

Network Working Group
Johns
Request for Comments: 2786
Excite@Home
Category: Experimental
2000

M. St.

March

Diffie-Helman USM Key
Management Information Base and Textual Convention

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

IESG Note

This document specifies an experimental MIB. Readers, implementers and users of this MIB should be aware that in the future the IETF may charter an IETF Working Group to develop a standards track MIB to address the same problem space that this MIB addresses. It is quite possible that an incompatible standards track MIB may result from that effort.

Abstract

This memo defines an experimental portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it defines a textual convention for doing Diffie-Helman key agreement key exchanges and a set of objects which extend the usmUserTable to permit the use of a DH key exchange in addition to the key change method described in [12]. In otherwords, this MIB adds the possibility of forward secrecy to the USM model. It also defines a set of objects that can be used to kick start security on an SNMPv3 agent when the out of band path is authenticated, but not necessarily private or confidential.

The KeyChange textual convention described in [12] permits secure key changes, but has the property that if a third-party has knowledge of

the original key (e.g. if the agent was manufactured with a standard default key) and could capture all SNMP exchanges, the third-party would know the new key. The Diffie-Helman key change described here

St. Johns Experimental [Page
1]

RFC 2786 Diffie-Helman USM Key March
2000

limits knowledge of the new key to the agent and the manager making the change. In otherwords, this process adds forward secrecy to the key change process.

The recommendation in [12] is that the usmUserTable be populated out of band - e.g. not via SNMP. If the number of agents to be configured is small, this can be done via a console port and manually. If the number of agents is large, as is the case for a cable modem system, the manual approach doesn't scale well. The combination of the two mechanisms specified here - the DH key change mechanism, and the DH key ignition mechanism - allows managable use of SNMPv3 USM in a system of millions of devices.

This memo specifies a MIB module in a manner that is compliant to the SNMP SMIV2[5][6][7]. The set of objects is consistent with the SNMP framework and existing SNMP standards and is intended for use with the SNMPv3 User Security Model MIB and other security related MIBs.

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 [16].

This memo is a private submission by the author, but is applicable to the SNMPv3 working group within the Internet Engineering Task Force.

Comments are solicited and should be addressed to the the author.

Table of Contents

1	The SNMP Management Framework	2
1.1	Structure of the MIB	3

2	Theory of Operation	4
4	2.1 Diffie-Helman Key Changes	4
4	2.2 Diffie-Helman Key Ignition	4
6	3 Definitions	6
17	4 References	17
18	5 Security Considerations	18
19	6 Intellectual Property	19
19	7 Author's Address	19
20	8 Full Copyright Statement	20

1. The SNMP Management Framework The SNMP Management Framework presently consists of five major components:

- o An overall architecture, described in RFC 2271 [1].
- o Mechanisms for describing and naming objects and events for the purpose of management. The first version of this Structure of Management Information (SMI) is called SMIV1 and described in STD

St. Johns 2]	Experimental	[Page
RFC 2786 2000	Diffie-Helman USM Key	March

16, RFC 1155 [2], STD 16, RFC 1212 [3] and RFC 1215 [4]. The second version, called SMIV2, is described in STD 58, RFC 2578 [5], STD 58, RFC 2579 [6] and STD 58, RFC 2580 [7].

- o Message protocols for transferring management information. The first version of the SNMP message protocol is called SNMPv1 and described in STD 15, RFC 1157 [8]. A second version of the message protocol, which is not an Internet standards track protocol, is called SNMPv2c and described in RFC 1901 [9] and RFC 1906 [10]. The third version of the message protocol is called SNMPv3 and described in RFC 1906 [10], RFC 2272 [11] and RFC 2274 [12].

- o Protocol operations for accessing management information. The first set of protocol operations and associated PDU formats is described in STD 15, RFC 1157 [8]. A second set of protocol operations and associated PDU formats is described in RFC 1905

[13].

- o A set of fundamental applications described in RFC 2273 [14] and the view-based access control mechanism described in RFC 2275 [15].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. Objects in the MIB are defined using the mechanisms defined in the SMI.

This memo specifies a MIB module that is compliant to the SMIV2. A MIB conforming to the SMIV1 can be produced through the appropriate

translations. The resulting translated MIB must be semantically equivalent, except where objects or events are omitted because no translation is possible (use of Counter64). Some machine readable information in SMIV2 will be converted into textual descriptions in

SMIV1 during the translation process. However, this loss of machine

readable information is not considered to change the semantics of the MIB.

1.1. Structure of the MIB

This MIB is structured into three groups and a single textual convention:

- o The DHKeyChange textual convention defines the process for changing a secret key value via a Diffie-Helman key exchange.
- o The usmDHPublicObjects group contains a single object which describes the public Diffie-Helman parameters required by any instance of a DHKeyChange typed object.

St. Johns
3]

Experimental

[Page

RFC 2786
2000

Diffie-Helman USM Key

March

o The usmDHUserKeyTable augments and extends the usmUserTable defined in the SNMPv3 User-based Security Model MIB [12] by providing objects which permit the updating of the Authentication and Privacy keys for a row in this table through the use of a Diffie-Helman key exchange.

- o The usmDHKickstartTable provides a mechanism for a management station to be able to agree upon a set of authentication and confidentiality keys and their associated row in the usmUserTable.

2. Theory of Operation

2.1. Diffie-Helman Key Changes

Upon row creation (in the usmUserTable), or object change (either of the object in the usmDHUserKeyTable or its associated value in the usmUserTable), the agent generates a random number. From this random number, the agent uses the DH parameters and transforms to derive a DH public value which is then published to the associated MIB object.

The management station reads one or more of the objects in the usmDHUserKeyTable to get the agent's DH public values.

The management station generates a random number, derives a DH public value from that random number (as described in the DHKeyChange Textual Convention), and does an SNMP SET against the object in the usmDHUserKeyTable. The set consists of the concatenation of the agent's derived DH public value and the manager's derived DH public value (to ensure the DHKeyChange object hasn't otherwise changed in the meantime).

Upon successful completion of the set, the underlying key (authentication or confidentiality) for the associated object in the usmUserTable is changed to a key derived from the DH shared secret.

Both the agent and the management station are able to calculate this value based on their knowledge of their own random number and the other's DH public number.

2.2. Diffie-Helman Key Ignition

[12] recommends that the usmUserTable be populated out of band, for example - manually. This works reasonably well if there are a small number of agents, or if all the agents are using the same key material, and if the device is physically accessible for that action.

It does not scale very well to the case of possibly millions of devices located in thousands of locations in hundreds of markets in

multiple countries. In other words, it doesn't work well with a

cable modem system, and may not work all that well with other large-scale consumer broadband IP offerings.

The methods described in the objects under the `usmDhKickstartGroup` can be used to populate the `usmUserTable` in the circumstances where

you may be able to provide at least limited integrity for the provisioning process, but you can't guarantee confidentiality. In addition, as a side effect of using the DH exchange, the operational

USM keys for each agent will differ from the operational USM keys for

every other device in the system, ensuring that compromise of one device does not compromise the system as a whole.

The vendor who implements these objects is expected to provide one or

more `usmSecurityNames` which map to a set of accesses defined in the

VACM [15] tables. For example, the vendor may provide a 'root' user

who has access to the entire device for read-write, and 'operator' user who has access to the network specific monitoring objects and can also reset the device, and a 'customer' user who has access to a

subset of the monitoring objects which can be used to help the customer debug the device in conjunction with customer service questions.

To use, the system manager (the organization or individual who own the group of devices) generates one or more random numbers - R.

The

manager derives the DH Public Numbers R' from these random numbers,

associates the public numbers with a security name, and configures the agent with this association. The configuration would be done either manually (in the case of a small number of devices), or via some sort of distributed configuration file. The actual mechanism is

outside the scope of this document. The agent in turn generates a random number for each name/number pair, and publishes the DH

Public

Number derived from its random number in the `usmDhKickstartTable` along with the manager's public number and provided security name.

Once the agent is initialized, an SNMP Manager can read the contents

of the `usmDhKickstartTable` using the security name of 'dhKickstart'

with no authentication. The manager looks for one or more entries in

this table where it knows the random number used to derive the `usmDhKickstartMgrPublic` number. Given the manager's knowledge of the

private random number, and the `usmDhKickstartMyPublic` number, the manager can calculate the DH shared secret. From that shared secret,

it can derive the operational authentication and confidentiality keys

for the usmUserTable row which has the matching security name.
Given
the keys and the security name, the manager can then use normal
USM
mechanisms to access the remainder of the agent's MIB space.

St. Johns 5]	Experimental	[Page
RFC 2786 2000	Diffie-Helman USM Key	March

3. Definitions

```
SNMP-USM-DH-OBJECTS-MIB DEFINITIONS ::= BEGIN
```

```
IMPORTS
```

```
    MODULE-IDENTITY, OBJECT-TYPE,  
    -- OBJECT-IDENTITY,  
    experimental, Integer32  
    FROM SNMPv2-SMI  
    TEXTUAL-CONVENTION  
    FROM SNMPv2-TC  
    MODULE-COMPLIANCE, OBJECT-GROUP  
    FROM SNMPv2-CONF  
    usmUserEntry  
    FROM SNMP-USER-BASED-SM-MIB  
    SnmpAdminString  
    FROM SNMP-FRAMEWORK-MIB;
```

```
snmpUsmDHObjectsMIB MODULE-IDENTITY
```

```
    LAST-UPDATED "200003060000Z" -- 6 March 2000, Midnight  
    ORGANIZATION "Excite@Home"  
    CONTACT-INFO "Author: Mike StJohns  
                  Postal: Excite@Home  
                    450 Broadway  
                    Redwood City, CA 94063  
                  Email: stjohns@corp.home.net  
                  Phone: +1-650-556-5368"
```

```
DESCRIPTION
```

```
"The management information definitions for providing forward  
secrecy for key changes for the usmUserTable, and for providing a  
method for 'kickstarting' access to the agent via a Diffie-Helman  
key agreement."
```

```
REVISION "200003060000Z"
```

```
DESCRIPTION
```

```
"Initial version published as RFC 2786."
```

```
::= { experimental 101 } -- IANA DHKEY-CHANGE 101
```

```
-- Administrative assignments
```

```
usmDHKeyObjects OBJECT IDENTIFIER ::= { snmpUsmDHObjectsMIB 1 }
```

usmDHKeyConformance OBJECT IDENTIFIER ::= { snmpUsmDHObjectsMIB 2 }

-- Textual conventions

St. Johns Experimental [Page
6]

RFC 2786 Diffie-Helman USM Key March
2000

DHKeyChange ::= TEXTUAL-CONVENTION
STATUS current
DESCRIPTION

"Upon initialization, or upon creation of a row containing an object of this type, and after any successful SET of this value,

a GET of this value returns 'y' where $y = g^{xa} \text{ MOD } p$, and where g is

the base from usmDHParameters, p is the prime from usmDHParameters, and xa is a new random integer selected by the agent in the interval $2^{(l-1)} \leq xa < 2^l < p-1$. 'l' is the optional privateValueLength from usmDHParameters in bits. If 'l' is omitted, then xa (and xr below) is selected in the interval $0 \leq xa < p-1$. y is expressed as an OCTET STRING 'PV' of length

'k' which satisfies

$$y = \sum_{i=1}^k 2^{(8(k-i))} PV^i$$

and where PV1, ..., PVk are the octets of PV from first to last, where PV1 <> 0.

A successful SET consists of the value 'y' expressed as an OCTET STRING as above concatenated with the value 'z' (expressed as an OCTET STRING in the same manner as y) where $z = g^{xr} \text{ MOD } p$, where g, p and l are as above, and where xr is a new random integer selected by the manager in the interval $2^{(l-1)} \leq xr < 2^l < p-1$. A SET to an object of this type will fail with the error wrongValue if the current 'y' does not match the 'y' portion of the value of the varbind for the object. (E.g. GET yout, SET concat(yin, z), yout <> yin).

Note that the private values xa and xr are never transmitted from manager to device or vice versa, only the values y and z. Obviously, these values must be retained until a successful SET on the associated object.

The shared secret 'sk' is calculated at the agent as $sk = z^{xa} \text{ MOD } p$, and at the manager as $sk = y^{xr} \text{ MOD } p$.

Each object definition of this type MUST describe how to map from the shared secret 'sk' to the operational key value used by the

protocols and operations related to the object. In general, if n bits of key are required, the author suggests using the n right-most bits of the shared secret as the operational key value."

REFERENCE

-- Diffie-Hellman Key-Agreement Standard, PKCS #3;
RSA Laboratories, November 1993"

SYNTAX OCTET STRING

St. Johns Experimental [Page
7]

RFC 2786 Diffie-Helman USM Key March
2000

-- Diffie Hellman public values

usmDHPublicObjects OBJECT IDENTIFIER ::= { usmDHKeyObjects 1 }

usmDHParameters OBJECT-TYPE

SYNTAX OCTET STRING
MAX-ACCESS read-write
STATUS current
DESCRIPTION

"The public Diffie-Hellman parameters for doing a Diffie-Hellman key agreement for this device. This is encoded as an ASN.1 DHParameter per PKCS #3, section 9. E.g.

```
DHParameter ::= SEQUENCE {  
    prime    INTEGER,    -- p  
    base     INTEGER,    -- g  
    privateValueLength  INTEGER OPTIONAL }
```

Implementors are encouraged to use either the values from Oakley Group 1 or the values of from Oakley Group 2 as specified in RFC-2409, The Internet Key Exchange, Section 6.1, 6.2 as the default for this object. Other values may be used, but the security properties of those values MUST be well understood and MUST meet the requirements of PKCS #3 for the selection of Diffie-Hellman primes.

In addition, any time usmDHParameters changes, all values of type DHKeyChange will change and new random numbers MUST be generated by the agent for each DHKeyChange object."

REFERENCE

-- Diffie-Hellman Key-Agreement Standard, PKCS #3,
RSA Laboratories, November 1993
-- The Internet Key Exchange, RFC 2409, November 1998,
Sec 6.1, 6.2"

::= { usmDHPublicObjects 1 }

usmDHUserKeyTable OBJECT-TYPE

SYNTAX SEQUENCE OF UsmDHUserKeyEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION

"This table augments and extends the usmUserTable and provides

4 objects which exactly mirror the objects in that table with the textual convention of 'KeyChange'. This extension allows key changes to be done in a manner where the knowledge of the current secret plus knowledge of the key change data exchanges (e.g. via wiretapping) will not reveal the new key."

St. Johns
8]

Experimental

[Page

RFC 2786
2000

Diffie-Hellman USM Key

March

```
::= { usmDHPublicObjects 2 }
```

usmDHUserKeyEntry OBJECT-TYPE

SYNTAX UsmDHUserKeyEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"A row of DHKeyChange objects which augment or replace the functionality of the KeyChange objects in the base table row."

AUGMENTS { usmUserEntry }

```
::= {usmDHUserKeyTable 1 }
```

UsmDHUserKeyEntry ::= SEQUENCE {

usmDHUserAuthKeyChange DHKeyChange,

usmDHUserOwnAuthKeyChange DHKeyChange,

usmDHUserPrivKeyChange DHKeyChange,

usmDHUserOwnPrivKeyChange DHKeyChange

}

usmDHUserAuthKeyChange OBJECT-TYPE

SYNTAX DHKeyChange

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The object used to change any given user's Authentication Key

using a Diffie-Hellman key exchange.

The right-most n bits of the shared secret 'sk', where 'n' is the number of bits required for the protocol defined by usmUserAuthProtocol, are installed as the operational authentication key for this row after a successful SET."

```
::= { usmDHUserKeyEntry 1 }
```

usmDHUserOwnAuthKeyChange OBJECT-TYPE

SYNTAX DHKeyChange

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The object used to change the agents own Authentication Key using a Diffie-Hellman key exchange.

The right-most n bits of the shared secret 'sk', where 'n' is the number of bits required for the protocol defined by

usmUserAuthProtocol, are installed as the operational authentication key for this row after a successful SET."
 ::= { usmDHUserKeyEntry 2 }

usmDHUserPrivKeyChange OBJECT-TYPE

St. Johns Experimental [Page
9]

RFC 2786 Diffie-Hellman USM Key March
2000

SYNTAX DHKeyChange
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The object used to change any given user's Privacy Key using a Diffie-Hellman key exchange.

The right-most n bits of the shared secret 'sk', where 'n' is the number of bits required for the protocol defined by usmUserPrivProtocol, are installed as the operational privacy key for this row after a successful SET."
 ::= { usmDHUserKeyEntry 3 }

usmDHUserOwnPrivKeyChange OBJECT-TYPE

SYNTAX DHKeyChange
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"The object used to change the agent's own Privacy Key using
a
Diffie-Hellman key exchange.

The right-most n bits of the shared secret 'sk', where 'n' is the number of bits required for the protocol defined by usmUserPrivProtocol, are installed as the operational privacy key for this row after a successful SET."
 ::= { usmDHUserKeyEntry 4 }

usmDHKickstartGroup OBJECT IDENTIFIER ::= { usmDHKeyObjects 2 }

usmDHKickstartTable OBJECT-TYPE

SYNTAX SEQUENCE OF UsmDHKickstartEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"A table of mappings between zero or more Diffie-Hellman key agreement values and entries in the usmUserTable. Entries in
this
table are created by providing the associated device with a Diffie-Hellman public value and a usmUserName/usmUserSecurityName pair during initialization. How these values are provided is outside the scope of this MIB, but could be provided manually, or through a configuration file. Valid public value/name pairs result in the creation of a row in this table as well as the creation of an associated row (with keys derived as indicated) in the usmUserTable. The actual access the related usmSecurityName

has is dependent on the entries in the VACM tables. In general, an implementor will specify one or more standard security names and will provide entries in the VACM tables granting various levels of access to those names. The actual content of the VACM

St. Johns Experimental [Page
10]

RFC 2786 Diffie-Helman USM Key March
2000

table is beyond the scope of this MIB.

Note: This table is expected to be readable without authentication using the `usmUserSecurityName 'dhKickstart'`. See the conformance statements for details."

```
::= { usmDhKickstartGroup 1 }
```

```
usmDhKickstartEntry OBJECT-TYPE  
SYNTAX      UsmDhKickstartEntry  
MAX-ACCESS  not-accessible  
STATUS      current  
DESCRIPTION
```

```
"An entry in the usmDhKickstartTable. The agent SHOULD  
either  
delete this entry or mark it as inactive upon a successful SET of  
any of the KeyChange-typed objects in the usmUserEntry or upon a  
successful SET of any of the DhKeyChange-typed objects in the  
usmDhKeyChangeEntry where the related usmSecurityName (e.g. row  
of  
usmUserTable or row of usmDhKeyChangeTable) equals this entry's  
usmDhKickstartSecurityName. In otherwords, once you've changed  
one or more of the keys for a row in usmUserTable with a  
particular security name, the row in this table with that same  
security name is no longer useful or meaningful."
```

```
INDEX      { usmDhKickstartIndex }  
::= { usmDhKickstartTable 1 }
```

```
UsmDhKickstartEntry ::= SEQUENCE {  
    usmDhKickstartIndex      Integer32,  
    usmDhKickstartMyPublic   OCTET STRING,  
    usmDhKickstartMgrPublic  OCTET STRING,  
    usmDhKickstartSecurityName SnmpAdminString  
}
```

```
usmDhKickstartIndex OBJECT-TYPE  
SYNTAX      Integer32 (1..2147483647)  
MAX-ACCESS  not-accessible  
STATUS      current  
DESCRIPTION  
    "Index value for this row."  
::= { usmDhKickstartEntry 1 }
```

```
usmDhKickstartMyPublic OBJECT-TYPE  
SYNTAX      OCTET STRING
```

MAX-ACCESS read-only
STATUS current
DESCRIPTION
"The agent's Diffie-Hellman public value for this row. At

St. Johns Experimental [Page
11]

RFC 2786 Diffie-Helman USM Key March
2000

initialization, the agent generates a random number and derives its public value from that number. This public value is published here. This public value 'y' equals $g^r \text{ MOD } p$ where g is the from the set of Diffie-Hellman parameters, p is the prime from those parameters, and r is a random integer selected by the agent in the interval $2^{(l-1)} \leq r < p-1 < 2^l$. If l is unspecified, then r is a random integer selected in the interval $0 \leq r < p-1$

The public value is expressed as an OCTET STRING 'PV' of length 'k' which satisfies

$$y = \sum_{i=1}^k 2^{8(k-i)} PV^i$$

and where PV_1, \dots, PV_k are the octets of PV from first to last, where $PV_1 \neq 0$.

The following DH parameters (Oakley group #2, RFC 2409, sec 6.1, 6.2) are used for this object:

$g = 2$
 $p =$ FFFFFFFF FFFFFFFF C90FDAA2 2168C234 C4C6628B 80DC1CD1
29024E08 8A67CC74 020BBEA6 3B139B22 514A0879 8E3404DD
EF9519B3 CD3A431B 302B0A6D F25F1437 4FE1356D 6D51C245
E485B576 625E7EC6 F44C42E9 A637ED6B 0BFF5CB6 F406B7ED
EE386BFB 5A899FA5 AE9F2411 7C4B1FE6 49286651 ECE65381
FFFFFFFF FFFFFFFF

$l=1024$

"

REFERENCE

-- Diffie-Hellman Key-Agreement Standard, PKCS#3v1.4;
RSA Laboratories, November 1993
-- The Internet Key Exchange, RFC2409;
Harkins, D., Carrel, D.; November 1998"
 ::= { usmDHKickstartEntry 2 }

usmDHKickstartMgrPublic OBJECT-TYPE

SYNTAX OCTET STRING
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The manager's Diffie-Hellman public value for this row.

Note

that this value is not set via the SNMP agent, but may be set via some out of band method, such as the device's configuration file.

St. Johns
12]

Experimental

[Page

RFC 2786
2000

Diffie-Helman USM Key

March

The manager calculates this value in the same manner and using the same parameter set as the agent does. E.g. it selects a random number 'r', calculates $y = g^r \text{ mod } p$ and provides 'y' as the public number expressed as an OCTET STRING. See usmDHKickstartMyPublic for details.

When this object is set with a valid value during initialization, a row is created in the usmUserTable with the following values:

usmUserEngineID	localEngineID
usmUserName	[value of usmDHKickstartSecurityName]
usmUserSecurityName	[value of usmDHKickstartSecurityName]
usmUserCloneFrom	ZeroDotZero
usmUserAuthProtocol	usmHMACMD5AuthProtocol
usmUserAuthKeyChange	-- derived from set value
usmUserOwnAuthKeyChange	-- derived from set value
usmUserPrivProtocol	usmDESPrivProtocol
usmUserPrivKeyChange	-- derived from set value
usmUserOwnPrivKeyChange	-- derived from set value
usmUserPublic	' '
usmUserStorageType	permanent
usmUserStatus	active

A shared secret 'sk' is calculated at the agent as $sk = mgrPublic^r \text{ mod } p$ where r is the agents random number and p is the DH prime from the common parameters. The underlying privacy key for this row is derived from sk by applying the key derivation function PBKDF2 defined in PKCS#5v2.0 with a salt of 0xd1310ba6, and iterationCount of 500, a keyLength of 16 (for usmDESPrivProtocol), and a prf (pseudo random function) of 'id-hmacWithSHA1'. The underlying authentication key for this row is derived from sk by applying the key derivation function PBKDF2 with a salt of 0x98dfb5ac, an interation count of 500, a keyLength of 16 (for usmHMAC5AuthProtocol), and a prf of 'id-hmacWithSHA1'. Note: The salts are the first two words in the ks0 [key schedule 0] of the BLOWFISH cipher from 'Applied Cryptography' by Bruce Schneier - they could be any relatively random string of bits.

The manager can use its knowledge of its own random number and the

- [9] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Introduction to Community-based SNMPv2", RFC 1901, January 1996.

St. Johns 17]	Experimental	[Page
RFC 2786 2000	Diffie-Helman USM Key	March

- [10] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1906, January 1996.

- [11] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", RFC 2572, April 1999.

- [12] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", RFC 2574, April 1999.

- [13] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1905, January 1996.

- [14] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC 2573, April 1999.

- [15] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", RFC 2575, April 1999.

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

- [17] "Diffie-Hellman Key-Agreement Standard, Version 1.4", PKCS #3, RSA Laboratories, November 1993.

- [18] Harkins, D. and D. Carrel, "The Internet Key Exchange", RFC 2409, November 1988.

- [19] Eastlake, D., Crocker, S. and J. Schiller, "Randomness Recommendations for Security", RFC 1750, December 1994.

5. Security Considerations

this standard. Please address the information to the IETF
Executive
Director.

7. Author's Address

Michael C. StJohns
Excite@Home
450 Broadway
Redwood City, CA 94063
USA

Phone: +1-650-556-5368
EMail: stjohns@corp.home.net

St. Johns 19]	Experimental	[Page
RFC 2786 2000	Diffie-Helman USM Key	March

9. Full Copyright Statement

Copyright (C) The Internet Society (2000). All Rights Reserved.

This document and translations of it may be copied and furnished
to
others, and derivative works that comment on or otherwise explain
it
or assist in its implementation may be prepared, copied, published
and distributed, in whole or in part, without restriction of any
kind, provided that the above copyright notice and this paragraph
are
included on all such copies and derivative works. However, this
document itself may not be modified in any way, such as by
removing
the copyright notice or references to the Internet Society or
other
Internet organizations, except as needed for the purpose of
developing Internet standards in which case the procedures for
copyrights defined in the Internet Standards process must be
followed, or as required to translate it into languages other than
English.

The limited permissions granted above are perpetual and will not
be
revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on
an

"AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET
ENGINEERING
TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING
BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF
MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the
Internet Society.