

NAME

sasl_server_new - Cyrus SASL documentation

SYNOPSIS

```
#include <sasl/sasl.h>
```

```
int sasl_server_new(const char *service,  
                  const char *serverFQDN,  
                  const char *user_realm,  
                  const char *iplocalport,  
                  const char *ipremoteport,  
                  const sasl_callback_t *callbacks,  
                  unsigned flags,  
                  sasl_conn_t ** pconn);
```

DESCRIPTION

```
int sasl_server_new(const char *service,  
  
const char *serverFQDN,  
  
const char *user_realm,  
  
const char *iplocalport,  
  
const char *ipremoteport,  
  
const sasl_callback_t *callbacks,  
  
unsigned flags,  
  
sasl_conn_t ** pconn);
```

sasl_server_new() creates a new SASL context. This context will be used for all SASL calls for one connection. It handles both authentication and integrity/encryption layers after authentication.

Parameters

- ⊕ **service** - is the registered name of the service (usually the protocol name) using SASL (e.g. "imap").

- ⊕ **serverFQDN** - is the fully qualified server domain name. NULL means use `gethostname()`. This is useful for multi-homed servers.
- ⊕ **user_realm** - is the domain of the user agent. This is usually not necessary (NULL is default)
- ⊕ **iplocalport** -

is the IP and port of the local side of the connection, or NULL. If `iplocalport` is NULL it will disable mechanisms that require IP address information. This strings must be in one of the following formats:
 - ⊕ "a.b.c.d;port" (IPv4),
 - ⊕ "e:f:g:h:i:j:k:l;port" (IPv6), or
 - ⊕ "e:f:g:h:i:j:a.b.c.d;port" (IPv6)
- ⊕ **ipremoteport** - is the IP and port of the remote side of the connection, or NULL (see `iplocalport`)
- ⊕ **flags** - are connection flags (see below)
- ⊕ **pconn** - is a pointer to the connection context allocated by the library. This structure will be used for all future SASL calls for this connection.

Connection flags

Flags that may be passed to `sasl_server_new()`:

- ⊕ **SASL_SUCCESS_DATA**: The protocol supports a server-last send

⊕

SASL_NEED_PROXY: Force the use of a mechanism that supports an authorization id that is not the authentication id.

RETURN VALUE

SASL callback functions should return SASL return codes. See `sasl.h` for a complete list. **SASL_OK** indicates success.

Other return codes indicate errors and should either be handled or the authentication session should be quit.

SEE ALSO

RFC 4422, *saslman:sasl(3)*, *sasl_server_init(3)*, *sasl_server_start(3)*, *sasl_server_step(3)*, *sasl_setprop(3)*, *sasl_errors(3)*

AUTHOR

The Cyrus Team

COPYRIGHT

1993-2016, The Cyrus Team