class Auth::SCRAM

Authentication using SCRAM

Table of Contents


```
unit package Auth;
class SCRAMM::Client { ... }
```

Synopsis

Read and writable attributes

Defined as

```
has Int $.c-nonce-size is rw = 24;
has Str $.c-nonce is rw;
```

Control the generation of a nonce yourself. When not set, the class will makeup one of the default length of 24 octets and encode in base64.

```
$!c-nonce = encode-base64(
  Buf.new((for ^$!c-nonce-size { (rand * 256).Int })),
  :str
);


has Str $.reserved-mext is rw;
has Hash $.extensions is rw = %();
```

...

```
has Int $.s-nonce-size is rw = 24;
has Str $.s-nonce is rw;
```

...

All these attributes can be set before scram authentication is started.

Methods

new

Defined as

```
submethod BUILD (
  Str:D :$username!,
  Str:D :$password!,

  Callable :$CGH = &sha1,
  Str :$authzid,
  :$client-side,
  :$server-side,
)
```

Initialize the process. The Cryptographic Hash function $CGH is by default set to SHA1. The authorization id($authzid) is needed when you want things done with the privileges of someone else. The client-side and server-side are objects with methods called by the SCRAM methods. Only one of the client or server object can be defined.

Client side object

The purpose of the client object is to start communication with the server using the client first message. Then the server response is returned to the scram process for further calculations. Then the second step must be performed to finalize the authetication.

The client side object must provide the following methods:

```
method client-first ( Str:D $client-first-message --> Str )
```

This method is called by the scram process to deliver the client first message to the server. For further information please take a look here at rfc 5802. The method must return the server response in what is called server first message.

```
method client-final ( Str:D $client-final-message --> Str )
```

This method is called like above to send the final message to the server which in turn will return the server final message. This must be returned to the scram procedures.

```
method error ( Str:D $message --> Str )
```

When any part of the scram process fails, an error is given to the error method. This method must handle the cleanup of the servers connection because after an error, the cleanup method, if defined, will not be called in this case.

The following methods are optional:

```
method mangle-password (
  Str :$username,
  Str :$password,
  Str :$!authzid
  --> Buf
)
```

By default the password is mangled or hashed by the following statement

```
my Buf $mangled-password = Buf.new($!password.encode);
```

Sometimes the server asks for more elaborate methods such as with the MongoDB server. This result is used to create a salted password using pbkdf2. See also rfc 2898.

```
method cleanup (  )
```

Method called at the end of the authentication which enables the class object to finish the communication with the server or do other final tasks. The method is not called when errors are encountered.

Server side object

The server object must perform the server side role in this authentication process. This means that the first step is to wait for a specific moment that a client wants to start the authentication process. Normally a server sets up a socket on which it listens to. Every incoming communication is processed for commands. The client sends the server a command to start authetication with the client first message as acompanying data to the server after which the server object must process this message. The result will be the server first message and is send back to the client, etcetera.

The server side object must provide the following methods:

...

skip-saslprep

Defined as

```
method skip-saslprep ( Bool:D :$skip )
```

Call this before starting authentication with start-scram. The username, authorization id and password must be processed before authention is started. When old fashion ASCII letters and digits are used, no conversion is needed but it is when utf characters are used. The preparation is performed by default.

start-scram

Defined as

```
method start-scram( Str :$client-first-message --> Str ) {
```

Start authentication. An error message is returned when an error is encountered. This is the same error sent to the error() method mentioned above. When successful, it returns an empty string (''). $client-first-message must be defined when a server object is provided to the new() method.