OIF Control Plane Logging and Auditing with Syslog version 1.1

IA # OIF-SLG-01.3

October 8, 2012

Implementation Agreement created and approved
by the Optical Internetworking Forum
www.oiforum.com

TITLE: **OIF Control Plane Logging and Auditing with Syslog version 1.1**

TECHNICAL EDITOR
Richard Graveman, RFG Security
for Department of Defense
15 Park Avenue
Morristown, NJ 07960 USA
+1 973 984 8780
rfg@acm.org

WORKING GROUP CHAIRS
Rémi Theillaud, Marben Products

Evelyne Roch, Ciena Corporation

Doug Zuckerman, Telcordia Technologies

ABSTRACT: This IA specifies how to use the IETF's standard version of Syslog to log OIF control plane traffic as the basis for an audit capability. It also covers securing the log files and controlling their generation and collection. It is an optional component of the UNI and E-NNI intended to be used in conjunction with the *Security Extension for UNI and E-NNI.*

Table of Contents

List of Figures

List of Tables

None.

List of Contributors to Prior Versions of this Document

Renée Esposito, Booz Allen Hamilton
esposito_renee@bah.com

Byron Forrest, Booz Allen Hamilton
Forrest_Byron@bah.com

Richard Graveman, Department of Defense
rfg@acm.org

Bruce Potter, Booz Allen Hamilton
potter_bruce@bah.com

Jonathan Sadler, Tellabs
Jonathan.Sadler@tellabs.com

Tom Tarman, Sandia National Labs
tdtarma@sandia.gov

OIF Control Plane Logging and Auditing with Syslog version 1.1

1 Introduction

This Implementation Agreement (IA) defines the protocols, record types, data structures, and fields for an audit log generated by a Network Element (NE). It also addresses controlling and securing the generation, transport, and storage of log data to enable an auditing capability for the OIF's UNI and E-NNI.

1.1 Problem Statement

Interoperable methods are needed for (1) gathering network statistics for network planning, configuration, tuning, etc.; (2) generating, transmitting, and collecting log files for diagnosing networking problems or the proper functioning of a NE; (3) using a log to verify usage and to audit billing information; and (4) creating and processing log records to help identify and respond to protocol errors, security violations, unauthorized modifications, and other mis-configurations or malfunctions.

The OIF has defined a method for secure transmission of signaling and routing messages from one NE (or its clients, components, or proxies) to the next, for example over a UNI-C to UNI-N interface. This protection is effective against attackers who would impersonate a NE and forge, modify, or eavesdrop on these messages. It does not, however, given the semantics of commonly used signaling and link state routing protocols, allow UNI interfaces or non-adjacent E-NNI entities to obtain assurance that the messages delivered end to end accurately represent the originating party's intentions. Securing such protocols from one end to the other or across non-adjacent entities appears to be a more difficult problem to solve efficiently with commonly used signaling protocol semantics and protocol security technology. Therefore, the ability to log such signaling and routing messages as they traverse multiple UNI or E-NNI interfaces adds a useful and effective way to increase assurance that the signaling and routing entities are operating securely and correctly.

1.2 Scope

This IA specifies a standard method for logging the operation of NEs, collecting log data, and facilitating a subsequent audit of the NEs' operations. It defines a profile of a logging protocol that runs on TCP/IP networks, so it only applies to NEs running IPv4 or IPv6. In particular, this IA is RECOMMENDED for use on NEs that implement the OIF's *Security Extension for UNI and E-NNI version 2.0* [OIF12a] or earlier versions of that IA, but it MAY be used in other cases as well.

This IA covers audit data formats, encoding, records and fields, transport protocols, control of which events are logged, and security for the logging system.

1.3 Goals and Objectives

The main goals of this IA are (1) to specify a flexible and interoperable logging mechanism that enables auditing and diagnosis of events and conditions; (2) to provide a method for increasing end-to-end assurance that the OIF's signaling and routing protocols are functioning properly; and (3) to define appropriate management and security services and mechanisms for the logging protocol.

This IA defines multiple types of log messages. Redundancy exists among the log messages defined here and may also exist between log records defined in this IA and information available from other sources. Because logging can generate large amounts of data, an important aspect of this IA is to define a mechanism for selectively controlling the generation and disposition of log data. The intention is to define flexible mechanisms that can be configured to log only desired events, messages, or status information and can be enabled or disabled as required.

This IA complements and enhances the usefulness of the methods defined in the OIF's UNI signaling, E-NNI signaling and routing, and *Security Extension for UNI and E-NNI version 2.0* IAs by defining a logging mechanism compatible with the protocols used in these IAs. This logging mechanism supports a system that can audit the operation of these protocols.

1.4 Relationship to Other Standards Bodies

This IA uses the ideas, requirements, reference models, protocols, and terminology contained in:

- RFCs and Internet Drafts produced by the IETF's Security Area and Operations and Management Area.
- The ATM Forum's specification on auditing and logging [Confil].

It also provides a secure method for logging alarms specified in the ITU-T's X.733 and X.736 recommendations.

1.5 Viewpoint

A NE may be viewed as a single component but actually be composed of a "distributed system" divided, for example, into control and transport components or their proxies. This IA applies to all such components.

The components of a NE have one or more data, control (i.e., signaling or routing) and management (i.e., OAM&P) interfaces as well as interfaces to other systems within the same NE. Syslog, the logging and auditing mechanism profiled in this IA, can be configured to generate log records describing conditions or actions that occur on any or all of these interfaces.

Typically, syslog data generated at a NE are transmitted for further processing over one of its interfaces, but it is possible to transmit log data selectively over multiple interfaces, or, alternatively, to treat logging as an entirely local matter and not transmit log data at all. No relationship necessarily exists between the interface(s) on which an action to be logged occurs and the interface(s) over which the logging takes place.

The operational roles for the syslog service are designated Originators, Relays, and Collectors. An Originator, previously called a device, is a source that generates log message content. A Collector, also known as a syslog server, receives log messages for further analysis. Typically, it captures the logs into a file for review by an administrator. Collectors are capable of implementing advanced filtering rules, which determine how messages are (1) stored locally, (2) retained in a centralized management system, or (3) displayed for an operator to review. There is no notion of a one-to-one relationship between Originators and Collectors: messages from an Originator may go to many Collectors and vice versa. A Relay sits between an Originator and a Collector. Relays accept syslog messages from Originators and other Relays and forward those messages to Collectors or other Relays. Thus, it is possible to have multiple Relays between an Originator and a Collector. A Relay may filter messages based on Facility and Severity and selectively forward messages accordingly. Originators, Relays, and Collectors may run on separate systems, or their functionality may be combined on a single system, which acts in more than one of these roles. Originators, Relays, and Collectors are independent of underlying transport protocols. At a lower layer, Transport Senders pass syslog messages to a specific transport protocol, and Transport Receivers accept them from a specific transport protocol.

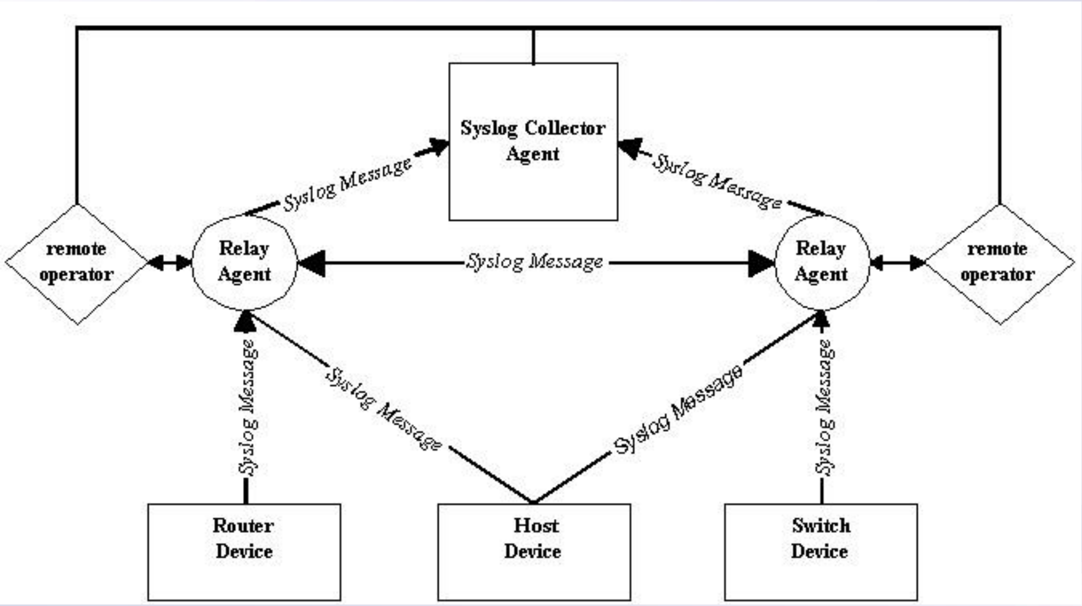

Figure 1: Syslog System Block Diagram.

Figure 1, above, illustrates how the syslog process is designed to operate: Originators send messages to Relays (or directly to Collectors); Relays accept and forward messages to other Relays or to Collectors for access by a remote operator. In some configurations, systems can play a dual role as Collector and Relay. Remote operators are those systems that actively interact with Relays or Collectors to compile the data received for further processing.

1.6 Acknowledgements

This IA uses prior work from the OIF and other standards development organizations, in particular “The Syslog Protocol” [Ger09], “Transmission of Syslog Messages over UDP”

[Okm09], "Transport Layer Security (TLS) Transport Mapping for Syslog" [MMS09], and "Signed Syslog Messages" [KCC10] defined by the IETF. The main thrust of this IA, to allow users to diagnose and to verify end-to-end behavior of signaling and link state routing protocols, evolved from the work cited in the ATM Forum [Confil]. Security requirements for logging and auditing were developed in the OIF, based on the ATM Forum's work, and in the IETF, based on the opsec Working Group's requirements [OPSEC]. The IETF developed IPsec and TLS, and the OIF later specified profiles of IPsec [OIF12a] and TLS [OIF12b] applied in this IA.

The following companies provided helpful comments on the original version of this document: Alcatel (now Alcatel-Lucent), AT&T, Cisco, Lucent (now Alcatel-Lucent), Tellabs, and Verizon.

George Newsome (Ciena), Evelyne Roch (Nortel), Jonathan Sadler (Tellabs), and Rémi Theillaud (Marben Products) provided helpful comments during balloting.

2 Overview

2.1 Document Organization and Outline

Section 4 covers the problem statement, scope, expected outcome, and relationship to other work. Section 5 contains an outline and guidelines for use. Section 6 defines keywords, terminology, and acronyms. Section 7 combines, paraphrases, and reiterates logging requirements documented by the OIF and IETF. Section 8 introduces the syslog protocol and other supporting protocols. Log records and data to be logged are defined in Section 9. Section 10 specifies exactly how the syslog protocol and transport protocols are used in this IA. Section 11 describes methods for controlling the generation of the log data. Section 12 defines methods for securing the disposition of log data. Normative and informative references comprise Section 13. Two informative appendices are included, the first of which describes the configuration file used to control syslog on BSD UNIX systems and the second of which lists security threats to log records.

2.2 How to Use this Implementation Agreement

Syslog had been widely implemented and was later documented by the IETF [Lon01], so large amounts of experience and reference code exist to help implementers of this logging mechanism. The protocol is efficient with respect to message size, transport protocol, code size, buffering, and computational requirements. The IETF has defined revisions of and extensions to syslog used in this IA: a standard HEADER format, structured fields in the message [Ger09], and signatures to guarantee message authenticity [KCC10]. The security specified herein provides an optimized method for end-to-end authentication and integrity guarantees and optional, additional methods of adding confidentiality to the transmission of log records based on the existing security infrastructure defined in the OIF's Security Extension for UNI and E-NNI [OIF12a] or the TLS transport protocol. Procedures for on-line and off-line processing of signed log records are included in the referenced IETF documents, in particular [KCC10].

3 Terminology and Acronyms

3.1 Keywords

When written in ALL CAPITALS, 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 IETF RFC 2119 [Bra97]. The key word "NOT RECOMMENDED" can be rephrased to mean "SHOULD NOT."

3.2 Terminology

In this implementation agreement, the following definitions apply:

Collector: A system, possibly a Management System and also called a syslog server, that gathers syslog messages for further analysis.

HEADER: The first part of a syslog record containing a PRI, VERSION, TIMESTAMP, HOSTNAME, APP-NAME, PROCID, and MSGID. The PRI encodes values for a FACILITY and SEVERITY. Note that differences exist between the PRI, HEADER, and TAG defined in [Lon01] and the HEADER defined in [Ger09] and used in this IA.

Management System: A generic term for a terminal, network element, or system that provides services to manage a NE. It may also function as a Relay or Collector (see below).

MSG: The third and remaining part of a syslog record after the HEADER and STRUCTURED-DATA; it contains the unstructured contents of the message.

Network Administrator (NA): A person who is authorized to use a Management System. (Refer to [T1M1] for the many roles that may exist for a NA.) In this IA, a NA may be called an operator or administrator.

Network Element (NE): Any system (i.e., device) supporting one or more of the OIF's UNI or E-NNI interfaces or services. It may also support other interfaces or services.

Originator: An application that generates the content of syslog messages.

Relay: A system, possibly a Management System, that receives syslog messages from Originators or other Relays and forwards them to other Relays or Collectors.

STRUCTURED-DATA (SD): The middle part of a syslog record, after the HEADER and before the MSG; it contains structured data identifiers (SD-IDs) each having zero or more parameters (SD-PARAMs) written as name-value pairs.

Transport Receiver: An application that accepts syslog messages from a specific transport protocol.

Transport Sender: An application that passes syslog messages to a specific transport protocol.

3.3 Acronyms

The following acronyms or abbreviations are used in this implementation agreement:

AH	Authentication Header
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DTLS	Datagram Transport Layer Security
E-NNI	External NNI
ESP	Encapsulating Security Payload
FQDN	Fully Qualified Domain Name
IANA	Internet Assigned Numbers Authority
IKE	Internet Key Exchange
IP	Internet Protocol version 4 or Internet Protocol version 6
IPsec	IP Security
MAC	Message Authentication Code
MTU	Maximum Transmission Unit
NA	Network Administrator
NAT	Network Address Translation
NE	Network Element
NNI	Network Node Interface
NTP	Network Time Protocol
PMTU	Path Maximum Transmission Unit
RFC	Request for Comments
SA	Security Association
SAD	Security Association Database
SCN	Signaling Communications Network
SD	Structured Data
SD-ID	Structured Data Identifier
SD-PARAM	Structured Data Parameter
S/MIME	Secure Multipurpose Internet Mail Extension
SNMP	Simple Network Management Protocol
SPD	Security Policy Database
SSH	Secure Shell
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security

UDP	User Datagram Protocol
UNI	User-Network Interface
UTC	Coordinated Universal Time
VPN	Virtual Private Network
WG	Working Group

4 Logging and Auditing Requirements

One of the main purposes of logging and auditing is to determine, with some degree of certainty, what events have occurred in the case of an intrusion, malfunction, or unauthorized modification of a system. An important component of this is to supplement the security provided for signaling and routing in [OIF12a] with an additional method for diagnosis and verification of end-to-end operation. Therefore, the log files themselves need to be protected from tampering. In some cases, the log files also contain sensitive information about the network configuration, usage patterns, or customers. In such cases, operating system and network protections, such as access controls and firewalls, should be used to restrict access to log data on systems running Originators, Relays, or Collectors, but encryption is the only effective way to protect this information from eavesdropping during transmission.

Users need to provide sufficient resources for the transmission and storage of log files. Because all logging is optional and configurable, the total volume of log data is determined by local policy and use, which are outside the scope of this document. This volume will depend on the size of the network, traffic granularity, network dynamics, filtering criteria, and period of time over which the logs are collected and retained. Using off-line backup storage media (perhaps ”write-once” technology) is recommended.

The following requirements for logging and auditing have been documented by the IETF [OPSEC] and OIF and are combined and summarized here:

- **Audit Reporting and Logging Requirements**
  - *Reporting of events "out of band" shall be supported.* If circumstances cause an outage in the data network, then the logging mechanism needs an alternate physical path for reporting events.
  - *Local logging to remote Relays and Collectors shall be supported.* This provides flexibility and redundancy.
  - *Audit log events shall be generated upon startup and shutdown of audit functions on the Originator*. This allows one to determine the period of coverage.
  - *The records in an audit log file shall be self-identifying.* Given an arbitrary segment out of the middle of a log file, one should be able to find the record boundaries, parse the records, and verify the integrity of the records unambiguously.

- *The logging facility shall be capable of logging any event that affects system integrity.* This provides the ability to reconstruct the state of the system.
- *Log events may include:*
  - *Filter changes (e.g., changes to what is logged or collected)*
  - *Authentication failures (e.g., bad login attempts)*
  - *Authentication successes (e.g., user logins)*
  - *Authorization changes (e.g., user privilege level changes)*
  - *Configuration changes (e.g., command accounting)*
  - *Originator status changes (e.g., control plane or management plane interface up or down, etc.)*
- *The logging mechanism shall conform to open standards.* This promotes interoperability and widespread deployment.
- *Originators shall have the ability to log to a remote server.* This provides for a flexible configuration of the logging system and the ability to off-load processing from a NE to a Management System.
- *Administrators shall have the ability to configure the security of log messages.* It shall be possible to configure the security of the logging mechanism using independently controllable authenticity, integrity, confidentiality, and replay prevention mechanisms for log messages.
- *Logging mechanisms shall use standard protocols for end-to-end authentication and integrity and optionally confidentiality as defined by the IETF and OIF.* Authentication and confidentiality may be required for log messages. To minimize duplication of mechanisms, one or more of those specified by the IETF or OIF shall be used.
- *Log Servers shall have the ability to log events locally.* This allows continued operation in all cases.

- **Audit Authentication**
  - *Authenticated, dynamic remote configuration of filters shall be supported.* Filtering of log events at Originators allows a network administrator to reduce the load on Relays and Collectors.
- **Audit System Mechanisms**
  - *Reporting of communication session statistics shall be provided.* Log messages shall contain statistics of the parameters specific to each session including origination time, addressing, duration, and other configurable items.
  - *Time stamps shall be included in log messages.* Timestamps are required for event correlation and replay protection and shall be supplied by the Originators

at the highest supported resolution. Granularity and synchronization of timestamps shall be addressed in system specifications.

- *Originators shall have the ability to maintain accurate system time.* All displays of system time shall include a time zone. The default time zone shall be UTC (i.e., GMT). Originators should support a mechanism to allow the administrator to specify the local time zone.

- **Classification of Audit Events**
  - *Originators shall have the ability to classify logged events.* Originators shall assign a classification chosen from a well-known list to all logged messages.
  - *A field shall exist in the message to identify the Originator and, optionally, the process that originated the event.* The message source shall be recorded for later processing or analysis.
  - *Originators shall be able to relay events with different classifications to different log servers.* For example, security-related messages may go to one log server, whereas operational messages go to another.

- **Additional Audit Requirements**
  - *Numeric values for message source and severity level indicators shall be provided.* Numeric values (as opposed to string representations) allow efficient filtering and determination of actions that should be taken by Relays, Collectors, or network administrators.
  - *Logs shall contain untranslated addresses.* Log messages shall contain specific IP addresses for several reasons:
    - Network-based attacks often use spoofed source addresses. Source addresses should not be trusted unless verified by other means.
    - Addresses may be reassigned to different individuals, for example, in a desktop environment using DHCP or stateless autoconfiguration. In such cases individual accountability afforded by this requirement is weak.
    - Network topologies may change. Even in the absence of dynamic address assignment, network topologies and address block assignments do change. Logs of an attack one month ago may not give an accurate indication of which host, network, or organization owned the system(s) in question at the time.
  - *By default, log messages shall not contain DNS names.* Originators may provide a capability to incorporate translated DNS names in addition to IP addresses. This is important because IP-to-DNS mappings change over time and mappings done at one time may not be valid later. Also, the use of resources (memory, processor, time, and bandwidth) required to do the translation could result in no

data being sent or logged, and, in the extreme case, could lead to degraded performance or resource exhaustion.

5 Protocol Overview

5.1 Summary of Syslog

Syslog, with the UDP transport protocol [Okm09], is an unsecured, simplex (one-way), UDP-based protocol for logging events generated on networked systems including NEs and the applications that run on NEs.

Such systems are called syslog Originators. Syslog provides for relaying and collecting log data to enable on-line or off-line auditing. The format of syslog messages evolved over time and became a de facto standard that was documented by the IETF [Lon01]. The syslog protocol used in this IA is defined in [Ger09] and [Okm09]. Syslog is designed to collect data from Originators within three functional categories:

1. *Problem monitoring:* used as an on-line tool to help monitor problems as they occur
2. *Intrusion Detection:* used to detect inappropriate events that depict attempts to penetrate a system and gain unauthorized access
3. *Reconstructing events:* used to establish the sequence of events before or after a problem has occurred

Syslog allows messages to be categorized based on the urgency of their contents. Messages have different Severities for alerting an administrator, which are defined as listed below [Ger09, CG09] and encoded together with a Facility value in the PRI field at the beginning of each Syslog message:

Severity Code	Syslog Definition	Corresponding ITU-T Designation
0	Emergency: system is unusable	
1	Alert: action must be taken immediately	Critical
2	Critical: critical conditions	Major
3	Error: error conditions	Minor
4	Warning: warning conditions	Warning
5	Notice: normal but significant conditions	Indeterminate or Cleared
6	Informational: informational messages	
7	Debug: debug-level messages	

Syslog is inherently a simplex UDP protocol. No communications from a Transport Receiver to a Transport Sender are required. However, certain transport protocols (e.g., TCP as in

[NR01] and [GL12] or TLS as in [MMS09]) or security mechanisms (e.g., IKEv2) used with Syslog may require two-way communications.

5.2 The Syslog-Sign Protocol

Signed syslog messages, as defined in [KCC10], are an enhancement to the Syslog protocol that specifies message origin authentication, message integrity, replay resistance, message sequencing, and detection of missing messages. The listed security services of [KCC10] are provided with a minimal overhead or impact on an existing syslog system. It is possible for current infrastructures to deploy [KCC10] as an overlay system to obtain its security services without changing the basic mode of operation. This enhancement defines a signature block, which carries a separate digital signature for an entire group of previously sent messages. Facility and Severity (PRI) values are examined to determine whether messages are sent to single or multiple Collectors and to send the signatures to the appropriate places as well. Certificate blocks indicate *how* the messages were signed for later verification. UDP messages may be delivered out of sequence, but [KCC10] shows how a Relay or Collector can examine the signature block and determine the correct order. Sequenced delivery can also alert the reviewer to replayed messages. Use of the syslog-sign signature block therefore allows a Collector to verify message integrity, message authenticity, and reliable, replay-free delivery.

5.3 Syslog Message Lengths

Implementations MUST support all UDP packet sizes (for IPv4 and IPv6) and the total message length defined in [KCC10]. IP-layer fragmentation SHOULD be avoided. A minimum PMTU of 2048 is RECOMMENDED, and Transport Receivers SHOULD, in addition, accept longer UDP packets. Also, NAs deploying syslog over UDP SHOULD ensure a low packet loss rate. Because these conditions may not always be achievable, implementations SHOULD provide TCP [GL12] as an alternative.

Implementations SHOULD allow administrators to set a longer maximum UDP packet size if they can verify that such messages can be delivered to all Relays and Collectors without IP fragmentation. Automated determination of a longer PMTU is NOT RECOMMENDED.

Originator applications SHOULD be designed so that more important data items in Syslog messages appear closer to the beginning and SHOULD truncate messages to an acceptable size before integrity checks such as signatures are generated, so that later truncation [Ger09] does not occur and invalidate such integrity checks.

5.4 NTP and Time Stamps

Time stamps are used in the analysis of log records. Messages may not be delivered in order, so after messages are received by a Collector, time stamps allow the messages to be sorted into the order in which they were generated to recapture a sequence of events. Time stamps also aid in sorting messages into event order when multiple Originators transmit messages to a single Collector. Syslog’s TIMESTAMP has a maximum resolution of one microsecond, so Originators SHOULD use the greatest precision available up to this limit.

The Network Time Protocol (NTP) [Mil10] synchronizes system clocks among a set of distributed time servers and clients. Clock synchronization allows events to be correlated and reconstructed when system logs are collected and compared. The NTP protocol is RECOMMENDED to synchronize the time stamps included in syslog messages. Originators SHOULD use the timeQuality SD-ID (see Section 9), registered with IANA and described in [Ger09], to describe the following parameters:

- tzKnown: indicates whether the Originator knows its time zone
- isSynced: indicates whether the Originator is synchronized to a reliable external time source, e.g., via NTP
- syncAccuracy: indicates the Originator's perception of its time synchronization accuracy

6 Log File Records

This section defines seven types of log records for Originators that implement the Security Extension for UNI and NNI version 2.0 [OIF12a]. Vendor-specific and user-specific types are also included and left undefined. Each type MUST be formatted as STRUCTURED-DATA [Ger09] with its own identifier (SD-ID) and parameters (SD-PARAM). SD-PARAM names MUST NOT contain spaces, so spaces MUST be replaced with the underscore character "_". Additional free-form information MAY be included in any of these records, but such additional information MUST follow the REQUIRED and RECOMMENDED information. More than one of the SD-IDs defined in this section MUST NOT be included in a single syslog record (but other information such as timeQuality or origin MAY be included if doing so does not risk message truncation). Values of parameters are encoded in UTF-8 shortest form [YER03], unless otherwise specified. Generation and transmission of these records are subject to source filtering.

1. The NE_ID@26041 SD-ID[1] identifies the network element (Originator). It is written periodically and after reboots or configuration changes. Each NE_ID@26041 record MUST contain the following parameters:

   - The Originator's name "NAME=" with value equal to FQDN
   - For each OIF control plane protocol implemented, "PROT=" with value equal to an OIF protocol, e.g., "UNI-C," "UNI-N," or "E-NNI"
   - Following each "PROT=," one or more "VER=" with values equal to the versions of the protocol supported, e.g., "UNI1.0r2," "UNI2.0." etc.

   Each NE_ID@26041 record MAY contain the following parameters:

[1] IANA has assigned the value 26041 to the OIF (www.iana.org/assignments/enterprise-numbers). The OIF should keep IANA's point of contact with the OIF's Executive Director current.

- Additional protocols or options, including the Security Extension or specific signaling, routing, and discovery protocols (e.g., RSVP or OSPFv2), specified as “OPTION=”
- The event (e.g., re-boot) triggering the message, “EVENT=”
- Security details (e.g., IPsec parameters) “SEC_DETAILS=”

This record MAY contain additional IANA-registered structured data fields of types, for example, “timeQuality,” after the NE_ID@26041.

2. The CONFIG@26041 SD-ID describes the NE’s configuration. It is generated periodically and when the configuration changes. Each CONFIG@26041 record MUST contain the following parameters:
    - The Originator’s name “NAME=” with value equal to FQDN
    - List of interfaces “INFC=” (name or number) followed by their addresses “ADDR=” (one or more for each); types “TYP=” (between control plane NEs, between management plane NEs, between control plane NE and management plane NEs, OAM&P); and protocols “PROT=” supported on each interface
    - Description of configuration changes (additions “ADD=”, deletions “DEL=”, and changes “CHG=”) since the last OIF_CONFIG message

    This record MAY contain additional IANA-registered structured data fields of types, for example, “timeQuality,” after the NE_ID@26041.

The CALL and CONNECT SD-IDs are intended to provide a trace of UNI and E-NNI abstract messages, whereas the PROT SD-ID is intended to capture the RSVP-TE encodings of these messages. See Section 10, below, for details on encoding binary data.

Note that examining Syslog messages containing the CALL, CONNECT, or PROT SD-ID for a node’s received and transmitted signaling messages allows one to correlate parameters such as Local Connection IDs used in the upstream and downstream directions.

3. The CALL@26041 SD-ID logs each abstract call messages implicitly sent or received by the SCN at the UNI-C, UNI-N, or E-NNI. Each CALL@26041 record MUST contain the following parameters, where applicable:
    - Received or transmitted indication, “DIR=R” or “DIR=T”, respectively.
    - Abstract message type, “TYPE=<value>”, where <value> is one of the ten implicitly signaled CALL messages in Table 5-8 of [UNI2.0] (or equivalently in Table 5 of [E-NNI2.0]) and formatted as specified above, e.g., TYPE=CALL_SETUP_REQUEST.
    - At the UNI-C or UNI-N, call message attributes, as specified in Tables 5-11 through 5-17 and 5-19 through 5-21 of [UNI2.0] and formatted as specified above, e.g., “Source_TNA_Name=<value>”.

- At the E-NNI, call message attributes, as appropriate, as listed in Table 17 of [E-NNI], and formatted as specified above, e.g., “DST_SNP_ID=<value>”.
- The following SD-PARAMs used to represent traffic parameters, as applicable (see [UNI2.0] Section 10.13.2.3): ST (for Signal type and Signal Type), RCC, NCC, NVC, MT (for Multiplier), Transparency, Profile, MNC, MVC, CIR, CBS, EIR, EBS, CM, and CF.
- If the security extension is implemented, an indication of whether security is applied to this message, as follows:
  - SEC=N, no protection
  - SEC=I, integrity protection
  - SEC=CI, confidentiality and integrity protection

Implementers are encouraged to follow these same conventions and keep the CALL@26041 SD-ID values for SD-PARAM TYPE and SD-PARAMs for attributes up to date with future Implementation Agreements and their Addenda.

In addition to filtering based on Syslog header fields and Syslog SD-ID (see Section 11), implementations MAY provide fine-grained control of which abstract messages and attributes are logged.

4. The CONNECT@26041 SD-ID logs abstract connection messages at the UNI or E-NNI. Each CONNECT@26041 record MUST contain the following parameters, as appropriate:

- Received or transmitted indication, “DIR=R” or “DIR=T”, respectively.
- At the UNI, the abstract message type, TYPE=<value>, where <value> is one of the types listed in Tables 5-9 and 5-10 of [UNI2.0] and formatted as specified above, e.g., TYPE=CONNECTION_SETUP_REQUEST.
- At the UNI, connection message attributes, as specified in Tables 5-11 through 5-21 of [UNI2.0] and formatted as specified above, e.g., “Source_TNA_Name=<value>”.
- At the E-NNI, the abstract message type, TYPE=<value>, where value is listed in column 2 of Table 5 in [E-NNI2.0] (e.g., TYPE= ConnectionQueryIndication).
- At the E-NNI, connection message attributes, as specified in Table 17 of [E-NNI2.0], as appropriate, and formatted as specified above, e.g., “Initiating_NCC_PC_ID=<value>”.
- At the UNI and at the E-NNI, for the Signaling Adjacency Message, TYPE=HELLO and the SD-PARAMs MESSAGE_ID_ACK, MESSAGE_ID_NACK, and RESTART_CAP are defined.

- The following SD-PARAMs used to represent traffic parameters, as applicable (see [UNI2.0] Section 10.13.2.3): ST (for Signal type and Signal Type), RCC, NCC, NVC, MT (for Multiplier), Transparency, Profile, MNC, MVC, CIR, CBS, EIR, EBS, CM, and CF.
- If the security extension is implemented, an indication of whether security is applied to this message, as follows:
  - SEC=N, no protection
  - SEC=I, integrity protection
  - SEC=CI, confidentiality and integrity protection

Implementers are encouraged to follow these same conventions and keep the CONNECT@26041 SD-ID values for SD-PARAM TYPE and SD-PARAMs for attributes up to date with future Implementation Agreements and their Addenda.

In addition to filtering based on Syslog header fields and the SD-ID (see Section 11), implementations MAY provide fine-grained control of which abstract messages and attributes are logged.

5. The PROT@26041 SD-ID logs messages received and transmitted for each OIF control plane protocol (i.e., signaling, routing, and discovery). Each PROT@26041 record MUST contain the following parameters:

- The protocol name “PROT=” , e.g., RSVP-TE or OSPFv2
- Time of receipt “R-TIME=” or transmission “X-TIME=”, if different from the Time Stamp of the log record
- Interface on which it was received “R-INFC=” or sent “X-INFC=” (with the same name or number as listed in the CONFIG@26041 SD-ID record)
- Address of originating party “O-ADDR=”
- The exact contents (header and body) of the received or transmitted message “MSG=” (possibly truncated to conform with the maximum message size).
- If the security extension is implemented, an indication of whether security is applied to this message, as follows:
  - SEC=N, no protection
  - SEC=I, integrity protection
  - SEC=CI, confidentiality and integrity protection

6. The SEC@26041 SD-ID describes the security configuration. It is generated periodically and when the security configuration changes (e.g., when security associations are created or expire). Each SEC@26041 record MUST contain the following parameters:

- The security policy as specified in the IPsec SPD “SPD=”
- List of IPsec SAs “SA=” and their parameters (but not their keys)
- Statistics for each SA, including counts of messages transmitted “X-CNT=”, received “R-CNT=”, and in error “E-CNT=”

7. The SUM@26041 SD-ID provides periodic statistics on the OIF protocols (e.g., signaling, routing, link management, etc.) and security system including usage counts, errors, failures, capacity utilization, etc. The exact parameters for each protocol are protocol specific, but each SUM@26041 record SHOULD contain the following:
    - Statistics on configuration and availability
    - Cumulative statistics of OIF_CALL, OIF_PROT, and OIF_SEC records
8. VENDOR@26041 SD-ID. The first parameter MUST be the vendor’s name “V-NM=.” The remaining contents of the VENDOR@26041 SD-ID are outside the scope of this IA.
9. USER@26041 SD-ID. The first parameter MUST be the user’s name “U-NM=.” The remaining contents of the USER@26041 SD-ID are outside the scope of this IA.

Vendors and users may, of course, use their own enterprise numbers instead, but these last two allow an easy way to filter OIF-protocol-related messages based on one enterprise number.

The syntax of these records is defined in [Ger09], [KCC10], and Section 10, below. Records are encoded in printable ASCII format with escapes as defined in [Ger09], except as explicitly specified in Section 10, below, for binary data.

In addition to the messages defined above, other pertinent information MAY be logged with the methods described in the document. In particular:

- Specific OIF control plane IAs may modify or supplement the log messages listed above with messages specific to the functions they define.
- Implementations using the ITU-T’s Alarm Reporting Function described in [X.733] or Security Alarm Reporting Function described in [X.736] MUST provide the capability to format these alarms with the methods in [CG09] and to log them with the methods in this IA.
- Implementations generating SNMP Notifications SHOULD provide the capability to format them with the methods described in [MS09] and to log them with the methods in this IA.

7 Syslog Profile

The Syslog messages defined in this IA MUST contain the two discernable message parts, HEADER and STRUCTURED-DATA, separated by a space as described in [Ger09]. They

MAY contain other information in the MSG field. The total length of a UDP packet MUST comply with Section 8.3 of this IA. The HEADER MUST be encoded as described in [Ger09].

The use of Relays is OPTIONAL. An Originator and Collector MAY be on the same system. Originators and Relays MAY transmit to multiple Relays and Collectors, and Collectors MAY receive messages from multiple Originators and Relays.

This IA uses the syslog protocol as described in [Ger09] over UDP [Okm09] or TLS [MMS09], with the following clarifications:

- Syslog MUST be able to run over UDP as defined in [Okm09] (with destination port 514 and arbitrary source ports) but MAY be configured to run over TCP (port to be assigned) as defined in [GL12] or over TLS using TCP port 6514 as defined in [MMS09].
- As stated in [Ger09], the HEADER MUST contain the Facility and Severity encoded in the PRI field, the VERSION of the specification to which it complies, the TIMESTAMP, the HOSTNAME, the APP-NAME, the PROCID, and the MSGID. The code set used in the HEADER MUST be seven-bit ASCII in an eight-bit field as described in RFC 4234 [CO05].
- Messages that do not conform to this format ([Ger09]) SHOULD be accepted by Relays and Collectors. Note that a Collector may receive messages from Originators that implement this IA and from Originators that do not implement this IA.
- Syslog messages MUST be secured by using the Signed syslog Messages protocol defined in [KCC10] and MAY additionally be secured using (1) IPsec as described in Section 12 of this IA [OIF12a], or (2) TLS as described in [MMS09]. With [KCC10], using SNMPEngineBoots as the REBOOT SESSION ID is OPTIONAL, as long as a mechanism guarantees that the syslog counter *never* wraps around. The use of Kerberos, SSL, SSH, or other security mechanisms is out of scope. Using the methods in RFC 3195 [NR01] or subsequent updates to that document is NOT RECOMMENDED.
- Because signatures on messages are REQUIRED, Relays and Collectors MUST NOT modify, reformat, truncate, or split messages.
- Normal events such as time of day messages and summary statistics SHOULD be logged with Severity Informational (6)."
- Normal changes of state and entire messages (e.g., signaling, routing, IKE) SHOULD be logged with Severity Notice (5).
- All error conditions MUST be logged with a higher (i.e., lower numbered) Severity than Notice. This includes, e.g., protocol errors, security errors, call setup failures, and SA timeouts. Warning (4) SHOULD be used for errors or failures that occur in the normal course of events and are usually correctable. Error (3) SHOULD be used for hardware, software, and protocol events that are unexpected or otherwise violate system

design specifications. Critical (2) SHOULD be used for conditions that may impact service, unexpected system restarts, and potential security violations. Alert (1) SHOULD be used to indicate that an attack or persistent service-affecting condition likely exists.

- Severities Emergency (0) and Debug (7) SHOULD NOT be used for the log messages defined in this IA.
- Passwords (including erroneous passwords) and cryptographic keys used for integrity checks, confidentiality, or entity authentication MUST NOT be logged.
- Methods MUST be provided for a NA to turn logging on and off, selectively and securely.
- Originators MUST use PARAM fields within STRUCTURED-DATA [Ger09] to log binary (non-printable) data, such as the exact contents of Control Plane messages. The SD-ID MUST identify the type of binary data and the SD-PARAM "FMT=" MUST indicate the encoding. The encoding MUST be FMT=Base64 (without the trailing \n) as specified in [Jos06] or FMT=UTF-8 as specified in [Yer03]. (IPv4 and IPv6 addresses embedded in printable strings MUST however be written in dotted decimal and RFC-4291 [HD06] formats, respectively.)
- Transport Senders MUST follow the UDP packet length guidelines specified in Section 8.3 of this IA. Transport Receivers SHOULD, where possible, accept longer messages.
- The Facility for the messages defined in Section 9 MUST be numerical code 3 (system daemons), so the PRI values for Severities 1–6 MUST be 25–30.
- Originators generating syslog messages MUST use timestamps as specified in [Ger09].
- If IPv4 or IPv6 addresses are used as names (contrary to recommendation), Originators MUST consistently use only one such address, regardless of which address and interface is used to transmit the Syslog message.

8 Filtering with Syslog

A NE has the capability to generate a large volume of audit data depending on the level of logging that is performed. It is useful to have mechanisms for fine-grained filtering of unwanted syslog messages at various locations within a given syslog system (Figure 1).

This IA specifies a common logging format and list of potential events to log. It does not specify whether these events and data are actually logged. When using the methods in this IA, a NA MUST have the ability to configure what events cause an Originator to generate a syslog record based on all HEADER fields and the single @26041 SD-ID in such messages. The HEADER fields and unique @26041 SD-ID if present SHOULD be used on all Originators and Relays to define filtering and forwarding rules for delivering syslog messages to appropriate Collectors.

NAs MUST use a secure method to update this configuration. If the configuration is updated over a networked connection, then one of the methods in [OIF12b] MUST be used to protect this process. For example, alternative methods of securely carrying out this process include a secured remote command interface (with, e.g., Secure Shell [SSH]), secure file transfer (based, e.g., on SSH), Kerberized telnet, possibly a Web server on the NE protected with TLS and Web Services Security, or any of these methods secured with an IPsec VPN.

If an Originator is configured to forward syslog messages to a Relay or a Collector based on SD-ID or the HEADER fields, care must be taken to ensure that the Signed syslog Messages protocol is not invalidated. If filtering is to be performed on syslog Relays, the Relay needs to exercise similar care. A message MUST be passed through the Relay exactly as it was received. It is strongly RECOMMENDED that an Originator or Relay forward either all or no messages in a given Signature Group [KCC10]. Finally, a Collector SHOULD have the ability to filter received Syslog messages.

The most commonly used method for specifying how filtering is done with the version of syslog defined in [Lon01] is with a syslog.conf file. See [FreeBSD] and Appendix A for a description of syslog.conf and its format. Here, the disposition of syslog records may be specified according to their Hostname, TAG, Severity, and Facility. The Hostname and TAG fields allow the messages listed in Section 9 to be categorized and controlled according to program, process, and NE (Originator) on which they were generated. The Severity value allows error conditions to be distinguished from routine entries. When using a similar approach, the syslog system SHOULD examine the status of the syslog.conf file at least once per minute to check for updates, and NAs MUST use a secure method to update this file. If syslog.conf is used and the syslog.conf file is updated over a networked connection, then the methods in [OIF12b] MUST be used to protect this process. Examples of alternative methods of securely carrying out this process are listed above.

9 Security for Syslog

Appendix B contains a description of potential security threats against a system using syslog. Section 12.1 specifies a REQUIRED mechanism to protect the authenticity and integrity of syslog messages, and Section 12.2 specifies OPTIONAL methods of protecting their confidentiality as well.

9.1 Message Authentication and Integrity

If the integrity of messages is not guaranteed, then an attacker can inject forged messages, intercept and modify messages, or replay messages.

The usual method of guaranteeing message origin authenticity and message integrity is to append a message authentication code (MAC), that is, a secure checksum computed with a shared, secret key. This is, however, insufficient for protocols like email, where the secured message may traverse many systems and need to be stored and later verified. In this case, and in the somewhat similar case of syslog records, digital signatures are preferred, because they can, in principle, be verified by any party at any later time and do not rely on a shared secret, which is difficult to manage over a long timeframe or when shared among more than two parties. Time stamps in the signed syslog messages can be used to verify that replay of stale messages has not occurred. The method specified for digitally signing syslog records in [KCC10] MUST be implemented. Messages that cannot be authenticated MUST be discarded. This Signed syslog Messages protocol specifies both efficiency measures to reduce the computational requirements of computing digital signatures and protocol-specific measures to help ensure that messages of different priorities delivered to different Collectors can be verified together. Some further points about the Signed syslog Messages protocol [KCC10] are as follows:

- The mechanism for delivering the digital signatures is to generate signature blocks (messages) containing the hashes of the messages being signed and a digital signature on the sequence of hashes. That is, individual syslog messages are hashed, the hash is stored separately from the message, and, for efficiency, a single signature is applied to a batch of such hashes.
- Because Severity and Facility values, which are always positive integers, are used to categorize messages and designate whether they should be sent to one or more Collectors, signature blocks can be batched by Severity and Facility values or ranges of Severity and Facility values. Signature Groups SG = 0 and SG = 1 MUST be implemented. SG = 2 and SG = 3 MAY be implemented as well.
- Key management information is sent between the Originator and Collector in certificate blocks (messages). Within the certificate blocks are fields that denote the type of the key material (e.g., PKIX or OpenPGP certificates). To guarantee interoperability, VER = 0111 MUST be implemented. Key Blob Types C, K, and N MUST be implemented; Key Blob Type P SHOULD be implemented, and Key Blob Type U MAY be implemented.
- Message origin authentication and message integrity (as well as support for a non-repudiation service) can be established upon verification of signature blocks, which may be received and processed on-line or stored for later use.

If UDP is the (unreliable) transport protocol, security packets may be lost. Therefore, the guidelines in [KCC07] for resending critical packets SHOULD be designed into the protocol. The times and counts should be configured administratively according to expected traffic characteristics.

9.2 Message Confidentiality

Confidentiality may be needed, as an additional, optional security service, to prevent an attacker from gaining knowledge about a network, its usage, and its users' activities. Competitors of the network operator and the network's users may be motivated to eavesdrop on syslog messages to obtain such information. If syslog is run over the UDP transport defined in [Okm09], the recommended method for securing management messages with IPsec specified in Section 5.1 of [OIF12b] MAY be used to ensure the confidentiality of Syslog messages. IPsec can provide key management, authentication, message integrity, replay detection, and confidentiality at the IP layer. If syslog transmissions are secured with IPsec, the following notes apply in this IA:

- Because confidentiality is the main security service provided by this protocol, ESP SHOULD be used and the ESP encryption algorithm SHOULD NOT be NULL. Note, however, that using ESP with NULL encryption (or AH) does provide some protection against certain denial of service attacks.
- Even though the primary reason for using IPsec is confidentiality, the message integrity service of ESP MUST be used to guard against various attacks based on tampering with the ciphertext in certain modes of operation, and the anti-replay service SHOULD be used.
- When dealing with network configuration issues such as NAT, especially with IPv4, using ESP in Tunnel Mode and UDP encapsulation to traverse NAT is RECOMMENDED. In other cases, Transport Mode has lower overhead and may be preferable.
- The SA used to protect syslog between a Transport Sender and a Transport Receiver MAY be used to protect traffic transported over other protocols or ports as well.

TLS as described in [MMS09] and profiled in [OIF12b] MAY be used as an alternative to IPsec if (a) confidentiality is required and using IPsec is not a practical choice; (b) UDP message lengths force fragmentation; or (c) congestion control, lacking in UDP, is needed.

10 Rationale for the Choice of Security Mechanisms

The security measures proposed in [NR01] are NOT RECOMMENDED because (1) they add significant communications overhead; (2) the usefulness of digital signatures for the long-term security of log messages is an overwhelming reason for choosing the Signed syslog Messages protocol instead; and (3) the application-layer security methods in [NR01] have certain shortcomings. The UDP transport protocol may be preferred for syslog, because it is important to provide the most efficient transport possible when the Originator's system is operating at full capacity.

The main purpose of this IA is to enable logging of the OIF's signaling and routing protocols, which are secured during transmission using IPsec [OIF12a]. The rationale for specifying IPsec as an optional confidentiality mechanism for syslog is that a NE running [OIF12a] already contains an implementation of IPsec, which is also a required component of IPv6 [DH98]. Thus, a new security protocol need not be deployed in these Transport Senders. If confidentiality is not needed, then only the Signed syslog Messages protocol in [KCC10] need be used.

The IETF, however, has chosen to specify, as the mandatory-to-implement alternative to UDP, a secure TLS over TCP transport protocol for Syslog [MMS09]. Therefore, if syslog is run over TCP and confidentiality is required, this is an acceptable alternative. Note, however, that IPsec may be more effective than TLS for preventing certain denial of service attacks at the

Network or Transport Layer (e.g., TCP SYN floods or IP fragmentation attacks) or for inhibiting traffic analysis.

Other conceivable methods of encrypting or providing integrity checks for Syslog messages (SSH, S/MIME, or Kerberos, for example) are out of scope for the above reasons. Note also that many of these other methods are oriented towards protecting TCP connections, not UDP.

11 References

11.1 Normative references

[Bra97] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," IETF RFC 2119 (also BCP0014), March 1997.

[CG09] Chisholm, S., and R. Gerhards, "Alarms in SYSLOG," IETF RFC 5674, October 2009.

[CO05] Crocker, D., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," IETF RFC 4234, October 2005.

[E-NNI2.0] Optical Internetworking Forum Implementation Agreement, "OIF E-NNI Signaling Specification," OIF-E-NNI-Sig-02.0, April 16, 2009.

[Ger07] Gerhards, R., "The Syslog Protocol," IETF RFC 5424, March 2009.

[GL12] Gerhards, R., and C. Lonvick, "Transmission of Syslog Messages over TCP," IETF RFC 6587, April 2012.

[HD06] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," IETF RFC 4291, February 2006.

[Jos06] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings," IETF RFC 4648, October 2006.

[KCC10] Kelsey, J., J. Callas, and A. Clemm "Signed Syslog Messages," IETF RFC 5848, May 2010.

[Mil10] Mills, D. et al., "Network Time Protocol (Version 4): Protocol and Algorithms," IETF RFC 5905, June 2010.

[MMS09] Miao, F., Y. Ma, and J. Salowey, "Transport Layer Security (TLS) Transport Mapping for Syslog," IETF RFC 5425, March 2009.

[MS09] Marinov, V., and J. Schoenwaelder, "Mapping Simple Network Management Protocol (SNMP) Notifications to SYSLOG Messages, IETF RFC 5675, October 2009.

[OIF12a] Optical Internetworking Forum Implementation Agreement, "Security Extension for UNI and NNI version 2.0," OIF-SEP-03.2, October 2012.

[OIF12b] Optical Internetworking Forum Implementation Agreement, "Security for Management Interfaces to Network Elements 2.0," OIF-SMI-03.1, October 2012.

[Okm09] Okmianski, A., "Transmission of Syslog Messages over UDP," IETF RFC 5426, March 2009.

[UNI2.0] OIF Implementation Agreement, "User Network Interface (UNI) 2.0 Signaling Specification Common Part," OIF-UNI-02.0-Common, February 2008.

[Yer03] Yergeau, F., "UTF-8, a transformation format of ISO 10646," IETF RFC 3629 (also STD0063), November 2003.

11.2 Informative references

[ATMF02] "Methods for Securely Managing ATM Network Elements— Implementation Agreement," ATM Forum Specification AF-SEC-0179.000, April 2002.

[Confil] "ATM Connection Filtering MIB and Audit Log," ATM Forum Technical Committee Specification AF-SEC-0188.000, July 2002.

[DH98] Deering, S., and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," IETF RFC 2460, December 1998.

[FreeBSD] FreeBSD 8.2 syslog.conf(5) Manual Page, http://www.freebsd.org/cgi /man.cgi?query=syslog.conf&sektion=5, December 2008.

[Lon01] C. Lonvick, "The BSD Syslog Protocol," IETF RFC 3164, August 2001.

[NR01] New, D., and M. Rose, "Reliable Delivery for Syslog," IETF RFC 3195, November 2001.

[OPSEC] Jones, G., "Operational Security Requirements for Large ISP IP Network Infrastructure," IETF RFC 3871 (Informational), September 2004.

[T1M1] "Operations, Administration, Maintenance, and Provisioning Security Requirements for the Public Telecommunications Network: A Baseline of Security Requirements for the Management Plane," ATIS, T1.276-2003, July 2003.

[X.733] ITU-T, "Information Technology - Open Systems Interconnection -System Management: Alarm Reporting Function," ITU-T X.733, 1992.

[X.736] ITU-T, "Information Technology - Open Systems Interconnection - System Management: Security Alarm Reporting Function", ITU-T X.736, 1992.

Appendix A: Description of the syslog.conf File

SYSLOG.CONF(5)   FreeBSD File Formats Manual   SYSLOG.CONF(5)

NAME

syslog.conf -- syslogd(8) configuration file

DESCRIPTION

The **syslog.conf** file is the configuration file for the syslogd(8) program. It consists of blocks of lines separated by *program* and *hostname* specifications (separations appear alone on their lines), with each line containing two fields: the *selector* field which specifies the types of messages and priorities to which the line applies, and an *action* field which specifies the action to be taken if a message syslogd(8) receives matches the selection criteria. The *selector* field is separated from the *action* field by one or more tab characters or spaces.

Note that if you use spaces as separators, your **syslog.conf** might be incompatible with other Unices or Unix-like systems. This functionality was added for ease of configuration (e.g. it is possible to cut-and-paste into **syslog.conf**), and to avoid possible mistakes. This change however preserves backwards compatibility with the old style of **syslog.conf** (i.e., tab characters only).

The *selectors* are encoded as a *facility*, a period (``.''), an optional set of comparison flags ([!] [<=>]), and a *level*, with no intervening white-space. Both the *facility* and the *level* are case insensitive.

The *facility* describes the part of the system generating the message, and is one of the following keywords: **auth**, **authpriv**, **console**, **cron**, **daemon**, **ftp**, **kern**, **lpr**, **mail**, **mark**, **news**, **ntp**, **security**, **syslog**, **user**, **uucp**, and **local0** through **local7**. These keywords (with the exception of mark) correspond to similar ``LOG_'' values specified to the openlog(3) and syslog(3) library routines.

The *comparison flags* may be used to specify exactly what is logged. The default comparison is ``=>'' (or, if you prefer, ``>=''), which means that messages from the specified *facility* list, and of a priority level equal to or greater than *level* will be logged. Comparison flags beginning with ``!'' will have their logical sense inverted. Thus ``!=info'' means all levels except info and ``!notice'' has the same meaning as ``<notice''.

The *level* describes the severity of the message, and is a keyword from the following ordered list (higher to lower): **emerg**, **crit**, **alert**, **err**, **warning**, **notice**, **info** and **debug**. These keywords correspond to similar ``LOG_'' values specified to the syslog(3) library routine.

Each block of lines is separated from the previous block by a *program* or *hostname* specification. A block will only log messages corresponding to the most recent *program* and *hostname* specifications given. Thus, with a block which selects `ppp' as the *program*, directly followed by a block that selects messages from the *hostname* `dialhost', the second block will only log messages from the ppp(8) program on dialhost.

A *program* specification is a line beginning with `#!prog' or `!prog' (the former is for compatibility with the previous syslogd, if one is sharing **syslog.conf** files, for example) and the following blocks will be associated with calls to syslog(3) from that specific program. A *program* specification for `foo' will also match any message logged by the kernel with the prefix `foo: '. The `#!+prog' or `!+prog' specification works just like the previous one, and the `#!-prog' or `!-prog' specification will match any message but the ones from that program. Multiple programs may be listed, separated by commas: `!prog1,prog2' matches messages from either program, while `!-prog1,prog2' matches all messages but those from `prog1' or `prog2'.

A *hostname* specification of the form `#+hostname' or `+hostname' means the following blocks will be applied to messages received from the specified hostname. Alternatively, the *hostname* specification `#-hostname' or `-hostname' causes the following blocks to be applied to messages from any host but the one specified. If the hostname is given as `@', the local hostname will be used. As for program specifications, multiple comma-separated values may be specified for hostname specifications.

A *program* or *hostname* specification may be reset by giving the program or hostname as `*'.

See syslog(3) for further descriptions of both the *facility* and *level* keywords and their significance. It is preferred that selections be made on *facility* rather than *program*, since the latter can easily vary in a networked environment. In some cases, though, an appropriate *facility* simply does not exist.

If a received message matches the specified *facility* and is of the specified *level (or a higher level)*, and the first word in the message after the date matches the *program*, the action specified in the *action* field will be taken.

Multiple *selectors* may be specified for a single *action* by separating them with semicolon (``;'') characters. It is important to note, however, that each *selector* can modify the ones preceding it.

Multiple *facilities* may be specified for a single *level* by separating them with comma (``,'') characters.

An asterisk (``*'') can be used to specify all *facilities*, all *levels*, or all *programs*.

The special *facility* ``mark'' receives a message at priority ``info'' every 20 minutes (see syslogd(8)). This is not enabled by a *facility* field containing an asterisk.

The special *level* ``none'' disables a particular *facility*.

The *action* field of each line specifies the action to be taken when the *selector* field selects a message.There are five forms:

+o A pathname (beginning with a leading slash). Selected messages are appended to the file.

To ensure that kernel messages are written to disk promptly, **syslog.conf** calls fsync(2) after writing messages from the kernel. Other messages are not synced explicitly. You may prefix a pathname with the minus sign, ``-'', to forego syncing the specified file after every kernel message. Note that you might lose information if the system crashes immediately following a write attempt. Neverthe- less, using the ``-'' option may improve performance, especially if the kernel is logging many messages.

+o A hostname (preceded by an at (``@'') sign). Selected messages are forwarded to the syslogd(8) program on the named host. If a port number is added after a colon (`:') then that port will be used as the destination port rather than the usual syslog port.

+o A comma separated list of users. Selected messages are written to those users if they are logged in.

+o An asterisk. Selected messages are written to all logged-in users.

+o A vertical bar (``|''), followed by a command to pipe the selected messages to. The command is passed to sh(1) for evaluation, so usual shell metacharacters or input/output redirection can occur. (Note however that redirecting stdio(3) buffered output from the invoked command can cause additional delays, or even lost output data in case a logging subprocess exited with a signal.) The command itself runs with *stdout* and *stderr* redirected to */dev/null*. Upon receipt of a SIGHUP, syslogd(8) will close the pipe to the process. If the process did not exit voluntarily, it will be sent a SIGTERM signal after a grace period of up to 60 seconds.

The command will only be started once data arrives that should be piped to it. If it exited later, it will be restarted as necessary. So if it is desired that the subprocess should get exactly one line of input only (which can be very resource-consuming if there are a lot of messages flowing quickly), this can be achieved by exiting after just one line of input. If necessary, a script wrapper can be written to this effect.

Unless the command is a full pipeline, it is probably useful to start the command with *exec* so that the invoking shell process does not wait for the command to complete. Warning: the process is started under the UID invoking syslogd(8), normally the superuser.

Blank lines and lines whose first non-blank character is a hash (``#'') character are ignored. If `#' is placed in the middle of the line, the `#' character and the rest of the line after it is ignored. To prevent special meaning, the `#' character may be escaped with `\'; in this case

```
preceding `\' is removed and `#' is treated as an ordinary character.
```

IMPLEMENTATION NOTES

```
The ``kern'' facility is usually reserved for messages generated by the
local kernel.  Other messages logged with facility ``kern'' are usually
translated to facility ``user''.  This translation can be disabled; see
syslogd(8) for details.
```

FILES

```
/etc/syslog.conf  syslogd(8) configuration file
```

EXAMPLES

```
A configuration file might appear as follows:

# Log all kernel messages, authentication messages of
# level notice or higher, and anything of level err or
# higher to the console.
# Don't log private authentication messages!
*.err;kern.*;auth.notice;authpriv.none;mail.crit      /dev/console

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none        /var/log/messages

# Log daemon messages at debug level only
daemon.=debug                                         /var/log/daemon.debug

# The authpriv file has restricted access.
authpriv.*                                            /var/log/secure

# Log all the mail messages in one place.
mail.*                                                /var/log/maillog

# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                                               *
*.emerg                                               @arpa.berkeley.edu

# Root and Eric get alert and higher messages.
*.alert                                               root,eric

# Save mail and news errors of level err and higher in a
# special file.
uucp,news.crit                                        /var/log/spoolerr

# Pipe all authentication messages to a filter.
auth.*                                |exec /usr/local/sbin/authfilter

# Log all security messages to a separate file.
security.*                                            /var/log/security

# Log all writes to /dev/console to a separate file.
console.*                                             /var/log/console.log
```

```
# Save ftpd transactions along with mail and news
!ftpd
*.*                                                     /var/log/spoolerr

# Log ipfw messages without syncing after every message.
!ipfw
*.*                                                     -/var/log/ipfw
```

SEE ALSO

syslog(3), syslogd(8)

BUGS

The effects of multiple *selectors* are sometimes not intuitive. For example ``mail.crit,*.err'' will select ``mail'' facility messages at the level of ``err'' or higher, not at the level of ``crit'' or higher.

In networked environments, note that not all operating systems implement the same set of facilities. The facilities authpriv, cron, ftp, and ntp that are known to this implementation might be absent on the target system. Even worse, DEC UNIX uses facility number 10 (which is authpriv in this implementation) to log events for their AdvFS file system.

FreeBSD 8.2 December 23, 2008 FreeBSD 8.2

Appendix B: Description of Threats to Log Records

Masquerade Threats

Spoofing:

Usually implemented at the network and data link layers, spoofing refers to forging or manipulating packets to have the identity of Syslog entities or the format of legitimate Syslog packets. Spoofing can be conducted using automated eavesdropping and packet generation processes, to exploit vulnerabilities that allow fake log messages to be inserted and potentially processed or used to overflow the Syslog system and cause denial of service. An intruder gaining access to a network and learning the IP address of a target system can send it spoofed Syslog messages.

Session Hijacking:

Session hijacking occurs at the Network layer and is the process by which an intruder takes control of a TCP session or forges UDP responses. When IPv4 Source Routing or IPv6 Routing Headers are used, an intruder is able to inset itself between legitimate parties and gain access to sensitive data. Otherwise, the intruder attempts a blind hijacking of the session, by guessing the responses between a Transport Sender and Transport Receiver and then sending packets designed to take over the session.

Man-in-the-Middle:

A man-in-the-middle attack allows an attacker to intercept Syslog packets and therefore also any Certificate Blocks to insert its own certificate. The attacker succeeding in this process will be able to receive event messages from the Originator, relay messages, insert new messages, or delete them before passing them along.

Availability Threats

Syslog environments can be interrupted, when any malicious person sends enough messages to fill up the log space. Such attacks are carried out by writing programs or scripts that generate a large number of Syslog messages.

Successful Denial of Service (DoS) attacks launched from a remote Originator can send Syslog messages containing escape sequences that evade filtering and therefore may disrupt normal console operations. It is also possible for an attacker to flood the Relay or Collector system with plausible-looking messages, Signature Blocks, or Certificate Blocks. Logs may be an important input to intrusion detection systems. Without the availability of log messages, the IDS is threatened. Proper monitoring of disk space by the administrator helps avoid some of these problems.

Unauthorized Access

An intruder gaining unauthorized access to a network system that supports Syslog may be capable of hindering the operation and collection of log data. For example, a buffer overrun may be exploited on a system, which could possibly allow unauthorized access.

- If an attacker were to gain normal or privileged access, that entity may be able to change several management objects within the syslog system and considered sensitive or vulnerable. Examples include hosts tables, process tables, security parameters, and system defaults.

When monitoring the UDP port for packets containing event log information, some Syslog services may run with “super-user” or “root” privileges. If an intruder were to gain access to this privileged role during the process of capturing data, it would gain unauthorized access to critical systems and resources. Strong authentication methods should be required for those administrative roles and the elevated privileges. Authentication ensures the identity of the communicating party and provides a basic mechanism for logging and auditing the activities taken place [T1M1].

Data Integrity

Syslog must be able to guarantee the integrity of the logs, and the administrator viewing the data must be able to trust that the logs have not been altered.

Because the goal of adversaries is often to corrupt or erase log records and do their best to hide their activities (or at least make it difficult to prove what has happened), one main purpose of this document is to specify mechanisms for securing Syslog. The security mechanisms specified herein for Syslog consist of ways (1) to sign messages and thus ensure their source, integrity, timeliness, and correct sequencing, which can be verified upon receipt or later, and optionally (2) to encrypt their contents for transmission and thus ensure their confidentiality as they are sent to Transport Receivers.

Appendix C: List of Companies Belonging to OIF When Document Was Approved

Acacia Communications
Agilent Technologies
Altera
Amphenol Corp.
Applied Communication Sciences
Avago Technologies Inc.
Brocade
China Telecom
Cisco Systems
Cogo Optronics
ADVA Optical Networking
Alcatel-Lucent
AMCC
Anritsu
AT&T
Broadcom
Centellax, Inc.
Ciena Corporation
ClariPhy Communications
Comcast

Cortina Systems
Dell, Inc.
Deutsche Telekom
Ericsson
EXFO
Fiberhome Technologies Group
France Telecom Group/Orange
Fundacao.CPqD
GigOptix Inc.
Hitachi
Huawei Technologies
Infinera
IPtronics
Juniper Networks
KDDI R&D Laboratories
LeCroy
Luxtera
Marben Products
Mindspeed
Molex
NEC
Nokia Siemens Networks
Oclaro
PETRA
PMC Sierra
Reflex Photonics
SHF Communication Technologies
Sumitomo Osaka Cement
Tektronix
TeraXion
TriQuint Semiconductor
Verizon
Xilinx
Yamaichi Electronics Ltd.
CyOptics
Department of Defense
Emcore
ETRI
FCI USA LLC
Finisar Corporation
Fujitsu
Furukawa Electric Japan
Hewlett Packard
Hittite Microwave Corp
IBM Corporation
Inphi
JDSU
Kandou
Kotura, Inc.
LSI Corporation
M/A-COM Technology Solutions, Inc.
Metaswitch
Mitsubishi Electric Corporation
MoSys, Inc.
NeoPhotonics
NTT Corporation
Optoplex
Picometrix
QLogic Corporation
Semtech
Sumitomo Electric Industries
TE Connectivity
Tellabs
Time Warner Cable
u2t Photonics AG
Vitesse Semiconductor
Xtera Communications