Contribution Number: oif2003.119.06

Working Group: OAM&P

TITLE: OIF Control Plane Logging and Auditing with Syslog

DATE: October 19, 2004

SOURCE: Tom Tarman, Sandia National Labs, tdtarma@sandia.gov
Renée Esposito, Booz Allen Hamilton, esposito_renee@bah.com
Richard Graveman, Department of Defense, rfg@acm.org

Document Status: Baseline text

Project Name: Security for Management Interfaces to Network Elements and Auditing & Logging for Network Elements

Project Number: oif2002.346

ABSTRACT: This IA specifies how to use 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 NNI intended to be used in conjunction with the *Security Extension for UNI and NNI.*

For additional information contact:
The Optical Internetworking Forum, 39355 California Street,
Suite 307, Fremont, CA 94538
510-608-5990 phone, info@oiforum.com

List of Contributors

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

TABLE OF CONTENTS

LIST OF FIGURES

OIF Control Plane Logging and Auditing with Syslog

1 Document Summary

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 NNI.

1.1 Working Group

OAM&P.

1.2 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 log records to help identify and respond to protocol errors, security violations, unauthorized modifications, and other malfunctions.

The OIF has defined a method for secure transmission of signaling 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 protocols, allow end users or non-adjacent signaling entities to obtain assurance that the messages delivered end to end accurately represent the remote party's intentions. Securing such protocols from end to end or across non-adjacent entities appears to be a difficult problem to solve efficiently with commonly used protocol security technology. Therefore, the ability to log such signaling messages as they traverse multiple UNI or NNI interfaces adds a useful and effective way to increase assurance that the signaling entities are operating securely and correctly.

1.3 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 over UDP 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 NNI* [OIF03a], but it MAY be used in other cases as well.

This specification covers audit data formats, encoding, records and fields, transport protocols, control of auditable events, and security.

1.4 Expected Outcome

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 protocols are functioning properly; and (3) to define necessary and appropriate security services and mechanisms for the logging protocol.

This IA defines multiple types of log messages. Some 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 audit only desired fields or all fields and can be enabled or disabled as required.

1.5 Value to OIF

This IA complements and enhances the usefulness of the methods defined in UNI 1.0, E-NNI, and the *Security Extension for UNI and NNI* 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.

1.6 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 syslog WG and opsec WG.
- The ATM Forum specification on auditing and logging [Confil].
- T1M1's reference diagrams and terminology in their Security Requirements [T1M1].

1.7 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, when deployed.

The components of a NE have one or more control (i.e., signaling) and management (i.e., OAM&P) interfaces as well as interfaces to other components of the same NE. Syslog, the logging and auditing mechanism defined in this IA, can be configured to generate log records describing 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 three operational roles for the Syslog service are designated devices, relays, and collectors. A device is a source that generates a log message. A collector, also known as a Syslog server, receives log messages and does not forward them further. 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 devices and collectors: a device may communicate with many collectors and vice versa. A relay sits between a collector and a device. Relays have multiple roles; they accept Syslog messages from devices and other relays and forward those messages to collectors or other relays. Thus, it is possible to have multiple relays between a device and a collector. For administrative reasons a relay might authenticate all of the devices within a given perimeter and authenticate itself in turn to a boundary collector. It may filter messages based on facility and severity and selectively forward messages accordingly. Devices, collectors, and relays may run on separate systems, or their functionality may be combined on a single system. Thus, a single system may act in more than one of these three roles.

Figure 1 below illustrates how the Syslog process is designed to operate: devices send messages to relays; relays accept and forward messages to other relays or directly 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 devices that actively interact with relays or collectors to compile the data received for review.

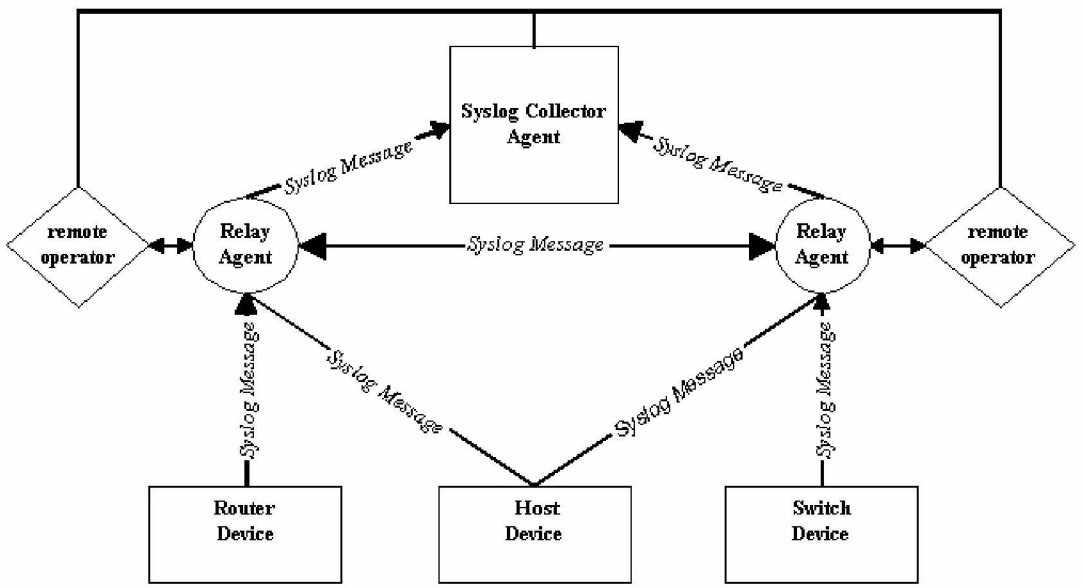

Figure 1: Syslog System Diagram.

1.8 Acknowledgements

This IA uses prior work in the OIF and the work in the other standards development organizations listed above, in particular the Syslog [Ger04], Syslog over UDP [Okm04], and Syslog-sign [KC04] protocols defined by the IETF. The main thrust of this IA, to allow users to diagnose and verify end-to-end behavior of signaling protocols, evolved from the work cited in the ATM Forum [Confil]. Security requirements for logging and

auditing were developed in the OIF [SecReq], 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 the OIF later specified a profile of IPsec [OIF03a] applied in this IA.

2 Introduction

2.1 Document Organization and Outline

Section 1 covers the problem statement, scope, expected outcome, and relationship to other work. Section 2 contains an outline and guidelines for use. Section 3 contains keywords, terminology, and acronyms. Section 4 combines, paraphrases, and reiterates logging requirements documented by the OIF and IETF for reference. Section 5 introduces the Syslog protocol and other supporting protocols. Log records and data to be logged are covered in Section 6. Section 7 specifies exactly how the Syslog protocol and transport are used in this IA. Section 8 describes methods of controlling the generation of the log data. Section 9 defines methods for securing the disposition of log data. Normative and informative references comprise Section 10.

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, 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 [Ger04], message segmentation [Okm04], and signatures to guarantee message authenticity [KC04]. The security specified herein provides an optimized method for end-to-end authentication and integrity guarantees and an optional, additional method 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 NNI* [OIF03a]. Procedures for on-line and off-line processing of signed log records are included in the referenced IETF documents, in particular [KC04].

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].

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 may receive Syslog messages and not forward them further.

Device: A system, for example a UNI client or a NE, that can generate log messages.

Element Management System: A terminal, network element, or system that provides specific services to manage specific NEs.

HEADER: The first part of a Syslog record containing a VERSION, FACILITY, SEVERITY, TIMESTAMP, HOSTNAME, SENDER-NAME, and SENDER-INST. Note that several differences exist between the PRI, HEADER, and TAG defined in [Lon01] and the HEADER defined in [Ger04] and used in this IA.

Management System: A generic term for an Element Management System or Network Management System. It may also function as a relay or collector (see below).

MSG: The second and remaining part of a Syslog record after the HEADER, which contains the detailed 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.

Network Management System: A terminal, NE, or system that provides services to manage a NE. It may be an overall management system that manages multiple NEs and Element Management Systems.

Relay: A system, possibly a Management System, that can receive and forward log messages.

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
ESP	Encapsulating Security Payload
IANA	Internet Assigned Numbers Authority
IKE	Internet Key Exchange
IP	Internet Protocol
IPsec	IP Security
MAC	Message Authentication Code
MIB	Management Information Base
MTU	Maximum Transmission Unit
NA	Network Administrator
NAT	Network Address Translation
NE	Network Element
NNI	Network Node Interface
NTP	Network Time Protocol

RFC Request for Comments
SA Security Association
SAD Security Association Database
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
VPN Virtual Private Network

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 device. An important component of this is to supplement the security provided for signaling in [OIF03a] 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 protections, such as access controls, should be used to restrict access to the log data on different devices, 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 the period of time over which the logs are collected and retained. Using off-line backup storage media is recommended.

The following requirements for logging and auditing have been documented by the IETF [OPSEC] and OIF [SecReq] 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 device.* 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's integrity.
- *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)*
  - *Device 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.
- *Devices 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 devices 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 originating devices at the highest supported resolution. Granularity and synchronization of timestamps shall be addressed in system specifications.
- *Devices 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). Devices should support a mechanism to allow the administrator to specify the local time zone.

• Classification of Audit Events

- *Devices shall have the ability to classify logged events.* Devices shall assign a classification chosen from a well-known list to all logged messages.
- *A field shall exist in the message to identify the device and, optionally, the process that originated the event.* The message source shall be recorded for later processing or analysis.
- *Devices 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 un-translated 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. 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.* Devices 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 is an unsecured UDP-based protocol for logging events generated on networked systems including NEs and the applications that run on NEs. Such systems are called devices. Syslog provides for relaying and collecting log data to enable on-line or off-line auditing. The Syslog protocol is defined in [Ger04] and the UDP transport mechanism to deliver event messages to a Syslog server in [Okm04]. The format of Syslog messages evolved over time and became an open standard documented by the IETF [Lon01]. Syslog is designed to collect data from devices 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, as follows [Lon01, Gert04]:

Severity Code	Definition
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

5.2 The Syslog-Sign Protocol

Syslog-sign, as defined in [KC04], is 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 Syslog-sign are provided with a minimal overhead or impact on an existing Syslog system. It is possible for current infrastructures to deploy Syslog-sign as an overlay system to obtain its security services without changing the basic mode of operation. The Syslog-sign enhancement defines a signature block, which carries a separate digital signature for an entire group of previously sent messages. Facility and Severity 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 Syslog-sign allows a receiver to look in 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 Segmenting Syslog Messages

Implementations MUST support all UDP packet sizes (for IPv4 and IPv6), the message fragmentation mechanism, and total message length defined in [Okm04]. IP-layer fragmentation SHOULD be avoided. Receivers SHOULD, however, accept and attempt to process longer UDP packets. The mechanism defined in [Ger04] for fragmenting messages is REQUIRED for *all* fragmented messages.

Note: These mechanisms have recently been discussed on the Syslog mailing list (September-October 2004) and are likely to change.

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 MTU is NOT RECOMMENDED.

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 devices send messages to a single collector. Syslog's TIMESTAMP has a maximum resolution of one microsecond, so devices SHOULD use the greatest precision available up to this limit.

The Network Time Protocol (NTP) [Mil92] 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. The following SD-IDs (see Section 6) are registered with IANA:

- time: describes the original sender's notion of system time
- tzknown: indicates whether the original sender knows its timezone
- issynced: indicates whether the original sender is synchronized to a reliable external time source, e.g., via NTP
- syncaccuracy: indicates the original sender's perception of time synchronization accuracy

6 Log File Records

This section defines six types of log records for devices that implement the *Security Extension for UNI and NNI.* Vendor-specific and user-specific types are also included and left undefined. Each type MUST be formatted as Structured Data [Ger04] with its

own identifier (SD-ID) and parameters (SD-PARAM). Additional free-form information MAY be included in any of these records, but such additional information MUST follow the MANDATORY and RECOMMENDED information. More than one of the SD-ID Structured Data fields defined in this section MUST NOT be included in a single Syslog record. Values of parameters are encoded in UTF-8 [YER98], unless otherwise specified. Generation of these records is subject to source filtering.

1. The x-OIF_NE_ID SD-ID identifies the network element (device). It is written periodically and when there are reboots or configuration changes. Each x-OIF_NE_ID record MUST contain the following parameters:

   - The device name "NAME=" with value equal to FQDN
   - OIF control plane protocols implemented "PROT=" with value equal to a comma-separated list of "UNI-C," "UNI-N," or "E-NNI"
   - OIF control plane protocol version implemented "VER=" with value equal to the version of the protocol (PROT=) e.g., UNI 1.0r2, UNI 2.0 etc.

   Each x-OIF_NE_ID record MAY contain the following parameters:

   - Additional protocols, including the Security Extension and specific signaling, routing, and discovery protocols (e.g., RSVP, OSPF)
   - The event (e.g., re-boot) "EVENT="
   - Security details (e.g., IPsec parameters) "SEC_DETAILS="

   This record MAY contain additional IANA-registered structured data fields of types "time," "origin," etc. after the x-OIF_NE_ID.

2. The x-OIF_CONFIG SD-ID describes the NE's configuration. It is generated periodically and when the configuration changes. Each x-OIF_CONFIG record MUST contain the following parameters:

   - The device name "NAME=" with value equal to FQDN
   - List of interfaces "INFC=" (name or number) with their addresses "ADDR=" (one or more each) and types "TYP=" (between control plane NEs, between management plane NEs, between control plane NE and management plane NEs, OAM&P)
   - List of 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 "time," "origin," etc. after the x-OIF_NE_CONFIG.

3. The x-OIF_CALL SD-ID describes each "call" or connection provided by the signaling system. Each x-OIF_CALL record MUST contain the following parameters, as appropriate:

   - Call origination details "C-ORG=" (including addresses and Service Level)

- Other connection attributes “C-ATR=” (including routing, parameters, and billing information)
- Call termination “C-TRM=” (including cause codes or other diagnostics)
- Call modifications “C-MOD=” (including connection ID, call ID and Service Level changes)

An x-OIF_CALL SD-ID record MAY contain other data defined in referenced OAM&P specifications.

4. The x-OIF_PROT SD-ID logs messages received and transmitted for each OIF control plane protocol (i.e., signaling, routing, and discovery). Each x-OIF_PROT record MUST contain the following parameters:
   - The protocol name “PROT=” , e.g., RSVP-TE
   - 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 x-OIF_CONFIG SD-ID record)
   - Address of originating party “O-ADDR=”
   - The exact contents (header and body) of the received or transmitted message “MSG=”
5. The x-OIF_SEC 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 x-OIF_SEC 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=”
6. The x-OIF_SUM 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 x-OIF_SUM record SHOULD contain the following:
   - Statistics on configuration and availability
   - Cumulative statistics of OIF_CALL, OIF_PROT, and OIF_SEC records
7. x-OIF_VENDOR SD-ID. The first parameter MUST be the vendor’s name “V-NM=.” The remaining contents of the x-OIF_VENDOR SD-ID is outside the scope of this IA.
8. x-OIF_USER SD-ID. The first is the user’s name “U-NM=.” The remaining contents of the x-OIF_USER SD-ID is outside the scope of this IA.

The syntax of these records is defined in [Ger04], [KC04], and Section 7, below. Records are encoded in printable ASCII format with escapes as defined in [Ger04], except as explicitly specified in Section 7, below, for binary data. Message fragmentation is described in Section 5.3 of this IA and [Ger04].

7 Syslog Profile

The Syslog message format MUST contain the two discernable message parts, the HEADER and MSG (message) as described in [Ger04]. The total length of a UDP packet MUST comply with Section 5.3 of this IA. The transport Header and HEADER MUST be encoded as described in [Ger04].

The use of relays is OPTIONAL. A device and collector MAY be on the same system. Devices and relays MAY report to multiple collectors, and collectors MAY collect from multiple devices and relays.

This IA uses the Syslog protocol as described in [Ger04] over UDP [Okm04], with the following clarifications:

- Syslog MUST run over UDP and use destination port 514. Source ports MUST be implemented as defined in [Okm04].
- As stated in [Ger04], the header MUST contain the token identifying the message as a syslog message complying with [Ger04], the version of the specification to which it complies, the facility, the severity, the timestamp, the hostname, sender-name and the sender-inst. The code set used in the header MUST be seven-bit ASCII in an eight-bit field as described in RFC 2234 [CO97].
- Messages that do not conform to this format ([Ger04]) SHOULD be accepted by relays and collectors. Note that a collector may receive messages from devices that implement this IA and from devices that do not implement this IA.
- Syslog messages MUST be secured by using the Syslog-sign protocol defined in [KC04] and MAY additionally be secured using IPsec as described in Section 9 of this IA and [OIF03b]. With [KC04], 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 TCP, TLS, SSL, SSH or other security mechanisms is out of scope. The use of RFC 3195 [NR01] is NOT RECOMMENDED.
- Relays and collectors MUST NOT modify, reformat, 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.
- Methods MUST be provided for a NA to turn logging on and off, selectively.
- Devices MUST use Structured Data [Ger04] 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 [KC04] or FMT=UTF-8 as specified in [Yer98]. (IPv4 and IPv6 addresses embedded in printable strings MUST however be written in dotted decimal and RFC-2373 [HD98] formats, respectively.)
- Devices MUST follow the UDP packet length guidelines specified in Section 5.3 of this IA. Relays and collectors SHOULD, where possible, accept and process longer messages.
- Devices MUST use the ExtendedHeader with MessageId, TotalLength, and FragmentOffset described in [Okm04] to split messages. Messages short enough not to be fragmented SHOULD NOT contain the ExtendedHeader. Relays and collectors MUST allow for fragments to be delivered out of sequence.
- The Facility for the messages defined in Section 6 MUST be code = 5196102 = 79*65536 + 73*256 + 70 = "OIF."
- Devices generating Syslog messages MUST use timestamps as specified in [Ger04].
- If IP addresses are used as names (contrary to recommendation), devices MUST use only one such address, regardless of which 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 generate a Syslog record on a device based on all Header fields and the SD-ID of x-OIF messages. The header fields and x-OIF SD-ID SHOULD be used on all devices 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 [OIF03b] 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), secure file transfer (based, e.g., on SSH), SNMP using the Syslog MIB [KP04] and

protected with SNMPv3, Kerberized telnet, possibly a Web server on the NE protected with TLS, or any of these methods secured with an IPsec VPN.

If a device 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 Syslog-Sign protocol is not invalidated. Similarly, 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 a device or relay forward either *all* or *no* messages in a given message group. 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 6 to be categorized and controlled according to program, process, and NE (device) on which they originated. 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 [OIF03b] 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 9.1 specifies a REQUIRED mechanism to protect the authenticity and integrity of Syslog messages, and Section 9.2 specifies an OPTIONAL method 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 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 mange for the long term or when shared among more than two parties. Time stamps in the Syslog messages can be used to verify that replay of stale messages has not occurred. The method specified for digitally signing Syslog records in [KC04] MUST be implemented. Messages that cannot be authenticated MUST be discarded. This 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 Syslog-sign protocol [KC04] 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.
- Key management information is sent between the device 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).
- 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.
- UDP is the (unreliable) transport protocol, so security packets may be lost. Therefore, a mechanism for resending critical packets MAY be designed into the protocol.

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. The recommended method for securing management messages with IPsec specified in Section 6.1 of [OIF03b] 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 require confidentiality, the following notes apply in this IA:

- Because confidentiality is the main security service provided by this protocol, AH MUST NOT be used, and the NULL encryption algorithm of ESP SHOULD NOT be used. (Using ESP with NULL encryption 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, and the anti-replay service SHOULD be used to guard against various attacks based on tampering with the ciphertext in certain modes of operation.
- When dealing with network configuration issues such as NAT, especially with IPv4, using ESP in Tunnel Mode and UDP encapsulation to traverse NAT are

RECOMMENDED. In other cases, Transport Mode has lower overhead and may be preferable.

- The SA used to protect Syslog between devices, relays, and collectors MAY be used to protect traffic transported over other protocols or ports as well.
- Forthcoming revisions to IPsec and IKE (see [OIF05]), when they are specified for protecting OIF Control Plane or other Management Plane traffic, SHOULD be used to protect Syslog messages as well.

9.3 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 Syslog-sign instead; and (3) the application-layer security methods in [NR01] have certain shortcomings. The UDP transport protocol is preferred for Syslog, because it is important to provide the most efficient transport possible when the device is operating at full capacity.

The main purpose of this IA is to enable logging of the OIF's signaling protocols, which are secured during transmission using IPsec [OIF03a]. The rationale for specifying IPsec as the optional confidentiality mechanism is that a NE running [OIF03a] 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 devices. If confidentiality is not needed, then only the Syslog-sign digital signature protocol in [KC04] need be used.

Other conceivable methods of encrypting or providing integrity checks for Syslog messages (SSL, TLS, SSH, 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.

10 References

10.1 Normative References

[Ed. note: The four syslog WG drafts and OIF Addendum to the Security Extension need to be tracked and updated until RFCs and IAs are issued.]

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

[CO97] Crocker, D., and P. Overell, "Augmented BNF for Syntax Specifications: ABNF," IETF RFC 2234, November 1997.

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

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

[Ger04] Gerhards, R., "The syslog Protocol," IETF Work in Progress draft-ietf-syslog-protocol-06, September 2004 (expires March 25 2005).

[HD98] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," IETF RFC 2373, July 1998.

[KC04] Kelsey, J., and J. Callas, "Syslog-Sign Protocol," IETF Work in Progress draft-ietf-syslog-sign-14, April 2004 (expired).

[KP04] Keeni, G., and B. Pape, "Syslog MIB," IETF Work in Progress draft-ietf-Syslog-device-mib-06, September 2004.

[Mil92] Mills, D, "Network Time Protocol (Version 3) Specification, Implementation and Analysis," IETF RFC 1305, March 1992.

[OIF03a] Optical Internetworking Forum Implementation Agreement, "Security Extension for UNI and NNI," OIF-SEP-01.1, May 2003.

[OIF03b] Optical Internetworking Forum Implementation Agreement, "Security for Management Interfaces to Network Elements," OIF-SMI-01.1, October 2003.

[OIF05] Optical Internetworking Forum Work in Progress, "Addendum to the Security Extension for UNI anf NNI," October 2004.

[Okm04] Okmianski, A., "Transmission of syslog messages over UDP," IETF Work in Progress draft-ietf-syslog-transport-udp-02, May 2004 (expires November 2004).

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

[Yer98] Yergeau, F., "UTF-8, a transformation format of ISO 10646," IETF RFC 2279, January 1998.

10.2 Informative References

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

[DH98] Deering, S., and R. Hinden, "Internet Protocol, Version 6 (Ipv6) Specification," IETF RFC 2460, December 1998.

[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.

[OIF01] Optical Internetworking Forum Implementation Agreement, "User Network Interface (UNI) 1.0 Signaling Specification" OIF-UNI-01.0, October 2001.

[OIF04] Optical Internetworking Forum Implementation Agreement, "Intra-Carrier E-NNI Signaling Specification" OIF-E-NNI-01.0, February 2004.

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

[SecReq] Tarman, T., et al., *OAM&P Security Requirements*, Optical Internetworking Forum Contribution oif2002_460_00, November 2002.

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 along on the line), 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) received 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, alert, crit, 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.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. 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's 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 doesn't 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:

- A pathname (beginning with a leading slash). Selected messages are appended to the file.
- A hostname (preceded by an at (``@'') sign). Selected messages are forwarded to the syslogd(8) program on the named host.
- A comma separated list of users. Selected messages are written to those users if they are logged in.
- An asterisk. Selected messages are written to all logged-in users.

- 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 didn't 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's 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.

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  /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

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

     # 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
```

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

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.

SEE ALSO

syslog(3), syslogd(8)

FreeBSD 4.7 June 9, 1993 FreeBSD 4.7

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 source-routing is turned on, an intruder is able to inset itself between legitimate parties and gain access to sensitive data. When turned off, the intruder attempts a blind hijacking of the session, by guessing the responses between a host, relay, or collector 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 sender, relay messages, insert new messages, or delete them before passing them along to the receiver.

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 device 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 device 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 device 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 device, 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 considered sensitive or vulnerable. Examples of these objects are:

- SyslogParamsProcessStatus: may be used to start, stop, or suspend the Syslog process itself.
- SyslogAllowedHostsTable: may affect the hosts from which Syslog messages are accepted. When improperly configured may lead to loss of messages from an important source or a flood of messages from a potentially rogue source.
- SyslogCtlSelectionTable: may affect the selection rules from messages. When improperly configured may lead to loss of relevant messages or the collection of useless, ill-intentioned, messages.
- SyslogCtlLogActionTable: may affect the actions carried on a received Syslog message. When improperly configured may lead to misdirection of messages or loss of relevant messages.
- SyslogCtlUserActionTable: may affect the proper configuration of user lists. May prevent a user from receiving an important message or spamming a users' console.
- SyslogCtlFwdActionTable: may affect the forwarding action of a message. If improperly configured, may prevent messages from reaching a specified destination, may contribute to a directed DoS attack or delivery of a Syslog message to an improper destination- resulting in a breach of user's privacy.
- SyslogCtlPipeActionTable: may affect the commands that invoke the process for logging messages. When improperly configured may cause arbitrary programs to be invoked in the Syslog receiver.

During the monitoring process of 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 relays or collectors.