NAME
    SSL_CTX_set1_sigalgs, SSL_set1_sigalgs, SSL_CTX_set1_sigalgs_list, SSL_set1_sigalgs_list,
    SSL_CTX_set1_client_sigalgs, SSL_set1_client_sigalgs, SSL_CTX_set1_client_sigalgs_list,
    SSL_set1_client_sigalgs_list - set supported signature algorithms

SYNOPSIS
    #include <openssl/ssl.h>

    long SSL_CTX_set1_sigalgs(SSL_CTX *ctx, const int *slist, long slistlen);
    long SSL_set1_sigalgs(SSL *ssl, const int *slist, long slistlen);
    long SSL_CTX_set1_sigalgs_list(SSL_CTX *ctx, const char *str);
    long SSL_set1_sigalgs_list(SSL *ssl, const char *str);

    long SSL_CTX_set1_client_sigalgs(SSL_CTX *ctx, const int *slist, long slistlen);
    long SSL_set1_client_sigalgs(SSL *ssl, const int *slist, long slistlen);
    long SSL_CTX_set1_client_sigalgs_list(SSL_CTX *ctx, const char *str);
    long SSL_set1_client_sigalgs_list(SSL *ssl, const char *str);

DESCRIPTION
    SSL_CTX_set1_sigalgs() and **SSL_set1_sigalgs()** set the supported signature algorithms for **ctx** or **ssl**.
    The array **slist** of length **slistlen** must consist of pairs of NIDs corresponding to digest and public key
    algorithms.

    SSL_CTX_set1_sigalgs_list() and **SSL_set1_sigalgs_list()** set the supported signature algorithms for
    ctx or **ssl**. The **str** parameter must be a null terminated string consisting of a colon separated list of
    elements, where each element is either a combination of a public key algorithm and a digest separated
    by +, or a TLS 1.3-style named SignatureScheme such as rsa_pss_pss_sha256.

    SSL_CTX_set1_client_sigalgs(), **SSL_set1_client_sigalgs()**, **SSL_CTX_set1_client_sigalgs_list()** and
    SSL_set1_client_sigalgs_list() set signature algorithms related to client authentication, otherwise they
    are identical to **SSL_CTX_set1_sigalgs()**, **SSL_set1_sigalgs()**, **SSL_CTX_set1_sigalgs_list()** and
    SSL_set1_sigalgs_list().

    All these functions are implemented as macros. The signature algorithm parameter (integer array or
    string) is not freed: the application should free it, if necessary.

NOTES
    If an application wishes to allow the setting of signature algorithms as one of many user configurable
    options it should consider using the more flexible SSL_CONF API instead.

The signature algorithms set by a client are used directly in the supported signature algorithm in the client hello message.

The supported signature algorithms set by a server are not sent to the client but are used to determine the set of shared signature algorithms and (if server preferences are set with SSL_OP_CIPHER_SERVER_PREFERENCE) their order.

The client authentication signature algorithms set by a server are sent in a certificate request message if client authentication is enabled, otherwise they are unused.

Similarly client authentication signature algorithms set by a client are used to determined the set of client authentication shared signature algorithms.

Signature algorithms will neither be advertised nor used if the security level prohibits them (for example SHA1 if the security level is 4 or more).

Currently the NID_md5, NID_sha1, NID_sha224, NID_sha256, NID_sha384 and NID_sha512 digest NIDs are supported and the public key algorithm NIDs EVP_PKEY_RSA, EVP_PKEY_RSA_PSS, EVP_PKEY_DSA and EVP_PKEY_EC.

The short or long name values for digests can be used in a string (for example "MD5", "SHA1", "SHA224", "SHA256", "SHA384", "SHA512") and the public key algorithm strings "RSA", "RSA-PSS", "DSA" or "ECDSA".

The TLS 1.3 signature scheme names (such as "rsa_pss_pss_sha256") can also be used with the **_list** forms of the API.

The use of MD5 as a digest is strongly discouraged due to security weaknesses.

RETURN VALUES

All these functions return 1 for success and 0 for failure.

EXAMPLES

Set supported signature algorithms to SHA256 with ECDSA and SHA256 with RSA using an array:

```
const int slist[] = {NID_sha256, EVP_PKEY_EC, NID_sha256, EVP_PKEY_RSA};
```

```
SSL_CTX_set1_sigalgs(ctx, slist, 4);
```

Set supported signature algorithms to SHA256 with ECDSA and SHA256 with RSA using a string:

```
SSL_CTX_set1_sigalgs_list(ctx, "ECDSA+SHA256:RSA+SHA256");
```

SEE ALSO

ssl(7), **SSL_get_shared_sigalgs**(3), **SSL_CONF_CTX_new**(3)

COPYRIGHT

Copyright 2015-2018 The OpenSSL Project Authors. All Rights Reserved.

Licensed under the Apache License 2.0 (the "License"). You may not use this file except in compliance with the License. You can obtain a copy in the file LICENSE in the source distribution or at <https://www.openssl.org/source/license.html>.