

Internet-Draft  
Expires: May 12, 2016

D. Bider  
Bitwise Limited  
November 12, 2015

Use of RSA Keys with SHA-2 256 and 512 in Secure Shell (SSH)  
draft-rsa-dsa-sha2-256-03.txt

## Abstract

This memo defines an algorithm name, public key format, and signature format for use of RSA keys with SHA-2 512 for server and client authentication in SSH connections.

## Status

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/lid-abstracts.html>

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

## Copyright

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. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## 1. Overview and Rationale

Secure Shell (SSH) is a common protocol for secure communication on the Internet. In [\[RFC4253\]](#), SSH originally defined the signature methods "ssh-rsa" for server and client authentication using RSA with SHA-1, and "ssh-dss" using 1024-bit DSA and SHA-1.

A decade later, these signature methods are considered deficient. For US government use, NIST has disallowed 1024-bit RSA and DSA, and use of SHA-1 for signing [\[800-131A\]](#).

This memo defines a new algorithm name allowing for interoperable use of RSA keys with SHA-2 256 and SHA-2 512, and a mechanism for servers to inform SSH clients of signature algorithms they support and accept.

### 1.1. Requirements Terminology

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 [\[RFC2119\]](#).

## 2. Public Key Algorithms

This memo adopts the style and conventions of [\[RFC4253\]](#) in specifying how the use of a signature algorithm is indicated in SSH.

The following new signature algorithms are defined:

|              |             |      |             |
|--------------|-------------|------|-------------|
| rsa-sha2-256 | RECOMMENDED | sign | Raw RSA key |
| rsa-sha2-512 | OPTIONAL    | sign | Raw RSA key |

These signature algorithms are suitable for use both in the SSH transport layer [\[RFC4253\]](#) for server authentication, and in the authentication layer [\[RFC4252\]](#) for client authentication.

Since RSA keys are not dependent on the choice of hash function, both new algorithms reuse the public key format of the existing "ssh-rsa" algorithm as defined in [\[RFC4253\]](#):

```
string    "ssh-rsa"
```

```
mpint      e
mpint      n
```

All aspects of the "ssh-rsa" format are kept, including the encoded string "ssh-rsa", in order to allow users' existing RSA keys to be used with the new signature formats, without requiring re-encoding, or affecting already trusted key fingerprints.

Signing and verifying using these algorithms is performed according to the RSASSA-PKCS1-v1\_5 scheme in [\[RFC3447\]](#) using SHA-2 [\[FIPS-180-4\]](#) as hash; MGF1 as mask function; and salt length equal to hash size.

Bider

[Page 2]

---

Internet-Draft

RSA Keys with SHA-2 in SSH

November 2015

For the algorithm "rsa-sha2-256", the hash used is SHA-2 256.  
For the algorithm "rsa-sha2-512", the hash used is SHA-2 512.

The resulting signature is encoded as follows:

```
string      "rsa-sha2-256" / "rsa-sha2-512"
string      rsa_signature_blob
```

The value for 'rsa\_signature\_blob' is encoded as a string containing S - an octet string which is the output of RSASSA-PKCS1-v1\_5, of length equal to the length in octets of the RSA modulus.

### [2.1.](#) Use for server authentication

To express support and preference for one or both of these algorithms for server authentication, the SSH client or server includes one or both algorithm names, "rsa-sha2-256" and/or "rsa-sha2-512", in the name-list field "server\_host\_key\_algorithms" in the SSH\_MSG\_KEXINIT packet [\[RFC4253\]](#). If one of the two host key algorithms is negotiated, the server sends an "ssh-rsa" public key as part of the negotiated key exchange method (e.g. in SSH\_MSG\_KEXDH\_REPLY), and encodes a signature with the appropriate signature algorithm name - either "rsa-sha2-256", or "rsa-sha2-512".

### [2.2.](#) Use for client authentication

To use this algorithm for client authentication, the SSH client sends an SSH\_MSG\_USERAUTH\_REQUEST message [\[RFC4252\]](#) encoding the "publickey" method, and encoding the string field "public key algorithm name" with the value "rsa-sha2-256" or "rsa-sha2-512". The "public key blob" field encodes the RSA public key using the "ssh-rsa" algorithm name.

The signature field, if present, encodes a signature using an algorithm name that matches the SSH authentication request - either "rsa-sha2-256", or "rsa-sha2-512".

### [3.](#) Discovery of signature algorithms supported by servers

When a public key format can use multiple signature algorithms, it can be useful for a mechanism to exist which a client can use to discover signature algorithms accepted by a server for user authentication without resorting to trial and error in authentication requests.

Such a mechanism is defined in [\[SSH-EXT-INFO\]](#), which describes general purpose extension negotiation for SSH, and specifies discovery of signature algorithms as a usage case.

### [4.](#) IANA Considerations

This document augments the Public Key Algorithm Names in [\[RFC4253\]](#) and [\[RFC4250\]](#).

IANA is requested to update the "Secure Shell (SSH) Protocol Parameters" registry with the following entry:

| Public Key Algorithm Name | Reference       | Note                      |
|---------------------------|-----------------|---------------------------|
| rsa-sha2-256              | [this document] | <a href="#">Section 2</a> |
| rsa-sha2-512              | [this document] | <a href="#">Section 2</a> |

### [5.](#) Security Considerations

The security considerations of [\[RFC4253\]](#) apply to this document.

The National Institute of Standards and Technology (NIST) Special Publication 800-131A [\[800-131A\]](#) disallows the use of RSA and DSA keys shorter than 2048 bits for US government use after 2013. Keys of 2048 bits or larger are considered acceptable.

The same document disallows the SHA-1 hash function, as used in the

"ssh-rsa" and "ssh-dss" algorithms, for digital signature generation after 2013. The SHA-2 family of hash functions is seen as acceptable.

## 6. Why no DSA?

A draft version of this memo also defined an algorithm name for use of 2048-bit and 3072-bit DSA keys with a 256-bit subgroup and SHA-2 256 hashing. It is possible to implement DSA securely by generating "k" deterministically as per [[RFC6979](#)]. However, a plurality of reviewers were concerned that implementers would not pay heed, and would use cryptographic libraries that continue to generate "k" randomly. This is vulnerable to biased "k" generation, and extremely vulnerable to "k" reuse. The relative speed advantage of DSA signing compared to RSA signing was not perceived to outweigh this shortcoming, especially since algorithms based on elliptic curves are faster yet.

Due to these disrecommendations, this document abstains from defining an algorithm name for large DSA keys, and recommends RSA instead.

Bider

[Page 4]

---

Internet-Draft

RSA Keys with SHA-2 in SSH

November 2015

## 7. References

### 7.1. Normative References

[FIPS-180-4]

National Institute of Standards and Technology (NIST),  
United States of America, "Secure Hash Standard (SHS)",  
FIPS Publication 180-4, August 2015,  
<<http://dx.doi.org/10.6028/NIST.FIPS.180-4>>.

[RFC2119]

Bradner, S., "Key words for use in RFCs to Indicate  
Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

- [RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", [RFC 3447](#), February 2003.
- [RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", [RFC 4252](#), January 2006.
- [RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", [RFC 4253](#), January 2006.

## [6.2](#). Informative References

- [800-131A] National Institute of Standards and Technology (NIST), "Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths", NIST Special Publication 800-131A, January 2011, <<http://csrc.nist.gov/publications/nistpubs/800-131A/sp800-131A.pdf>>.
- [RFC4250] Lehtinen, S. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Assigned Numbers", [RFC 4250](#), January 2006.
- [RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", [RFC 6979](#), August 2013.
- [SSH-EXT-INFO] Bider, D., "Extension Negotiation in Secure Shell (SSH)", [draft-ssh-ext-info-00](#), November 2015, <<https://tools.ietf.org/html/draft-ssh-ext-info-00>>.

Suites 41/42, Victoria House  
26 Main Street  
GI

Phone: +506 8315 6519  
EMail: ietf-ssh3@denisbider.com  
URI: <https://www.bitvise.com/>

#### Acknowledgments

Thanks to Jon Bright, Niels Moeller, Stephen Farrell, Mark D. Baushke, Jeffrey Hutzelman, Hanno Boeck, Peter Gutmann, and Damien Miller for comments and suggestions.