                        The BSD syslog Protocol

Status of this Memo

Copyright Notice

Abstract

   This document describes the observed behavior of the syslog protocol.
   This protocol has been used for the transmission of event
   notification messages across networks for many years.  While this
   protocol was originally developed on the University of California
   Berkeley Software Distribution (BSD) TCP/IP system implementations,
   its value to operations and management has led it to be ported to
   many other operating systems as well as being embedded into many
   other networked devices.

Table of Contents

1. Introduction

   Since the beginning, life has relied upon the transmission of
   messages.  For the self-aware organic unit, these messages can relay
   many different things.  The messages may signal danger, the presence
   of food or the other necessities of life, and many other things.  In
   many cases, these messages are informative to other units and require
   no acknowledgement.  As people interacted and created processes, this
   same principle was applied to societal communications.  As an
   example, severe weather warnings may be delivered through any number
   of channels - a siren blowing, warnings delivered over television and
   radio stations, and even through the use of flags on ships.  The
   expectation is that people hearing or seeing these warnings would
   realize their significance and take appropriate action.  In most
   cases, no responding acknowledgement of receipt of the warning is
   required or even desired.  Along these same lines, operating systems,
   processes and applications were written to send messages of their own
   status, or messages to indicate that certain events had occurred.
   These event messages generally had local significance to the machine
   operators.  As the operating systems, processes and applications grew
   ever more complex, systems were devised to categorize and log these
   diverse messages and allow the operations staff to more quickly

differentiate the notifications of problems from simple status
messages.  The syslog process was one such system that has been
widely accepted in many operating systems.  Flexibility was designed
into this process so the operations staff have the ability to
configure the destination of messages sent from the processes running
on the device.  In one dimension, the events that were received by
the syslog process could be logged to different files and also
displayed on the console of the device.  In another dimension, the
syslog process could be configured to forward the messages across a
network to the syslog process on another machine. The syslog process
had to be built network-aware for some modicum of scalability since
it was known that the operators of multiple systems would not have
the time to access each system to review the messages logged there.
The syslog process running on the remote devices could therefore be
configured to either add the message to a file, or to subsequently
forward it to another machine.

In its most simplistic terms, the syslog protocol provides a
transport to allow a machine to send event notification messages
across IP networks to event message collectors - also known as syslog
servers.  Since each process, application and operating system was
written somewhat independently, there is little uniformity to the
content of syslog messages.  For this reason, no assumption is made
upon the formatting or contents of the messages.  The protocol is
simply designed to transport these event messages.  In all cases,
there is one device that originates the message.  The syslog process
on that machine may send the message to a collector.  No
acknowledgement of the receipt is made.

One of the fundamental tenets of the syslog protocol and process is
its simplicity.  No stringent coordination is required between the
transmitters and the receivers.  Indeed, the transmission of syslog
messages may be started on a device without a receiver being
configured, or even actually physically present.  Conversely, many
devices will most likely be able to receive messages without explicit
configuration or definitions.  This simplicity has greatly aided the
acceptance and deployment of syslog.

1.1 Events and Generated Messages

The writers of the operating systems, processes and applications have
had total control over the circumstances that would generate any
message.  In some cases, messages are generated to give status. These
can be either at a certain period of time, or at some other interval
such as the invocation or exit of a program.  In other cases, the
messages may be generated due to a set of conditions being met.  In
those cases, either a status message or a message containing an alarm
of some type may be generated.  It was considered that the writers of

the operating systems, processes and applications would quantify
their messages into one of several broad categories.  These broad
categories generally consist of the facility that generated them,
along with an indication of the severity of the message.  This was so
that the operations staff could selectively filter the messages and
be presented with the more important and time sensitive notifications
quickly, while also having the ability to place status or informative
messages in a file for later perusal.   Other options for displaying
or storing messages have been seen to exist as well.

Devices MUST be configured with rules for displaying and/or
forwarding the event messages.  The rules that have been seen are
generally very flexible.  An administrator may want to have all
messages stored locally as well as to have all messages of a high
severity forwarded to another device.  They may find it appropriate
to also have messages from a particular facility sent to some or all
of the users of the device and displayed on the system console.
However the administrator decides to configure the disposition of the
event messages, the process of having them sent to a syslog collector
generally consists of deciding which facility messages and which
severity levels will be forwarded, and then defining the remote
receiver.  For example, an administrator may want all messages that
are generated by the mail facility to be forwarded to one particular
event message collector.  Then the administrator may want to have all
kernel generated messages sent to a different syslog receiver while,
at the same time, having the critically severe messages from the
kernel also sent to a third receiver.  It may also be appropriate to
have those messages displayed on the system console as well as being
mailed to some appropriate people, while at the same time, being sent
to a file on the local disk of the device.  Conversely, it may be
appropriate to have messages from a locally defined process only
displayed on the console but not saved or forwarded from the device.
In any event, the rules for this will have to be generated on the
device.  Since the administrators will then know which types of
messages will be received on the collectors, they should then make
appropriate rules on those syslog servers as well.

The contents of a message have also been at the discretion of its
creator.  It has been considered to be good form to write the
messages so that they are informative to the person who may be
reading them.  It has also been considered good practice to include a
timestamp and some indication of the sending device and the process
that originated it in the messages.  However, none of those are
stringently required.

It should be assumed that any process on any device might generate an
event message.  This may include processes on machines that do not
have any local storage - e.g., printers, routers, hubs, switches, and

diskless workstations.  In that case, it may be imperative that event
messages are transported to a collector so that they may be recorded
and hopefully viewed by an operator.

1.2 Operations of the Message Receivers

   It is beyond the scope of this document to specify how event messages
   should be processed when they are received.  Like the operations
   described in Section 1.1, they generally may be displayed to the
   appropriate people, saved onto disk, further forwarded, or any
   combination of these.  The rules for determining the disposition of
   received messages have been seen to be identical to the rules for
   determining the disposition of locally generated messages.

   As a very general rule, there are usually many devices sending
   messages to relatively fewer collectors.  This fan-in process allows
   an administrator to aggregate messages into relatively few
   repositories.

2. Transport Layer Protocol

   syslog uses the user datagram protocol (UDP) [1] as its underlying
   transport layer mechanism.  The UDP port that has been assigned to
   syslog is 514.  It is RECOMMENDED that the source port also be 514 to
   indicate that the message is from the syslog process of the sender,
   but there have been cases seen where valid syslog messages have come
   from a sender with a source port other than 514.  If the sender uses
   a source port other than 514 then it is RECOMMENDED and has been
   considered to be good form that subsequent messages are from a single
   consistent port.

3. Definitions and Architecture

   The following definitions will be used in this document.

        A machine that can generate a message will be called a
        "device".

        A machine that can receive the message and forward it to
        another machine will be called a "relay".

        A machine that receives the message and does not relay it to
        any other machines will be called a "collector".  This has been
        commonly known as a "syslog server".

        Any device or relay will be known as the "sender" when it sends
        a message.

Any relay or collector will be known as the "receiver" when it
receives the message.

The architecture of the devices may be summarized as follows:

Senders send messages to relays or collectors with no knowledge
of whether it is a collector or relay.

Senders may be configured to send the same message to multiple
receivers.

Relays may send all or some of the messages that they receive
to a subsequent relay or collector.  In the case where they do
not forward all of their messages, they are acting as both a
collector and a relay.  In the following diagram, these devices
will be designated as relays.

Relays may also generate their own messages and send them on to
subsequent relays or collectors.  In that case it is acting as
a device.  These devices will also be designated as a relay in
the following diagram.

The following architectures shown in Diagram 1 are valid while the
first one has been known to be the most prevalent.  Other
arrangements of these examples are also acceptable.  As noted above,
in the following diagram relays may pass along all or some of the
messages that they receive along with passing along messages that
they internally generate.

```
        +------+              +---------+
        |Device|---->----|Collector|
        +------+              +---------+


        +------+              +-----+              +---------+
        |Device|---->----|Relay|---->----|Collector|
        +------+              +-----+              +---------+


        +------+      +-----+              +-----+      +---------+
        |Device|-->--|Relay|-->--..-->--|Relay|-->--|Collector|
        +------+      +-----+              +-----+      +---------+


        +------+              +-----+              +---------+
        |Device|---->----|Relay|---->----|Collector|
        |      |-\           +-----+              +---------+
        +------+  \
                   \          +-----+              +---------+
                    \-->--|Relay|---->----|Collector|
                          +-----+              +---------+


        +------+              +---------+
        |Device|---->----|Collector|
        |      |-\           +---------+
        +------+  \
                   \          +-----+              +---------+
                    \-->--|Relay|---->----|Collector|
                          +-----+              +---------+


        +------+              +-----+              +---------+
        |Device|---->----|Relay|---->-------|Collector|
        |      |-\           +-----+            /--|         |
        +------+  \                            /   +---------+
                   \          +-----+         /
                    \-->--|Relay|-->--/
                          +-----+
```
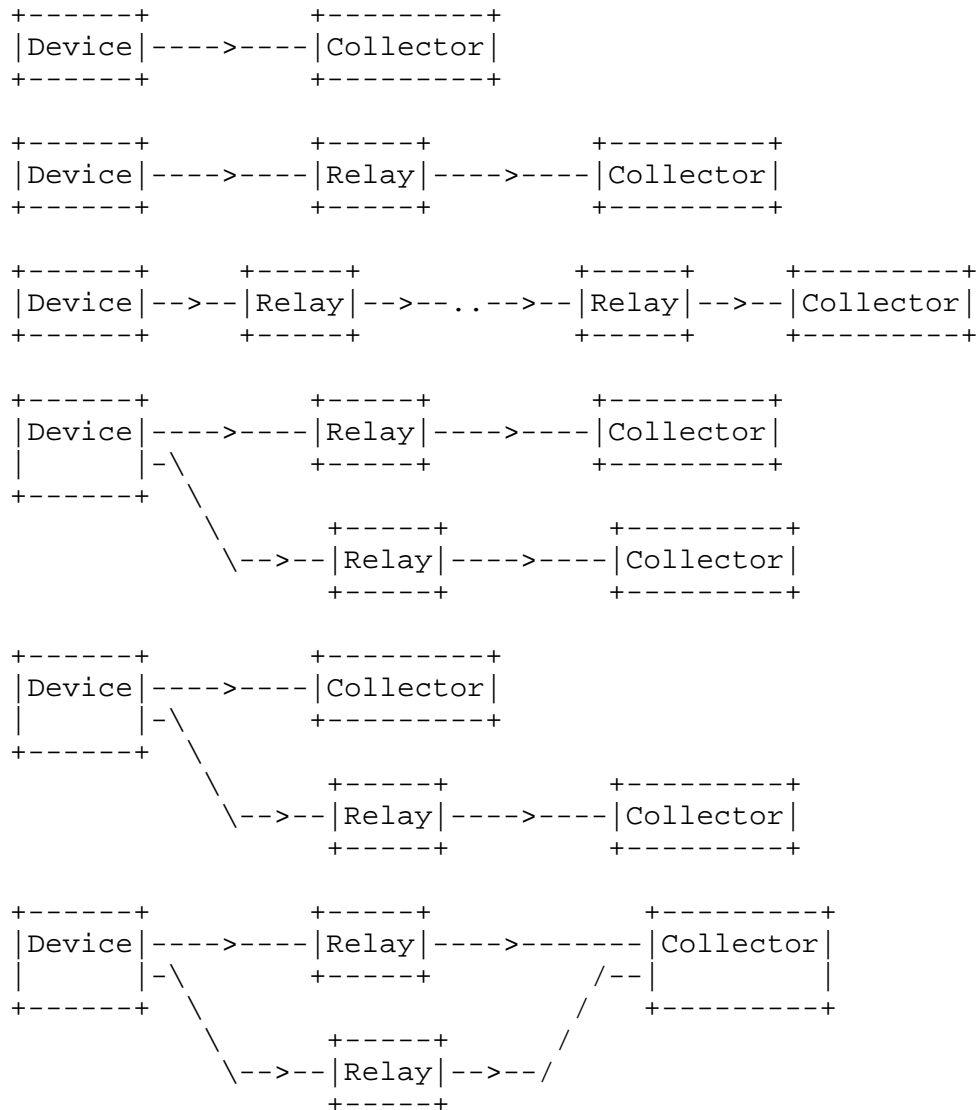
            Diagram 1.   Some Possible syslog Architectures


4. Packet Format and Contents

   The payload of any IP packet that has a UDP destination port of 514
   MUST be treated as a syslog message.   There MAY be differences
   between the format of an originally transmitted syslog message and
   the format of a relayed message.   In essence, it is RECOMMENDED to
   transmit a syslog message in the format specified in this document,
   but it is not required.   If a relay is able to recognize the message
   as adhering to that format then it MUST retransmit the message
   without making any changes to it.   However, if a relay receives a

message but cannot discern the proper implementation of the format,
it is REQUIRED to modify the message so that it conforms to that
format before it retransmits it.  Section 4.1 will describe the
RECOMMENDED format for syslog messages.  Section 4.2 will describe
the requirements for originally transmitted messages and Section 4.3
will describe the requirements for relayed messages.

4.1 syslog Message Parts

   The full format of a syslog message seen on the wire has three
   discernable parts.  The first part is called the PRI, the second part
   is the HEADER, and the third part is the MSG.  The total length of
   the packet MUST be 1024 bytes or less.  There is no minimum length of
   the syslog message although sending a syslog packet with no contents
   is worthless and SHOULD NOT be transmitted.

4.1.1 PRI Part

   The PRI part MUST have three, four, or five characters and will be
   bound with angle brackets as the first and last characters.  The PRI
   part starts with a leading "<" ('less-than' character), followed by a
   number, which is followed by a ">" ('greater-than' character). The
   code set used in this part MUST be seven-bit ASCII in an eight-bit
   field as described in RFC 2234 [2].  These are the ASCII codes as
   defined in "USA Standard Code for Information Interchange" [3].  In
   this, the "<" character is defined as the Augmented Backus-Naur Form
   (ABNF) %d60, and the ">" character has ABNF value %d62.  The number
   contained within these angle brackets is known as the Priority value
   and represents both the Facility and Severity as described below.
   The Priority value consists of one, two, or three decimal integers
   (ABNF DIGITS) using values of %d48 (for "0") through %d57 (for "9").

   The Facilities and Severities of the messages are numerically coded
   with decimal values.  Some of the operating system daemons and
   processes have been assigned Facility values.  Processes and daemons
   that have not been explicitly assigned a Facility may use any of the
   "local use" facilities or they may use the "user-level" Facility.
   Those Facilities that have been designated are shown in the following
   table along with their numerical code values.

        Numerical           Facility
          Code

           0                kernel messages
           1                user-level messages
           2                mail system
           3                system daemons
           4                security/authorization messages (note 1)

```
            5              messages generated internally by syslogd
            6              line printer subsystem
            7              network news subsystem
            8              UUCP subsystem
            9              clock daemon (note 2)
           10              security/authorization messages (note 1)
           11              FTP daemon
           12              NTP subsystem
           13              log audit (note 1)
           14              log alert (note 1)
           15              clock daemon (note 2)
           16              local use 0  (local0)
           17              local use 1  (local1)
           18              local use 2  (local2)
           19              local use 3  (local3)
           20              local use 4  (local4)
           21              local use 5  (local5)
           22              local use 6  (local6)
           23              local use 7  (local7)
```

Table 1.  syslog Message Facilities

Note 1 - Various operating systems have been found to utilize
   Facilities 4, 10, 13 and 14 for security/authorization,
   audit, and alert messages which seem to be similar.
Note 2 - Various operating systems have been found to utilize
   both Facilities 9 and 15 for clock (cron/at) messages.

Each message Priority also has a decimal Severity level indicator.
These are described in the following table along with their numerical
values.

```
     Numerical          Severity
       Code

        0       Emergency: system is unusable
        1       Alert: action must be taken immediately
        2       Critical: critical conditions
        3       Error: error conditions
        4       Warning: warning conditions
        5       Notice: normal but significant condition
        6       Informational: informational messages
        7       Debug: debug-level messages
```

Table 2. syslog Message Severities

The Priority value is calculated by first multiplying the Facility
number by 8 and then adding the numerical value of the Severity. For
example, a kernel message (Facility=0) with a Severity of Emergency
(Severity=0) would have a Priority value of 0.  Also, a "local use 4"
message (Facility=20) with a Severity of Notice (Severity=5) would
have a Priority value of 165.  In the PRI part of a syslog message,
these values would be placed between the angle brackets as <0> and
<165> respectively.  The only time a value of "0" will follow the "<"
is for the Priority value of "0". Otherwise, leading "0"s MUST NOT be
used.

4.1.2 HEADER Part of a syslog Packet

The HEADER part contains a timestamp and an indication of the
hostname or IP address of the device.  The HEADER part of the syslog
packet MUST contain visible (printing) characters.  The code set used
MUST also be seven-bit ASCII in an eight-bit field like that used in
the PRI part.  In this code set, the only allowable characters are
the ABNF VCHAR values (%d33-126) and spaces (SP value %d32).

The HEADER contains two fields called the TIMESTAMP and the HOSTNAME.
The TIMESTAMP will immediately follow the trailing ">" from the PRI
part and single space characters MUST follow each of the TIMESTAMP
and HOSTNAME fields.  HOSTNAME will contain the hostname, as it knows
itself.  If it does not have a hostname, then it will contain its own
IP address.  If a device has multiple IP addresses, it has usually
been seen to use the IP address from which the message is
transmitted.  An alternative to this behavior has also been seen.  In
that case, a device may be configured to send all messages using a
single source IP address regardless of the interface from which the
message is sent.  This will provide a single consistent HOSTNAME for
all messages sent from a device.

The TIMESTAMP field is the local time and is in the format of "Mmm dd
hh:mm:ss" (without the quote marks) where:

      Mmm is the English language abbreviation for the month of the
      year with the first character in uppercase and the other two
      characters in lowercase.  The following are the only acceptable
      values:

      Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec

      dd is the day of the month.  If the day of the month is less
      than 10, then it MUST be represented as a space and then the
      number.  For example, the 7th day of August would be
      represented as "Aug  7", with two spaces between the "g" and
      the "7".

      hh:mm:ss is the local time.  The hour (hh) is represented in a
      24-hour format.  Valid entries are between 00 and 23,
      inclusive.  The minute (mm) and second (ss) entries are between
      00 and 59 inclusive.

   A single space character MUST follow the TIMESTAMP field.

   The HOSTNAME field will contain only the hostname, the IPv4 address,
   or the IPv6 address of the originator of the message.  The preferred
   value is the hostname.  If the hostname is used, the HOSTNAME field
   MUST contain the hostname of the device as specified in STD 13 [4].
   It should be noted that this MUST NOT contain any embedded spaces.
   The Domain Name MUST NOT be included in the HOSTNAME field.  If the
   IPv4 address is used, it MUST be shown as the dotted decimal notation
   as used in STD 13 [5].  If an IPv6 address is used, any valid
   representation used in RFC 2373 [6] MAY be used.  A single space
   character MUST also follow the HOSTNAME field.

4.1.3 MSG Part of a syslog Packet

   The MSG part will fill the remainder of the syslog packet.  This will
   usually contain some additional information of the process that
   generated the message, and then the text of the message.  There is no
   ending delimiter to this part.  The MSG part of the syslog packet
   MUST contain visible (printing) characters.  The code set
   traditionally and most often used has also been seven-bit ASCII in an
   eight-bit field like that used in the PRI and HEADER parts.  In this
   code set, the only allowable characters are the ABNF VCHAR values
   (%d33-126) and spaces (SP value %d32).  However, no indication of the
   code set used within the MSG is required, nor is it expected.  Other
   code sets MAY be used as long as the characters used in the MSG are
   exclusively visible characters and spaces similar to those described
   above.  The selection of a code set used in the MSG part SHOULD be
   made with thoughts of the intended receiver.  A message containing
   characters in a code set that cannot be viewed or understood by a
   recipient will yield no information of value to an operator or
   administrator looking at it.

   The MSG part has two fields known as the TAG field and the CONTENT
   field.  The value in the TAG field will be the name of the program or
   process that generated the message.  The CONTENT contains the details
   of the message.  This has traditionally been a freeform message that
   gives some detailed information of the event.  The TAG is a string of
   ABNF alphanumeric characters that MUST NOT exceed 32 characters.  Any
   non-alphanumeric character will terminate the TAG field and will be
   assumed to be the starting character of the CONTENT field.  Most
   commonly, the first character of the CONTENT field that signifies the

conclusion of the TAG field has been seen to be the left square
bracket character ("["), a colon character (":"), or a space
character.  This is explained in more detail in Section 5.3.

4.2 Original syslog Packets Generated by a Device

   There are no set requirements on the contents of the syslog packet as
   it is originally sent from a device.  It should be reiterated here
   that the payload of any IP packet destined to UDP port 514 MUST be
   considered to be a valid syslog message.  It is, however, RECOMMENDED
   that the syslog packet have all of the parts described in Section 4.1
   - PRI, HEADER and MSG - as this enhances readability by the recipient
   and eliminates the need for a relay to modify the message.

   For implementers that do choose to construct syslog messages with the
   RECOMMENDED format, the following guidance is offered.

        If the originally formed message has a TIMESTAMP in the HEADER
        part, then it SHOULD be the local time of the device within its
        timezone.

        If the originally formed message has a HOSTNAME field, then it
        will contain the hostname as it knows itself.  If it does not
        have a hostname, then it will contain its own IP address.

        If the originally formed message has a TAG value, then that
        will be the name of the program or process that generated the
        message.

4.3 Relayed syslog Packets

   When a relay receives a packet, it will check for a valid PRI.  If
   the first character is not a less-than sign, the relay MUST assume
   that the packet does not contain a valid PRI.  If the 3rd, 4th, or
   5th character is not a right angle bracket character, the relay again
   MUST assume that the PRI was not included in the original message.
   If the relay does find a valid PRI part then it must check for a
   valid TIMESTAMP in the HEADER part.  From these rules, there will be
   three general cases of received messages.  Table 3 gives the general
   characteristics of these cases and lists the subsequent section of
   this document that describes the handling of that case.

             Case                                          Section
        Valid PRI and TIMESTAMP                             4.3.1
        Valid PRI but no TIMESTAMP or invalid TIMESTAMP     4.3.2
        No PRI or unidentifiable PRI                        4.3.3

           Table 3. Cases of Received syslog Messages

4.3.1 Valid PRI and TIMESTAMP

   If the relay does find a valid PRI and a valid TIMESTAMP, then it
   will check its internal configuration.  Relays MUST be configured to
   forward syslog packets on the basis of their Priority value.  If the
   relay finds that it is configured to forward the received packet,
   then it MUST do so without making any changes to the packet.  To
   emphasize the point one more time, it is for this reason that it is
   RECOMMENDED that the syslog message originally transmitted adhere to
   the format described in Section 4.1.

   It should be noted here that the message receiver does not need to
   validate the time in the TIMESTAMP field.  The assumption may be made
   that a device whose date has not been correctly set will still have
   the ability to send valid syslog messages.  Additionally, the relay
   does not need to validate that the value in the HOSTNAME field
   matches the hostname or IP address of the device sending the message.
   A reason for this behavior may be found in Section 4.1.2.

4.3.2 Valid PRI but no TIMESTAMP or invalid TIMESTAMP

   If a relay does not find a valid TIMESTAMP in a received syslog
   packet, then it MUST add a TIMESTAMP and a space character
   immediately after the closing angle bracket of the PRI part.  It
   SHOULD additionally add a HOSTNAME and a space character after the
   TIMESTAMP.  These fields are described here and detailed in Section
   4.1.2.  The remainder of the received packet MUST be treated as the
   CONTENT field of the MSG and appended.  Since the relay would have no
   way to determine the originating process from the device that
   originated the message, the TAG value cannot be determined and will
   not be included.

   The TIMESTAMP will be the current local time of the relay.

   The HOSTNAME will be the name of the device, as it is known by the
   relay.  If the name cannot be determined, the IP address of the
   device will be used.

   If the relay adds a TIMESTAMP, or a TIMESTAMP and HOSTNAME, after the
   PRI part, then it MUST check that the total length of the packet is
   still 1024 bytes or less.  If the packet has been expanded beyond
   1024 bytes, then the relay MUST truncate the packet to be 1024 bytes.
   This may cause the loss of vital information from the end of the
   original packet.  It is for this reason that it is RECOMMENDED that
   the PRI and HEADER parts of originally generated syslog packets
   contain the values and fields documented in Section 4.1.

4.3.3 No PRI or Unidentifiable PRI

   If the relay receives a syslog message without a PRI, or with an
   unidentifiable PRI, then it MUST insert a PRI with a Priority value
   of 13 as well as a TIMESTAMP as described in Section 4.3.2.  The
   relay SHOULD also insert a HOSTNAME as described in Section 4.3.2.
   The entire contents of the received packet will be treated as the
   CONTENT of the relayed MSG and appended.

   An example of an unidentifiable PRI would be "<00>", without the
   double quotes.  It may be that these are the first 4 characters of
   the message.  To continue this example, if a relay does receive a
   syslog message with the first four characters of "<00>", then it will
   consult its configuration.  If it is configured to forward syslog
   messages with a Priority value of 13 to another relay or collector,
   then it MUST modify the packet as described above.  The specifics of
   doing this, including the RECOMMENDED insertion of the HOSTNAME, are
   given below.

   Originally received message
     <00>...
   Relayed message
     <13>TIMESTAMP HOSTNAME <00>...

   If the relay adds a TIMESTAMP, or a TIMESTAMP and HOSTNAME, after the
   PRI part, then it MUST check that the total length of the packet is
   still 1024 bytes or less.  If the packet has been expanded beyond
   1024 bytes, then the relay MUST truncate the packet to be 1024 bytes.
   This may cause the loss of vital information from the end of the
   original packet.  It is for this reason that it is RECOMMENDED that
   the PRI and HEADER parts of originally generated syslog packets
   contain the values and fields documented in Section 4.1.

5. Conventions

   Although Section 4 of this document specifies all requirements for
   the syslog protocol format and contents, certain conventions have
   come about over time for the inclusion of additional information
   within the syslog message.  It must be plainly stated that these
   items are not mandated but may be considered by implementers for
   completeness and to give the recipient some additional clues of their
   origin and nature.

5.1 Dates and Times

   It has been found that some network administrators like to archive
   their syslog messages over long periods of time.  It has been seen
   that some original syslog messages contain a more explicit time stamp
   in which a 2 character or 4 character year field immediately follows
   the space terminating the TIMESTAMP.  This is not consistent with the
   original intent of the order and format of the fields.  If
   implementers wish to contain a more specific date and time stamp
   within the transmitted message, it should be within the CONTENT
   field.  Implementers may wish to utilize the ISO 8601 [7] date and
   time formats if they want to include more explicit date and time
   information.

   Additional methods to address this desire for long-term archiving
   have been proposed and some have been successfully implemented.  One
   such method is that the network administrators may choose to modify
   the messages stored on their collectors.  They may run a simple
   script to add the year, and any other information, to each stored
   record.  Alternatively, the script may replace the stored time with a
   format more appropriate for the needs of the network administrators.
   Another alternative has been to insert a record into the file that
   contains the current year.  By association then, all other records
   near that informative record should have been received in that same
   year.  Neither of these however, addresses the issue of associating a
   correct timezone with each record.

5.2 Domain Name and Address

   To readily identify the device that originated the message, it may be
   a good practice to include its fully qualified domain name (FQDN) and
   its IP address within the CONTENT field.  Traditionally, however,
   only the hostname has been included in the HOSTNAME field.

5.3 Originating Process Information

   It has also been considered to be a good practice to include some
   information about the process on the device that generated the
   message - if that concept exists.  This is usually the process name
   and process id (often known as the "pid") for robust operating
   systems.  The process name is commonly displayed in the TAG field.
   Quite often, additional information is included at the beginning of
   the CONTENT field.  The format of "TAG[pid]:" - without the quote
   marks - is common.  The left square bracket is used to terminate the
   TAG field in this case and is then the first character in the CONTENT
   field.  If the process id is immaterial, it may be left off.

In that case, a colon and a space character usually follow the TAG. This would be displayed as "TAG: " without the quotes. In that case, the colon is the first character in the CONTENT field.

5.4 Examples

As examples, these are valid messages as they may be observed on the wire between two devices. In the following examples, each message has been indented, with line breaks inserted in this document for readability.

Example 1

<34>Oct 11 22:14:15 mymachine su: 'su root' failed for lonvick on /dev/pts/8

This example shows an authentication error in an attempt to acquire additional privileges. It also shows the command attempted and the user attempting it. This was recorded as an original message from the device called mymachine. A relay receiving this would not make any changes before sending it along as it contains a properly formatted PRI part and TIMESTAMP field in the HEADER part. The TAG value in this example is the process "su". The colon has terminated the TAG field and is the first character of the CONTENT field. In this case, the process id (pid) would be considered transient and anyone looking at this syslog message would gain no useful information from knowing the pid. It has not been included so the first two characters of the CONTENT field are the colon and a space character.

Example 2

Use the BFG!

While this is a valid message, it has extraordinarily little useful information. This message does not have any discernable PRI part. It does not contain a timestamp or any indication of the source of the message. If this message is stored on paper or disk, subsequent review of the message will not yield anything of value.

This example is obviously an original message from a device. A relay MUST make changes to the message as described in Section 4.3 before forwarding it. The resulting relayed message is shown below.

<13>Feb  5 17:32:18 10.0.0.99 Use the BFG!

In this relayed message, the entire message has been treated as the
CONTENT portion of the MSG part.  First, a valid PRI part has been
added using the default priority value of 13.  Next, a TIMESTAMP has
been added along with a HOSTNAME in the HEADER part.  Subsequent
relays will not make any further changes to this message.  It should
be noted in this example that the day of the month is less than 10.
Since single digits in the date (5 in this case) are preceded by a
space in the TIMESTAMP format, there are two spaces following the
month in the TIMESTAMP before the day of the month.  Also, the relay
appears to have no knowledge of the host name of the device sending
the message so it has inserted the IPv4 address of the device into
the HOSTNAME field.

     Example 3

      <165>Aug 24 05:34:00 CST 1987 mymachine myproc[10]: %% It's
      time to make the do-nuts.  %%  Ingredients: Mix=OK, Jelly=OK #
      Devices: Mixer=OK, Jelly_Injector=OK, Frier=OK # Transport:
      Conveyer1=OK, Conveyer2=OK # %%

This message does have a valid PRI part with a Priority value
indicating that it came from a locally defined facility (local4) with
a severity of Notice.  The HEADER part has a proper TIMESTAMP field
in the message.  A relay will not modify this message before sending
it.  However, the HOSTNAME and TAG fields are not consistent with the
definitions in Section 4.  The HOSTNAME field would be construed to
be "CST" and the beginning of the MSG part would be "1987".

It should be noted that the information contained in the CONTENT of
this example is not telemetry data, nor is it supervisory control or
data acquisition information.  Due to the security concerns listed in
Section 6 of this document, information of that nature should
probably not be conveyed across this protocol.

     Example 4

      <0>1990 Oct 22 10:52:01 TZ-6 scapegoat.dmz.example.org 10.1.2.3
      sched[0]: That's All Folks!

This example has a lot of extraneous information throughout.  A human
or sufficiently adaptable automated parser would be able to determine
the date and time information as well as a fully qualified domain
name (FQDN) [4] and IP address.  The information about the nature of
the event is, however, limited.  Due to the indicated severity of the
event, the process may not have been able to gather or send anything
more informative.  It may have been fortunate to have generated and
sent this message at all.

This example is obviously an original message from a device.  Since
the first field in the HEADER part is not a TIMESTAMP in the format
defined in Section 4.1.2, it MUST be modified by a relay.  A relay
will add a TIMESTAMP and SHOULD add a HOSTNAME as follows and will
treat the entire received packet after the PRI part from the original
packet as the CONTENT field of the new packet.  The value used in the
HOSTNAME field is only the hostname without the domain name as it is
known by the relay.  A TAG value will not be added to the relayed
packet.  While the inclusion of the domain name and IPv4 address in
the original message is a noble endeavor, it is not consistent with
the use of the field as described in Section 4.1.2.

        <0>Oct 22 10:52:12 scapegoat 1990 Oct 22 10:52:01 TZ-6
        scapegoat.dmz.example.org 10.1.2.3 sched[0]: That's All Folks!

6. Security Considerations

   An odor may be considered to be a message that does not require any
   acknowledgement.  People tend to avoid bad odors but are drawn to
   odors that they associate with good food.  The acknowledgement of the
   receipt of the odor or scent is not required and indeed it may be the
   height of discretion to totally ignore some odors.  On the other
   hand, it is usually considered good civility to acknowledge the
   prowess of the cook merely from the ambiance wafting from the
   kitchen.  Similarly, various species have been found to utilize odors
   to attract mates.  One species of moth uses this scent to find each
   other.  However, it has been found that bolas spiders can mimic the
   odor of the female moths of this species.  This scent will then
   attract male moths, which will follow it with the expectation of
   finding a mate.  Instead, when they arrive at the source of the
   scent, they will be eaten [8].  This is a case of a false message
   being sent out with inimical intent.

   In its local use, the syslog process places event notification
   messages into files on that system.  This relies upon the integrity
   of the system for the protection of the messages.  The subsequent
   configuration of the syslog process to use the syslog protocol to
   transport the messages to a remote collector was an extension of the
   delivery of event notification messages and it exhibits the same
   trust of the network.  There are several security consequences of the
   fundamental simplicity of syslog and there are some concerns about
   the applicability of this protocol in situations that require robust
   delivery.  Along the lines of the analogy, computer event messages
   may be sent accidentally, erroneously and even maliciously. At the
   time of this writing, however, there have not been any reports of any
   networked device consuming any other device.

6.1 Packet Parameters

   As was described above, the message length MUST NOT exceed 1024
   bytes.  Attacks have seen where syslog messages are sent to a
   receiver that have message lengths greater than 1024 bytes.  In some
   older versions of syslog, the receipt of syslog packets that had a
   message greater than 1024 bytes caused problems.  syslog message
   receivers must not malfunction upon the receipt of packets where the
   message length is greater than 1024 bytes.  Various behaviors have
   been seen on receivers that do receive messages greater than 1024
   bytes.  Some have been seen to log the entire contents of the
   message, while others have been seen to log only portions of the
   message.  Still others have been known to discard the message
   altogether.  Devices MUST NOT retransmit messages whose received
   length exceeds 1024 bytes.

   Similarly, the receiver must rigidly enforce the correctness of the
   message body.  syslog collectors must not malfunction if received
   messages do not have the less-than and greater-than characters around
   a valid Priority value.  They MUST treat these messages as the
   unformatted CONTENT as was described in Section 4.3.3 if they relay
   it.

   Also, received messages must contain printable text in the message as
   was described throughout Section 4.  Devices must not malfunction if
   they receive a message containing characters other than the
   characters described above.

6.2 Message Authenticity

   The syslog delivery mechanism does not strongly associate the message
   with the message sender.  The receiver of that packet will not be
   able to ascertain that the message was indeed sent from the reported
   sender, or if the packet was sent from another device.  It should be
   noted here that the message receiver does not need to verify that the
   HOSTNAME in the HEADER part match the name of the IP address
   contained in the Source Address field of the IP packet.

6.2.1 Authentication Problems

   One possible consequence of this behavior is that a misconfigured
   machine may send syslog messages to a collector representing itself
   as another machine.  The administrative staff may become confused
   that the status of the supposed sender of the messages may not be
   accurately reflected in the received messages.  The administrators
   may not be able to readily discern that there are two or more
   machines representing themselves as the same machine.

It should also be noted that some cases of filling the HOSTNAME field
in the HEADER part might only have local significance and that may
only be ephemeral.  If the device had obtained an IP address from a
DHCP pool, then any association between an identifier and an actual
source would not always hold true.  The inclusion of a fully
qualified domain name in the CONTENT may give the administrators the
best chance of identifying the source of each message if it can
always be associated with an IP address or if it can always be
associated with a unique machine.

6.2.2 Message Forgery

Malicious exploits of this behavior have also been noted.  An
attacker may transmit syslog messages (either from the machine from
which the messages are purportedly sent or from any other machine) to
a collector.  In one case, an attacker may hide the true nature of an
attack amidst many other messages.  As an example, an attacker may
start generating forged messages indicating a problem on some
machine.  This may get the attention of the system administrators who
will spend their time investigating the alleged problem.  During this
time, the attacker may be able to compromise a different machine, or
a different process on the same machine.  Additionally, an attacker
may generate false syslog messages to give untrue indications of
status or of events.  As an example, an attacker may stop a critical
process on a machine, which may generate a notification of exit.  The
attacker may subsequently generate a forged notification that the
process had been restarted.  The system administrators may accept
that misinformation and not verify that the process had indeed been
restarted.

6.3 Sequenced Delivery

As a general rule, the forensics of a network anomaly rely upon
reconstructing the sequence of events.  In a perfect world, the
messages would be received on the syslog collector in the order of
their generation from the other devices and anyone looking at these
records would have an accurate picture of the sequence of events.
Unfortunately, the syslog process and protocol do not ensure ordered
delivery.  This section details some of the problems that may be
encountered from this.

6.3.1 Single Source to a Destination

The syslog records are usually presented (placed in a file, displayed
on the console, etc.) in the order in which they are received.  This
is not always in accordance with the sequence in which they were
generated.  As they are transported across an IP network, some out of
order receipt should be expected.  This may lead to some confusion as

messages may be received that would indicate that a process has
stopped before it was started.  This may be somewhat rectified if the
originating process had timestamped or numbered each of the messages
before transmission.  In this, the sending device should utilize an
authoritative time source.  It should be remembered, however, that
not all devices are capable of receiving time updates, and not all
devices can timestamp their messages.

6.3.2 Multiple Sources to a Destination

   In syslog, there is no concept of unified event numbering.  Single
   devices are free to include a sequence number within the CONTENT but
   that can hardly be coordinated between multiple devices.  In such
   cases, multiple devices may report that each one is sending message
   number one.  Again, this may be rectified somewhat if the sending
   devices utilize a timestamp from an authoritative source in their
   messages.  As has been noted, however, even messages from a single
   device to a single collector may be received out of order.  This
   situation is compounded when there are several devices configured to
   send their syslog messages to a single collector.  Messages from one
   device may be delayed so the collector receives messages from another
   device first even though the messages from the first device were
   generated before the messages from the second.  If there is no
   timestamp or coordinated sequence number, then the messages may be
   presented in the order in which they were received which may give an
   inaccurate view of the sequence of actual events.

6.3.3 Multiple Sources to Multiple Destinations

   The plethora of configuration options available to the network
   administrators may further skew the perception of the order of
   events.  It is possible to configure a group of devices to send the
   status messages -or other informative messages- to one collector,
   while sending messages of relatively higher importance to another
   collector.  Additionally, the messages may be sent to different files
   on the same collector.  If the messages do not contain timestamps
   from the source, it may be difficult to order the messages if they
   are kept in different places.  An administrator may not be able to
   determine if a record in one file occurred before or after a record
   in a different file.  This may be somewhat alleviated by placing
   marking messages with a timestamp into all destination files.  If
   these have coordinated timestamps, then there will be some indication
   of the time of receipt of the individual messages.

6.3.4 Replaying

   Without any sequence indication or timestamp, messages may be
   recorded and replayed at a later time.  An attacker may record a set
   of messages that indicate normal activity of a machine.  At a later
   time, that attacker may remove that machine from the network and
   replay the syslog messages to the collector.  Even with a TIMESTAMP
   field in the HEADER part, an attacker may record the packets and
   could simply modify them to reflect the current time before
   retransmitting them.  The administrators may find nothing unusual in
   the received messages and their receipt would falsely indicate normal
   activity of the machine.

6.4 Reliable Delivery

   As there is no mechanism within either the syslog process or the
   protocol to ensure delivery, and since the underlying transport is
   UDP, some messages may be lost.  They may either be dropped through
   network congestion, or they may be maliciously intercepted and
   discarded.  The consequences of the drop of one or more syslog
   messages cannot be determined.  If the messages are simple status
   updates, then their non-receipt may either not be noticed, or it may
   cause an annoyance for the system operators.  On the other hand, if
   the messages are more critical, then the administrators may not
   become aware of a developing and potentially serious problem.
   Messages may also be intercepted and discarded by an attacker as a
   way to hide unauthorized activities.

6.5 Message Integrity

   Besides being discarded, syslog messages may be damaged in transit,
   or an attacker may maliciously modify them.  In the case of a packet
   containing a syslog message being damaged, there are various
   mechanisms built into the link layer as well as into the IP [9] and
   UDP protocols which may detect the damage.  An intermediary router
   may discard a damaged IP packet [10].  Damage to a UDP packet may be
   detected by the receiving UDP module, which may silently discard it.
   In any case, the original contents of the message will not be
   delivered to the collector.  Additionally, if an attacker is
   positioned between the sender and collector of syslog messages, they
   may be able to intercept and modify those messages while in-transit
   to hide unauthorized activities.

6.6 Message Observation

   While there are no strict guidelines pertaining to the event message
   format, most syslog messages are generated in human readable form
   with the assumption that capable administrators should be able to

read them and understand their meaning.  Neither the syslog protocol
nor the syslog application have mechanisms to provide confidentiality
of the messages in transit.  In most cases passing clear-text
messages is a benefit to the operations staff if they are sniffing
the packets off of the wire.  The operations staff may be able to
read the messages and associate them with other events seen from
other packets crossing the wire to track down and correct problems.
Unfortunately, an attacker may also be able to observe the human-
readable contents of syslog messages.  The attacker may then use the
knowledge gained from those messages to compromise a machine or do
other damage.

6.7 Message Prioritization and Differentiation

   While the processes that create the messages may signify the
   importance of the events through the use of the message Priority
   value, there is no distinct association between this value and the
   importance of delivery of the packet.  As an example of this,
   consider an application that generates two event messages.  The first
   is a normal status message but the second could be an important
   message denoting a problem with the process.  This second message
   would have an appropriately higher Severity value associated with the
   importance of that event.  If the operators had configured that both
   of these messages be transported to a syslog collector then they
   would, in turn, be given to UDP for transmission.  Under normal
   conditions, no distinction would be made between them and they would
   be transmitted in their order.

   Again, under normal circumstances, the receiver would accept syslog
   messages as they are received.  If many devices are transmitting
   normal status messages, but one is transmitting an important event
   message, there is no inherent mechanism within the syslog protocol to
   prioritize the important message over the other messages.

   On a case-by-case basis, device operators may find some way to
   associate the different levels with the quality of service
   identifiers.  As an example, the operators may elect to define some
   linkage between syslog messages that have a specific Priority value
   with a specific value to be used in the IPv4 Precedence field [9],
   the IPv6 Traffic Class octet [11], or the Differentiated Services
   field [12].  In the above example, the operators may have the ability
   to associate the status message with normal delivery while
   associating the message indicating a problem with a high reliability,
   low latency queue as it goes through the network.  This would have
   the affect of prioritizing the essential messages before the normal
   status messages.  Even with this hop-by-hop prioritization, this
   queuing mechanism could still lead to head of line blocking on the
   transmitting device as well as buffer starvation on the receiving

device if there are many near-simultaneous messages being sent or
received.  This behavior is not unique to syslog but is endemic to
all operations that transmit messages serially.

There are security concerns for this behavior.  Head of line blocking
of the transmission of important event messages may relegate the
conveyance of important messages behind less important messages.  If
the queue is cleared appropriately, this may only add seconds to the
transmission of the important message.  On the other hand, if the
queue is not cleared, then important messages may not be transmitted.
Also at the receiving side, if the syslog receiver is suffering from
buffer starvation due to large numbers of messages being received
near-simultaneously, important messages may be dropped
indiscriminately along with other messages.  While these are problems
with the devices and their capacities, the protocol security concern
is that there is no prioritization of the relatively more important
messages over the less important messages.

6.8 Misconfiguration

Since there is no control information distributed about any messages
or configurations, it is wholly the responsibility of the network
administrator to ensure that the messages are actually going to the
intended recipient.  Cases have been noted where devices were
inadvertently configured to send syslog messages to the wrong
receiver.  In many cases, the inadvertent receiver may not be
configured to receive syslog messages and it will probably discard
them.  In certain other cases, the receipt of syslog messages has
been known to cause problems for the unintended recipient [13].  If
messages are not going to the intended recipient, then they cannot be
reviewed or processed.

6.9 Forwarding Loop

As it is shown in Figure 1, machines may be configured to relay
syslog messages to subsequent relays before reaching a collector. In
one particular case, an administrator found that he had mistakenly
configured two relays to forward messages with certain Priority
values to each other.  When either of these machines either received
or generated that type of message, it would forward it to the other
relay.  That relay would, in turn, forward it back.  This cycle did
cause degradation to the intervening network as well as to the
processing availability on the two devices.  Network administrators
must take care to not cause such a death spiral.

6.10 Load Considerations

   Network administrators must take the time to estimate the appropriate
   size of the syslog receivers.  An attacker may perform a Denial of
   Service attack by filling the disk of the collector with false
   messages.  Placing the records in a circular file may alleviate this
   but that has the consequence of not ensuring that an administrator
   will be able to review the records in the future. Along this line, a
   receiver or collector must have a network interface capable of
   receiving all messages sent to it.

   Administrators and network planners must also critically review the
   network paths between the devices, the relays, and the collectors.
   Generated syslog messages should not overwhelm any of the network
   links.

7. IANA Considerations

   The syslog protocol has been assigned UDP port 514.  This port
   assignment will be maintained by IANA exclusively for this protocol.

   The syslog protocol provides for the definition of named attributes
   to indicate the Severity of each message and the Facility that
   generated the message as described in Section 4.  The name space
   identifiers for these attributes are defined as numbers.  The
   protocol does not define the specific assignment of the name space
   for these numbers; the application developer or system vendor is
   allowed to define the attribute, its semantics, and the associated
   numbers.  This name space will not be controlled to prevent
   collisions as systems are expected to use the same attributes,
   semantics and associated numbers to describe events that are deemed
   similar even between heterogeneous devices.

8. Conclusion and Other Efforts

   The syslog protocol may be effectively used to transport event
   notification messages across a network.  In all cases, it is
   important that the syslog message receiver embody the principle of
   "be liberal in what you accept".  It is highly recommended that the
   network operators who choose to use this understand the
   characteristics of the protocol and its security implications.

   There have been attempts in the past to standardize the format of the
   syslog message.  The most notable attempt culminated in a BOF at the
   Fortieth Internet Engineering Task Force meeting in 1997.  This was
   the Universal Logging Protocol (ulp) BOF and the minutes of their
   meeting are on-line at the IETF Proceedings web site [14].

Many good thoughts came from that effort and interested implementers
may want to find some of the notes or papers produced from that
effort.

At the time of this writing, efforts are underway to allow the usage
of international character sets in applications that have been
traditionally thought of as being text-only.  The HOSTNAME and
TIMESTAMP fields described above are representative of this.  Also,
the entire CONTENT field has traditionally been printing characters
and spaces in the code set known as US-ASCII.  It is hoped that the
proponents of these internationalization efforts will find a suitable
way to allow the use of international character sets within syslog
messages without being disruptive.  It should also be hoped that
implementers will allow for the future acceptance of additional code
sets and that they may make appropriate plans.  Again, it must be
cautioned that the simplicity of the existing system has been a
tremendous value to its acceptance.  Anything that lessens that
simplicity may diminish that value.

Acknowledgements

The following people provided content feedback during the writing of
this document:

        Jon Knight <J.P.Knight@lboro.ac.uk>
        Magosanyi Arpad <mag@bunuel.tii.matav.hu>
        Balazs Scheidler <bazsi@balabit.hu>
        Jon Callas <jon@counterpane.com>
        Eliot Lear <lear@cisco.com>
        Petter Reinholdtsen <pere@hungry.com>
        Darren Reed <darrenr@reed.wattle.id.au>
        Alfonso De Gregorio <dira@speedcom.it>
        Eric Allman <eric@sendmail.com>
        Andrew Ross <andrew@kiwi-enterprises.com>
        George Maslyar <george.maslyar@primark.com>
        Albert Mietus <albert@ons-huis.net>
        Russ Allbery <rra@stanford.edu>
        Titus D. Winters <titus@cs.hmc.edu>
        Edwin P. Boon <Edwin.Boon@consul.com>
        Jeroen M. Mostert <Jeroen.Mostert@consul.com>

Eric Allman is the original inventor and author of the syslog daemon
and protocol.  The author of this memo and the community at large
would like to express their appreciation for this work and for the
usefulness that it has provided over the years.

A large amount of additional information about this de-facto standard operating system feature may usually be found in the syslog.conf file as well as in the man pages for syslog.conf, syslog, syslogd, and logger, of many Unix and Unix-like devices.

References

1   Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

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

3   USA Standard Code for Information Interchange, USASI X3.4-1968

4   Mockapetris, P., "Domain Names - Concepts and Facilities", STD 13, RFC 1034, November 1987.

5   Mockapetris, P., "Domain names - Implementation and Specification", STD 13, RFC 1035, November 1987.

6   Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 2373, July 1998.

7   Data elements and interchange formats - Information exchange - Representation of dates and times, International Organization for Standardization, Reference number ISO 8601 : 1988 (E), 1988

8   Stowe, M., et al, "Chemical Mimicry: Bolas Spiders Emit Components of Moth Prey Species Sex Pheromones", Science, 1987

9   Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

10  Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

11  Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

12  Nichols, K., Blake, S., Baker, F. and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

13  Cisco Systems Product Security Incident Response Team (PSIRT), "Field Notice: Cisco IOS(r) Syslog Crash", January 11, 1999 http://www.cisco.com/warp/public/707/advisory.html

   14 Walker, D., IETF Secretariat, "Proceedings of the Fortieth
      Internet Engineering Task Force, Washington, DC, USA, December 8-
      12, 1997
      http://www.ietf.org/proceedings/97dec/index.html

Author's Address

   Chris Lonvick
   Cisco Systems
   12515 Research Blvd.
   Austin, TX, USA

   Phone:   +1.512.378.1182
   EMail:   clonvick@cisco.com

Full Copyright Statement

Acknowledgement