Hasq JavaScript Client

23 May 2015, ver 0.4.1

Introduction

Hasq JavaScript Client (the Client) is a simple client program used to connect to Hasq servers and to manipulate database records. Its simple design makes it quick to learn and easy to work with. At the same time, the ability to issue Hasq network commands directly, as well as having full control over the content of records Key, Generator and Owner fields makes it a powerful tool for performing low level operations with tokens (DN's).

The Client's primary purpose is to serve as a research tool for those who want to understand Hasq technology better, or as a no-nonsense client program for someone who is comfortable with direct manipulation of records.

Limitations

The current version of the Client works with Hasq databases which have one Generator field. The Client's interface supports only two operations with records out of many permitted by Hasq servers - creating a new record in a chain and extracting the last record. High level token management is also not supported (token issuance, statistics etc).

Description of Client's User Interface

Server tab

This tab provides information about the server the Client is connected to. The information includes system parameters (disk and memory use, CPU load) as well as specifics like server name, version etc. Pressing the [Refresh] button in the top right corner causes the Client to update this information.

The lower part of the tab window contains a table of neighbouring Hasq servers connected to the current server. A mouse click on any of the links in the table causes the Client to connect to the corresponding server. Information in all of the Client's tabs will be updated accordingly. This feature allows for exploration of the network of servers that the current server belongs to.

Database tab

The Hasq server may have a number of databases that it maintains. When the Client first connects to a server, it requests a list of databases located on this server. It then selects the first database to be current. Any user command (except direct commands issued in the Command tab) is performed on the current database.

This tab allows users to change the current database. This is achieved by clicking on the button at the top of the tab window and choosing a name from the list. The table below the button contains properties (traits) of the chosen database. While all properties are important, some of them are more informative for users than others. These include the database name, hash type (e.g. MD5) used by this database, and the records' data limit.

The data limit shown in the table has the following format:

(*n* | *n*b | *n*B) | (*n*k | *n*K) | (*n*m | *n*M) | (*n*h | *n*H) | -1

where

n is a number

n | *n*b | *n*B specifies data limit in bytes, e.g. 512B (512 bytes)

*n*k | *n*K specifies data limit in kilobytes, e.g. 4k (4096 bytes)

*n*m | *n*M specifies data limit in megabytes, e.g. 2M (2097152 bytes)

*n*h | *n*H specifies data limit in hashes used by this database. The corresponding size in bytes can be calculated using the following formula:

data-limit-in-bytes = *n* * *hex-hash-size* + *n*

Example. Say, a database uses MD5. The hexadecimal text representation of any MD5 hash is 32 characters long, so if the data limit is specified as 4H, its size in bytes will be 132 (4 * 32 + 4 = 132)

-1 means that there is no records' data limit in place

Records tab

This tab is used to manipulate records in the chosen database. The name of the current database is shown at the top left corner of the tab window.

Two operations with records can be performed here - extraction of the last record in a chain and creation of a new record.

Either operation requires a DN (token) to be specified first. A DN can be typed directly in the [DN] window or automatically generated from the content of the [Raw DN] field above. Since any DN is a hash-like string, some checks are performed if the [DN] field is changed manually. The purpose of the checks is to prevent incorrect characters from being used in a hash as well as to make sure that the length of a string in the [DN] field matches the length of a hexadecimal text representation of the database's hash. If there is no match, a red frame will be displayed around the [DN] field. The same applies to all other editable fields where a hash-like string is expected. For example, if the database's hash is MD5, the content of the [DN] field must be 32 characters long.

Pressing the [Get Last Record] button causes the Client to send a request to the server to extract the last record for the DN specified in the [DN] field. If the request is fulfilled successfully, the components of the extracted record (Record number, Key, Generator, Owner and Data) will be displayed in the corresponding fields on the right hand side of the button. The status of the operation (success or failure) will be shown at the bottom of the tab window.

The lower part of the window contains controls for creating a new record for the DN shown in the [DN] field above. These include fields for the record's passwords and for the record's components. All of the component fields can be edited manually or filled in automatically.

Key, Generator and Owner hashes can be produced with the help of passwords. There are two options available: one password for all three components or a separate password for each

component. Two checkboxes in the lower part of the window are provided for switching between these options.

Typing in the password fields causes the Client to produce Key, Generator and Owner components of the new record. The newly generated values will be displayed in the corresponding fields above the [Submit] button.

Once the components of the new record are known (either generated automatically or entered manually), the Client checks if this record can be linked to the previous record in a chain. A message, containing the result of the check is displayed at the bottom of the tab window. The Client may not have enough information to perform such a check in which case the corresponding message is also displayed. If, however, the check returns a negative result, a red frame around the [Submit] button will be shown. A green frame indicates that the check was successful. Pressing the [Submit] button causes the Client to send a command to the server to add a new record to the DN's chain. The result of this command (success or failure) will be displayed at the bottom of the window.

Hash calc

This tab provides users with a handy hash calculator. The types of hash functions supported include MD5, RIPEMD-160, SHA2-256, SHA2-512 and WORD.

WORD is a Hasq own hash type. Records built with the use of WORD are short, which makes them easy to read/print. This makes WORD an ideal hash type for anyone who wants to better understand how records link to each other and needs to use hashes in their calculations/modelling. WORD can also be used by developers as a temporary replacement for other hashes while they debug their source code. Due to the short length of WORD hashes (4 characters only), other real life applications are limited.

The type of hash used by the calculator can be chosen by pressing a button at the top of the tab window. Text data whose hash value needs to be calculated should be typed in or copied into the area directly below the button. Every time the content of this area changes, its hash value is calculated and displayed in the window below.

Three buttons at the bottom of the tab window are provided for user convenience. They copy the calculated hash value into the Key, Generator or Owner fields located in the 'Records' tab. This may be handy if a user needs to manually calculate hash values for those fields.

Admin tab

The 'Admin' tab provides users with a way to communicate with Hasq servers directly. Any Hasq network command can be typed in the text field on the right-hand side of the [Send] button. Pressing the button causes the Client to send this command to the server. The complete server reply is displayed below.

Tokens tab

The "Tokens" tab provides a user with an option to manipulate tokens (DNs) at a higher level. Five operations are permitted - "Create", "Verify", "Data", "Send" and "Receive". Each operation will be performed for a group of tokens specified in the top text area. If some of the tokens in a group are invalid, a corresponding error message will be displayed.

Tokens can be added to a group manually (hash or raw value in square brackets) or with the use of a token range generator located below the top text area.

A password, specified in the field below the range generator, will be used for every token in the group.

Create tab

This tab contains only one button "Create". Pressing this button causes the Client to send a token creation request to the Server. The Server's response will be displayed below.

Verify tab

This tab contains only one button "Verify". Pressing this button causes the Client to get the last available record for every token in a group and display these records below.

Data tab

"Data" tab is used to add or modify tokens' data. New data should be enetered into the provided text field. Pressing "Update" button causes the Client to send data modification request for each token in the group to the Server. The Server's response will be dispalyed below.

Send/Receive tab

"Send" and "Receive" tabs contain controls that are used by a user when they wish to pass tokens' ownership to another person or to accept a new token from somebody else.

There are 4 scenarios how ownership can be transferred. Scenarios 1 and 3 assume that the current token owner initiate the steps necessary to change the ownership. Pressing buttons "Initiate", "Initiate Step 1" and "Initiate Step 2" generate hex strings that should by passed to a new token owner via any communication channel.

Upon receiveing the generated hex strings, the new token owner should insert them into the corresponding fields in the "Receive" tab and press one of "Finalize", "Finalize Step 1" or "Finalize Step 2" buttons. Pressing the button either finalizes token ownership transfer or initiates the second step.

Scenarios 2 and 4 are similar to scenarios 1 and 3 with the difference being the new token owner initiating the token ownership transfer.