

Use of RSA Keys with SHA-2 512 in Secure Shell (SSH)
draft-rsa-dsa-sha2-256-01.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. A new mechanism is also defined for servers to inform clients of supported signature algorithms during client authentication.

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/1id-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-2 512, and a mechanism for servers to inform SSH clients of signature algorithms they support and accept.

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.

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 Algorithm

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 algorithm is defined:

```
rsa-sha2-512    RECOMMENDED    sign    Raw RSA key
```

This signature algorithm is 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, this algorithm reuses 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 format, without requiring re-encoding, or affecting already trusted key fingerprints.

Signing and verifying using this algorithm is performed according to the RSASSA-PSS scheme in [\[RFC3447\]](#) using SHA-2 512 [\[FIPS-180-4\]](#) as hash; MGF1 as mask function; and salt length equal to hash size.

The resulting signature is encoded as follows:

```
string    "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-PSS, of length equal to the length in octets of the RSA modulus.

[2.1](#) Use for server authentication

To express support and preference for this algorithm for server authentication, the SSH client and server include the algorithm name, "rsa-sha2-512", in the name-list field "server_host_key_algorithms" in the SSH_MSG_KEXINIT packet [\[RFC4253\]](#). If this host key algorithm 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 algorithm name "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-sha2-512". 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 the "rsa-sha2-512" algorithm name.

3. Informing clients of supported signature algorithms

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

Unfortunately, SSH lacks proper extension negotiation, and existing clients of at least one ubiquitous implementation will disconnect if any additional fields are added to SSH_MSG_SERVICE_REQUEST, SSH_MSG_SERVICE_ACCEPT, or SSH_MSG_USERAUTH_FAILURE.

Therefore, an SSH Server SHOULD inform the client of a list of public key signature algorithms it accepts by sending the following SSH_MSG_IGNORE message immediately before SSH_MSG_SERVICE_ACCEPT:

```
byte    SSH_MSG_IGNORE
string  data:
  string    "sig-algs-accepted"
  name-list signature algorithm names accepted
```

This message SHOULD be sent immediately before SSH_MSG_SERVICE_ACCEPT, so that the client can immediately use this information in a public key authentication request triggered by SSH_MSG_SERVICE_ACCEPT. However, the server MAY also send this message again later, to update the list of accepted algorithms due to the client's authentication progress, failure, or external circumstance. In this case, the message SHOULD be sent immediately before an SSH_MSG_USERAUTH_FAILURE.

If a server does not send this message, a client SHALL NOT make any assumptions about the server's signature algorithm support.

If a server sends this message with an empty algorithm list, the client SHOULD take this to mean that the server will not accept any public key authentication request; even if the "publickey" method is included by the server in the "authentications that can continue" field in the message SSH_MSG_USERAUTH_FAILURE.

Since this notification is sent as SSH_MSG_IGNORE, any incorrect or inconsistent encoding of this message MUST be silently discarded.

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
rsa-sha2-512

Reference
[this document]

Note
[Section 2.1](#)

Bider

[Page 4]

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. References

6.1. 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.

Author's Address

Denis Bider
Bitvise Limited
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.

