Терминальный клиент. Описание

Терминальный клиент, это frontend часть к системе CLI, выполненная в виде консольного приложения. Данное консольное приложение предназначено для использования в качестве командного интерпертатора. Это означает, что мы можем создать пользователя, назначит ему терминальный клиент в качестве командного интерпретатора, после чего при заходе в систему под этим пользователем (как локально, так и через telnet, ssh) мы будем попадать в терминальный клиент. При этом один экземпляр терминального клиента предназначен для взаимодействия только с одним пользователем, при этом в системе одновременно может выполняться несколько терминальных клиентов.

Функциональность:

* Один экземпляр терминального клиента взаимодействует только с одним пользователем.
* Принимает ввод пользователя.
* Позволяет редактировать ввод пользователя при помощи той же комбинации клавиш, что и командная оболочка bash.
* Поддерживает историю команд в том же виде (с тем же набором комбинаций клавиш и команд), что и командная оболочка bash.
* Поддерживает автодополнение вводимой строки; за данными об автодополнении обращается на cli сервис.
* Поддерживает несколько разных режимов работы (состояния) терминального клиента. В зависимости от режима изменяется функциональность и поведение терминального клиента. На данный момент поддерживаются следующие режимы работы: режим ввода команд, режим обработки команд, режим ввода данных аутентификации (режим ввода имени пользователя и пароля).
* Поддерживает жестко заданный набор команд, выполнение которых происходит на самом терминальном клиенте. На данный момент в этот набор команд входят следующие команды: **login** и **bye**.
* Отправляет ввод пользователя, не являющийся одной из команд терминального клиента, на cli сервис для его обработки.
* Принимает сообщения от cli сервиса; реакция на эти сообщения зависит как от вида сообщений, так и от текущего режима работы (состояния) терминального клиента.

Поведение

* При запуске терминального клиента происходит обращение на cli сервис для получения начального состояния (строки подсказки или prompt) при помощи синхронного обмена сообщениями.
* В режимах работы терминального клиента, поддерживающих автодополнение, за данными автодополнения терминальный клиент обращается на cli сервис при помощи синхронного обмена сообщениями.
* Ввод пользователя считается завершенным при нажатии на клавишу «Enter». После этого ввод пользователя обрабатывается как команда (либо на терминальном клиенте, либо на cli сервисе).
* После завершения ввода пользователем, этот ввод сначала проверяется на соответствие одной из команд терминального клиента. Если ввод пользователя соответствует одной из команд терминального клиента, то выполняется эта команда. Если ввод пользователя не соответствует какой-либо команде терминального клиента, то ввод пользователя считается командой cli сервиса и посылается на cli сервис при помощи асинхронного сообщения.
* При переходе в режим обработки команд (после отправки ввода пользователя на cli сервис) терминальный клиент ожидает одно или несколько сообщений следующих типов: **command\_out**, **command\_err** и **end**. При получении сообщения типа **command\_out**, терминальный клиент выводит содержимое этого сообщения в стандартный поток вывода. При получении сообщения типа **command\_err**, терминальный клиент выводит содержимое этого сообщения в стандартный поток вывода ошибок. При получении сообщения типа **end**, терминальный клиент переходит в режим ввода команд. При этом содержимое сообщения типа **end** трактуется как строка подсказки (prompt).
* Результат прерывания работы по нажатию на комбинацию клавиш Ctrl+C зависит от текущего режима работы терминального клиента: для режима ввода команд прерывание приведет к сбросу ввода пользователя, для режима обработки команд — к отправке на cli сервис сообщения **interrupt**, для режима ввода данных аутентификации — к сбросу введенных данных аутентификации и переходу в режим ввода команд.
* Прерывание работы по нажатию на комбинацию клавиш Ctrl+C для режима обработки команд приведет к отправке на cli сервис асинхронного сообщения **interrupt**. В ответ на это сообщение cli сервис может как сразу прислать сообщение типа **end**, так и после отсылки ряда других сообщений (сообщений типа **command\_out** и **command\_err**).
* Прерывание работы по нажатию на комбинации клавиш Ctrl+Z или Ctrl+\ приводит к завершению работы терминального клиента. Перед завершением работы на cli сервис посылается асинхронное сообщение **exit**.
* При нажатии на комбинации клавиш Ctrl+D происходит выход из текущего режима работы (если это возможно). При этом на сервер шлется специальная команда и терминальный клиент переходит в режим обработки команд.
* При нажатии на клавишу ? происходи запрос помощи или списка доступных команд. При этом происходит синхронное взаимодействие с сервером. Выбор того, что на самом деле запрашивается на сервере зависит от последнего введенного символа (перед нажатием на клавишу ?). Если этот символ пробельный (пробел, табуляция), то происходит запрос помощи. В противном случае происходит запрос списка доступных команд. Если в результате запроса помощи или списка доступных команд ничего не будет найдено, то на экран ничего не выводится; в том числе и символ '?'. Если в результате запроса помощи или списка доступных команд будет найдена некоторая информация, то на экран выводится символ '?', символ перевода строки и найденная информация (уже на новой строке).
* Если пользователь бездействует в течении некоторого времени, то терминальный клиент получит уведомление о завершении клиентской сессии данного пользователя на cli\_service. Данное сообщение выводится в поток стандартного вывода, после чего терминальный клиент завершит свою работу.

Режимы работы терминального клиента

Терминальный клиент во время своей работы находится в том или ином режиме работы (состоянии). В зависимости от режима работы изменяется функциональность и поведение терминального клиента. На данный момент определены следующие режимы работы терминального клиента:

* Режим ввода команд. В этом режиме терминальный клиент ожидает ввода команд от пользователя, позволяя пользователю редактировать свой ввод, пользоваться историей команд и автодополнением. При завершении ввода, введенная команда выполняется либо локально (на терминальном клиенте), либо передается на cli сервис. Если введенная команда является командой cli сервиса (т.е. передается на cli сервис), то терминальный клиент переходит в режим обработки команд; если введенная команда является командой **login** терминального клиента, то терминальный клиент переходит в режим ввода данных аутентификации. Терминальный клиент не ожидает никаких сообщений со стороны cli сервиса (за исключением синхронного запроса данных об автодополнении) в этом режиме работы. Прерывание работы по нажатию на комбинацию клавиш Ctrl+C приводит к сбросу ввода пользователя.
* Режим обработки команд. В этом режиме терминальный клиент ожидает окончания обработки команды на cli сервисе. Ввода от пользователя не ожидается; поэтому возможность редактирования ввода, использования истории команд и автодополнения отсутствует. В режиме обработки команд терминальный клиент ожидает сообщения следующего типа от cli сервиса: **command\_out**, **command\_err** и **end** (см. протокол взаимодействия между терминальным клиентом и cli сервисом). При получении сообщения типа **command\_out**, терминальный клиент выводит содержимое этого сообщения в стандартный поток вывода. При получении сообщения типа **command\_err**, терминальный клиент выводит содержимое этого сообщения в стандартный поток вывода ошибок. При получении сообщения типа **end**, терминальный клиент переходит в режим ввода команд. Прерывание работы по нажатию на комбинацию клавиш Ctrl+C приводит к отсылке сообщения типа **interrupt** с терминального клиента на cli сервис.
* Режим ввода данных аутентификации (режим ввода имени пользователя и пароля). В этом режиме терминальный клиент ожидает от пользователь ввода имени пользователя и пароля, при этом имя пользователя маскируется (каким-либо способом). В этом режиме терминальный клиент позволяет пользователю редактировать свой ввод (по большей части при вводе имени пользователя), но не позволяет использовать историю команд и автодополнение. После ввода данных аутентификации, эти данные передаются на cli сервис с помощью команды cli сервиса, после чего терминальный клиент переходит в режим обработки команд. Прерывание работы по нажатию на комбинацию клавиш Ctrl+C переводит терминальный клиент в режим ввода команд.

Вполне возможно, что в будущем появится еще несколько режимов работы терминального клиента; например, режим текстового редактора, режим взаимодействия с интерактивной командой и т.п.

Команды терминального клиента

Терминальный клиент поддерживает ряд команд, выполнение которых происходит на самом терминальном клиенте, а не передается для выполнения на cli сервис. При завершении ввода пользователем, его ввод проверяется на соответствие команда терминального клиента. Если ввод пользователя распознается, как некоторая команда терминального клиента, то будет выполняться эта команда. Если ввод пользователя не распознается, как некоторая команда терминального клиента, то этот ввод будет послан на cli сервис.

На данный момент в набор команд терминального клиента входят следующие команды:

* Команда **login**; переводит терминальный клиент в режим аутентификации и позволяет пользователю ввести имя и пароль (при этом ввод пароля маскируется тем или иным способом). После ввода имени пользователя и пароля они передаются на cli сервис при помощи обычной команды cli сервиса.
* Команда **bye**; завершает работу терминального клиента. Перед завершением работы на cli сервис посылается сообщение **exit**.

Детали реализации

* Для поддержки такой функциональности, как редактирование ввода пользователя, автодополнение, история команд в терминальном клиенте мы используем библиотеку readline.
* Для поддержки запроса помощи и списка доступных команд при нажатии на клавишу ?, мы задаем обработчик нажатия этой клавишы при помощи функции **rl\_bind\_key**.
* При обмене сообщениями с cli сервисом в качестве сообщений мы используем объекты среды выполнения Erlang. Для их сериализации в массив байт (для передачи по сети) мы используем функцию **erl\_encode**, а для их десериализации из массива байт (для приема по сети) — функцию **erl\_decode** из библиотеки **erl\_marshal**. Эти объекты на самом деле являются кортежами, состоящими минимум из одного элемента (на данный момент — кортежами из одного или двух элементов)., причем первый элемент всегда является описателем сообщения (тип этого первого элемента всегда атом).
* После отправки терминальным клиентом cli команды на обработку cli сервисом, терминальный клиент ожидает одно или несколько сообщений следующих типов: **command\_out**, **command\_err** и **end** (см. протокол взаимодействия между терминальным клиентом и cli сервисом). Все эти сообщения являются кортежем из двух элементов, вторым элементом кортежа является всегда строка с данными (а первым элементом — описатель сообщения). Поэтому, в терминальном клиенте все эти три типа сообщений будут десериализоваться в объекты одного типа данных, состоящего из описателя и данных. Различать мы эти объекты будем по описателю.
* Такие операции, как обработка завершения ввода от пользователя, обработка автодополнения и обмен сообщений с cli сервисом не прерываются сигналами (SIGINT, SIGQUIT, SIGTSTP); это означает, что перед выполнение этих операций мы блокируем получение сигналов, а после выполнения этих операций мы разблокируем получение сигналов. При этом при блокировке сигналов мы используем пессимистический подход: блокируем получение сигналов для как можно более крупных блоков (например, блокируем полностью обработчик завершения пользовательского ввода).
* Терминальный клиент и cli сервис всегда работают на одном компьютере. По этой причине простые запросы данных терминальным клиентом с cli всегда выполняются синхронно. К таким простым запросам данных относятся запрос текущего состояния (обмен сообщениями с типами **current\_state\_request** и **current\_state\_response**) и запрос данных об автодополнении (обмен сообщениями с типами **extension\_request** и **extension\_response**). Весь остальной обмен сообщениями между терминальным клиентом и cli сервисом выполняется асинхронно.
* Терминальный клиент и cli сервис обмениваются сообщениями через сокеты TCP. В связи с тем, что терминальный клиент и cli сервис всегда располагаются на одном компьютере, более логично было бы использовать сокеты Unix. Однако, библиотека языка Erlang не содержит модулей, позволяющих использовать сокеты Unix, но содержит модуль **gen\_tcp**, позволяющий использовать сокеты TCP.

Конфигурирование

Терминальный клиент может быть сконфигурирован двумя возможными (не пересекающимися) способами: с использованием конфигурационного файла и с помощью опций командной строки. Наличие конфигурационного файла является обязательным; использование опций командной строки обязательным не является. Конфигурационный файл обрабатывается построчно; существуют следующие правила по его обработке (разбору или парсингу):

* Строки, начинающиеся с символа **'#'** считаются комментариями и игнорируются.
* Строки вида **KEY**=**VALUE** разбираются в конфигурационный параметр **KEY** имеющий строковое значение **VALUE**. Если конфигурационный параметр с именем **KEY** встречается в конфигурационном файле несколько раз, то поведение (на данный момент) не определено.
* Все остальные строки считаются некорректными. При появлении такой строки поведение (на данный момент) не определено (если быть более точным, то на данный момент мы пропускаем некорректные строки).

Следующие параметры конфигурационного файла определены на данный момент:

* Параметр, определяющий номер listen порта cli\_service: **port\_number=число**. Здесь **число** должно иметь значение из диапазона 0-65535. Параметр **port\_number** является обязательным.

По умолчанию, конфигурационный файл должен иметь следующее имя: */etc/cli\_terminal/cli\_terminal.conf*.

Опции командный строки являются не обязательными; следующие опции командной строки определены на данный момент:

* Параметр, определяющий местоположение конфигурационного файла: **--config=config-file-location**. Здесь **config-file-location** — путь (относительный или абсолютный) до конфигурационного файла терминального клиента. Данный параметр позволяет использовать конфигурационный файл с произвольным именем, распологающийся в произвольном месте файловой системы.