ПРОТЕЙ
НАУЧНО-ТЕХНИЧЕСКИЙ ЦЕНТР

mAccess.MAK

Утилита _debug

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Оглавление

1 Общие сведения

Данный документ содержит руководство пользователя, описывающее использование утилиты **_debug**.

Документ предназначен для сотрудников технической поддержки и системных администраторов, занимающихся настройкой и сопровождением оборудования МАК.

Внимание! Производитель оставляет за собой право на изменение имен, состава и формат команд утилиты. Производитель обязуется выпускать обновленную версию данного документа в случае модификации команд утилиты **_debug**. При получении новой версии программного обеспечения МАК пользователь вправе требовать от производителя обновленную версию данного документа или подтверждение неизменности содержимого документа.

Техническая поддержка

Техническая поддержка, а также дополнительное консультирование по вопросам, возникающим в процессе установки и эксплуатации изделия, осуществляются производителем и службой технической поддержки.

Производитель

НТЦ «ПРОТЕЙ»

194044, Санкт-Петербург

Большой Сампсониевский пр., д. 60, лит. А

Бизнес-центр «Телеком СПб»

Тел.: (812) 449-47-27

Факс: (812) 449-47-29

WEB: http://www.protei.ru

E-mail: info@protei.ru

Служба технической поддержки

НТЦ «ПРОТЕЙ»

194044, Санкт-Петербург

Большой Сампсониевский пр., д. 60, лит. А

Бизнес-центр «Телеком СПб»

Тел.: (812) 449-47-27 доп. 5999 (круглосуточно)

(812) 449-47-31

Факс: (812) 449-47-29

WEB: http://www.protei.ru

E-mail: support@protei.ru

2 Доступ к утилите _debug

Доступ к МАК осуществляется с внешнего компьютера, подключенного к общей с МАК локальной сети. Другой способ доступа к МАК с внешнего компьютера — RS232-порт, для этого соедините внешний компьютер и МАК RS232-кабелем, входящий в комплект поставки.

Обычно при работе с утилитой _debug используется подключение к локальной сети, при этом используется протокол telnet.

Если на компьютере пользователя установлена операционная система Linux, то подключиться по telnet к МАК достаточно просто. Для этого надо вызвать программу консоли (konsole), и в ней набрать строку вида:

telnet IP-адрес МАК.

Примечание. Далее в этом разделе строки, которые пользователь должен набрать на клавиатуре выделены **полужирным шрифтом**.

Пример входа на удаленный компьютер с использованием telnet:

*строка приглашения ОС>**telnet 192.168.1.23***

После успешного соединения с МАК по telnet, на экране появится запрос на ввод имени пользователя (login):

login:

Введите имя пользователя и нажмите клавишу <Enter>, появится запрос пароля:

Password:

Введите пароль. Если введены верные имя пользователя и пароль, произойдет вход в систему. Далее набрать строку следующего вида (только то, что выделено полужирным шрифтом):

*строка приглашения>**_debug***

При появлении проблем с доступом к МАК или вызовом утилиты _debug обратитесь к системному администратору.

Если на компьютере пользователя установлена операционная система Windows, то воспользуйтесь программой PuTTY, предназначенная для соединения с внешними устройствами через локальную сеть с использованием протокола telnet, или через RS232-порт. После вызова PuTTY, на экране появится диалоговое окно приложения, где необходимо указать IP-адрес МАК и порт telnet или выбрать их из списка ранее сохраненных соединений. По нажатию клавиши «ОК» появится окно консоли. Далее все действия идентичны действиям, которые были описаны выше в этом разделе для Linux.

Логин, пароль, IP-адрес МАК, необходимо узнать у системного администратора.

3 Общие сведения о журналах

Программное обеспечение МАК ведет несколько журналов, большинство из которых отвечают за сообщения, генерируемые одной конкретной подсистемой. Под подсистемами в данном случае понимаются модули обработки конкретной системы сигнализации (например, SIP, SS7, DSS1), модуль управления конфигурацией, модуль аварийной индикации и т.п.

С точки зрения функционального назначения журналов их можно разделить на классы, которые представлены в таблице ниже.

Таблица. Классы журналов МАК

Название класса	Описание
alarms	Статистическая информация и события, получаемые от подсистемы аварийной индикации
cdr	Сводная информация по каждому вызову
com	Информация подсистемы управления конфигурацией
debug	Подробная отладочная информация
diagnostic	Информация о причинах разрушения или неуспешности установления соединения в результате ошибочных ситуаций возникших в рамках диагностирования оборудования
event	Трассировка событий
info	Общая статистическая информация по нескольким подсистемам а также вывод файлов конфигурации
message	Сообщения сигнализации
message_bad	Некорректные сообщения сигнализации
monitor	Низкоуровневый мониторинг сигнальных каналов
rtp	Статистическая информация по RTP-сессиям
rtp_bad	Статистическая информация по RTP-сессиям, в рамках которых наблюдался значительный процент потерь пакетов
warning	Ошибки и предупреждения

4 Использование утилиты _debug

После запуска утилиты на экране появляется приглашение к вводу команды в виде знака «**>**». Для получения списка доступных команд следует ввести знак вопроса или команду **help** и нажать клавишу <Enter>:

>?

Available commands:

help	*Print this help*
exit	*Exit utility*
commit	*Apply new debug configuration*
debug <log_name ...>	*Online debug*
global tracing {on|off}	*Enable/disable tracing*
default-debug-configuration	*Use it on first start*
local write {on|off}	*Enable/disable local logs*
remote tracing {on|off}	*Enable/disable remote tracing*
remote ip <ip>	*Set remote tracing IP*
remote port <port>	*Set remote tracing UDP-port*
set-level journal <journal_name> <level 0-10>	*Set level for specified journal*
set-level subsystem <subsystem_name> <level 0-4>	*Set level for specified subsystem*
set-level log <log_name> <level 0-4>	*Set level for specified log*
show debug-level journal <journal_name>	*Show level for specified journal*
show debug-level subsystem <subsystem_name>	*Show level for specified subsystem*
show debug-level log <log_name>	*Show level for specified log*

Если знак вопроса введен после каких-либо символов, то будут выведены все возможные варианты продолжения команды:

>show ?

Available commands:

show debug-level journal <journal_name>	*Show level for specified journal*
show debug-level subsystem <subsystem_name>	*Show level for specified subsystem*
show debug-level log <log_name>	*Show level for specified log*

Также с помощью знака вопроса можно получить список имен доступных журналов, подсистем или файлов:

>show debug-level subsystem ?

List of system subsystems:

ap

com

common

dss1

sip

ss7

sub

tr_sl

>

Для запуска команды не обязательно вводить ее имя полностью - достаточно ввести первые символы слов, однозначно определяющие команду, например, вместо

>**show debug-level subsystem sip**

можно ввести

>**sh d s si**

В обоих случаях результат выполнения этой команды будет выглядеть примерно так:

=== List of journals in "sip" subsystem ===

Journal "SIP_IB" level is 10

Journal "sip" level is 4

Journal "sip_call_trace" level is 10

Journal "sip_cevent_trace" level is 10

Journal "sip_diagnostic" level is 10

Journal "sip_diagnostic_warning" level is 10

Journal "sip_ib_call_trace" level is 10

Journal "sip_ib_cevent_trace" level is 10

Journal "sip_ib_diagnostic" level is 10

Journal "sip_ib_diagnostic_warning" level is 10

Journal "sip_transport" level is 10

Все команды утилиты **_debug** делятся на модифицирующие и немодифицирующие. Немодифицирующие команды — это в основном команды чтения (вывода) текущей конфигурации.

После выполнения модифицирующих команд необходимо выполнить команду **commit** («применить»), которая сохранит изменения.

В таблице ниже приведено описание команд утилиты _debug.

Таблица. Команды утилиты _debug

Команда	Описание
commit	Применение изменений конфигурации, осуществленных после предыдущего выполнения команды commit
debug <log_name ...>	Online вывод сообщений, записываемых в определенный лог (например, message, info, warning). Все возможные имена логов можно

Команда	Описание
	получить с помощью команды >debug ? Вместо имени конкретного лога можно указать идентификатор all – в этом случае будет производится вывод сообщений, записываемых во все логи.
global tracing {on|off}	Отключение/включение режима трассировки. Отключение ведет к полному отсутствию какой-либо отладочной информации.
default-debug-configuration	Установка конфигурации подсистемы трассировки по умолчанию и создание необходимых служебных файлов.
local write {on|off}	Отключение/включение записи сообщений подсистемы трассировки на локальный носитель
remote tracing {on|off}	Отключение/включение записи сообщений подсистемы трассировки на удаленный модуль
remote <ip>	IP-адрес удаленного модуля, на который ведется запись сообщений подсистемы трассировки
remote port <port>	Порт назначения для удаленном модуле, на котором принимаются сообщения подсистемы трассировки
set-level journal <journal name> <level 0-10>	Изменение уровня вывода сообщений определенного журнала (например, DSS1_message, ISUP_info, MTP_warning). Возможные имена журналов можно получить с помощью команды >set-level journal ? Вместо имени конкретного журнала можно указать идентификатор all – в этом случае будет производится вывод сообщений, записываемых во все журналы.
set-level subsystem <subsystem_name> <level 0-4>	Изменение уровня вывода сообщений определенной подсистемы (например, sip, ss7 или dss1). Возможные имена подсистем можно получить с помощью команды >set-level subsystem ? Вместо имени конкретной подсистемы можно указать идентификатор all – в этом случае будет

Команда	Описание
	производится вывод сообщений, записываемых во все подсистемы.
set-level log <log_name> <level 0-4>	Изменение уровня вывода сообщений в определенный лог (например, message, info, warning). Все возможные логов можно получить с помощью команды >set-level log ? Вместо имени конкретного лога можно указать идентификатор all – в этом случае будет производится вывод сообщений, записываемых во все логи.
show debug-level journal <journal_name>	Вывод текущего уровня трассировки конкретного журнала
show debug-level subsystem <subsystem_name>	Вывод текущего уровня трассировки конкретной подсистемы
show debug-level log <log_name>	Вывод текущего уровня трассировки конкретного лога

Если с использованием утилиты **debug** была разрешена удаленная запись журналов, то на компьютере, куда предполагается писать журналы необходимо запустить утилиту **remote_logger**. Описание работы с утилитой **remote_logger** приведено в приложении.

ПРИЛОЖЕНИЕ. Утилита удаленной записи журналов

Утилита remote_logger предназначена для удаленного приема и записи журналов, формируемых программным обеспечением МАК.

Утилита **remote_logger** должна располагаться на компьютере, куда предполагается писать журналы с удаленного компьютера (МАК).

Утилита **remote_logger** имеет настраиваемые параметры, которые хранятся в файле — **config/dump.cfg**.

При упрощенном представлении об утилите remotr_logger можно сказать, что данная утилита «прослушивает» порт (порты), который определяется в dump.cfg, и записывает поступающую информацию в файл (файлы).

Важным параметром утилиты remote_logger является период обновления, фактически это означает, что по окончании текущего периода закрывается текущий файл журнала, и открывается новый файл журнала, в который будет писаться продолжение журнала. Таким образом, в результате, журнал - это набор нескольких файлов, каждый из которых создается утилитой remote_logger через период обновления.

Формат конфигурационного файла config/dump.cfg:

```
[default]
_remote_logger_default_common =
        # настройки этого журнала будут использоваться для ведения журналов,
        # которые не определены заранее и для их порта не определены свои
        # настройки по умолчанию
{
    # <описание параметров журнала в формате файла trace.cfg, параметр file игнорируется>
};

[server]
{
    port = <номер порта>;
    dir = <директория для сохранения журналов, полученных с этого порта>;
    logs =
    {
        _remote_logger_default =      # настройки этого журнала будут использоваться
                                      # для ведения журналов, получаемых с этого
                                      # порта и не определенные заранее
        {
            # <описание журнала в формате файла trace.cfg, параметр file игнорируется>
        };

        # <описание конкретных журналов в формате файла trace.cfg, параметр file учитывается>
   };
};
```

```
...
{
   port = <номер порта>;
   dir = <директория для сохранения журналов, полученных с этого порта>;
   logs =
   {
        _remote_logger_default =       # настройки этого журнала будут использоваться
                                       # для ведения журналов, получаемых с этого порта
                                       # и не определенные заранее
        {
          # <описание журнала в формате файла trace.cfg, параметр file игнорируется>
        };

        # <описание конкретных журналов в формате файла trace.cfg, параметр file учитывается>
   };
};
```

В описании формата представленного выше, есть строки - «описание журнала в формате trace.cfg», это означает наличие набора конфигурируемых параметров (имя параметра — описание параметра):

- **level** - уровень журнала (сообщения полученные с удаленной стороны выводятся утилитой remote_logger с уровнем 1, сообщения самой утилиты remote_logger - с уровнем, указанном в данном параметре);
- **mask** - маска формата вывода автоматических полей в журнале;
- **type** - тип журнала (по умолчанию `type = log + append`, для любого заданного типа журнала при повторном его переоткрытии, новые данные будут добавляться в конец файла, чтобы этого не происходило нужно явно указать в типе truncate);
- **file** - имя выходного файла с маской;
- **separator** - разделитель автоматических полей;
- **limit** - ограничение на максимальное количество записей;
- **period** - период обновления файла лога.

Примечание. Если получен заранее неопределенный журнал, и нет настроек по умолчанию (ни для конкретного порта, ни общих), то журнал не создается и не ведется.

Пример файла dump.cfg:

```
[default]
_remote_logger_default_common =
{
      level = 2;
};
```

```
[server]
{
	port = 32006;
	dir = ./logs/32006Dir;
	logs =
	{
		trace =
		{
			file = trace_%H_%M_%S.log;
			level = 2;
			period = 5hour;
		};

		my_own =
		{
			file = my_own_%H_%M_%S.log;
			level = 2;
			period = 5hour;
		};
	};
};
```

Еще пример пример файла dump.cfg:

```
[default]
_remote_logger_default_common =
{
	level = 2;
};

[server]
{
	port = 32006;
	dir = ./logs/32006Dir;
	logs =
	{
		_remote_logger_default =
		{
			#type = truncate;
			level = 2;
			mask = date;
			period = 30min;
```

```
            };

            trace =
            {
                  file = trace_predef.log;
                  level = 2;
                  mask = date;
                  #type = truncate;
                  period = 1sec;
            };
      };
};

{
      port = 32005;
      dir = ./logs/32005Dir;
      logs =
      {
            _remote_logger_default =
            {
                  #type = truncate;
                  level = 2;
                  mask = date;
                  period = 30min;
            };

            trace =
            {
                  file = trace_predef.log;
                  level = 2;
                  #type = truncate;
                  mask = date;
                  period = 2min + 1sec;
            };
      };
};
```