Network Working Group
Request for Comments: 5675
Category: Standards Track

V. Marinov
J. Schoenwaelder
Jacobs University Bremen

October 2009

Mapping Simple Network Management Protocol (SNMP) Notifications to SYSLOG Messages

Abstract

This memo defines a mapping from Simple Network Management Protocol (SNMP) notifications to SYSLOG messages.

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

Table of Contents

1. Introduction

SNMP and SYSLOG are two widely used protocols to communicate event notifications. Although co-existence of several management protocols in one operational environment is possible, certain environments require that all event notifications be collected by a single system daemon, such as a SYSLOG collector or an SNMP notification receiver, via a single management protocol. In such environments, it is necessary to translate event notifications between management protocols.

The latest version of SYSLOG, specified in [RFC5424], supports a structured data element format. Structured data elements allow us to map between SNMP notifications and SYSLOG messages without losing information. In this memo, we specify a concrete mapping from SNMP event notifications [RFC3416] into SYSLOG messages [RFC5424]. We specify how the SYSLOG message format should be utilized to carry the information contained in an SNMP notification message. A new SYSLOG structured data element is defined, which carries the PDU portion of an SNMP notification message.

1.1. Conventions

A system that has the capability of receiving SNMP notification messages from an SNMP notification originator and sending the SNMP data contained inside in a SYSLOG message format to a SYSLOG collector is referred to in this memo as an "SNMP-to-SYSLOG translator". By definition, such a system should have an SNMP

notification receiver application and a SYSLOG originator running in order to be able to perform the functions of an "SNMP-to-SYSLOG translator".

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. Background

2.1. SNMP Notifications

A detailed introduction to the SNMP Management Framework can be found in [RFC3410]. The SNMP Management Architecture is described in [RFC3411]. Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB [RFC3418]. Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI) [RFC2578].

An SNMP notification message is generated and transmitted by an SNMP entity on behalf of a notification originator application [RFC3413]. SNMP notifications are often used to notify a notification receiver application at a logically remote SNMP entity that an event has occurred or that a certain condition is present. There are two types of SNMP protocol operations that are associated with SNMP notification messages [RFC3416]:

- SNMPv2-Trap-PDU, an unconfirmed notification delivery mechanism
- InformRequest-PDU, a confirmed notification delivery mechanism

The scopedPDU portion of an SNMPv3 trap or inform message has the following format [RFC3412]:

```
ScopedPDU ::= SEQUENCE {
    contextEngineID  OCTET STRING,
    contextName      OCTET STRING,
    data             ANY -- e.g., PDUs as defined in [RFC3416]
}
```

The data member of the SEQUENCE ScopedPDU carries an SNMPv2-Trap-PDU or an InformRequest-PDU. They both have the same structure:

```
PDUs ::= [7] IMPLICIT SEQUENCE {
    request-id           INTEGER,
    error-status         INTEGER,    -- ignored in notifications
    error-index          INTEGER,    -- ignored in notifications
    variable-bindings    VarBindList
}

-- variable binding

VarBind ::= SEQUENCE {
    name ObjectName,

    CHOICE {
        value          ObjectSyntax,
        unSpecified    NULL,    -- in retrieval requests
                                -- exceptions in responses
        noSuchObject   [0] IMPLICIT NULL,
        noSuchInstance [1] IMPLICIT NULL,
        endOfMibView   [2] IMPLICIT NULL
    }
}

-- variable-binding list

VarBindList ::= SEQUENCE (SIZE (0..max-bindings)) OF VarBind
```

The first two variable bindings in the variable binding list of an SNMPv2-Trap-PDU or InformRequest-PDU are sysUpTime.0 [RFC3418] and snmpTrapOID.0 [RFC3418], respectively. If the OBJECTS clause is present in the invocation of the corresponding NOTIFICATION-TYPE macro, then each corresponding variable, as instantiated by this notification, is copied, in order, to the variable-bindings field. If any additional variables are being included (at the option of the generating SNMP entity), then each is copied to the variable-bindings field.

In the case of SNMPv1 or SNMPv2c notifications, the contextEngineID and the contextName parameters are not present in notification messages.

This document assumes that notifications are in the format defined in [RFC3416]. Notifications in the SNMPv1 notification format MUST be translated as described in Section 3.1 of [RFC3584].

2.2. SYSLOG Notifications

The SYSLOG protocol is defined in [RFC5424]. The message contains a global header and a number of structured data elements. The ABNF [RFC5234] representation of a SYSLOG message is defined in RFC 5424 [RFC5424]. The relevant productions for structured data elements are:

```
STRUCTURED-DATA = NILVALUE / 1*SD-ELEMENT
SD-ELEMENT      = "[" SD-ID *(SP SD-PARAM) "]"
SD-PARAM        = PARAM-NAME "=" %d34 PARAM-VALUE %d34
SD-ID           = SD-NAME
PARAM-NAME      = SD-NAME
PARAM-VALUE     = UTF-8-STRING ; characters '"', '\' and
                               ; ']' MUST be escaped.
SD-NAME         = 1*32PRINTUSASCII
                  ; except '=', SP, ']', %d34 (")

UTF-8-STRING    = *OCTET ; Any VALID UTF-8 String
                  ; "shortest form" MUST be used

OCTET           = %d00-255
SP              = %d32
PRINTUSASCII    = %d33-126
NILVALUE        = "-"
```

3. Mapping SNMP Notifications to SYSLOG Messages

In this section, we define how the scopedPDU portion from an SNMP notification message is used to generate a message in the SYSLOG format. The notification receiver application at the SNMP-to-SYSLOG translator is listening for incoming notifications. After a notification is received by the SNMP engine, the data portion is forwarded to the notification receiver application. The data portion contains the scopedPDU of the message, which is used by the SYSLOG originator on the SNMP-to-SYSLOG translator to generate a SYSLOG message and send it to a SYSLOG collector (or proxy). Note that every SNMP notification maps to exactly one SYSLOG message.

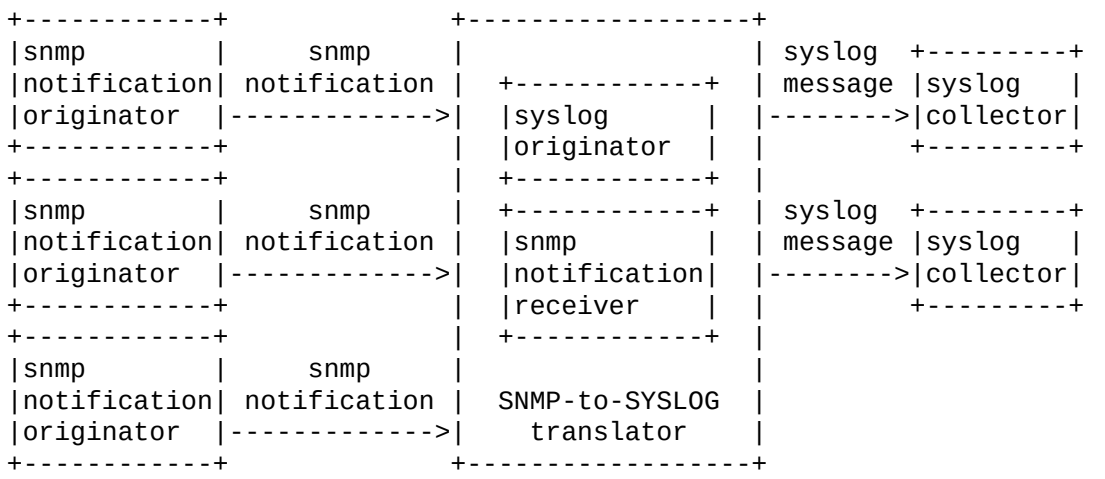

Figure 1: SNMP-to-SYSLOG Translator Deployment

A common deployment scenario is shown in Figure 1. There can be many SNMP notification originators that send SNMP event notifications to an SNMP-to-SYSLOG translator. The SNMP-to-SYSLOG translator extracts the data portion of the notification, generates a SYSLOG message, and sends the SYSLOG message to a SYSLOG collector, which is responsible for collecting and storing all notification messages. The arrows in Figure 1 indicate message flows, not individual messages.

The SNMP-to-SYSLOG translator is not transparent for a SYSLOG collector. The global header of the SYSLOG message generated by the SNMP-to-SYSLOG translator is filled with parameters that are specific for the system running the SNMP-to-SYSLOG translator, such as its hostname, timestamp, etc. The data portion (scopedPDU for SNMPv3 or PDU for SNMPv1/SNMPv2c) of the SNMP notification message is contained in the structured data of the SYSLOG message.

Implementations MUST drop invalid SNMP messages before they are passed to the SNMP-to-SYSLOG translator.

3.1. SYSLOG Header

The SNMP-to-SYSLOG translator fills the HEADER field of a SYSLOG message with parameters specific to the system on which it is running. The default facility level for SYSLOG messages containing SNMP notifications SHOULD be 3, which corresponds to messages generated by system daemons. The default severity level SHOULD be 5, which corresponds to "Notice: normal but significant condition". If the SNMP-to-SYSLOG translator has a notion of the type of notification that has been received, it might choose other values for facility and severity level.

The VERSION, TIMESTAMP, HOSTNAME, APP-NAME, PROCID, and MSGID fields in the SYSLOG message header are filled with values that are specific to the system on which the SNMP-to-SYSLOG translator is running. The character set used in the HEADER MUST be seven-bit ASCII in an eight-bit field, as described in [RFC5424].

3.2. Structured Data

The STRUCTURED-DATA field of a SYSLOG message carries the ScopedPDU (or PDU) portion of an SNMP notification message. For the purpose of carrying SNMP notification data, a new SD-ID element is defined. The ABNF [RFC5234] representation of the new structured element is:

```
SNMP-SD-ELEMENT = "[" SNMP-SD-ID [CTX] *VARBIND "]"
SNMP-SD-ID      = %x73.6E.6D.70          ; snmp
CTX             = CTXENGINE CTXNAME
CTXENGINE       = SP "ctxEngine=" %d34 HEXSTRING %d34
CTXNAME         = SP "ctxName=" %d34 PARAM-VALUE %d34
VARBIND         = SP VARNAME [SP VARLABEL] SP VARVALUE [SP VALSTRING]
VARNAME         = %d118 NUM "=" %d34 OID %d34          ; "vN="
VARLABEL        = %d108 NUM "=" %d34 PARAM-VALUE %d34 ; "lN="
VARVALUE        = VALOID / VALHEXSTRING / VALCOUNTER32 / VALCOUNTER64
                  / VALUNSIGNED32 / VALINTEGER32 / VALIP / VALNULL
                  / VALOPAQUE / VALTIMETICKS / VALSTRING

VALOID          = %d111 NUM "=" %d34 OID %d34          ; "oN="
VALHEXSTRING    = %d120 NUM "=" %d34 HEXSTRING %d34    ; "xN="
VALCOUNTER32    = %d99  NUM "=" %d34 UNSIGNED32 %d34   ; "cN="
VALCOUNTER64    = %d67  NUM "=" %d34 UNSIGNED64 %d34   ; "CN="
VALUNSIGNED32   = %d117 NUM "=" %d34 UNSIGNED32 %d34   ; "uN="
VALINTEGER32    = %d100 NUM "=" %d34 INTEGER32 %d34    ; "dN="
VALIP           = %d105 NUM "=" %d34 IPV4ADDRESS %d34 ; "iN="
VALNULL         = %d110 NUM "=" %d34 %d34              ; "nN="
VALOPAQUE       = %d112 NUM "=" %d34 HEXSTRING %d34    ; "pN="
VALTIMETICKS    = %d116 NUM "=" %d34 UNSIGNED32 %d34   ; "tN="
VALSTRING       = %d97  NUM "=" %d34 PARAM-VALUE %d34 ; "aN="

NUM             = NONZERODIGIT 0*DIGIT

OID             = OIDSTART *("." OIDSUBID)
OIDSTART        = (("0." / "1.") [%d49-51] DIGIT) / ("2." OIDSUBID)
OIDSUBID        = ZERO / (NONZERODIGIT *DIGIT)

PARAM-VALUE     = UTF-8-STRING ; characters '"', '\' and
                               ; ']' MUST be escaped.
UTF-8-STRING    = *OCTET ; Any VALID UTF-8 String
                         ; "shortest form" MUST be used
HEXSTRING       = *HEX
```

```
INTEGER32       = ["-"] NONZERODIGIT 0*DIGIT
UNSIGNED32      = NONZERODIGIT 0*DIGIT
UNSIGNED64      = NONZERODIGIT 0*DIGIT
IPV4ADDRESS     = d8 "." d8 "." d8 "." d8

d8              = DIGIT               ; 0-9
                  / %d49-57 DIGIT       ; 10-99
                  / "1" 2DIGIT          ; 100-199
                  / "2" %d48-52 DIGIT   ; 200-249
                  / "25" %d48-53        ; 250-255

HEX             = DIGIT / %x41-46 / %x61-66       ; 0-9 / A-F / a-f
NONZERODIGIT    = %d49-57
ZERO            = %d48
DIGIT           = ZERO / NONZERODIGIT
SP              = %d32
```

Each SNMP-SD-ELEMENT starts with the SD-ID "snmp". The first two SD-ID parameters are "ctxEngine" and "ctxName". The context MUST be present in an SNMPv3 notification and therefore "ctxEngine" and "ctxName" MUST be present in a SYSLOG message generated by an SNMP-to-SYSLOG translator from an SNMPv3 notification. The contextEngineID is encoded as an hexadecimal string while the contextName is encoded as a UTF8 string.

The remaining parameters in the "snmp" SD-ID correspond to the varbind list elements contained in the SNMP PDU. The name of a varbind is encoded as an OID in dotted notation. The rendered OID is carried in a "vN" parameter, where N identifies the position of the varbind in the varbind list of the SNMP message (the first varbind having the position 1). A MIB-aware implementation may in addition generate a parameter "lN" carrying the descriptor of the associated MIB object plus the instance identifier suffix (also called an OID label). The number N again identifies the position of the varbind in the varbind list of the SNMP message.

The value of a varbind is encoded depending on its type according to the rules shown in Table 1, and type-specific parameter names are used to convey the type information. The number N again identifies the position of the varbind in the varbind list of the SNMP message. A MIB-aware implementation may in addition generate a parameter "aN" carrying an alternate textual representation of the value, which is obtained by applying DISPLAY-HINTs and translating named numbers into corresponding labels or OBJECT IDENTIFIER values to descriptors. For SNMP object types that have a DISPLAY-HINT of the form 'Ma' or 'Mt', where M is some number, a MIB-aware implementation can choose to include the "aN" parameter and to suppress the corresponding "xN" parameter. This special case saves space for textual objects. A

receiver receiving an "aN" parameter without a matching value at position N can unambiguously convert the value carried in the "aN" parameter back to an OCTET STRING value.

While the inclusion of additional parameters carrying OID labels or alternate value representations increases human readability, this comes at the cost of increased message size, which may cause truncation of SYSLOG messages. Therefore, implementations SHOULD provide a configuration mechanism to enable/disable the generation of parameters carrying OID labels or alternate value representations.

SNMP Type	PARAM-NAME	Value Encoding
OBJECT IDENTIFIER	oN	dotted-decimal notation
OCTET STRING	xN	hexadecimal string
Counter32	cN	unsigned decimal number
Counter64	CN	unsigned decimal number
Unsigned32	uN	unsigned decimal number
INTEGER, Integer32	dN	signed decimal number
IpAddress	iN	dotted quad notation
Opaque	pN	hexadecimal (BER) string
TimeTicks	tN	unsigned decimal number
NULL	nN	zero-length string

Table 1: Mapping of SNMP Types to SD Params

The SYSLOG message generated by the SNMP-to-SYSLOG translator may, in addition to the SNMP-SD-ELEMENT, include other structured data elements in its structured data part. These additional structured data elements MUST comply with the specification in [RFC5424].

In particular, the parameters in the "origin" SD-ID SHOULD identify the originator of the SNMP notification. A suitable value for the "ip" parameter MAY be taken from the snmpTrapAddress varbind if present, and a suitable value for the "enterpriseId" parameter MAY be extracted from the snmpTrapOID varbind.

3.3. MSG Data

The MSG part of the SYSLOG message is optional and may contain a free-form message that provides a textual description of the SNMP event notification. According to [RFC5424], the character set used in MSG SHOULD be Unicode, encoded using UTF-8 as specified in [RFC3629]. If the originator cannot encode the MSG in Unicode, it

MAY use any other encoding. The originator MAY use the "language" parameters defined in [RFC5424] to convey information about the natural language used inside MSG.

4. Relationship to the SYSLOG-MSG-MIB

A companion document [RFC5676] defines an SNMP MIB module to represent SYSLOG messages and to send SYSLOG messages as SNMP notifications to SNMP notification receivers. This section discusses the possibilities of using both specifications in combination.

A SYSLOG collector implementing the SYSLOG-MSG-MIB module and the mapping of SNMP notifications to SYSLOG messages may be configured to translate received SYSLOG messages containing SNMP notifications back into the original SNMP notification. In this case, the relevant tables of the SYSLOG-MSG-MIB will not be populated for SYSLOG messages carrying SNMP notifications. This configuration allows operators to build a forwarding chain where SNMP notifications are "tunneled" through SYSLOG messages. Due to size restrictions of the SYSLOG transports and the more verbose textual encoding used by SYSLOG, there is a possibility that SNMP notification content will get truncated when tunneled through SYSLOG, and thus the resulting SNMP notification may be incomplete.

An SNMP management application supporting the SYSLOG-MSG-MIB and the mapping of SNMP notifications to SYSLOG messages may process information from the SYSLOG-MSG-MIB in order to emit a SYSLOG message representing the SYSLOG message recorded in the SYSLOG-MSG-MIB module. This configuration allows operators to build a forwarding chain where SYSLOG messages are "tunneled" through SNMP messages. A notification receiver can determine whether a syslogMsgNotification contained all structured data element parameters of a SYSLOG message. In case parameters are missing, a forwarding application MUST retrieve the missing parameters from the SYSLOG-MSG-MIB. Regular polling of the SYSLOG-MSG-MIB can be used to take care of any lost SNMP notifications.

5. Usage Example

Here we provide an example of how an SNMP linkUp trap message is mapped into a SYSLOG message by using the mappings defined in Section 3.1 and Section 3.2.

The linkUp notification is defined in [RFC2863] as follows:

```
linkUp NOTIFICATION-TYPE
    OBJECTS { ifIndex, ifAdminStatus, ifOperStatus }
    STATUS  current
```

```
      DESCRIPTION
         "A linkUp trap signifies that the SNMP entity, acting in an
         agent role, has detected that the ifOperStatus object for
         one of its communication links left the down state and
         transitioned into some other state (but not into the
         notPresent state).  This other state is indicated by the
         included value of ifOperStatus."
      ::= { snmpTraps 4 }
```

The scopedPDU portion of an SNMP linkUp trap sent using the SNMPv3 message format is shown below (the left column shows the Basic Encoding Rules (BER) encoding, while the right column indicates the corresponding ASN.1 definitions):

```
30:7C                                              SEQUENCE {
   04:08:80:00:02:B8:04:61:62:63                     800002b804616263
   04:04:63:74:78:31                                 "ctx1"
   A7:6A                                             SNMPv2-Trap-PDU {
      02:03:6D:08:67                                   INTEGER 7145575
      02:01:00                                         INTEGER 0
      02:01:00                                         INTEGER 0
      30:5D                                            SEQUENCE OF {
         30:0F                                           SEQUENCE {
            06:08:2B:06:01:02:01:01:03:00                  sysUpTime.0
            43:03:01:72:8C                                 94860 }
         30:17                                           SEQUENCE {
            06:0A:2B:06:01:06:03:01:01:04:01:00            snmpTrapOID.0
            06:09:2B:06:01:06:03:01:01:05:04               linkUp }
         30:0F                                           SEQUENCE {
            06:0A:2B:06:01:02:01:02:02:01:01:03            ifIndex.3
            02:01:03                                       3 }
         30:0F                                           SEQUENCE {
            06:0A:2B:06:01:02:01:02:02:01:07:03            ifAdminStatus.3
            02:01:01                                       up(1) }
         30:0F                                           SEQUENCE {
            06:0A:2B:06:01:02:01:02:02:01:08:03            ifOperStatus.3
            02:01:01                                       up(1) } } } }
```

The corresponding SYSLOG message generated by the SNMP-to-SYSLOG translator is shown below. (SYSLOG examples should be considered to be on one line. They are wrapped on multiple lines in this document for readability purposes only.)

```
<29>1 2003-10-11T22:14:15.003Z mymachine.example.com snmptrapd - ID47
  [snmp ctxEngine="800002b804616263" ctxName="ctx1"
    v1="1.3.6.1.2.1.1.3.0" l1="sysUpTime.0" d1="94860"
    v2="1.3.6.1.6.3.1.1.4.1.0" l2="snmpTrapOID.0"
      o2="1.3.6.1.6.3.1.1.5.4" a2="linkUp"
```

```
    v3="1.3.6.1.2.1.2.2.1.1.3" d3="3"
    v4="1.3.6.1.2.1.2.2.1.7.3" d4="1" a4="up"
    v5="1.3.6.1.2.1.2.2.1.8.3" d5="1" a5="up"]
```

The corresponding SYSLOG message has a priority value of 29, which means a facility level of 3 (system daemons) and a severity level of 5 (Notice: normal but significant condition) according to the algorithm for calculation of priority value specified in Section 6.2.1 of [RFC5424]. The rest of the fields in the header of the SYSLOG message are parameters that are specific to the system running the SNMP-to-SYSLOG translator. The SYSLOG version is 1 and the message was generated at 22:14:15.003Z on 2003-10-11T by the host "mymachine.example.com". The application on the SNMP-to-SYSLOG translator that generated the message was "snmptrapd"; there is no information about the process id, and the message on the SNMP-to-SYSLOG system is identified with the MSGID of ID47.

The SYSLOG message contains one structured data element with an SD-ID of "snmp", which means that this is the scopedPDU portion of an SNMP event notification message. The data that is contained in the notification is associated with the ContextEngineID "123456" and ContextName "ctx1". The request-id of the SNMP notification message was "7145575". Then follows the data portion of the scopedPDU. The first two variables contained in the data portion are always the sysUpTime.0 and snmpTrapOID.0. An snmpTrapOID.0 with a value of "1.3.6.1.6.3.1.1.5.4" means that this is a linkUp trap. The parameters v3="1.3.6.1.2.1.2.2.1.1.3" d3="3" mean that the SNMP notification message is carrying the ifIndex object, which has a type INTEGER and a value of 3. The parameters v4="1.3.6.1.2.1.2.2.1.7.3" d4="1" mean that the SNMP notification message is carrying the object ifAdminStatus, which has a type INTEGER and a value of 1. The parameters v5="1.3.6.1.2.1.2.2.1.8.3" d5="1" mean that the SNMP notification message is carrying the object ifOperStatus, which has a type INTEGER and a value of "1".

6. IANA Considerations

IANA registered the SD-ID value "snmp" together with the PARAM-NAME values specified in Section 3.2 in the registry for SYSLOG Structured Data ID Values according to Section 9 in [RFC5424]. The notation <N> indicates a position number.

SD-ID	PARAM-NAME	
snmp		OPTIONAL
	ctxEngine	OPTIONAL
	ctxName	OPTIONAL
	v<N>	OPTIONAL
	l<N>	OPTIONAL

o<N>	OPTIONAL
x<N>	OPTIONAL
c<N>	OPTIONAL
C<N>	OPTIONAL
u<N>	OPTIONAL
d<N>	OPTIONAL
i<N>	OPTIONAL
n<N>	OPTIONAL
p<N>	OPTIONAL
t<N>	OPTIONAL
a<N>	OPTIONAL

7. Security Considerations

The security considerations discussed in [RFC5424] apply to this document.

The SNMP architecture supports an access control mechanism, ensuring that SNMP notifications are only sent to receivers who are authorized to receive the notification. Network operators using this mapping of SNMP notifications to SYSLOG messages should enforce a consistent policy, preventing people from accessing SNMP notifications via the SYSLOG mapping that would otherwise not be accessible.

8. Acknowledgments

The editors wish to thank the following individuals for providing helpful comments on various versions of this document: Martin Bjorklund, Washam Fan, Rainer Gerhards, Tom Petch, and Dan Romascanu.

9. References

9.1. Normative References

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

[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.

[RFC3412] Case, J., Harrington, D., Presuhn, R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3412, December 2002.

[RFC3413] Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, December 2002.

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

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, 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.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 5234, January 2008.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, March 2009.

[RFC5676] Schoenwaelder, J., Clemm, A., and A. Karmakar, "Definitions of Managed Objects for Mapping SYSLOG Messages to Simple Network Management Protocol (SNMP) Notifications", RFC 5676, October 2009.

9.2. Informative References

[RFC2578] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIv2)", RFC 2578, STD 58, April 1999.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

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

Authors' Addresses

Vladislav Marinov
Jacobs University Bremen
Campus Ring 1
28725 Bremen
Germany

EMail: v.marinov@jacobs-university.de

Juergen Schoenwaelder
Jacobs University Bremen
Campus Ring 1
28725 Bremen
Germany

EMail: j.schoenwaelder@jacobs-university.de