# SPAKE2, a PAKE
## <draft-irtf-cfrg-spake2-02.txt>



Abstract

This Internet-Draft describes SPAKE2, a secure, efficient password based

key exchange protocol.

Table of Contents



## 1. Introduction

This document describes a means for two parties that share a password to derive a shared key. This method is compatible with any group, is computationally efficient, and has a strong security proof.

## 2. Definition of SPAKE2

### 2.1 Setup

Let G be a group in which the Diffie-Hellman problem is hard of order ph, with p a big prime and h a cofactor. We denote the operations in the group additively. Let H be a hash function from arbitrary strings to bit strings of a fixed length. Common choices for H are SHA256 or SHA512. We assume there is a representation of elements of G as byte strings: common choices would be SEC1 uncompressed for elliptic curve groups or big endian integers of a particular length for prime field DH.

|| denotes concatenation of strings. We also let len(S) denote the length of a string in bytes, represented as an eight-byte big-endian number.

We fix two elements M and N as defined in the table in this document for common groups, as well as a generator G of the group. G is specified in the document defining the group, and so we do not recall it here.

Let A and B be two parties. We will assume that A and B are also representations of the parties such as MAC addresses or other names (hostnames, usernames, etc). We assume they share an integer w. Typically w will be the hash of a user-supplied password, truncated and taken mod p. Protocols using this protocol must define the method used to compute w: it may be necessary to carry out normalization.

We present two protocols below. Note that it is insecure to use the same password with both protocols, this MUST NOT be done.

### 2.2 SPAKE2

A picks x randomly and uniformly from the integers in [0,ph) divisible by h, and calculates X=xG and T=wM+X, then transmits T to B.

B selects y randomly and uniformly from the integers in [0,ph), divisible by h and calculates Y=yG, S=wN+Y, then transmits S to A.

Both A and B calculate a group element K. A calculates it as x(S-wN), while B calculates it as y(T-wM). A knows S because it has received it, and likewise B knows T.

This K is a shared secret, but the scheme as described is not secure. It is essential to combine K with the values transmitted and received via a hash function to have a secure protocol. If higher-level protocols prescribe a method for doing so, that SHOULD be used. Otherwise we can compute K' as H(len(A)||A||len(B)||B||len(S)||S|| len(T)||T||len(K)||K) and use K' as the key.

### 2.3 SPAKE2+

This protocol and security proof appear in [TDH]. We use the same setup as for SPAKE2, except that we have two secrets, w0 and w1. The server, here Bob, stores L=w1*g and w0.

When executing SPAKE2+, Alice selects x uniformly at random from the numbers in the range [0, ph) divisible by h, and lets X=xG+w0*M, then transmits X to Bob. Bob selects y uniformly at random from the numbers in [0, ph) divisible by h, then computes Y=yG+w0*N, and transmits it to Alice.

Alice computes Z as x(Y-w0*N), and V as w1(Y-w0*N). Bob computes Z as y(X-w0*M) and V as yL. Both share Z and V as common keys. It is essential that both Z and V be used in combination with the transcript to derive the keying material. For higher-level protocols without sufficient transcript hashing, let K' be H(len(A)||A||len(B)||B||len(X)||X||len(Y)||Y||len(Z)||Z||len(V)||V) and use K' as the established key.

## 3. Table of points for common groups

Every curve presented in the table below has an OID from [OID]. We construct a string using the OID and the needed constant, for instance "1.3.132.0.35 point generation seed (M)" for P-512.  This string is turned into an infinite sequence of bytes by hashing with SHA256, and hashing that output again to generate the next 32 bytes, and so on.

The initial segment of bytes of length equal to that of an encoded

group element is taken, and is then formatted as required for the group. In the case of Weierstrass points, this means setting the first byte to 0x02. For Ed25519 style formats this means taking all the bytes as the representation of the group element. This string of bytes is then interpreted as a point in the group. If this is impossible, then the next non-overlapping segment of sufficient length is taken.

These bytes appear in the format commonly associated with each group.

For P256:

M =
02886e2f97ace46e55ba9dd7242579f2993b64e16ef3dcab95afd497333d8fa12f

N =
03d8bbd6c639c62937b04d997f38c3770719c629d7014d49a24b4f98baa1292b49

For P384:

M =
030ff0895ae5ebf6187080a82d82b42e2765e3b2f8749c7e05eba366434b363d3dc
36f15314739074d2eb8613fceec2853

N =
02c72cf2e390853a1c1c4ad816a62fd15824f56078918f43f922ca21518f9c543bb
252c5490214cf9aa3f0baab4b665c10

For P521:

M =
02003f06f38131b2ba2600791e82488e8d20ab889af753a41806c5db18d37d85608
cfae06b82e4a72cd744c719193562a653ea1f119eef9356907edc9b56979962d7aa

N =
0200c7924b9ec017f3094562894336a53c50167ba8c5963876880542bc669e494b25
32d76c5b53dfb349fdf69154b9e0048c58a42e8ed04cef052a3bc349d95575cd25

## 4. Security Considerations

A security proof of SPAKE2 for prime order groups is found in [REF]. Note that the choice of M and N is critical for the security proof. The generation method specified in this document is designed to eliminate concerns related to knowing discrete logs of M and N.

SPAKE2+ appears in [TDH], along with proof.

There is no key-confirmation as this is a one round protocol. It is

expected that a protocol using this key exchange mechanism provides key confirmation separately if desired.

Elements should be checked for group membership: failure to properly validate group elements can lead to attacks. In particular it is essential to verify that received points are valid compressions of points on an elliptic curve when using elliptic curves. It is not necessary to validate membership in the prime order subgroup: the multiplication by cofactors eliminates this issue.

The choices of random numbers should be uniformly at random. Note that to pick a random multiple of h in [0, ph) one can pick a random integer in [0,p) and multiply by h. Reuse of ephemerals results in dictionary attacks and should not be done.

SPAKE2 does not support augmentation. As a result, the server has to store a password equivalent. This is considered a significant drawback, and so SPAKE2+ also appears in this document.

As specified the shared secret K is not suitable for use as a shared key. It should be passed to a hash function along with the public values used to derive it and the party identities to avoid attacks. In protocols which do not perform this separately, the value denoted K' should be used instead. This is critical for security.

## 5. IANA Considerations

No IANA action is required.

## 6. Acknowledgments

Special thanks to Nathaniel McCallum for generation of test vectors. Thanks to Mike Hamburg for advice on how to deal with cofactors. Greg Hudson suggested addition of warnings on the reuse of x and y. Thanks to Fedor Brunner and the members of the CFRG for comments and advice. Trevor Perrin informed me of SPAKE2+.

## 7. References

[REF] Abdalla, M. and Pointcheval, D. Simple Password-Based Encrypted Key Exchange Protocols. Appears in A. Menezes, editor. Topics in Cryptography-CT-RSA 2005, Volume 3376 of Lecture Notes in Computer Science, pages 191-208, San Francisco, CA, US Feb. 14-18, 2005. Springer-Verlag, Berlin, Germany.

[TDH] Cash, D. Kiltz, E. and Shoup, V. The Twin-Diffie Hellman Problem and Applications. Advances in Cryptology--EUROCRYPT 2008. Volume 4965 of Lecture notes in Computer Science, pages 127-145.

Springer-Verlag, Berlin, Germany.

[OID] Turner, S. and D. Brown and K. Yiu and R. Housley and T. Polk. Elliptic Curve Cryptography Subject Public Key Information. RFC 5480. March 2009.

## Author Addresses

Watson Ladd
watsonbladd@gmail.com
Berkeley, CA