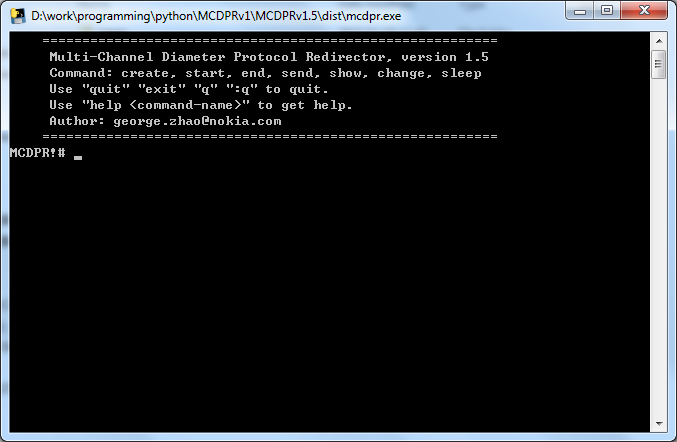
**Multi-Connection Diameter Protocol Redirector 1.5 Guide**

# MCDPR简介

MCDPR的出现背景是因为作者([george.zhao@nokia.com](mailto:george.zhao@nokia.com))参加中国移动研究院的相关测试，并且由于面对Diameter信令的复杂模拟要求而编写的。由于目前流行的很多Diameter发生器多用于性能测试，没有方便的转发功能，所以MCDPR可以作为一个很好的补充应对复杂的功能性测试，以及小量的性能测试。

MCDPR是一个可以同时支持多管道的Diameter发生器以及模拟器，同时通过不同的规则配置，还可以实现Diameter消息在多条管道之间的转发、阻止。同时MCDPR支持批处理文件，也支持了一些命令可以帮助用户在实际测试中进行简单的逻辑控制。

# MCDPR入门

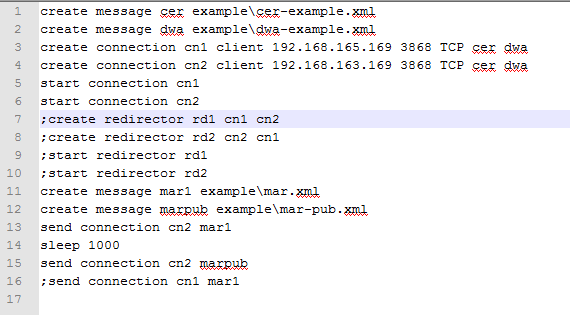
1. MCDPR包括一个可执行文件mcdpr.exe，以及一个AvpDict.xml的AVP字典文件。
2. mcdpr.exe运行后，是一个可执行命令的窗口，如下图所示，用户需要在这个窗口上执行相应的命令，来建立多条管道，Diameter消息，以及转发规则。 
3. 可以通过执行命令来创建连接，Diameter消息，转发规则等。所有的这些内容在创建时都被视为一个对象，如下图：

创建一个message，对象名为cer，使用examples\cer-example.xml的描述来建立

创建一个连接对象cn1,作为客户端连接192.168.165.169的3868端口，使用TCP协议，Diameter层使用cer对象来发送Diameter建立链接时候的CER，对象dwa用于Diameter链接建立好后，用于维持链路的DWA(Watch-Dog Answer)



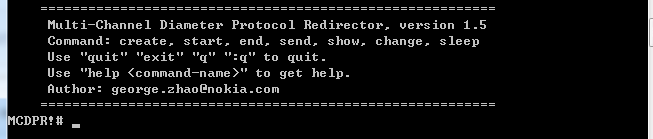
后面将要详细介绍对象建立的方法

1. MCDPR同时支持批处理文件。撰写一个文件，文件内容例如下面的形式：

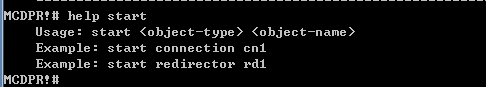
调用mcdpr.exe时加上参数即可执行该文件：



分号起头的行将被视为注释，被mcdpr忽略。如果不希望使用某行命令，但是临时不想删去他，可以使用分号来注视掉。

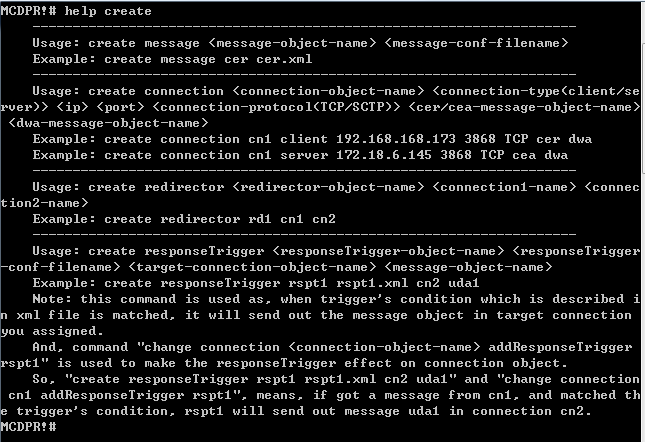
1. MCDPR有一个简略的帮助系统，在MCDPR的命令行中输入help可以得到简单的帮助信息，如下：

命令有create, start, end, send, show, change, sleep，使用help <cmd-name>可以获得某个命令的具体帮助信息。



1. 在命令行中，随时键入quit, exit, q,以及:q可以退出系统。:q是为那些使用unix/linux系统vi比较多的同事准备的。

# MCDPR命令create

1. 本章节主要解释如何在MCDPR中建立各类的对象，这些对象是MCDPR的基础，随后的发生器行为都依赖于这些对象。
2. 在MCDPR中，使用create命令来建立对象。
3. create命令可以建立的对象有message，connection，redirector，responseTrigger。
4. 输入help create可以获取create命令的帮助信息。
5. message对应一个Diameter消息。建立链接时候使用的CER，维护链接使用的DWA，以及随后希望发送的消息，都需要通过create命令，来创建一个messsage。

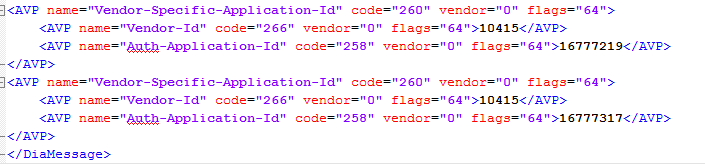
例如下面的示例1



示例1中建立了一个message对象，对象名为cer，使用文件example目录下的cer-example.xml文件。文件的位置最好使用相对于mcdpr.exe的相对路径名表示。

下面来看示例1中的文件内容，这个内容至关重要。

。。。。。。



这个范例可以在软件的example目录里面找到。这里的贴图被截断了，只是起到一个示范作用。

实际上了解Diameter的同事应该可以很快地掌握这个xml的格式。

特别注意的是，Diameter的每个AVP都应该有一个type的定义，这个type在xml文件中并没有出现。每一个AVP的Type都是在和mcdpr.exe在同一目录的AvpDict.xml中定义的。

下面来看AvpDict.xml的格式。



AvpDict的节点下定义有vendorcode，每个vendorcode下面都定义了AVP，每个AVP有自己的code，以及type。

如果你需要的AVP没有出现在AvpDict.xml中，但是你又需要在Diameter消息中使用，那么请将你自己的AVP定义在相应的vendorcode下，如果没有相应的vendorcode，也请自行添加。添加前，请备份原始的AvpDict.xml。

AvpDict.xml的type很重要，不但用于create一个message使用，而且也用来decode来自于其他节点的外来Diameter消息，并且可以和转发原则进行比较。

再回来之前的示例1中，有几个点需要特别强调。

首先是刚才提到的问题，就是在cer-example.xml中，AVP没有定义type，那是因为所有的type信息都是在AvpDict.xml中定义好的。

如，cer-example.xml中的

使用的是AvpDict.xml中的



注意AVP的名字本身没有意义，MCDPR也不用这个名字作对应，只是便于大家的记忆和分别，MCDPR在区分AVP的时候，只是使用vendorcode和avp code两个值。

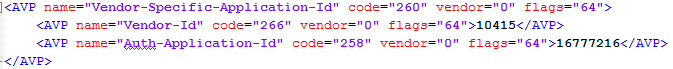
第二个要注意的是，字符串类型，即str型，这个类型在使用的时候可以直接在值得位置填写字符串，如在cer-example.xml中：



也可以使用16进制方式来表示，如在cer-example中：

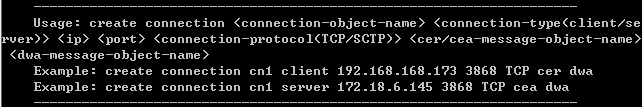


最后就是群类型的AVP，是嵌套格式，这个应该很好理解，如在cer-example.xml中：



AVP的类型包括: int, int64, str, grp

关于建立message对象，先这些内容。

1. connection是指MCDPR到某一个Diameter对端的链接，可以使用create来建立connection对象。

下面来看示例2：

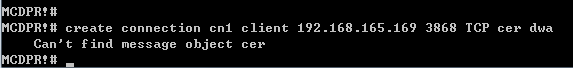


命令含义既是建立一个connection，对象名为cn1，MCDPR作为client来连接192.168.165.169这个地址的3868。建立的时候，使用cer这个对象来做CER消息，使用dwa这个对象来做DWA消息。当然cer和dwa对象都是在这条命令执行前已经建立好的。

所以你的命令应该类似于如下的情况：



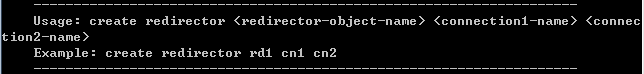
如果没有建立cer和dwa对象，那么create connection的时候，会报错：



建立好的connection对象不能直接使用，随后的start命令用于启动连接。

值得注意的是，目前MCDPR虽然支持server类型，但是对于server类型处理的不好，所以尽量回避使用，如果非使用不可，就将client位置的参数改为server即可。如果是server类型，在CER的位置上，应该填写CEA消息的对象，而不是CER。

对于Device-Watchdog，MCDPR只支持回复DWA，不支持主动发送DWR。

1. Redirector是一个可以在多个connection之间转发Diameter消息的转发器。使用create进行建立。

下面这个示例3，表示在connection cn1和cn2之间建立一个转发器。



注意，redirector是单向的，上面的rd1，只负责将来自于cn1的消息，转发给cn2。

如果需要反向的转发，则需要再建立一个redirector。

如果在建立redirector时，connection的对象不存在，则会报错：



Redirector被建立后，不能及时生效，需要使用start命令才可以生效。

1. responseTrigger是一个响应触发器，作用是某个connection接收到某个特定消息的时候产生反应，这个反应可以给原connection发送一条Diameter消息。或者给另外一个connection发送任意一个Diameter消息。

下面看示例4：

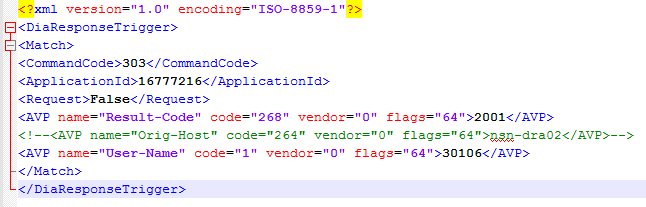


这个命令的含义是：

create一个responseTrigger，对象名为rspt1，使用example\rspt1.xml这个文件的描述来建立。如果匹配成功了，则在cn2这个connection上，发送message对象uda1这个Diameter消息。

注意一点，responseTrigger在建立时没有指出是在哪个链接上收到消息时进行匹配，并且判断是否进行后续动作的。后面的change命令里面会提到responseTrigger会挂在一个connection上，并且针对这个connection上面所有收到的消息进行匹配。

下面来看rspt1.xml：

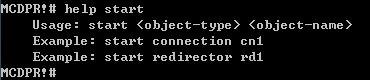
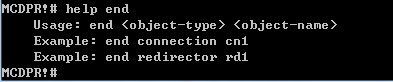


上面的示例5，表示如果消息的applicationid是16777216，command的code是303，并且不是一个Request（看起来这是在匹配一个Answer）。并且这个Answer中的Result-Code这个值是2001（Diameter\_Success），并且User-Name里面包含有30106这个字符串，则匹配成功。

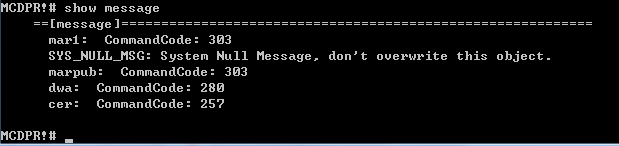
注意，MCDPR不关心AVP名称，只关心vendor和avp code。AVP名称只是便于我们区分AVP用的。

1. 所有的create命令都已经列举完毕。

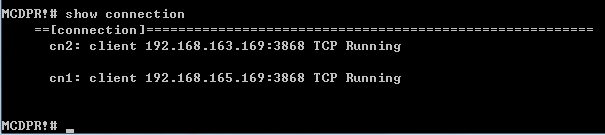
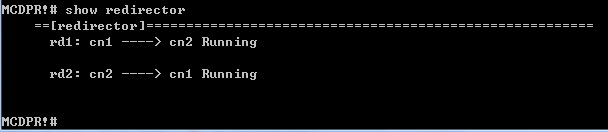
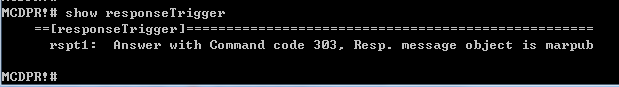
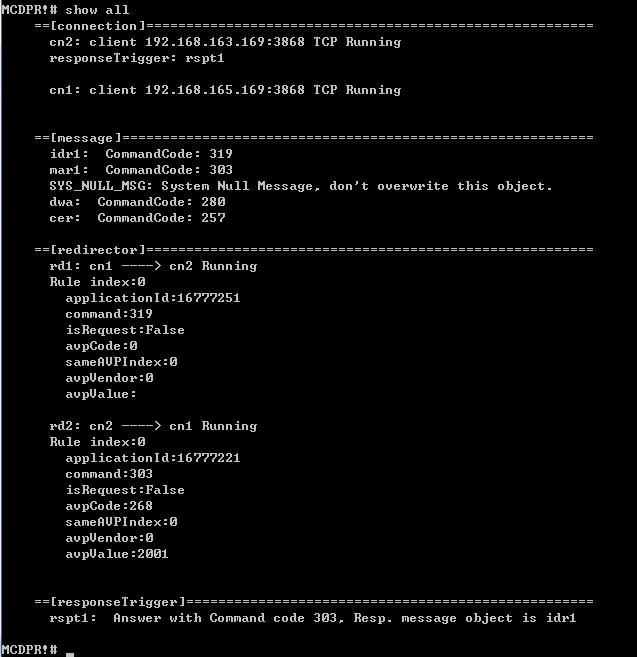
# MCDPR命令start/end

1. start命令用去启动connection和redirector对象，用法简单。
2. 只有使用start命令后connection和redirector对象才能被启动并且工作。
3. end命令用于停止connection和redirector
4. 停止的connection和redirector处于不工作状态。

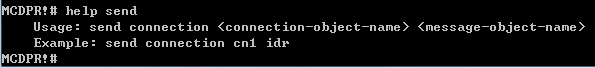
# MCDPR命令show

1. show 命令用于显示当前系统状态。
2. show message可以显示当前MCDPR正在运行中的message都有哪些。上面的示例显示系统中有5个message对象，名称为何，以及对应的commandcode是多少。

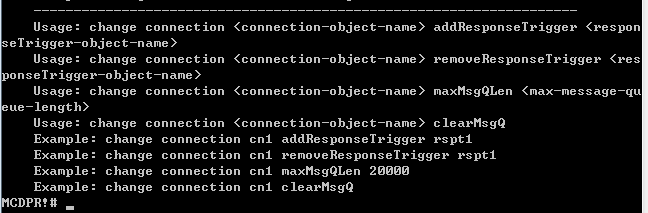
值得注意的是，SYS\_NULL\_MSG这是一个系统使用的空消息，不要将其覆盖。

1. show connection可以显示当前MCDPR正在运行中的connection以及他们的状态。
2. show redirector可以显示当前MCDPR正在运行中的redirector以及他们的状态。
3. show responseTrigger可以显示当前MCDPR正在运行的responseTrigger以及他们的状态。
4. show all可以显示所有对象的状态。

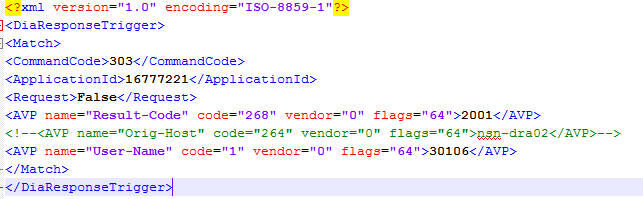
# MCDPR命令send

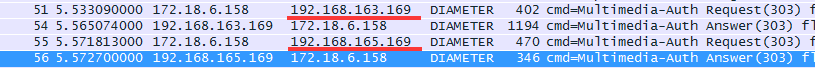
1. send命令用于在命令行中手动的向一个连接发送某个消息对象。
2. 例如下面的示例中，向cn1发送了mar1这个消息对象。
3. 下面是wireshark抓取的消息：

# MCDPR命令change

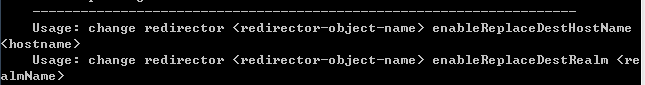
1. change用于修改某些对象的属性，目前支持的可以修改的对象类型为connection和redirector
2. connection可以修改的属性包括responseTrigger、MsgQ等，下面是帮助信息。
3. 参见create命令中第8小节，我们提到过在建立responseTrigger的时候，并没有指出这个responseTrigger是为哪一个connection工作的。指定connection的工作由change命令来完成。

rspt1.xml

wireshark



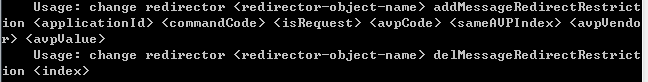
关于responseTrigger以及redirector的工作原理，在MCDPR进阶章节中还有详细解释。

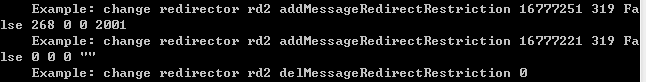
1. redirector的可修改属性是两部分内容，如下所示：



上面的示例可以允许redirector在转发信令的时候更改DestHostName以及DestRealm。目前版本的MCDPR不支持disable的动作，未enable之前即disable。

另外可以修改的属性是，为redirector添加一个禁止转发原则。即遇到什么样的消息，则不再转发。



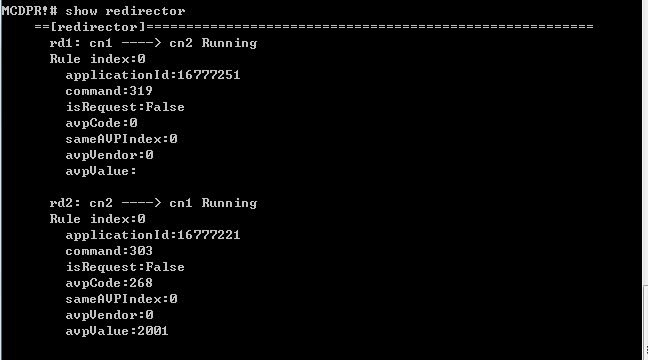


上面例子中，第一条命令是给rd2添加一个禁止转发原则，当遇到application id为16777251且319的commandcode，并且不是一个Request，且vendor 0的avp code 268 (Diameter Result)的值是2001，则此消息不转发。

第二条命令，是rd2添加一个禁止转发原则，当遇到消息是application id为16777221，commandcode为319，不是一个Request，则立刻停止转发。

第三条命令为移除禁止转发原则0。

添加了禁止转发规则的redirector，使用show命令查看时，类似于下图：



注意，进行禁止转发原则添加的时候，相关的AVP的类型信息必须在全局文件AvpDict.xml中正确配置。

1. 每个Connection都有一个MessageQueue(MsgQ)，这是一个缓冲区，缓冲Connection收到的消息。如果没有类似Redirector这样的对象来消耗掉缓冲区内的消息。则该缓冲区中的数据会越来越多。

MCDRP提供了一个方法来设置缓冲区(MsgQ)的大小。





也可以通过命令来清理缓冲区：



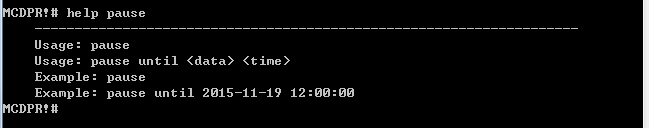


# MCDPR命令sleep

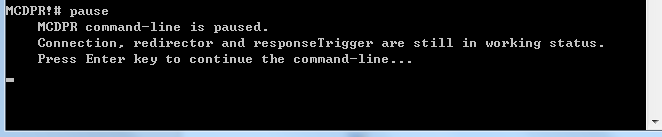
1. sleep命令用于暂停当前命令行的输入，用户按Enter键可以继续执行。
2. sleep后面使用微妙表示要sleep的时间长度。

# MCDPR命令pause

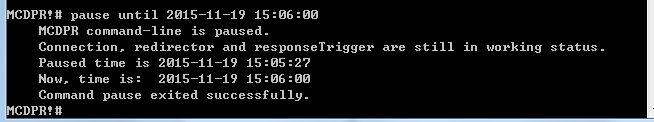
1. pause命令的文档说明是在1.4版本中加入的，但是pause命令本身并非如此。pause命令本身在MCDPR的最初原型版本0.1中即已经存在，但是在最初的版本说明中作者将这个重要命令遗忘了。直到1.4版本修正pause命令(加入until用法)，才一并加入说明。
2. pause的帮助信息如下：(pause的帮助信息是在版本1.4中加入的)



1. pause命令无参数情况下，直接暂停命令行窗口的输入。直到用户输入下次回车为止。



1. 另一种用法是pause until <date> <time>



pause命令会暂停命令行执行，直到规定时刻。

这个用法针对两种情况，一种是用户希望在某个特定时刻开始某个动作。另一种情况是，如果用户希望打开若干个MCDPR的实例，并且希望他们在同一时刻开始某个动作。

# MCDPR命令repeat

1. repeat命令是在1.4版本中加入的最重要的命令。在系统中没有任何关于repeat命令的帮助信息。一个是因为repeate属于流程控制命令，另外它还有需要完善的地方，所以帮助信息将在今后的版本中加入。
2. repeat的使用方式为：

repeat <times>

command 1

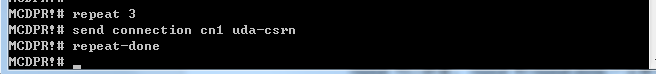
command 2

repeat-done

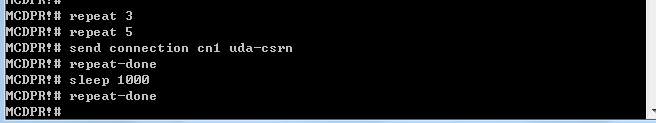
在repeat和repeat-done之间的命令，会被执行times指定的次数。

repeat可以嵌套， repeat和repeat-done一定要严格匹配。

1. 下面是一个repeat使用的例子，进行3次uda-csrn对象的发送：



1. repeat命令建议写在命令文件中，而不是手动输入，手动输入的时候，虽然循环次数是不会出错的，但是在输入repeat-done之前，输入的任何命令会被立即执行而不会等到repeat-done出现后一起执行。
2. repeat嵌套情况：



上面的例子是，循环3次，每次循环里面发送5次uda-csrn，发送5次后休息1秒钟。一定要注意repeat-done和repeat的嵌套方式。

# MCDPR进阶

1. 关于MCDPR进阶以及实例分析的内容，请参见文档Guide2。