## NAME

SSL_get_shared_sigalgs, SSL_get_sigalgs - get supported signature algorithms

## SYNOPSIS

```
#include <openssl/ssl.h>

int SSL_get_shared_sigalgs(SSL *s, int idx,
                           int *psign, int *phash, int *psignhash,
                           unsigned char *rsig, unsigned char *rhash);

int SSL_get_sigalgs(SSL *s, int idx,
                    int *psign, int *phash, int *psignhash,
                    unsigned char *rsig, unsigned char *rhash);
```

## DESCRIPTION

**SSL_get_shared_sigalgs()** returns information about the shared signature algorithms supported by peer **s**. The parameter **idx** indicates the index of the shared signature algorithm to return starting from zero. The signature algorithm NID is written to ***psign**, the hash NID to ***phash** and the sign and hash NID to ***psignhash**. The raw signature and hash values are written to ***rsig** and ***rhash**.

**SSL_get_sigalgs()** is similar to **SSL_get_shared_sigalgs()** except it returns information about all signature algorithms supported by **s** in the order they were sent by the peer.

## RETURN VALUES

**SSL_get_shared_sigalgs()** and **SSL_get_sigalgs()** return the number of signature algorithms or **0** if the **idx** parameter is out of range.

## NOTES

These functions are typically called for debugging purposes (to report the peer's preferences) or where an application wants finer control over certificate selection. Most applications will rely on internal handling and will not need to call them.

If an application is only interested in the highest preference shared signature algorithm it can just set **idx** to zero.

Any or all of the parameters **psign**, **phash**, **psignhash**, **rsig** or **rhash** can be set to **NULL** if the value is not required. By setting them all to **NULL** and setting **idx** to zero the total number of signature algorithms can be determined: which can be zero.

These functions must be called after the peer has sent a list of supported signature algorithms: after a

client hello (for servers) or a certificate request (for clients). They can (for example) be called in the certificate callback.

Only TLS 1.2, TLS 1.3 and DTLS 1.2 currently support signature algorithms. If these functions are called on an earlier version of TLS or DTLS zero is returned.

The shared signature algorithms returned by **SSL_get_shared_sigalgs()** are ordered according to configuration and peer preferences.

The raw values correspond to the on the wire form as defined by RFC5246 et al. The NIDs are OpenSSL equivalents. For example if the peer sent **sha256**(4) and **rsa**(1) then ***rhash** would be 4, ***rsign** 1, ***phash** NID_sha256, ***psig** NID_rsaEncryption and ***psighash** NID_sha256WithRSAEncryption.

If a signature algorithm is not recognised the corresponding NIDs will be set to **NID_undef**. This may be because the value is not supported, is not an appropriate combination (for example MD5 and DSA) or the signature algorithm does not use a hash (for example Ed25519).

## SEE ALSO

**SSL_CTX_set_cert_cb**(3), **ssl**(7)

## COPYRIGHT