

BSR/ASHRAE Addendum *j*
to ANSI/ASHRAE Standard 135-2004

Public Review Draft

ASHRAE® Standard

Proposed Addendum *j* to Standard 135-2004, *BACnet®—A Data Communication Protocol for Building Automation and Control Networks*

First Public Review (**March 2007**)
(Draft Shows Proposed Changes to
Current Standard)

This draft has been recommended for public review by the responsible project committee. To submit a comment on this proposed addendum, use the comment form and instructions provided with this draft. The draft is subject to modification until it is approved for publication by the Board of Directors and ANSI. Until this time, the current edition of the standard (as modified by any published addenda on the ASHRAE web site) remains in effect. The current edition of any standard may be purchased from the ASHRAE Bookstore @ <http://www.ashrae.org> or by calling 404-636-8400 or 1-800-527-4723 (for orders in the U.S. or Canada).

This standard is under continuous maintenance. To propose a change to the current standard, use the change submittal form available on the ASHRAE web site @ <http://www.ashrae.org>.

The appearance of any technical data or editorial material in this public review document does not constitute endorsement, warranty, or guaranty by ASHRAE of any product, service, process, procedure, or design, and ASHRAE expressly disclaims such.

© **March 15, 2007**. This draft is covered under ASHRAE copyright. Permission to reproduce or redistribute all or any part of this document must be obtained from the ASHRAE Manager of Standards, 1791 Tullie Circle, NE, Atlanta, GA 30329. Phone: 404-636-8400, Ext. 1125. Fax: 404-321-5478. E-mail: standards.section@ashrae.org.

AMERICAN SOCIETY OF HEATING,
REFRIGERATING AND AIR-CONDITIONING
ENGINEERS, INC.
1791 Tullie Circle, NE · Atlanta GA 30329-2305



[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 a proposed change for public review. 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 proposed changes are summarized below.

SSPC 135 wishes to recognize the efforts of the following people in developing this addendum: **td.**

- 135-2004j-1. Add a new Access Point object type, p. 1.
- 135-2004j-2. Add a new Access Zone object type, p. 23.
- 135-2004j-3. Add a new Access User object type, p. 33.
- 135-2004j-4. Add a new Access Rights object type, p. 37.
- 135-2004j-5. Add a new Access Credential object type, p. 42.
- 135-2004j-6. Add a new Authentication Factor Input object type, p. 50.
- 135-2004j-7. Add a new ACCESS_EVENT event algorithm, p. 57.

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135-2004 and Addenda is indicated through the use of *italics*, while deletions are indicated by ~~strike through~~. Where entirely new subclauses are proposed to be added, plain type is used throughout. Only this new and deleted text is open to comment as this time. All other material in this addendum is provided for context only and is not open for public review comment except as it relates to the proposed changes.

135-2004j-1. Add a new Access Point object type.

Rationale

Support for access control in BACnet includes requires a standardized object whose properties represent the externally visible characteristics associated with the authentication and authorization process of an access controlled point. (e.g., door, gate, turnstile).

Addendum 135-2004j-1

[Insert the following new clauses in 3.2, pp. 2-5, distributing them alphabetically and renumbering subsequent clauses]

3.2.x authentication factor: a piece of information used to verify a credential's identity.

3.2.x authorization (physical access control): the process of determining whether the access user is permitted to access the zone that they have requested to enter through an access control door.

3.2.x credential (physical access control): the combination of authentication factors and access rights.

3.2.x access user (physical access control): the person or asset holding one or more credentials.

3.2.x access rights (physical access control): the access privileges granted to a credential.

3.2.x role-based access control (RBAC): access privileges that are assigned to specific roles. Access users acquire privileges through their assigned role.

[Insert new clause 12.X and Table 12-X, p. 251]

12.X Access Point Object Type

The Access Point object type defines a standardized object whose properties represent the externally visible characteristics associated with the authentication and authorization process of an access controlled point. (e.g., door, gate, turnstile). Access through this point is directional in that it represents access in one direction only. A door, in which access is controlled in both directions, is represented by two separate Access Point objects.

Authentication is the process of verifying the identity of an access user requesting access through an access-controlled door. This can be as simple as a single-factor authentication, in which one authentication factor (i.e., magnetic-stripe card, proximity-card, smart card) is used to identify a known user. In multifactor authentication, a combination of two or more authentication factors (e.g., card + PIN, card + biometric) are used to verify the identity of the access user. The Access Point object supports the definition of single-factor and multi-factor authentication policies, including the functionality to switch the policy in effect. On false attempts to authenticate, the Access Point may lock-down, for an infinite or specific amount of time.

Authorization is the process of determining whether the access user is permitted to access the zone that he or she has requested to enter. Once the access user has been authenticated successfully, a list of criteria is checked to determine whether access can be granted. If one or more of the authorization criteria fail, then the access user is denied access. Once the access user is granted access, the door will be commanded unlocked at the specified command priority and the access user can access the zone. The door which is controlled is specified in the Access Point. Authorization criteria supported by the Access Point are authorization modes, occupancy enforcement and threat level.

Authentication and authorization begins when an access user presents an authentication factor at the access controlled point. The process this object represents consumes the authentication factors from the corresponding Authentication Factor Input objects and performs the authentication and authorization functions. The result is to grant or deny access and to generate corresponding access events. If the object is out of service or not reliable, then these functions are not performed.

The Access Point object generates access events. Access events are stateless (i.e., NORMAL to NORMAL transitions

only). A single transaction, such as a request to enter or an operator action, can result in one or more access events.

For intrinsic reporting, the ACCESS_EVENT algorithm is applied for both Access Alarm Events and Access Transaction Events:

Access Alarm Events: These are events requiring operator attention and handling, and the Access Point object may request human operator acknowledgement of these events.

Access Transaction Events: These are events that are to be logged, not requiring immediate operator attention. The Access Point object does not request human operator acknowledgement of these events.

Access Points which authorize entrance to an access controlled zone are ingress points of that zone. Access Points which authorize exit from an access controlled zone are egress points of that zone. In the typical case a specific Access Point is an egress point from one zone and an ingress point to an adjacent zone. If the Access Point leads from an Access Zone to no zone (i.e., outside) then the Access Point is an egress point only. If the Access Point leads from no zone (i.e., outside) to an Access Zone, then the Access Point is an ingress point only. If the Access Point does not lead from or to an Access Zone (e.g., internal check point or muster point) then the Access Point is neither an ingress or egress point.

The Access Point object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Access Point 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
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Authentication_Policy_Names	BACnetARRAY[N] of CharacterString	O ¹
Authentication_Policy_List	BACnetARRAY[N]of BACnetAuthenticationPolicy	O ²
Authentication_Policy_Timeouts	BACnetARRAY[N] of Unsigned	O ²
Active_Authentication_Policy	Unsigned	O ¹
Authorization_Mode	BACnetAuthorizationMode	O
Lockdown	BOOLEAN	O ³
Lockdown_Relinquish_Time	Unsigned	O
Failed_Attempts	Unsigned	O
Max_Failed_Attempts	Unsigned	O ⁴
Failed_Attempts_Time	Unsigned	O ⁴
Threat_Level	BACnetAccessThreatLevel	O
Occupancy_Upper_Threshold_Enforced	BOOLEAN	O
Occupancy_Lower_Threshold_Enforced	BOOLEAN	O
Last_Access_Event	BACnetAccessEvent	O ^{5,6}
Update_Time	BACnetTimeStamp	O ^{5,6}
Last_Access_Credential	BACnetObjectIdentifier	O ^{5,6}
Last_Authentication_Factor	BACnetAuthenticationFactor	O
Access_Doors	BACnetARRAY[N] of BACnetDeviceObjectReference	R
Priority_For_Writing	Unsigned (1...16)	R
Muster_Point	BOOLEAN	O
Zone_To	BACnetDeviceObjectReference	O

Zone_From	BACnetDeviceObjectReference	O
Notification_Class	Unsigned	O ⁵
Transaction_Notification_Class	Unsigned	O
Access_Alarm_Events	List of BACnetAccessEvent	O ⁵
Access_Transaction_Events	List of BACnetAccessEvent	O ⁵
Event_Enable	BACnetEventTransitionBits	O ⁵
Acked_Transitions	BACnetEventTransitionBits	O ⁵
Notify_Type	BACnetNotifyType	O ⁵
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O ⁵
Profile_Name	CharacterString	O

¹ These properties are required to be present if this object supports authentication policies.

² The size of this array shall be the same as the size of the Authentication_Policy_Names array.

³ This property is required to be present if Lockdown_Relinquish_Time is present.

⁴ If this property is present, then the Failed_Attempts property shall be present.

⁵ These properties are required to be present if the object supports intrinsic reporting.

⁶ These properties are required if the object supports COV 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 ACCESS_POINT.

12.X.4 Description

This optional 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 Access Point. 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 Logical TRUE (1) if the Access Point has been overridden by some mechanism local to the BACnet Device. Otherwise, the value is logical FALSE (0).

OUT_OF_SERVICE Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.X.6 Event_State

The Event_State property, of type BACnetEventState, is included in order to provide a way to determine if this object has an active event state associated with it. If the object supports intrinsic reporting, then the Event_State property shall indicate the event state of the object. The allowed event states are NORMAL and FAULT. If the object does not support intrinsic reporting, then:

- (a) if the Reliability property is not present, then the value of Event_State shall be NORMAL, or
- (b) if the Reliability property is present and Reliability is NO_FAULT_DETECTED, then Event_State shall be NORMAL, or
- (c) if the Reliability property is present and Reliability is not NO_FAULT_DETECTED, then Event_State shall be FAULT.

12.X.7 Reliability

The optional Reliability property, of type BACnetReliability, provides an indication of whether the authentication and authorization process this object represents is "reliable" as far as the BACnet Device can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, PROCESS_ERROR, CONFIGURATION_ERROR, UNRELIABLE_OTHER}.

If Reliability has a value other than NO_FAULT_DETECTED, the process that this object represents shall not perform any authentication or authorization. No access events are generated in this case.

12.X.7.1 Conditions for Generating a TO-FAULT Event

A TO-FAULT event is generated under these conditions:

- (a) the Reliability property becomes not equal to NO_FAULT_DETECTED, and
- (b) the TO-FAULT flag is set in the Event_Enable property.

12.X.8 Out_Of_Service

The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the authentication and authorization process this object represents is out of service. If out of service, then the process that this object represents shall not perform any authentication or authorization.

12.X.9 Authentication_Policy_Names

This optional property, of type BACnetARRAY [N] of CharacterString, specifies the names of the defined authentication policies. Authentication policies define which authentication factors have to be presented for valid authentication. If authentication policies are not explicitly defined in Authentication_Policy_List, then the authentication policies are a local matter.

An Authentication_Policy_Names entry shall have a minimum length of one character, otherwise it is considered invalid.

This property is required to be present if the object supports authentication policies.

12.X.9.1 Resizing Authentication_Policy_Names Array, Authentication_Policy_List Array and Authentication_Policy_Timeouts Array by Writing Any of these Array Properties

The size of the Authentication_Policy_Names, Authentication_Policy_List, and Authentication_Policy_Timeouts arrays shall be maintained so that all have the same size. If any of these arrays are present and writable and the number of elements of one is reduced, each of the arrays shall be truncated to the new reduced size. If any of these arrays are present and writable and the number of elements of one is increased, then each of the arrays shall be increased to the new expanded size and the new array elements initialized according to the requirements of each property. See 12.X.9.2, 12.X.10.2 and 12.X.11.2.

12.X.9.2 Initializing New Array Elements When the Array Size is Increased

If the size of the `Authentication_Policy_Names` array is increased without entry values being provided, then the new array entries shall be initialized with empty character strings.

12.X.10 Authentication_Policy_List

This optional property, of type `BACnetARRAY[N]` of `BACnetAuthenticationPolicy`, shall specify the authentication policies defined at this Access Point.

Each `BACnetAuthenticationPolicy` is a sequence of elements of type `BACnetAuthenticationRule`. `BACnetAuthenticationRule` contains two fields. The first, 'Authentication-Factor-Requirement', is a choice of options that specifies the authentication factors required for successful authentication. The second, 'Authentication-Factor-Input', is an optional field of type `BACnetDeviceObjectPropertyReference` and contains a reference to a property of type `BACnetAuthenticationFactor`, which specifies where to obtain the authentication factor.

The options for 'Authentication-Factor-Requirement' are:

REQUIRED	The authentication factor specified is required to be presented as defined in this policy for successful authentication. The Authentication-Factor-Input field shall be present.
CHOICE_BEGIN	Opens a group of OPTION entries. One of the authentication factors specified by the OPTION entries following this and before the CHOICE_END entry is required for successful authentication. Before another CHOICE_BEGIN follows, a CHOICE_END shall be present. The Authentication-Factor-Input field shall not be present.
CHOICE_END	Closes a group of OPTION entries opened with a CHOICE_BEGIN entry. The Authentication-Factor-Input field shall not be present.
OPTION	The authentication factor specified is an option of a choice of authentication factors specified by all OPTION entries within the CHOICE_BEGIN and CHOICE_END boundaries. The Authentication-Factor-Input field shall be present.

An `Authentication_Policy_List` entry shall be considered invalid if the entry is empty or if it is not well formed.

If this property is present, the `Authentication_Policy_Names` property shall also be present. The i^{th} element of this array corresponds to the i^{th} element of the `Authentication_Policy_Names` array. The size of this array shall equal the size of the `Authentication_Policy_Names` array.

12.X.10.1 Resizing Authentication_Policy_Names Array, Authentication_Policy_List Array, and Authentication_Policy_Timeouts Array by Writing Any of these Array Properties

See 12.X.9.1.

12.X.10.2 Initializing New Array Elements When the Array Size is Increased

If the size of the `Authentication_Policy_List` array is increased without entry values being provided, then the new array entries shall be initialized with an empty sequence of elements of type `BACnetAuthenticationRule`.

12.X.11 Authentication_Policy_Timeouts

This optional property, of type `BACnetARRAY[N]` of `Unsigned`, exposes the total time period in seconds available to present all the authentication factors as defined by the corresponding policy; a value of zero indicates an unlimited time. If a timeout occurs, the timer is reset and the all the authentication factors are required to be presented again.

If this property is present, then the Authentication_Policy_Names property shall be present. The i^{th} element of this array corresponds to the i^{th} element of the Authentication_Policy_Names array. The size of this array shall equal the size of the Authentication_Policy_Names array.

12.X.11.1 Resizing Authentication_Policy_Names Array, Authentication_Policy_List Array, and Authentication_Policy_Timeouts Array by Writing Any of these Array Properties

See 12.X.9.1.

12.X.11.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Authentication_Policy_Timeouts array is increased without entry values being provided, the new array entries shall be initialized to zero.

12.X.12 Active_Authentication_Policy

This optional property, of type Unsigned, shall specify the active authentication policy. The value of this property is an array index which corresponds to the active authentication policy as defined in the Authentication_Policy_Names property. If no valid authentication policy exists or the current active authentication policy is not valid (i.e., if the Authentication_Policy_Names array is empty or the Authentication_Policy_List entry is empty or not well formed), this property shall take a value of zero. If the Authentication_Policy_Names array size is reduced such that the value of this property becomes greater than the size of the Authentication_Policy_Names array, then this property shall take a value of zero. If this property has the value of zero, then the Out_Of_Service property shall be TRUE and the Reliability property shall have the value CONFIGURATION_ERROR.

The size of the Authentication_Policy_Names array specifies the maximum value that can be written to Active_Authentication_Policy. If a value is written greater than the maximum value or specifies an index of an invalid entry in that array, then the write attempt is denied and a Result(-) specifying an 'Error Class' of PROPERTY and an 'Error Code' of VALUE_OUT_OF_RANGE shall be returned.

If this property is present, the Authentication_Policy_Names property shall be present.

12.X.13 Authorization_Mode

This optional property, of type BACnetAuthorizationMode, determines how authorization is performed at the Access Point. The enumeration BACnetAuthorizationMode has the values:

AUTHORIZE	The access rights of an active credential are evaluated, in addition to other possible authorization checks.
GRANT_ACTIVE	An active credential is granted access without evaluating the access rights assigned to the credential. Other authorization checks can still lead to denying access.
DENY_ALL	Any credential, whether active or not, is denied access. Other authorization checks may be performed, but shall not lead to granting access.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary authorization modes other than those defined by the standard. For proprietary extensions of this enumeration see clause 23.1 of this standard.

12.X.14 Lockdown

This optional property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the access controlled point this object represents is in a lockdown state. When the Access Point is in a lockdown state any authentication and authorization requests shall fail, and for each failed attempt the Last_Access_Event shall be set to DENIED_LOCKDOWN. An Access Point may be set to a lockdown state due to too many failed access attempts, as defined in the Max_Failed_Attempts property, or by writing TRUE to this property.

When the property Lockdown becomes TRUE due to too many failed authentication attempts, the Access Point object

shall set Last_Access_Event to LOCKDOWN_MAX_ATTEMPTS. If TRUE is written to this property for any other reason, the Access Point object shall set Last_Access_Event to LOCKDOWN_OTHER. When the Lockdown property becomes FALSE, the Access Point shall set Last_Access_Event to LOCKDOWN_RELINQUISHED.

12.X.15 Lockdown_Relinquish_Time

This optional property, of type Unsigned, shall specify the time, in seconds, to delay after the Lockdown property has taken on the value TRUE, before automatically relinquishing the lockdown state. The lockdown state is relinquished by writing FALSE to the Lockdown property. A value of zero indicates that the lockdown state will not automatically be relinquished.

If the Lockdown_Relinquish_Time is present, the Lockdown property shall be present.

12.X.16 Failed_Attempts

This optional property, of type Unsigned, shall indicate the current count of successive failed authentication attempts. Any successive failed authentication attempt shall increment the value of this property.

This property shall be set to zero when a successful authentication occurs or when the property Lockdown becomes FALSE.

12.X.17 Max_Failed_Attempts

This optional property, of type Unsigned, shall specify the maximum number of successive failed authentication attempts before the Lockdown property is set to TRUE. If the Failed_Attempts property becomes greater than or equal to the value of this property and this property is not zero, the Lockdown property is set to TRUE. Zero indicates that the Lockdown property is not set to TRUE as the result of failed authentication attempts.

If the Max_Failed_Attempts property is present, the Failed_Attempts property shall be present.

12.X.18 Failed_Attempts_Time

This optional property, of type Unsigned, shall specify the time, in seconds, to delay before setting the Failed_Attempts property to zero, after the last failed authentication attempt. If an intervening successful access occurs, the Failed_Attempts property is set to zero.

If the Failed_Attempts_Time is present the Failed_Attempts property shall be present.

12.X.19 Threat_Level

This optional property, of type BACnetAccessThreatLevel, shall specify the current threat level for this Access Point. Zero is the lowest threat level, effectively disabling the threat level check, while 100 is the maximum threat level. If the threat authority of the authenticated credential is lower than the value of this property, then the authorization fails.

12.X.20 Occupancy_Upper_Threshold_Enforced

This optional property, of type BOOLEAN, indicates whether the upper occupancy threshold of the access controlled zone, for which this object is an ingress point, is enforced (TRUE) or not (FALSE). If enforced, authorization shall fail if the access controlled zone's occupancy is greater than or equal to its upper occupancy threshold, unless the credential is exempted from this authorization check.

12.X.21 Occupancy_Lower_Threshold_Enforced

This optional property, of type BOOLEAN, indicates whether the lower occupancy threshold of the access controlled zone, for which this object is an egress point, is enforced (TRUE) or not (FALSE). If enforced, authorization shall fail if the access controlled zone's occupancy is lower than or equal to its lower occupancy threshold, unless the credential is

exempted from this authorization check.

12.X.22 Last_Access_Event

This optional property, of type BACnetAccessEvent, shall specify the last access event at the Access Point.

BACnetAccessEvent is an enumeration of authentication and authorization decisions, subsequent actions, and results of these actions. An Access Point is not required to support all values of this enumeration.

NONE	The Access Point did not yet determine any access event. This is not a reported event. It is required to enable algorithmic ACCESS_EVENT reporting.
MUSTER	If the Access Point is a muster point a muster event is generated when an Access Credential is presented.
PASSBACK_DETECTED	A passback violation for the presented Access Credential has been detected.
DURESS	A duress incident was detected at this Access Point.
TRACE	The Access Credential presented has the Trace_Flag set.
LOCKDOWN_MAX_ATTEMPTS	The Access Point is in a lockdown state due to maximum failed authentication attempts.
LOCKDOWN_OTHER	The Access Point is in a lockdown state due to any reason other than maximum failed authentication attempts.
LOCKDOWN_RELINQUISHED	The Access Point has relinquished the lockdown state.
LOCKED_BY_HIGHER_PRIORITY	The controlled Access Door is commanded at a higher priority.
GRANTED	Access granted to the presented Access Credential
DENIED_DENY_ALL	Access denied because the authorization mode of the Access Point is set to DENY_ALL.
DENIED_UNKNOWN_CREDENTIAL	Access denied due to unknown credential value
DENIED_MISSING_AUTHENTICATION_FACTOR	Access denied due to missing authentication factor for multi-factor authentication.
DENIED_ZONE_NO_ACCESS_RIGHTS	Access denied due to no matching Access Rights to the Access Zone found for the presented Access Credential.
DENIED_POINT_NO_ACCESS_RIGHTS	Access denied due to no matching Access Rights to the Access Point found for the presented Access Credential.
DENIED_OUT_OF_TIME_RANGE	Access denied due to the presented Access Credential not being valid at this Access Point or Access Zone at this time.
DENIED_THREAT_LEVEL	Access denied due to insufficient threat authority for the presented Access Credential.
DENIED_PASSBACK	Access denied due to a passback violation for the Access Credential.
DENIED_UNEXPECTED_LOCATION_USAGE	Access denied due to the Access Credential used at a location which is not expected.

DENIED_MAX_ATTEMPTS	Access denied due to too many retries for the presented Access Credential.
DENIED_LOW_OCCUPANCY_THRESHOLD	Exit from a zone for which this Access Point is an Egress Access Point is denied due to zone occupancy below or at the minimum threshold.
DENIED_HIGH_OCCUPANCY_THRESHOLD	Access to a zone for which this Access Point is an Ingress Access Point is denied due to zone occupancy at or above the maximum threshold.
DENIED_CREDENTIAL_UNASSIGNED	Access denied due to the Access Credential used has not yet been assigned to an Access User.
DENIED_CREDENTIAL_INVALID	Access denied due to the Access Credential used is invalid.
DENIED_CREDENTIAL_NOT_YET_ACTIVE	Access denied due to the Access Credential used is not yet active.
DENIED_CREDENTIAL_EXPIRED	Access denied due to the Access Credential used is expired.
DENIED_CREDENTIAL_MANUAL_DISABLE	Access denied due to the Access Credential used is manually disabled.
DENIED_CREDENTIAL_LOCKED_OUT	Access denied due to the Access Credential used is locked out.
DENIED_CREDENTIAL_EXCEEDED_USE	Access denied due to the number of allowed uses of the Access Credential used has been exceeded.
DENIED_CREDENTIAL_INACTIVITY	Access denied due to the Access Credential used being disabled after a period of inactivity.
DENIED_CREDENTIAL_MISSING	Access denied due to the Access Credential used being reported as missing.
DENIED_CREDENTIAL_DAMAGED	Access denied due to the Access Credential used being reported as damaged.
DENIED_CREDENTIAL_DESTROYED	Access denied due to the Access Credential used being reported as destroyed.
DENIED_CREDENTIAL_DISABLED	Access denied due to the Access Credential used is disabled for unspecified or unknown reasons.
DENIED_CREDENTIAL_TIMEOUT	Access denied due to the proper Access Credential not being presented within a specified time.
DENIED_INCORRECT_CREDENTIAL	Access denied due to the Access Credential entered is incorrect or not the credential expected.
DENIED_NO_ACCOMPANIMENT	Access denied due to the expected accompanying Access Credential not being presented.
DENIED_INCORRECT_ACCOMPANIMENT	Access denied due to the accompanying Access Credential presented was incorrect
DENIED_LOCKDOWN	Access denied due to the Access Point being in lockdown state.
DENIED_OTHER	Access is denied for unspecified reasons.

<Proprietary Enum Values>

A vendor may use other proprietary enumeration values to indicate Access Events other than those defined by this standard. For proprietary extensions of this enumeration see clause 23.1 of this standard.

This property is required if intrinsic reporting is supported by this object. This property is required if the object supports COV reporting.

12.X.23 Update_Time

This optional property, of type BACnetTimeStamp, indicates the most recent update time of the Last_Access_Event property. This property shall update its value on each update of Last_Access_Event. Update times of type Time or Date shall have X 'FF' in each octet, and Sequence number update times shall have the value 0 if no update has yet occurred.

This property shall be present if the Last_Access_Event property is present or if the object supports COV reporting.

12.X.24 Last_Access_Credential

This optional property, of type BACnetObjectIdentifier, shall specify the Access Credential object that corresponds to the access event specified in the Last_Access_Event property if applicable, otherwise it shall contain 4194303 for the object instance.

This property shall be present if the Last_Access_Event property is present or if the object supports COV reporting.

12.X.25 Last_Authentication_Factor

This optional property, of type BACnetAuthenticationFactor, shall specify the last authentication factor read. If there was no Authentication Factor read up to now, or the device chooses not to expose the last authentication factor, this property shall contain UNDEFINED with NULL in the value field.

12.X.26 Access_Doors

This property, of type BACnetARRAY[N] of BACnetDeviceObjectReference, shall specify the references to those Access Door objects whose Present_Value properties are commanded after successful authorization. If this Access Point object does not command Access Door objects (e.g., muster point), or is used to control access to other resources or functions, or commands other objects, then this array shall be empty.

12.X.27 Priority_For_Writing

This property, of type Unsigned (1..16), defines the priority at which the referenced Access Door objects Present_Value properties are commanded. It corresponds to the 'Priority' parameter of the WriteProperty service. The value 1 is considered the highest priority and 16 the lowest. See Clause 19.2.

12.X.28 Muster_Point

This optional property, of type BOOLEAN, indicates whether this Access Point generates MUSTER access events (TRUE) or not (FALSE).

12.X.29 Zone_To

This optional property, of type BACnetDeviceObjectReference, shall specify the Access Zone object for which this Access Point object is an ingress access controlled point, allowing entrance to the zone. This property shall not reference the same Access Zone object as the Zone_From property. If the Access Point is not an ingress point to an access controlled zone, then this property shall contain 4194303 in the instance part of the object identifier.

12.X.30 Zone_From

This optional property, of type BACnetDeviceObjectReference, shall specify the Access Zone object for which this Access Point object is an egress access controlled point, allowing exit from the zone. This property shall not reference the same Access Zone object as the Zone_To property. If the Access Point is not an egress point from an access controlled zone, then this property shall contain 4194303 in the instance part of the object identifier.

12.X.31 Notification_Class

This optional property, of type Unsigned, shall specify the notification class to be used when generating Access Alarm Event notifications for this object. The Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. This property is required if intrinsic reporting is supported by this object.

12.X.32 Transaction_Notification_Class

This optional property, of type Unsigned, shall specify the notification class of Access Transaction Events. The Transaction_Notification_Class property implicitly refers to a Notification Class object that has a Notification_Class property with the same value. If this property is not present, then the Notification Class specified by the property Notification_Class shall be used for Access Transaction Events.

12.X.33 Access_Alarm_Events

This optional property, of type List of BACnetAccessEvent, shall specify any access events which are reported as Access Alarm Events. This property is required if intrinsic reporting is supported by this object.

An Access Alarm Event is reported when the following conditions are true:

- (a) The Last_Access_Event is updated and the updated value is equal to one of the values in Access_Alarm_Events, and
- (b) the TO-NORMAL flag is enabled in the Event_Enable property.

The notification is sent with Notify Type as specified by the property Notify_Type.

The Notification Class object referenced by the Notification_Class property is used to report Access Alarm Events.

12.X.34 Access_Transaction_Events

This optional property, of type List of BACnetAccessEvent, shall specify any access events that are reported as Access Transaction Events. This property is required if intrinsic reporting is supported by this object.

An Access Transaction Event is reported when the following conditions are true:

- (a) The Last_Access_Event is updated and the updated value is equal to one of the values in Access_Transaction_Events, and
- (b) the TO-NORMAL flag is set in the Event_Enable property.

The value of Notify_Type is ignored and the notification is sent with a Notify Type of EVENT. The Event_Time_Stamps TO-NORMAL element is not affected. The Acked_Transitions TO_NORMAL bit is not affected.

The Notification Class object referenced by the Transaction_Notification_Class property is used to report Access Transaction Events. If Transaction_Notification_Class is not present, the Notification Class object referenced by Notification_Class is used. The Ack_Required property of the respective Notification Class object is ignored and the value FALSE is conveyed in the AckRequired parameter of the event notification message.

12.X.35 Event_Enable

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT and TO-NORMAL events.

This property is required if intrinsic reporting is supported by this object.

12.X.36 Acked_Transitions

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

- (a) upon receipt of the corresponding acknowledgment;
- (b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgment is expected);
- (c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgment is expected).

This property is required if intrinsic reporting is supported by this object.

12.X.37 Notify_Type

This optional property, of type BACnetNotifyType, shall convey whether the Access Alarm Event notifications generated by the object should be Events or Alarms. This property is required if intrinsic reporting is supported by this object.

Access Transaction Events generated by the object are always of type EVENT, regardless of the value of this property.

12.X.38 Event_Time_Stamps

This optional property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have 'FF' in each octet, and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if intrinsic reporting is supported by this object.

12.X.39 Profile_Name

This optional 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Change **Table 13-1**, p.254]

Table 13-1. Standardized Objects That May Support COV Reporting

Object Type	Criteria	Properties Reported
...
<i>Access Point</i>	<i>If Update_Time changes at all</i>	<i>Last_Access_Event, Status_Flags, Update_Time, Last_Access_Credential, Last_Authentication_Factor (if present)</i>
...		

[Add new clause **19.2.7**, p.365]

19.2.7 Prioritization for Access Point Objects

Access Point objects interact with the Present_Value property of Access Door objects; however, if the Access Door objects are local to the device, they will not use BACnet services to do so. Each Access Point object has a Priority_For_Writing property that designates the priority to be used to command the Access Door objects.

[Add new **BACnetAccessEvent** production to Clause **21**, p. 408]

```

BACnetAccessEvent ::= ENUMERATED {
    none (0),
    muster (1),
    passback-detected (2),
    duress (3),
    trace (4),
    lockdown-max-attempts (5),
    lockdown-other (6),
    lockdown-relinquished (7),
    locked-by-higher-priority (8),
    granted (9),
    denied-deny-all (10),
    denied-unknown-credential (11),
    denied-missing-authentication-factor (12),
    denied-zone-no-access-rights (13),
    denied-point-no-access-rights (14),
    denied-out-of-time-range (15),
    denied-threat-level (16),
    denied-passback (17),
    denied-unexpected-location-usage (18),
    denied-max-attempts (19),
    denied-low-occupancy-threshold (20),
    denied-high-occupancy-threshold (21),
    denied-credential-unassigned (22),
    denied-credential-invalid (23),
    denied-credential-not-yet-active (24),
    denied-credential-expired (25),
    denied-credential-manual-disable (26),
    denied-credential-locked-out (27),
    denied-credential-exceeded-use (28),
    denied-credential-inactivity (29),
    denied-credential-missing (30),
    denied-credential-damaged (31),
    denied-credential-destroyed (32),

```

denied-credential-disabled (33),
denied-credential-timeout (34),
denied-incorrect-credential (35),
denied-no-accompaniment (36),
denied-incorrect-accompaniment (37),
denied-lockdown (38),
denied-other (39),

...
}

-- Enumerated values 0-511 are reserved for definition by ASHRAE. Enumerated values
-- 512-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

[Add new **BACnetAccessThreatLevel** production to Clause 21, p. 408]

BACnetAccessThreatLevel ::= Unsigned(0..100)

[Note: **BACnetAuthenticationFactor** is defined in Addendum 135-2004j-6.]

[Note: **BACnetAuthenticationFactorType** is defined in Addendum 135-2004j-6.]

[Add new **BACnetAuthenticationPolicy** production to Clause 21, p. 408]

BACnetAuthenticationPolicy ::= SEQUENCE OF BACnetAuthenticationRule

[Add new **BACnetAuthenticationRule** production to Clause 21, p. 408]

BACnetAuthenticationRule ::= SEQUENCE {
 authenticationFactorRequirement [0] ENUMERATED {
 required (0),
 choice-begin (1),
 choice-end (2),
 option (3)
 },
 authenticationFactorInput [1] BACnetDeviceObjectPropertyReference OPTIONAL
}

[Add new **BACnetAuthorizationMode** production to Clause 21, p. 408]

BACnetAuthorizationMode ::= ENUMERATED {

 authorize (0),
 grant-active (1),
 deny-all (2),

 ...
}

-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.

[Change **BACnetObjectType** production in Clause 21, pp. 421-422]

[Note: enumeration values 25-31 are defined in Addendum 135-2004b-1, d-1, e-1, f-1 and i-1.]

```
BACnetObjectType ::= ENUMERATED {  
    access-credential           (32),  
    access-point               (33),  
    access-rights              (34),  
    access-user                (35),  
    access-zone                (36),  
    accumulator               (23),  
    authentication-factor-input (37),  
    ...  
    -- see pulse-converter      (24),  
    -- see access-credential    (32),  
    -- see access-point        (33),  
    -- see access-rights       (34),  
    -- see access-user         (35),  
    -- see access-zone         (36),  
    -- see authentication-factor-input (37),  
    ...  
}
```

[Change **BACnetObjectTypesSupported** production in Clause 21, p. 422]

[Note: bit positions 25-31 are defined in Addendum 135-2004b-1, d-1, e-1, f-1 and i-1.]

```
BACnetObjectTypesSupported ::= BIT STRING {  
    -- access-credential       (32),  
    -- access-point           (33),  
    -- access-rights          (34),  
    -- access-user            (35),  
    -- access-zone            (36),  
    -- accumulator           (23),  
    -- authentication-factor-input (37),  
    ...  
    (24) pulse-converter      (24),  
    access-credential       (32),  
    access-point           (33),  
    access-rights          (34),  
    access-user            (35),  
    access-zone            (36),  
    authentication-factor-input (37)  
}
```

[Change **BACnetPropertyIdentifier** production, Clause 21, pp. 423-428]

[Note: the renaming of the "log-enable" property identifier to "enable" also occurs in Addendum 135-2004b-2.]

```
BACnetPropertyIdentifier ::= ENUMERATED {  
    absentee-limit           (244),  
    accepted-modes         (175),  
    access-alarm-events     (245),  
    access-doors           (246),  
    access-rights-enable    (247),  
    access-rules           (248),  
    access-rules-enable    (249),  
    access-transaction-events (250),  
    accompanied           (251),
```

<i>activation-time</i>	(252),
<i>active-authentication-policy</i>	(253),
acked-transitions	(0),
...	
archive	(13),
<i>assigned-access-rights</i>	(254),
attempted-samples	(124),
<i>authentication-factors</i>	(255),
<i>authentication-policy-list</i>	(256),
<i>authentication-policy-names</i>	(257),
<i>authentication-policy-timeouts</i>	(258),
<i>authorization-mode</i>	(259),
auto-slave-discovery	(169),
...	
backup-failure-timeout	(153),
<i>belongs-to</i>	(260),
bias	(14),
...	
cov-resubscription-interval	(128),
<i>credential-disable</i>	(261),
<i>credential-status</i>	(262),
<i>credentials</i>	(263),
<i>credentials-in-zone</i>	(264),
-- current-notify-time	(129), This property was deleted in version 1 revision 3.
...	
daylight-savings-status	(24),
<i>days-remaining</i>	(265),
deadband	(25),
...	
effective-period	(32),
<i>egress-points</i>	(266),
elapsed-active-time	(33),
<i>enable</i>	(133), --renamed from previous version
error-limit	(34),
...	
exception-schedule	(38),
<i>expiry-time</i>	(267),
<i>extended-time-enable</i>	(268),
<i>failed-attempts</i>	(269),
<i>failed-attempts-time</i>	(270),
fault-values	(39),
...	
firmware-revision	(44),
<i>format-class-supported</i>	(271),
<i>format-type</i>	(272),
high-limit	(45),
<i>ingress-points</i>	(273),
inactive-text	(46),
...	
in-process	(47),
<i>inherited-access-rights</i>	(274),
input-reference	(181),
...	
-- issue-confirmed-notifications	(51), This property was deleted in version 1 revision 4.
<i>last-access-credential</i>	(275),
<i>last-access-event</i>	(276),

<i>last-access-point</i>	(277),
<i>last-authentication-factor</i>	(278),
<i>last-credential-added</i>	(279),
<i>last-credential-removed</i>	(280),
last-notify-record	(173),
last-restore-time	(157),
<i>last-use-time</i>	(281),
life-safety-alarm-values	(166),
...	
location	(58),
<i>lockdown</i>	(282),
<i>lockdown-relinquish-time</i>	(283),
log-buffer	(131),
...	
log-enable	(133),
...	
max-apdu-length-accepted	(62),
<i>max-failed-attempts</i>	(284),
max-info-frames	(63),
...	
max-segments-accepted	(167),
<i>members</i>	(285),
member-of	(159),
...	
modification-date	(71),
<i>muster-point</i>	(286),
notification-class	(17), -- renamed from previous version
...	
object-type	(79),
<i>occupancy-count</i>	(287),
<i>occupancy-count-enable</i>	(288),
<i>occupancy-count-exemption</i>	(289),
<i>occupancy-lower-threshold</i>	(290),
<i>occupancy-lower-threshold-enforced</i>	(291),
<i>occupancy-state</i>	(292),
<i>occupancy-upper-threshold</i>	(293),
<i>occupancy-upper-threshold-enforced</i>	(294),
operation-expected	(161),
...	
output-units	(82),
-- see event-parameters	(83),
<i>passback-exemption</i>	(295),
<i>passback-mode</i>	(296),
<i>passback-timeout</i>	(297),
polarity	(84),
...	
read-only	(99),
<i>read-status</i>	(298),
reason-for-halt	(100),
<i>reason-for-disable</i>	(299),
-- recipient	(101),
...	
system-status	(112),
<i>threat-authority</i>	(300)
<i>threat-level</i>	(301),
time-delay	(113),

...	
total-record-count	(145),
<i>trace-flag</i>	(302),
tracking-value	(164),
<i>transaction-notification-class</i>	(303),
units	(117),
...	
update-time	(189),
<i>user-external-identifier</i>	(304),
<i>user-information-reference</i>	(305),
<i>user-name</i>	(306),
<i>user-type</i>	(307),
<i>uses-remaining</i>	(308),
utc-offset	(119),
...	
variance-value	(151),
<i>vendor-format-identifier</i>	(309),
vendor-identifier	(120),
...	
weekly-schedule	(123),
<i>zone-from</i>	(310),
<i>zone-to</i>	(311),
-- see attempted-samples	(124),
...	
-- see value-change-time	(192),
-- see <i>absentee-limit</i>	(244),
-- see <i>access-alarm-events</i>	(245),
-- see <i>access-doors</i>	(246),
-- see <i>access-rights-enable</i>	(247),
-- see <i>access-rules</i>	(248)
-- see <i>access-rules-enable</i>	(249)
-- see <i>access-transaction-events</i>	(250),
-- see <i>accompanied</i>	(251),
-- see <i>activation-time</i>	(252),
-- see <i>active-authentication-policy</i>	(253),
-- see <i>assigned-access-rights</i>	(254),
-- see <i>authentication-factors</i>	(255),
-- see <i>authentication-policy-list</i>	(256),
-- see <i>authentication-policy-names</i>	(257),
-- see <i>authentication-policy-timeouts</i>	(258),
-- see <i>authorization-mode</i>	(259),
-- see <i>belongs-to</i>	(260),
-- see <i>credential-disable</i>	(261),
-- see <i>credential-status</i>	(262),
-- see <i>credentials</i>	(263),
-- see <i>credentials-in-zone</i>	(264),
-- see <i>days-remaining</i>	(265),
-- see <i>egress-points</i>	(266),
-- see <i>expiry-time</i>	(267),
-- see <i>extended-time-enable</i>	(268),
-- see <i>failed-attempts</i>	(269),
-- see <i>failed-attempts-time</i>	(270),
-- see <i>format-class-supported</i>	(271),
-- see <i>format-type</i>	(272),
-- see <i>ingress-points</i>	(273),
-- see <i>inherited-access-rights</i>	(274),

-- see *last-access-credential* (275),
-- see *last-access-event* (276),
-- see *last-access-point* (277),
-- see *last-authentication-factor* (278),
-- see *last-credential-added* (279),
-- see *last-credential-removed* (280),
-- see *last-use-time* (281),
-- see *lockdown* (282),
-- see *lockdown-relinquish-time* (283),
-- see *max-failed-attempts* (284),
-- see *members* (285)
-- see *muster-point* (286),
-- see *occupancy-count* (287),
-- see *occupancy-count-enable* (288),
-- see *occupancy-count-exemption* (289),
-- see *occupancy-lower-threshold* (290),
-- see *occupancy-lower-threshold-enforced* (291),
-- see *occupancy-state* (292),
-- see *occupancy-upper-threshold* (293),
-- see *occupancy-upper-threshold-enforced* (294),
-- see *passback-exemption* (295),
-- see *passback-mode* (296),
-- see *passback-timeout* (297),
-- see *read-status* (298),
-- see *reason-for-disable* (299),
-- see *threat-authority* (300),
-- see *threat-level* (301),
-- see *trace-flag* (302),
-- see *transaction-notification-class* (303),
-- see *user-external-identifier* (304),
-- see *user-information-reference* (305),
-- see *user-name* (306),
-- see *user-type* (307),
-- see *uses-remaining* (308),
-- see *vendor-format-identifier* (309),
-- see *zone-from* (310),
-- see *zone-to* (311),
...
}

[Change **Table 23-1**, p.437]

Table 23-1. Extensible Enumerations

Enumeration Name	Reserved Range	Maximum Value
error-class	0-63	65535
error-code	0-255	65535
BACnetAbortReason	0-63	255
<i>BACnetAccessCredentialDisable</i>	<i>0-255</i>	<i>65535</i>
<i>BACnetAccessCredentialDisableReason</i>	<i>0-255</i>	<i>65535</i>
<i>BACnetAccessEvent</i>	<i>0-511</i>	<i>65535</i>
<i>BACnetAccessUserType</i>	<i>0-63</i>	<i>65535</i>
<i>BACnetAccessZoneOccupancyState</i>	<i>0-63</i>	<i>65535</i>
<i>BACnetAuthorizationMode</i>	<i>0-63</i>	<i>65535</i>
BACnetDeviceStatus	0-63	65535
...
BACnetVTClass	0-63	65535

[Add to **Annex C**, p.453]

```

ACCESS-POINT ::= SEQUENCE {
    object-identifier          [75]  BACnetObjectIdentifier,
    object-name                [77]  CharacterString,
    object-type                [79]  BACnetObjectType,
    description                [28]  CharacterString OPTIONAL,
    status-flags               [111] BACnetStatusFlags,
    event-state                [36]  BACnetEventState,
    reliability                 [103] BACnetReliability OPTIONAL,
    out-of-service             [81]  BOOLEAN,
    authentication-policy-names [257] SEQUENCE OF CharacterString OPTIONAL,
                                -- accessed as a BACnetARRAY
    authentication-policy-list [256] SEQUENCE OF BACnetAuthenticationPolicy OPTIONAL,
                                -- accessed as a BACnetARRAY
    authentication-policy-timeouts [258] SEQUENCE OF Unsigned OPTIONAL,
                                -- accessed as a BACnetARRAY
    active-authentication-policy [253] Unsigned OPTIONAL,
    authorization-mode         [259] BACnetAuthorizationMode OPTIONAL,
    lockdown                   [282] BOOLEAN OPTIONAL,
    lockdown-relinquish-time   [283] Unsigned OPTIONAL,
    failed-attempts            [269] Unsigned OPTIONAL,
    max-failed-attempts        [284] Unsigned OPTIONAL,
    failed-attempts-time       [270] Unsigned OPTIONAL,
    threat-level               [301] BACnetAccessThreatLevel OPTIONAL,
    occupancy-upper-threshold-enforced [294] BOOLEAN OPTIONAL,
    occupancy-lower-threshold-enforced [291] BOOLEAN OPTIONAL,
    last-access-event          [276] BACnetAccessEvent OPTIONAL,
    update-time                [189] BACnetTimeStamp OPTIONAL,
    last-access-credential     [275] BACnetObjectIdentifier OPTIONAL,
    last-authentication-factor  [278] BACnetAuthenticationFactor OPTIONAL,
    access-doors               [246] SEQUENCE OF BACnetDeviceObjectReference,
                                -- accessed as a BACnetARRAY
    priority-for-writing       [88]  Unsigned(1..16),
    muster-point              [286] BOOLEAN OPTIONAL,
    zone-to                   [311] BACnetDeviceObjectReference OPTIONAL,
    zone-from                 [310] BACnetDeviceObjectReference OPTIONAL,
    notification-class        [17]  Unsigned OPTIONAL,
    transaction-notification-class [303] Unsigned OPTIONAL,

```

```

access-alarm-events           [245] SEQUENCE OF BACnetAccessEvent OPTIONAL,
access- transaction -events  [250] SEQUENCE OF BACnetAccessEvent OPTIONAL,
event-enable                  [35]  BACnetEventTransitionBits  OPTIONAL,
acked-transitions             [0]   BACnetEventTransitionBits  OPTIONAL,
notify-type                   [72]  BACnetNotifyType  OPTIONAL,
event-time-stamps            [130] SEQUENCE OF BACnetTimeStamp OPTIONAL,
-- accessed as a BACnetARRAY
profile-name                   [167] CharacterString  OPTIONAL
}

```

[Add new ANNEX D.X, p.484]

D.X Example of an Access Point object

In this example, authentication and authorization at an entrance to a secured zone is represented as an Access Point object “Main Entrance 1”. The secured zone to enter is assumed to be represented by an Access Zone object instance 23.

There are three Authentication Factor Input objects used, all located in remote Device 12:

- (a) Proximity Card Reader – Instance 3
- (b) PIN Keypad – Instance 4
- (c) Iris Scanner – Instance 5

Three different authentication policies are defined and can be activated:

- (a) OFFICE HOURS – Low Security – Proximity Card Reader or PIN Keypad
- (b) LATE SHIFT – Medium Security Proximity Card Reader and PIN Keypad
- (c) NIGHT – High Security – Proximity Card Reader and Iris Scanner

The current authorization mode is AUTHORIZE, authentication and authorization is in effect. The current authentication policy is set to “OFFICE-HOURS”.

The Access Point is currently not locked down, no failed-attempts have occurred.

The current threat level is 20.

Access Door objects 44 and 45, local in the device, are controlled at a command priority 12.

Occupancy lower thresholds are not checked but upper thresholds are checked if present at the Access Zone.

The last event at the Access Point happened 23-MAR-07, 18:50:21.2, and was a passback violation done with the Access Credential object instance 41445 using the simple number authentication factor 1334234499 of the credential class 2, and is not yet acknowledged.

```

Property:  Object_Identifier = (Access Point, Instance 2)
Property:  Object_Name = "MAIN-ENTRANCE-01"
Property:  Object_Type = ACCESS_POINT
Property:  Description = "Main Entrance 1"
Property:  Status_Flags = {FALSE, FALSE, FALSE, FALSE}
Property:  Event_State = (NORMAL)
Property:  Reliability = NO_FAULT_DETECTED
Property:  Out_Of_Service = FALSE
Property:  Authentication_Policy_Names = ("OFFICE_HOURS", "LATE_SHIFT", "NIGHT")
Property:  Authentication_Policy_List = ( ( (CHOICE_BEGIN),
                                         (OPTION, ( (Device, Instance 12),
                                                    (Authentication Factor Input, Instance 3),
                                                    Present_Value ) ),
                                         (OPTION, ( (Device, Instance 12),

```

```

( ( ( ( (Authentication Factor Input, Instance 4),
      Present_Value ) ),
    (CHOICE_END) ),
( (REQUIRED, ((Device, Instance 12),
              (Authentication Factor Input, Instance 3),
              Present_Value ) ),
  (REQUIRED, ((Device, Instance 12)
              (Authentication Factor Input, Instance 4),
              Present_Value ) ) ),
( (REQUIRED, ((Device, Instance 12),
              (Authentication Factor Input, Instance 3),
              Present_Value ) ),
  (REQUIRED, ((Device, Instance 12),
              (Authentication Factor Input, Instance 5),
              Present_Value ) ) ) )

Property: Authentication_Policy_Timeouts = (20, 20, 10)
Property: Active_Authentication_Policy = 1
Property: Authorization_Mode = AUTHORIZE
Property: Lockdown = FALSE
Property: Lockdown_relinquish Time = 60
Property: Failed_Attempts = 0
Property: Max_Failed_Attempts = 3
Property: Failed_Attempts_Time = 10
Property: Threat_Level = 20
Property: Occupancy_Upper_Threshold_Enforced = TRUE
Property: Occupancy_Lower_Threshold_Enforced = FALSE
Property: Last_Access_Event = PASSBACK_DETECTED
Property: Update_Time = ((23 MAR 2007, FRIDAY), 18:50:21.2)
Property: Last_Access_Credential = (Access Credential, Instance 41445)
Property: Last_Authentication_Factor = (2, 1334234499)
Property: Access_Doors = ((Access Door, Instance 44),
                          (Access Door, Instance 45))

Property: Priority_For_Writing = 12
Property: Muster_Point = FALSE
Property: Zone_To = (Access Zone, Instance 23)
Property: Zone_From = (Access Zone, Instance 1)
Property: Notification_Class = 39
Property: Transaction_Notification_Class = 48
Property: Access_Alarm_Events = (DURESS, PASSBACK_DETECTED)
Property: Access_Transaction_Events = (GRANTED, DENIED_PASSBACK,
                                       DENIED_UNKNOWN_CREDENTIAL)

Property: Event_Enable = {TRUE, TRUE, TRUE}
Property: Acked_Transitions = {TRUE, TRUE, FALSE}
Property: Notify_Type = EVENT
Property: Event_Time_Stamps = ((*-*-*, *:*:*.*),
                              (*-*-*, *:*:*.*),
                              ((23 MAR 2007, FRIDAY), 18:50:21.3))

```

135-2004j-2. Add a new Access Zone object type.

Rationale

Support for access control in BACnet requires a standardized object whose properties represent the externally visible characteristics associated with a secured geographical zone for which authentication and authorization of a credential takes place to obtain physical access.

Addendum 135-2004j-2

[Insert new clause **12.X** and Table **12-X**, p. 251]

12.X Access Zone Object Type

The Access Zone object type defines a standardized object whose properties represent the externally visible characteristics associated with a secured geographical zone for which authentication and authorization of a credential takes place to obtain physical access. Entrance to the zone takes place through *ingress* access controlled points while the zone is exited through *egress* access controlled points. These access controlled points are represented by Access Point objects.

The Access Zone object supports occupancy counting and the specification of occupancy thresholds. Access may be denied if thresholds are violated. The enforcement rules are specified at the corresponding ingress and/or egress Access Point objects. The Access Zone object's Occupancy_State is the state of occupancy. This state is derived from the actual occupancy count and occupancy thresholds. If occupancy count is not exposed, the evaluation of the Occupancy_State is a local matter.

Intrinsic reporting of this object is based on the Occupancy_State property and uses the CHANGE_OF_STATE algorithm.

"Who's in" reporting is supported through a list of the credentials which are currently in the zone. Credentials are added on successful entrance through an access controlled ingress point and removed on successful exit through an access controlled egress point. This list may also be maintained based on time conditions or other local methods.

The Access Zone object supports passback detection and allows the selection of hard, soft or no-passback enforcement. A passback violation occurs when entrance to this zone is requested at an access controlled point while the credential is assumed to be in this zone. The list of credentials in this zone may be used to detect a passback violation.

In a physical access control system each instance of this object may be replicated over multiple devices. In this case, objects representing the same credential shall have the same instance number in each device; however, it is a local matter as to how the objects are synchronized.

The Access Zone object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Access Zone Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Occupancy_State	BACnetAccessZoneOccupancyState	R
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O ¹
Out_Of_Service	BOOLEAN	R
Occupancy_Count	Unsigned	R ¹
Occupancy_Count_Enable	BOOLEAN	R
Adjust_Value	INTEGER	O
Occupancy_Upper_Threshold	Unsigned	O
Occupancy_Lower_Threshold	Unsigned	O
Credentials_In_Zone	List of BACnetObjectIdentifier	O
Last_Credential_Added	BACnetObjectIdentifier	O
Last_Credential_Removed	BACnetObjectIdentifier	O
Passback_Mode	BACnetAccessPassbackMode	O
Passback_Timeout	Unsigned	O ²
Ingress_Points	List of BACnetDeviceObjectReference	R
Egress_Points	List of BACnetDeviceObjectReference	R
Time_Delay	Unsigned	O ³
Notification_Class	Unsigned	O ³
Alarm_Values	List of BACnetAccessZoneOccupancyState	O ³
Event_Enable	BACnetEventTransitionBits	O ³
Acked_Transitions	BACnetEventTransitionBits	O ³
Notify_Type	BACnetNotifyType	O ³
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O ³
Profile_Name	CharacterString	O

¹ These properties, if present, shall be writeable when Out_Of_Service is TRUE.

² If this property is present, then Passback_Mode shall be present.

³ These properties are required 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 ACCESS_ZONE.

12.X.4 Description

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

12.X.5 Occupancy_State

This property, of type `BACnetAccessZoneOccupancyState`, reflects the occupancy state of the zone.

`BACnetAccessZoneOccupancyState` is an enumeration of possible occupancy states.

<code>NORMAL</code>	This is the default occupancy state when no other standard or proprietary states are applicable or when occupancy counting is disabled.
<code>BELOW_LOWER_THRESHOLD</code>	If <code>Occupancy_Lower_Threshold</code> property is present and the <code>Occupancy_Count</code> property is lower than this value.
<code>AT_LOWER_THRESHOLD</code>	If <code>Occupancy_Lower_Threshold</code> property is present and the <code>Occupancy_Count</code> property is equal to this value.
<code>AT_UPPER_THRESHOLD</code>	If <code>Occupancy_Upper_Threshold</code> property is present and the <code>Occupancy_Count</code> property is equal to this value.
<code>ABOVE_UPPER_THRESHOLD</code>	If <code>Occupancy_Upper_Threshold</code> property is present and the <code>Occupancy_Count</code> property is greater than this value.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to indicate other states based on the <code>Occupancy_Count</code> property other than those defined by this standard. For proprietary extensions of this enumeration see clause 23.1 of this standard.

12.X.6 Status_Flags

This property, of type `BACnetStatusFlags`, represents four Boolean flags that indicate the general "health" of the Access Zone 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:

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

12.X.7 Event_State

The `Event_State` property, of type `BACnetEventState`, is included in order to provide a way to determine if this object has an active event state associated with it. If the object supports intrinsic reporting, then the `Event_State` property shall indicate the event state of the object. If the object does not support intrinsic reporting then:

- (a) if the `Reliability` property is not present, then the value of `Event_State` shall be `NORMAL`, or
- (b) if the `Reliability` property is present and `Reliability` is `NO_FAULT_DETECTED`, then `Event_State` shall be `NORMAL`, or

- (c) if the Reliability property is present and Reliability is not NO_FAULT_DETECTED, then Event_State shall be FAULT.

12.X.8 Reliability

The optional Reliability property, of type BACnetReliability, provides an indication of whether the Occupancy_State, Occupancy_Count and/or Credentials_In_Zone properties of this object are "reliable" as far as the BACnet Device can determine and, if not, why. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, UNRELIABLE_OTHER}.

The Reliability property shall be writable when Out_Of_Service is TRUE.

12.X.8.1 Conditions for Generating a TO-FAULT Event

A TO-FAULT event is generated under these conditions:

- (a) the Reliability property becomes not equal to NO_FAULT_DETECTED, and
- (b) the TO-FAULT flag is enabled in the Event_Enable property.

12.X.9 Out_Of_Service

The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Occupancy_Count property is decoupled from the processing of occupancy counting. In addition, the Reliability property and the corresponding state of the FAULT flag of the Status_Flags property shall be decoupled when Out_Of_Service is TRUE. While the Out_Of_Service property is TRUE, writing to the Adjust_Value property shall not modify the Occupancy_Count or the Occupancy_Count, and Reliability properties may be changed to any value as a means of simulating specific fixed conditions or for testing purposes. Other functions that depend on the state of the Occupancy_State or Reliability properties shall respond to changes made to these properties while Out_Of_Service is TRUE.

12.X.10 Occupancy_Count

This property, of type Unsigned, is used to indicate the actual occupancy count of a zone. If the value of the Occupancy_Count_Enable property is FALSE, then this property shall have a value of zero. The value of the Occupancy_Count property may be adjusted by writing to the Adjust_Value property. The Occupancy_Count property shall be writable when Out_Of_Service is TRUE. When Out_Of_Service becomes FALSE, it is a local matter as to what value the occupancy count is set to.

12.X.11 Occupancy_Count_Enable

This property, of type BOOLEAN, indicates whether occupancy counting is in effect (TRUE) or not (FALSE).

12.X.12 Adjust_Value

This optional property, of type INTEGER, when written, shall adjust the Occupancy_Count property.

The following series of operations shall be performed atomically when this property is written and the value of the Occupancy_Count_Enable property is TRUE:

- (1) The value written to Adjust_Value shall be stored in the Adjust_Value property.
- (2) If the value written is non-zero, then this value shall be added to the value of the Occupancy_Count property. If the value written is zero, then the value of the Occupancy_Count property shall be set to zero. If the resulting value of the Occupancy_Count property would be less than zero, then the Occupancy_Count property shall be set to zero.
- (3) The Adjust_Value property shall be set to zero.

When this property is written and the value of the `Occupancy_Count_Enable` property is `FALSE`, then the `Adjust_Value` property shall be set to zero.

If `Adjust_Value` has never been written, it shall have a value of zero.

12.X.13 Occupancy_Upper_Threshold

This optional property, of type `Unsigned`, specifies the occupancy upper threshold of the zone. If this property has a value of zero, then there is no upper threshold. If this value is not zero, it shall be greater than the value of the `Occupancy_Lower_Threshold`, if present.

12.X.14 Occupancy_Lower_Threshold

This optional property, of type `Unsigned`, specifies the occupancy lower threshold of the zone. If this property has a value of zero, then there is no lower threshold.

12.X.15 Credentials_In_Zone

This optional property, of type `List of BACnetObjectIdentifier`, is used to list references to those `Access Credential` objects that represent credentials assumed to be in this zone. This information may be used to verify whether a specific credential is already in the zone for passback detection purposes. If the zone does not support listing credentials this list may be empty. It is a local matter as to how this list is updated.

12.X.16 Last_Credential_Added

This optional property, of type `BACnetObjectIdentifier`, indicates the reference to the `Access Credential` object which has last been added to the `Credentials_In_Zone` property. If no credential has been added yet, the instance part of the object identifier holds a value of 4194303. If `COV` property subscriptions for this property are present, then any update, even one with the same value, is reported by a `COV` notification.

12.X.17 Last_Credential_Removed

This optional property, of type `BACnetObjectIdentifier`, indicates the reference to the `Access Credential` object which has last been removed from the `Credentials_In_Zone` property. When no credential has been removed yet, the instance part of the object identifier holds a value of 4194303. If `COV` property subscriptions for this property are present, then any update, even one with the same value, is reported by a `COV` notification.

12.X.18 Passback_Mode

This optional property, of type `BACnetAccessPassbackMode`, specifies how all access controlled ingress points to this zone shall handle passback violations. Passback modes are:

<code>PASSBACK_OFF</code>	Passback violations are not checked.
<code>HARD_PASSBACK</code>	Passback violations are checked and enforced (i.e., denied access) and violations are reported by the access controlled ingress points.
<code>SOFT_PASSBACK</code>	Passback violations are checked but not enforced, but violations are reported by the access controlled ingress points.

12.X.19 Passback_Timeout

This optional property, of type `Unsigned`, specifies the passback timeout in minutes. The timeout is evaluated individually for every credential used to enter the zone. The timeout period for a particular credential begins at the time of successful access to the zone. After the timeout has expired for a particular credential, a passback violation of this credential will no longer be detected. A value of zero or absence of this property indicates passback violations will never time out.

If `Passback_Timeout` is present, `Passback_Mode` shall be present.

12.X.20 Ingress_Points

This property, of type `List of BACnetDeviceObjectReference`, references all Access Point objects that lead into the zone.

12.X.21 Egress_Points

This property, of type `List of BACnetDeviceObjectReference`, references all Access Point objects that lead out of the zone.

12.X.22 Time_Delay

This optional property, of type `Unsigned`, shall specify the minimum period of time in seconds that the `Occupancy_State` remains:

- (a) equal to any one of the values in the `Alarm_Values` property before a `TO-OFFNORMAL` event is generated, or
- (b) not equal to any of the values in the `Alarm_Values` property before a `TO-NORMAL` event is generated.

This property is required if intrinsic reporting is supported by this object.

12.X.23 Notification_Class

This optional property, of type `Unsigned`, shall specify the notification class to be used when handling and generating event notifications for this object. The `Notification_Class` property implicitly refers to a `Notification Class` object that has a `Notification_Class` property with the same value. This property is required if intrinsic reporting is supported by this object.

12.X.24 Alarm_Values

This optional property, of type `List of BACnetAccessZoneOccupancyState`, shall specify any states the `Occupancy_State` shall equal before a `TO-OFFNORMAL` event is generated. This property is required if intrinsic reporting is supported by this object.

12.X.24.1 Conditions for Generating a TO-OFFNORMAL Event

A `TO-OFFNORMAL` event is generated under these conditions:

- (a) the `Occupancy_State` equals at least one of the values in the `Alarm_Values` list, and
- (b) the `Occupancy_State` remains equal to the same value for a minimum period of time, specified by the `Time_Delay` property, and
- (c) the `TO-OFFNORMAL` flag is enabled in the `Event_Enable` property.

12.X.24.2 Conditions for Generating a TO-NORMAL Event

Once equal, if the `Occupancy_State` becomes not equal to any of the states in this property and not equal to any of the states in the `Fault_Values` property, then a `TO-NORMAL` event is generated under these conditions:

- (a) the `Occupancy_State` remains not equal to any of the states in the `Alarm_Values` property for a minimum period of time, specified by the `Time_Delay` property, and
- (b) the `Occupancy_State` remains not equal to any of the states in the `Fault_Values` property, and
- (c) the `TO-NORMAL` flag is enabled in the `Event_Enable` property.

12.X.25 Event_Enable

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately enable and disable reporting of TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. This property is required if intrinsic reporting is supported by this object.

12.X.26 Acked_Transitions

This optional property, of type BACnetEventTransitionBits, shall convey three flags that separately indicate the receipt of acknowledgments for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events. These flags shall be cleared upon the occurrence of the corresponding event and set under any of these conditions:

- (a) upon receipt of the corresponding acknowledgment;
- (b) upon the occurrence of the event if the corresponding flag is not set in the Event_Enable property (meaning event notifications will not be generated for this condition and thus no acknowledgment is expected);
- (c) upon the occurrence of the event if the corresponding flag is set in the Event_Enable property and the corresponding flag in the Ack_Required property of the Notification Class object implicitly referenced by the Notification_Class property of this object is not set (meaning no acknowledgment is expected).

This property is required if intrinsic reporting is supported by this object.

12.X.27 Notify_Type

This optional property, of type BACnetNotifyType, shall convey whether the notifications generated by the object are Events or Alarms. This property is required if intrinsic reporting is supported by this object.

12.X.28 Event_Time_Stamps

This optional property, of type BACnetARRAY[3] of BACnetTimeStamp, shall convey the times of the last event notifications for TO-OFFNORMAL, TO-FAULT, and TO-NORMAL events, respectively. Time stamps of type Time or Date shall have X'FF' in each octet and Sequence number time stamps shall have the value 0 if no event notification of that type has been generated since the object was created. This property is required if intrinsic reporting is supported by this object.

12.X.29 Profile_Name

This optional property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name begins 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Change 13.2, p. 255]

In the case of Life Safety Zone and Life Safety Point, the Life_Safety_Alarm_Values property lists each of the possible Present_Value states that shall be interpreted as LIFE_SAFETY_ALARM. The Alarm_Values property lists each of the possible Present_Value states that shall be interpreted as OFFNORMAL. The Fault_Values property lists each of the possible Present_Value states that shall be interpreted as FAULT. All other Present_Value states shall be interpreted as NORMAL. Transitions to any of the states may generate notifications if the corresponding flags are set in Event_Enable.

In the case of the Access Zone object, the Alarm_Values property lists each of the possible Occupancy_State states that shall be interpreted as OFFNORMAL. All other Occupancy_State states shall be interpreted as NORMAL. Transitions to any of the states may generate notifications if the corresponding flags are set in Event_Enable.

[Change **Table 13-2** and **Table 13-3**, p.256-257]

Table 13-2. Standard Objects That May Support Intrinsic Reporting

Object Type	Criteria	Event Type
...		
<i>Access Zone</i>	<i>If Occupancy_State changes to a new state for longer than Time_Delay AND the new transition is enabled in Event_Enable</i>	<i>CHANGE_OF_STATE</i>
...		

Table 13-3. Standard Object Property Values Returned in Notifications

Object	Event Type	Notification Parameters	Referenced Object's Properties
...			
<i>Access Zone</i>	<i>CHANGE_OF_STATE</i>	<i>New_State Status_Flags</i>	<i>Occupancy_State Status_Flags</i>
....			

[Add new **BACnetAccessPassbackMode** production to Clause **21**, p. 408]

```
BACnetAccessPassbackMode ::= ENUMERATED {
    passback-off      (0),
    hard-passback    (1),
    soft-passback     (2)
}
```

[Add new **BACnetAccessZoneOccupancyMode** production to Clause **21**, p. 408]

```
BACnetAccessZoneOccupancyState ::= ENUMERATED {
    normal              (0),
    below-lower-threshold (1),
    at-lower-threshold  (2),
    at-upper-threshold  (3),
    above-upper-threshold (4),
    ...
}
-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

[Note: changes to the **BACnetObjectType** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectTypesSupported** BITSTRING appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetPropertyIdentifier** enumeration appear in Addendum 135-2004j-1.]

[Change **Clause 21, BACnetPropertyStates** production, pp. 428-429]

[Note: other additions to this production appear in Addendum 135-2004b-5, f-1 and h-3.]

```

BACnetPropertyStates ::= CHOICE {
    ...
    occupancy-state      [292] BACnetAccessZoneOccupancyState,
    ...
}

```

[Note: changes to **Table 23-1 Extensible Enumerations** appear in Addendum 135-2004j-1.]

[Add new object type structure to **ANNEX C**, p.453]

```

ACCESS-ZONE ::= SEQUENCE {
    object-identifier      [75]   BACnetObjectIdentifier,
    object-name           [77]   CharacterString,
    object-type           [79]   BACnetObjectType,
    description           [28]   CharacterString OPTIONAL,
    occupancy-state       [292]  BACnetAccessZoneOccupancyState
    status-flags          [111]  BACnetStatusFlags,
    event-state           [36]   BACnetEventState,
    reliability           [103]  BACnetReliability OPTIONAL,
    out-of-service        [81]   BOOLEAN,
    occupancy-count       [287]  Unsigned,
    occupancy-count -enable [288] BOOLEAN,
    adjust-value          [176]  INTEGER OPTIONAL,
    occupancy-upper-threshold [293] Unsigned OPTIONAL,
    occupancy-lower-threshold [290] Unsigned OPTIONAL,
    credentials-in-zone   [264]  SEQUENCE OF BACnetObjectIdentifier OPTIONAL,
    last-credential-added [279]  BACnetObjectIdentifier OPTIONAL,
    last-credential-removed [280] BACnetObjectIdentifier OPTIONAL,
    passback-mode         [296]  BACnetAccessPassbackMode OPTIONAL,
    passback-timeout      [297]  Unsigned OPTIONAL,
    ingress-points        [273]  SEQUENCE OF BACnetDeviceObjectReference
    egress-points         [266]  SEQUENCE OF BACnetDeviceObjectReference
    time-delay            [113]  Unsigned OPTIONAL,
    notification-class    [17]   Unsigned OPTIONAL,
    alarm-values          [7]    SEQUENCE OF BACnetAccessZoneOccupancyState OPTIONAL,
    event-enable          [35]   BACnetEventTransitionBits OPTIONAL,
    acked-transitions     [0]    BACnetEventTransitionBits OPTIONAL,
    notify-type           [72]   BACnetNotifyType OPTIONAL,
    event-time-stamps     [130]  SEQUENCE OF BACnetTimeStamp OPTIONAL,
    -- accessed as a BACnetARRAY
    profile-name          [167]  CharacterString OPTIONAL,
}

```

[Add new **ANNEX D.X**, p.484]

D.X Example of an Access Zone object

In this example, a secured zone is represented as an Access Zone object. This Access Zone has an upper occupancy threshold of 100 access credentials to be inside, and reports an alarm if the count is at or above the upper threshold. Occupancy_State is NORMAL because two access credentials are inside. Passback detection is switched off.

The ingress Access Points (Local Device, Instance 2 and 3) are leading into the zone and the egress Access Points (Local Device, Instance 12 and 13) are leading out of the Access Zone.

Property: Object_Identifier = (Access Zone, Instance 23)
 Property: Object_Name = "OFFICE_MAIN_FLOOR"
 Property: Object_Type = ACCESS_ZONE
 Property: Description = "Office Main Floor"
 Property: Occupancy_State = NORMAL
 Property: Status_Flags = {FALSE, FALSE, FALSE, FALSE}
 Property: Event_State = NORMAL
 Property: Reliability = NO_FAULT_DETECTED
 Property: Out_Of_Service = FALSE
 Property: Occupancy_Count = 2
 Property: Occupancy_Count_Enable = TRUE
 Property: Adjust_Value = 0
 Property: Occupancy_Upper_Threshold = 100
 Property: Occupancy_Lower_Threshold = 0
 Property: Credentials_In_Zone = ((Access Credential, Instance 12110),
 (Access Credential, Instance 12245))
 Property: Last_Credential_Added = (Access Credential, Instance 12110)
 Property: Last_Credential_Removed = (Access Credential, Instance 14430)
 Property: Passback_Mode = PASSBACK_OFF
 Property: Passback_Timeout = 10
 Property: Ingress_Points = ((Access Point, Instance 2), (Access Point, Instance 3))
 Property: Egress_Points = ((Access Point, Instance 12), (Access Point, Instance 13))
 Property: Time_Delay = 10
 Property: Notification_Class = 39
 Property: Alarm_Values = (AT_UPPER_THRESHOLD, ABOVE_UPPER_THRESHOLD)
 Property: Event_Enable = {TRUE, TRUE, TRUE}
 Property: Acked_Transitions = {TRUE, TRUE, TRUE}
 Property: Notify_Type = EVENT
 Property: Event_Time_Stamps = ((23 MAR 2007, FRIDAY), 09:11:14.3)),
 (*-*.*.*.*.*.*),
 ((23 MAR 2007, FRIDAY), 11:13:21.3))

135-2004j-3. Add a new Access User object type.

Rationale
 Support for access control in BACnet includes requires a standardized object whose properties represent the externally visible characteristics associated with a user of a physical access control system, which may be an individual person, a group of users, or an asset.

Addendum 135-2004j-3

[Insert new clause **12.X** and Table **12-X**, p. 251]

12.X Access User Object Type

The Access User object type defines a standardized object whose properties represent the externally visible characteristics associated with a user of a physical access control system.

The Access User object is used to represent an individual person, a group of users, or an asset. Relationships among access users are supported for representation of hierarchical organizations (e.g., companies, departments, or groups of any kind) or for representing ownership of assets.

The Access User object is not directly involved in authentication and authorization. It is used for informational purposes. It can hold a name, a reference number and a reference to an external system providing details of the access user.

The Access User object can have Access Credential objects assigned. This information can be used for administrative purposes (e.g., disabling all credentials of a person).

In a physical access control system, each instance of this object can be replicated over multiple devices. In this case, objects representing the same user shall have the same instance number in each device; however, it is a local matter as to how the objects are synchronized.

The Access User object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Access User Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
User_Type	BACnetAccessUserType	R
User_Name	CharacterString	O
User_External_Identifier	CharacterString	O
User_Information_Reference	CharacterString	O
Members	List of BACnetDeviceObjectReference	O
Member_Of	List of BACnetDeviceObjectReference	O
Credentials	List of BACnetDeviceObjectReference	R
Profile_Name	CharacterString	O

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 `ACCESS_USER`.

12.X.4 Description

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

12.X.5 User_Type

This property, of type `BACnetAccessUserType`, specifies the access user type this object represents. The following user types are defined:

ASSET	The Access User object represents a physical item.
GROUP	The Access User object represents a group of access users.
PERSON	The Access User object represents an individual person.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to allow proprietary access user types other than those defined by this standard. For proprietary extensions of this enumeration, see clause 23.1 of this standard.

12.X.6 User_Name

This optional property, of type `CharacterString`, is a string of printable characters which specifies the name of the access user. The content is not restricted and can contain multiple lines.

12.X.7 User_External_Identifier

This optional property, of type `CharacterString`, specifies an external identifier associated with the access user. While the content is typically unique, its interpretation is a local matter.

12.X.8 User_Information_Reference

This optional property, of type `CharacterString`, specifies a reference to an external system where additional information of the user can be found. The interpretation of the content is a local matter.

12.X.9 Members

This optional property, of type `List of BACnetDeviceObjectReference`, references the Access User objects that represent the associated access users. Each object referenced shall be an Access User object.

12.X.10 Member_Of

This optional property, of type `List of BACnetDeviceObjectReference`, references the Access User objects that represent the access users to which this access user is associated. Each object referenced shall be an Access User object.

12.X.11 Credentials

This property, of type `List of BACnetDeviceObjectReference`, references all Access Credential objects that represent those credentials which are owned by this access user. Each object referenced shall be an Access Credential object.

12.X.12 Profile_Name

This optional property, of type `CharacterString`, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name begins 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Add new **BACnetAccessUserType** enumeration to Clause 21, p. 408]

```
BACnetAccessUserType ::= ENUMERATED {
    asset    (0),
    group   (1),
    person  (2)
    ...
}
-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
-- 64-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

[Note: changes to the **BACnetObjectType** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectTypesSupported** BITSTRING appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetPropertyIdentifier** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to **Table 23-1 Extensible Enumerations** appear in Addendum 135-2004j-1.]

[Add new object type structure to **ANNEX C**, p.453]

```
ACCESS_USER ::= SEQUENCE {
    object-identifier [75] BACnetObjectIdentifier,
    object-name       [77] CharacterString,
    object-type       [79] BACnetObjectType,
    description       [28] CharacterString OPTIONAL,
    user-type         [307] BACnetAccessUserType,
    user-name         [306] CharacterString OPTIONAL,
    user-external-identifier [304] CharacterString OPTIONAL,
    user-information-reference [305] CharacterString OPTIONAL,
    members           [285] SEQUENCE OF BACnetDeviceObjectReference OPTIONAL,
    member-of         [159] SEQUENCE OF BACnetDeviceObjectReference OPTIONAL,
    credentials       [263] SEQUENCE OF BACnetDeviceObjectReference,
    profile-name      [167] CharacterString OPTIONAL
}
```

[Add new **ANNEX D.X**, p.484]

D.X Example of an Access User object

In this example, the person Dorothy H. Miller is represented as an Access User object. Note that this is a fictitious person.

Property:	Object_Identifier =	(Access User, Instance 2)
Property:	Object_Name =	"Miller37"
Property:	Object_Type =	ACCESS_USER
Property:	Description =	"Dorothy H. Miller, CEO"
Property:	User_Type =	PERSON
Property:	User_Name =	"Dorothy H. Miller"
Property:	User_External_Identifier =	"SSN-575-99-1234"
Property:	User_Information_Reference =	"HRMS:D93345666"
Property:	Members =	((Access User, Instance 734), (Access User, Instance 41))
Property:	Member_Of =	((Access User, Instance 12), (Access User, Instance 1))
Property:	Credentials =	((Access Credential, Instance 12110), (Access Credential, Instance 41445))

135-2004j-4. Add a new Access Rights object type.

Rationale

Support for access control in BACnet requires a standardized object whose properties represent the externally visible characteristics associated with access rights for physical access control. This object is a collection of individual access rule specifications that define privileges for entering and leaving access controlled zones or for accessing other resources or functions.

Addendum 135-2004j-4

[Insert new clause **12.X** and Table **12-X**, p. 251]

12.X Access Rights Object Type

The Access Rights object type defines a standardized object whose properties represent the externally visible characteristics associated with access rights for physical access control.

The Access Rights object is a collection of individual access rule specifications which define privileges for entering and leaving access controlled zones or for accessing other resources or functions. One or many credentials can share this collection of access rules. This object supports role-based access control models.

Each access rule specifies where and when access is granted. The location where access is granted can be specified as an access controlled point or a zone. The condition when access is granted is specified by referencing a property that can be evaluated to TRUE or FALSE. In the most typical case this references the Present_Value of a Schedule object that specifies time ranges. Each of these access rules can be enabled or disabled individually, or in general.

The Access Rights object can specify an accompaniment requirement that defines the access user that owns the accompanying credential, the credential required to accompany, or the access rights required to be assigned to the accompanying credential.

This object supports inheritance of access rights. Access rules are inherited from other referenced Access Rights objects. Inheritance is recursive, which allows simple modeling of complex relationships. The inheritance mechanism facilitates role-based access control modeling of access rights.

In a physical access control system each instance of this object can be replicated over multiple devices. In this case, objects representing the same user shall have the same instance number in each device; however, it is a local matter as to how the objects are synchronized.

The Access Rights object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Access Rights Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Enable	BOOLEAN	R
Access_Rules	BACnetARRAY[N] of BACnetAccessRule	R
Access_Rules_Enable	BACnetARRAY[N] of BOOLEAN	R ¹
Accompanied	BACnetObjectIdentifier	O
Inherited_Access_Rights	List of BACnetObjectIdentifier	O
Profile_Name	CharacterString	O

¹ The size of this array shall be the same as the size of the Access_Rules array.

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 ACCESS_RIGHTS.

12.X.4 Description

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

12.X.5 Enable

This property, of type BOOLEAN, indicates whether this object is enabled (TRUE) or disabled (FALSE). When this object is disabled all the access rules specified in the Access_Rules property are disabled. In addition, inheritance of access rules does not take place.

12.X.6 Access_Rules

This property, of type BACnetARRAY[N] of BACnetAccessRule, specifies the access rules. Each element of the array is a structure with the following fields:

Time-Range This field, of type BACnetObjectPropertyReference, references a property that can be evaluated to TRUE or FALSE, which defines if the rule is valid (TRUE) or not (FALSE).

If the value of the referenced property is of type Unsigned, a value of zero shall evaluate to FALSE, while any other value shall evaluate to TRUE.

If the value of the referenced property is of type INTEGER, a value of less than or equal to zero shall evaluate to FALSE, while any value greater than zero shall evaluate to TRUE.

If the value of the referenced property is of type BACnetBinaryPV, then INACTIVE shall evaluate to FALSE, while ACTIVE evaluates to TRUE.

If the referenced property does not exist or its value cannot be retrieved, or the value is of type NULL, the Time-Range evaluates to FALSE.

If the reference property is of any other type, then the evaluation is a local matter.

If Time-Range contains an undefined reference (i.e., 4194303 in the instance part), then Time-Range evaluates to TRUE.

Note: This field can reference a Schedule object Present_Value property for the specification of time ranges.

Location This field, of type `BACnetObjectIdentifier`, refers to the Access Point or Access Zone this access rule is valid for.

When Location refers to an Access Point object, this access controlled point is required to be the location where the credential used to request access has been authenticated.

When Location refers to an Access Zone object, the access controlled point where the credential used to request access has been authenticated is required to be an ingress point to this zone.

If Location contains an undefined reference (i.e., 4194303 in the instance part), then this access rule is considered as valid for any location.

If an element of the array can be found where `Time_Range` evaluates to `TRUE`, Location matches the location of authentication, and the corresponding element of the `Access_Rules_Enable` property is `TRUE`, the evaluation of access rules is considered successful. The size of this array shall equal the size of the `Access_Rules_Enable` array. The i^{th} element of this array corresponds to the i^{th} element of the `Access_Rules_Enable` property.

12.X.6.1 Resizing Access_Rules Array and Access_Rules_Enable Array by Writing any of these Properties

The size of the `Access_Rules` and `Access_Rules_Enable` arrays shall be maintained so that all have the same size. If any of these arrays are writable and the number of elements of one is reduced, then each of the arrays shall be truncated to the new reduced size. If any of these arrays are writable and the number of elements of one is increased, then each of the arrays shall be increased to the new expanded size and the new array elements initialized according to the requirements of each property. See 12.X.6.2 and 12.X.7.2.

12.X.6.2 Initializing New Array Elements When the Array Size is Increased

If the size of the `Access_Rules` array is increased without entry values being provided, the new array entries shall be initialized to contain undefined references in both the Location and Time-Range.

12.X.7 Access_Rules_Enable

This property, of type `BACnetARRAY[N]` of `BOOLEAN`, specifies, for each access rule in the `Access_Rules` property, whether that access rule is enabled (`TRUE`) or not (`FALSE`). The size of this array shall equal the size of the `Access_Rules` array. The i^{th} element of this array corresponds to the i^{th} element of the `Access_Rules` property.

12.X.7.1 Resizing Access_Rules Array and Access_Rules_Enable Array by Writing any of these Properties

See 12.X.6.1.

12.X.7.2 Initializing New Array Elements When the Array Size is Increased

If the size of the `Access_Rules_Enable` array is increased without entry values being provided, the new array entries shall be initialized to `FALSE`.

12.X.8 Accompanied

This optional property, of type `BACnetObjectIdentifier`, specifies that the original credential is required to be accompanied by a second credential that meets the accompaniment criteria. It is a local matter as to whether the accompanying credential is required to be presented before or after the original credential. If this condition is not fulfilled, no access rules apply.

The accompaniment condition is specified as:

- (a) If this property refers to an Access Rights object, then the accompanying credential is required to refer to or inherit from that Access Rights object.
- (b) If this property refers to an Access Credential object, then this object is required to represent the accompanying credential.
- (c) If this property refers to an Access User object, then this object is required to represent the access user which owns the accompanying credential.

If the instance part of this object reference has a value of 4194303, no accompaniment requirement is specified.

12.X.9 Inherited_Access_Rights

This optional property, of type List of BACnetObjectIdentifier, specifies the Access Rights objects from which this object inherits access rules. This recursively includes access rules inherited by the referenced Access Rights objects. If an inherited Access Rights object is disabled, then its access rules and those which it inherits shall not be inherited by this object.

Only enabled access rules are inherited from Access Rights objects.

Accompaniment requirements are also inherited from inherited Access Rights objects. The local accompaniment requirement shall supersede an inherited accompaniment requirement.

This property shall contain references to Access Rights objects only.

12.X.10 Profile_Name

This optional property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name begins 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Add new **BACnetAccessRule** production to Clause 21, p. 408]

```
BACnetAccessRule ::= SEQUENCE {  
    timeRange [1] BACnetObjectPropertyReference,  
    location [2] BACnetObjectIdentifier  
}
```

[Note: changes to the **BACnetObjectType** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectTypesSupported** BITSTRING appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetPropertyIdentifier** enumeration appear in Addendum 135-2004j-1.]

[Add new object type structure to **ANNEX C**, p.453]

```
ACCESS-RIGHT ::= SEQUENCE {  
    object-identifier [75] BACnetObjectIdentifier,  
    object-name [77] CharacterString,
```

```

object-type      [79]   BACnetObjectType,
description     [28]   CharacterString OPTIONAL,
enable         [133]  BOOLEAN,
access-rules    [248]  SEQUENCE OF BACnetAccessRule, -- accessed as a BACnetARRAY
access-rules-enable [249] SEQUENCE OF BOOLEAN, -- accessed as a BACnetARRAY
accompanied    [251]  BACnetObjectIdentifier OPTIONAL,
inherited-access-rights [274] List of BACnetObjectIdentifier OPTIONAL,
profile-name    [167]  CharacterString OPTIONAL
}

```

[Add new ANNEX D.X, p.484]

D.X Example of an Access Rights object

In this example, rules for access are represented as an Access Rights object. It specifies rules that allow access to two secured zones, represented by the local Access Zone object instance 44 and instance 74, during the same time range specified by a local Schedule object instance 44. The rules may apply only if the access credential represented by Access Credential object instance 33 is authenticated and authorized as well. Other access rules are inherited from Access Rights object instance 1.

```

Property: Object_Identifier = (Access Rights, Instance 2)
Property: Object_Name = "AR_OFF_NIGHT"
Property: Object_Type = ACCESS_RIGHTS
Property: Description = "Access Rights for Offices at night shift"
Property: Enable = TRUE
Property: Access_Rules = ( ( ((Schedule, Instance 44), Present_Value), (Access Zone, Instance 1) ),
                          ( ((Schedule, Instance 44), Present_Value), (Access Zone, Instance 23) ))
Property: Access_Rules_Enable = (TRUE, TRUE)
Property: Accompanied = (Access Credential, Instance 33)
Property: Inherited_Access_Rights = ((Access Rights, Instance 1))

```

135-2004j-5. Add a new Access Credential object type.

Rationale

Support for access control in BACnet requires a standardized object whose properties represent the externally visible characteristics associated with a credential used for authentication and its assigned access rights for authorization when requesting access. The credential can be owned by an access user of any type; ownership is represented by a reference to the Access User object that represents the owning access user.

Addendum 135-2004j-5

[Insert new clause **12.X** and Table **12-X**, p. 251]

12.X Access Credential Object Type

The Access Credential object type defines a standardized object whose properties represent the externally visible characteristics associated with a credential used for authentication and for determining its assigned access rights for authorization when requesting access.

The credential can be owned by an access user of any type. This ownership is represented by a reference to the Access User object that represents the owning access user.

The Access Credential object is a container of related authentication factors. An Access Credential object can represent a single authentication factor, a group of authentication factors each having identical access rights, or multiple authentication factors required for multi-factor-authentications.

The access rights assigned to the credential are specified by referencing Access Rights objects. Each reference can be individually enabled or disabled.

The Credential_Status indicates the validity of this credential for authentication. The status is derived from other properties of this object or can be set from an external process.

The credential can be restricted in its use for authentication. It can be restricted based on activation and expiry dates, the number of days it can be used or the number of uses. It can be disabled if it is not used for a specified number of days. The credential can be exempted from authorization checks such as passback violation enforcement and occupancy enforcements. It can indicate whether an extended time is required to pass through a door.

A threat authority can be specified for the credential. If this value is lower than the threat level at the access controlled point, then access is denied.

The credential can be flagged to be traced. Any access controlled point recognizing this credential will generate a corresponding TRACE access event.

In a physical access control system each instance of this object can be replicated over multiple devices. In this case, objects representing the same credential shall have the same instance number in each device; however, it is a local matter as to how the objects are synchronized.

The Access Credential object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Access Credential Object Type

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Description	CharacterString	O
Credential_Status	BACnetBinaryPV	R
Reason_For_Disable	List of BACnetAccessCredentialDisableReason	R
Authentication_Factors	BACnetARRAY[N] of BACnetAuthenticationFactor	R
Activation_Time	BACnetDateTime	O
Expiry_Time	BACnetDateTime	O
Credential_Disable	BACnetAccessCredentialDisable	R
Days_Remaining	Unsigned	O
Uses_Remaining	Unsigned	O
Absentee_Limit	Unsigned	O ¹
Belongs_To	BACnetDeviceObjectReference	O
Assigned_Access_Rights	BACnetARRAY[N] of BACnetObjectIdentifier	R
Access_Rights_Enable	BACnetARRAY[N] of BOOLEAN	R ²
Last_Access_Point	BACnetDeviceObjectReference	O
Last_Access_Event	BACnetAccessEvent	O
Last_Use_Time	BACnetDateTime	O
Trace_Flag	BOOLEAN	O
Threat_Authority	BACnetAccessThreatLevel	O
Extended_Time_Enable	BOOLEAN	O
Passback_Exemption	BOOLEAN	O
Occupancy_Count_Exemption	BOOLEAN	O
Profile_Name	CharacterString	O

¹ If this property is present the property Last_Use_Time shall be present.

² The size of this array shall be the same as the size of the Assigned_Access_Rights array.

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 ACCESS_CREDENTIAL.

12.X.4 Description

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

12.X.5 Credential_Status

This property, of type BACnetBinaryPV, indicates whether the credential is active or inactive. Only the value ACTIVE enables the credential to be used for authentication. While the list in property Reason_For_Disable is nonempty, the status of the credential shall be INACTIVE, otherwise it shall be ACTIVE.

12.X.6 Reason_For_Disable

This property, of type List of BACnetAccessCredentialDisableReason, contains a list of reasons why the credential has been disabled. The credential can be disabled for multiple reasons at the same time. While the Credential_Status property has a value INACTIVE this list shall not be empty. When an entry is removed from this list that results in the list becoming empty, the Credential_Status shall be set to ACTIVE.

The list of reasons for which the credential can be disabled are as follows:

DISABLED	The credential is disabled for unspecified reasons.
DISABLED_NEEDS_PROVISIONING	The credential needs further provisioning which can include vendor proprietary data.
DISABLED_UNASSIGNED	The credential is not currently assigned to any access user. This status is assigned only if the property Belongs_To is present and contains instance 4194303 in the object identifier.
DISABLED_NOT_YET_ACTIVE	The credential is not yet valid at this time. The current time is before the Activation_Time.
DISABLED_EXPIRED	The credential is not valid any more at this time. The current time is after the Expiry_Time.
DISABLED_LOCKED_OUT	Too many retries in multi-factor authentications have been performed.
DISABLED_USE_COUNT	The maximum number of uses has been reached.
DISABLED_INACTIVITY	The credential has exceeded the allowed period of inactivity.
DISABLED_MANUAL	The credential is commanded to be disabled by a human operator.
DISABLED_MISSING	The physical credential is missing; it may be stolen or lost.
DISABLED_DAMAGED	The physical credential is damaged.
DISABLED_DESTROYED	The physical credential is destroyed.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values to indicate disable reasons other than those defined by this standard. For proprietary extensions of this enumeration, see clause 23.1 of this standard.

12.X.7 Authentication_Factors

This property, of type BACnetARRAY[N] of BACnetAuthenticationFactor, specifies the authentication factors which belong to this credential.

12.X.8 Activation_Time

This optional property, of type BACnetDateTime, indicates the date and time at or after which the credential is active. If the current time is before the activation time, the credential shall be disabled and the value DISABLED_NOT_YET_ACTIVE shall be added to the Reason_For_Disable list. The value DISABLED_NOT_YET_ACTIVE shall be removed from the list when this condition no longer applies. If any of the fields of the BACnetDateTime contain "wildcard" values or this property is not present, then the credential is active from 'start of time'.

12.X.9 Expiry_Time

This optional property, of type BACnetDateTime, indicates the date and time after which the credential is expired. This defines the end of the validity period of the credential. If the current time is after the expiry time, the credential shall be disabled and the value DISABLED_EXPIRED shall be added to the Reason_For_Disable list. The value DISABLED_EXPIRED shall be removed from the list when this condition no longer applies. If any of the fields of the

BACnetDateTime contain "wildcard" values or this property is not present, then the credential is active until 'end of time'.

12.X.10 Credential_Disable

This property, of type BACnetAccessCredentialDisable, allows an operator or external process to disable the credential. When this property takes on any value other than NONE, the credential shall be disabled and the corresponding disable reason shall be added to the Reason_For_Disable list. When this property is changed, any value previously added to the Reason_For_Disable list as a result of changing this property shall be removed from that list.

The following disable values are defined:

NONE	The credential has not been disabled by an operator or external process.
DISABLE	The credential has been disabled for unspecified reasons. The value DISABLED shall be added to the Reason_For_Disable property.
DISABLE_MANUAL	The credential has been disabled by a human operator. The value DISABLED_MANUAL shall be added to the Reason_For_Disable property.
DISABLE_MISSING	The credential is disabled because it has been reported to be stolen or lost. The value DISABLED_MISSING shall be added to the Reason_For_Disable property.
DISABLE_DAMAGED	The credential is disabled because it has been damaged. The value DISABLED_DAMAGED shall be added to the Reason_For_Disable property.
DISABLE_DESTROYED	The credential is disabled because it has been destroyed. The value DISABLED_DESTROYED shall be added to the Reason_For_Disable property.
DISABLE_LOCKOUT	The credential is disabled because has been locked out by an external process. The value DISABLED_LOCKOUT shall be added to the Reason_For_Disable property.
<Proprietary Enum Values>	A vendor may use other proprietary enumeration values for disabling a credential other than those defined by this standard. A disable reason shall be added to the Reason_For_Disable property. It is a local matter which disable reason is added. For proprietary extensions of this enumeration, see clause 23.1 of this standard.

12.X.11 Days_Remaining

This optional property, of type Unsigned, indicates the number of remaining days for which the credential can be used. The remaining days do not need to be contiguous days. If this property is present and has a value of zero, authentication is no longer possible using this credential. If this property becomes zero, the Access Credential shall be disabled and the value DISABLED_USE_COUNT shall be added to the Reason_For_Disable list. The value DISABLED_USE_COUNT shall be removed from the list when this condition no longer applies.

12.X.12 Uses_Remaining

This optional property, of type Unsigned, indicates the number of remaining uses which the credential can be used for authentication. Access granted at an access controlled point shall count as a single use. If this property is present and has a value of zero, the Access Credential shall be disabled and the value DISABLED_USE_COUNT shall be added to the Reason_For_Disable list. The value DISABLED_USE_COUNT shall be removed from the list when this condition no longer applies.

12.X.13 Absentee_Limit

This optional property, of type Unsigned, specifies the maximum number of consecutive days for which the credential can remain inactive (i.e. unused) before it becomes disabled. The calculation of inactivity duration is based on the time of last use as indicated by the property Last_Use_Time. If Last_Use_Time does not have a valid time and date, then the absentee limit shall be considered to not be exceeded. When the absentee limit is exceeded, the Access Credential shall be disabled and the value DISABLED_INACTIVITY shall be added to the Reason_For_Disable list. The value DISABLED_INACTIVITY shall be removed from the list when this condition no longer applies.

If Absentee_Limit is present, Last_Use_Time shall be present.

12.X.14 Belongs_To

This optional property, of type BACnetDeviceObjectReference, references an Access User object that represents the owning access user (i.e. person, group, or asset). If this property is present and the credential is not assigned to an access user, this property shall contain an instance number of 4194303. The determination of whether the credential is valid for authentication, based on the value of this property, is a local matter. If the credential has not been assigned to an access user and the policy of the site requires that it be assigned, then the credential shall be disabled and the value DISABLED_UNASSIGNED shall be added to the Reason_For_Disable list. The value DISABLED_UNASSIGNED shall be removed from the list when this condition no longer applies.

12.X.15 Assigned_Access_Rights

This property, of type BACnetARRAY [N] of BACnetObjectIdentifier, refers to Access Rights objects that define the access rights assigned to this credential. Each object referenced in the Assigned_Access_Rights property shall be an Access Rights object. Any entry with an instance number of 4194303 or a reference to a non-existent Access Rights object shall be ignored.

12.X.15.1 Resizing Assigned_Access_Rights Array and Access_Rights_Enable Array by Writing any of these Properties

The size of the Assigned_Access_Rights and Access_Rights_Enable arrays shall be maintained so that all have the same size. If any of these arrays are writable and the number of elements of one is reduced, each of the arrays shall be truncated to the new reduced size. If any of these arrays are writable and the number of elements of one is increased, then each of the arrays shall be increased to the new expanded size and the new array elements initialized according to the requirements of each property. See 12.X.15.2 and 12.X.16.2.

12.X.15.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Assigned_Access_Rights array is increased without entry values being provided, the new array entries shall be initialized to contain an instance number of 4194303.

12.X.16 Access_Rights_Enable

This property, of type BACnetARRAY [N] of BOOLEAN, enables (TRUE) or disables (FALSE) the corresponding access rights. The i^{th} element applies to the i^{th} element of the Assigned_Access_Rights property. The size of this array shall be the same as the array size of the Assigned_Access_Rights property.

12.X.16.1 Resizing Assigned_Access_Rights Array and Access_Rights_Enable Array by Writing any of these Properties

See 12.X.15.1.

12.X.16.2 Initializing New Array Elements When the Array Size is Increased

If the size of the Access_Rights_Enable array is increased without entry values being provided, the new array entries shall be initialized to contain the value FALSE.

12.X.17 Last_Access_Point

This optional property, of type BACnetDeviceObjectReference, refers to the last Access Point object where one of the authentication factors of the credential has been used. If property level COV is in effect for this property, any update of this property shall cause a COV notification to be issued, regardless of whether the value of this property changes. If the credential this object represents has never been used, this property shall contain 4194303 for the object instance.

12.X.18 Last_Access_Event

This optional property, of type BACnetAccessEvent, shall specify the last access event generated at an access controlled point on use of this credential. If the credential this object represents has never been used, this property shall have a value of NONE.

12.X.19 Last_Use_Time

This optional property, of type BACnetDateTime, indicates the date and time of the last use of the credential at an access controlled point, independent of whether access was granted or denied. If the credential this object represents has never been used, this property shall have wildcard values for all date and time fields.

12.X.20 Trace_Flag

This optional property, of type BOOLEAN, instructs, if TRUE, an Access Point object to generate a TRACE access event when this credential is used.

12.X.21 Threat_Authority

This optional property, of type BACnetAccessThreatLevel, indicates the maximum threat level for which this credential is valid. If this value is less than the Threat_Level property of the Access Point object where the access credential is used, access is denied. If this property is not present, the threat authority of this credential is assumed to be zero.

12.X.22 Extended_Time_Enable

This optional property, of type BOOLEAN, indicates which command of type BACnetDoorValue shall be used to unlock the access door when access is granted. If extended time is enabled (TRUE), EXTENDED_PULSE_UNLOCK is used, otherwise (FALSE) PULSE_UNLOCK is used.

12.X.23 Passback_Exemption

This optional property, of type BOOLEAN, specifies an exemption from passback enforcement. If passback exemption is enabled (TRUE), the credential is not denied access due to passback violations.

12.X.24 Occupancy_Count_Exemption

This optional property, of type BOOLEAN, specifies an exemption from occupancy enforcement. If occupancy count exemption is enabled (TRUE), the occupancy count in the Access Zone object shall be updated as normal; however, the access credential shall not be denied access due to occupancy threshold enforcement.

12.X.25 Profile_Name

This optional property, of type CharacterString, is the name of an object profile to which this object conforms. To ensure uniqueness, a profile name begins 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Add new **BACnetAccessCredentialDisable** enumeration to Clause 21, p. 408]

```
BACnetAccessCredentialDisable ::= ENUMERATED {
  none           (0),
  disable        (1),
  disable-manual (2),
  disable-missing (3),
  disable-damaged (4),
  disable-destroyed (5),
  disable-lockout (6),
  ...
}
-- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
-- 256-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

[Add new **BACnetAccessCredentialDisableReason** enumeration to Clause 21, p. 408]

```
BACnetAccessCredentialDisableReason ::= ENUMERATED {
  disabled (0),
  disabled-needs-provisioning (1),
  disabled-unassigned (2),
  disabled-not-yet-active (3),
  disabled-expired (4),
  disabled-locked-out (5),
  disabled-use-count (6),
  disabled-inactivity (7),
  disabled-manual (8),
  disabled-missing (9),
  disabled-damaged (10),
  disabled-destroyed (11),
  ...
}
-- Enumerated values 0-255 are reserved for definition by ASHRAE. Enumerated values
-- 256-65535 may be used by others subject to the procedures and constraints described
-- in Clause 23.
```

[Note: **BACnetAccessThreatLevel** is defined in Addendum 135-2004j-1.]

[Note: **BACnetAuthenticationFactor** is defined in Addendum 135-2004j-6.]

[Note: **BACnetAccessEvent** is defined in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectType** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectTypesSupported** BITSTRING appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetPropertyIdentifier** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to **Table 23-1 Extensible Enumerations** appear in Addendum 135-2004j-1.]

[Add new object type structure to **ANNEX C**, p.453]

```
ACCESS-CREDENTIAL ::= SEQUENCE {
  object-identifier      [75]  BACnetObjectIdentifier,
  object-name           [77]  CharacterString,
  object-type           [79]  BACnetObjectType,
  description           [28]  CharacterString OPTIONAL,
  credential-status     [262] BACnetBinaryPV,
  reason-for-disable    [299] LIST OF BACnetAccessCredentialDisableReason
  authentication-factors [255] SEQUENCE OF BACnetAuthenticationFactor,
                        -- accessed as a BACnetARRAY
  activation-time       [252] BACnetDateTime OPTIONAL,
  expiry-time          [267] BACnetDateTime OPTIONAL,
  credential-disable    [261] BACnetAccessCredentialDisable,
  days-remaining        [265] Unsigned OPTIONAL,
  uses-remaining        [308] Unsigned OPTIONAL,
  absentee-limit        [244] Unsigned OPTIONAL,
  belongs-to           [260] BACnetDeviceObjectReference OPTIONAL,
  access-rights         [248] SEQUENCE OF BACnetObjectIdentifier, -- accessed as a BACnetARRAY
  access-rights-enable  [249] SEQUENCE OF BOOLEAN, -- accessed as a BACnetARRAY
  last-access-point     [277] BACnetDeviceObjectReference OPTIONAL,
  last-access-event     [276] BACnetAccessEvent OPTIONAL,
  last-use-time         [281] BACnetDateTime OPTIONAL,
  trace-flag           [302] BOOLEAN OPTIONAL,
  threat-authority      [300] BACnetAccessThreatLevel OPTIONAL,
  extended-time-enable  [268] BOOLEAN OPTIONAL,
  passback-exemption   [295] BOOLEAN OPTIONAL,
  occupancy-count-exemption [289] BOOLEAN OPTIONAL,
  profile-name          [167] CharacterString OPTIONAL
}
```

[Add new **ANNEX D.X**, p.484]

D.X Example of an Access Credential object

In this example, a two track magnetic stripe card access credential is represented as an Access Credential object.

The credential has status **ACTIVE** and can be used for authentication and authorization.

The credential contains two simple number authentication factors.

The credential is valid from 1-MAR-2007 09:00h until 31-MAR-2008 midnight.

The credential can be used at 10 days inside the validity period; it can be used for 30 granted access requests and must be used within 5 days to remain **ACTIVE**.

The credential belongs to an Access User Instance 2.

The credential is assigned to access rights defined by Access Rights object instances 1 and 219.

The assignment to Access Rights object instance 1 is enabled while assignment to instance 219 is disabled.

There is extended access time enabled, but neither passback exemption nor occupancy count exemption is set for this credential.

Property: Object_Identifier = (Access Credential, Instance 33)

Property: Object_Name = "CARD-223444"
Property: Object_Type = ACCESS_CREDENTIAL
Property: Description = "Magnetic Stripe Card#: 223444"
Property: Credential_Status = ACTIVE
Property: Reason_For_Disable = (
Property: Authentication_Factors = (442234444444, 395995999995)
Property: Activation_Time = ((01 MAR 2007, THURSDAY), 09:50:21.3))
Property: Expiry_Time = ((30 MAR 2007, FRIDAY), 23:59:59.9))
Property: Credential_Disable = NONE
Property: Days_Remaining = 10
Property: Uses_Remaining = 30
Property: Absentee_Limit = 5
Property: Belongs_To = (Access User, Instance 2)
Property: Assigned_Access_Rights = ((Access Rights, Instance 1), (Access Rights, Instance 219))
Property: Access_Rights_Enable = (TRUE, FALSE)
Property: Last_Access_Point = (Access Point, Instance 2)
Property: Last_Access_Event = PASSBACK_DETECTED
Property: Last_Use_Time = ((27 MAR 2007, TUESDAY), 11:11:33.2))
Property: Trace_Flag = TRUE
Property: Threat_Authority = 50
Property: Extended_Time_Enable = TRUE
Property: Passback_Exemption = FALSE
Property: Occupancy_Count_Exemption = FALSE

135-2004j-6. Add a new Authentication Factor Input object type.

Rationale

Support for access control in BACnet requires a standardized object whose properties represent the externally visible characteristics of an authentication factor input device, such as card readers, keypads, biometric readers, etc. An authentication factor is a unique digital identifier that is used to verify the identity of a credential.

Addendum 135-2004j-6

[Insert new clause **12.X** and Table **12-X**, p. 251]

12.X Authentication Factor Input Object Type

The Authentication Factor Input object type defines a standardized object whose properties represent the externally visible characteristics of an authentication factor input. Examples of authentication factor input devices are card readers, keypads, biometric readers, etc. An authentication factor is a unique digital identifier that is used to verify the identity of a credential.

The Authentication Factor Input object type and its properties are summarized in Table 12-X and described in detail in this subclause.

Table 12-X. Properties of the Authentication Factor Input Object

Property Identifier	Property Datatype	Conformance Code
Object_Identifier	BACnetObjectIdentifier	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetAuthenticationFactor	R ¹
Description	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Format_Type	BACnetAuthenticationFactorType	R
Format_Class_Supported	Unsigned	O
Read_Status	BACnetAuthenticationFactorReadStatus	R
Update_Time	BACnetTimeStamp	R
Vendor_Identifier	Unsigned16	O ²
Vendor_Format_Identifier	Unsigned	O ²
Profile_Name	CharacterString	O

¹ This property is required to be writable when Out_Of_Service is TRUE.

² These properties are required if the choice PROPRIETARY is used for the 'Value' field of Present_Value.

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 AUTHENTICATION_FACTOR_INPUT.

12.X.4 Present_Value

This property, of type BACnetAuthenticationFactor, is a structure that holds a site specific value which identifies the class of authentication factor (Format-Class) and the current authentication factor value (Value).

Format-Class	This is a site specific value, of type Unsigned, that identifies the class of authentication factors which can be read by this input. If the optional property Format_Class_Supported exists then this value shall correspond to the value of that property. This field is used in sites where different formats of authentication factors are used that have the same authentication factor format type. The format class value is used to differentiate between the different formats. A value of zero is used as the default where no differentiation of authentication factors is required.
Value	This is the authentication factor. It is a choice of different authentication factor format types where the option supported is specified by the Format_Type property. The current choice in the Value field shall correspond to the type specified by the Format_Type property. If there is no current value yet it shall take the option UNDEFINED with a NULL value. If an authentication factor is read that contains errors or that cannot be interpreted as the specified type, this field shall take the option UNDEFINED, and any information related to the error may be placed as the value.

The Present_Value property shall be writable when Out_Of_Service is TRUE. When the object is not in service the allowable values that can be written to the Format-Class and Value fields shall still comply with the restrictions defined in the corresponding definitions.

12.X.5 Description

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

12.X.6 Status_Flags

This property, of type BACnetStatusFlags, represents four Boolean flags that indicate the general "health" of the Authentication Factor Input 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	The value of this flag shall be 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	The value of this flag shall be logical FALSE (0).
OUT_OF_SERVICE	Logical TRUE (1) if the Out_Of_Service property has a value of TRUE, otherwise logical FALSE (0).

12.X.7 Reliability

This optional property, of type BACnetReliability, provides an indication of whether the Present_Value is "reliable" as far as the BACnet Device or operator can determine. The Reliability property for this object type may have any of the following values:

{NO_FAULT_DETECTED, CONFIGURATION_ERROR, UNRELIABLE_OTHER}.

12.X.8 Out_Of_Service

The Out_Of_Service property, of type BOOLEAN, is an indication whether (TRUE) or not (FALSE) the Present_Value of the Authentication Factor Input object is prevented from being modified by software local to the BACnet device in which the object resides. When Out_Of_Service is TRUE, the Present_Value property may be written to.

12.X.9 Format_Type

This property, of type BACnetAuthenticationFactorType, is an enumeration that determines the type of authentication factor returned in the Value field of the Present_Value. When this value is changed it shall cause the Value field of the Present_Value property to take the option UNDEFINED with a value of NULL and Read_Status to take the value NOT_INITIALIZED.

12.X.10 Format_Class_Supported

This optional property, of type Unsigned, specifies the value used in the Format-Class field of the Present-Value property. This property is used in sites where different formats of authentication factors are used that have the same authentication factor format type. The format class value is used to differentiate between the different formats. If this property exists, a value of zero is used as default where no differentiation is required. Otherwise, the value is site specific and can be any non-zero value. When this value is changed, it shall cause the Value field of the Present_Value property to take the option UNDEFINED with a value of NULL and Read_Status to take on the value NOT_INITIALIZED.

12.X.11 Read_Status

This required property of type BACnetAuthenticationFactorReadStatus, indicates the status of the value of the Present_Value property. The values of this enumeration are as follows:

NOT_INITIALIZED	The Present_Value property does not yet contain a valid or erroneous reading. The Value field of the Present_Value has a value of option UNDEFINED, with a NULL value or zero length string of any type.
VALID	The Present_Value contains a valid authentication factor.
ERROR	The Present_Value contains an erroneous reading. The Value field of the Present_Value has a value of option UNDEFINED and may provide the erroneous raw authentication factor.

12.X.12 Update_Time

This property, of type BACnetTimeStamp, indicates the most recent update time when the Present_Value was updated. This property shall update its value on each update of the Present_Value. Update times of type Time or Date shall have X'FF' in each octet, and Sequence number update times shall have the value 0 if no update has yet occurred.

12.X.13 Vendor_Identifier

This optional property, of type Unsigned16, holds the vendor identifier of the vendor defining the proprietary format. This property shall be present if a proprietary format is supported. This value may differ from the Vendor_Identifier property value in the Device object, which indicates the device manufacturer.

12.X.14 Vendor_Format_Identifier

This optional property of type Unsigned holds the vendor's proprietary format identifier. This property shall be present if a proprietary format is supported.

12.X.15 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 begins 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 document named by the remainder of the profile name. 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. The definition of the profiles themselves is outside the scope of this standard.

[Change Table 13-1, p. 254]

Table 13-1. Standardized Objects That May Support COV Reporting

Object Type	Criteria	Properties Reported
...
<i>Authentication Factor Input</i>	<i>If Update_Time changes at all</i>	<i>Present_Value, Status_Flags, Update_Time, Read_Status</i>
...		

[Add new **BACnetAuthenticationFactor** production to Clause 21, p. 408]

```

BACnetAuthenticationFactor ::= SEQUENCE {
    formatClass      [0] Unsigned,
    value            [1] CHOICE {
        undefined    [0] CHOICE {
            null      [0] NULL,
            errorUnsigned [1] Unsigned,
            errorOctetString [2] OCTET STRING,
            errorBitString [3] BITSTRING,
            errorCharacterString [4] CharacterString
        },
        simpleNumber [1] Unsigned,
        simpleAlphaNumeric [2] CharacterString,
        facilityAndCardNumber [3] SEQUENCE {
            facility Unsigned,
            card Unsigned
        },
        fascn [4] SEQUENCE { -- FIPS 201 FASC-N structure
            agencyCode Unsigned (0..9999),
            systemSiteCode Unsigned (0..9999),
            credentialNumber Unsigned (0..999999)
        },
        fascnFull [5] SEQUENCE { -- FIPS 201 FASC-N full structure
            agencyCode Unsigned (0..9999),
            systemSiteCode Unsigned (0..9999),
            credentialNumber Unsigned (0..999999),
    }
}
    
```

```

seriesCode          Unsigned (0..9),
individualCredentialIssue Unsigned (1..9),
personIdentifier    Unsigned (0..9999999999),
organizationalCategory Unsigned (0..9),
organizationalIdentifier Unsigned (0..9999)
associationCategory Unsigned (0..9)
},
gsa75                [6] SEQUENCE { -- GSA 75 format
agencyCode           Unsigned (0..16383),
systemSiteCode       Unsigned (0..16383),
credentialNumber      Unsigned (0.. 1048575),
expirationDate        Date
},
gsa75hmac            [7] SEQUENCE { -- GSA 75 format with 32 bit HMAC code
agencyCode           Unsigned (0..16383),
systemSiteCode       Unsigned (0..16383),
credentialNumber      Unsigned (0.. 1048575),
expirationDate        Date,
hmac32               OCTET STRING (SIZE(4))
},
cbeffA              [8] OCTET STRING, -- NIST CBEFF Patron Format A (CBEFF)
-- content formatted according to
-- CBEFF definitions
cbeffB              [9] OCTET STRING, -- NIST CBEFF Patron Format B (BioAPI)
-- content formatted according to
-- CBEFF definitions
cbeffC              [10] OCTET STRING, -- NIST CBEFF Patron Format C (ANSI
-- Standard X9.84) content formatted
-- according to CBEFF definitions
userPassword        [11] SEQUENCE {
username             CharacterString,
password             CharacterString
},
-- Context tag 254 is used for proprietary formats as follows:
proprietary         [254] SEQUENCE {
vendorID             [0] Unsigned,
typeID               [1] Unsigned,
value                [2] ABSTRACT-SYNTAX.&Type
}
-- This format is to be used for proprietary formats not fitting into any standard format. The exact
-- format definition of the value and the associated type ID is vendor specific.
}
}

```

[Add new **BACnetAuthenticationFactorReadStatus** enumeration to Clause 21, p. 408]

```

BACnetAuthenticationFactorReadStatus ::= ENUMERATED {
notInitialized      (0),
valid               (1),
error               (2)
}

```

[Add new **BACnetAuthenticationFactorType** enumeration to Clause **21**, p. 408]

```
BACnetAuthenticationFactorType ::= ENUMERATED {
    undefined           (0),
    simple-number       (1),
    simple-alpha-numeric (2),
    facility-and-card-number (3),
    fasc-n              (4),
    fasc-n-full         (5),
    gsa75               (6),
    gsa75hmac           (7),
    cbeff-A             (8),
    cbeff-B             (9),
    cbeff-C             (10),
    user-password       (11),
    reserved             (253),
    proprietary         (254)
}
```

[Note: changes to the **BACnetObjectType** enumeration appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetObjectTypesSupported** BITSTRING appear in Addendum 135-2004j-1.]

[Note: changes to the **BACnetPropertyIdentifier** enumeration appear in Addendum 135-2004j-1.]

[Add new object type structure to **ANNEX C**, p.453]

```
AUTHENTICATION_FACTOR_INPUT::= SEQUENCE {
    object-identifier [75] BACnetObjectIdentifier,
    object-name       [77]  CharacterString,
    object-type       [79]  BACnetObjectType,
    present-value     [85]  BACnetAuthenticationFactor,
    description       [28]  CharacterString OPTIONAL,
    status-flags      [111] BACnetStatusFlags,
    reliability        [103] BACnetReliability OPTIONAL,
    out-of-service    [81]  BOOLEAN,
    format-type       [272] BACnetAuthenticationFactorType,
    format-class-supported [271] Unsigned OPTIONAL,
    read-status       [298] BACnetAuthenticationFactorReadStatus,
    update-time       [189] BACnetTimeStamp,
    vendor-identifier [120] Unsigned16 OPTIONAL,
    vendor-format-identifier [309] Unsigned OPTIONAL,
    profile-name      [168] CharacterString OPTIONAL
}
```

[Add new **ANNEX D.X**, p.484]

D.X Example of an Authentication Factor Input object

This example of an authentication factor input object represents a card reader on the unsecure side of the door. The format type for this reader is facility-and-card-number and the specific format read is 26 bit Wiegand. A valid card was recently read with a facility code of 763 and a card number of 20926. The optional *Vendor_Identifier* and *Vendor_Format_Identifier* properties are not supported.

```
Property: Object_Identifier = (Authentication Factor Input, Instance 1)
Property: Object_Name = "Main Entrance Card Reader"
```

Property:	Object_Type =	AUTHENTICATION_FACTOR_INPUT
Property:	Present_Value =	(0, (763, 20926)}
Property:	Description =	"Main entrance south building card reader – 26 bit Wiegand"
Property:	Status_Flags =	(FALSE, FALSE, FALSE, FALSE)
Property:	Reliability =	NO_FAULT_DETECTED
Property:	Out_Of_Service =	FALSE
Property:	Format_Type =	FACILITY_AND_CARD_NUMBER
Property:	Format_Class_Supported =	0
Property:	Read_Status =	VALID
Property:	Update_Time =	((7 JAN 2007, SUNDAY), 15:30:51.70)

135-2004j-7. Add a new ACCESS_EVENT event algorithm.

Rationale

A new event algorithm is needed for access control support in BACnet that is related to user actions, authentication and authorization decisions at an Access Point.

Addendum 135-2004j-7

[Change 12.12.5 and Table 12-15, Event Enrollment object, pp. 185-186]

12.12.5 Event_Type

This read only property, of type BACnetEventType, indicates the type of event algorithm that is to be used to detect the occurrence of events and report to enrolled devices. This parameter is an enumerated type that may have any of the following values:

{CHANGE_OF_BITSTRING, CHANGE_OF_STATE, CHANGE_OF_VALUE, COMMAND_FAILURE, FLOATING_LIMIT, OUT_OF_RANGE, BUFFER_READY, CHANGE_OF_LIFE_SAFETY, EXTENDED, ACCESS_EVENT}.

There is a specific relationship between each event algorithm, the parameter list, and the event types that are valid for the event. The Event_Type reflects the algorithm that is used to determine the state of an event. The algorithm for each Event_Type is specified in Clause 13. The Event_Parameters property provides the parameters needed by the algorithm.

The valid combinations of Event_Type, Event_State, and Event_Parameters values are summarized in Table 12-15.

Table 12-15. Event_Types, Event_States, and their Parameters

Event_Type	Event_State	Event_Parameters
...
CHANGE_OF_LIFE_SAFETY	NORMAL OFFNORMAL LIFE_SAFETY_ALARM	Time_Delay List_Of_Alarm_Values List_Of_Life_Safety_Alarm_Values Mode_Property_Reference
ACCESS_EVENT	NORMAL	List_Of_Access_Events Update_Time_Reference

[Change 12.12.7, pp. 186-188]

12.12.7 Event_Parameters

The Event_Parameters property, of type BACnetEventParameter, determines the algorithm used to monitor the referenced object and provides the parameter values needed for this algorithm. The meaning of each value in the Event_Parameters depends on the algorithm as indicated by the Event_Type column in Table 12-15. Each of the possible parameters is described below.

...
Mode_Property_Reference

This parameter, of type BACnetDeviceObjectPropertyReference, applies to the CHANGE_OF_LIFE_SAFETY algorithm. It identifies the object and property that provides the operating mode of the referenced object providing life safety functionality (normally the Mode property). This parameter may reference only object properties that are of type BACnetLifeSafetyMode.

List_Of_Access_Events

This parameter is a list of BACnetAccessEvent values that applies to the ACCESS_EVENT algorithm. If the value of the referenced property is updated to one of the values in the

List_Of_Access_Events, then the *Event_State* property of the *Event_Enrollment* object makes a transition *TO-NORMAL* and appropriate notifications are sent.

Update_Time_Reference

This parameter, of type BACnetDeviceObjectPropertyReference, applies to the ACCESS_EVENT algorithm. It identifies the object and property that provide the last update time of the referenced property that is monitored for access events (normally the Update_Time property). This parameter may only reference object properties that are of type BACnetTimeStamp.

[Change 13.2 and Table 13-2 through Table 3-4, pp.255-257]

13.2 Intrinsic Reporting

...

In the case of Life Safety Zone and Life Safety Point, the *Life_Safety_Alarm_Values* property lists each of the possible *Present_Value* states that shall be interpreted as *LIFE_SAFETY_ALARM*. The *Alarm_Values* property lists each of the possible *Present_Value* states that shall be interpreted as *OFFNORMAL*. The *Fault_Values* property lists each of the possible *Present_Value* states that shall be interpreted as *FAULT*. All other *Present_Value* states shall be interpreted as *NORMAL*. Transitions to any of the states may generate notifications if the corresponding flags are set in *Event_Enable*.

In the case of Access Point, the ACCESS_EVENT algorithm is applied for both Access Alarm Events and Access Transaction Events:

(1) Access Alarm Events

The Access_Alarm_Events property of the Access Point lists each of the Last_Access_Event values that shall be reported, when Last_Update_Time changes, as EVENT or ALARM notifications according to the Notify_Type, if the TO-NORMAL bit of Event_Enable is TRUE. The Event_State remains NORMAL.

The Notification Class object referenced by Notification_Class is used to distribute Access Alarm Events.

(2) Access Transaction Events

The Access_Transaction_Events property of the Access Point lists each of the Last_Access_Event values that shall be reported when Last_Update_Time changes, if the TO-NORMAL bit of Event_Enable is TRUE, as EVENT notifications ignoring Notify_Type. The Event_State remains NORMAL. The Event_Time_Stamps TO-NORMAL element is not affected. Acked_Transitions is not affected.

The Notification Class object referenced by Transaction_Notification_Class is used to distribute Access Transaction Events. If Transaction_Notification_Class is not present in the Access Point object, then the Notification Class object referenced by Notification_Class is used. Ack_Required of the respective Notification Class object is ignored and the value FALSE is conveyed in the AckRequired parameter of the event notification message.

Table 13-2. Standard Objects That May Support Intrinsic Reporting

Object Type	Criteria	Event Type
...
Accumulator	If Pulse_Rate exceeds range from Low_Limit through High_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable, OR Pulse_Rate returns to range from Low_Limit through High_Limit for longer than Time_Delay AND the new transition is enabled in Event_Enable and Limit_Enable	UNSIGNED_RANGE
Access Point	<i>(Access Alarm Events) If Update_Time changes and Last_Access_Event is equal to one of the values in the Access_Alarm_Events list</i>	ACCESS_EVENT
Access Point	<i>(Access Transaction Events) If Update_Time changes and Last_Access_Event is equal to one of the values in the Access_Transaction_Events list</i>	ACCESS_EVENT

Table 13-3. Standard Object Property Values Returned in Notifications

Object	Event Type	Notification Parameters	Referenced Object's Properties
...
Accumulator	UNSIGNED_RANGE	Exceeding_Value Status_Flags Exceeded_Limit	Pulse_Rate Status_Flags Low_Limit or High_Limit
Access Point	ACCESS_EVENT	Access_Event Status_Flags Update_Time Access_Credential Authentication_Factor (if present)	Last_Access_Event Status_Flags Update_Time Last_Access_Credential Last_Authentication_Factor (if present)

¹ This parameter conveys a reference to the Log_Buffer property of the Trend Log object.

Table 13-4. Notification Parameters for Standard Event Types

Event Type	Notification Parameters	Description
...
UNSIGNED_RANGE	Exceeding_Value Status_Flags Exceeded_Limit	The value that exceeded a limit The Status_Flags of the referenced object The limit that was exceeded
ACCESS_EVENT	Access_Event Status_Flags Update_Time Access_Credential Authentication_Factor (if present)	The new value of the referenced property The Status_Flags of the referenced object The new value of the referenced Update_Time property The Last_Access_Credential of the referenced object The Last_Authentication_Factor of the referenced object (if present)

[Change 13.3, p. 258]

13.3 Algorithmic Change reporting

...

The following event type algorithms are specified in this standard because of their widespread occurrence in building automation and control systems. They are

...

- (i) UNSIGNED_RANGE
- (j) ACCESS_EVENT

[Add new clause 13.3.X and Figure 13-X, renumbering subsequent figures, p. 264]

13.3.X ACCESS_EVENT Algorithm

An ACCESS_EVENT event occurs when the value of the property referred to by Update_Time_Reference changes and the referenced property is equal to one of the values contained in the List_Of_Access_Events. This type of event may only be applied to properties that are of type BACnetAccessEvent. For the purposes of event notification, ACCESS_EVENT events generate a TO-NORMAL transition.

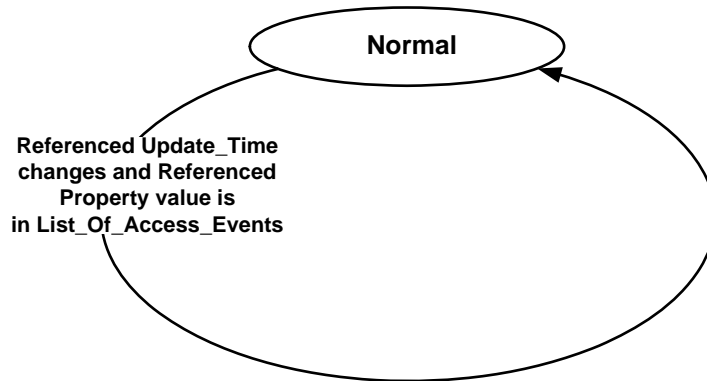


Figure 13-X. ACCESS_EVENT Algorithm

[Change BACnetEventParameter production, clause 21, pp. 415-416]

[Note: context tag 12 is used in Addendum 135-2004b-2.]

```

BACnetEventParameter ::= CHOICE {
  ...
  access-event      [13] SEQUENCE {
    list-of-access-events      [0] SEQUENCE OF BACnetAccessEvent,
    update-time-reference     [1] BACnetDeviceObjectPropertyReference
  }
}
  
```

[Change BACnetEventType enumeration, clause 21, p. 417]

[Note: enumeration 12 is used in Addendum 135-2004b-2.]

```

BACnetEventType ::= ENUMERATED {
  ...
  unsigned-range      (11),
  access-event        (13)
  ...
}
  
```

-- Enumerated values 0-63 are reserved for definition by ASHRAE. Enumerated values
 -- 64-65535 may be used by others subject to the procedures and constraints described
 -- in Clause 23. It is expected that these enumerated values will correspond to the use of the
 -- complex-event-type CHOICE [6] of the BACnetNotificationParameters production.

~~—The last enumeration used in this version is 11.~~

[Change **BACnetNotificationParameters**, clause **21**, pp. 419-420]

[Note: context tag 12 is used in Addendum 135-2004b-2.]

[Note: This could be due to an alphabetization issue in this table... is defined in Addendum 135-2004j-1.]

[Note: **BACnetAuthenticationFactor** is defined in Addendum 135-2004j-6.]

```
BACnetNotificationParameters ::= CHOICE {  
  -- These choices have a one-to-one correspondence with the Event_Type enumeration with the exception of the  
  -- complex-event-type, which is used for proprietary event types.  
  ...  
  access-event [13] SEQUENCE {  
    access-event           [0] BACnetAccessEvent,  
    status-flags          [1] BACnetStatusFlags,  
    update-time          [2] BACnetTimeStamp,  
    access-credential    [3] BACnetObjectIdentifier,  
    authentication-factor [4] BACnetAuthenticationFactor OPTIONAL  
  }  
}
```