Network Working Group
Request for Comments: 4088
Category: Standards Track

D. Black
EMC Corporation
K. McCloghrie
Cisco Systems
J. Schoenwaelder
International University Bremen

June 2005

Uniform Resource Identifier (URI) Scheme for the Simple Network Management Protocol (SNMP)

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

The Simple Network Management Protocol (SNMP) and the Internet Standard Management Framework are widely used for the management of communication devices, creating a need to specify SNMP access (including access to SNMP MIB object instances) from non-SNMP management environments. For example, when out-of-band IP management is used via a separate management interface (e.g., for a device that does not support in-band IP access), a uniform way to indicate how to contact the device for management is needed. Uniform Resource Identifiers (URIs) fit this need well, as they allow a single text string to indicate a management access communication endpoint for a wide variety of IP-based protocols.

This document defines a URI scheme so that SNMP can be designated as the protocol used for management. The scheme also allows a URI to designate one or more MIB object instances.

Table of Contents

1. Introduction

SNMP and the Internet-Standard Management Framework were originally devised to manage IP devices via in-band means, in which management access is primarily via the same interface(s) used to send and receive IP traffic. SNMP's wide adoption has resulted in its use for managing communication devices that do not support in-band IP access (e.g., Fibre Channel devices); a separate out-of-band IP interface is often used for management. URIs provide a convenient way to locate that interface and specify the protocol to be used for management; one possible scenario is for an in-band query to return a URI that indicates how the device is managed. This document specifies a URI scheme to permit SNMP (including a specific SNMP context) to be designated as the management protocol by such a URI. This scheme also allows a URI to refer to specific object instances within an SNMP MIB.

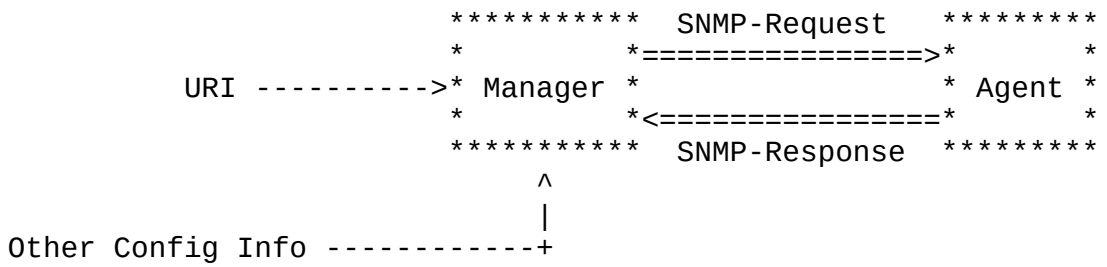
For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to Section 7 of [RFC3410].

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. Usage

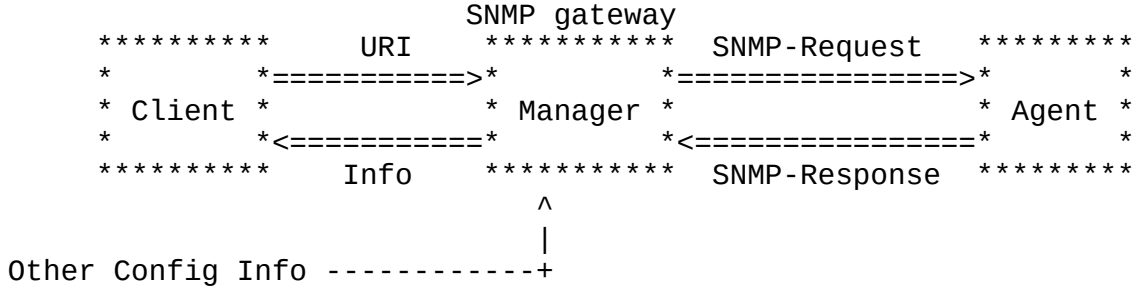
There are two major classes of SNMP URI usage: configuration and gateways between SNMP and other protocols that use SNMP URIs.

An SNMP URI used for configuration indicates the location of management information as part of the configuration of an application containing an SNMP manager. The URI can be obtained from a configuration file or may be provided by a managed device (see Section 1 for an example). Management information is exchanged between the SNMP manager and agent, but it does not flow beyond the manager, as shown in the following diagram:



Additional configuration information (e.g., a security secret or key) may be provided via an interface other than that used for the URI. For example, when a managed device provides an SNMP URI in an unprotected fashion, that device should not provide a secret or key required to use the URI. The secret or key should instead be pre-configured in or pre-authorized to the manager; see Section 6.

For gateway usage, clients employ SNMP URIs to request management information via an SNMP URI to SNMP gateway (also called an SNMP gateway in this document). The SNMP manager within the SNMP gateway accesses the management information and returns it to the requesting client, as shown in the following diagram:



Additional configuration information (e.g., security secrets or keys) may be provided via an interface other than that used for the URI. For example, some types of security information, including secrets

and keys, should be pre-configured in or pre-authorized to the manager rather than be provided by the client; see Section 6.

3. Syntax of an SNMP URI

An SNMP URI has the following ABNF [RFC2234] syntax, based on the ABNF syntax rules for userinfo, host, port, and (path) segment in [RFC3986] and the ABNF syntax rule for HEXDIG in [RFC2234]:

```
snmp-uri        = "snmp://" snmp-authority [ context [ oids ]]

snmp-authority  = [ securityName "@" ] host [ ":" port ]
securityName    = userinfo    ; SNMP securityName

context         = "/" contextName [ ";" contextEngineID ]
contextName     = segment     ; SNMP contextName
contextEngineID = 1*(HEXDIG HEXDIG)    ; SNMP contextEngineID

oids            = "/" ( oid / oid-group ) [ suffix ]
oid-group       = "(" oid *( "," oid ) ")"
oid             = < as specified by [RFC 3061] >
suffix          = "+" / ".*"
```

The userinfo and (path) segment ABNF rules are reused for syntax only. In contrast, host and port have both the syntax and semantics specified in [RFC3986]. See [RFC3411] for the semantics of securityName, contextEngineID, and contextName.

The snmp-authority syntax matches the URI authority syntax in Section 3.2 of [RFC3986], with the additional restriction that the userinfo component of an authority (when present) MUST be an SNMP securityName. If the securityName is empty or not given, the entity making use of an SNMP URI is expected to know what SNMP securityName to use if one is required. Inclusion of authentication information (e.g., passwords) in URIs has been deprecated (see Section 3.2.1 of [RFC3986]), so any secret or key required for SNMP access must be provided via other means that may be out-of-band with respect to communication of the URI. If the port is empty or not given, port 161 is assumed.

If the contextName is empty or not given, the zero-length string ("") is assumed, as it is the default SNMP context. An SNMP contextEngineID is a variable-format binary element that is usually discovered by an SNMP manager. An SNMP URI encodes a contextEngineID as hexadecimal digits corresponding to a sequence of bytes. If the contextEngineID is empty or not given, the context engine is to be discovered by querying the SNMP agent at the specified host and port; see Section 4.1 below. The contextEngineID component of the URI

SHOULD be present if more than one context engine at the designated host and port supports the designated context.

An SNMP URI that designates the default SNMP context ("") MAY end with the "/" character that introduces the contextName component. An SNMP URI MUST NOT end with the "/" character that introduces an oid or oid-group component, as the empty string is not a valid OID for SNMP.

The encoding rules specified in [RFC3986] MUST be used for SNMP URIs, including the use of percent encoding ("%" followed by two hex digits) as needed to represent characters defined as reserved in [RFC3986] and any characters not allowed in a URI. SNMP permits any UTF-8 character to be used in a securityName or contextName; all multi-byte UTF-8 characters in an SNMP URI MUST be percent encoded as specified in Sections 2.1 and 2.5 of [RFC3986]. These requirements are a consequence of reusing the ABNF syntax rules for userinfo and segment from [RFC3986].

SNMP URIs will generally be short enough to avoid implementation string-length limits (e.g., that may occur at 255 characters). Such limits may be a concern for large OID groups; relative references to URIs (see Section 4.2 of [RFC3986]) may provide an alternative in some circumstances.

Use of IP addresses in SNMP URIs is acceptable in situations where dependence on availability of DNS service is undesirable or must be avoided; otherwise, IP addresses should not be used (see [RFC1900] for further explanation).

3.1. Relative Reference Considerations

Use of the SNMP default context (zero-length string) within an SNMP URI can result in a second instance of "//" in the URI, such as the following:

```
snmp://<host>//<oid>
```

This is allowed by [RFC3986] syntax; if a URI parser does not handle the second "//" correctly, the parser is broken and needs to be fixed. This example is important because use of the SNMP default context in SNMP URIs is expected to be common.

On the other hand, the second occurrence of "//" in an absolute SNMP URI affects usage of relative references to that URI (see Section 4.2 of [RFC3986]) because a "//" at the start of a relative reference always introduces a URI authority component (host plus optional userinfo and/or port; see [RFC3986]). Specifically, a relative

reference of the form //<oid2> will not work, because the "//" will cause <oid2> to be parsed as a URI authority, resulting in a syntax error when the parser fails to find a host in <oid2> . To avoid this problem, relative references that start with "//" but do not contain a URI authority component MUST NOT be used. Functionality equivalent to any such forbidden relative reference can be obtained by prefixing "." or ".." to the forbidden relative reference (e.g., ..//<oid2>). The prefix to use depends on the base URI.

4. Semantics and Operations

An SNMP URI that does not include any OIDs is called an SNMP service URI because it designates a communication endpoint for access to SNMP management service. An SNMP URI that includes one or more OIDs is called an SNMP object URI because it designates one or more object instances in an SNMP MIB. The expected means of using an SNMP URI is to employ an SNMP manager to access the SNMP context designated by the URI via the SNMP agent at the host and port designated by the URI.

4.1. SNMP Service URIs

An SNMP service URI does not designate a data object, but rather an SNMP context to be accessed by a service; the telnet URI scheme [RFC1738] is another example of URIs that designate service access. If the contextName in the URI is empty or not given, "" (the zero-length string) is assumed, as it is the default SNMP context.

If a contextEngineID is given in an SNMP service URI, the context engine that it designates is to be used. If the contextEngineID is empty or not given in the URI, the context engine is to be discovered; the context engine to be used is the one that supports the context designated by the URI. The contextEngineID component of the URI SHOULD be present if more than one context engine at the designated host and port supports the designated context.

Many common uses of SNMP URIs are expected to omit (i.e., default) the contextEngineID because they do not involve SNMP proxy agents, which are the most common reason for multiple SNMP context engines to exist at a single host and port. Specifically, when an SNMP agent is local to the network interface that it manages, the agent will usually have only one context engine, in which case it is safe to omit the contextEngineID component of an SNMP URI. In addition, many SNMP agents that are local to a network interface support only the default SNMP context (zero-length string).

4.2. SNMP Object URIs

An SNMP object URI contains one or more OIDs. The URI is used by first separating the OID or OID group (including its preceding slash plus any parentheses and suffix) and then processing the resulting SNMP service URI as specified in Section 4.1 (above) to determine the SNMP context to be accessed. The OID or OID group is then used to generate SNMP operations directed to that SNMP context.

The semantics of an SNMP object URI depend on whether the OID or OID group has a suffix and what that suffix is. There are three possible formats; in each case, the MIB object instances are designated within the SNMP context specified by the service URI portion of the SNMP object URI. The semantics of an SNMP object URI that contains a single OID are as follows:

(1) An OID without a suffix designates the MIB object instance named by the OID.

(2) An OID with a "+" suffix designates the lexically next MIB object instance following the OID.

(3) An OID with a ".*" suffix designates the set of MIB object instances for which the OID is a strict lexical prefix; this does not include the MIB object instance named by the OID.

An OID group in an SNMP URI consists of a set of OIDs in parentheses. In each case, the OID group semantics are the extension of the single OID semantics to each OID in the group (e.g., a URI with a "+" suffix designates the set of MIB object instances consisting of the lexically next instance for each OID in the OID group).

When there is a choice among URI formats to designate the same MIB object instance or instances, the above list is in order of preference (no suffix is most preferable), as it runs from most precise to least precise. This is because an OID without a suffix precisely designates an object instance, whereas a "+" suffix designates the next object instance, which may change, and the ".*" suffix could designate multiple object instances. Multiple syntactically distinct SNMP URIs SHOULD NOT be used to designate the same MIB object instance or set of instances, as this may cause unexpected results in URI-based systems that use string comparison to test URIs for equality.

SNMP object URIs designate the data to be accessed, as opposed to the specific SNMP operations to be used for access; Section 4.2.1 provides examples of how SNMP operations can be used to access data for SNMP object URIs. Nonetheless, any applicable SNMP operation,

including GetBulk, MAY be used to access data for all or part of one or more SNMP object URIs (e.g., via use of multiple variable bindings in a single operation); it is not necessary to use the specific operations described in Section 4.2.1 as long as the results (returned variable bindings or error) could have been obtained by following Section 4.2.1's descriptions. The use of relative references that do not change the contextName (i.e., ./<oid>) should be viewed as a hint that optimization of SNMP access across multiple SNMP URIs may be possible.

An SNMP object URI MAY also be used to specify a MIB object instance or instances to be written; this causes generation of an SNMP Set operation instead of a Get. The "+" and ".*" suffixes MUST NOT be used in this case; any attempt to do so is an error that MUST NOT generate any SNMP Set operations. Values to be written to the MIB object instance or instances are not specified within an SNMP object URI.

SNMP object URIs designate data in SNMP MIBs and hence do not provide the means to generate all possible SNMP protocol operations. For example, data access for an SNMP object URI cannot directly generate either Snmpv2-Trap or InformRequest notifications, although side effects of data access could cause such notifications (depending on the MIB). In addition, whether and how GetBulk is used for an SNMP object URI with a ".*" suffix is implementation specific.

4.2.1. SNMP Object URI Data Access

Data access based on an SNMP object URI returns an SNMP variable binding for each MIB object instance designated by the URI, or an SNMP error if the operation fails. An SNMP variable binding binds a variable name (OID) to a value or an SNMP exception (see [RFC3416]). The SNMP operation or operations needed to access data designated by an SNMP object URI depend on the OID or OID group suffix or absence thereof. The following descriptions are not the only method of performing data access for an SNMP object URI; any suitable SNMP operations may be used as long as the results (returned variable bindings or error) are functionally equivalent.

(1) For an OID or OID group without a suffix, an SNMP Get operation is generated using each OID as a variable binding name. If an SNMP error occurs, that error is the result of URI data access; otherwise, the returned variable binding or bindings are the result of URI data access. Note that any returned variable binding may contain an SNMP "noSuchObject" or "noSuchInstance" exception.

(2) For an OID or OID group with a "+" suffix, an SNMP GetNext operation is generated using each OID as a variable binding name. If an SNMP error occurs, that error is the result of URI data access; otherwise, the returned variable binding or bindings are the result of URI data access. Note that any returned variable binding may contain an SNMP "endOfMibView" exception.

(3) For an OID or OID group with a ".*" suffix, an SNMP GetNext operation is initially generated using each OID as a variable binding name. If the result is an SNMP error, that error is the result of URI data access. If all returned variable bindings contain either a) an OID for which the corresponding URI OID is not a lexical prefix or b) an SNMP "endOfMibView" exception, then the returned variable bindings are the result of URI data access.

Otherwise, the results of the GetNext operation are saved, and another SNMP GetNext operation is generated using the newly returned OIDs as variable binding names. This is repeated (save the results and generate a GetNext with newly returned OIDs as variable binding names) until all the returned variable bindings from a GetNext contain either a) an OID for which the corresponding URI OID is not a lexical prefix or b) an SNMP "endOfMibView" exception. The results from all of the GetNext operations are combined to become the overall result of URI data access; this may include variable bindings whose OID is not a lexical extension of the corresponding URI OID. If the OID subtrees (set of OIDs for which a specific URI OID is a lexical prefix) are not the same size for all OIDs in the OID group, the largest subtree determines when this iteration ends. SNMP GetBulk operations MAY be used to optimize this iterated access.

Whenever a returned variable binding contains an OID for which the corresponding URI OID is not a lexical prefix or an SNMP "endOfMibView" exception, iteration of that element of the OID group MAY cease, reducing the number of variable bindings used in subsequent GetNext operations. In this case, the results of URI data access for the SNMP URI will not consist entirely of OID-group-sized sets of variable bindings. Even if this does not occur, the last variable binding returned for each member of the OID group will generally contain an SNMP "endOfMibView" exception or an OID for which the corresponding URI OID is not a lexical prefix.

4.3. OID Groups in SNMP URIs

Parenthesized OID groups in SNMP URIs are intended to support MIB object instances for which access via a single SNMP operation is required to ensure consistent results. Therefore, the OIDs within an OID group in an SNMP URI SHOULD be accessed by a single SNMP operation containing a variable binding corresponding to each OID in the group. A specific example involves the InetAddress and InetAddressType textual conventions defined in [RFC4001], for which the format of an InetAddress instance is specified by an associated InetAddressType instance. If two such associated instances are read via separate SNMP operations, the resulting values could be inconsistent (e.g., due to an intervening Set), causing the InetAddress value to be interpreted incorrectly.

This single operation requirement ("SHOULD") also applies to each OID group resulting from iterated access for an SNMP URI with a ".*" suffix. When members of an SNMP URI OID group differ in the number of OIDs for which each is a lexical prefix, this iteration may overrun by returning numerous variable bindings for which the corresponding OID in the OID group is not a lexical prefix. Such overrun can be avoided by using relative references within the same context (i.e., ./<oid>.*) when it is not important to access multiple MIB object instances in a single SNMP operation.

4.4. Interoperability Considerations

This document defines a transport-independent "snmp" scheme that is intended to accommodate SNMP transports other than UDP. UDP is the default transport for access to information specified by an SNMP URI for backward compatibility with existing usage, but other transports MAY be used. If more than one transport can be used (e.g., SNMP over TCP [RFC3430] in addition to SNMP over UDP), the information or SNMP service access designated by an SNMP URI SHOULD NOT depend on which transport is used (for SNMP over TCP, this is implied by Section 2 of [RFC3430]).

An SNMP URI designates use of SNMPv3 as specified by [RFC3416], [RFC3417], and related documents, but older versions of SNMP MAY be used in accordance with [RFC3584] when usage of such older versions is unavoidable. For SNMPv1 and SNMPv2c, the securityName, contextName, and contextEngineID elements of an SNMP URI are mapped to/from the community name, as described in [RFC3584]. When the community name is kept secret as a weak form of authentication, this mapping should be configured so that these three elements do not reveal information about the community name. If this is not done, then any SNMP URI component that would disclose significant information about a secret community name SHOULD be omitted. Note

that some community names contain reserved characters (e.g., "@") that require percent encoding when they are used in an SNMP URI. SNMP versions (e.g., v3) have been omitted from the SNMP URI scheme to permit use of older versions of SNMP, as well as any possible future successor to SNMPv3.

5. Examples

```
snmp://example.com
```

This example designates the default SNMP context at the SNMP agent at port 161 of host example.com .

```
snmp://tester5@example.com:8161
```

This example designates the default SNMP context at the SNMP agent at port 8161 of host example.com and indicates that the SNMP securityName "tester5" is to be used to access that agent. A possible reason to use a non-standard port is for testing a new version of SNMP agent code.

```
snmp://example.com/bridge1
```

This example designates the "bridge1" SNMP context at example.com. Because the contextEngineID component of the URI is omitted, there SHOULD be at most one SNMP context engine at example.com that supports the "bridge1" context.

```
snmp://example.com/bridge1;800002b804616263
```

This example designates the "bridge1" context at snmp.example.com via the SNMP context engine 800002b804616263 (string representation of a hexadecimal value). This avoids ambiguity if any other context engine supports a "bridge1" context. The above two examples are based on the figure in Section 3.3 of [RFC3411].

```
snmp://example.com//1.3.6.1.2.1.1.3.0
snmp://example.com//1.3.6.1.2.1.1.3+
snmp://example.com//1.3.6.1.2.1.1.3.*
```

These three examples all designate the sysUpTime.0 object instance in the SNMPv2-MIB or RFC1213-MIB for the default SNMP context ("") at example.com as sysUpTime.0 is:

a) designated directly by OID 1.3.6.1.2.1.1.3.0,

b) the lexically next MIB object instance after the OID 1.3.6.1.2.1.1.3, and

c) the only MIB object instance whose OID has 1.3.6.1.2.1.1.3 as a lexical prefix.

These three examples are provided for illustrative purposes only, as multiple syntactically distinct URIs SHOULD NOT be used to designate the same MIB object instance, in order to avoid unexpected results in URI-based systems that use string comparison to test URIs for equality.

snmp://example.com/bridge1/1.3.6.1.2.1.2.2.1.8.*

This example designates the ifOperStatus column of the IF-MIB in the bridge1 SNMP context at example.com.

snmp://example.com//(1.3.6.1.2.1.2.2.1.7,1.3.6.1.2.1.2.2.1.8).*

This example designates all (ifAdminStatus, ifOperStatus) pairs in the IF-MIB in the default SNMP context at example.com.

6. Security Considerations

An intended use of this URI scheme is designation of the location of management access to communication devices. Such location information may be considered sensitive in some environments, making it important to control access to this information and possibly even to encrypt it when it is sent over the network. All uses of this URI scheme should provide security mechanisms appropriate to the environments in which such uses are likely to be deployed.

The SNMP architecture includes control of access to management information (see Section 4.3 of [RFC3411]). An SNMP URI does not contain sufficient security information to obtain access in all situations, as the SNMP URI syntax is incapable of encoding SNMP securityModels, SNMP securityLevels, and credential or keying information for SNMP securityNames. Other means are necessary to provide such information; one possibility is out-of-band pre-configuration of the SNMP manager, as shown in the diagrams in Section 2.

By itself, the presence of a securityName in an SNMP URI does not authorize use of that securityName to access management information. Instead, the SNMP manager SHOULD match the securityName in the URI to an SNMP securityName and associated security information that have been pre-configured for use by the manager. If an SNMP URI contains a securityName that the SNMP manager is not provisioned to use, SNMP operations for that URI SHOULD NOT be generated.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example, via use of IPsec), there is no control over who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in MIB modules. It is RECOMMENDED that implementers consider the security features provided by the SNMPv3 framework (see [RFC3410], Section 8, for an overview), including full support for SNMPv3 cryptographic mechanisms (for authentication and privacy). This is of additional importance for MIB elements considered sensitive or vulnerable because GETs have side effects.

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to a MIB module instance is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (read/change/create/delete) them.

6.1. SNMP URI to SNMP Gateway Security Considerations

Additional security considerations apply to SNMP gateways that generate SNMP operations for SNMP URIs and return the results to clients (see Section 2) because management information is communicated beyond the SNMP framework. In general, an SNMP gateway should have some knowledge of the structure and function of the management information that it accesses via SNMP. Among other benefits, this allows an SNMP gateway to avoid SNMP access control failures because the gateway can reject an SNMP URI that will cause such failures before generating any SNMP operations.

SNMP gateways SHOULD impose authorization or access-control checks on all clients. If an SNMP gateway does not impose authorization or access controls, the gateway MUST NOT automatically obtain or use SNMP authentication material for arbitrary securityNames, as doing so would defeat SNMP's access controls. Instead, all SNMP gateways SHOULD authenticate each client and check the client's authorization to use a securityName in an SNMP URI before using the securityName on behalf of that client.

An SNMP gateway is also responsible for ensuring that all of its communication is appropriately secured. Specifically, an SNMP gateway SHOULD ensure that communication of management information with any client is protected to at least the SNMP securityLevel used for the corresponding SNMP access (see Section 3.4.3 of [RFC3411] for more information on securityLevel). If the client provides SNMP security information, the SNMP gateway SHOULD authenticate the client and SHOULD ensure that an authenticated cryptographic integrity check

is used for that communication to prevent modification of the security information. In addition, if a client provides any key or secret, the SNMP gateway SHOULD ensure that encryption is used in addition to the integrity check for that communication to prevent disclosure of keys or secrets.

There are management objects defined in SNMP MIBs whose MAX-ACCESS is read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. SNMP gateway support for SNMP SET operations in a non-secure environment without proper protection can have a negative effect on network operations. The individual MIB module specifications, and especially their security considerations, should be consulted for further information.

Some readable objects in some MIB modules (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET access to these objects via an SNMP gateway and possibly to even encrypt the values of these objects when they are sent over the network. The individual MIB module specifications, and especially their security considerations, should be consulted for further information. This consideration also applies to objects for which read operations have side effects.

7. IANA Considerations

The IANA has registered the URL registration template found in Appendix A in accordance with [RFC2717].

8. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[RFC3061] Mealling, M., "A URN Namespace of Object Identifiers", RFC 3061, February 2001.

[RFC3411] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.

[RFC3416] Presuhn, R., "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3416, December 2002.

[RFC3417] Presuhn, R., "Transport Mappings for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3417, December 2002.

[RFC3584] Frye, R., Levi, D., Routhier, S., and B. Wijnen, "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework", BCP 74, RFC 3584, August 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

9. Informative References

[RFC1738] Berners-Lee, T., Masinter, L., and M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, December 1994.

[RFC1900] Carpenter, B. and Y. Rekhter, "Renumbering Needs Work", RFC 1900, February 1996.

[RFC2717] Petke, R. and I. King, "Registration Procedures for URL Scheme Names", BCP 35, RFC 2717, November 1999.

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC3430] Schoenwaelder, J., "Simple Network Management Protocol Over Transmission Control Protocol Transport Mapping", RFC 3430, December 2002.

[RFC3617] Lear, E., "Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP)", RFC 3617, October 2003.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

10. Acknowledgements

Portions of this document were adapted from Eliot Lear's TFTP URI scheme specification [RFC3617]. Portions of the security considerations were adapted from the widely used security considerations "boilerplate" for MIB modules. Comments from Ted Hardie, Michael Mealing, Larry Masinter, Frank Strauss, Bert Wijnen, Steve Bellovin, the mreview@ops.ietf.org mailing list and the uri@w3c.org mailing list on earlier versions of this document have resulted in significant improvements and are gratefully acknowledged.

Appendix A. Registration Template

URL scheme name: snmp
URL scheme syntax: Section 3
Character encoding considerations: Section 3
Intended usage: Sections 1 and 2
Applications and/or protocols which use this scheme: SNMP, all versions, see [RFC3410] and [RFC3584]. Also SNMP over TCP, see [RFC3430].
Interoperability considerations: Section 4.4
Security considerations: Section 6
Relevant publications: See [RFC3410] for list. Also [RFC3430] and [RFC3584].
Contact: David L. Black, see below
Author/Change Controller: IESG

Authors' Addresses

David L. Black
EMC Corporation
176 South Street
Hopkinton, MA 01748

Phone: +1 (508) 293-7953
EMail: black_david@emc.com

Keith McCloghrie
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA USA 95134

Phone: +1 (408) 526-5260
EMail: kzm@cisco.com

Juergen Schoenwaelder
International University Bremen
P.O. Box 750 561
28725 Bremen
Germany

Phone: +49 421 200 3587
EMail: j.schoenwaelder@iu-bremen.de

Full Copyright Statement

Copyright (C) The Internet Society (2005).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM 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.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

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