

```

1  static OSStatus
2  SSLVerifySignedServerKeyExchange(SSLContext *ctx, bool isRsa, SSLBuffer
signedParams,
3                                     uint8_t *signature, UInt16 signatureLen)
4  {
5      OSStatus      err;
6      SSLBuffer     hashOut, hashCtx, clientRandom, serverRandom;
7      uint8_t       hashes[SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN];
8      SSLBuffer     signedHashes;
9      uint8_t       *dataToSign;
10     size_t         dataToSignLen;
11
12     signedHashes.data = 0;
13     hashCtx.data = 0;
14
15     clientRandom.data = ctx->clientRandom;
16     clientRandom.length = SSL_CLIENT_SRVR_RAND_SIZE;
17     serverRandom.data = ctx->serverRandom;
18     serverRandom.length = SSL_CLIENT_SRVR_RAND_SIZE;
19
20
21     if(isRsa) {
22         /* skip this if signing with DSA */
23         dataToSign = hashes;
24         dataToSignLen = SSL_SHA1_DIGEST_LEN + SSL_MD5_DIGEST_LEN;
25         hashOut.data = hashes;
26         hashOut.length = SSL_MD5_DIGEST_LEN;
27
28         if ((err = ReadyHash(&SSLHashMD5, &hashCtx)) != 0)
29             goto fail;
30         if ((err = SSLHashMD5.update(&hashCtx, &clientRandom)) != 0)
31             goto fail;
32         if ((err = SSLHashMD5.update(&hashCtx, &serverRandom)) != 0)
33             goto fail;
34         if ((err = SSLHashMD5.update(&hashCtx, &signedParams)) != 0)
35             goto fail;
36         if ((err = SSLHashMD5.final(&hashCtx, &hashOut)) != 0)
37             goto fail;
38     }
39     else {
40         /* DSA, ECDSA - just use the SHA1 hash */
41         dataToSign = &hashes[SSL_MD5_DIGEST_LEN];
42         dataToSignLen = SSL_SHA1_DIGEST_LEN;
43     }
44
45     hashOut.data = hashes + SSL_MD5_DIGEST_LEN;
46     hashOut.length = SSL_SHA1_DIGEST_LEN;
47     if ((err = SSLFreeBuffer(&hashCtx)) != 0)
48         goto fail;
49
50     if ((err = ReadyHash(&SSLHashSHA1, &hashCtx)) != 0)
51         goto fail;
52     if ((err = SSLHashSHA1.update(&hashCtx, &clientRandom)) != 0)
53         goto fail;
54     if ((err = SSLHashSHA1.update(&hashCtx, &serverRandom)) != 0)
55         goto fail;

```

```
56     if ((err = SSLHashSHA1.update(&hashCtx, &signedParams)) != 0)
57         goto fail;
58     goto fail;
59     if ((err = SSLHashSHA1.final(&hashCtx, &hashout)) != 0)
60         goto fail;
61
62     err = sslRawVerify(ctx,
63                       ctx->peerPubKey,
64                       dataToSign,          /* plaintext */
65                       dataToSignLen,      /* plaintext length */
66                       signature,
67                       signatureLen);
68     if(err) {
69         sslErrorLog("SSLDecodeSignedServerKeyExchange: sslRawVerify "
70                   "returned %d\n", (int)err);
71         goto fail;
72     }
73
74 fail:
75     SSLFreeBuffer(&signedHashes);
76     SSLFreeBuffer(&hashCtx);
77     return err;
78
79 }
```