**The distrib-protocol 2015**

**General**

* The purpose of this protocol is to ensure compatibility between implementations. **You are NOT allowed to change this protocol in any way**. (extensions are allowed).
* If either endpoint of the communication notices that this protocol is not followed correctly, the connection should be terminated immediately.
* Protocol can contain any octects (bytes) with values between 0-255. For human readable fields the accepted ascii characters are: a-zA-Z0-9/:.-\_~%', space and linefeed characters CR and LF (in C-language marks: '\n' and '\r').
* Any ascii data is marked in this document between quotes ("). So for example "1" is one byte, which numeric value is 49.
* Ascii-hex is a string of character bytes which contain only values between: 0-9 or a-f.
* Any binary number which is larger than one byte is stored in [network byte order](http://en.wikipedia.org/wiki/Endianness#Endianness_in_networking). Four byte values (32 bit numbers) can be converted using htonl function and two byte values (16 bits) with htons function.

**TCP messages**

* TCP messages are variable length frames of bytes.
* The form of one frame is **always**:

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | 11 | 1 | 1-N |
| message length (binary number) | String "DISTRIB2015" | Protocol mode "D" or "1" (or some extension defined value) | Command bytes (at least 1) |

* In the table above the first row contains the field lengths in bytes and the second row the description for each of the fields.
* In the frame fields:
  + First word (2 bytes) contains the frame length in bytes (including the length bytes). This is a two byte binary number in network byte order. The shortest possible frame is 15 bytes (length + protocol ID + mode + one byte command data).
  + Next eleven bytes are always the protocol identification string: "DISTRIB2015".
  + Mode byte is always an ascii character ("D" or "1" or implementation extension defined).
    - If operation mode is DEBUG then modebyte is "D"
    - If operation mode is AUTH1 then modebyte is "1"
    - Other values can be used if you implemnt extensions to the protocol

Server and client should process only those frames for which it knows the mode (unknown modes are an error condition).

* + Rest of the frame is the protocol command (documented in next section).
* If either end of the communication detects a frame, which does NOT follow this specification, the communications should be terminated immediately.

**Commands**

A command is a collection of bytes (at least one) and it is located as the last bytes of distrib protocol Frame.

**User authentication**

1. When TCP connection is established the client sends this command to the server:

|  |  |  |
| --- | --- | --- |
| 2 | 1 | 1-64 |
| "A0" | length | "username" |

1. Length-field tells how long the username data is (1-64 bytes). Please note that the username contains only the printable characters, NOT for example a terminating null byte.
2. The server will generate a 32 bit random number (AUTH1), which is sent as a challenge value to the client in network byte order (in DEBUG mode this number is always constant: 123456).

|  |  |
| --- | --- |
| 2 | 4 |
| "A1" | 32 bit binary number |

1. Client will calculate, using user password and the challenge number, a [SHA256-hash](http://en.wikipedia.org/wiki/SHA_hash_functions) value and send it to the server as a 64 byte ascii-hex value (AUTH1). (In DEBUG mode this string always contains 64 bytes of character 'f').

|  |  |
| --- | --- |
| 2 | 64 |
| "A2" | "SHA256-hash string in ascii-hex" |

1. After these [challenge-response](http://en.wikipedia.org/wiki/CRAM-MD5) commands, the server will decide if the connection is accepted (in DEBUG mode connection is always accepted). In accepted connection the status byte is zero, any other value means an unaccepted connection. Accepted connections returns also a Session Identification number chosen by the server (32 bit number in network byteorder). In failed connection and DEBUG mode this number is always 654321.

|  |  |  |
| --- | --- | --- |
| 2 | 1 | 4 |
| "A3" | status byte | SID |

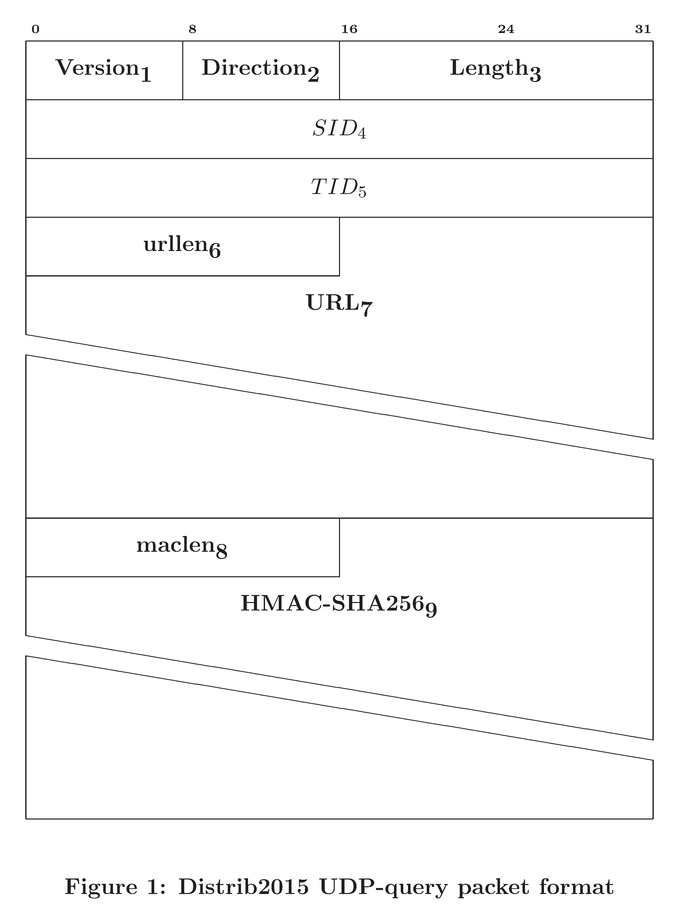
If any of the commands arrive in wrong order or contain illegal data, the connection should be terminated.

After exchanging the authentication frames, both endpoints terminate the TCP-connection and continue traffic using UDP-messages.

**UDP-messages**

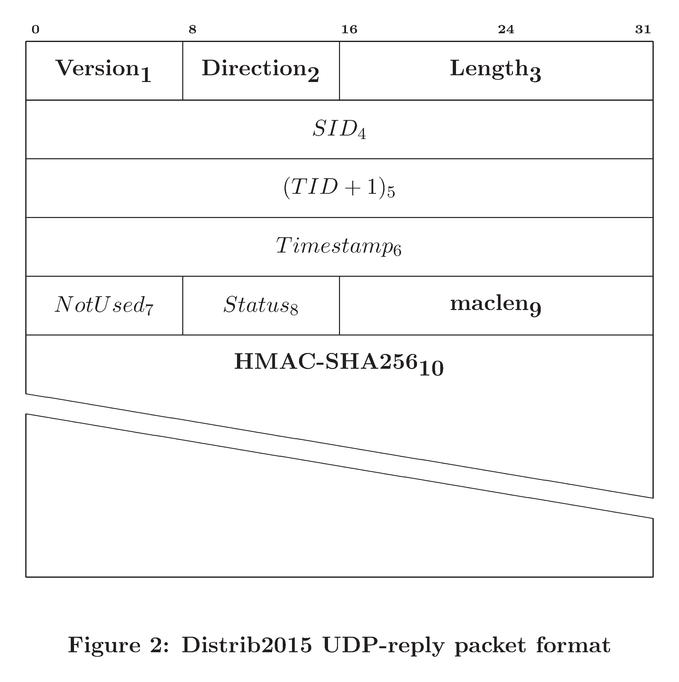
Size for any UDP packet in this protocol must not exceed 2000 bytes.

**Query packet**



|  |  |  |  |
| --- | --- | --- | --- |
| Field | Name | Length (in bytes) | Description |
| 1 | Version | 1 | Packet identification. "D" in debug mode and "1" in AUTH1 mode (or an extension defined value). |
| 2 | Direction | 1 | Byte "S" to mark, that the packet is going to the server. |
| 3 | Length | 2 | 16 bit number in network byteorder, which tells the length of the packet payload (Fields 1-7, NOT including HMAC fields 8 and 9). |
| 4 | SID | 4 | A 32 bit number in network byteorder. This *Session ID* is provided by the server in the authentication phase and it identifies the client to the server (server should only reply to correctly authenticated clients). |
| 5 | TID | 4 | A 32 bit number in network byteorder. This *Transaction ID* is selected by the client to identify this query. |
| 6 | urllen | 2 | 16 bit number in network byteorder, which tells the length of the URL query string (field 6). This length is the number of characters in the query. |
| 7 | URL | N | Characters of the URL query. Only printable characters are part of the data. No terminating zero characters or similar values are allowed. |
| 8 | maclen | 2 | 16 bit number in network byteorder, which tells the length of the HMAC value (field 8). In DEBUG mode length is always 1. |
| 9 | HMAC-SHA256 | N | A HMAC-SHA256 signature calculated from the fields 1 to 6 and stored in ascii-hex format. In DEBUG mode this field is always byte "D". |

**Reply packet**



|  |  |  |  |
| --- | --- | --- | --- |
| Field | Name | Length (in bytes) | Description |
| 1 | Version | 1 | Packet identification. "D" in debug mode and "1" in AUTH1 mode (or an extension defined value). |
| 2 | Direction | 1 | Byte "C" to mark, that the packet is going to the client. |
| 3 | Length | 2 | 16 bit number in network byteorder, which tells the length of the packet payload (Fields 1-8, NOT including HMAC fields 9 and 10). |
| 4 | SID | 4 | Session ID. |
| 5 | TID + 1 | 4 | 32 bit number in network byteorder. This value is the clients TID incremented by 1. |
| 6 | Timestamp | 4 | Servers UNIX-timestamp as a 32 bit number in network byteorder. |
| 7 | NotUsed | 1 | This field is not used (can contain any byte). |
| 8 | Status | 1 | Server reply to the URL query. Zero means OK and any other byte denotes some kind of ERROR condition. |
| 9 | maclen | 2 | 16 bit number in network byteorder, which tells the length of the HMAC value (field 8). In DEBUG mode length is always 1. |
| 10 | HMAC-SHA256 | N | A HMAC-SHA256 signature calculated from the fields 1 to 7 and stored in ascii-hex format. In DEBUG mode this field is always byte "D". |

**Please note**

* This protocol is not completely formally specified. This choice is intentional - in real world protocols you'll find huge differences in these kinds of specifications (for better and worse compared to this one).
* Internet protocol implementors have a [mantra](http://tools.ietf.org/html/rfc2360#section-2.9):

*be conservative in what you send and be liberal in what you accept*

In other words: always follow the protocol in your own code, but always be prepared to receive any kind of data from the network. E.g. if the protocol states that some field can only be 255 bytes in maximum, your code should not crash if you happen to receive 420000 bytes from the network...