

Internet Draft  
<[draft-irtf-cfrg-spake2-01.txt](#)>  
Category: Informational  
Expires 20 August 2015

W. Ladd  
UC Berkeley

16 February 2015

**SPAKE2, a PAKE**  
<[draft-irtf-cfrg-spake2-01.txt](#)>

Status of this Memo

Distribution of this memo is unlimited.

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/1id-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on date.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

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

based key exchange protocol.

Table of Contents

- [1. Introduction](#) .....[3](#)
- [2. Definition of SPAKE2](#).....[3](#)
- [3. Table of points](#) .....[4](#)
- [4. Security considerations](#) .....[5](#)
- [5. IANA actions](#) .....[5](#)
- [6. Acknowledgements](#).....[5](#)
- [7. References](#).....[5](#)

**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**

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.

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  $H(\text{len}(A)||A||\text{len}(B)||B||\text{len}(S)||S||\text{len}(T)||T||\text{len}(K)||K)$  and use  $K'$  as the key.

Note that the calculation of  $S=wN+yG$  may be performed more efficiently than by two separate scalar multiplications via Strauss's algorithm.

### 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 is taken, and the first byte has all bits but the low-order one cleared, and the second-order bit set. This string of bytes is then interpreted as a SEC1 compressed point. If this is impossible, then the next non-overlapping segment of sufficient length is taken.

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 for prime order groups is found in [\[REF\]](#). Note that the choice of M and N is critical for the security proof. The points in the table of points were generated via the use of a hash function to mitigate this risk.

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  $h$ .

This PAKE does not support augmentation. As a result, the server has to store a password equivalent. This is considered a significant drawback.

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. Thanks to Fedor Brunner and the members of the CFRG for comments and advice.

## 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.

[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