecutive GID-CHID array entries, then this is considered a protocol violation. The error must be rejected by an SMC-Dv2.1 server and will be ignored by a down-level non SMC-Dv2.1 server.
    d. SMC version 2.1 introduced an initial set of features that were specific to SMC-Rv2.1 only. The Emulated-ISM device feature was created after the initial set of v2.1 features. All features introduced after the initial v2.1 features are considered a "v2.1 supplemental feature". All supplemental features must be defined within the SMCv2.1 feature bit mask. SMCv2.1 hosts that support Emulated-ISM devices must set the corresponding v2.1 feature bit. Refer to the v2.1 feature bit array for the Emulated-ISM device feature bit.
    e. SMCv2 requires an EID, either an SEID or UEID. Emulated-ISM devices impose no changes on the EID specifications. However, since the scope of the Emulated-ISM device support is the OS instance or the hypervisor, the original concept of "System" within SEID takes on a unique meaning. SEID is not excluded by the Emulated-ISM specifications.

See table 1 below for a side-by-side comparison of ISM and Emulated-ISM related resources.

SMC-Dv2 / ISMv2 (S/390®)		SMC-Dv2.1 / ISMv2.1 Emulated-ISM feature (any platform)	
ISM artifacts, resources, and elements provided by the platform	Description	Emulated-ISM artifacts, resources, and elements provided by software	Description
(V) CHID	Internal Fabric	Emulated-ISM CHID	Internal Fabric (reserved values)
(V) ISM Device	Device (access to fabric)	Emulated-ISM Device	Device (access to fabric)
ISM GID	Device address (on fabric)	Extended GID	Device address (on fabric)
SEID/UEID	SMCv2 EID (SMC group)	SEID/UEID	SMCv2 EID (SMC group)
DMB Token	Remote address to DMB	DMB Token	Remote address to DMB
ISM Write operation	PCI-STB (firmware operation) Instruction(s) to move data	ISM Write operation	Implementation (software operation) Instruction(s) to move data

Table 1. Comparison of ISM and Emulated-ISM

Table 1 provides a side-by-side comparison of ISM and Emulated-ISM related resources. In some cases, information about the platform ISM architecture can informally use the term “virtual”, such as virtual ISM or virtual CHID shown above as (V) CHID. The list of ISM resources or functions shown in this table describe some of the key ISM resources and is not intended to be a complete list.

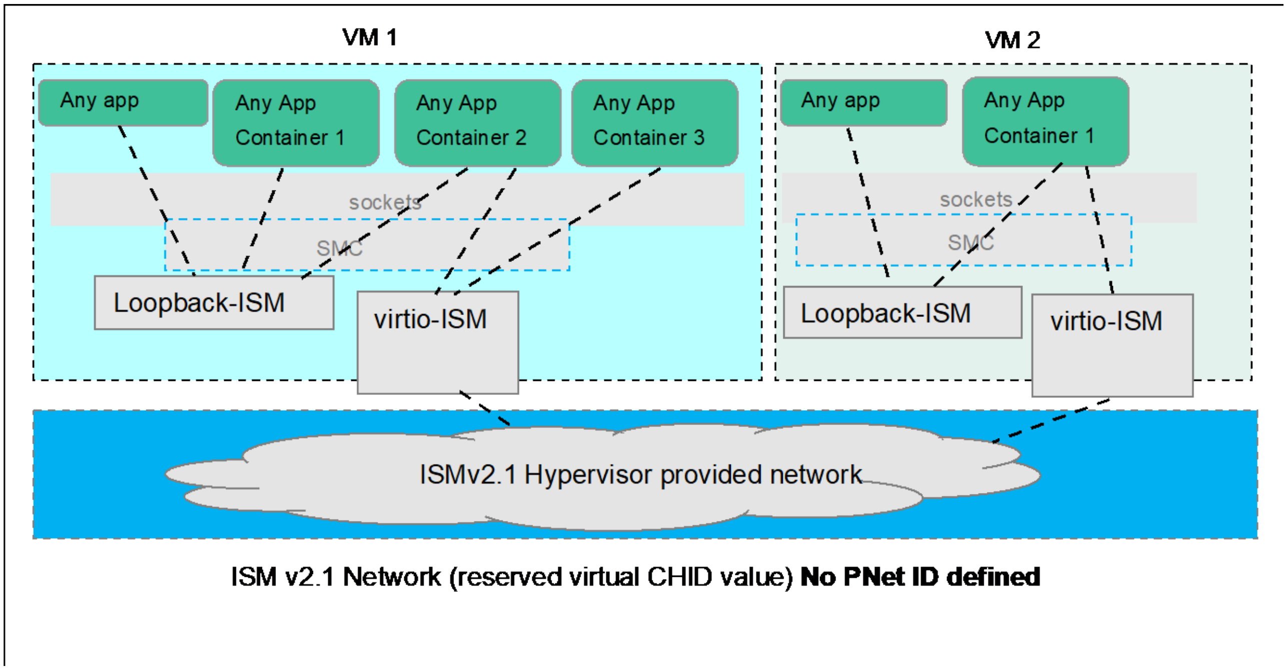


Figure 17. ISM v2.1 Emulated-ISM Device Architecture Overview – Software Provided Networks

Figure 17 illustrates examples of Emulated-ISM devices defined by SMCv2.1 for both loopback and virtio. Emulated-ISM devices are software provided and managed devices. There are two types of Emulated-ISM devices:

1. The loopback-ISM device is provided by the OS and provides loopback connectivity within a VM (or OS instance).
2. The virtio-ISM device provides VM to VM connectivity provided within a hypervisor instance.

Refer to the previous *ISMv2.1 Emulated-ISM Device Feature* topic for additional details.

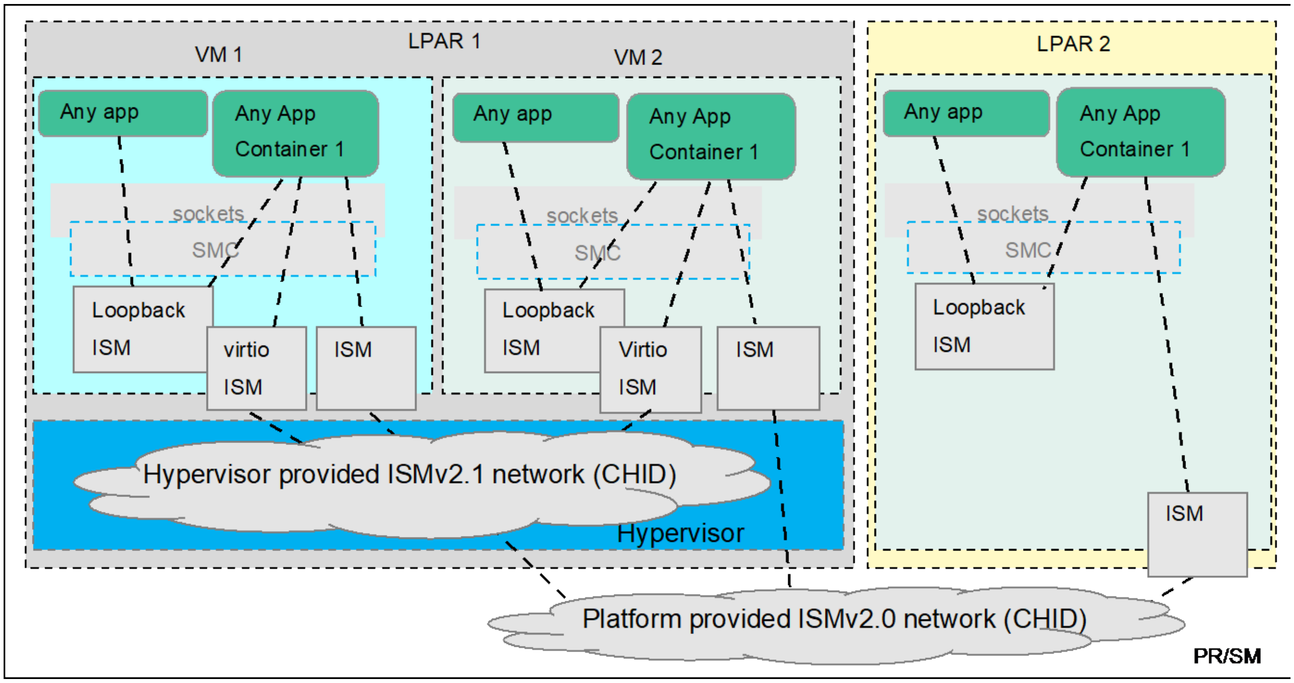


Figure 18. ISM v2.1 Emulated-ISM Device Architecture with ISM Device Architecture Overview

Figure 18 illustrates the ISMv2.1 Emulated-ISM Device architecture (software provided) combined with ISMv2 existing ISM Device architecture (platform provided). In this example LPAR 1 runs with a hypervisor that provides the Emulated-ISM device support for virtio-ISM that provides VM to VM connectivity. The Operating Systems running in VM 1 and VM2 provides the Loopback ISM support.

LPAR 2 does not run with a hypervisor and instead runs native LPAR (bare metal). In this example the platform illustrates the IBM S390 architecture which provides the LPAR support where the platform provided hypervisor is PR/SM. Applications in LPAR 1 can communicate with applications in LPAR 2 using ISMv2 device support.

Refer to the previous *ISMv2.1 Emulated-ISM Device Feature* topic for additional details. Refer to the ISMv2 topic for ISM device support.

SMC-R VERSION 2 (SMC-RV2)

To exploit RoCE over multiple IP subnets, an update is required for the RoCE standards and for the SMC-R protocol to exploit the new standards. This section describes those updates and introduces the main concepts of the next version of SMC-R referred to as SMC-R Version 2 (SMC-Rv2).

The SMCv2 topic earlier in this paper described the common SMCv2 aspects such as defining the “SMCv2 scope”. Since the IP subnet no longer defines SMC connectivity eligibility (scope), a new rule is required to define TCP connections that are eligible to connect with SMCv2. The Enterprise ID is introduced to define a group of SMC hosts or system eligible to communicate using SMCv2.

Both SMC protocols (SMC-Dv2 and SMC-Rv2) exploit the user-defined EID. However, only SMC-Dv2 uses the System EID (SEID). The SEID only applies to SMC-Dv2 within a single CPC.

SMC-RV1 REVIEW

Recall that the initial version of SMC was based on the original RoCE standards that were limited to layer 2 communications (i.e., a LAN). For RoCE, this means the scope of communications for SMC-R was defined or limited by the physical Ethernet network (LAN) and possibly a virtual LAN (when VLANs are defined). Since communications were limited to layer 2, the IP subnet could not be crossed and was not applicable. SMC-R communications were not routable and could not cross IP subnets (i.e., communications were limited to a single IP subnet on the LAN). The original version of SMC-R is referred to as SMC-R Version 1 (SMC-Rv1).

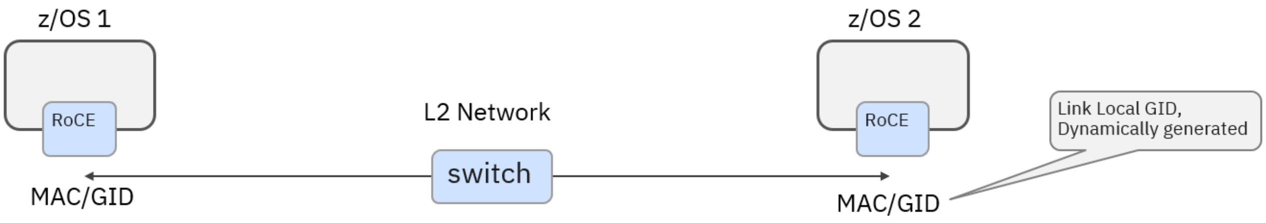


Figure 19. RoCE Layer 2 Communications

SMC-Rv1 interfaces were defined with a link local GID (not routable) and communications occurred using Layer 2 Ethernet frames instead of IP packets. The SMC-R RoCE Layer 2 connection formed over the LAN was defined as an SMC-R link. Each RoCE endpoint created a QP to represent the layer 2 link. RoCE adapters are provisioned in pairs to form SMC-R Link Groups to provide High Availability (HA) and load balancing. The format of the RoCE frames is shown in the next figure.

SMC-RV2 AND ROCEV2 SOLUTION

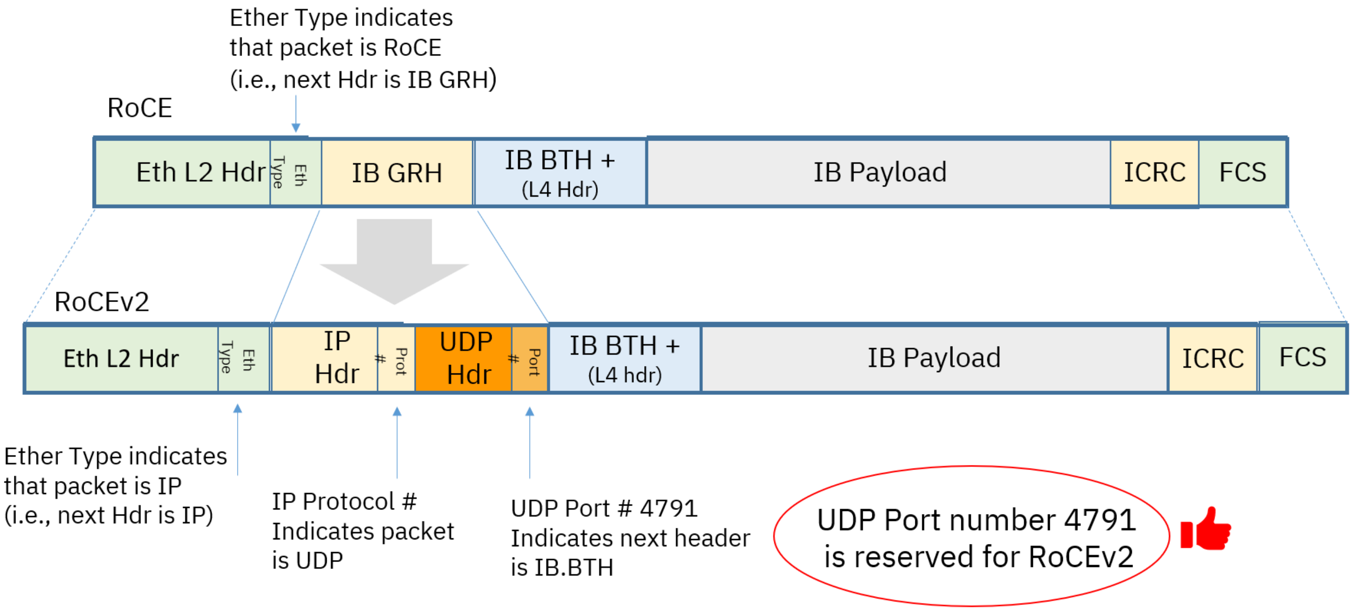


Figure 20. RoCE versus RoCEv2 ("Routable RoCE")

Figure 20 shows the differences in the RoCE frame format between the original version of RoCE and RoCEv2. RoCEv2 introduces "Routable RoCE" by encapsulating RoCE frames in UDP/IP. The RoCEv2 standards replace the layer 2 IB GRH with a layer 3 IP header making the RoCEv2 packets IP routable. UDP port 4791 is defined for RoCEv2. The RoCEv2 standards are defined in Annex A17 to the Supplement InfiniBand Architecture Specification Volume 1 release 1.2.1.

SMC-Rv2 is created to exploit the new RoCEv2 standards. For SMC-Rv2 the IB payload is TCP socket user data or SMC-Rv2 Link Layer Control (LLC) messages. SMC-Rv2 with RoCEv2 removes the single IP subnet limitation imposed by the original version of RoCE and SMC-Rv1. Removing this limitation extends the benefits of RDMA technology to additional (in some cases many more) application workloads in your Enterprise.

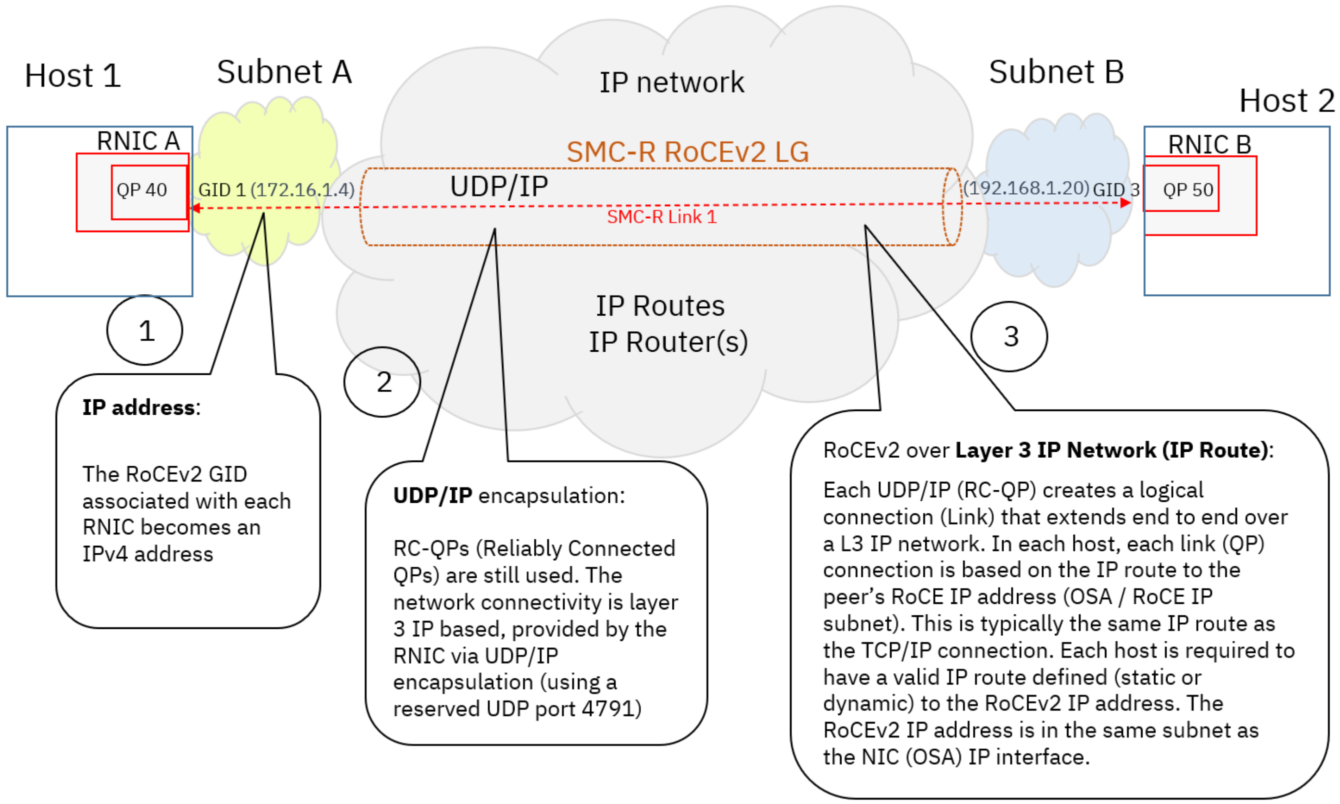
When the IBM Z operating systems[8] that support SMC-R are updated to support RoCEv2 connectivity feature, the multiple IP subnet solution of the SMC-Rv2 protocol can be exploited.

[8] The z/OS SMC-Rv2 support is provided by z/OS V2R5. For the Linux SMC-Rv2 support, refer to the following URL: https://linux-on-z.blogspot.com/p/smc-for-linux-on-ibm-z.html

The SMC-Rv2 and RoCEv2 multiple IP subnet eligibility requirements are summarized as follows, both the TCP client and server must:

1. Be updated (and enabled) to support SMC-Rv2
2. Be executing on an IBM z15 with access to an IBM RoCE Express2 feature that supports RoCEv2
3. Be defined with a common EID
4. Have an IP Route to the peer's RoCEv2 IP address (typically the same IP route defined for the TCP/IP connection)
5. **Not** be associated with a TCP/IP connection that requires IPSec encryption

The next figure illustrates the main concepts of SMC-Rv2 with RoCEv2 connected over multiple IP subnets. The IP route associated with the TCP/IP connection is used to form the SMC-Rv2 connection.



IP address:

The RoCEv2 GID associated with each RNIC becomes an IPv4 address

UDP/IP encapsulation:

RC-QPs (Reliably Connected QPs) are still used. The network connectivity is layer 3 IP based, provided by the RNIC via UDP/IP encapsulation (using a reserved UDP port 4791)

RoCEv2 over **Layer 3 IP Network (IP Route)**:

Each UDP/IP (RC-QP) creates a logical connection (Link) that extends end to end over a L3 IP network. In each host, each link (QP) connection is based on the IP route to the peer's RoCE IP address (OSA / RoCE IP subnet). This is typically the same IP route as the TCP/IP connection. Each host is required to have a valid IP route defined (static or dynamic) to the RoCEv2 IP address. The RoCEv2 IP address is in the same subnet as the NIC (OSA) IP interface.

Figure 21. SMC-Rv2 with RoCEv2 Connectivity

Figure 21 shows the key 3 differences in the connectivity with SMC-Rv2 with RoCEv2. The SMC-R Link architecture is reused for SMC-Rv2. Here, the GID becomes a real IP address replacing the link local GID. Note that RoCEv2 can also be used when the hosts are directly connected on the same LAN. SMC-Rv2 reuses the existing IP topology of the TCP/IP connection that formed the SMC-Rv2 connection. Similar to SMC-Rv1, for HA and load balancing it is recommended that user's provision two RoCE adapters and define equal cost IP routes (similar IP route recommendations for TCP/IP HA).

As shown in figure 4, SMC-R connections begin with the TCP/IP 3-way handshake. The TCP Option 254 is exchanged on the TCP Syn and Syn-Ack. When the TCP option is exchanged by both client and server, the next level of the SMC exchange occurs with the SMC Connection Level Control (CLC) 3-way handshake. This same SMCv1 connection setup model continues to be used for SMCv2 where SMC-Rv2 connections are formed using the TCP/IP connection. SMCv2 introduces a version in the SMC messages. The Version 2 updates for the SMC CLC and Link Layer Control (LLC) messages are shown in the appendix.

Now that the SMCv2 connection can set up over your IP network it is important that any IP Firewalls that are in the IP network path of the TCP/IP connection is configured to allow (or are updated to allow):

1. TCP Option 254 to flow and
2. UDP port 4791 (RoCEv2) to flow (open port 4791)

SMC-RV2 AND ROCEV2 HIGH AVAILABILITY

When each host administrator provisions two RoCE adapters with equal cost IP routes, SMC-Rv2 will form a fully redundant SMC-Rv2 Link Group.

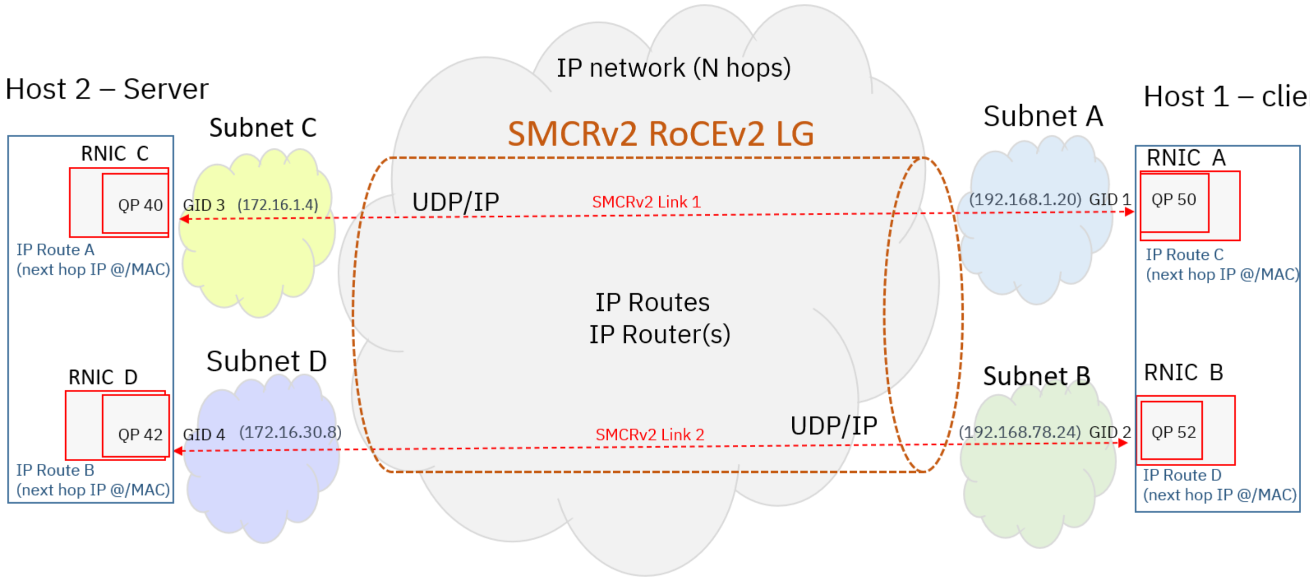


Figure 22. SMC-Rv2 / RoCEv2 Link Group Redundancy and High Availability

Some of the key concepts of SMC-Rv2 Link Groups are:

1. For HA, multiple RoCE adapters should be provisioned along with multiple equal cost IP routes to the peer host (i.e., reusing the TCP/IP routing topology).
2. The z/OS Netstat commands provide the “level of redundancy” for RoCEv2 LGs.
3. This example shows a typical IP configuration. There are several possible variations.
4. SMC-Rv2 Links can be Direct Links (same subnet) or Indirect Links (multiple IP subnets/IP hops).

SMC-RV2 AND ROCEV2 AT DISTANCE

SMC-Rv2 / RoCEv2 is designed for the Enterprise Data Center. SMC-Rv2 with RoCEv2 does not target IP traffic over the WAN. While SMC-Rv2 with RoCEv2 extends the scope and benefits of RDMA to additional workloads, there are considerations for limiting connections within the data center.

The previous topic that illustrated the concepts of the SMCv2 EID and considerations for centers at distance (see figures 9 and 10) provided some basic considerations for defining SMCv2 groups. The definition of a data center can be very fluid. Today's Hybrid Cloud Enterprise Data Centers continue to evolve in IP topology, shape, size, architecture, and their overall model regarding locality and distance. In addition, operating systems, guest images and containers can be frequently moved from System to System, cluster to cluster, and location to location, to meet various needs.

As illustrated, when defining EIDs current location of the OS should be a factor. In some cases, the operating system can offer configuration variables that can factor current location before setting a configuration variable. For z/OS, system symbolics can be used to set EID based on the current data center location (EastPlex versus WestPlex).

For these reasons, IBM does not offer a concise definition of maximum distance or maximum number of IP router hops for exploiting SMC-Rv2. Instead, the recommendation is to limit SMC-Rv2 communications among systems within the Enterprise. In the next section IBM offers considerations and guidelines for forming SMC EIDs.

USING EID(S) TO DEFINE SMC GROUPS.

Since SMC-Rv2 spans multiple IP subnets, considerations must be given for when the span or the "distance" among any two systems might be considered too far or out of scope. RoCEv2 is designed to extend the optimizations offered by RDMA technology across multiple IP subnets within the Enterprise Data Center. Within the data center, administrators have control over most of the key factors of the network design and the network allocated resources (e.g., bandwidth) that can have a direct impact on network performance. Several factors such as lossy or congested networks, competing / mix of traffic patterns, long distances (across external carriers), large number of IP hops and firewalls can all contribute to the overall performance of the network. IBM does not offer a specific set of rules (i.e., max distance, max no. of IP hops or packet loss rate) but instead recommends that you exploit SMC-Rv2 / RoCEv2 within your Enterprise data center over networks that you **directly control, monitor, and influence** the related network resources. Users should monitor the performance of their application workloads with and without SMC-Rv2 for meeting your required performance objectives (Service Level Agreements).

SMC-RV2 AND ROCEV2 MIGRATION CONSIDERATIONS

The SMCv2 protocol is compatible with down-level SMCv1 hosts. SMCv2 allows for v2 enabled hosts to continue to communicate with down-level SMCv1 hosts. This is true for both SMC-Dv2 and SMC-Rv2. The version and type of SMC to be used on each connection is negotiated within the SMC 3-way handshake. When all four types and versions of SMC (SMC-Dv1, SMC-Dv2, SMC-Rv1, SMCv2) are offered by the client the SMCv2 server will select the first eligible type and version of SMC in the following order, SMC-Dv2, SMC-Dv1, SMC-Rv2, and SMC-Rv1.

Along with the z/OS SMC-Rv2 support, SMC filters were also introduced to provide administrators with another level of control. The SMC Filters control all SMC connection types (i.e., when an SMC connection is excluded by a z/OS SMC Filter then TCP option 254 is not passed preventing all forms of SMC with the peer host).

Administrators can elect to enable SMC-Rv2 only or enable both SMC-Rv1 and SMC-Rv2. Using SMC filters z/OS users can provide an even more granular level of control. RoCEv2 offers QoS benefits over RoCEv1. For this reason, IBM recommends migrating to SMC-Rv2 when possible.

For SMC connections to setup over multiple IP subnets, users must verify that TCP option 254 and UDP port 4791 are both allowed to pass through any applicable firewalls.

SMC VERSION 2.1 FEATURE UPDATE OVERVIEW

A set of features are introduced in SMC Version 2 that are defined as an **SMC Version 2.1 (v2.1) feature update.** The v2.1 changes have a minor impact on the SMCv2 protocol. This section introduces v2.1 along with an overview of the v2.1 features. This section also briefly describes how each new feature impacts and interoperates with the Version 2 protocol (SMC V2 message updates). The details of v2.1 changes to message specifications are included in the Appendix.

SMC VERSION AND RELEASE SPECIFICATIONS OVERVIEW

The concept of an SMC dot release described as ***Version.Release*** is created to offer a continuous delivery of minor protocol enhancements of the current SMC version. A characteristic of minor features introduced with a dot release are features that have either no or minor impact on the current version of the SMC protocol.

Major enhancements of the SMC protocol (e.g. such as multiple IP subnet support) that have a significant impact on the current version of the protocol require a new version of the SMC protocol (such as SMCv2).

The terminology for the specifications for adherence to a dot release are described as follows:

1. Features with minor protocols changes are introduced with an SMC **Version.Release** number in two parts:
    a. **base features**: a set of base features defined with a new dot release and
    b. **supplemental features**: a set of supplemental (additional) features of the dot release introduced over time and defined after the initial definition of the dot release that are defined and exchanged in the dot release feature bit mask of the CLC exchange.
2. The dot release supplemental feature bits (bit mask):
    a. Are associated with a specific dot release. Each new dot release will reset the feature bits back to zero (i.e. feature bits only apply to a single version and release).
    b. Are designed to be added over time of the release along with their corresponding new features bits. If a host sets a release supplemental feature bit, it indicates the supplemental feature is implemented and exploited (used).
    c. Will typically also have an associated protocol change. If the feature bit is set, then the host must implement (define) and exploit (use) the associated protocol change.
3. The **Version.Release** is defined in the SMCv2 CLC messages. The CLC Version.Release exchange rules are described as follows:
    a. The client indicates the Version.Release in the CLC Proposal.
    b. The server reflects the supported level in the CLC Accept.
    c. The Accept message must either match the Proposal Version.Release or specify a supported previous Version.Release (of the same version).
    d. The Accept cannot specify a higher Version.Release.
    e. As each new Version.Release is defined it will also define the valid previous supported Version.Release numbers.
4. There are 3 levels of possible ***Version.Release*** host (client or server) support:
    a. **toleration**: a server is defined as a toleration server when it is down level, has not added support for a dot release specified in the CLC Proposal, but can tolerate (interoperate with or ignore) all protocol updates for the dot release without disruption. A toleration host applies only to the server.
    b. **compliant**: a host (client or server) is defined as compliant with a dot release when it has defined and implemented all base features of the current dot release but does not exploit the new features.

i. *A compliant server:*
    1. is compatible with clients of the same dot release.
    2. indicates support for the Version.Release in the CLC Accept message and specifies all base features in the CLC Accept message.
    3. does not exploit (use) any base or supplemental features. This means the vendor options will be set to zero.

ii. *A compliant client*:
    1. supports the Version.Release but does not exploit (use) any release features.
    2. The compliant client indicates Version.Release (e.g. v2.1) in the CLC Proposal message and indicates all existing base values (settings) for each feature.
    3. Does not pass (exploit) vendor options.
    4. If the server changes any of the base values in the CLC Accept, the compliant client should accept the values, but it can optionally decline the connection.

c. **exploitation**: a host (server or client) is defined as an exploitation host when it has implemented and exploits (uses) one or more features of the dot release. The exploitation host indicates support for the Version.Release in all CLC messages. The exploited features could be a base feature, or a dot release supplemental feature defined in the release features bit mask. An exploitation host could support the passing of vendor options.

5. For an OS to support a given dot release that is not the first dot release of a version, it must achieve ***complete compatibility*** with all features of all the previous dot releases. Support for dot releases cannot be skipped.

    Complete compatibility of a dot release:

    a. is defined as having compliance of all base features and
    b. compliance of all supplemental features defined in the release feature bit mask but
    c. does not require exploitation (usage) of any features.

    Supporting the next dot release does not require exploitation of any of the release features, but compliancy (compatibility) of all features must be achieved. For example, if v2.2 were to be defined in the future, then a v2.2 host must achieve complete compatibility (be compatible with) all features of v2.1.

V2.1 FEATURES OVERVIEW

Multiple base features are introduced and bundled under the initial v2.1 feature update. The initial v2.1 feature updates are specific to SMC-Rv2 only and are defined in the 3rd edition of this v2.1 specification. SMC-Rv2.1 or SMC-Dv2.1 features introduced (defined) after the initial SMC-Rv2.1 features will be considered v2.1 supplemental features.

SMCv2.1 features that are defined in the CLC Proposal message that are specific to an SMC type are only defined in the CLC Proposal message if the specific SMC Type is offered by the client. The client should only include the v2.1 features that correspond to the SMC type offered by the v2.1 client in the CLC Proposal. For example, when the v2.1 client includes SMC-Rv2.1 features the client should also offer SMC-Rv2. A v2.1 server should ignore v2.1 features that are included in the CLC Proposal that don't match the offered SMC type.

The 4th edition of this specification defines SMC-Dv2.1 features. The SMC-Dv2.1 features are the first v2.1 features defined as v2.1 supplemental features. In the future, additional v2.1 features could be introduced for SMC-Rv2 or SMC-Dv2. All subsequent supplemental v2.1 updates will be defined with updates to the v2.1 Supplemental Feature Bit Mask (see below).

The following list provides a summary of the v2.1 base features.

1. Feature: Maximum connections per LG
   Purpose: Allows each host to provision memory (i.e., RMBs) resources in a more efficient manner when an agreed max connections per LG can be exchanged, potentially improving scalability and storage management. The valid range for maximum connections per LG is 16 to 255.
2. Feature: Maximum Links per LG
   Purpose: Allows hosts that either don't have a need for a second link for High Availability (HA) or for platforms that have other forms of system HA architecture (not requiring redundant hardware such as a 2nd RNIC or QP). When a 2nd link is not used, first contact connections can setup much faster (avoiding ADD Link flows). When a LG is single link only, then the LLC Add Link is N/A for the life of the LG. Both symmetrical and asymmetrical LGs do not apply (single link only). The valid range for Maximum Links per LG is 1 or 2.
3. Feature: Vendor Unique Experimental Options
   Purpose: Allows unique vendors to exchange experimental SMC options. The options are undefined (opaque) to the protocol and have no impact on the protocol. If the experiment is proven successful, then consideration could be given to making the unique vendor option generally available as a formal SMC option (potential new feature).
4. Feature: v2.1 Supplemental Features Bit Mask
   Purpose: A two-byte v2.1 supplemental (additional) features bit mask is defined to allow the introduction of new v2.1 features for both SMC-Rv2.1 and SMC-Dv2.1 over time. Using the dot release supplemental feature bit mask, peer hosts can indicate support for new features as they are introduced in the future (features added after the initial v2.1 base feature update). The objective of the feature bit mask is to minimize ambiguity with migration scenarios. Each bit will indicate support for a unique new feature (up to 16 new features).
   a. *Exchanging Supplemental Features Bit in the CLC Exchange*

   i. The feature bits within the bit mask are sent from the client in the CLC Proposal indicating the client supported v2.1 supplemental features.
   ii. The server will respond with the server's supported v2.1 supplemental features in the CLC Accept.
   iii. The client will respond with any **common v2.1 features** (indicated in both the Proposal and Accept) in the CLC Confirm. Supplemental features can only be exploited when both the server and client both indicate mutual support.

b. New supplemental features will typically also require additional protocol CLC message changes. The host must indicate if a feature is supported in the bit mask and comply with any potential associated changes in the CLC messages applicable to the new feature. Setting a feature bit in the v2.1 dot release supplemental bit mask **indicates both compliance (definition) and exploitation (usage) of the new feature**. Usage of a new feature can be configuration dependent (i.e. a host can exploit a new feature but not be configured to use the new feature for a specific use case, e.g. SMC-D is not enabled).

V2.1 CONCEPTS AND COMPATIBILITY

A v2.1 host is based on SMCv2. A host is defined as v2.1 when v2.1 is indicated in the CLC messages within the existing version and release fields. When v2.1 is indicated in the CLC Proposal, then depending on the level of support of the server, it will either process the features (accept or negotiate) or drop the features when down level. A v2.1 host must also include a process for the v2.1 supplemental feature bit mask allowing for future v2.1 features. The v2.1 host is not required to exploit any of the specific v2.1 features, but it must comply with (define) all features, meaning that a v2.1 host cannot reject a CLC message due to the presence of a new v2.1 feature. Note that there are two levels or types of v2.1 servers. Refer to the definition of v2.1 compliant vs. exploitation server.

The v2.1 update does not change the length of any SMC-Rv2 messages. This aspect allows a down level SMC-R host to tolerate the feature updates. The v2.1 update does change the length of the v2.1 CLC Accept and Confirm messages. The length increase is due to an increased length of the First Contact Extension (FCE).

With this architecture, there will not be any migration (down level) or incompatibility issues for the introduction of the v2.1 features. A v2.1 client is defined as v2.1 or up-level when it indicates v2.1 in the CLC Proposal message. A v2.1 client must include all v2.1 fields defined for the SMC types being offered in the Proposal. A v2.1 compliant server must process all v2.1 definitions (features) and indicate v2.1 in the CLC Accept message (when the server supports v2.1). The v2.1 compliant server is not required to exploit any of the new 2.1 features. The v2.1 server must reply with all valid v2.1 fields for the SMC type that is being accepted (v2.1 fields must be non-zero). Optionally, the v2.1 host can exploit some of or all of the v2.1 features (see examples). Therefore, a v2.1 server must include all 2.1 fields with a valid supported value, as defined in the v2.1 specifications (appendix) as either the echoed value or support some form of negotiated value as defined by each 2.1 feature.

V2.1 FEATURE NEGOTIATION AND CONSIDERATIONS

1. The initial v2.1 features are specific to SMC-Rv2 and affect the SMC connectivity for the entire LG, for duration of the LG and therefore are only exchanged during first contact processing within the CLC First Contact Extension (FCE). When the CLC is for a subsequent contact for an existing known peer / LG, then the new features are not exchanged (FCE is not defined).
2. The First Contact Extension (FCE) was introduced with v2.0. First contact is defined when the server either:
    a. Recognizes that the client is a new client requiring a new LG (QP) to be created or
    b. Decides to create a new LG for an existing client (for other reasons). Creating Parallel link groups is supported for v2.0.
       First Contact is further defined to mean anytime a CLC Proposal results in the creation of a new LG (QP), as decided (required) by the server.
3. The First Contact Extension (FCE) is also applicable to SMC-Dv2. Although SMC-D does not support LGs (QPs or links) it does support a similar concept of a new vs. a known peer where existing SMC-D connections already exist. The SMC-Dv2 server uses the CLC Proposal ISMv2 GID/CHID to detect an existing peer[9].
4. It is possible that in the future new v2.1 features could be introduced that apply only to a specific SMC connection vs. the entire LG or ISM peer.
5. New v2.1 features are defined in the CLC handshake as:
    a. CLC Proposal = client’s preferred values
    b. CLC Accept = server’s default or accepted and final values
    c. CLC Confirm = client’s confirmation that the new values are acceptable (base features must match CLC Accept and supplemental features must be common among server and client)
       Note: If the client sends incorrect or other values for the 2.1 features in the Confirm message, then this is considered a protocol violation which requires the server to send a late DECLINE.
6. As with existing SMC protocol, the SMC server makes the final determination of the usage and values for the connection and LG (or ISM peer), including the new features. Some new features can be negotiated by the server.
7. The client must be given a means to DECLINE the new values. New DECLINE Reason Code(s) for incompatibility of 2.1 options (e.g. maxlinks and maxconns) will be defined by the OS implementation. This is for the case in which a client cannot accept the final value(s) from the server (this is believed to be rare).

   Note:
   CLC Decline Reason Codes are not defined by the SMC specifications. The RCs are defined by the OS implementation. Adding an OS type will aid in diagnosis (OS-specific RCs).
   New Decline reason codes will be necessary for v2.1 unique conditions. For the interpretation of the

[9] Programming Note: The OS implementation should define a structure (similar in concept to a QP) to anchor all connections with a specific SMC-Dv2.1 peer. For this SMC-Dv2.1 peer, all v2.1 features are defined to be the same.

reason codes, refer to the specific source OS documentation.

8. A v2.1 server “could” honor or negotiate the client’s preferred maxconns (or at least allow a new lower maximum (floor) value acceptable to the OS, e.g., client prefers 16, server offers 32, instead of 255). Maximum connections per LG could be a storage resource issue. See examples 4 and 5.
9. A v2.1 compliant OS defines v2.1 but does not exploit any new v2.1 options. This includes supporting the passing of External Vendor options. This means as a client the options (values) will not be included in the CLC Proposal and as a server the values must be zero, must not to be returned (echoed) in the CLC Accept. A v2.1 compliant OS must recognize the definition of the vendor options fields (verify that the field is set to zero and the fields are not to be used for any other purpose).

V2.1 CLC 3-WAY HANDSHAKE EXAMPLES

Example 1. Toleration Server (down level server that tolerates the new release)

v2.0 Server v2.1 Client

← Proposal (v2.1, max conns(16), max links(1), features(c-mask bits), vendor opts(options))

Accept (v2.0). (Down-level server, all new features are dropped / ignored, client must drop back to v2.0)

→

Confirm (v2.0)

←

Example 2. Compliant or Exploitation Server (down-level client tolerated by up-level server)

v2.1 Server v2.0 Client

← Proposal (v2.0)

Accept (v2.0). (Down-level client, all new 2.1 features are N/A, must drop back to v2.0)

→

Confirm (v2.0)

←

Example 3. v2.1 Compliant Server (implements new release base features, v2.1 compatible but does not exploit any new features)

← Proposal (v2.1, max conns(16), max links(1), features(c-mask bits), vendor opts(options))

Accept (v2.1, max conns(255), max links(2), features(zero), Vendor opts(zero))

→

Confirm (v2.1, max conns(255), max links(2), features(zero), Vendor Opts(zero))

←

Example 4. v2.1 Exploitation Server and Client (server implements and exploits some new features, max conns only)

← Proposal (v2.1, max conns(16), max links(1), features(c-mask bits), vendor opts(options))

Accept (v2.1, max conns(16), max links(2), features(zero), Vendor opts(zero))

→

Confirm (v2.1, max conns(16), max links(2), features(zero), Vendor Opts(zero))

←

Example 5. v2.1 Exploitation Server and Client (server implements and supports all new release base and supplemental features)

← Proposal (v2.1, max conns(16), max links(1), features(c-mask bits), vendor opts(options))

Accept (v2.1, max conns(32), max links(1), features(s-mask bits), Vendor opts(options))

→

Confirm (v2.1, max conns(32), max links(1), features(s&c-mask bits), Vendor Opts(options))

←

Example 6. v2.1 Exploitation Server and Client, Client supports SMC-Dv2.1 and only offers SMC-Dv2.1 (server implements and supports all v2.1 base features but does not support any supplemental features)

← Proposal (v2.1, features(c-mask bits-Emulated-ISM), vendor opts(options))

Decline (v2.1, no matching ISM GID)

→

Example 7. v2.1 Exploitation Server and Client, Client supports SMC-Dv2.1 and only offers SMC-Dv2.1 (server implements and supports all v2.1 base features and the Emulated-ISM supplemental feature)

← Proposal (v2.1, features(c-mask bits-Emulated-ISM), vendor opts(options))

Accept (v2.1, features(s-mask bits-Emulated-ISM), Vendor opts(options))

→

Confirm (v2.1, features(s&c-mask bits-Emulated-ISM device), Vendor Opts(options))

←

Example 8. v2.1 Exploitation Server and Client, Client offers both SMC-R and SMC-D (server implements and supports all v2.1 base features and the Emulated-ISM supplemental feature, Accepts SMC-Rv2.1)

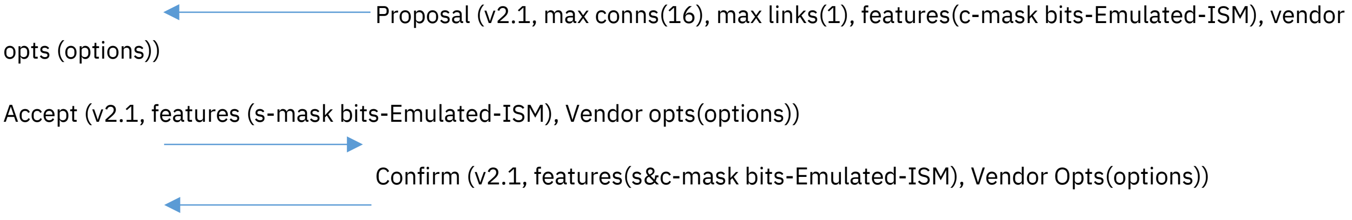
← Proposal (v2.1, max conns(16), max links(1), features(c-mask bits-Emulated-ISM), vendor opts (options)) Accept (v2.1, max conns(32), max links(1), features(s-mask bits-Emulated-ISM), Vendor opts(options))

→

Confirm (v2.1, max conns(32), max links(1), features(s&c-mask bits-Emulated-ISM), Vendor Opts(options))

←

Example 9. v2.1 Exploitation Server and Client, Client offers both SMC-R and SMC-D (server implements and supports all v2.1 base features and the Emulated-ISM supplemental feature, Accepts SMC-Dv2.1)



Proposal (v2.1, max conns(16), max links(1), features(c-mask bits-Emulated-ISM), vendor opts (options))

Accept (v2.1, features (s-mask bits-Emulated-ISM), Vendor opts(options))

Confirm (v2.1, features(s&c-mask bits-Emulated-ISM), Vendor Opts(options))

Notes:

1. When the client is down-level, sends a v2.0 proposal, then the server must return a 2.0 Accept (all v2.1 options are not defined). See example 2.
2. Max conns and max links values sent in the CLC Proposal are the client preferred values.
3. With the exception of the feature bit mask (supplemental features), the v2.1 values sent in the Accept message are the final values. The client must accept the values or DECLINE the connection. See item 9 for a description of how the feature bit mask is exchanged.
4. Max conns and links values sent in the CLC Accept are the final values (server dictates). The server can either honor the client's preferred values or return different (negotiated but final) values.
5. Vendor unique experimental options are defined in the CLC proposal, accept, and confirm messages. The values that flow in each message are based on vendor definitions as defined by each unique vendor. The vendor options are not defined to and have no impact on the protocol. The protocol does not define how the vendor options are defined or passed to the SMC stack (implementation). If a v2.1 host does not support the passing of vendor options (i.e. a compliant v2.1 host), then the field must be set to zero.
6. Max conns is defined as a range of 16 to 255.
7. Max links has a defined range of 1 or 2.
8. Max links is also an existing field in the LLC Confirm Link message. The LLC max links field must reflect the final negotiated value from the v2.1 CLC First Contact Processing.
9. The supplemental features (exchanged in the feature bit mask) sent in the CLC Confirm message are the common supplemental features exchanged by the client and server and are used for this connection. See example 4.
10. SMC-Dv2.1 introduces the Emulated-ISM device feature. This is the first v2.1 supplemental feature. A client can support Emulated-ISM devices but not offer a Emulated-ISM device (based on how the client is configured). The client should set the feature supported bit even when an Emulated-ISM device is not offered.

    A client can support v2.1 and enabled for SMC-Dv2 but:

    a. not support SMC-Dv2.1 Emulated-ISM, or
    b. support v2.1 and offer an Emulated-ISM device only, or
    c. support v2.1 and offer both an Emulated-ISM device combined with an ISM device. The client should order the ISM devices in the GID/CHID array in the client's preferred order of connectivity.

SUMMARY

Shared Memory Communications is a powerful IBM Z Enterprise Data Center network communications solution that has the potential to offer savings in network-related CPU cost, reduce latency, and increase throughput. The original version of SMC is limited to TCP connections for hosts (client and server) that have direct access to the same IP subnet. The single IP subnet limitation restricts SMC exploitation to a subset of IBM Z workloads. In some configurations, this limitation could be overly restrictive.

Shared Memory Communications Version 2 (SMCv2) lifts the single IP subnet limitation for both SMC-D and SMC-R. SMCv2 extends the SMC connectivity eligibility but SMCv2 does not change the SMC communications capability. The performance of SMCv2 should provide the same benefits as SMCv1 (relative to TCP/IP over standard Ethernet NICs). When TCP/IP connections are eligible for both versions of SMC, for various QoS reasons SMCv2 is preferred.

Shared Memory Communications-Direct Version 2 (SMC-Dv2) with ISMv2 lifts the single IP subnet limitation for an IBM Z system (CPC) extending the SMC-D solution and potential savings to additional IBM Z workloads in the Enterprise.

Shared Memory Communications-RDMA Version 2 (SMC-Rv2) with RoCEv2 lifts the single IP subnet limitation for workloads across separate physical IBM Systems Z machines spanning multiple IP subnets extending the potential savings to many additional workloads within the Enterprise Data Center.

APPENDIX

SMCV2 PROTOCOL MESSAGE SPECIFICATIONS (V2 WIRE FLOWS, UPDATED FOR V2.1)

This section describes the updates to the SMC messages required for SMCv2. The most significant change is to the CLC Proposal Message. The CLC Proposal Message is sent by the client and is common to both SMC-D and SMC-R. The client can propose (offer) a single SMC Type (e.g., SMC-Rv2 only) or the client can propose (offer) all four SMC types (SMC-Rv1, SMC-Dv1, SMC-Rv1, and SMC-Rv2). Once the server selects the specific SMC type, the server will respond with the appropriate CLC Accept message. The CLC Accept and Confirm messages are specific to the specific SMC Type. Note that the V2 is compatible with SMCv1 only (down-level) hosts. The CLC Proposal format retains the V1 definitions (same offset and contents are unaltered) allowing V1 only (down-level) hosts to continue processing the original V1 CLC Proposal.

SMC-Rv2 also defines updates to the LLC messages used to manage SMC-Rv2 Links and Link Groups.

SMCV2 TOLERATION

Operating systems will be required to provide toleration updates to their existing base SMC server-side CLC Proposal logic that receives a SMCv2 CLC Proposal message to “tolerate” (allow/ignore) the new version and length yet continue to process the V1 only fields of the CLC message.

SMCV2.1 FEATURE UPDATE

The SMCv2 CLC messages defined in this section have been updated for the SMCv2.1 feature update. All CLC messages are now shown in the v2.1 format.

SMC-V2 CLC PROPOSAL MESSAGE (SMCV2.1 FORMAT)

CLC PROPOSAL BASE MESSAGE

Offset	Length	Description
0	4	Eye catcher ‘SMCR’ (EBCDIC) message start
4	1	CLC Message Type 01 (CLC Proposal)
5	2	CLC Message Length (variable)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0001’ Version 1) (b’0010’ Version 2) **Note:** The SMCv2 CLC Proposal defines both SMCv1 and SMCv2. After the CLC Proposal, once the SMC Type has been selected by the server all remaining CLC messages are either V1 or V2 only. Bits 4-5 SMCv2 Type (b‘00’ = SMC-R, b‘01’ = SMC-D, b‘11’ = Both SMC-R and SMC-D, b’10’ = None, neither V2 types offered) Bits 6-7 SMCv1 Type (b‘00’ = SMC-R, b‘01’ = SMC-D, b‘11’ = Both SMC-R and SMC-D, b’10’ = none, neither V1 types offered)
8	8	SMC-R Client Peer ID
16	*	SMC Client V1 GID Information SMCv1 can offer up to two types of GIDs (SMC-R and SMC-D) SMCv2 GIDs are in the V2 Client Options Area GIDs are only valid (present) when the SMCv1 or SMCv2 Types indicates that the corresponding SMC version is offered.

16	32	SMCv1 Client Information (SMC-R and SMC-D)
16	16	SMC-R V1 Client Preferred GID
32	6	SMC-R V1 Client Preferred MAC address
38	2	SMCv1 IP Subnet Extension Offset (can be zero when SMCRv1 is not offered – SMCv1 Type field)
40	8	SMC-D V1 Client ISM-GID (associated ISM GID)
48	2	ISMv2 CHID (associated ISM CHID)
50	2	SMC Version 2 Extension Offset (applicable when SMC V2)
52	8	SMCv2.1 Vendor Unique Experimental Options Area – see notes
52	3	V2.1 Vendor Organizationally Unique Identifier (OUI)
55	5	V2.1 Vendor Unique Experimental Options (format defined by vendor)
60	20	Reserved for growth (zeros)
80	*	End of base (fixed length) portion of CLC Proposal Message
		V1 IP Subnet Extension (when applicable)
		V2 Extension (when applicable)
*	4	Eye catcher ‘SMCR’ (EBCDIC) message end **Notes:** • For V1, this field follows the Subnet extension. • For V2, this field follows the last extension.

Notes:

1. All existing V1 fields in the CLC Proposal Base message are not altered, moved, or changed in any way (must remain compatible with down-level hosts).
2. Version 2 indicates the SMC V2 protocol. The version release number is also indicated in the V2 extension. The release number allows for future V2 protocol, maximums, or format changes.

3. SMCv2 Types indicate which types of SMCv2 are offered.
4. The ISMv2 CHID is present when both SMCDv2 is offered and an associated ISM GID is offered. The associated ISM GID can be used for both V1 and V2.
5. The offset to the V2 Extension is always present for V2. The offset field indicates the number of bytes that must be skipped after this offset field to access the V2 Extension. All offset fields in the Proposal message follow this convention.
6. The base CLC message is followed by the IP subnet extension (when present) and then the V2 extension (when present). The SMCR EBCDIC trailer is appended to the end of the entire message (after the last extension).
7. The MAC address field is always required.
8. SMCv2.1 Vendor Unique Experimental Options Area Definitions and Usage Notes:
    a. The Vendor Experimental Options Area is defined with the SMCv2.1 feature update to allow unique vendors to exchange or negotiate experimental SMC Version 2 options during the SMCv2 CLC handshake.
    b. This area is only defined and valid for SMCv2.1. The client must offer SMC-Rv2, SMC-Dv2 or both in the proposal and indicate v2.1.
    c. The Experimental Options are intended to be used in a controlled (test) environment; the protocol allows for SMCv2 servers to ignore non-supported 2.1 options.
    d. The OUI is defined by IEEE registration authority standards. If the OUI field is zero, then the remaining 5-byte options area must also be zero. A zero OUI field indicates that no options are offered.
    e. If the OUI is not zero, then SMCv2 Vendor Experimental options are offered by the client host.
    f. The definition of the 5-byte options area that accompanies the OUI is defined by the specific vendor identified by the OUI.
    g. If the server does not support Vendor Experimental Options or does not recognize the OUI, then the options are ignored and set to zero in the CLC Accept message.
    h. If the server supports the experimental options, then the SMCv2 Accept Message defines a 4-byte experimental options area. The OUI is not returned in the Accept message.

CLC PROPOSAL IP SUBNET EXTENSION AREA

0	*	V1 IP Subnet Extension Area The IP subnet extension is present (applicable) when SMCRv1 or SMCDv1 is offered. When present, this extension is addressed by using the Subnet Extension Offset field (at +38). This extension must precede the V2 Extension (when both are present).
0	**5**	**Client IPv4 Subnet Mask (IPv4 only)**
0	4	Subnet Mask
4	1	Number of Significant Bits in mask
5	2	Reserved
7	*****	**Client IPv6 Prefix Array (zero for IPv4)**
7	1	Number of IPv6 Prefixes in Prefix array (1 - 8)
8	*	Prefix Array, variable length array (based number of entries in array) N number of 17-byte entries (16-byte prefix followed by 1-byte prefix length field)

Note:

This extension is optional and not present with SMC V2 when SMC V1 types are not offered. If the extension is present without offering V1 types, it is N/A and is ignored.

CLC PROPOSAL V2 EXTENSION (SMCV2.1 FORMAT)

0	*	SMCv2 Extension - Client Options Area (SMCRv2 & SMCDv2) The V2 Extension is always present for SMC V2. The applicable fields are based on the SMC options offered by the client. The V2 Extension is addressed by using the V2 Extension offset field at + 50. This extension follows the V1 subnet extension when present.
0	8	SMCv2 Extension - Client Options Area Header
0	1	EID Number (No of user-defined EIDs in the EID Array Area)
1	1	ISMv2 GID Number (No of ISMv2 Unassociated GID Entries in ISMv2 GID Array Area)
2	1	Flag 1 (bit 8) - Reserved
3	1	Flag 2 (bit 8) - Bit 0-3 b‘xxxx’ SMC Version Release Number Same format as Version number b’0000’ Release 0 - V2.0 (Release 0) b’0001’ Release 1 - V2.1 (Release 1 Feature Update) Bit 4-6 b’000’ - Reserved Bit 7 b’0’ - SEID indicator B’0’ - SEID not present B’1’ - SEID present (offered by client)
4	2	Reserved
6	2	SMCDv2 Extension Offset (if present) (Extension is not always present. See notes.)
8	16	RoCEv2 GID (IPv4 or IPv6 address)
8	16	RoCEv2 GID IPv6 address (when IPv6)

8	12	RoCEv2 GID IPv4 reserved (when IPv4)
20	4	RoCEv2 GID IPv4 address (right aligned)
24	1	Max Conn per LG (v2.1 client preferred max)
25	1	Max Links per LG (v2.1 client preferred max)
26	2	V2.1 Feature bit mask (v2.1 Client Supplemental Features Support) Bits 0- 15 b'xxxx xxxx xxxx xxx1' Emulated-ISM Device Support **Note:** X indicates a reserved (future) feature bit and must be off.
28	12	Reserved
40	*	EID Array Area – variable length (32 bytes * EID Number) Min of 0 and max of 8 EIDs. See format notes.
*	0	End of CLC Proposal Message V2 Extension

Notes:

1. The V2 Extension defines all V2 common attributes.
2. At least one user-defined EID is required for SMC V2 for RoCEv2 and UEIDs are optional for SMCDv2. The SEID (only) could be offered for SMCDv2.
3. SMCDv2 can be offered without requiring or offering any ISMv2 unassociated GIDs (i.e., the associated ISM GID can also be used for V2 connectivity).
4. The RoCEv2 GID is an IP address. The GID uses 16 bytes for IPv6 or 4 bytes (right aligned) when IPv4. The IP version must match the IP version of the associated TCP connection. All IP addresses (GIDs) are defined as 16 bytes. An IPv4 address can be represented in two formats:
    a. IPv4 address is right aligned (leftmost 3 words of zeros) or
    b. IPv4 mapped IPv6 address format (leftmost 80 bits of zeros + xFFFF + IPv4 address)
    See RFC 4291.
5. SMC V2 is defined with a dot release number as SMC V2.0. The dot release defines the format of the V2 extension and allows for future minor protocol or format changes within SMC V2. In the initial V2 version, V2.0 is the only release number defined for SMC V2.
6. The SMCDv2 Extension defines attributes that are unique to SMCDv2 and might not be present.
7. The EID is fixed length 32 bytes in ASCII format. The supported ASCII characters are (alpha-numeric) capital A - Z, 0 - 9 only with special characters hyphen and dot. The first char cannot be a special character and consecutive dots are not allowed. When necessary, the UEID is padded with character blanks to form a full fixed-length 32-byte UEID.
8. V2.1 Maximum connections per LG valid range is 16 to 255.
9. V2.1 Maximum links per LG valid range is 1 or 2.

10. V2.1 feature bit mask is defined to allow for additional v2.1 supplemental features added after the initial v2.1 feature update. Each of the 16 bits will define a potential future feature allowing the client to indicate in the v2.1 Proposal message support for each client supported v2.1 supplemental feature.

CLC PROPOSAL SMC-DV2 EXTENSION (SMC-DV2.1 FORMAT)

0	*	SMCDv2 Extension	
0	32	System EID System EID (32 bytes) applies to ISMv2 only - may or may not be offered by the client. If SEID is not offered this field is binary zeros.	
32	16	Reserved	
48	*	ISMv2 GID-CHID Array Area - Variable Length Based on ISMv2 GID Number (contains 0 - 8 entries)	
	ISM GID Entry	ISMv2 GID	8-byte ISM GID (see Extended GID format note 4)
		ISMv2 CHID	2-byte ISM CHID (see Emulated-ISM CHID format note 4)
*	0	End of CLC Proposal Message V2 SMCDv2 Extension	

Notes:

1. The SMCDv2 extension is specific to SMCDv2. This extension is only present when SMCDv2 is offered and either SEID is also offered or at least one unassociated ISMv2 GID is offered.
2. The SEID (System EID) applies to SMCDv2 only and is optional. The format of the SEID is identical to the UEID format (32-byte character format).
3. The ISM GID entries in this extension define unassociated ISM GIDs offered for this connection. The entries are each 10 bytes, 8-byte ISM-GID, and 2-byte ISM CHID (VCHID). Also see the Extended GID format in the next note. SMCDv2 connectivity using the internal ISMv2 L3 connectivity can be formed using the associated ISM GID (passed in the base CLC message) or using the ISMv2 unassociated GIDs passed in this extension. Unassociated ISM GIDs support V2 connectivity only.
4. Extended GID Format
   The SMC-Dv2.1 client can send Extended ISM GID(s) associated with Emulated-ISM devices in the SMC-Dv2.1 CLC Proposal message in the existing GID-CHID array. Extended GIDs consume two consecutive entries within the array with the following format:
   a. Extended GID 8-byte part 1 (1st 8 bytes)
   b. Emulated-ISM CHID (reserved values)
   c. Extended GID 8-byte part 2 (2nd 8 bytes)
   d. Emulated-ISM CHID (same Emulated-ISM CHID value repeated from part 1)

   It is considered a protocol violation when a reserved CHID value is used but the CHID value is not repeated in two consecutive array entries. For additional usage information and the specifications for Emulated-ISM devices refer to the ISMv2.1 Emulated-ISM Device Feature topic.

V2 CLC ACCEPT AND CONFIRM FIRST CONTACT EXTENSION (FCE) - SMC-DV2 AND SMC-RV2

SMC V2 defines additional SMC-Rv2 Link Group control and SMCv2 network management information about an SMC V2 peer host accepting a V2 connection and the type of V2 Link being created (for SMC-Rv2). The host *first contact information* is exchanged in both the V2 CLC Accept and Confirm messages. The FC information is only passed for the first contact sequence with a new peer. For subsequent connections with a known peer using the same SMC-Rv2 LG or SMC-Dv2 logical link the FC information is redundant and should not be provided. **Any reason for the creation of a new SMC-Rv2 LG (even when the client is known) or creating a new SMC-Dv2 Link is considered a first contact sequence (i.e., creating a new LG (first QP of the LG) = first contact), FCE is required. Creating the alternate link (QP) is not considered FC.**

The V2 CLC Accept and Confirm messages define a First Contact Extension (FCE) area. The FCE is only valid when the server sets the CLC Accept First Contact flag on (indicating a new LG). The client only includes the FCE in the CLC Confirm message when it is responding to the first contact Accept message.

The SMC V2 CLC First Contact Extension (FCE) is:

1. Provided to identify information related to forming a new SMC-Rv2 LG, such as the V2 Link Type (direct versus indirect). An additional purpose is to improve SMC V2 network management capabilities by providing administrators (system programmers, operations, and support or diagnostics) with additional information about SMC V2 host connectivity (peer OS type and hostname).
2. Only applicable for first contact CLC exchange sequence with a new peer SMC host when forming a new LG. The FCE **must be provided** when the server sets the first contact flag on in the CLC Accept and when the client sets the FC flag in the corresponding CLC Confirm message.

   Notes:

   - If the FCE is included in the Accept/Confirm without the FC flag set on (not first contact sequence), the presence of the FCE must be tolerated but ignored. All FCE information is to be ignored by the receiver.
   - All CLC messages must provide an accurate length. The presence or absence of the FCE must be accurately reflected in the CLC message length.

   In addition to identifying the V2 Link type, the FCE represents a significant improvement in SMCv2 network management capability. When the FCE is present, all FCE fields are required.

The FCE hostname is a character field with the following specifications:

1. ASCII character name representing the CLC message sender's hostname (the peer host).
2. Valid ASCII(7) characters for hostname are letters a to z (mixed case), digits 0 - 9, dot and hyphen (-).
3. The FCE hostname field is fixed length 32 bytes. When the hostname is less than 32 bytes, the field is padded with ASCII blanks. When the name source is greater than 32 bytes, it must be truncated.
4. A blank character within the name denotes the end of the name. The receiver can elect to save the shorter name or save the entire padded 32 bytes.
5. If an invalid ASCII character appears within the hostname, the hostname is invalid and might not be saved or used by the receiving host (i.e., the receiver is not required to validate hostname).
6. Missing or invalid FCE information such as hostnames has no impact on the CLC setup processing. The invalid field might not be used but a connection failure is not warranted.
7. The source of or how an OS derives or creates the FCE hostname is based on OS implementation and is not defined by the V2 protocol. If the hostname comes from the domain name, it must not exceed the 32-byte limit.

How an OS saves, uses, or implements the FCE network management-related information, such as the peer's hostname, for network management displays, APIs, diagnostics, or other tools are beyond the scope of this paper (based on OS implementation).

The definition and format of the FCE is described along with the CLC Accept and Confirm messages on the following pages.

CLC ACCEPT MESSAGE (SMC-DV2.1 FORMAT)

Offset	**Length**	**Description**
0	4	Eye catcher ‘SMCD’ (EBCDIC) message start (constant xE2D4C3C4)
4	1	CLC Message Type 02 (CLC Accept)
5	2	CLC Accept Message (SMC-D) Length 78 Bytes (without FCE)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0010’ Version 2) Bit 4 b‘0’ Contact (b’1’ = first contact, b’0’ subsequent contact) Bit 5 b’0’ Reserved (zero) Bit 6-7 b’00’ SMCv2 Type (b‘01’ = SMC-Dv2)
8	8	SMCv2 Server ISMv2-GID
16	**10**	**Server DMB Information**
16	8	DMB Token
24	1	DMBE (connection) Index
25	1	DMBE Size Bit 0-3 b’0000’ DMBE Size (interpretation is the same as SMC-R RMBE size) Bit 4-7 b’0000’ Reserved
26	2	Reserved - Available
28	4	Server Link ID
32	2	ISMv2 VCHID (common internal fabric with the server and client) See note 1
34	32	EID (Negotiated Common EID selected by the server)
66	8	SMC-Dv2.1 Server Extended GID Part 2 (concatenate with Server GID Part 1)

74	*	First Contact Extension – FCE only present when first contact flag is on SMC-Dv2 FCE Length = 36 bytes SMC-Dv2.1 FCE Length = 52 bytes
74	4	FCE Header
74	1	FCE Header – reserved
75	1	FCE Header Flag 1 (bit 8) Bit 0-3 OS Type (b’0001’ = z/OS, b’0010’ = Linux, b’0011’ = AIX, b’1111’ = unknown) Bit 4-7 b‘xxxx’ SMC Version Release Number Same format as Version number b’0000’ Release 0 – V2.0 (Release 0) b’0001’ Release 1 – V2.1 (Release future)
76	2	FCE Header – reserved
78	32	FCE Peer Host Name (ASCII character – padded with ASCII blanks)
110	2	FCE – Reserved (v2.1)
112	2	FCE v2.1 Feature bit mask (bit 16) – Server v2.1 Supplemental Features supported Bits 0- 15 b’xxxx xxxx xxxx xxx1’ Emulated-ISM Device Support **Note:** X indicates a reserved (future) feature bit. Must be off.
114	4	FCE Vendor Unique Experimental Options Area (v2.1)
118	8	FCE – reserved (v2.1)
*	4	Eye catcher ‘SMCD’ (EBCDIC) message end (offset is either 74 or for v2 FCE then offset is 110 or for v2.1 FCE then offset is 126) (constant xE2D4C3C4)

Notes:

1. When the server replies (CLC Accept) with an Emulated-ISM CHID, then the server must include an Extended GID. Part 2 of the Extended GID is included in the CLC Accept Server Extended GID field.
2. The FCE v2.1 Feature bit mask indicates which specific v2.1 supplemental features are supported by the server. Features in the Accept message can only be used if the feature was also supported by the client CLC Proposal.

CLC ACCEPT MESSAGE (SMC-RV2.1 FORMAT)

Offset	Length	Description
0	4	Eye catcher ‘SMCR’ (EBCDIC) message start (constant xE2D4C3D9)
4	1	CLC Message Type 02 (CLC Accept)
5	2	CLC Accept Message (SMC-R) Length 108 bytes (without FCE)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0010’ Version 2) Bit 4 b‘0’ Contact (b’1’ = first contact, b’0’ subsequent contact) Bit 5 b’0’ SMC Type RC Saved (this bit is only defined when the Proposal indicated SMC Type = Both. b’1’ RC Saved by Server indicates why SMC-D was not selected by the server, b’0’ server did not save RC and is most likely down level or not enabled for SMC-D) Bit 6-7 b’00’ SMCv2 Type (b‘00’ = SMCRv2) – commentary only
8	8	SMCv2 Sender (server) Peer ID
16	16	SMCRv2 Server GID (IPv4 or IPv6 address, IP protocol version must match TCP connection)
32	6	SMCRv2 MAC Address (See MAC address note.)
38	3	SMCRv2 Server RoCEv2 QP Number
41	**9**	**Server RMB Information**
41	4	Rkey
45	1	RMBE (connection) Index
46	4	RMBE Alert Token

50	1	Flag 2 B’0000’ RMBE Size B’0000’ MTU Size (See MTU note.)
51	1	Reserved
52	8	RMB PCI Virtual Address
60	1	Reserved
61	3	PSN
64	32	EID (Negotiated EID selected by server)
96	8	Reserved
104	36	First Contact Extension - FCE only present when first contact flag is on
104	4	FCE Header
104	1	FCE Header Flag 0 (Bit 8) Bit 0 V2 Link Type b‘1‘ = v2_Direct, b‘0‘ = v2_Indirect Bit 1-7 Reserved (zero)
105	1	FCE Header Flag 1 (bit 8) Bit 0-3 OS Type (b’0001’ = z/OS, b’0010’ = Linux, b’0011’ = AIX, b’1111’ = unknown) Bit 4-7 b‘xxxx’ SMC Version Release Number Same format as Version number b’0000’ Release 0 – V2.0 (Release 0) B’0001’ Release 1 – V2.1 (Release 1 Feature Update)
106	2	FCE Header - reserved
108	32	FCE Peer Host Name (ASCII character - padded with ASCII blanks)

140	01	FCE Max Connections per LG (v2.1)
141	01	FCE Max Links per LG (v2.1)
142	02	FCE Feature Bit Mask (v2.1) – v2.1 supplemental features supported by the server
144	04	FCE Vendor Unique Experimental Options Area (v2.1)
148	8	Reserved
*	4	Eye catcher ‘SMCR’ (EBCDIC) message end (offset is 104 or 156) (constant xE2D4C3D9)

Notes:

1. The MAC address is defined for the SMC-Rv2 CLC Accept. However, the purpose of the MAC address in the CLC Accept is based on link type. When the:
    a. Link type is **direct**. The MAC address must be used by the client to create the client’s v2 QP for the next hop MAC address to reach the server.
    b. Link type is **indirect**. The MAC address becomes informational (e.g., possibly used for diagnostics or administrative purposes).
2. The MTU is used just as it was in SMC-Rv1 (the smaller value exchanged is set by the server and used for the MTU size for the link in both directions). When the link is indirect, it is possible that an IP router hop could have a lower value and use IP fragmentation that should be transparent to the hosts (other than potential performance-related issues).
3. All IP addresses (GIDs) defined within all SMC messages are defined as 16 bytes. An IPv4 address can be represented in two formats:
    a. IPv4 address is right aligned (leftmost 3 words of zeros) or
    b. IPv4 mapped IPv6 address format (leftmost 80 bits of zeros + xFFFF + IPv4 address)
    See RFC 4291.
4. Maximum connections per LG from the server represents the final value for this LG. Valid values are 16 to 255. The client must accept this value or send DECLINE. Added with v2.1.
5. Maximum links per LG from the server represents the final value for this LG. The client must accept this value or send DECLINE. Valid values are 1 or 2. Added with v2.1.
6. Vendor Unique Experimental Options Area Definition and Usage Notes.
    a. Vendor Experimental Options Area is defined in v2.1 only to allow unique vendors to exchange or negotiate experimental SMC Version 2.1 options during the SMCv2 CLC handshake.
    b. Refer to the SMCv2 Proposal Message for complete usage notes and definition.
    c. The presence of options (non-zero) in the v2.1 CLC Accept message indicates that the v2.1 server supports the offered experimental options received from the v2.1 client in the v2.1 CLC Proposal message. The options in the Accept message represent the corresponding server options. The

definition of the options is unique to and defined by the vendor identified in the OUI of the CLC Proposal.

7. The feature bit mask represents v2.1 features (added after the v2.1 initial feature update) supported by the server. Features in the Accept message can only be used if the feature was also supported by the client CLC Proposal.

CLC CONFIRM MESSAGE (SMC-DV2.1 FORMAT)

Offset	Length	Description
0	4	Eye catcher ‘SMCD’ (EBCDIC) message start (constant xE2D4C3C4)
4	1	CLC Message Type 03 (CLC Confirm)
5	2	CLC Confirm Message (SMC-D) Length 78 Bytes (without FCE)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0010’ Version 2) Bit 4 b‘0’ SMCv2 Contact (b’1’ = first contact, b’0’ = subsequent contact) applicable to V2 only Bit 5 b’0’ Reserved (zero) Bit 6-7 b’00’ SMCv2 Type (b‘01’ = SMC-Dv2)
8	8	SMC-Dv2 Sender (client) ISMv2-GID
16	**10**	**Client DMB Information**
16	8	DMB Token
24	1	DMBE (connection) Index
25	1	DMBE Size Bit 0-3 B’0000’ DMBE Size (interpretation is the same as SMC-R RMBE size) Bit 4-7 B’0000’ Reserved
26	2	Reserved
28	4	Client Link ID
32	2	ISMv2 VCHID (common internal fabric confirmed by the client)- see note 1
34	32	EID (Negotiated EID confirmed by client)

66	8	SMC-Dv2.1 Client Extended GID Part 2 (concatenate with Client GID Part 1)
74	*	First Contact Extension - only present when first contact flag is on SMC-Dv2 FCE Length = 36 bytes SMC-Dv2.1 FCE Length = 52 bytes
74	4	FCE Header
74	1	FCE Header - reserved
75	1	FCE Header Flag 1 (Bit 8) Bit 0-3 OS Type (b’0001’ = z/OS, b’0010’ = Linux, b’0011’ = AIX, b’1111’ = unknown) Bit 4-7 b‘xxxx’ SMC Version Release Number Same format as Version number b’0000’ Release 0 - Version N.0 (Release 0) B’0001’ Release 1 - Version N.1 (Release 1 - future)
76	2	FCE Header - reserved
78	32	FCE Peer Host Name (ASCII character - padded with ASCII blanks)
110	2	FCE – Reserved (v2.1)
112	2	FCE v2.1 Feature bit mask (bit 16) - client and server common Features Bits 0- 15 b’xxxx xxxx xxxx xxx1’ Emulated-ISM Device Support **Note:** X indicates a reserved (future) feature bit. Must be off.
114	4	FCE Vendor Unique Experimental Options Area (v2.1)
118	8	FCE – reserved (v2.1)
*	4	Eye catcher ‘SMCD’ (EBCDIC) message end (offset is either 74 or for v2 FCE then offset is 110 or for v2.1 FCE then offset is 126) (constant xE2D4C3C4)

Notes:

1. When the client replies (CLC Confirm) with an Emulated-ISM CHID, then the client must include an Extended GID. Part 2 of the Extended GID is included in the Client Extended GID field.
2. The FCE v2.1 feature bit mask returned by the client in the CLC Confirm message represents all common v2.1 features supported by both the client and server. The bit mask in the CLC Confirm represents a summary of common features passed by the client in the CLC Proposal and passed by the server in the CLC Accept message. The common features are the v2.1 supplemental features that are eligible to be used by this connection.

CLC CONFIRM MESSAGE (SMC-RV2.1 FORMAT)

Offset	Length	Description
0	4	Eye catcher ‘SMCR’ (EBCDIC) message start (constant xE2D4C3D9)
4	1	CLC Message Type 03 (CLC Confirm)
5	2	CLC Confirm Message (SMC-R) Length 108 bytes (without FCE)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0010’ Version 2) Bit 4 b‘0’ SMCv2 Contact (b’1’ = first contact, b’0’ subsequent contact) - applicable to V2 only Bit 5 b’0’ Reserved (zero) Bit 6-7 b’00’ SMCv2 Type (b’00’=SMCRv2) - commentary only
8	8	SMCv2 Sender (client) Peer ID
16	16	SMCRv2 Client GID (IPv4 or IPv6 address, IP protocol must match TCP connection)
32	6	Client MAC Address (See MAC address note.)
38	3	SMC-Rv2 Client RoCEv2 QP Number
41	**9**	**Client RMB Information**
41	4	Rkey
45	1	RMBE (connection) Index
46	4	RMBE Alert Token

50	1	Flag 2 B’0000’ RMBE Size B’0000’ MTU Size
51	1	Reserved
52	8	RMB PCI Virtual Address
60	1	Reserved
61	3	PSN
64	32	EID (Negotiated EID confirmed by client)
96	8	Reserved
104	36	First Contact Extension - FCE is only present when first contact flag is on
104	4	FCE Header
104	1	FCE Header Flag 0 (Bit 8) Bit 0 V2 LG Type b‘1‘ = v2_Direct, b‘0‘ = v2_Indirect Bit 1-7 Reserved (zero)
105	1	FCE Header Flag 1 (Bit 8) Bit 0-3 OS Type (b’0001’ = z/OS, b’0010’ = Linux Z, b’0011’ = AIX, b’1111 = unknown) Bit 4-7 b‘xxxx’ SMC Version Release Number Same format as Version number b’0000’ Release 0 – v2.0 (Release 0) b’0001’ Release 1 – v2.1 (Release 1 – Feature Update)
106	2	FCE Header - reserved
108	32	FCE Peer Host Name (ASCII character - padded with ASCII blanks)
140	1	FCE Max Connections per LG (v2.1)

141	1	FCE Max Links per LG (v2.1)
142	2	FCE v2.1 Feature bit mask (bit 16) - client and server common features
144	4	FCE Vendor Unique Experimental Options Area (v2.1)
148	8	FCE - Reserved for growth
156	*	FCE Client RoCEv2 GID List - Variable Length Area, Client GID List is a required field - always present. For format and specs, see the Client RoCEv2 GID List definition.
*	4	Eye catcher 'SMCR' (EBCDIC) message end (offset is variable based on length of the variable length Client RoCEv2 GID List and a possible FCE) (constant xE2D4C3D9)

Notes:

1. The MAC address is defined for the SMC-Rv2 CLC Confirm. However, the purpose of the MAC address in the CLC Confirm is based on link type. When the:
    a. Link type is **direct**. The MAC address is used (must match the MAC from the original Proposal) by the server to confirm the v2 QP for the next hop address back to the client.
    b. Link type is **indirect**. The MAC address becomes informational (e.g., possibly used for diagnostics or administrative purposes).
2. All IP addresses (GIDs) defined within all SMC messages are defined as 16 bytes. An IPv4 address can be represented in two formats:
    a. IPv4 address is right aligned (leftmost 3 words of zeros) or
    b. IPv4 mapped IPv6 address format (leftmost 80 bits of zeros + xFFFF + IPv4 address) See RFC 4291.
3. V2.1 Max Connections must match the CLC Accept value.
4. V2.1 Max Links must match the CLC Accept value.
5. V2.1 feature bit mask, features common to both client and server.
   The FCE v2.1 feature bit mask returned by the client in the CLC Confirm message represents all common v2.1 features supported by both the client and server. The bit mask in the CLC Confirm represents a summary of common features passed by the client in the CLC Proposal and passed by the server in the CLC Accept message. The common features are the v2.1 supplemental features that are eligible to be used by this connection.
6. V2.1 the vendor unique options must match the CLC Accept values.

CLIENT ROCEV2 GID LIST

0	*	Client RoCEv2 GID List Describes the client’s RoCEv2 IP Address Configuration
0	4	GID List Header
0	1	GID List No of Entries (1 - 8)
1	3	Reserved
4	*	GID List Array Area – Variable Length Size is based on GID List No of Entries (contains 1 - 8 16-byte IP address entries)
4	16	GID List Entry - RoCEv2 IP address (IPv4 or IPv6) Entries are order sensitive. See notes 1 and 2.
*	*	End of Client GID List

Notes:

1. The GID List is a required field and is always included within the following two messages sent from the v2 client:
    a. CLC Confirm message (within the FCE) and
    b. LLC Request Add Link message (request only, not included in the response).
2. The purpose of the GID List is for the client to provide the server with the client’s entire eligible RoCEv2 IP configuration (i.e., alternate RoCEv2 IP addresses). Eligible means the RoCEv2 interface (with this RoCEv2 IP address) is available, and the client has an available IP route to the server’s FC TCP/IP address over the associated IP interfaces. It is unknown to the client (until ADD Link is received by the client) if the client will also have an IP route to the server’s selected RoCEv2 IP address. The server will use this information to build the ADD Link to create an alternate link.
3. The GID List is a variable-length array. The length is defined by the GID List No of Entries. The value of the GID List No. of Entries must be:
    a. In the range 1 - 8 (inclusive). Any other value is an error.
    b. Consistent with the overall message length (e.g., if the value of GID List No. of Entries is 6 but the outer (including) message is not long enough to contain 6 entries, then this is an error.
4. The GID List length has a:
    a. Minimum length of 20 bytes (header + 1 IP Address, the active link only)
    b. Typical length of 36 bytes (header + 2 IP addresses, 1 active + 1 alternate link) and
    c. Maximum length of 132 (header + 8 IP addresses)

5. A v2 client can elect to support a lower maximum no. of alternate RoCEv2 IP addresses.
   Note: z/OS will support sending up to 8 RoCEv2 IP addresses (1 active and a maximum of 7 alternates).
6. The RoCEv2 IP addresses in the GID list are order sensitive defined as follows:
   a. The first IP address entry is always present and represents the RoCEv2 IP address of the current active link (can be the FC link).
   b. The remaining (if any) IP address entries in the list represent the client's eligible alternate RoCEv2 IP addresses. The addresses are listed in the client's preferred order for usage as an alternate link (to be selected by the server). The client's preference is based on the links that provide the highest level of HA from the client's perspective or configuration (relative to the current active link). When possible, the server should select the first IP address in the list. When this is not possible, the server should select the next IP address in the list (in preferred order). The server can select any IP address from the Client list when sending ADD LINK.
      Notes:
      - The server will not select a Client IP Address if it is not reachable, i.e., the server does not have an IP Route to Client's RoCE IP address.
      - The server will select the local RNIC based on forming the highest level of HA from the server's perspective. This selection can result in selecting a GID lower in preference from the client's perspective.
7. The server should cache (within the LG) the list of available client RoCEv2 IP addresses and refresh when a new Request Add Link is received. .
8. All IP addresses (GIDs) defined within all SMC messages (CLC or LLC) are defined as 16 bytes. An IPv4 address can be represented in two formats:
   a. IPv4 address is right aligned (leftmost 3 words of zeros) or
   b. IPv4 mapped IPv6 address format (leftmost 80 bits of zeros + xFFFF + IPv4 address). See RFC 4291.

CLC V2 DECLINE MESSAGE (SMC-DV2 AND SMC-RV2 FORMAT)

Offset	Length	Description
0	4	Eye catcher ‘SMCR’ (EBCDIC) message start [1]
4	1	CLC Message Type 04 (CLC Decline)
5	2	CLC Message Length (44)
7	1	Flag 1 (bit 8) Bit 0-3 b‘0000’ SMC Version (b’0001’ Version 1 or b’0010’ Version 2) Bit 4 b‘1’ Out of Sync Bit 5-7 b’0’ Reserved (zero)
8	8	Sender Peer ID
16	4	Sender Diagnosis Information
20	1	Flag 2 (bit 8) Bit 0-3 OS Type (b’0001’ = z/OS, b’0010’ = Linux, b’0011’ = AIX, b’1111’ = unknown) Bit 4-7 b‘0000’ Reserved
21	3	Reserved
24	16	SMC Type Diagnosis Information[3]
24	4	SMC-Dv2 Reason Code
28	4	SMC-Dv1 Reason Code
32	4	SMC-Rv2 Reason Code
36	4	SMC-Rv1 Reason Code

40	4	Eye catcher ‘SMCR’ (EBCDIC) message end [1]

The CLC Decline Message format is changed by Version 2 to add a v2 flag and add additional Reason Codes (see notes) which increases the length to 44 bytes.

Notes:

1. It is acceptable to use SMCR or SMCD as the CLC Decline message eye catcher.
2. SMCv2 defines a 44-byte Decline message that includes an SMC Type diagnosis field that describes all four possible reason codes for all four SMC types.
3. When the Decline is sent by the server in response to a CLC Proposal message (i.e., the typical reason for a Decline) then:
    a. The SMC Types reason code fields **should** be set for all applicable SMC types (all SMC types offered by the client). When any of the Type fields are set to zero, it indicates that the type for this connection was N/A (not offered).

       Note: If a v2 Decline is sent without including the SMC Type Diagnosis fields (shorter Decline message) or included but zero, this condition must be tolerated (i.e., some reason the additional information was not provided).

    b. The Sender Diagnosis Information field **must be** provided and should represent the last Reason Code set that was evaluated for this connection that prevented SMC from being used. It should also match one of the four SMC type RCs.
4. In all other cases in which a Decline is sent, such as a “late” Decline sent by the client, then there is only one applicable Reason Code and possibly only one SMC type. For these other Decline cases, only one reason code is set in the Sender Diagnosis Information field and the SMC type RC fields are N/A (set to zero).
5. CLC Decline Reason Codes are not defined by the SMC specifications. The RCs are defined by the OS implementation. Adding an OS type will aid in diagnosis (OS-specific RCs).
   New Decline reason codes will be necessary for V2 unique conditions. For the interpretation of the reason codes, refer to the specific source OS documentation.

LLC V2 MESSAGES (SMC-RV2)

This section defines the updates for the SMCRv2 LLC messages.

A summary of the key SMC-Rv2 LLC changes is provided here:

1. The LLC SMC-R version field did not exist in V1. For V2, the Version is now defined within the LLC Type field. All V2 LLC messages will be defined as shown below.
2. SMCv1 limited the LLC message length to 44 bytes and used LLC continuation messages as necessary. V2 supports large LLC messages (up to 8k maximum) and eliminates LLC continuation messages.
3. V2 introduces the concept of direct (same subnet) vs indirect (multiple IP subnets) Links that is reflected in the LLC signaling.
4. LLC Request Add Link replaces Add Link from the client.
5. The client communicates all available RoCEv2 GIDs in the Client GID List to the server. The Client GID list flows on CLC Confirm and LLC Request Add Link.

Starting with SMCRv2, the LLC Type field has the following description and format:

Offset	Length	Description
00	01	LLC SMC-R Version and Type Field
00	01	Bit 4 (bits 0-3) b’1xxx xxxx’ - LLC Type Code Unsupported (bit 0 position) b’x000’ - LLC SMC-R Version (bits 1-3) (b’010’ = version 2) (b’011’ = version 3) (b’100’ = version 4) etc. up to b’111’ = V7 Bit 4 (bits 4-7) b’xxxx 0000’ LLC Type Codes (see Type Codes below)

SMC-RV2 LLC MESSAGE TYPE DEFINITIONS

LLC Message Types	V1 LLC Message Types	V2 LLC Message Types	V2 Response Required Dependent versus Independent
Confirm Link	x01	x21	Dependent
Add Link	x02	x22	Dependent (large req & rsp)
Add Link Continuation	x03	N/A	N/A
Delete Link	x04	x24	From server = Dependent From client = Independent
Request Add Link	N/A	x25	Dependent (only sent by client) Note 1
Confirm Rkey	x06	x26	Dependent
Test Link	x07	x27	Independent
Cnf RKey Continuation	x08	N/A	N/A
Delete Rkey	x09	x29	Dependent

Notes:

1. Request Add Link is a dependent message sent only from the client. The server is not required to send Add Link. The server must respond to Request Add Link to terminate the dependent message flow by sending an LLC response, an Add Link request or sending a response followed by the Add Link Request. The client must accept either (or the first) message as the acknowledgment of the dependent msg flow initiated by the client.
2. The LLC message dependent versus independent requirement is a v1 definition that does not change for v2.

3. All LLC messages are either exactly (padded to) 44 bytes (which can be dependent or independent) or variable length greater than 44 bytes. If an LLC message is greater than 44 bytes (up to 8192 for the v2 Large LLC message support), it must be a dependent message (requires a response). There are no continuation messages in v2.

SMC-RV2 LLC CONFIRM LINK MESSAGE (SMCR-V2.1 FORMAT)

Offset	Length	Description
0	1	LLC Type Field (x21 = Confirm Link)
1	2	Length (fixed length 44 bytes)
3	1	Flags (bit 8) Bit 0 b'0' = Request, b'1' = Reply Bit 1-3 b'0' Reserved (zero) Bit 4-7 b'0000' System Cache Size (Aligned RDMA-w)
4	6	MAC Address See MAC address note.
10	16	RoCEv2 GID (IPv4 or IPv6 address)
26	3	QP number
29	1	Link number
30	4	Link ID
34	1	Max Links per LG – v2.1 see notes
35	1	Max Connections per LG – v2.1 see notes
35	9	Reserved

Notes:

1. When RoCEv2 GID is IPv4, the IP address is right aligned in the last four bytes.
2. All IP addresses (GIDs) defined within all SMC messages are defined as 16 bytes. An IPv4 address can be represented in two formats:
   a. IPv4 address is right aligned (leftmost 3 words of zeros) or
   b. IPv4 mapped IPv6 address format (leftmost 80 bits of zeros + xFFFF + IPv4 address) See RFC 4291.
3. The MAC address is defined for the SMC-Rv2 LLC Confirm Link. The purpose of the MAC address in the LLC Confirm Link is based on link type.
   When the:
   a. Link type is **direct**. The MAC address is used by the receiver to confirm the v2 QP that was previously created for the newly created link has the appropriate next hop MAC.
   b. Link type is **indirect**. The MAC address becomes informational (e.g., possibly used for diagnostics or administrative purposes).

4. Max Links and Max Connections Usage notes:
   The LLC Confirm Link message v2.1 fields are defined as v2.1 format when the first contact processing that establishes the LG indicates v2.1. With v2.1, maximum links and maximum connections per LG values are exchanged. The values are determined and potentially negotiated in the CLC messages exchanged during first contact processing passed in the CLC First Contact Extension (FCE). The CLC Accept message from the server determines and sets the final values. v2.1 hosts must pass the same final values (from the CLC exchange) in all associated LLC Confirm Link (req/rsp) messages. The values exchanged here serve as a confirmation of the values being used for the life of the LG. The values are not negotiated in the LLC exchange. If the values do not match the CLC Accept FCE, then this is a protocol violation.

SMC-RV2 LLC ADD LINK MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x22 = Add Link)
1	2	Length - Variable Length
3	1	Flags (bit 8) Bit 0 b’0’ = Request, b’1’ = Reply Bit 1 Response Type, b’0’ = Positive, b’1’ = Negative Bit 2-3 b’0’ Reserved (zero) Bit 4-7 b’nnnn’ Reject Reason Code (when negative response)
4	6	MAC Address See MAC address note.
10	2	Reserved
12	16	RoCEv2 GID (IPv4 or IPv6 address)
28	3	QP number
31	1	Link number
32	1	Flag 2 Bit 0-3 Reserved Bit 4-7 QP Enumerated MTU
33	3	Initial PSN
36	8	Reserved

44	1	V2 Flags Bit 0 - link type - only valid in response from client: o b’0’ - indirect: link attachment to peer is indirect o b’1’ - direct: link attachment to peer is direct Bits 1-7 available
45	1	Reserved
46	16	Client Target GID (Client Alternate RoCEv2 IP Address provided by the client and preferred by the server) - Can be zero. See note 7.
62	8	Reserved
70	2	No of RKeys (RMB Token Pair Entries) in RKey Array (maximum entries = 255)
72	*	RKey Array Area (variable size based on no of RKeys)
72	16	RMB RToken Entry Format
72	16	RMB RToken Pair #1
72	4	Rkey (existing) on this link
76	4	Rkey (new equivalent Rkey) on new link
80	8	Virtual address for new link
88	16	RMB RToken Pair #2 (when present)
104	16	RMB RToken Pair #3 (when present)
120	16	RMB RToken Pair #4 (when present), etc.
*	0	End of Message (message ends with last RToken entry)

Notes:

1. When RoCEv2 GID is IPv4, the IP address is right aligned in the last four bytes.
2. The format of RMB RToken Entry (RMB RToken Pair) is identical to the format of the RMB RTOKEN fields defined in the SMCRv1 ADD Link Continuation Message
3. The SMCR protocol limits the number of RMBs per LG to 255. This makes the maximum length of the V2 ADD Link Message = 4128 (48 (base) + (16 x 255)).
4. The v2 Add Link Message is from the server only and is a dependent LLC (message). The Request Add link is from the client only and it is also a dependent message. Refer to the Request Add Link message for details (next section).
5. The MAC address is defined for the SMC-Rv2 LLC Add Link. However, the purpose of the MAC address in the Add Link message is based on link type. When the:
    a. Link type is **direct**. The MAC address is used by the receiver to create the v2 QP next hop MAC address to create connectivity back to the sender (i.e., when exchanging the Add Link request and response).
    b. Link type is **indirect**. The MAC address becomes informational (e.g., possibly used for diagnostics or administrative purposes).
6. The v2 LG contains links that can be either direct or indirect. The MAC addresses are only used for direct links, but MAC addresses are always required in the LLC messages. It is considered a protocol violation when the sender does not include the MAC address field.
7. The Client Target GID is a RoCEv2 IP address that was previously provided by the client in the CLC Confirm or LLC Req Add Link messages as a preferred eligible alternate link RoCEv2 IP address to be used as a potential future Add Link (this message). When possible or available, the client should select this RoCEv2 IP address to complete the add link process (creating the new link). When the target GID is no longer available, the client has the option of selecting any eligible RoCEv2 IP address. It is also possible for this field to be zero (Client did not previously provide the server with an alternate GID). This field is only defined in the Add Link Request. The contents of this field in the Add Link Response are ignored by the server.

SMC-RV2 LLC REQUEST ADD LINK MESSAGE (FROM CLIENT)

Offset	Length	Description
0	1	LLC Type Field (x25 = Request Add Link)
1	2	Length - Variable Length (can be 44 or greater than 44 bytes) (always dependent).
3	1	Flags (bit 8) Bit 0 b’0’ = Request, b’1’ = Reply Bit 1 Response Type, b’0’ = Positive, b’1’ = Negative Bit 2-3 b’0’ Reserved (zero) Bit 4-7 b’nnnn’ Reject Reason Code (when negative response)
4	20	Reserved for growth
24	*	Client RoCEv2 GID List - Variable Length Area included only in the Request (the GID List is not included in the response from the server). See format of Client RoCEv2 GID List.
*	*	End of Request Add Link Message

Notes:

1. This message is new for v2 replacing the Add Link request from the client. In v2, the client never sends LLC Add Link (not defined for v2). The stimulus to send this message remains the same (when a new RNIC becomes available to the client allowing the server to convert the LG to symmetric).
2. The Request Add Link message is a dependent message and requires a response (acknowledgment) from the server. See note 7 below regarding potential LLC collision and response handling.
3. **Request Add Link** flows only from the v2 client! It is sent by the client when the LG is not currently symmetrical, and the client has a new RoCEv2 IP address (RNIC) for the server to use for a subsequent ADD Link. This message provides a list of eligible GIDs for alternate links. This is a notification message indicating the client’s RoCEv2 configuration has changed, for example, a new RNIC might have become available indicating which IP addresses can be used if / when the server sends an Add Link. The client should not send this message when the LG is symmetric.
4. This is a variable length request message that has the following possible lengths:
    a. A minimum length of 44 bytes (Client GID list with 1 entry), this length is defined (permitted) but does not provide any alternative GIDs and
    b. The **expected or typical length of 60 bytes** (Client GID list with 2 entries) and

   c. A variable / maximum length of up to 156 bytes (Client GID list with greater than 2 GID entries and up to 8 entries)

5. The **Request Add Link response does not include the GID list and is always a small (fixed length 44 bytes) LLC response message.** If the server can't send an Add Link, the server should not reject the Request Add Link (i.e., send a negative response e.g., the server does not have an IP route to the new client GID). The response just acknowledges the receipt of a valid LLC message.
6. The Request Add Link request is a dependent message that requires a response from the server in one of the following forms (all variations satisfy the outstanding dependent message from the client):
   a. An LLC (44 bytes) response (the server is not required to send Add Link) or
   b. An LLC Add Link Request or
   c. Both (Request Add Link response, followed by an Add Link Request)
7. The Request Add Link message could also pass the server's already inflight Add Link message. Dependent LLC messages originating from each peer host can cross (can still collide). When this occurs, the client should consider the new Request to act as the response and terminate the dependent LLC message flow (i.e., the server is not required to respond to Request Add Link when Add Link is already in progress).
8. The Client RoCEv2 GID list is defined separately. Reference the Client RoCEv2 GID List definition for format information. The Client RoCEv2 GID List is defined to be included within the following two messages:
   a. CLC Confirm Message.
   b. LLC Req Add Link Message

SMC-RV2 LLC DELETE LINK MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x24 = Delete Link)
1	2	Length (decimal 44)
3	1	Flags (bit 8) Bit 0 b’0’ = Request, b’1’ = Reply Bit 1 b’0 = single link, b’1’ = all links in the LG Bit 2 b’0’ = disorderly, b’1’ orderly Bit 3-7 b’0’ Reserved (zero)
4	1	Link number
5	4	Reason Code
9	35	Reserved

Note：

LLC messages shorter than 44 bytes are padded with zeros.

SMC-RV2 LLC CONFIRM RKEY MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x26 = Confirm RKey)
1	2	Length (fixed length 44)
3	1	Flags (bit 8) Bit 0 b'0' = Request, b'1' = Reply Bit 1 b'0' Reserved (zero) Bit 2 b'0' Reply type (positive), b'1' negative reply Bit 3 b'0' Not retry, b'1' Retry RKey set Bits 4-7 b'nnnn 0000' Reserved (zero)
4	1	No of other RToken pairs to be communicated
5	4	New RMB RKey
9	8	New RMB Virtual Address
17	13	Other RMB RToken Pair
30	13	Other RMB RToken Pair
43	1	Reserved
44	0	End of Message

Notes:

1. The CRK defines a single new Rkey along with a single Other RMB RToken Pair (for the alternate link) supporting LGs with 2 links (the current max no. of links supported).
2. The format of the Other RMB RToken pair is defined as follows:

Offset	Length	Description
0	1	Link Number
1	4	Rkey for this link
5	8	RMB Virtual Address for this link

3. A second “Other RMB Rtoken” field is defined at + 30 ending at + 43 with the remaining single byte defined as reserved (padding to 44 for a small LLC message). This second Other RMB Rtoken would accommodate a third link (if 3 links were ever supported). If in the future LGs will support greater than 4 links, this message will need to grow becoming a large LLC message.
4. The architectural maximum length of the V2 CRK Message = 44 bytes (small LLC message).
5. If the need or requirement arises in the future to send multiple new Rkeys with a single CRK, with each new RKey having alternate RTokens, the format of the message would need to change along with the error handling.

SMC-RV2 LLC DELETE RKEY REQUEST MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x29 = Delete RKey)
1	2	Variable Length
3	1	Flags (bit 8) Bit 0 b'0' = Request, b'1' = Reply Bit 1 b'0' Reserved (zero) Bit 2 b'0' Reply type (positive), b'1' negative reply Bits 3-7 b'nnn0 0000' Reserved (zero)
4	1	Count - No of RKeys other RToken deleted on this command
5	1	Reserved
6	2	Reserved
8	4	1st RKey to be deleted
12	4	2nd RKey to be deleted
16	4	3rd RKey to be deleted
20	4	4th RKey to be deleted
24	4	5th RKey to be deleted
28	4	6th RKey to be deleted
32	4	7th RKey to be deleted
36	4	8th RKey to be deleted
40	4	9th Rkey to be deleted

44	4	10th Rkey to be deleted
48	4	11th Rkey to be deleted etc. (up to 255 Rkeys)
*	0	Message ends with last Rkey (or padded with zeros to 44 bytes)

Notes:

1. The Delete Rkey Request and Response have different formats. See the response format in the next section.
2. The SMCRv2 maximum no of Rkeys is 255. With 255 x 4-byte keys makes the V2 maximum message length 1028 bytes (8 bytes (base) + 255 Rkeys)
3. When less than 8 RKeys are included in this message, the message is padded with zeros to 44 bytes.

SMC-RV2 LLC DELETE RKEY RESPONSE MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x29 = Delete RKey)
1	2	Length - fixed 44 bytes
3	1	Flags (bit 8) Bit 0 b'0' = Request, b'1' = Reply Bit 1 b'0' Reserved (zero) Bit 2 b'0' Reply type (positive), b'1' negative reply Bits 3-7 b'nnn0 0000' Reserved (zero)
4	1	Count - No of RKeys other RToken deleted on this command
5	1	Invalid Rkey Count - No of Rkeys that were not deleted
6	2	Reserved
8	4	1st RKey not deleted
12	4	2nd RKey not deleted
16	4	3rd RKey not deleted
20	4	4th RKey not deleted
24	4	5th RKey not deleted
28	4	6th RKey not deleted
32	4	7th RKey not deleted
36	4	8th RKey not deleted etc. (up to 8 invalid RKeys)
40	4	Reserved (4 bytes of Pad to 44-byte response)

44	0	Message end (fixed-length negative response)

Notes:

1. Delete Rkey response has a fixed length of 44 bytes.
2. Rkeys that were successfully deleted are not included in the (positive or negative) response.
3. The count field in the response matches or indicates the original count in the request.
4. The invalid count could be zero. When nonzero, it indicates how many RKeys that were not deleted (invalid). This count could be greater than the no of included invalid RKeys (which is a max of 8 RKeys to be included). In a positive response, the invalid RKey count is zero.
5. Only the negative Delete RKey response contains RKeys (up to 8) that were not deleted.

SMC-RV2 LLC TEST LINK MESSAGE

Offset	Length	Description
0	1	LLC Type Field (x27 = Test Link)
1	2	Length (decimal 44)
3	1	Flags (bit 8) Bit 0 b’0’ = Request, b’1’ = Reply Bits 1-7 b’n000 0000’ Reserved (zero)
4	16	User Data
20	24	Reserved

GLOSSARY

The SMCv2 glossary provides a list of some SMCv1 base terms along with some new terms introduced with SMCv2.

DMB

The ***Direct Memory Buffer*** (DMB) is the memory buffer that is shared among peer hosts. The socket sending host will write into the peer's DMB and the target host will consume the data during application socket rcv() processing. The **DMB** (Direct Memory Buffer) is virtual memory that is managed by the software stack and registered with the PCI IOAT services and the ISM adapter. The adapter registration will generate a DMB token that is passed to a single remote peer (by software during CLC exchange). The peer host uses the DMB token to store data via the SMC-D protocol for TCP socket applications. The DMB is similar (in purpose) to the RMB (used for SMC-R).

DMB Owner

The **DMB Owner** is the host (LP) which allocates virtual memory and registers the DMB with the ISM adapter. The owner shares access to the DMB with the peer host by passing the DMB token to the peer. This peer host is considered the DMB user.

DMB Token

The **DMB token** is a 64-bit token assigned to a DMB when it is registered by the owning OS to the ISM adapter (function). When the DMB is registered, a peer host also must be identified (by its ISM-GID). The token is opaque to software. The peer host is the only host that can use this token for remote access. The DMB token is similar to the RMB Rkey.

DMB User

The **DMB User** is the host (LP) which receives a DMB token from the DMB owner. The DMB user can remotely access (store into) the DMB using the DMB token and PCI ST instructions. Each DMB can only have a single DMB user.

DMBE

The **DMBE** (Local Memory Buffer Element) is created when the DMB owner logically divides a DMB into logical elements. Each element is assigned to a specific TCP connection.

DMBE Index

The **DMBE index** uses zero based indexing, describing the index (offset) into the DMB. For example, for a given DMBE (i.e., index 0 = DMBE 1, index 1 = DMBE 2, etc.). A DMB can (optionally) be subdivided into multiple elements. If the OS does not subdivide a DMB into multiple elements, the DMB consist of a single element (DMBE). The elements must be of equal length. The DMBE index can also be used to derive the SBA DMBE bit mask (i.e., index 0 = bitmask 0000, index 1 = 0001, index 2 = 0010 etc.)

EID

SMC Version 2 Enterprise ID (EID) is an Enterprise Identifier (EID) that represents a group of systems that are allowed to communicate within the “Enterprise” using SMC over multiple IP subnets using the SMCv2 protocol. EIDs are fixed length 32 IDs defined in character format. The ID can either be a user-defined ID (UEID) or system-defined ID (SEID). The characters are alphanumeric. The user-defined EIDs are also referred to as UEIDs. IBM provides guidelines for creating globally unique EIDs. Also, see System EID (SEID).

Emulated-ISM Device

An **Emulated-ISM** device is provided by the supplemental Emulated-ISM Device feature defined by SMC-Dv2.1 and ISMv2.1. The Emulated-ISM device conforms to the SMC-Dv2.1 protocol but provides the Emulated-ISM functionality within software (Operating System or Hypervisor) vs. platform firmware / hardware. Emulated-ISM devices use an Emulated-ISM CHID and an Extended GID format.

Emulated-ISM CHID

An Emulated-ISM device is associated with an **Emulated-ISM CHID** created and managed by software and serves as the internal fabric. Emulated-ISM CHIDs are defined with a reserved range of values of xFF00-xFFFE and xFFFF used for loopback.

Extended GID

An Emulated-ISM device uses a longer 128 bit (vs. 64 bit) GID (Global ID) that is defined as an **Extended GID.** Extended GIDs are created and managed by software using the UUID format as defined by the IETF in RFC 4122. Version 4 UUIDs are recommended.

ISM

Internal-Shared-Memory is a term that defines a set of IBM Z functions associated with supporting the vPCI ISM architecture. Also, see ISM function and ISM network.

ISMv1

ISMv1 is Internal Shared Memory (ISM) that is specific to the original ISM function that is restricted to layer 2 connectivity (single IP subnet) within an IBM Z CPC.

ISMv2

ISMv2 is Internal Shared Memory (ISM) that is specific to the second version of the ISM function that is not restricted to layer 2 connectivity. ISMv2 allows connectivity when the related TCP/IP connection is over multiple IP subnets. The ISMv2 connectivity provides internal layer 3 connectivity (multiple IP subnets) within an IBM Z CPC. ISMv2 does not use IP packets or IP routing.

ISMv2.1

ISMv2.1 is a version of Internal Shared Memory that is based on SMCv2 that defines supplemental features of SMCv2 and ISMv2 associated with SMC-Dv2.1. The first ISMv2.1 supplemental feature is the Emulated-ISM Device feature.

ISM CHID

CHID (Channel ID) applies to the IBM Z architecture to describe an I/O “channel” in the system I/O configuration. Typically, a channel provides a path to a physical channel (**PCHID**) or the devices accessible through this channel. Since ISM is a virtual (firmware) device, the CHID is defined as and considered to be a virtual CHID (**ISM VCHID**). The ISM VCHID provides similar functionality as a PCHID. In most cases, both the CHID and VCHID terms can be used interchangeably. In addition to a virtual CHID, each ISM CHID also logically represents a unique internal network ("ISM fabric").

ISM device

The term **Internal-Shared-Memory** applies to the vPCI function that is created in support of SMC-D. ISM is also used to denote the VCHID or CHID type and internal network type. ISM devices are represented as PCI FIDs.

ISM Function

The **ISM function** is the IBM Z vPCI function created in support of the ISM adapter created within IBM Z firmware in support of the SMC-D protocol.

ISM FID

The **ISM Function ID (FID)** is the System zPCI Function ID created by the HCD/IOCDS definition that uniquely identifies a single instance of an ISM function within an ISM VCHID. In some cases, the ISM device and FID mean the same thing.

ISM GID

The **ISM-GID (Internal Shared Memory Global ID)** is the Identifier generated by system firmware when the user enables the ISM function. The GID is based on the CPC and must be unique among all IBM Z processors. The ID represents (identifies) an instance of an SMC-D user (Virtual Function) within the ISM network (VCHID).

ISM Network

The **Internal Shared Memory Network** is represented by the ISM VCHID. Peer hosts that have access to the same ISM VCHID can communicate over this network.

ISM Virtual Function

The **ISM Virtual Function** (VF) is the defined VF value (defined in HCD/IOCDS for zPCI functions) that enables sharing capability (virtualization) of the physical resources associated with an ISM adapter.

ISM VCHID

The **ISM VCHID** (virtual CHID) is the HCD/IOCDS definition of the Channel ID (CHID) for ISM. Also, see ISM CHID.

Associated ISMv2 GID

An associated **ISMv2 GID** represents an ISMv2 device (FID) that has an associated IP device. The association is made by ISM being defined with a matching PNetID (matching with OSA or HiperSockets). Associated ISM VCHIDs can also inherit the L2 network attributes of the associated IP device. Associated ISMv2 devices support both SMCv1 (L2) and SMCv2 (L3) connectivity. Also, see unassociated ISMv2 GID.

Unassociated ISMv2 GID

An **unassociated ISMv2 GID** represents an ISMv2 device (FID) that does not have a system defined PNetID and therefore does not have an associated IP device or an associated Layer 2 network. The PNetID for an unassociated ISM GID is N/A. Unassociated ISMv2 devices only support SMCv2 (L3) connectivity. Also, see associated ISMv2 GID.

LP

Within this document, the term or acronym **"LP"** (Logical Partition) refers to a IBM Z Logical Partition executing on PR/SM (as an individual LPAR). Specific references to other types of guest virtualization and their underlying hypervisors (i.e., z/VM, kVM, and guest virtual machines) will be mentioned when necessary. Also, see the term "configuration".

Physical Network ID

The **Physical Network Identifier** or **PNetID** is a user-defined (HCD) identifier (up to 16 bytes) associated with a NIC (OSA, HiperSockets, or RoCE) port. The ID represents the ID of the physical layer 2 network (broadcast domain) that the NIC port will be connected with. The ID is used by software to associate or group NICs together. The PNetID is also supported for IQD (HiperSockets) and ISM VCHIDs. VCHIDs do not have physical ports (one logical PNetID for the entire VCHID).

SMC

Shared Memory Communications is an IBM Z communications protocol that transparently transports the messages (socket stream data) related to the socket-based Transmission Control Protocol (TCP) over a direct or shared memory access model via RDMA (SMC-R) or local (SMC-D). The SMC transport and network fabric are superimposed on top of or associated with an existing IP network.

SMCDv1

SMCDv1 represents the original SMC V1 protocol that is either specific to or related to SMC-D that is limited to TCP connections within a single IP subnet.

SMC-Dv2

SMC-Dv2 represents the SMCv2 protocol that is either specific to or related to SMC-D that supports TCP connections over multiple IP subnets. The IP subnets of the TCP hosts are N/A.

SMC-Dv2.1

SMC-Dv2.1 is based on the SMCv2 protocol providing the ability to introduce supplemental SMC-Dv2 features that require minor protocol changes. SMC-Dv2.1 defines ISMv2.1 which defines supplemental feature Emulated-ISM Device support.

SMC Rendezvous

The SMC Rendezvous defines a TCP (Experimental) Option that flows within the existing TCP/IP (TCP) 3-way handshake (SYN, SYN-ACK, ACK) and a set of SMC defined Connection Layer Control (CLC) messages (3-way handshake) that flow within the TCP connection that allows two SMC peers to rendezvous, negotiate and switch to the SMC protocol for a given TCP connection.

SMCv2

SMC Version 2 is the second version of the SMC protocol. SMCv2 applies to SMC-D. SMCv2 provides SMC connectivity for client – server configurations over multiple IP subnets. IP subnets are N/A for SMCv2. SMCv2 also introduces a Version Release number that further defines the SMC version. SMCv2 is initially defined as SMC V2.0.

SMCv2.1

SMCv2.1 is based on the SMCv2 protocol and provides the ability to introduce supplemental SMCv2 features that require minor protocol changes. SMCv2.1 defines new features for both SMC-Rv2.1 and SMC-Dv2.1.

System EID (SEID)

The **System EID** (SEID) is an internal EID that is built by the SMCv2 software stack that has a predefined - constant value representing the CPC that the OS is executing on. The SEID format is consistent with the UEID (32-character bytes) format. The SEID is only applicable to SMC-Dv2 and ISMv2.

TCP Client

The **TCP Client** is a TCP/IP socket-based application that initiates a TCP/IP connection through the TCP/IP stack using the existing (unchanged) socket Connect API call that initiates the TCP 3-way handshake (SYN, SYN-ACK, ACK exchange).

TCP Server

The **TCP Server** is a TCP/IP socket-based application that creates listening socket that accepts a TCP/IP connection through the TCP/IP stack using the existing (unchanged) socket Accept (API) call.

UEID (User EID)

See **EID.**

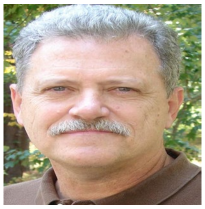
ACKNOWLEDGMENTS AND CONTRIBUTIONS

This paper was a collaborative effort by multiple IBM teams. Thanks to the following individuals for their contributions to this paper.

- Gus Kassimis, IBM Raleigh NC, US
- Stefan Raspl, IBM Boeblingen, Germany

About the Authors:

Randall Kunkel is a Senior Technical Staff Member Software Engineer in the IBM Systems Enterprise Networking Solutions development team, focusing on communications. He has over 30 years of experience developing IBM operating systems communications solutions. Randy can be reached at kunkel@us.ibm.com



Jerry Stevens is a Senior Technical Staff Member Software Engineer in the IBM Systems Enterprise Networking Solutions architecture and development team, focusing on communications architecture. He has over 35 years of experience developing IBM operating systems communications solutions.
Jerry can be reached at sjerry@us.ibm.com

Trademarks