M21 Networking

M21 Networking

Table of Contents

Chapter 1. Overview

This document describes the operation of the M21 Request Server. The M21 request server enables cross-system requests to be passed both between multiple M21 systems and between M21 systems and systems that communicate using the Distributed Data Processing (DDP) protocol. The DDP protocol enables networking between M21 and MSM, DSM or ISM systems.

Chapter 2. Introduction

A cross-system request is generated when an M program encounters a reference to a global which, after translation, is found to reside on a different system. This system may be on the same machine or on a different machine that can be accessed via the network. A cross-system request is also generated for replicated globals where the replication specifies a different system.

Most communication with the Request Server will take place using TCP/IP sockets. However communication with remote non-M21 systems using the DDP protocol requires communication at the data-link level to access the ethernet directly. To facilitate this lower level communication the Packet Capture Library ***libpcap*** is used. This is public domain software that provides a layer of abstraction hiding the different methods that different flavours of UNIX have for capturing packets at the data-link level. This library has been modified to make it more specific to the capture of DDP packets and also to enable the writing of packets at the data-link level.

M21 networking is based on request-response transactions. The local system request server will send a request for a global operation to the remote system and wait for a response. The remote M21 Request Server or non-M21 system will receive the request, perform the required operation and return a response. In DDP terminology, the system making the request is called the client, and the system processing the request and returning the response is called the server.

Chapter 3. M Jobs and The Local M Database

Each M job runs within an M environment. Part of this environment is the local M database where global data is stored. The local database must be specified when the M-job starts. In an M21 environment the local database is specified either by the **DB-NAME** environment variable or by using the **-d** command line parameter to ***mpmse***.

Chapter 4. M Databases and M Systems

The M21 Request Server deals with individual M systems.

In an M21 environment the local database equates to an M system where the system name is that given to it explicitly when the database was created using ***dbinit***. Where a reference to a global (following any translation) does not explicitly specify a system, or specifies a system with the same name as the local M database, then the local M database is accessed.

In a non-M21 environment things may be a little more complicated. Here a database might be divided into multiple volume groups depending on the version of M that the target machine is running. The M21 Request Server considers each volume group on a remote system to be equivalent to an M21 system.

All M systems are identified by a three-letter system identifier string. For the current UCI this is the second part of the value returned by **$ZU(0)**.

Chapter 5. Extended Global Syntax

M21's extended global syntax is used to access global information held in other UCIs than the one in which the M routine is executing. These UCIs may be within the same database or on a different database, either on the same machine or on another machine. The syntax used for accessing globals in another UCI or on another system is as follows:

^|UCI|Global or ^[UCI]Global

Is used to access a global in another UCI within the same database

^[UCI,SYS]Global or ^[UCI,SYS]Global

Is used to access a global in a UCI in a different database

UCI is the three-character name of the UCI.

SYS is the three-character identifier of the database which was supplied when the database was created using ***dbinit***.

The format using vertical bars conforms to the ANSI Standard environment specification. The format with square brackets is the format historically used by other version of M prior to the standardization of the ANSI environment.

Chapter 6. Translation & Replication

The M21 system includes Translation and Replication support. Translation allows the system manager to specify that reference to certain global variables will be translated to actually fetch and store those variables in a different UCI or on a different database. Whenever an M routine makes an implicit reference to one of these translated globals, the system will internally determine the real location of the global from an entry in the Translation table. Using this methodology, globals can be moved between databases and UCIs in a manner that is transparent to any application routines. Replication allows duplicate copies of globals to be maintained in other UCIs and on other databases. When a global with an entry in the Replication table is updated the system will duplicate the update in the globals specified in the Replication table. For more detail refer to the Translation and Replication documentation.

Chapter 7. Cross-System Requests

Where a reference to a global (following any translation) specifies a system that has a different system name to the local system then a cross-system request is constructed. The types of cross-system request of interest to M21 Request Servers are those where the source is a locally registered M21 system and/or the target is a locally registered M21 system.

Chapter 8. Request Server

A Request Server is an M21 process that maintains a table of both M21 and non-M21 systems that have been successfully registered for cross-system requests. The Request Server listens for cross system requests and responses to cross-system requests that come from these registered systems and will then pass the requests and responses on to the appropriate processes on the same or remote machines. Translation between M21 and DDP protocols will be carried out automatically by the Request Server where one of the source and target systems is M21 and the other is non-M21 (communicating via DDP).

The Request Server demands that all systems that are registered with it have unique names. Should it receive a duplicate system name that system will not be registered.

Chapter 9. M Network Servers

An M Network Server is an M21 process that listens on a TCP/IP port for requests from the local request server. There must be at least one M-server running for each M21 system that is started on the local machine. The M-server executes any request routed to a given M21 system in its local system and sends a response back to the request server.

Dependent on the expected number of cross-system requests that the M21 system will be expected to service, the number of M Network Server jobs running on the M21 system can be configured. As traffic increases, client requests could be delayed if there are not enough M Network Server jobs to process the cross-system requests. When multiple M Network Servers are running, the cross-system requests are distributed among the servers, allowing multiple requests to be processed in parallel. This approach will increase the throughput and can minimize the time required to respond to client requests.

The number of M Network Servers to run in a particular configuration is a matter for deterministic testing, but in low traffic situations a single server will often suffice. For more active configurations, then a maximum of one server per remote system that will make requests to the local M21 system is recommended.

Chapter 10. Configuration File Parameters

Associated with M21 networking are a series of parameters. These are specified in the external configuration file (***xxxxx.cfg***), where ***xxxxx*** is the database name. This file can be found in the directory where the database files reside and will have been created by the database initialisation program ***dbinit***. For more information on these parameters see the documentation relating to the configuration file.

CONFIGUREDDP

This parameter determines whether the M21 system will be configured to allow DDP requests. The default value for this parameter if not specified is **NO**. If the parameter is set to **YES**, then both DDP client requests (requests to non-M21 systems) and DDP server requests (requests from non-M21 systems) will be allowed for this system.

CONFIGUREXSYS

This parameter determines whether the M21 system will be configured to allow cross-system requests. The default value for this parameter if not specified is **NO**. If the parameter is set to **YES**, then both client and server requests will be allowed for this system.

DDPGROUPS

This parameter is only relevant for access to and from non-M21 systems using the DDP protocol. It is used to specify which DDP groups the M21 system will belong to for security purposes. When using the DDP protocol, systems may be organized into related groups where only those systems who are members of the same group will be able to communicate with one other. An individual system can belong to multiple groups up to a maximum of 16.

DDP groups are identified by numbers in the range 0 to 15. In order to specify groups using this parameter, each group is treated as a bit position from 0 to 15. So, in order to specify membership of groups 0, 5 and 12 specify **DDPGROUPS=4129** (because bit position 12 = 4096, bit position 5 = 32 and bit position 0 = 1, so 4096+32+1 = 4129). If this parameter is not specified, then the default is for the M21 system to be automatically assigned membership of DDP group 0 (zero).

DDPPASSWORD

This parameter is only relevant for access to and from non-M21 systems using the DDP protocol. It is used to specify the DDP password that will be supplied by the M21 system when accessing non-M21 systems. When a request is sent to establish a link to a system running a non-M21 version of M using the DDP protocol, the remote system will check the password. If they do not match, the connection will not be allowed and no communication will be permitted. This parameter allows the specification of a password, which can be between 1 and 8 characters in length. This password will be broadcast to all other non-M21 systems on the network when DDP is enabled. If these other systems use the security challenge feature of DDP then they must have the password in their password list in order to allow the connection. Any system for which the security challenge feature is not enabled will allow connection whatever the password is. M21 systems do not themselves use the security challenge feature of DDP. The default, if this parameter is not present, is for there to be no DDP password.

IPL_NetworkServers

This parameter specifies the number of M Network Server jobs to be started automatically when the system is IPL'ed, providing that the parameter **IPL_StartNetworkServers** is set to **YES**. This parameter can take a value between 1 and 16 and will default to 1 if not specified.

IPL_StartNetworkServers

This parameter specifies whether M Network Server jobs are started automatically when the system is IPL'ed. The number of M Network Server jobs to start is specified using the parameter **IPL_NetworkServers**. This parameter defaults to **NO** if not specified.

- MPMSEPORT
- MSERVERPORT
- NETWORK_CARD
- REQUESTSRVDEBUG
- RSERVERPORT
- STARTDDP
- WATCHDOGPORT

Chapter 11. Affect of CONFIGUREXSYS and CONFIGUREDDP on Request Server

CONFIGUREXSYS and **CONFIGUREDDP** are two system configuration parameters that affect the request server. These are specified in the external configuration file (***xxxxx.cfg***), where ***xxxxx*** is the database name. This file can be found in the directory where the database files reside and will have been created by the database initialisation program ***dbinit***.

A system can only send and receive cross-system messages (whether M21 or DDP) if **CONFIGUREXSYS** is set to **YES** in the system's configuration file. When an M21 system is started, if **CONFIGUREXSYS** is not set (defaults to **NO**) or is set to **NO**, then the a request server will not be started. If **CONFIGUREXSYS** is set to **YES** then M21 startup will start a Request Server process and also start the M Network Servers within the M21 environment. These M Network Servers will then register with the started request server. If there is already a Request Server running on the local system using the same communications ports, then the started Request Server will terminate immediately leaving the existing Request Server running without affecting it in any way and the M Network Servers will register with the existing Request Server.

If **CONFIGUREXSYS** is not set (defaults to **NO**) or is set to **NO** then no M Network Server jobs will be started and no cross-system request sent. Therefore the system is unknown to the request server.

When the first M-server for a system registers (i.e. **CONFIGUREXSYS** must be set to **YES**) the request server will examine the DDP flag in the registration message header. If this is set then DDP access is granted to that system, otherwise DDP access is disabled. If DDP access is disabled then the request server will not pass on any DDP requests for that system or any DDP requests from that system. The M-server determines DDP access for a given system from the value of the **CONFIGUREDDP** parameter, which must be set to **YES** in the system's configuration file to enable DDP for an M21 system.

The following table summarises the types of cross-system messages that can be sent and received according to a system's configuration parameters.

Table 11-1. Types of cross-system messages that can be sent and received according to a system's configuration parameters

	CONFIGUREDDP = YES	CONFIGUREDDP = NO
CONFIGUREXSYS = YES	**M21 and DDP**	**M21 only**
CONFIGUREXSYS = NO	**Neither**	**Neither**

Chapter 12. System Registration

All systems must be registered to get an entry in a Request Server system table. Only when the system is registered with the local Request Server can the request server accept a request from the system or pass on a request to the system.

Local M21 Systems - M-Server Registration

One or more M-servers will be created each time a system is started. Each M-server will register with the local request server. On receiving the first registration request from one of the M-servers the request server will add a new entry for that system in its system table. It then informs any remote request servers that it is in contact with that the new system is now available for cross-system requests. The M-server is then sent a message indicating successful registration. For further M-servers registering for the same system the socket details for the M-server are added to the list of local M-servers available for that system and the M-server is sent a message indicating successful registration.

There is however a limit of 16 M-servers that may register for a local system. Registration will fail for M-servers exceeding this limit.

Registration will also fail for a local or remote M-server if the system table already has a different system registered under the same name (i.e. from a different machine). In this case the request server will return a failure message and the M Network Server should close down.

When an M21 system is stopped the associated M-servers will terminate. The M-servers do not have to inform the local request server, the request server will detect the closure of the permanent socket connection between the M-server and request server. When the request server detects the closure of the final socket it will remove the system from its system table and inform each remote request server that it is in contact with that the system is no longer available for cross-system requests.

Remote M21 Systems - Remote Request Server Registration

All cross-system requests for an M21 system and the responses from those requests must be passed through the Request Server on the same machine as the system.

Currently when a Request Server starts up it broadcasts a ***Startup*** message across the local subnet. Any Request Server over the network that receives the ***Startup*** message responds with a ***Startup Response*** message listing all local systems that it has registered in its system list. In this way the new Request Server knows all the systems that are available across the network and which Request Server should be sent requests for that system.

As new local systems become available and local systems become unavailable the local Request Server broadcasts ***Update*** messages to the other Request Servers to keep them up to date with which systems are available.

When a Request Server closes down it broadcasts a ***Shutdown*** message to the other Request Servers across the network which then remove all systems from their system table that are local to the closing Request Server.

Non-M21 Systems - DDP System Registration

When a Request Server starts up it broadcasts a ***DDP Startup*** message across the local subnet. All DDP systems over the network respond with ***DDP Startup Response*** messages, listing all volume groups that are available to that DDP system. Each volume group is treated as an M21 system and has an entry placed in the Request Server's system list.

The Request Server may receive ***DDP Startup*** messages from other DDP systems listing that system's volume groups. It does not to acknowledge ***DDP Startup*** messages from other M21 Request Servers as the M21 to M21 communication is via the M21 protocol not DDP.

The Request Server will receive ***DDP Update*** and ***DDP Shutdown*** messages and maintains its system list accordingly. Again it ignores such messages originating from M21 Request Servers.

Message Type Summary

In summary the following message types are sent between M jobs, Request Servers, M Network Servers and DDP systems:

Table 12-1. System Registration

Message Type	From	To	Protocol
M21 Startup, Shutdown, Update, Startup Response	**Request Server**	**Request Server**	M21
DDP Startup, Shutdown, Update, Startup Response	**DDP system**	**Request Server**	DDP
DDP Startup, Shutdown, Update, Startup Response	**Request Server**	**DDP System**	DDP
Local System Registration	**M-server**	**Request server**	M21
Local System Registration Acknowledgement	**Request server**	**M-server**	M21

Table 12-2. Cross System Requests

Message Type	From	To	Protocol
Cross-System Request	**M21 M job**	**Request server**	M21
Cross-System Request	**Request server**	**M-server**	M21

Message Type	From	To	Protocol
Cross-System Request	**Request server**	**Request server**	M21
Cross-System Request	**Request server**	**Non-M21 M Job**	DDP
Cross-System Request	**Non-M21 M Job**	**Request Server**	DDP

Table 12-3. Cross System Responses

Message Type	From	To	Protocol
Cross-System Response	**M-server**	**Request server**	M21
Cross-System Response	**Request server**	**Request server**	M21
Cross-System Response	**Request Server**	**M job**	M21
Cross-System Response	**Non-M21 M Job**	**Request Server**	DDP
Cross-System Response	**Request Server**	**Non-M21 M Job**	DDP

Chapter 13. Starting The Request Server

The request server can be run automatically (started by ***mpctl*** on running ***m21up***) or can be started manually. For the request server to be started automatically when an M21 database is brought up the system must be configured for cross-system requests.

Configuring a System For Cross-System Requests

A system is configured for making cross-system requests by setting the configuration parameter **CONFIGUREXSYS** to **YES** in the system's configuration file (***xxxxx.cfg***) file where ***xxxxx*** is the database name. This parameter defaults to **NO** if it does not appear in the configuration file.

Note that this only affects the sending of configuration requests by jobs running in this system. As long as a request server and one or more M-servers are running in this system then cross-system requests targeted to this system can be answered.

Starting A Request Server Automatically

When a M21 system is brought up using ***m21up*** the process ***mpctl*** runs. This process starts other administrative processes. If **CONFIGUREXSYS** is set to **YES** in the system's configuration file ***mpctl*** will attempt to start a request server.

Multiple request servers can run on a single system. Each request server running will have 4 ports open to listen for communication from other processes as follows:

mpmse Port

This port is used to listen for connections from M-jobs on the same machine that wish to send cross-system requests.

Default value: **/tmp/RequestServer**

Watchdog Port

This port is used to listen for connections from other mpctl processes on the same machine. If the request server is configured to close down when not required then it is watchdog connections on this port that inform the request server that it is still required.

Default value: **/tmp/RequestServerWD**

M-server Port

This port is used to listen for connections from M-servers running in local systems. The request server can pass on cross-system requests to any local system that has one or more M-servers running.

Default Value: **4900**

R-server Port

This port s used to listen for connections from other request servers, which may be running on the same machine or remotely over the network. When two request servers are connected they may pass on cross-system requests targeted to the other request server's local systems.

Default Value: **4950**

Defaults

These ports can have their default values changed in the configuration file as shown in the following example:

```
MPMSEPORT=/tmp/Mpmse1
WATCHDOGPORT=/tmp/Watchdog1
MSERVERPORT=4901
RSERVERPORT=4951
```

Running Multiple Request Servers

The port numbers are important when multiple request servers are required on a single machine. Imagine that a request server is already running on the local machine with ports set to their default values. Should a second M21 system be started with cross-system requests configured then another request server will be automatically started. Should any of the four ports clash with those of the existing request server then the second request server will immediately close down, and only the original request server will remain.

By default then, if the ports are not set in the configuration file, only a single request server will ever run.

To successfully start a new request server when one or more request servers are already running the ports in the configuration file must be set to values different from the ports in any other running request server.

Starting A Request Server Explicitly From The Command Line

A request server may be started from the command line as follows:

mprequestsrv { options }

The request server has many options with which it can be configured. Each option is identified by a minus sign followed by a single alphabetic character (case sensitive). Each option may or may not be followed by a value. The options are listed below.

-d

This option configures the request server to listen for DDP requests. If this option is not set then the request server defaults to not processing DDP messages. If this option is set then each M21 system that is running and has the **CONFIGUREDDP** configuration parameter set to YES also acts as a DDP system with a single volume group. Both the DDP system name and its single volume group take the same three-character name as the M21 system.

-D

The request server has a debug file **/tmp/rs.**<**process-id**> which is opened when it starts. If the **-D** option is set then full debug information is sent to this file. Output of debug to this file can be toggled on and off by sending a **SIGUSR1** signal to the request server. Note that for this option to work the request server must have been built with the **RS_DEBUG** compilation flag defined. If this option is not present debugging will be turned off when the request server runs (but the debug file will still be present).

-e<ethernet address>

When DDP is being processed by the request server the ethernet is accessed directly and the ethernet address of the local host is required by the request server. Usually the request server will determine the ethernet address automatically, but on platforms where this has not yet been successfully implemented the local host ethernet address can be passed into the request server. The ethernet address must be in the format "**xx.xx.xx.xx.xx.xx**", where 'x' represents a hexadecimal digit.

-f<BPF string>

When DDP is being processed using the Berkeley Packet Filtering mechanism (IBM AIX) then the filter string can be changed from its default using this option. The default string is "**ether proto 0x8039**". The string is limited to 80 characters, and if the BPF filter mechanism cannot parse the string the request server will close down.

-g<Graceful Shutdown Period>

When the request server is shut down (for example by **kill -INT**) it waits for a given time to enable its output buffers to be written. The default time period of 10 seconds can be changed using this option. The time specified is in seconds.

-h

When run from the command line this option prints out all the options that may be set to standard error.

-H<M21 Multicast hop limit>

Can be set to between 1 and 255, but should be restricted to less than 32. Defaults to 5. See **-M** option for more details, or **pp490 Stevens, Unix Network Programming.**

-l<Local Mpmse Port>

Specifies the local port for M-jobs to connect to. The port must be a valid UNIX path, but the socket file itself needn't exist. Defaults to **/tmp/RequestServer.**

-m<M-server Port>

Specifies the port for M-servers to connect to. The value must be above 1024 (limit of registered port numbers) and below 65535. Defaults to **4900**.

-M<multicast IP address>

The multicast address specified by this option is for the multicasting of startup messages from one request server to another. The multicast IP address must be in the format "**xx.xx.xx.xx**", where 'x' represents a hexadecimal digit. The range is restricted to 224.0.0.0 to 239.255.255.255, the valid range for multicast IP addresses. The preferred range is from 239.0.0.0 to 239.255.255.255, which are all administratively scoped - that is addresses in this range can be assigned locally within an organization but are not guaranteed unique across organizational boundaries. It is therefore down to the network administrator to prevent IP messages with this address entering or leaving the organisational network. The default value is 0xefc0409a.

In order for the multicasting to work correctly the IPv4 TTL scope (usually time-to-live but here used as a measure of the message hop limit) must be se in conjunction with the multicast IP address. For the recommended multicast IP address range specified above the hop limit must be less than 32.

-n<Network Card>

Specifies the network card device name to use for DDP. This is useful for specifying a specific network card to use on systems with more than one network card.

-r<Request Server Port>

Specifies the local port for other request servers to connect to. The value must be above 1024 (limit of registered port numbers) and below 65535. Defaults to **4950**.

-w<Watchdog Port>

Specifies the local port for mpctl processes to connect to. The port must be a valid UNIX path, but the socket file itself needn't exist. Default **/tmp/RequestServerWD.**

-W<Watchdog Timeout>

Specifies the number of minutes that the request server should wait following the receipt of the last watchdog before closing down. By default watchdog messages are ignored and the system must be closed down explicitly (for example by **kill -INT**). The timeout value must be greater or equal to 2 (as watchdogs are sent once per minute).

-x<Max M-servers per M21 system>

Specifies the maximum number of M-servers that will be permitted by the request server for a single M21 system. Must be a value between 1 and 16. The default is **16**.

Multiple Request Servers On The Same Machine

Request servers running on the same machine do so in isolation. A Request Server communicates using its Rserver port and a second request server can only start up if it has a different Rserver port. Since this is the case then the two request servers cannot communicate as they will send M21 control messages (to see what other M21 request servers are out there) specifying the same Rserver port as they themselves are listening on. So although multiple request servers can run on a given machine, each request server will be a member of a different group of request servers over the network.

Chapter 14. Closing Down The Request Server

If a request server has been started with the watchdog timeout option then it will automatically close down N minutes after the last local M21 system registered with it closes down, where N is the value passed for the watchdog timeout option.

However a request server can be closed explicitly at any time by sending a **SIGINT** signal with the UNIX ***kill*** command. The request server will not disappear immediately because it will wait to ensure that its output buffers are flushed.

Once the request server has shutdown all local M21 systems that were using that request server will now be unable to send or receive cross-system requests. To continue sending and receiving cross-system requests another request server would need starting up configured for the same ports.

NOTE: If the request server fails or is killed (**kill -KILL**) then the mpmse and watchdog socket files must be manually deleted, otherwise a request server will fail to start if it uses one of these same sockets. The bind to the socket will fail as the file already exists. The request server cannot just remove the file before it tries to bind as there are potentially multiple request servers on the same machine, and the file might actually be in use by another.