NAME

SSL_set1_host, SSL_add1_host, SSL_set_hostflags, SSL_get0_peername - SSL server verification parameters

SYNOPSIS

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

 int SSL_set1_host(SSL *s, const char *hostname);
 int SSL_add1_host(SSL *s, const char *hostname);
 void SSL_set_hostflags(SSL *s, unsigned int flags);
 const char *SSL_get0_peername(SSL *s);
```

DESCRIPTION

These functions configure server hostname checks in the SSL client.

SSL_set1_host() sets the expected DNS hostname to **name** clearing any previously specified hostname. If **name** is NULL or the empty string, the list of hostnames is cleared and name checks are not performed on the peer certificate. When a nonempty **name** is specified, certificate verification automatically checks the peer hostname via **X509_check_host**(3) with **flags** as specified via **SSL_set_hostflags()**. Clients that enable DANE TLSA authentication via **SSL_dane_enable**(3) should leave it to that function to set the primary reference identifier of the peer, and should not call **SSL_set1_host()**.

SSL_add1_host() adds **name** as an additional reference identifier that can match the peer’s certificate. Any previous names set via **SSL_set1_host()** or **SSL_add1_host()** are retained, no change is made if **name** is NULL or empty. When multiple names are configured, the peer is considered verified when any name matches. This function is required for DANE TLSA in the presence of service name indirection via CNAME, MX or SRV records as specified in RFC7671, RFC7672 or RFC7673.

SSL_set_hostflags() sets the **flags** that will be passed to **X509_check_host**(3) when name checks are applicable, by default the **flags** value is 0. See **X509_check_host**(3) for the list of available flags and their meaning.

SSL_get0_peername() returns the DNS hostname or subject CommonName from the peer certificate that matched one of the reference identifiers. When wildcard matching is not disabled, the name matched in the peer certificate may be a wildcard name. When one of the reference identifiers configured via **SSL_set1_host()** or **SSL_add1_host()** starts with ".", which indicates a parent domain prefix rather than a fixed name, the matched peer name may be a sub-domain of the reference identifier. The returned string is allocated by the library and is no longer valid once the associated **ssl** handle is cleared or freed, or a renegotiation takes place. Applications must not free the return value.

SSL clients are advised to use these functions in preference to explicitly calling **X509_check_host**(3). Hostname checks may be out of scope with the RFC7671 **DANE-EE**(3) certificate usage, and the internal check will be suppressed as appropriate when DANE is enabled.

RETURN VALUES

SSL_set1_host() and **SSL_add1_host()** return 1 for success and 0 for failure.

SSL_get0_peername() returns NULL if peername verification is not applicable (as with RFC7671 **DANE-EE**(3)), or no trusted peername was matched. Otherwise, it returns the matched peername. To determine whether verification succeeded call **SSL_get_verify_result**(3).

EXAMPLES

Suppose "smtp.example.com" is the MX host of the domain "example.com". The calls below will arrange to match either the MX hostname or the destination domain name in the SMTP server certificate. Wildcards are supported, but must match the entire label. The actual name matched in the certificate (which might be a wildcard) is retrieved, and must be copied by the application if it is to be retained beyond the lifetime of the SSL connection.

```
 SSL_set_hostflags(ssl, X509_CHECK_FLAG_NO_PARTIAL_WILDCARDS);
 if (!SSL_set1_host(ssl, "smtp.example.com"))
     /* error */
 if (!SSL_add1_host(ssl, "example.com"))
     /* error */

 /* XXX: Perform SSL_connect() handshake and handle errors here */

 if (SSL_get_verify_result(ssl) == X509_V_OK) {
     const char *peername = SSL_get0_peername(ssl);

     if (peername != NULL)
         /* Name checks were in scope and matched the peername */
 }
```

SEE ALSO

ssl(7), **X509_check_host**(3), **SSL_get_verify_result**(3). **SSL_dane_enable**(3).

HISTORY

These functions were added in OpenSSL 1.1.0.

COPYRIGHT