ADDENDA

ANSI/ASHRAE Addendum bi to
ANSI/ASHRAE Standard 135-2016

A Data Communication Protocol for Building Automation and Control Networks

Approved by ASHRAE on June 15, 2018, and by the American National Standards Institute on June 15, 2018.

Approved American National Standard
ANSI
www.ansi.org

SPECIAL NOTE

This American National Standard (ANS) is a national voluntary consensus Standard developed under the auspices of ASHRAE. *Consensus* is defined by the American National Standards Institute (ANSI), of which ASHRAE is a member and which has approved this Standard as an ANS, as "substantial agreement reached by directly and materially affected interest categories. This signifies the concurrence of more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered, and that an effort be made toward their resolution." Compliance with this Standard is voluntary until and unless a legal jurisdiction makes compliance mandatory through legislation.

ASHRAE obtains consensus through participation of its national and international members, associated societies, and public review.

ASHRAE Standards are prepared by a Project Committee appointed specifically for the purpose of writing the Standard. The Project Committee Chair and Vice-Chair must be members of ASHRAE; while other committee members may or may not be ASHRAE members, all must be technically qualified in the subject area of the Standard. Every effort is made to balance the concerned interests on all Project Committees.

The Senior Manager of Standards of ASHRAE should be contacted for

a. interpretation of the contents of this Standard,
b. participation in the next review of the Standard,
c. offering constructive criticism for improving the Standard, or
d. permission to reprint portions of the Standard.

[This foreword and the "rationales" on the following pages are not part of this standard. They are merely informative and do not contain requirements necessary for conformance to the standard.]

FOREWORD

The purpose of this addendum is to present changes to ANSI/ASHRAE Standard 135-2016. These modifications are the result of change proposals made pursuant to the ASHRAE continuous maintenance procedures and of deliberations within Standing Standard Project Committee 135. The changes are summarized below.

In the following document, language to be added to existing clauses of ANSI/ASHRAE Standard 135-2016 is indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are added, plain type is used throughout.

The use of placeholders like X, Y, Z, X1, X2, etc., should not be interpreted as literal values of the final standard. These placeholders will be assigned actual numbers/letters only with incorporation of this addendum into the standard for republication.

135-2016*bi*-1. Add Audit Reporting.

Rationale

The standard currently has no definitions for an interoperable controller and workstation based audit reporting and logging.

This addendum adds a new Audit Reporter object type and new audit notification services to report auditable actions. A new Audit Log object type and a new audit query service is added to log and retrieve audit notifications.

Both BACnet clients and servers are allowed to report auditable actions. Servers report changes to local objects, clients report successful and attempted changes along with extra information such as reason for change. The consumer of the logs will be responsible for correlating the multiple entries for a single action.

Configuration of audit reporting is supported on a per object basis with different levels of auditing.

[Insert new **Clause 19.Y**, p. 756]

19.Y Audit Logging

Audit logging consists of recording records in audit logs which document the sequence of activities that affect the system.

BACnet audit logging is made up of four actors: the client that requests the operation being logged (the operation source), the server that performs the operation (the operation target), the logger that stores the audit records, and the viewer that consumes the audit log. In this framework, both the operation source and the operation target are given the opportunity to provide audit log records to the audit logger. This allows BACnet devices to provide these features:

1) audit logging when either the operation source or the operation target does not support audit logging;
2) logging of extended audit information that is not available for reporting by the operation target;
3) recording of failed actions by the operation source when the operation target does not respond;
4) recording of failed actions by the operation target where unauthorized operation sources do not log their attempts.

The operation source generates audit log entries which describe the operation it requested and which optionally contain extra information including the BACnet object which is the source of the operation (if applicable), user identification (if applicable), and a description of, or reason for, the operation.

The server generates audit log entries which describe the operation it has been requested to perform.

The logger receives notifications from local and remote operation sources and operation targets and places the notifications in Audit Log objects.

Operation source side auditing is controlled via:

1) an Audit Reporter object in the operation source device which generates the audit notifications; and
2) the Audit_Notification_Recipient property in the operation source's Device object.

Operation target side auditing is controlled via:

1) configuration properties in the object operated upon by the operation being reported;
2) Audit Reporter objects which generate the audit notifications; and
3) the Audit_Notification_Recipient property in the operation target's Device object.

The following figures provide a diagrammatic overview of the architecture discussed in the rest of this clause.

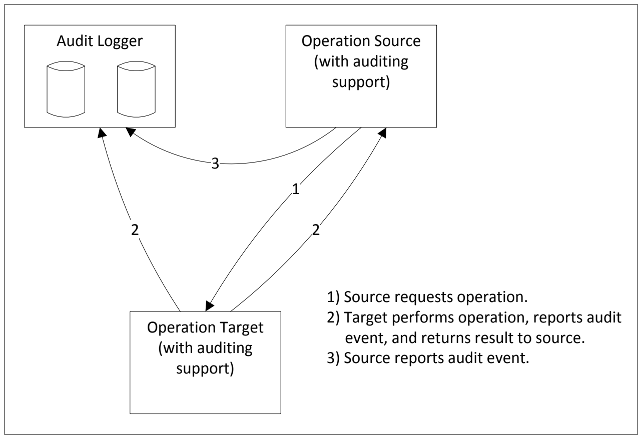
Audit Logger
Operation Source (with auditing support)
Operation Target (with auditing support)
1
2
2
3
1) Source requests operation.
2) Target performs operation, reports audit event, and returns result to source.
3) Source reports audit event.

Figure 19-Y1: General flow of operations and audit notifications from both operation sources and operation targets.

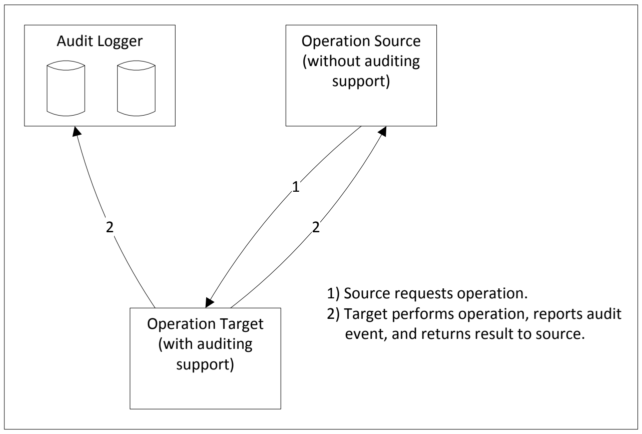
Audit Logger
Operation Source (without auditing support)
Operation Target (with auditing support)
1
2
2
1) Source requests operation.
2) Target performs operation, reports audit event, and returns result to source.

Figure 19-Y2: General flow of operations and audit notifications from operation targets with no operation source notifications.

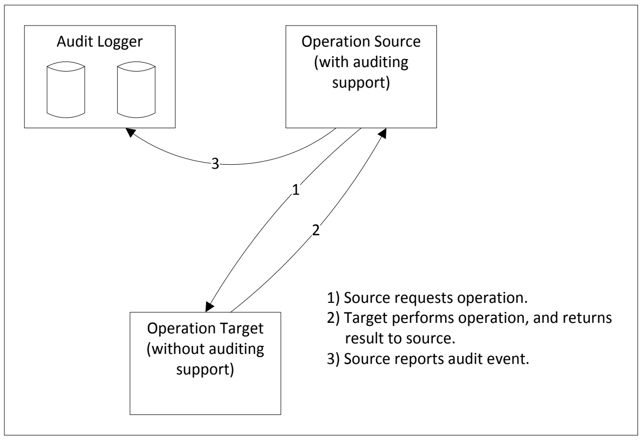
Audit Logger
Operation Source (with auditing support)
Operation Target (without auditing support)
1
2
3
1) Source requests operation.
2) Target performs operation, and returns result to source.
3) Source reports audit event.

Figure 19-Y3: General flow of operations and audit notifications from operation source only.

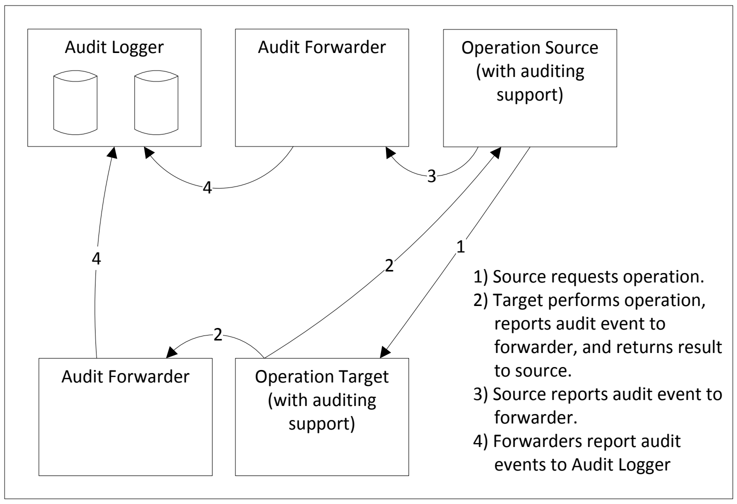
Audit Logger
Audit Forwarder
Operation Source (with auditing support)
4
3
1
4
2
2
Audit Forwarder
Operation Target (with auditing support)
1) Source requests operation.
2) Target performs operation, reports audit event to forwarder, and returns result to source.
3) Source reports audit event to forwarder.
4) Forwarders report audit events to Audit Logger

Figure 19-Y4: General flow of operations and audit notifications from both operation sources and operation targets with audit forwarders.

19.Y.1 Audit Notification Generation

Operation targets generate audit notifications for the following operations (subject to filtering):

1) a BACnet request for an operation type described in Table 19-Y3, which is executed successfully;
2) a local request for an operation type described in Table 19-Y3, which is successfully performed but which is not the result of the normal operation of the object (for example, updating of an input object's Present_Value based on reading of the physical input would not be auditable, but the setting of the Present_Value of a value object by a local program would be auditable); and
3) any action the local device determines to be relevant in an audit log.

It is a local matter whether or not an operation target generates audit notifications for BACnet requests for an operation type described in Table 19-Y3, which fail. It is important to consider when such operations will result in notifications as this could form the basis of a denial of service attack through the increased workload and network traffic required to report the failed operations.

Operation Sources generate audit notifications for the following operations (subject to filtering):

1) a BACnet request for an operation type described in Table 19-Y3, which is requested; and
2) any action the local device determines to be relevant in an audit log.

When reporting a successful operation, no result code is included in the audit notification. When reporting a failed operation, a result code shall be included.

19.Y.1.1 Audit Notification Generation Through Monitoring

It is possible for a device that is not involved in the actions to monitor the network traffic and generate audit notifications on behalf of operation sources and/or operation targets.

Such a product is installed so that it is able to monitor all communications of interest that originate with an operation source (or all operation sources), or all communications destined for an operation target (or all operation targets). When BACnet messages are detected which contain auditable actions, the product generates audit records as if it were the operation source or the operation target device (depending on which device it is performing auditing for.)

The configuration and control of products that provide such functionality is outside the scope of this standard.

19.Y.2 Audit Reporter Objects

Audit notifications are generated by Audit Reporter objects. Both operation source and operation target devices which generate audit notifications contain Audit Reporter objects.

In the most basic case, there will be a single Audit Reporter object responsible for all audit notifications. In complex scenarios, a device may have multiple Audit Reporter objects each generating audit notifications for a different set of objects to allow for a finer granularity of control over audit notifications.

When a device supports audit reporting and is a BACnet client, exactly one Audit Reporter object shall have the Audit_Source_Reporter property set to TRUE and is used for configuration of operation source audit reporting. An Audit Reporter object with this flag set shall not be deletable. If creation of Audit Reporter objects is supported by the device, any created Audit Reporter objects shall have the Audit_Source_Reporter flag set to FALSE.

All write operations to an Audit Reporter object shall result in an audit notification by the containing device, if Audit_Level is not NONE.

Audit Reporter objects may optionally support delaying the sending of audit notifications so that multiple notifications can be sent in a single request. The Maximum_Send_Delay property limits how long audit notifications may be delayed.

19.Y.3 Audit Notification Configuration

Any device which supports auditing shall contain at least one Audit Reporter object.

Auditing can be configured on a per Audit Reporter level and on a per object level. Group enabling and disabling of audit reporting is controlled via the Audit_Level and Auditable_Operations properties of the Audit Reporter objects.

Audit_Level controls the level of audit reporting at the Audit Reporter or at the object level. When the Audit_Level property in the Audit Reporter object is NONE, the Audit Reporter object shall not generate any audit notifications, with the exception of generating a notification when its Audit_Level property is modified. When the Audit_Level property in an object (other than an Audit Reporter object) is NONE, the device shall not generate any audit notifications for the object, with the exception of generating a notification when the object’s Audit_Level is modified.

The Auditable_Operations property in an object controls which operations on the object an Audit Reporter object generates audit notifications for. If the property is not present, then the Auditable_Operations property in the associated Audit Reporter object is used instead. Irrespective of the value of an Auditable_Operations property, changes to an Auditable_Operations property will always result in an audit notification by the server device if Audit_Level is not NONE.

When a device is acting as an operation source, the Auditable_Operations property of the Audit Reporter object, with the Audit_Source_Reporter property set to TRUE, controls which operations initiated by the device will be reported. The Auditable_Operations property of the target object has no impact on the audit notifications generated by the operation source.

The Audit_Priority_Filter property specifies the priorities for which audit notifications will be generated for write operations. Any auditable write operation to the commandable property of an object with a priority associated with a bit set to 0 in this property shall not result in an audit notification. For example, if bit 7 of this property is set and bit 10 is not set, a manual override at priority 8 would result in an audit notification, whereas a WriteProperty request at priority 11 would not (the difference in the bit numbers versus the priorities in this example is due to bits being 0-based and priorities being 1-based). This property has no effect on the auditing of non-write operations nor on the auditing of operations on properties other than commandable properties. If this property is not present in an object with a commandable property, then no command priority based audit filtering will be applied. The purpose of this property is to allow normal control actions to occur without audit notifications being generated while ensuring that override actions are audited.

The Audit_Level property determines which properties in the object are auditable. Irrespective of the value of this property, changes to this property shall always result in an audit notification by the server device.

Table 19-Y1. Audit Level

Operation	Comment
NONE	No operations on the object will result in audit notifications except for the noted modifications to the audit configuration.

DEFAULT	The value for the audit level for this object is taken from the Audit_Level property of the associated Audit Reporter object.
AUDIT_CONFIG	Operations on configuration properties will result in audit notifications. Operations on the Present_Value will not result in audit notifications unless Present_Value has been designated as a configurable item for the product.
AUDIT_ALL	Operations on all properties will result in audit notifications.

The BACnetAuditLevel enumeration is extensible providing a method for vendors to specify new audit levels for objects that require them. No extensible audit level value shall be equivalent to NONE (result in no auditable operations on the object.)

For the audit level AUDIT_CONFIG, it is a local matter whether or not any given property in an object is considered a configuration property. This is a local matter specifically to allow implementations to make this designation based on the needs of the specific application of the product.

19.Y.4 Audit Notifications

The parameters sent in an audit notification are described in Table 19-Y2.

Table 19-Y2. Audit Notification Parameters

Parameter	Data Type	Value
Source Timestamp	BACnetTimeStamp	The time the operation was requested by the operation source. This field shall not be included in operation target notifications unless the source and target are the same device.
Target Timestamp	BACnetTimeStamp	The time the operation was performed by the operation target. This field shall not be included in operation source notifications unless the source and target are the same device.
Source Device	BACnetRecipient	The device requesting the operation. If known, the BACnetObjectIdentifier form shall be provided.
Source Object	BACnetObjectIdentifier	The object requesting the operation. This field shall not be included in operation target notifications, unless the source and target are the same device, or when there is no source object to report.
Operation	BACnetAuditOperation	The operation being requested.

Source Comment	CharacterString	A human readable description of the operation. This value may be system generated. For systems where human comments are required for documenting actions such as required by some pharmaceutical installations, the human entered comment shall be provided in this parameter in addition to any system generated comment. For GENERAL operation audit notifications, this field shall include a system generated description of the operation.
Target Comment	CharacterString	A human readable description of the operation. For GENERAL operation audit notifications, this field shall include a system generated description of the operation.
Invoke Id	Unsigned8	The invoke id from the service request that resulted in this audit record. This provides the ability to more easily match operation source and operation target audit records. This field shall not be included in operations consisting of unconfirmed service requests. This field shall not be included if the source and target are the same device.
Source User Id	Unsigned16	The user performing the operation. The value in this field depends on the user authentication model in use. See Clause 24.14 on user authentication for information on this value. This field shall not be included if no user identification is available.
Source User Role	Unsigned8	The role of the user performing the operation. The value in this field depends on the user authentication model in use. See Clause 24.14 on user authentication for information on this value. This field shall not be included if no user role information is available.
Target Device	BACnetRecipient	The target device of the operation. If known, the BACnetObjectIdentifier form shall be provided.
Target Object	BACnetObjectIdentifier	The target object of the operation. This field shall be absent if the target is not an object.
Target Property	BACnetPropertyReference	The target of the operation if it is targeted at a property. This field shall be absent if the target is not a property.
Target Priority	Unsigned(1..16)	For operations which include a priority (such as when a command occurs) this field is present and contains the provided priority. This field may be absent if the provided priority is 16.

Target Value	ABSTRACT-SYNTAX.&Type	For operations which include a value (such as when a write occurs) this field is present and contains the new value. When the value being written is larger than 32 encoded octets, it is a local matter whether the target value is included in the audit notification. This allowance is included so that large audit notifications can be avoided where the data value is large. Audit loggers may discard this parameter value if it exceeds 500 octets when not acting as a forwarder.
Current Value	ABSTRACT-SYNTAX.&Type	When an operation includes a value (such as when a write occurs) and the current value (before the operation) of the target property is known, this field is present and contains the current property value. When the current value is larger than 32 encoded octets, it is a local matter whether the current value is included in the audit notification. This allowance is included so that large audit notifications can be avoided where the data value is large. Audit loggers may discard this parameter value if it exceeds 500 octets when not acting as a forwarder.
Result	Error	When the operation fails, this field shall be present and indicates the error that occurred.

19.Y.5 Audit Operations

The 'Operation' field of an audit notification indicates the operation requested or performed. The mapping of standard BACnet services to the BACnetAuditOperation enumeration is shown in Table 19-Y3. When a service allows multiple object or property targets (e.g., ReadPropertyMultiple, WritePropertyMultiple, etc.), each target shall be reported in a separate notification.

Table 19-Y3. Service to Audit Operation Mapping

Audit Operation	BACnet Service	Comment
READ	ReadProperty ReadPropertyMultiple ReadRange AtomicReadFile AuditLogQuery	When reporting a file operation, no 'Target Property' value is provided. This setting should be used with caution as it can result in a large increase in network traffic.
WRITE	WriteProperty WritePropertyMultiple AtomicWriteFile WriteGroup AddListElement RemoveListElement	When reporting a file operation, no 'Target Property', 'Target Value', or 'Current Value' values are provided. These entries can be discarded if the write operation does not result in a change (the 'Target Value' equals the 'Current Value', and re-writing the property has no side-effects).

CREATE	CreateObject	Initial values are not provided in a CREATE operation audit record. Initial values may be optionally reported by reporting WRITE operations.
DELETE	DeleteObject	No 'Target Property', 'Target Value', or 'Current Value' values are provided.
LIFE_SAFETY	LifeSafetyOperation	The 'Target Value' shall include the 'Requesting Process Identifier' and 'Request' parameters in a BACnetLifeSafetyOperationInfo sequence. The 'Source Comment' shall be set to, or include, the 'Requesting Source' parameter value.
ACKNOWLEDGE_ALARM	AcknowledgeAlarm	The 'Target Value' shall include the 'Event State Acknowledged', and 'Timestamp' parameters in a BACnetAcknowledgeAlarmInfo sequence. The 'Source Comment' shall be set to, or include, the 'Acknowledgement Source' parameter value.
DEVICE_DISABLE_COMM	DeviceCommunicationControl	Used when a DISABLE or DISABLE_INITIATION DeviceCommunicationControl operation is performed. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_ENABLE_COMM	DeviceCommunicationControl	Used when an ENABLE DeviceCommunicationControl operation is performed, or when a DISABLE or DISABLE_INITIATION DeviceCommunicationControl operation's lifetime expires. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_RESET	ReinitializeDevice	WARMSTART and COLDSTART options of the ReinitializeDevice service. An audit record is generated when the device is reset for any reason. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_BACKUP	ReinitializeDevice	START_BACKUP, ABORT_BACKUP, or END_BACKUP options of the ReinitializeDevice service. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.
DEVICE_RESTORE	ReinitializeDevice	START_RESTORE, ABORT_RESTORE, END_RESTORE options of the ReinitializeDevice service. No 'Target Object', 'Target Property', 'Target Value', or 'Current Value' values are provided.

SUBSCRIPTION	SubscribeCOV SubscribeCOVProperty	The 'Target Value' parameter shall be of type BOOLEAN and indicates whether (TRUE) the operation is a subscription, or (FALSE) the operation is an unsubscription. Subscription timeouts shall be reported as unsubscriptions. No 'Current Value' parameter is provided.
NOTIFICATION	Event Notification COV Notification	Receipt of event notification and COV notification messages are treated as source auditable operations; whereas the generation of event notifications and COV notifications are treated as target auditable operations. This setting should be used with caution as it can result in a large increase in network traffic. The 'Target Property' shall be the Event_State property, and the 'Target Value' shall be the new Event_State.
AUDITING_FAILURE		Used to report that audit records were dropped. The 'Target Timestamp' field is used to report the time of earliest record that was dropped. The 'Source Device' and 'Target Device' fields are set to reference the local device. The 'Current Value' field is set to an Unsigned count of the number of records dropped.
NETWORK_CHANGES		Used to report changes to Network Port objects which are a result of something other than application layer BACnet services (such as BVLL messages or registration timeouts).
GENERAL		Any other actions that the device needs to report to the audit log. Given the non-standard nature of this audit record type, it is important that information be formatted into the 'Comment' parameter. Where a device reports a wide variety of other actions, or there is a large variation in the expected frequency of different types of actions, it is recommended that implementers provide a method of configuring which actions result in notifications when the GENERAL operation notifications are enabled.

19.Y.6 High Volume Situations

Under some high traffic conditions, a device might be unable to generate audit notifications for all auditable actions due to local resource constraints. When such a condition exists, the device shall selectively drop audit records attempting to keep the most important records for distribution to the audit logger. The device shall report to the audit logger the number of dropped records using an audit notification with an operation of AUDITING_FAILURE.

Except for notifications mandated by this standard for changes in audit settings, the relative priority of audit notifications is a local matter, it is suggested that when dropping notifications, non-modification audit notifications should be dropped first (e.g., audit operation is READ, NOTIFICATION, etc.). Audit notifications that are mandated by this standard for changes in audit settings shall be deemed to be the highest priority records and the newest of these shall be the last records to be dropped.

19.Y.7 Audit Logs

19.Y.7.1 Audit Logger

The logging device is the central repository for audit logs. Since it is expected that audit logs will be very large, such a device is expected to have a large amount of storage dedicated to audit logging.

The general structure is that one or more Audit Log objects exist in the audit logger into which received audit notifications are placed. It is a local matter how many and which Audit Log objects any particular audit notification is placed into with the exception that, for any specific object, there is at least one Audit Log object which contains the complete audit history for the object.

Audit loggers differ from other loggers in that archiving of audit log records is accomplished by operation sources and operation targets pushing notifications to the logger and BUFFER_READY is not used to inform archival loggers of new notification records. Also, given that auditing should not be interrupted, there is no support for the Stop_When_Full function.

19.Y.7.2 Audit Forwarder

An audit forwarder is a specialized audit logger that is used to reduce network traffic caused by audit logging.

The audit forwarder collects audit notifications from multiple requests into an Audit Log object and then forwards them to a parent audit logger. Once forwarded successfully, audit notifications may be removed from the forwarder’s Audit Log object.

Audit forwarders are not required to support the audit querying functionality and as such are not considered a source of audit information for audit reporting. The audit forwarder is simply used for reducing the impact of audit reporting on network usage.

Audit forwarders shall not discard, or change, parameter values in audit notifications from other devices when forwarding the notifications.

While audit forwarder functionality can be placed in any device in the system, it is recommended that it be placed in routers as this minimizes the impact on network traffic.

19.Y.7.3 Audit Logger Hierarchies

When the operation source and operation target devices involved in an auditable operation report audit notifications to different audit loggers, there will be two distinct records of the operation in two different audit loggers. If there is to be a consolidated log, then a hierarchy of audit loggers shall be setup. A complete log with all audit record pairs will only be available in the topmost audit logger in the hierarchy. The hierarchy is achieved through the Member_Of property of the Audit Log object.

19.Y.8 Security of Audit Notifications

The BACnet audit logging framework does not provide any security. Security of the network shall be achieved through other means such as the security framework described in Clause 24.

Given that values are stored in Audit Logs, care should be taken in configuring access to Audit Log objects so that information that needs to be restricted can be.

[Insert into **Clause 3.2**, p. 2]

persistent: when used to describe property values, the property value is stored in non-volatile storage so as to survive device resets and power failures.

[Modify **Table 12-13**, p. 211]

Audit_Notification_Recipient	*BACnetRecipient*	*O*[22]

...

...

[22] If present, this property shall be writable.

...

[x] *This property shall be present if, and only if, the device supports audit reporting.*

[Add new **Clause 12.11.X1**, p. 221]

12.11.X1 Audit_Notification_Recipient

This property, of type BACnetRecipient, specifies the audit logging device to which audit notifications are sent.

When this property is changed, an audit notification shall be generated and it shall be either globally broadcast, or sent to both the original audit notification recipient and the new audit notification recipient.

If this property is present, it shall be writable.

[Add new **Clause 12.X**, p. 584]

12.X Audit Reporter Object Type

The Audit Reporter object type defines a standardized object whose properties represent the externally visible audit settings for a BACnet device which reports auditable actions. BACnet devices which report auditable actions shall contain at least one Audit Reporter object. A detailed description of BACnet audit reporting can be found in Clause 19.Y.

Audit Reporter objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Audit Reporter objects that support intrinsic reporting shall apply the NONE event algorithm.

The Audit Reporter object and its properties are summarized in Table 12-X and described in detail in this clause.

Table 12-X. Properties of the Audit Reporter Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	R
Event_State	BACnetEventState	R
Audit_Level	BACnetAuditLevel	R
Audit_Source_Reporter	BOOLEAN	R
Auditable_Operations	BACnetAuditOperationFlags	R
Audit_Priority_Filter	BACnetPriorityFilter	R
Issue_Confirmed_Notifications	BOOLEAN	R
Monitored_Objects	BACnetARRAY[N] of BACnetObjectSelector	O
Maximum_Send_Delay	Unsigned	O[1,2]
Send_Now	BOOLEAN	O[1,2]
Event_Detection_Enable	BOOLEAN	O[3,5]
Notification_Class	Unsigned	O[3,5]
Event_Enable	BACnetEventTransitionBits	O[3,5]
Acked_ Transitions	BACnetEventTransitionBits	O[3,5]
Notify_Type	BACnetNotifyType	O[3,5]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,5]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4,5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[5]
Reliability_Evaluation_Inhibit	BOOLEAN	O
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] If present, these properties shall be writable.
[2] If the object supports delaying of audit notifications, then these properties shall be present.
[3] These properties are required if the object supports intrinsic reporting.
[4] This property, if present, is required to be read-only.
[5] These properties shall be present only if the object supports intrinsic reporting.

12.X.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.X.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.X.3 Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be AUDIT_REPORTER.

12.X.4 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.X.5 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Audit Reporter object. Three of the flags are associated with the values of other properties of this object. A more detailed status could be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical TRUE (1) if the Event_State property is present and does not have a value of NORMAL, otherwise logical FALSE (0).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	Logical FALSE (0).
OUT_OF_SERVICE	Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.X.6 Reliability

The Reliability property, of type BACnetReliability, provides an indication that the properties of the Audit Reporter object are in a consistent state and whether the object is operating properly.

If this Audit Reporter object is enabled and configured to monitor one or more objects that another enabled Audit Reporter object is also configured to monitor, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Reporter object is enabled and the device is unable to resolve the Audit_Notification_Recipient device, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Reporter object fails to successfully send audit notifications, or fails to receive acknowledgements when using ConfirmedAuditNotification requests, the Reliability property shall be set to COMMUNICATION_FAILURE. The existence of this reliability condition shall not stop the object from attempting to send audit notifications. It is a local matter whether or not the object will retry sending of audit notifications.

12.X.7 Event_State

The Event_State property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.X.8 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for all objects this Audit Reporter object reports for which do not have an Audit_Level property, or for which the Audit_Level property is set to DEFAULT.

This property shall not have the value DEFAULT.

If not configurable, this property shall not have the value NONE.

For details on auditing and the use of this property, see Clause 19.Y.2.

12.X.9 Audit_Source_Reporter

This read-only property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) this audit reporter object is the reporter object which controls operation source audit reporting.

If this property is TRUE, this object generates notifications for auditable operations initiated by this device.

For details on auditing and the use of this property, see Clause 19.Y.2.

12.X.10 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the object will report.

If not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

For details on auditing and the use of this property, see Clause 19.Y.3.

12.X.11 Audit_Priority_Filter

This property, of type BACnetPriorityFilter, specifies the auditable command priorities for write operations on the commandable property of the objects for which this object performs audit reporting. Bits which are set indicate the priorities for which audit notifications will be generated. The bit number is 1 less than the priority it is associated with (bit 7 is associated with priority 8).

For details on auditing and the use of this property, see Clause 19.Y.3.

12.X.12 Issue_Confirmed_Notifications

This property, of type BOOLEAN, shall convey whether confirmed (TRUE) or unconfirmed (FALSE) audit notifications shall be issued.

12.X.13 Monitored_Objects

This property, of type BACnetARRAY of BACnetObjectSelector, is used to indicate the objects that the Audit Reporter object will generate notifications for.

If the array is of length 0, or all entries in the array are NULL entries, then the Audit Reporter object does not generate notifications for any objects in the device.

Entries of type BACnetObjectIdentifier indicate specific objects that are reportable by the Audit Reporter object. Entries of type BACnetObjectType indicate that all objects of the specified type are reportable by the Audit Reporter object. Entries of type NULL are ignored.

If there are multiple Audit Reporter objects which are configured to report for a specific object, only the one with the lowest object instance shall generate notifications for the object. This allows a generic catch-all Audit Reporter object to be created with a high instance number and lower instance number Audit Reporter objects which provide customized audit reporting for specific objects.

12.X.14 Maximum_Send_Delay

This property, of type Unsigned, specifies the maximum amount of time, in seconds, that the Audit Reporter object will delay sending audit notifications for the purpose of batch sending notifications.

The value in this property does not mandate that notifications shall be delayed, but rather allows the object to delay sending of notifications by up to the specified number of seconds.

This property shall support all values in the range 0 to 3600.

12.X.15 Send_Now

This property, of type BOOLEAN, when written to TRUE, forces the Audit Reporter object to send any and all audit notifications that are being delayed. After all delayed audit notifications are sent, the Audit Reporter object resets the property to FALSE.

It is a local matter whether or not the Audit Reporter object stops attempting to immediately send its delayed notifications if the value is written to FALSE.

12.X.16 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.X.17 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.X.18 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.X.19 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.X.20 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.X.21 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type Time or Date shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.X.22 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.X.23 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.X.24 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.X.25 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.X.26 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tag and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.X.27 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.X.28 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

[Add new **Clause 12.Y**, p. 584]

12.Y Audit Log Object Type

An Audit Log object combines audit notifications from operation sources and operation targets and stores the combined record in an internal buffer for subsequent retrieval. Each timestamped buffer entry is called an audit log "record."

Each Audit Log object maintains an internal, persistent, optionally fixed-size log buffer. This log buffer fills or grows as audit log records are added. If the log buffer becomes full, the least recent log records are overwritten when new log records are added. Log buffers are transferred as a list of BACnetAuditLogRecord values using the ReadRange and AuditLogQuery services. Each log record in the log buffer has an implied sequence number that is equal to the value of the Total_Record_Count property immediately after the record is added. See Clause 19.Y for a full description of how audit notifications are added to audit logs.

As records are added into the log, the Audit Log object will scan existing entries for a matching record. A record is a match if:

1) the record contains the timestamp for the opposite actor (the record contains the operation source timestamp when merging in an operation target notification and vice-versa);
2) the operation-source, operation, invoke-id, target-device, target-property, are all equal;
3) if the user-id, user-role, target-value fields are provided in both notifications then they are equal; and
4) if the source-timestamp and target-timestamp values are approximately equal (+/- APDU_Timeout * 2).

If a match is found, the existing log record is updated. Otherwise, a new record is created. If a match is found, and it already contains both an operation source and an operation target portion, then the notification is dropped. When creating a new record, those fields which are not supplied in the notification (such as the 'Source Timestamp' when a server notification is received) shall be absent from the record. When updating an existing record, those fields not supplied in the original notification are updated from the new notification, if present. For the 'Current Value' field, a value provided by the operation target device shall always take precedence over a value provided by an operation source device. As such, if the values provided in the peer notifications differ, the operation target value shall be the one used in the record.

Logging may be enabled and disabled through the Enable property. Audit Log enabling and disabling is recorded in the audit log buffer.

Unlike other log objects, Audit Log objects do not use the BUFFER_READY event algorithm.

The acquisition of log records by remote devices has no effect upon the state of the Audit Log object itself. This allows completely independent, but properly sequential, access to its log records by all remote devices. Any remote device can independently update its log records at any time.

Audit Log objects may optionally support forwarding of audit notifications to “parent” audit logs. This functionality improves the reliability of the audit system by allowing intermediaries to buffer audit notifications in the case where the ultimate audit logger is offline for a short period of time. It is expected that intermediaries be capable of storing a larger number of records than devices which report auditable actions. It is also useful for buffering of audit notifications so they can be sent in bulk to the parent audit log. When operating in this mode, with the Delete_On_Forward property set to TRUE, the object is not required to perform audit notification matching and combining.

Audit Log objects may optionally support intrinsic reporting to facilitate the reporting of fault conditions. Audit Log objects that support intrinsic reporting shall apply the NONE event algorithm.

The Audit Log object and its properties are summarized in Table 12-Y and described in detail in this subclause.

Table 12-Y. Properties of the Audit Log Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Property_List	BACnetARRAY[N] of BACnetPropertyIdentifier	R

Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Enable	BOOLEAN	W
Buffer_Size	Unsigned32	R
Log_Buffer	BACnetLIST of BACnetAuditLogRecord	R
Record_Count	Unsigned64	R
Total_Record_Count	Unsigned64	R
Member_Of	BACnetDeviceObjectReference	O[1]
Delete_On_Forward	BOOLEAN	O[1,2]
Issue_Confirmed_Notifications	BOOLEAN	O[1]
Event_Detection_Enable	BOOLEAN	O[3,4]
Notification_Class	Unsigned	O[3,4]
Event_Enable	BACnetEventTransitionBits	O[3,4]
Acked_Transitions	BACnetEventTransitionBits	O[3,4]
Notify_Type	BACnetNotifyType	O[3,4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[3,4]
Event_Message_Texts	BACnetARRAY[3] of CharacterString	O[4,5]
Event_Message_Texts_Config	BACnetARRAY[3] of CharacterString	O[4]
Event_Detection_Enable	BOOLEAN	O[3,4]
Reliability_Evaluation_Inhibit	BOOLEAN	O[6]
Audit_Level	BACnetAuditLevel	O[7]
Auditable_Operations	BACnetAuditOperationFlags	O[7]
Tags	BACnetARRAY[N] of BACnetNameValue	O
Profile_Location	CharacterString	O
Profile_Name	CharacterString	O

[1] These properties shall be present if the object supports forwarding.
[2] This property shall be read-only and TRUE if the object does not support matching and combining of records.
[3] These properties are required if the object supports intrinsic reporting.
[4] These properties shall be present only if the object supports intrinsic reporting.
[5] This property, if present, is required to be read-only.
[6] If this property is present, then the Reliability property shall be present.
[7] This property shall be present only if the device supports audit reporting.

12.Y.1 Object_Identifier

This property, of type BACnetObjectIdentifier, is a numeric code that is used to identify the object. It shall be unique within the BACnet device that maintains it.

12.Y.2 Object_Name

This property, of type CharacterString, shall represent a name for the object that is unique within the BACnet device that maintains it. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.

12.Y.3 Object_Type

This property, of type BACnetObjectType, indicates membership in a particular object type class. The value of this property shall be AUDIT_LOG.

12.Y.4 Property_List

This read-only property is a BACnetARRAY of property identifiers, one property identifier for each property that exists within the object. The Object_Name, Object_Type, Object_Identifier, and Property_List properties are not included in the list.

12.Y.5 Description

This property, of type CharacterString, is a string of printable characters whose content is not restricted.

12.Y.6 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of an Audit Log object. The IN_ALARM and FAULT flags are associated with the values of other properties of this object. A more detailed status may be determined by reading the properties that are linked to these flags. The relationship between individual flags is not defined by the protocol. The four flags are

{IN_ALARM, FAULT, OVERRIDDEN, OUT_OF_SERVICE}

where:

IN_ALARM	Logical FALSE (0) if the Event_State property has a value of NORMAL, otherwise logical TRUE (1).
FAULT	Logical TRUE (1) if the Reliability property is present and does not have a value of NO_FAULT_DETECTED, otherwise logical FALSE (0).
OVERRIDDEN	The value of this flag shall be Logical FALSE (0).
OUT_OF_SERVICE	The value of this flag shall be Logical FALSE (0).

If the object supports event reporting, then this property shall be the pStatusFlags parameter for the object's event algorithm. See Clause 13.3 for event algorithm parameter descriptions.

12.Y.7 Event_State

The Event_State property, of type BACnetEventState, is included in order to provide a way to determine whether this object has an active event state associated with it (see Clause 13.2.2.1). If the object supports event reporting, then the Event_State property shall indicate the event state of the object. If the object does not support event reporting, then the value of this property shall be NORMAL.

12.Y.8 Reliability

The Reliability property, of type BACnetReliability, provides an indication that the properties of the Audit Log object are in a consistent state and whether the object is operating properly.

If the Audit Log object is enabled and configured to forward audit notifications, and the device is unable to resolve the Member_Of device, then this property shall have the value CONFIGURATION_ERROR.

If the Audit Log object, when configured to forward audit notifications, fails to successfully send audit notifications, or fails to receive acknowledgements when using ConfirmedAuditNotification requests, the Reliability property shall be set to COMMUNICATION_FAILURE. The existence of this reliability condition shall not stop the object from attempting to send audit notifications. It is a local matter whether or not the object will retry sending of audit notifications.

12.Y.9 Enable

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging is enabled. Logging occurs if and only if Enable is TRUE. Log_Buffer records of type log-status are recorded without regard to the value of the Enable property.

12.Y.10 Buffer_Size

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

12.Y.11 Log_Buffer

This property, of type BACnetLIST of BACnetAuditLogRecord, is a list of timestamped log records of datatype BACnetAuditLogRecord, each of which conveys the audit notification parameters or status changes in the Audit Log object. Each log record has data fields as follows:

Timestamp	The local date and time that the entry was placed into the audit log.
Log Datum	The audit notification information, or a change in status or operation of the Audit Log object itself.

The choices available for the 'Log Datum' are listed below:

log-status — This choice represents a change in the status or operation of the Audit Log object. Whenever one of the events represented by the flags listed below occurs, a log record shall be appended to the buffer.

LOG_DISABLED	This flag is changed whenever collection of log records by the Audit Log object is enabled or disabled. It shall be TRUE if Enable is FALSE; otherwise it shall be FALSE.
BUFFER_PURGED	This flag shall be set to TRUE whenever the log buffer is cleared. After this value is recorded in the log buffer, the subsequent immediate change to FALSE shall not be recorded. A log record indicating the purging of the log buffer shall be placed into the log buffer even if logging is disabled, the local time is outside of the time range defined by the Start_Time and Stop_Time properties, or the log buffer was completely empty before the request for clearing.
LOG_INTERRUPTED	This flag indicates that the collection of log records by the Audit Log object was interrupted by a power failure, device reset, object reconfiguration, or other such disruption, such that samples prior to this log record might have been missed.

audit-notification — This choice represents an audit notification that was received. It consists of the audit information from the operation source and operation target generated ConfirmedAuditNotification and/or UnconfirmedAuditNotification requests. If the audit notification was generated locally, this choice shall hold what would be received if the Audit Log object existed on a remote device.

time-change — This choice represents a change in the clock setting in the device, records the number of seconds and fraction of a second by which the clock changed. If, and only if, the number is not known, such as when the clock is initialized for the first time, the value recorded shall be 0.0. This log record shall be recorded after changing the local time of the device, and the timestamp shall reflect the new local time of the device.

Also associated with each log record is an implied sequence number, the value of which is equal to Total_Record_Count at the point where the log record has been added into the log buffer and Total_Record_Count has been adjusted accordingly. All clients shall be able to correctly handle the case where the Audit Log object is reset such that its Total_Record_Count is returned to zero and also the case where Total_Record_Count has wrapped back to one. While no interoperable method is provided for resetting the Audit Log object, catastrophic failures can result in replacement of the logging device which will be equivalent to resetting of the object. It is for situations such as this that clients need to be prepared.

The log buffer is not network accessible except through the use of the ReadRange and AuditLogQuery services in order to avoid problems with record sequencing when segmentation is required.

12.Y.12 Record_Count

This read-only property, of type Unsigned64, shall represent the number of log records currently resident in the log buffer. Unlike other log object types, this property is not writable as audit logs are never expected to be reset.

12.Y.13 Total_Record_Count

This property, of type Unsigned64, shall represent the total number of log records collected by the Audit Log object since creation. When the value of Total_Record_Count reaches its maximum possible value of 2^{64} - 1, the next value it takes shall be one. Once this value has wrapped to one, its semantic value (the total number of log records collected) has been lost.

12.Y.14 Member_Of

This property, of type BACnetDeviceObjectReference, shall indicate the device which this Audit Log object forwards all audit log records to.

All log records collected by the Audit Log shall be sent to the referenced device via audit notification requests.

12.Y.15 Delete_On_Forward

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) the notifications will be removed from the log buffer after being successfully forwarded.

12.Y.16 Issue_Confirmed_Notifications

This property, of type BOOLEAN, shall convey whether confirmed (TRUE) or unconfirmed (FALSE) audit notifications shall be issued when forwarding audit notifications.

12.Y.17 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.Y.18 Notification_Class

This property, of type Unsigned, shall specify the instance of the Notification Class object to use for event-notification-distribution.

12.Y.19 Event_Enable

This property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable the distribution of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL notifications (see Clause 13.2.5). A device is allowed to restrict the set of supported values for this property but shall support (T, T, T) at a minimum.

12.Y.20 Acked_Transitions

This read-only property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the acknowledgment state for TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1.5). Each flag shall have the value TRUE if no event of that type has ever occurred for the object.

12.Y.21 Notify_Type

This property, of type BACnetNotifyType, shall convey whether the notifications generated by the object should be Events or Alarms. The value of the property is used as the value of the 'Notify Type' service parameter in event notifications generated by the object.

12.Y.22 Event_Time_Stamps

This read-only property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). Timestamps of type Time or Date shall have X'FF' in each octet, and Sequence Number timestamps shall have the value 0 if no event of that type has ever occurred for the object.

12.Y.23 Event_Message_Texts

This read-only property, of type BACnetARRAY[3] of CharacterString, shall convey the message text values of the last TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events (see Clause 13.2.2.1). If a particular type of event has yet to occur, an empty string shall be stored in the respective array element.

12.Y.24 Event_Message_Texts_Config

This property, of type BACnetARRAY[3] of CharacterString, contains the character strings which are the basis for the 'Message Text' parameter for the event notifications of TO_OFFNORMAL, TO_FAULT, and TO_NORMAL events, respectively, generated by this object. The character strings may optionally contain proprietary text substitution codes to incorporate dynamic information such as date and time or other information.

12.Y.25 Event_Detection_Enable

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) intrinsic reporting is enabled in the object and controls whether (TRUE) or not (FALSE) the object will be considered by event summarization services.

This property is expected to be set during system configuration and is not expected to change dynamically.

When this property is FALSE, Event_State shall be NORMAL, and the properties Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts shall be equal to their respective initial conditions.

12.Y.26 Reliability_Evaluation_Inhibit

This property, of type BOOLEAN, indicates whether (TRUE) or not (FALSE) reliability-evaluation is disabled in the object. This property is a runtime override that allows temporary disabling of reliability-evaluation.

When reliability-evaluation is disabled, the Reliability property shall have the value NO_FAULT_DETECTED unless Out_Of_Service is TRUE and an alternate value has been written to the Reliability property.

12.Y.27 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object’s associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.Y.3.

If this property is present and not configurable, the value shall not be NONE.

12.Y.28 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.Y.3.

12.Y.29 Tags

This property, of type BACnetARRAY of BACnetNameValue, is a collection of tags for the object. See Clause Y.1.4 for restrictions on the string values used for the names of these tag and for a description of tagging and the mechanism by which tags are defined.

Each entry in the array is a BACnetNameValue construct which consists of the tag name and an optional value. If the tag is defined to be a "semantic tag" then it has no value, and the "value" field of the BACnetNameValue shall be absent.

While some tags may be known in advance when a device is manufactured, it is recommended that implementations consider that this kind of information might not be known until a device is deployed and to provide a means of configuration or writability of this property.

12.Y.30 Profile_Location

This property, of type CharacterString, is the URI of the location of an xdd file (See Clause X.2) containing the definition of the CSML type specified by the Profile_Name property and possible other information (See Annex X). The URI is restricted to using only the "http", "https", and "bacnet" URI schemes. See Clause Q.8 for the definition of the "bacnet" URI scheme.

If a Profile_Location value is not provided for a particular object, then the client shall use the Profile_Location of the Device object, if provided, to find the definition of the Profile_Name.

12.Y.31 Profile_Name

This property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name shall begin with a vendor identifier code (see Clause 23) in base-10 integer format, followed by a dash. All subsequent characters are administered by the organization registered with that vendor identifier code. The vendor identifier code that prefixes the profile name shall indicate the organization that publishes and maintains the profile. This vendor identifier need not have any relationship to the vendor identifier of the device within which the object resides.

A profile defines a set of additional properties, behavior, and/or requirements for this object beyond those specified here. This standard defines only the format of the names of profiles. If the Profile_Location property of this object or the Device object is present and nonempty, then the value of this property shall be the name of a CSML type defined in an xdd file referred to by the Profile_Location property.

[Add to all **Object Type Tables** in **Clause 12**, p 153]
[Note: These properties are added to the Audit Reporter and the Audit Log object types as of this addendum. See new Clauses 12.X. and 12.Y. In the Audit Reporter object type, these properties have specific requirements. In the Audit Log object type, the properties are included as shown below.]

Property Identifier	Property Datatype	Conformance Code
...	...	...
Audit_Level	*BACnetAuditLevel*	*O*[x]
Auditable_Operations	*BACnetAuditOperationFlags*	*O*[x]
...	...	...

[x] *This property shall be present only if the device supports audit reporting.*

[Add to all **Commandable Object Type Tables** in **Clause 12**, p 153]

Property Identifier	Property Datatype	Conformance Code
...	...	...
Audit_Priority_Filter	*BACnetOptionalPriorityFilter*	*O*[x]
...	...	...

[x] *This property shall be present only if the device supports audit reporting.*

[Add to all **Object Types** in **Clause 12**, p. 153]
[Note: These properties are added to the Audit Reporter and the Audit Log object types as of this addendum. See new Clauses 12.X. and 12.Y. In the Audit Reporter object type, these properties have specific requirements. In the Audit Log object type, the properties are included as shown below.]

12.Z.X1 Audit_Level

This property, of type BACnetAuditLevel, specifies the level of auditing to perform for the specific object. If this property has the value DEFAULT, or if this property is not present in the object, then the audit level value for the object shall be taken from the Audit_Level property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.Y.3.

If this property is present and not configurable, the value shall not be NONE.

12.Z.X2 Auditable_Operations

This property, of type BACnetAuditOperationFlags, specifies the operations that the device will report for this object.

If present and not configurable, the values of the READ, NOTIFICATION, and SUBSCRIPTION bits shall be 0, and the WRITE, CREATE, DELETE, and ACKNOWLEDGE_ALARM bits shall be 1.

If this property is not present, and the device supports auditing, then the Auditable_Operations property of the object's associated Audit Reporter object shall control the operations that are auditable for this object.

For details on auditing and the use of this property, see Clause 19.Y.3.

[Add to all **Commandable Object Types** in **Clause 12**, p. 153]

12.Z.X3 Audit_Priority_Filter

This property, of type BACnetOptionalPriorityFilter, specifies the auditable priorities for write operations on the commandable property of the object. Bits which are set (1) indicate the priorities for which audit notifications will be generated. The bit number is one less than the priority it is associated with (e.g., bit 7 is associated with priority 8).

If present and not configurable, this property shall have the value NULL.

If this property is present and has the value NULL or if the property is not present, the object shall use the priority filter setting from the Audit_Priority_Filter property of the object's associated Audit Reporter object.

For details on auditing and the use of this property, see Clause 19.Y.3.

[Insert new **Clause 13.X**, p. 671]

13.X AuditLogQuery

The AuditLogQuery service provides efficient access to Audit Log records. The service allows for a few common queries on Audit Logs.

The service allows devices to find and retrieve audit information without reading and processing a complete audit log.

The queries currently facilitated by the AuditLogQuery service are:

> **Audit Query By Target** - Retrieve the audit records for the specified target device, object, and property. Records are returned in reverse sequence number order (newest to oldest).
>
> **Audit Query By S**ource - Retrieve all audited actions initiated by the specified operation source. Records are returned in the reverse sequence number order (newest to oldest).

13.X.1 AuditLogQuery Service Structure

The structure of the AuditLogQuery service primitive is shown in Table 13-X. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-X1. Structure of AuditLogQuery Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Audit Log	M	M(=)		
Query Parameters	M	M(=)		
Start At Sequence Number	U	U(=)		
Requested Count	M	M(=)		
Result(+)			S	S(=)
Audit Log			M	M(=)
Records			M	M(=)
No More Items			M	M(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.X.1.1 Argument

This parameter shall convey the parameters for the AuditLogQuery confirmed service request.

13.X.1.1.1 Audit Log

This parameter, of type BACnetObjectIdentifier, specifies the Audit Log object to query.

13.X.1.1.2 Query Parameters

This parameter, of type BACnetAuditLogQueryParameters, specifies the query parameters to apply. The type of parameters varies by the type of query as shown below.

13.X.1.1.2.1 Audit Query By Target

Table 13-X2. Structure of 'Audit Query By Property' Query Parameters

Parameter Name	Req	Ind	Datatype
Target Device Identifier	M	M(=)	BACnetObjectIdentifier
Target Device Address	U	U(=)	BACnetAddress
Target Object Identifier	U	U(=)	BACnetObjectIdentifier
Target Property Identifier	U	U(=)	BACnetPropertyIdentifier
Target Array Index	U	U(=)	Unsigned
Target Priority	U	U(=)	Unsigned (1..16)
Operations	U	U(=)	BACnetAuditOperationFlags
Result Filter	M	M(=)	BACnetSuccessFilter

The 'Target Device Identifier', 'Target Device Address', 'Target Object Identifier', 'Target Property Identifier', 'Target Array Index' identify the device, object, or property for which audit records are requested.

When both 'Target Device Identifier' and 'Target Device Address' parameters are included, a record is a match if it matches either of the two parameters.

The 'Target Priority' parameter is used to restrict records to only those for the specified priority. Audit records without a 'Priority' will match any 'Target Priority'.

The 'Operations' parameter determines the operation types for which entries are requested. If not present, all operation types will be returned.

The 'Result Filter' parameter indicates whether audit records for successful and/or failed actions will be included in the response.

13.X.1.1.2.2 Audit Query By Source

Table 13-X3. Structure of 'Audit Query By Source' Query Parameters

Parameter Name	Req	Ind	Datatype
Source Device Identifier	M	M(=)	BACnetObjectIdentifier
Source Device Address	U	U(=)	BACnetAddress
Source Object Identifier	U	U(=)	BACnetObjectIdentifier
Operations	U	U(=)	BACnetAuditOperationFlags
Result Filter	M	M(=)	BACnetSuccessFilter

The 'Source Device Identifier' and 'Source Object Identifier' identify the operation source for which audit records are requested.

When both the 'Source Device Identifier' and 'Source Device Address' parameters are included, a record is a match if it matches either of the two parameters.

The 'Operations' parameter determines the operation types for which entries are requested. If not present, all operation types will be returned.

The 'Result Filter' parameter indicates whether audit records for successful and/or failed actions will be included in the response.

13.X.1.1.3 Start At Sequence Number

This optional parameter, of type Unsigned64, identifies, by sequence number, where in the log to start the query. Only those records which match the query parameters and which have a sequence number less than the value of this parameter are to be returned. If this parameter is not present, the query will start with the latest record in the log.

13.X.1.1.4 Requested Count

This parameter, of type Unsigned16, is used to limit the number of records that will be returned. The responding device shall not return more records than specified by this parameter.

13.X.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded. A successful result includes the following parameters.

13.X.1.2.1 Audit Log

This parameter, of type BACnetObjectIdentifier, specifies the Audit Log object the query was performed on.

13.X.1.2.2 Records

This parameter, of type list of BACnetAuditLogRecordResult, contains the returned records. Each entry is of the form:

Table 13-X4. Structure of an Entry of the 'Records' Parameter

Parameter Name	Rsp	Cnf	Parameter Type
Sequence Number	M	M(=)	Unsigned64
Record	M	M(=)	BACnetAuditLogRecord

13.X.1.2.3 No More Items

This parameter, of type BOOLEAN, specifies whether the query processed to the end of the Audit Log and found no more records (TRUE), or not (FALSE). When FALSE, there may, or may not, be more items in the log which match the query.

13.X.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

13.X.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The specified Audit Log object does not exist.	OBJECT	UNKNOWN_OBJECT
The device is not configured to perform audit logging.	SERVICES	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED

Note that there shall be no validation of object/property specification filter parameters. For example, 'Target Array Index' shall not be validated to ensure it is consistent with the 'Target Object Identifier' and 'Target Property Identifier'. The responding device shall just perform the comparison between the audit log records and the filter parameters as described below.

13.X.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall attempt to query the specified Audit Log object and construct a list of records that match the query.

Devices that support execution of this service are required to support all query types.

Records are returned latest first. The responding device shall return up to 'Requested Count' records. If the responding device reaches the end of the buffer and all records which match the query are returned, then the 'No More Items' parameter shall be set to TRUE; otherwise it shall be set to FALSE.

If there are no records in the Audit Log that match the query criteria, then a Result(+) shall be returned with no records included and the 'No More Items' parameter set to TRUE.

13.X.2.1 Audit Query By Target

This query is used to get a history of audit records for objects in a specific device. It selects the most recent records with a sequence number less then 'Start At Sequence Number', and have 'Target Device', 'Target Object', 'Target Property', and 'Priority' fields which match the 'Target Device Identifier' or 'Target Device Address', 'Target Object Identifier', 'Target Property Identifier', 'Target Array Index', and 'Target Priority' query parameters.

Some of the query filter parameters ('Target Object Identifier', 'Target Property Identifier', 'Target Array Index', 'Target Priority') are optional. When one of these parameters is not provided, all values of the corresponding field in the audit log record will be considered a match.

13.X.2.2 Audit Query By Source

This query is used to get a history of audit records of actions initiated by a specific operation source. It selects the most recent records with a sequence number less then 'Start At Sequence Number', and have 'Source Device', and 'Source Object' fields which match the 'Source Device Identifier' and 'Source Object Identifier' query parameters.

The 'Source Object Identifier' and 'Operations' query filter parameter is optional. When one of these parameters is not provided, all values of the corresponding field in the audit log record will be considered a match.

[Insert new **Clause 13.Y**, p. 671]

13.Y ConfirmedAuditNotification

The ConfirmedAuditNotification service is used by an operation source, operation target, or audit forwarder to provide audit notifications to an audit logger. See Clause 19.Y.

13.Y.1 ConfirmedAuditNotification Service Structure

The structure of the ConfirmedAuditNotification service primitive is shown in Table 13-Y1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-Y1. Structure of ConfirmedAuditNotification Service Primitives

Parameter Name	Req	Ind	Rsp	Cnf
Argument	M	M(=)		
Notifications	M	M(=)		
Result(+)			S	S(=)
Result(-)			S	S(=)
Error Type			M	M(=)

13.Y.1.1 Argument

This parameter shall convey the parameters for the ConfirmedAuditNotification confirmed service request.

13.Y.1.1.1 Notifications

This parameter specifies one or more audit notifications of type BACnetAuditNotification. For the content of audit notifications, see Clause 19.Y.4.

13.Y.1.2 Result(+)

The 'Result(+)' parameter shall indicate that the service request succeeded.

13.Y.1.3 Result(-)

The 'Result(-)' parameter shall indicate that the service request failed. The reason for failure is specified by the 'Error Type' parameter.

13.Y.1.3.1 Error Type

This parameter consists of two component parameters: (1) an 'Error Class' and (2) an 'Error Code'. See Clause 18.

The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The device is not configured as an audit logger.	SERVICE	SERVICE_REQUEST_DENIED

13.Y.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall add the audit notifications to the local Audit Log object.

[Insert new **Clause 13.Z**, p. 671]

13.Z UnconfirmedAuditNotification

The UnconfirmedAuditNotification service is used by an operation source, operation target, or audit forwarder to provide audit notifications to an audit logger. See Clause 19.Y.

13.Z.1 UnconfirmedAuditNotification Service Structure

The structure of the UnconfirmedAuditNotification service primitive is shown in Table 13-Z1. The terminology and symbology used in this table are explained in Clause 5.6.

Table 13-Z1. Structure of UnconfirmedAuditNotification Service Primitives

Parameter Name	Req	Ind
Argument	M	M(=)
Notifications	M	M(=)

13.Z.1.1 Argument

This parameter shall convey the parameters for the UnconfirmedAuditNotification confirmed service request.

13.Z.1.1.1 Notifications

This parameter specifies one or more audit notifications of type BACnetAuditNotification. For the content of audit notifications, see Clause 19.Y.4.

13.Z.2 Service Procedure

After verifying the validity of the request, the responding BACnet-user shall add the audit notifications to the local Audit Log object.

[Insert into production **BACnetPropertyIdentifier** in **Clause 21**, preserving the alphabetical and numerical order, p. 845]

```
        ...
        audit-source-reporter                (497),
        audit-level                          (498),
        audit-notification-recipient         (499),
        audit-priority-filter                (500),
        auditable-operations                 (501),
        delete-on-forward                    (502),
        maximum-send-delay                   (503),
        monitored-objects                    (504),
        send-now                             (505),
        ...
-- numerical order reference
        ...
        -- audit-source-reporter             (497),
        -- audit-level                       (498),
        -- audit-notification-recipient      (499),
        -- audit-priority-filter             (500),
        -- auditable-operations              (501),
        -- delete-on-forward                 (502),
        -- maximum-send-delay                (503),
        -- monitored-objects                 (504),
        -- send-now                          (505),
        ...
        }
-- The special property identifiers ...
```

[Change production **BACnetPropertyIdentifier** in **Clause 21**, p. 845]

        ...
        issue-confirmed-notifications   (51),
        ...
        -- ~~formerly:~~ *see* issue-confirmed-notifications (51),   removed in version 1 revision 4, *added back*
        --   *in version 1 revision 20*
        ...

[Insert into production **BACnetObjectType** in **Clause 21**, preserving the alphabetical and numerical order, p. 840]

```
        audit-log                            (61),
        audit-reporter                       (62),
        ...
        -- see audit-log                     (61),
        -- see audit-reporter                (62),
```

[Insert into production **BACnetObjectTypesSupported** in **Clause 21**, p. 842]

```
        audit-log                            (61),
        audit-reporter                       (62),
```

[Insert into production **BACnetPropertyStates** in **Clause 21**, p. 862]

```
        audit-level                          [59] BACnetAuditLevel,
        audit-operation                      [60] BACnetAuditOperation,
```

[Change production **BACnetServicesSupported** in **Clause 21**, p. 866]

BACnetServicesSupported ::= BIT STRING {
-- Alarm and Event Services
    ...
    -- *confirmed-audit-notification*   *(44),*

...

-- Object Access Services
    ...
    -- *audit-log-query*   *(45),*

-- Unconfirmed Services
    ...
    -- *unconfirmed-audit-notification*   *(46),*
...

    unconfirmed-cov-notification-multiple   ~~(43)~~ *(43),* -- Alarm and Event Service

-- *Services added after 2016*
    confirmed-audit-notification   *(44),*   *-- Alarm and Event Service*
    audit-log-query   *(45),*   *-- Object Access Service*
    unconfirmed-audit-notification   *(46)*   *-- Alarm and Event Service*
    }

[Insert new productions into **Clause 21**, section 'Base Types', preserving the alphabetical order, p. 805]

BACnetAcknowledgeAlarmInfo ::= SEQUENCE {
    event-state-acknowledged   [0] BACnetEventState,
    timestamp   [1] BACnetTimeStamp
    }

BACnetAuditLevel ::= ENUMERATED {
    none   (0),
    audit-all   (1),
    audit-config   (2),
    default   (3),
    ...
    }
-- Enumerated values 0-127 are reserved for definition by ASHRAE. Enumerated values 128-255 may be used
-- by others subject to the procedures and constraints described in Clause 23.

BACnetAuditOperation ::= ENUMERATED {
    read   (0),
    write   (1),
    create   (2),
    delete   (3),
    life-safety   (4),
    acknowledge-alarm   (5),
    device-disable-comm   (6),
    device-enable-comm   (7),
    device-reset   (8),
    device-backup   (9),
    device-restore   (10),
    subscription   (11),
    notification   (12),
    auditing-failure   (13),
    network-changes   (14),
    general   (15),

```
        ...
        }
-- Enumerated values 0-31 are reserved for definition by ASHRAE. Enumerated values 32-63 may be used
-- by others subject to the procedures and constraints described in Clause 23.
-- The enumerated values match the bit positions in BACnetAuditOperationFlags.
```

BACnetAuditOperationFlags ::= BIT STRING {

```
        read                    (0),
        write                   (1),
        create                  (2),
        delete                  (3),
        life-safety             (4),
        acknowledge-alarm       (5),
        device-disable-comm     (6),
        device-enable-comm      (7),
        device-reset            (8),
        device-backup           (9),
        device-restore          (10),
        subscription            (11),
        notification            (12),
        auditing-failure        (13),
        network-changes         (14),
        general                 (15),
        ...
        }
-- Bit positions correspond one-to-one with the BACnetAuditOperation enumeration values.
-- Bits 0-31 are reserved for definition by ASHRAE. Bits 32-63 may be used by others subject to
-- the procedures and constraints described in Clause 23 for BACnetAuditOperation enumeration values.
```

BACnetLifeSafetyOperationInfo ::= SEQUENCE {

```
        requesting-process-identifier     [0] Unsigned32,
        request                           [1] BACnetLifeSafetyOperation
        }
```

BACnetPriorityFilter ::= BIT STRING {

```
        manual-life-safety                (0),
        automatic-life-safety             (1),
        priority-3                        (2),
        priority-4                        (3),
        critical-equipment-controls       (4),
        minimum-on-off                    (5),
        priority-7                        (6),
        manual-operator                   (7),
        priority-9                        (8),
        priority-10                       (9),
        priority-11                       (10),
        priority-12                       (11),
        priority-13                       (12),
        priority-14                       (13),
        priority-15                       (14),
        priority-16                       (15)
        }
```

BACnetObjectSelector ::= CHOICE {

```
        none            NULL,
        object          BACnetObjectIdentifier,
        object-type     BACnetObjectType
        }
```

```
BACnetOptionalPriorityFilter ::= CHOICE {
        default             NULL,
        filter              BACnetPriorityFilter
        }

BACnetAuditNotification ::= SEQUENCE {
        source-timestamp            [0] BACnetTimeStamp OPTIONAL,
        target-timestamp            [1] BACnetTimeStamp OPTIONAL,
        source-device               [2] BACnetRecipient,
        source-object               [3] BACnetObjectIdentifier OPTIONAL,
        operation                   [4] BACnetAuditOperation,
        source-comment              [5] CharacterString OPTIONAL,
        target-comment              [6] CharacterString OPTIONAL,
        invoke-id                   [7] Unsigned8 OPTIONAL,
        source-user-id              [8] Unsigned16 OPTIONAL,
        source-user-role            [9] Unsigned8 OPTIONAL,
        target-device               [10] BACnetRecipient,
        target-object               [11] BACnetObjectIdentifier,
        target-property             [12] BACnetPropertyReference OPTIONAL,
        target-priority             [13] Unsigned (1..16) OPTIONAL,
        target-value                [14] ABSTRACT-SYNTAX.&Type OPTIONAL,
        current-value               [15] ABSTRACT-SYNTAX.&Type OPTIONAL,
        result                      [16] Error OPTIONAL
        }

BACnetAuditLogRecord ::= SEQUENCE {
        timestamp           [0] BACnetDateTime,
        log-datum           [1] CHOICE {
                                    log-status          [0] BACnetLogStatus,
                                    audit-notification  [1] BACnetAuditNotification,
                                    time-change         [2] REAL
                                    }
        }

BACnetAuditLogRecordResult ::= SEQUENCE {
        sequence-number             [0] Unsigned64,
        log-record                  [1] BACnetAuditLogRecord
        }

BACnetAuditLogQueryParameters ::= CHOICE {
        by-target           [0] SEQUENCE {
                                    target-device-identifier     [0] BACnetObjectIdentifier,
                                    target-device-address        [1] BACnetAddress OPTIONAL,
                                    target-object-identifier     [2] BACnetObjectIdentifier OPTIONAL,
                                    target-property-identifier   [3] BACnetPropertyIdentifier OPTIONAL,
                                    target-array-index           [4] Unsigned OPTIONAL,
                                    target-priority              [5] Unsigned(1..16) OPTIONAL,
                                    operations                   [6] BACnetAuditOperationFlags OPTIONAL,
                                    successful-actions-only      [7] BOOLEAN
                                    },
        by-source           [1] SEQUENCE {
                                    source-device-identifier     [0] BACnetObjectIdentifier,
                                    source-device-address        [1] BACnetAddress OPTIONAL,
                                    source-object-identifier     [2] BACnetObjectIdentifier OPTIONAL,
                                    operations                   [3] BACnetAuditOperationFlags OPTIONAL,
                                    successful-actions-only      [4] BOOLEAN
                                    }
        }
```

BACnetSuccessFilter ::= ENUMERATED {
    all   (0),
    successes-only   (1),
    failures-only   (2)
    }

[Insert into production **BACnetConfirmedServiceChoice** in **Clause 21**, p. 784]

-- Alarm and Event Services
    ...
    confirmed-audit-notification   (32),

    ...

-- Object Access Services
    ...
    audit-log-query   (33),

    ...

-- *Services added after 2016*
    -- *confirmed-audit-notification   (32) see Alarm and Event Services*
    -- *audit-log-query   (33) see Object Access Services*
    }
-- Other services to be added ...

[Insert into production **BACnet-Confirmed-Service-Request** in **Clause 21**, p. 785]

-- Alarm and Event Services
    ...
    confirmed-audit-notification   [32] ConfirmedAuditNotification-Request,

    ...

-- Object Access Services
    ...
    audit-log-query   [33] AuditLogQuery-Request,

    ...

-- *Services added after 2016*
    -- *confirmed-audit-notification   [32] see Alarm and Event Services*
    -- *audit-log-query   [33] see Object Access Services*
    }
-- Context-specific tags 0..~~31~~ *33* are NOT used in the encoding ...

[Insert into production **BACnet-Confirmed-Service-ACK** in **Clause 21**, p. 786]
-- Object Access Services
    ...
    audit-log-query   [33] AuditLogQuery-ACK,

    ...

-- Context-specific tags 3..~~29~~ *33* are NOT used in the encoding ...

[Insert into production **BACnetUnconfirmedServiceChoice** in **Clause 21**, p. 794]

```
    ...
    unconfirmed-cov-notification-multiple     ~~(11)~~ *(11),*
    *unconfirmed-audit-notification*            *(12)*
    }
-- Other services to be added ...
```

[Insert into production **BACnet-Unconfirmed-Service-Request** in **Clause 21**, p. 794]

```
    ...
    unconfirmed-cov-notification-multiple     [11] UnconfirmedCOVNotificationMultiple-~~Request~~*Request,*
    *unconfirmed-audit-notification*            *[12] UnconfirmedAuditNotification-Request*
    }
-- Context-specific tags 0..~~11~~ *12* are NOT used in the encoding ...
```

[Insert new production into **Clause 21**, at end of section 'Confirmed Alarm and Event Services', p. 786]

```
ConfirmedAuditNotification-Request ::= SEQUENCE {
    notifications               [0] SEQUENCE OF BACnetAuditNotification
    }
```

[Insert new productions into **Clause 21**, at end of section 'Confirmed Object Access Services', p. 790]

```
AuditLogQuery-Request ::= SEQUENCE {
    audit-log                   [0] BACnetObjectIdentifier,
    query-parameters            [1] BACnetAuditLogQueryParameters,
    start-at-sequence-number    [2] Unsigned32 OPTIONAL,
    requested-count             [3] Unsigned16
    }

AuditLogQuery-ACK ::= SEQUENCE {
    audit-log                   [0] BACnetObjectIdentifier,
    records                     [1] SEQUENCE OF BACnetAuditLogRecordResult,
    no-more-items               [2] BOOLEAN
    }
```

[Insert new production into **Clause 21**, at end of section 'Unconfirmed Alarm and Event Services', p. 794]

```
UnconfirmedAuditNotification-Request ::= SEQUENCE {
    notifications               [0] SEQUENCE OF BACnetAuditNotification
    }
```

[Insert into production **BACnet-Error** in **Clause 21**, p. 797]

```
...
-- Alarm and Event Services
    ...
    *confirmed-audit-notification*      *[32] Error,*
```

```
...
-- Object Access Services
        ...
        audit-log-query                 [33] Error,

...
-- Services added after 2016
        -- confirmed-audit-notification     [32]  see Alarm and Event Services
        -- audit-log-query                  [33]  see Object Access Services
        }
-- Context-specific tags 0..~~31~~ 33 and 127 are NOT used in the encoding ...
```

[Insert into Application Types section of **Clause 21**, p. 804]

Unsigned64 ::=   Unsigned (0..18446744073709551615)   -- 0 .. 'the 64th power of two'-1

[Insert new entries in **Table 23-1** in **Clause 23**, p. 875]

BACnetAuditLevel	*0-127*	*255*
BACnetAuditOperation	*0-31*	*63*

[Add new **Clause K.X1** and subclauses to **Annex K**, p. 1078]

K.X1 Audit Reporting BIBBs

K.X1.1 BIBB - Audit Reporting-Logging- A (AR-L-A)

The A device is an audit logger. It logs received audit notifications from operation sources and operation targets for merging into its Audit Logs and forwards notifications to parent audit loggers.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x	x
UnconfirmedAuditNotification	x	x
AuditLogQuery		x

The audit logger receives audit event notifications and places them into local Audit Log objects.

The audit logger performs all functions of an audit logger as described in Clause 19.Y.

Devices claiming conformance to this BIBB are interoperable with devices claiming conformance to AR-R-B.

K.X1.2 BIBB - Audit Reporting-Reporter-B (AR-R-B)

The B device generates audit notifications and supports Audit Reporter objects. It is an operation target which generates audit notifications for local objects, and, if it performs actions, it is an operation source which generates audit notifications for its actions. See Clause 19.Y for more information on audit logging.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x[1]	
UnconfirmedAuditNotification	x[1]	

[1] At least one of these shall be supported.

A device claiming this BIBB supports modification of its audit reporting configuration. Specifically, the following Audit Reporter properties shall be writable:

Table K-X1. Writable Audit Reporter Properties

Audit_Level
Auditable_Operations
Audit_Priority_Filter

K.X1.3 BIBB - Audit Reporting-Reporter-Simple-B (AR-R-S-B)

The B device generates audit notifications. It is an operation target which generates audit notifications for local objects, and, if it performs actions, it is an operation source which generates audit notifications for its actions. See Clause 19.Y for more information on audit logging.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x[1]	
UnconfirmedAuditNotification	x[1]	

[1] At least one of these shall be supported.

K.X1.4 BIBB - Audit Reporting-Forwarder-B (AR-F-B)

The B device is an audit logger which receives audit notifications and forwards them to parent audit loggers. See Clause 19.X for more information on audit logging.

BACnet Service	Initiate	Execute
ConfirmedAuditNotification	x	x
UnconfirmedAuditNotification	x	x

K.X1.5 BIBB - Audit Reporting-View-A (AR-V-A)

The A device queries audit loggers using AuditLogQuery or ReadRange and presents query results to the user.

BACnet Service	Initiate	Execute
AuditLogQuery	x[1]	
ReadRange	x[1]	

[1] At least one of these shall be supported.

K.X1.6 BIBB - Audit Reporting-Advanced View and Modify-A (AR-AVM-A)

Device A configures auditing in all standard objects in Device B. Device A shall support DS-RP-A, DS-WP-A, DM-OCD-A, and AR-V-A. The A device shall be capable of using ReadProperty to retrieve and WriteProperty to modify any auditing related properties. Device A may use alternate services where support for execution of the alternate service is supported by Device B. Device A shall be capable of creating/deleting Audit Reporter and Audit Log objects in the B device.

BACnet Service	Initiate	Execute
AuditLogQuery	x	
CreateObject	x	
DeleteObject	x	
ReadProperty	x	
WriteProperty	x	

Table K-X2. Auditing Related Properties

Audit_Level
Auditable_Operations
Audit_Priority_Filter

The A device is able to use ReadProperty to retrieve and present all standard properties of the Audit Reporter and Audit Log object types, except those listed in Table K-2.

The A device is able to use WriteProperty to modify any standard property of the Audit Reporter and Audit Log object types where the property is not required to be read-only, or to which access is otherwise restricted by the standard (e.g., Log_Buffer).

[Append to **Clause L.1.2**, **Clause L.2.1, Clause L.3.1** p. 1079]
[Adds audit reporting requirements to all advanced workstation device profiles]
[Note that there are other addenda to 135-2016 which define new advanced workstation device profiles. Similar additions will be made to those profiles by those addenda.]

Audit Reporting

- *Query and display of logged audit records*
- *Ability to configure audit reporting in BACnet devices*

[Add new table section to **Clause L.1**, p. 1079]

Audit Reporting

B-AWS	**B-OWS**	**B-OD**
AR-AVM-A		

[Add new table section to **Clause L.2**, p. 1081]

Audit Reporting

B-ALSWS	**B-LSWS**	**B-LSAP**
AR-AVM-A		

[Add new table section to **Clause L.3**, p. 1084]

Audit Reporting

B-AACWS	**B-ACWS**	**B-ACSD**
AR-AVM-A		

135-2016*bi*-2. Change DeviceCommunicationControl Service for Audit Reporting.

Rationale

Audit reporting should not be suppressible by disabling communication via the DeviceCommunicationControl service. Clarify that audit notifications for changes in response to BACnet service requests are sent even if initiation has been disabled for the device.

Deprecate the DISABLE option of the DeviceCommunicationControl service. Having long been noted as not very useful, this option of the service is being deprecated at this time due to the complications added when combined with audit notifications.

[Change **Clause 16.1**, p. 706]

16.1 DeviceCommunicationControl Service

The DeviceCommunicationControl service is used by a client BACnet-user to instruct a remote device to stop initiating *BACnet services* ~~and optionally stop responding to all APDUs (except DeviceCommunicationControl or, if supported, ReinitializeDevice)~~ on the communication network or internetwork for a specified duration of time. This service is primarily used by a human operator for diagnostic purposes. A password may be required from the client BACnet-user prior to executing the service. The time duration can be set to "indefinite," meaning communication must be re-enabled by a DeviceCommunicationControl or, if supported, ReinitializeDevice service, not by time.

[Change **Clause 16.1.1.1.1**, p. 706]

16.1.1.1.1 Time Duration

This optional parameter, of type Unsigned16, indicates the number of minutes that the remote device shall *not initiate BACnet services* ~~ignore all APDUs except DeviceCommunicationControl and, if supported, ReinitializeDevice APDUs~~. If the 'Time Duration' parameter is not present, then the time duration shall be considered indefinite, meaning that only an explicit DeviceCommunicationControl or ReinitializeDevice APDU shall enable communications. The 'Time Duration' parameter shall be ignored and the time period considered ~~to be~~ *being* indefinite if the 'Enable/Disable' parameter has a value of ENABLE.

If the responding BACnet-user does not have a clock and the time duration is not indefinite, then the request shall be considered invalid and the responding BACnet-user shall issue a Result(-) response.

[Change **Clause 16.1.1.3.1, p. 706]**

16.1.1.3.1 Error Type

This parameter consists of two ~~components~~ *component* parameters: (1) the 'Error Class' and (2) the 'Error Code'. See Clause 18. The 'Error Class' and 'Error Code' to be returned for specific situations are as follows:

Situation	Error Class	Error Code
The password is invalid or absent when one is required.	SECURITY	PASSWORD_FAILURE
The device does not have a clock and the 'Time Duration' parameter is not set to "indefinite".	SERVICES	OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
If the request is valid and the 'Enable/Disable' parameter is the deprecated value DISABLE	*SERVICES*	*SERVICE_REQUEST_DENIED*

[Change **Clause 16.1.2**, p. 707]

16.1.2 Service Procedure

After verifying the validity of the request, including the 'Time Duration' and 'Password' parameters, the responding BACnet-user shall respond with a 'Result(+)' service primitive. ~~If the request is valid and the 'Enable/Disable' parameter is DISABLE, the responding BACnet-user shall discontinue responding to any subsequent messages except DeviceCommunicationControl and, if supported, ReinitializeDevice messages, and shall discontinue initiating messages.~~ If the request is valid and the 'Enable/Disable' parameter is DISABLE_INITIATION, the responding BACnet-user shall discontinue the initiation of *BACnet* messages except for I-Am requests issued in accordance with the Who-Is service procedure *and ConfirmedAuditNotification and UnconfirmedAuditNotification requests*. Communication shall be disabled in this manner until either the 'Time Duration' has expired or a valid DeviceCommunicationControl (with 'Enable/Disable' = ENABLE) or, if supported, a valid ReinitializeDevice (with 'Reinitialized State of Device' = WARMSTART or COLDSTART) message is received.

If the responding BACnet-user does not have a clock and the 'Time Duration' parameter is not set to "indefinite," the APDU shall be ignored and a 'Result(-)' service primitive shall be issued. If the 'Password' parameter is invalid or absent when a password is required, the APDU shall be ignored and an Error-PDU with 'error class' = SECURITY and 'error code' = PASSWORD_FAILURE shall be issued.
If the request is valid and the 'Enable/Disable' parameter is the deprecated value DISABLE, the request shall be ignored and an Error-PDU with 'error class' = SERVICES and 'error code' = SERVICE_REQUEST_DENIED shall be issued.

[Change **COMMUNICATION_DISABLED** description in **Clause 18.6**, p. 739]

> **COMMUNICATION_DISABLED** - Communication has been disabled due to receipt of a DeviceCommunicationControl request. *This error is not expected in response to service requests but rather is expected to be used internally when actions that would normally result in initiation of a BACnet service fail due to communications being disabled. As such, this error code would be seen in objects that record the result of failed operations, such as logging objects.*

[Change **Clause 21**, p. 793]

DeviceCommunicationControl-Request ::= SEQUENCE {
```
        timeDuration      [0] Unsigned16 OPTIONAL,
        enable-disable    [1] ENUMERATED {
                                  enable              (0),
                                  ~~disable             (1),~~ -- 'disable' deprecated in version 1 revision 20
                                  disable-initiation  (2)
                                   },
        password          [2] CharacterString (SIZE(1..20)) OPTIONAL
        }
```

[Change Clause E.4.1, p. 954]

E.4.1 An Example of the DeviceCommunicationControl Service

While troubleshooting a problem on a BACnet network, it becomes necessary to stop communication exchanges from a particular device for a period of five minutes. This is accomplished by means of the DeviceCommunicationControl Service as follows.

Service =	DeviceCommunicationControl
'Time Duration' =	5
'Enable/Disable' =	DISABLE_*INITIATION*
'Password' =	"#egbdf!"

[Change Clause F.4.1, p. 982]

F.4.1 Encoding for Example E.4.1 - DeviceCommunicationControl Service

X'00'	PDU Type=0 (BACnet-Confirmed-Request-PDU, SEG=0, MOR=0, SA=0)

X'04'	Maximum APDU Size Accepted=1024 octets
X'05'	Invoke ID=5
X'11'	Service Choice=17 (DeviceCommunicationControl-Request)

X'09'	SD Context Tag 0 (Time Duration, L=1)
X'05'	5
X'19'	SD Context Tag 1 (Enable-Disable, L=1)
X'0~~1~~*2*'	~~1~~*2* (DISABLE_*INITIATION*)
X'2D'	SD Context Tag 2 (Password, L>4)
X'08'	Extended Length=8
X'00'	ISO 10646 (UTF-8) Encoding
X'23656762646621'	"#egbdf!"

135-2016*bi*-3. Modify Logging Objects to Allow for Extremely Large Logs.

Rationale

Allow for logging objects to store more than 2^{32}-1 entries and to remove the need to re-allocate log buffer entries for large logging objects.

This change recognizes the inability of many implementations to pre-determine the size required for records and thus the number of records that can be stored by the object.

[Change **Enable** property descriptions in logging objects, Clauses **12.25.5, 12.27.8, 12.30.8**]

This property, of type BOOLEAN, indicates and controls whether (TRUE) or not (FALSE) logging of events and collected data is enabled. Logging occurs if and only if Enable is TRUE, Local_Time is on or after Start_Time, and Local_Time is before Stop_Time. If Start_Time contains an unspecified datetime, then it shall be considered equal to 'the start of time'. If Stop_Time contains an unspecified datetime, then it shall be considered equal to 'the end of time'. Log records of type log-status are recorded without regard to the value of the Enable property.

Attempts to write the value TRUE to the Enable property while Stop_When_Full is TRUE and *either* Record_Count is equal to Buffer_Size, *or Buffer_Size is 2^{32}-1 and there is no space for another record*, shall cause a Result(-) response to be issued, specifying an 'Error Class' of OBJECT and an 'Error Code' of LOG_BUFFER_FULL.

[Change **Stop_When_Full** property descriptions in logging objects, **Clauses 12.25.12, 12.27.11, 12.30.17**]

This property, of type BOOLEAN, specifies whether (TRUE) or not (FALSE) logging should cease when the log buffer is full. When logging ceases because the addition of one more log record would cause the log buffer to be full, Enable shall be set to FALSE and the event recorded.

If Stop_When_Full is writable, attempts to write the value TRUE to the Stop_When_Full property while Record_Count is equal to Buffer_Size shall result in the oldest Log_Buffer record being discarded, and shall cause the Enable property to be set to FALSE and the event to be recorded.

If Stop_When_Full is TRUE and Buffer_Size is 2^{32}-1 and the addition of a new record fails due to a lack of space, then the oldest Log_Buffer record shall be discarded, the Enable property shall be set to FALSE and the event is recorded.

[Change **Buffer_Size** property descriptions in logging objects, **Clauses 12.25.13, 12.27.12, 12.30.18**]

This property, of type Unsigned32, shall specify the maximum number of log records the log buffer may hold. If writable, it may not be written when Enable is TRUE. The disposition of existing log records when Buffer_Size is written is a local matter. If all records are deleted when the Buffer_Size is written then the object shall act as if the Record_Count was set to zero.

For products that support very large log objects, the value 2^{32}-1 is reserved to indicate that the buffer size is unknown and is constrained solely by currently available resources.

[Change **Table 15-14**, p. 693]

Reference Sequence Number	M	M(=)	Unsigned~~32~~

[Change **Clause 15.8.1.1.4.1.1**, p. 693]

15.8.1.1.4.1.1 Reference Index

The 'Reference Index' parameter specifies the index of the first (if 'Count' is positive) or last (if 'Count' is negative) item to be read. If the item with the index specified in this parameter does not exist, then no items match the criteria for being read and returned, regardless of the value of the 'Count' parameter.

Devices that execute ReadRange shall support Unsigned32 values for reference indexes unless the device contains object types which have Total_Record_Count properties of type Unsigned64, or which are capable of containing list properties with more than 2^{32}-1 entries, in which case the device shall support Unsigned64 reference indexes.

Devices which initiate the By Position form of ReadRange with arbitrary reference indexes shall support Unsigned32 values for reference indexes unless the device interacts with object types which have Total_Record_Count properties of type Unsigned64, in which case the device shall support Unsigned64 reference indexes.

[Change **Clause 15.8.1.1.4.2.1**, p. 694]

15.8.1.1.4.2.1 Reference Sequence Number

The 'Reference Sequence Number' parameter specifies the sequence number of the first (if 'Count' is positive) or last (if 'Count' is negative) item to be read. If the item with the sequence number specified in this parameter does not exist, then no items match the criteria for being read and returned, regardless of the value of the 'Count' parameter.

Devices that execute ReadRange shall support Unsigned32 values for sequence numbers, unless the device contains object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

Devices which initiate ReadRange shall support Unsigned32 values for sequence numbers, unless the device interacts with object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

[Change **Clause 15.8.1.2.7**, p. 697]

15.8.1.2.7 First Sequence Number

This parameter, of type Unsigned~~32~~, specifies the sequence number of the first item returned. This parameter is only included if the 'Range' parameter of the request was of the type 'By Sequence Number' or 'By Time' and 'Item Count' is greater than 0.

Devices that execute ReadRange shall support Unsigned32 values for sequence numbers, unless the device contains object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

Devices which initiate ReadRange shall support Unsigned32 values for sequence numbers, unless the device is intended to interact with object types which have Total_Record_Count properties of type Unsigned64 in which case the device shall support Unsigned64 sequence numbers.

[Add a new entry to **History of Revisions**, p. 1364]

(This History of Revisions is not part of this standard. It is merely informative and does not contain requirements necessary for conformance to the standard.)

HISTORY OF REVISIONS

. . .	. . .	**. . .**
1	20	**Addendum *bi* to ANSI/ASHRAE Standard 135-2016** Approved by ASHRAE on June 15, 2018, and by the American National Standards Institute on June 15, 2018. 1. Add Audit Reporting. 2. Change DeviceCommunicationControl Service for Audit Reporting. 3. Modify Logging Objects to Allow for Extremely Large Logs.

POLICY STATEMENT DEFINING ASHRAE'S CONCERN FOR THE ENVIRONMENTAL IMPACT OF ITS ACTIVITIES

ASHRAE is concerned with the impact of its members' activities on both the indoor and outdoor environment. ASHRAE's members will strive to minimize any possible deleterious effect on the indoor and outdoor environment of the systems and components in their responsibility while maximizing the beneficial effects these systems provide, consistent with accepted Standards and the practical state of the art.

ASHRAE's short-range goal is to ensure that the systems and components within its scope do not impact the indoor and outdoor environment to a greater extent than specified by the Standards and Guidelines as established by itself and other responsible bodies.

As an ongoing goal, ASHRAE will, through its Standards Committee and extensive Technical Committee structure, continue to generate up-to-date Standards and Guidelines where appropriate and adopt, recommend, and promote those new and revised Standards developed by other responsible organizations.

Through its *Handbook*, appropriate chapters will contain up-to-date Standards and design considerations as the material is systematically revised.

ASHRAE will take the lead with respect to dissemination of environmental information of its primary interest and will seek out and disseminate information from other responsible organizations that is pertinent, as guides to updating Standards and Guidelines.

The effects of the design and selection of equipment and systems will be considered within the scope of the system's intended use and expected misuse. The disposal of hazardous materials, if any, will also be considered.

ASHRAE's primary concern for environmental impact will be at the site where equipment within ASHRAE's scope operates. However, energy source selection and the possible environmental impact due to the energy source and energy transportation will be considered where possible. Recommendations concerning energy source selection should be made by its members.