Network Working Group	N. Mavrogiannopoulos
Internet-Draft	KUL
Intended status: Standards Track	June 08, 2011
Expires: December 10, 2011	

Using transport layer security (TLS) with DSA and ECDSA ciphersuites
draft-mavrogiannopoulos-tls-dss-01

Abstract

This memo clarifies the usage of the digital signature algorithm (DSA) with extended key lengths, in the transport layer security (TLS) protocol earlier than 1.2, and makes clarifications for the usage of DSA and its elliptic curves equivalent (ECDSA) in TLS 1.2.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

The TLS protocols support the DSA algorithm even from its first incarnation in [RFC2246]. However the latest DSA publication from NIST at [DSS], suggests some changes that do not straightforwardly apply to the TLS protocols.
In this document we describe the differences on the new DSS algorithms[DSS], and define a profile for TLS implementations.

2. Terminology

This document uses the same notation and terminology used in the TLS Protocol specification [RFC5246].
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. DSA in FIPS-186-3

In this section we discuss the differences between the old DSS publication [OLDDSS] and the new one [DSS], that justify the need for a TLS profile.
DSA parameters include a prime modulus p and a prime divisor of p-1 called q. In [OLDDSS] the bit length of p was fixed to 1024 bits, the length of q to 160 bits and the underlying hash algorithm was fixed to SHA-1. However the DSA algorithm in [DSS] allows more lengths for p and

q, as well different hash algorithms, than the older version which is currently referred by TLS protocols.
The new document relies on the "bits of security" term defined in [SP800-57], and recommends that security strength of the hash algorithm matches the security strength of other DSA parameters. It is required either the bits of the hash algorithm to match the bits of length of q (N), or if the hash size is larger, only the N leftmost bits of the hash output are being used. The corresponding mappings are shown in Table 1 and Table 2.

Hash algorithm	Hash size	Bits of security
SHA-1	160	80
SHA-224	224	112
SHA-256	256	128
SHA-384	384	192
SHA-512	512	256

Length of p (L)	Length of q (N)	Bits of security	Matching hash algorithms
1024	160	80	SHA-1, SHA-224, SHA-256, SHA-384, SHA-512
2048	224	112	SHA-224, SHA-256, SHA-384, SHA-512
2048	256	(112,128)	SHA-256, SHA-384, SHA-512
3072	256	128	SHA-256, SHA-384, SHA-512

4. The SSL 3.0, TLS 1.0 and 1.1 protocols

4.1. DSA

The SSL 3.0, TLS 1.0 and 1.1 protocols support ciphersuites that utilize the DSA algorithms for signing. The digital signatures are used for the "Server key exchange" and "Certificate verify" messages. In those messages there is no indication of the signature algorithm used, thus the selection is implicit. The signature contained in both messages is defined, for the DSA algorithm, as:

```
select (SignatureAlgorithm)
{
    case dsa:
        digitally-signed struct {
            opaque sha_hash[20];
        };
} Signature;
```

This structure refers to the DSA algorithm with L=1024 and N=160, but this is not an explicit requirement of those protocols and several existing implementations are using the SHA-1 algorithm for all DSA key sizes. For this reason it is RECOMMENDED not to use DSA keys of sizes other than L=1024 and N=160 in combination with those protocols.
If however keys of sizes larger than L=1024 and N=160 have to be used, then the SHA-1 algorithm has to be used.

4.2. ECDSA

For TLS negotiation to proceed smoothly when an ECDSA enabled ciphersuite is negotiated both parties must agree to a curve. However given that [RFC4492] lists a very large number of curves but doesn't recommend any, it is unclear which curves should be used in certificates for TLS to achieve interoperability.
To improve interoperability implementations SHOULD use certificates with curves restricted to the recommended by [RFC5480]. Those are summarized in Table 3.

Curve
secp224r1
secp256r1
secp384r1
secp521r1

5. The TLS protocol 1.2

This version of the protocol also requires signatures for the "Server key exchange" and "Certificate verify" messages. However in this version signature algorithm negotiation is explicit via the "Signature algorithms" extension. The signature used is as below:

```
struct {
   SignatureAndHashAlgorithm algorithm;
   opaque signature<0..2^16-1>;
} DigitallySigned;
```

It is however desirable for interoperability reasons to restrict the available options. This would allow constrained clients to support only the required algorithms, and servers that do not cache all messages up to "Certificate verify" in order to calculate the signature, to carry a single hash state instead.

5.1. DSA

In this case a signature algorithm should be selected that matches the requirements as in Table 4. Implementations SHOULD select the algorithms shown on that table.

Length of p in bits	Length of q in bits	Hash algorithm	Hash size	Truncated hash size
1024	160	SHA-1	20	20
2048	224	SHA-256	32	28
2048	256	SHA-256	32	32
3072	256	SHA-256	32	32

Note: When the hash size does not match the length of q, then only the leftmost bytes of the hash, that match the length of q, are used. This is indicated in the "Truncated hash size" column of the table.

5.1.1. Parameters not allowed by DSS

TLS implementations MUST NOT support parameter lengths not allowed by [DSS]. If illegal parameters are encountered, the handshake should be aborted using an "illegal_parameter" alert.

5.2. ECDSA

The signature hash algorithm SHOULD be selected in way that matches the requirements of Table 5. Also implementations SHOULD use certificates with curves restricted to the recommended by [RFC5480]. Those are summarized in Table 3.

ECDSA key size	Hash algorithm	Hash size	Truncated hash size
192	SHA-256	32	24
224	SHA-256	32	28
256	SHA-256	32	32
384	SHA-384	48	48
512	SHA-512	64	64

Note: As with DSA, when the hash size does not match the curve key size, only the leftmost bytes of the hash are used. This size is shown in the "Truncated hash size" column of the table.

6. Security Considerations

When DSA keys are being used in connections that involve the SSL 3.0, TLS 1.0 or TLS 1.1 protocols then the entire connection security depends on the SHA-1 algorithm. This is about 80-bits of security irrespective of the sizes of the DSA keys.
All security considerations discussed in [RFC5246], apply to this document.

7. References

7.1. Normative References

[DSS]	NIST FIPS PUB 186-3, "Digital Signature Standard", National Institute of Standards and Technology, U.S. Department of Commerce , June 2009.
[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[RFC5480]	Turner, S., Brown, D., Yiu, K., Housley, R. and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.
[RFC4492]	Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C. and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.
[RFC5246]	Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

7.2. Informative References

[OLDDSS]	NIST FIPS PUB 186, "Digital Signature Standard", National Institute of Standards and Technology, U.S. Department of Commerce , May 1994.
[SP800-57]	NIST FIPS SP 800-57, "Recommendation for Key Management", National Institute of Standards and Technology, U.S. Department of Commerce , March 2007.
[RFC2246]	Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

Author's Address

Nikos Mavrogiannopoulos Mavrogiannopoulos ESAT/COSIC Katholieke Universiteit Leuven Kasteelpark Arenberg 10, bus 2446 Leuven-Heverlee, B-3001 Belgium EMail: nikos.mavrogiannopoulos@esat.kuleuven.be