

SEMI E132-0923 (Reapproved 0624) SPECIFICATION FOR EQUIPMENT CLIENT AUTHENTICATION AND AUTHORIZATION

This Standard was technically approved by the Information & Control Global Technical Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on April 26, 2024. Available at www.semiviews.org and www.semi.org in June 2024; originally published March 2004; previously published September 2023.

NOTICE: The section symbol (§) is used to reference a numbered section, paragraph, or header and all subordinate paragraphs, headers, and embedded material (i.e., EXCEPTIONs, lists) therein. The paragraph symbol (§) is used to reference a particular numbered paragraph and embedded material therein. When duplicated (i.e., §§ and §§) the symbols are used to reference more than one section or paragraph, respectively.

NOTICE: Paragraphs entitled ‘NOTE:’ are not an official part of this Standard or Safety Guideline and are not intended to modify or supersede the official Standard or Safety Guideline. These have been supplied by the global technical committee to enhance the usage of the Standard or Safety Guideline.

1 Purpose

1.1 This Standard specifies a method for restricting access to communication with equipment by requiring clients to authenticate to the equipment before proceeding with subsequent communication, and provides a flexible authorization scheme to control client access. Authorization allows parties who want to exchange information with or control the equipment to do so on a need-to-know and/or need-to-use basis.

1.2 The authorization scheme specified in this Standard allows equipment vendor the flexibility to provide access control at any level of granularity, ranging from no access control restrictions, predefined role-based access control, to very fine-grained control.

2 Scope

2.1 This Standard applies to manufacturing equipment that requires authentication and authorization for its services. It does not apply to communication that is governed by the SEMI E30 communication and control state models.

2.2 This Standard does not require data transmitted over an established session to be encrypted, encryption is only required as specified by the authentication protocol.

NOTICE: SEMI Standards and Safety Guidelines do not purport to address all safety issues associated with their use. It is the responsibility of the users of the Documents to establish appropriate safety and health practices, and determine the applicability of regulatory or other limitations prior to use.

3 Limitations

3.1 This Standard specifies the concept, behavior, and interfaces but does not specify details of the technologies for the implementation, such as the message, sequence, or schema of the data transferred. Subordinate Standards map specific technologies and protocols to this Standard.

3.2 The authentication specified in this Standard applies to computer applications. To prevent unauthorized access of equipment services, these computer applications should manage user-level access.

4 Referenced Standards and Documents

4.1 SEMI Standards and Safety Guidelines

SEMI E30 — Specification for the Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E128 — Specification for XML Message Structures

4.2 IETF® Documents¹

RFC 2396² — Uniform Resource Identifiers (URI): Generic Syntax

RFC 4122³ — A Universally Unique Identifier (UUID) URN Namespace

4.3 Other Documents

OMG®⁴ Specification 01-09-67 — Unified Modeling Language® (UML®) Specification, Version 1.4⁵

NOTICE: Unless otherwise indicated, all documents cited shall be the latest published versions.

5 Terminology

5.1 Abbreviations and Acronyms

5.1.1 LAN — Local Area Network

5.1.2 UUID — Universally Unique Identifier

5.2 Definitions

5.2.1 *Access Control List (ACL)* — a collection of ACL entries, each entry granting a subset of the available privileges to a specific subject.

5.2.2 *administration* — the process of configuring equipment client identities, associated credentials, and setting authorizations.

5.2.3 *Asymmetric Key Cryptography* — refer to Public Key Cryptography.

5.2.4 *authenticated communications* — communications with equipment compliant to this Specification to establish an authenticated session, to grant authorizations and for administration. This includes any communications over the authenticated session.

5.2.5 *authenticated session* — also referred to as session in this Specification, this is an application-level concept defined by this Specification and is independent of any physical or logical connection or communication protocol ‘session’ layer used in the actual exchange of messages. An authenticated session or session follows the session establishment procedure, state models, and the defined services as described in this Specification.

5.2.6 *authentication* — the process of determining whether a user or process is, in fact, who or what it is declared to be. In private and public computer networks, authentication is often achieved through the use of passwords. Knowledge of the password is assumed to guarantee that the user is authentic. This Standard refers only to the authentication of application processes, and does not support or address authentication of human users.

5.2.7 *authorization* — the process of granting the privilege to perform a specific action to a user or process.

5.2.8 *Client* — an application process that communicates with the equipment to access equipment resources and initiates the connection with the Equipment Server. The Client performs management functionality, and can receive notification and operational messages from the Equipment Server.

5.2.9 *communication session* — a series of two-way message exchanges between a client and the equipment. A session typically begins and ends at the request of the client.

5.2.10 *Consumer* — an application process that communicates with the equipment to get data. The Equipment Server initiates the connection to a specific endpoint to communicate with the Consumer. The Consumer does not perform any management functionality, can receive notification messages from the Equipment Server, and always receives operational messages from the Equipment Server when it exists.

¹ Internet Engineering Task Force, IETF Administration LLC, 1000 N West Street, Suite 1200, Wilmington, DE 19801, USA;
<http://www.ietf.org>. IETF trademark is owned by the IETF Trust.

² <http://www.ietf.org/rfc/rfc2396.txt>

³ <https://tools.ietf.org/html/rfc4122>

⁴ Trademark is owned by Object Management Group Inc.

⁵ http://www.omg.org/technology/documents/modeling_spec_catalog.htm

5.2.11 *cryptographic algorithm* — a transformation of meaningful data into some meaningless data, a process called encryption. If the transformation is reversible, then the reverse process is called decryption. Typically it is computationally difficult to correctly decrypt the data without knowing both the algorithm used and some additional information, referred to as keys.

5.2.12 *encryption* — refer to cryptographic algorithm.

5.2.13 *EDA Application* — software application that implements at least one Client or Consumer.

5.2.14 *Equipment Server* — software application that tracks authenticated sessions and provides data for the equipment.

5.2.15 *equipment user* — party who uses the equipment.

5.2.16 *guest session* — a session established by a Client whose specified clientId does not have a matching entry in the ACL. The Equipment Server supports guest sessions when it defines a PrivilegeAssignment or RoleAssignment ACLEntry for the reserved 'anyPrincipal'. An EDA application establishing a guest session is still authenticated.

5.2.17 *message digest* — refer to one-way hash.

5.2.18 *notification message* — type of message sent by the Equipment Server to notify the recipient something has changed on the equipment (typically related to status, management, or metadata).

5.2.19 *one-way hash* — a one-way transformation of an arbitrary length of data into a fixed-length code. The transformation is computationally difficult to reverse, and unlikely to have collisions. The same message will always result in the same hash, and any slight modification to the original message will result in a different hash, thus providing message integrity. The terms One-way Hash, Hash and Digest are used interchangeably in this Standard.

5.2.20 *operational message* — type of message sent by the Equipment Server to either a Client or Consumer with collected or analysis data from the equipment.

5.2.20.1 *Discussion* — SessionPing is a special type of operational message that requires a response that the Equipment Server sends to ensure it can communicate with the Client or Consumer.

5.2.21 *operational message recipient* — Client or Consumer associated with the session that is receiving operational messages for the session.

5.2.22 *plaintext* — unencrypted information that is input into a cryptographic algorithm.

5.2.23 *persistent session* — an authenticated session established that survives equipment shutdowns or power offs. Behavior of a persistent session during equipment shutdown and startup is as defined by the session state model described in this Standard.

5.2.24 *Public Key Cryptography (PKC)* — also known as Asymmetric Key Cryptography, a cryptographic algorithm that employs two keys for encryption and decryption of data. One key is designated as the public key and can be freely advertised, while the other key, designated as the private key, is never revealed. Data encrypted with one key can only be decrypted with the other key. PKC is widely used for key exchange and message signing.

5.2.25 *Secret Key Cryptography (SKC)* — also known as Symmetric Key Cryptography, a cryptographic algorithm that employs one single key for both encryption and decryption. SKC is widely used for bulk encryption of data due to its speed over PKC.

5.2.26 *secure communications* — communication between two entities where the content is protected using encryption.

5.2.27 *Security Admin Agent* — a Client that manages session access and privileges on the Equipment Server.

5.2.28 *session* — refers to an *Authenticated Session* unless specified otherwise. It has no relation to the sessions that may be defined elsewhere.

5.2.29 *Symmetric Key Cryptography* — refer to Secret Key Cryptography.

6 Conventions

6.1 *Unified Modeling Language (UML) Notation* — All class and sequence diagrams in this Standard make use of the Unified Modeling Language notation.

6.2 *Attribute Tables* — The table below provides an example of the tables used to list and describe attributes of classes defined in this Specification.

Table 1 Attribute Table Format

<i>Attribute Name</i>	<i>Definition</i>	<i>Form</i>
		Refer to ¶ 6.2.1.

6.2.1 *Form* — Defines the data type of the attribute. The terms used to describe data types used in this column are defined in the SEMI International Standards Compilation of Terms, or are included as part of the Specification. Refer to the compilation of terms for the definition of SEMI type name meanings.

6.2.2 *Association Tables* — The table below provides an example of the tables used to list and describe associations between classes defined in this Specification.

Table 2 Association Table Format

<i>Association Role Name</i>	<i>Definition</i>	<i>Comments</i>

6.2.2.1 *Association Role Name* — The name of the association role being specified.

6.2.2.2 *Definition* — Describes the function or purpose of the association.

6.2.2.3 *Comments* — Any additional comments or notes regarding the association.

6.3 *Operation Definition Tables* — The table below provides an example of the tables used to list and describe the interface operations defined in this Specification.

Table 3 Operation Definition

<i>Operation</i>	<i>Description</i>	<i>Type</i>	<i>Requestor/Sender</i>	<i>Responder/Receiver</i>
		Refer to ¶ 6.3.2.		

6.3.1 *Operation* — Specifies the name of the operation.

6.3.2 *Type* — Specifies the messaging semantics of the operation. Only Request-Response (RR) and Fire-and-Forget (FF) semantics are used in this Specification.

6.3.3 *Requestor/Sender* — For RR semantics, identifies the entity that makes the request, for FF semantics, identifies the entity that sends the message. Can be ‘Client’, ‘Security Admin Agent’, ‘Equipment Server’ or ‘Client Component’. ‘Client Component’ can be any software component communicating to the Equipment Server to query interface metadata, compliant to this Specification.

6.3.4 *Responder/Receiver* — For RR semantics, identifies the entity that responds to the request, for FF semantics, identifies the entity that receives the message. Can be ‘Client’, ‘Consumer’ or ‘Equipment Server’.

6.4 *Operation Parameter Definition Table* — The table below provides an example of the tables used to list and describe parameters for interface operations defined in this Specification.

Table 4 Operation Parameter Definitions

<i>Parameter</i>	<i>Description</i>	<i>Kind</i>	<i>Form</i>
		Refer to ¶ 6.4.2.	Refer to ¶ 6.4.3.

6.4.1 *Parameter* — Specifies the name of the parameter.

6.4.2 *Kind* — Specifies whether the parameter is an ‘in’, ‘out’, or ‘error’ parameter for the operation. ‘error’ parameters always function as ‘out’ parameters, but indicate that the operation did not complete successfully. Not all errors possible for a specific operation will be listed, only select typical ones are.

6.4.3 *Form* — Defines the data type of the parameter. The terms used to describe data types in this column are defined in the SEMI International Standards Compilation of Terms, or are included as part of the Specification. Refer to the compilation of terms for the definition of SEMI type name meanings.

6.4.3.1 When error message requirements are included in the Form column, the requirement identifies specific information to include in the error text. Required text shall be case sensitive. Implementations can include additional text (e.g., if the Form content is “Text, shall include the text ‘Unrecognized equipment identifier’ and the bad equipment identifier.”, then the error message needs to include both mentioned elements, and the requirement does not state the elements need to be organized in any specific way.

7 Communication Background

NOTE 1: In this section, client refers to the communication client-server architecture and the client application refers to the software application that is the client.

7.1 In conventional semiconductor equipment communication, such as that defined by SEMI E30, it is assumed that only a single client application can gain access to the equipment for communication. This assumption was valid when the physical transport medium was based on a serial cable connection to the equipment. In such an environment, it is possible to ensure that only a single client application is permitted to interface with the equipment by only providing the serial connection to the machine on which the application runs.

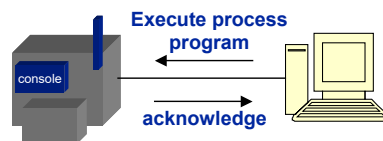


Figure 1
Equipment Communication via Serial Cable

7.2 As equipment communication moves from a serial cable medium to a local area network (LAN) such as Ethernet, it becomes possible for any client on the network to send requests to the equipment, provided there is a path from the client node to the equipment using encryption. Without taking specific measures to secure communication with the equipment, it is not possible to ensure that only authorized clients are permitted to communicate with the equipment.

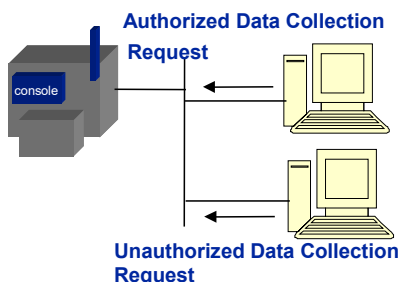


Figure 2
Equipment Communication via LAN

7.3 Even if the equipment accepts only a single connection request from the network, it is still vulnerable to a race condition among any clients that attempt to connect. This results in a ‘first-come, first-served’ policy in which the first client to attempt a connection with the equipment is granted access, even if that client is not authorized to communicate

with that specific equipment. Such an environment is insufficient for providing adequate protection of equipment resources.

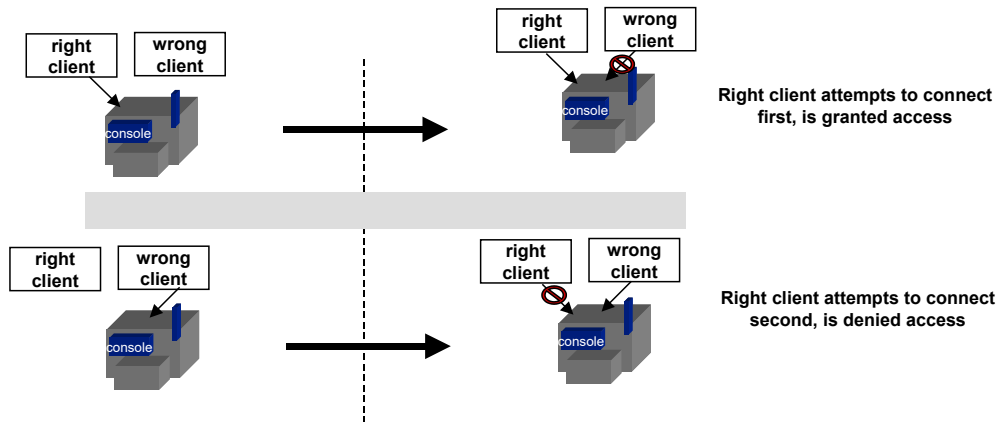


Figure 3
First-Come First-Served Communication

7.4 A more secure technique to ensure that only authorized clients are allowed to utilize equipment resources is to require clients to first prove their identity to the equipment. If the equipment knows a client's identity, then the equipment may be able to assume that the client is authorized to proceed with further requests.

8 Security Administration Overview

NOTE 2: In this section, client refers to the security administration client-server architecture and the client application refers to the software application that is the client.

8.1 This Standard defines a mechanism for clients of manufacturing equipment to authenticate to the equipment prior to any subsequent communication. To facilitate this, both the authentication procedure itself and a means for establishing client identities with the equipment beforehand is defined. This Standard also defines a flexible scheme for authorization and mechanisms to support a centralized administration model.

8.2 *Central Administration Model* — A conceptual view of a central security administration model is presented in Figure 4 below.

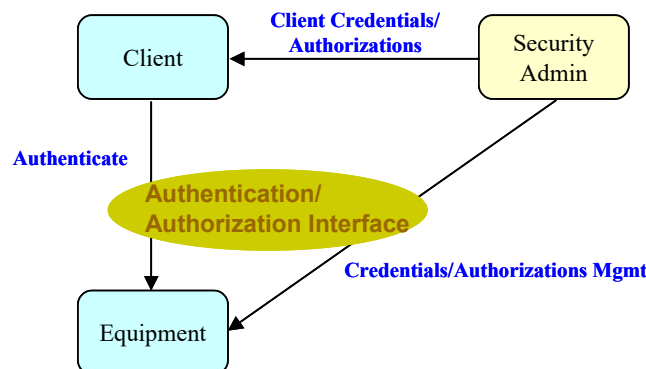


Figure 4
Security Administration Model

8.2.1 *Security Admin* — The security admin issues client credentials, establishes client authorizations and possibly client credentials on the equipment, defines custom roles and their authorizations. Client credentials are identifying

information for the client. The security admin manages equipment authorizations and client credentials using an administration interface defined in this Specification. The interface between security admin and client to communicate the credentials will not be defined in this Specification, and may be as simple as configurations inside the client application.

8.2.2 Client — Within the context of security administration, the client holds any issued credentials assigned to it, and authenticates to the equipment to establish an authenticated session.

8.2.3 Equipment — Within the context of security administration, the role of equipment in the administration model is to establish the security admin entity for the equipment, hold any defined authorizations, authenticate clients to establish a session, determine client authorization levels and enforce them for the established session.

8.2.4 Interfaces — The authentication, administration, and session establishment interfaces will be defined in later sections along with the concepts of establishing client identities and authorizations, though the actual technology and protocols used maybe defined in further detail in a separate adjunct standard.

8.3 Challenge/Response Authentication

8.3.1 The authentication process begins when a client first attempts to establish a communication session with the equipment. Upon receiving this request, the equipment challenges the client to prove its identity to the equipment. The client shall encrypt some session-generated data using a private or secret key (e.g., a password) that only the claimed client would know. The equipment can verify client's claim using the corresponding public key or shared secret key assigned to the client to decrypt the data, if decrypted data matches plaintext then client is verified.

8.3.2 For communication technologies in which a separate outgoing connection is established in order for the equipment to send events, the equipment shall also verify that the receiving address is valid. This is achieved by sending a ping message to the session endpoint (refer to ¶ 15.6.1).

8.4 Authorization

8.4.1 Any client application wishing to invoke privileged services on the equipment shall be authorized to do so. Authorization is implemented by granting access control privileges beforehand to defined clients and later checked against the client's requests. The user or process shall be authenticated before authorization takes place. The general flow is as follows:

- *Authentication and Credentials Exchange* — Client application authenticates to equipment as to its identity and establishes an authenticated session.
- *Session Established* — Equipment optionally caches client application's privileges and associates these privileges with the newly-created session.
- *Service Request* — Client application makes a service request.
- *Privilege Validation* — Equipment validates the client application's request against its assigned privileges.
- *Session Closed* — Client application closes session, all session state information is reset or cleaned.

8.4.2 Since the client does not authenticate itself for each service request, privilege validation is performed against the authenticated session created for that client. This does not necessarily mean that the implementation of a session requires the use of a stateful or connection-oriented communication protocol, or that authentication is performed during every protocol-level reconnect. If the implementation protocol is connectionless, the session can be identified by some mutually agreed upon session identifier, negotiated during the authentication process required to establish a session.

NOTE 3: This Specification does not define a mechanism for controlling end-user access to equipment services or information. The authentication and authorization scheme defined in this Specification applies at the application level only. Management of end-user access to equipment functions and information is strictly the responsibility of applications that make use of the authenticated communication sessions defined in this Specification.

9 Authentication

NOTE 4: In this section, client refers to the authentication client-server architecture and the client application refers to the software application that is the client.

9.1 *Authentication Message Flow* — The authentication message flow during a normal session establishment is presented in Table 5. Arrows represent the major messages exchanged during the handshake, with notes clarifying the key data exchanged in each message. Depending on the actual messaging/transport protocols and encryption algorithms used, some additional messages or data parameters may be added, but the concept remains the same. Specifics of algorithms and protocols will be detailed in a separate technology mapping specification. The terms server and equipment are used interchangeably.

Table 5 Client Authentication Scenario

<i>Client</i>	<i>Direction</i>	<i>Equipment Server</i>
Request a challenge token for a specific equipment and client ID.	→	
	←	Return a challenge token.
Request a session. Use the challenge token to provide information for authentication.	→	Verify the session request.
	←	Session authenticated.

9.2 Session Identifier

9.2.1 The session id returned from the EstablishSession request (refer to ¶ 13.2.6) can be used to identify the session for subsequent communications and to enforce authorizations. When possible, the identifier shall be embedded in special headers in the message transport, and not as formal parameters in the communication messages thus preserving their original structure as defined by the respective interface specifications. The session identifier shall be present in all requests, except when querying for the equipment's information or requesting a session. The session identifier in the special protocol header shall be empty for the GetEquipmentInformation(), EstablishSession(), and GetInterfaces() operations.

9.2.2 The SessionId shall be a Universally Unique Identifier (UUID) that is guaranteed unique across space and time, refer to the documents referenced in ¶ 4.2 for specific details on syntax and format of a UUID.

9.2.3 Subordinate Standards capture when and how the SessionId should be hashed to protect it when it is transmitted.

9.3 Equipment Identifier

9.3.1 The equipment identifier (equipmentId) identifies an equipment represented in the Equipment Server.

9.3.1.1 Typically, each Equipment Server supports one equipment.

9.3.1.2 Some implementations may have one Equipment Server instance supporting multiple equipment. Each equipment is represented by an equipment identifier, and has its own set of metadata, ACL entries, and sessions associated with it.

9.3.2 The equipment identifier is reported as part of operation GetInterfaces in the InterfaceDiscovery interface.

9.3.3 The equipment identifier can be provided as part of the SEMI E132 operations.

9.3.3.1 If the equipment identifier is not explicitly defined as part of an operation's interface (i.e., in the operation's list of parameters), the equipment identifier shall be embedded in special headers in the message transport, and not as formal parameters in the communication messages thus preserving their original structure as defined by the respective interface specifications.

9.3.3.2 The equipment identifier is passed as input parameters to operations GetEquipmentInformation and EstablishSession in the SessionManager interface. In all other operations, including the equipment identifier is allowed but not mandatory.

NOTE 5: Some implementations may include the equipment identifier in all operations so that it is recorded in message logs to help with diagnostics.

9.3.3.3 When the equipment identifier is present in the special message header, the equipment identifier shall be equal to the equipment identifier that was specified in the EstablishSession operation when the session was established. Otherwise the error ‘Unrecognized Equipment Identifier’ is returned.

9.3.4 Once a session is established, all operations apply only to the equipment identifier that was specified in the EstablishSession operation when the session was established.

9.3.5 The equipment identifier shall be a string of length 1 to 256 characters restricted to the ASCII characters from 20₁₆ through 7E₁₆ excluding the following ASCII characters:

- question mark '?' (3F₁₆)
- asterisk '*' (2A₁₆)
- tilde '~' (7E₁₆)
- 'greater than' symbol '>' (3E₁₆)
- 'less than' symbol '<' (3C₁₆)
- colon ':' (3A₁₆)
- backward slash '\' (5C₁₆)
- forward slash '/' (2F₁₆)
- vertical bar '|' (7C₁₆)

9.3.5.1 The equipment identifier shall not start or end with the space character ' ' (20₁₆).

9.3.5.2 The equipment identifier shall be case sensitive.

9.3.6 When an Equipment Server has multiple equipment identifiers, each equipment identifier shall be a unique string within the Equipment Server.

10 Authorization

NOTE 6: In this section, client refers to the authorization client-server architecture.

10.1 Authorization Overview

10.1.1 Any client wishing to invoke privileged services on the equipment shall be authorized to do so. Authorization is implemented by granting access control privileges beforehand to defined clients and later checked against the client's requests. The total set of available privileges may include standardized service-level access control, such as privileges to invoke specific services or functions, but may also include supplier-defined finer-grained privileges. A default privilege to allow access to all services available from an authenticated session is defined in this Specification, and shall be supported by all implementations. A user of the equipment therefore has the option to reduce security enforcement on the equipment to authentication alone, achieved by assigning this all-access privilege to all defined clients.

10.1.2 Figure 6 provides an abstract overview of the relationships among the Role, Principal, and Privilege concepts defined in this Specification. A Principal is either assigned to one and only one Role, or to one or more Privileges. A Role is defined as a uniquely-identifiable set of 1 or more Privileges, and cannot be composed from other Roles. Zero or more Principals can be assigned to at most one Role. These relationships are enforced through the management of Access Control List entries, described in ¶ 10.2.

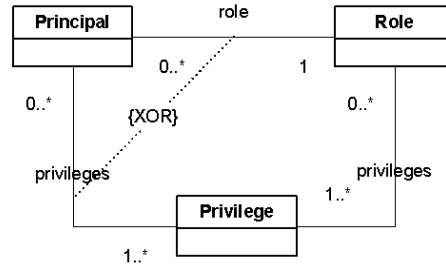


Figure 5
Abstract Model of ACL

10.2 Access Control List

10.2.1 Access control privileges are assigned to defined clients (principals) by using an Access Control List (ACL). An ACL is a collection of ACL entries, each entry granting a subset of the available privileges to a specific subject. A subject can represent a specific principal, a uniquely-named role, or all principals that successfully authenticate with the equipment. A principal can have privileges granted to it through either an explicit list of privileges, or indirectly through assignment to a role. The equipment shall maintain the ACL defined in this Specification in nonvolatile memory.

10.2.2 Roles are used to simplify the management of privilege assignments to principals. Each unique role has an ACL entry that specifies the privileges assigned to that role. Those privileges can then be granted to principals through a RoleAssignment ACL entry for each principal that should have those privileges. A principal shall either have a RoleAssignment or an explicit list of privileges assigned to it (PrivilegeAssignment); it is not permitted to have both.

10.2.3 The equipment shall refuse session requests from clients that have no ACLEntry, either explicitly through a role or privilege assignment, or implicitly through the existence of an ACLEntry for all authenticated principals.

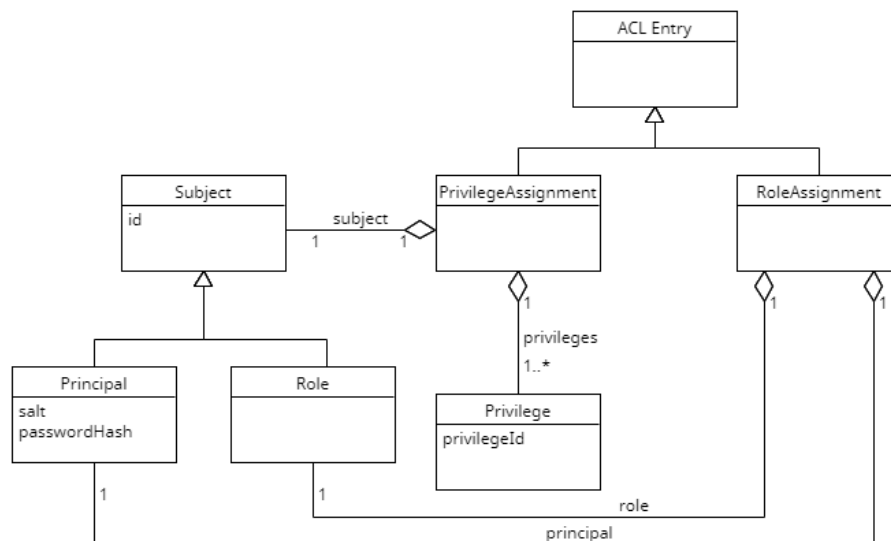


Figure 6
ACL Entry Description

10.2.4 ACLEntry

10.2.4.1 Figure 7 shows the classes used for requesting the creation of a new ACLEntry on the equipment (refer to ¶ 12.3.2.3). The ACLEntry is an abstract base class for the two kinds of ACL entries, PrivilegeAssignment, and RoleAssignment. There can be only one ACLEntry for each subject. Attempts to add a second entry for same subject shall generate a duplicate entry error (refer to ¶ 12.3.2.3). This class has no attributes or associations.

10.2.5 PrivilegeAssignment

10.2.5.1 This class defines an association between a given subject (either a Principal or a Role) and its set of assigned privileges on the equipment. A PrivilegeAssignment for a Principal establishes all privileges that are specifically granted to the Principal. A given Principal shall either have a PrivilegeAssignment ACLEntry or a RoleAssignment ACLEntry (refer to ¶ 10.2.6); it cannot have one (or more) of each. A PrivilegeAssignment for a Role defines the role itself and the privileges that will be granted to any Principal that is assigned to that Role (refer to ¶ 10.2.6). There can be only one PrivilegeAssignment ACLEntry for a given Role.

Table 6 PrivilegeAssignment Association Definitions

<i>Association Role Name</i>	<i>Definition</i>	<i>Comments</i>
subject	The Principal or Role that is being granted privileges.	One and only one element of any type derived from Subject, described in ¶ 10.2.7.
privileges	The set of privileges being granted.	List of one or more elements of type Privilege, described in ¶ 10.2.10.

10.2.6 RoleAssignment

10.2.6.1 This class assigns a specific Principal to a Role. The identified Role shall have a PrivilegeAssignment ACLEntry elsewhere in the ACL. All privileges assigned to the identified Role are granted to the identified Principal's sessions. A given Principal shall either have a RoleAssignment ACLEntry or a PrivilegeAssignment ACLEntry (refer to ¶ 10.2.5); it cannot have one (or more) of each.

10.2.6.2 A PrivilegeAssignment ACLEntry for a Role cannot be deleted from the ACL until all Principals that were assigned to that Role have been removed from it by deletion of their RoleAssignment ACLEntry. Any authenticated sessions that were established for such Principals prior to the deletion of their RoleAssignment ACLEntry shall continue to be operable with that Role's privileges until the session is terminated. Subsequent attempts by such Principals to request sessions shall be rejected unless there is a default set of privileges assigned to the reserved 'anyPrincipal' (refer to ¶ 10.2.8). Otherwise, in order for session requests to be accepted, these Principals will have to have specific privileges assigned to them directly through a PrivilegeAssignment ACLEntry, or indirectly through assignment to a new Role.

Table 7 RoleAssignment Association Definitions

<i>Association Role Name</i>	<i>Definition</i>	<i>Comments</i>
principal	The principal that is being assigned to the Role.	One and only one element of type Principal, described in ¶ 10.2.8.
role	The Role to which the principal is being assigned.	One and only one element of type Role, described in ¶ 10.2.9. The Subject id for the identified Role shall be equal to the Subject id of a PrivilegeAssignment for a Role appearing elsewhere in the ACL.

10.2.7 Subject

10.2.7.1 This is an abstract base class used to represent any entity that can be granted privileges in an ACL entry. This can be either a Principal or a Role.

10.2.7.2 The id is a text field that identifies a particular Subject.

10.2.7.2.1 If the Subject is a Principal, then the id is equal to that Principal's unique id used during authentication. If the Subject is a Role, then the id is equal to the name of the Role.

10.2.7.3 The id shall be a string of length 1 to 128 characters restricted to the ASCII characters from 20₁₆ through 7E₁₆.

10.2.7.3.1 The id shall not start or end with the space character ' ' (20₁₆).

10.2.7.4 The id shall be case sensitive.

Table 8 Subject Attribute Definitions

<i>Attribute Name</i>	<i>Definition</i>	<i>Form</i>
id	Identifies a Principal or Role.	Text

10.2.8 Principal

10.2.8.1 This class represents a specific Principal that can be authenticated and assigned privileges. The principal shall have a salt value and a passwordHash attributes. Both attributes are used as part of session authentication.

10.2.8.2 To be effective, the salt value should be random rather than known, predictable text. (e.g., do not use the equipmentId.) Using a Cryptographically Secure Pseudo-Random Number Generator (CSPRNG) is recommended.

10.2.8.3 Each ACL Entry has an ACL Password. Even if the plaintext ACL password is an empty string, the passwordHash will be a non-null and non-empty value.

10.2.8.4 When an ACLEntry is created by a security admin and when an ACLEntry password is changed, the Equipment Server does not receive a plaintext ACL password in the corresponding operation. Instead, the Equipment Server receives the passwordHash. In these cases, the Equipment Server should not require knowledge of the plaintext ACL password.

10.2.8.5 If the equipment supplier supports creating an ACLEntry at the Equipment Server through a user interface communicating directly to the Equipment Server (outside of SEMI E132), the Equipment Server should not persist the plaintext ACL password used to generate the passwordHash.

Table 9 Principal Attribute Definitions

<i>Attribute Name</i>	<i>Definition</i>	<i>Form</i>
salt	The salt value for this ACLEntry. When adding an ACLEntry, a null or empty value is not allowed.	
passwordHash	Hashed version of the plaintext ACL password used to authenticate the principal. When adding an ACLEntry, null and empty string values are not allowed.	Text.

10.2.8.6 This Specification defines a reserved Principal id that represents any Principal that successfully authenticates to the equipment. The value of the id for this reserved Principal is 'urn:semi-org:auth:anyPrincipal'. All implementations shall support this reserved Principal. If an equipment user's security policy permits the approach, a PrivilegeAssignment or RoleAssignment ACL entry can be created for the reserved Principal to assign a default set of privileges to any client that successfully authenticates, rather than creating an ACL entry for each Principal.

10.2.8.7 Each client that attempts to authenticate to the equipment shall have an id that matches the id of one PrivilegeAssignment or RoleAssignment ACLEntry for that Principal, or else there shall be one PrivilegeAssignment or RoleAssignment ACLEntry for the reserved 'anyPrincipal'. If there are no such entries for the authenticating client, the equipment shall reject the session request. If there is a PrivilegeAssignment or RoleAssignment ACLEntry for both the specific Principal and the reserved 'anyPrincipal', the equipment shall accept the session request, and shall

grant only those privileges assigned to the specific Principal to the session. Privileges assigned to the reserved 'anyPrincipal' shall not be granted to such sessions.

10.2.8.8 A session established using the PrivilegeAssignment or RoleAssignment ACLEntry for the reserved principal 'urn:semi-org:auth:anyPrincipal' is called a guest session.

10.2.9 Role

10.2.9.1 This class represents a Role that can be assigned privileges. Roles can simplify the assignment of privileges to Principals. This is achieved by creating a single PrivilegeAssignment ACL entry for the Role containing the appropriate privileges, and then assigning each Principal (or the reserved 'anyPrincipal') to that Role using a RoleAssignment ACL entry.

10.2.10 Privilege

10.2.10.1 This class represents a specific privilege that can be granted to subjects as part of an ACL entry. All defined privileges supported by the equipment can be obtained using the GetDefinedPrivileges() operation of the SecurityAdmin interface. How each privilege is applied to a subject is determined by the specification or equipment supplier that defines the privilege. The privilege id is a text field that identifies a particular privilege and shall be unique across all privileges defined on the equipment. Permission and privilege is used interchangeably in this Specification.

10.2.10.2 This Specification defines a reserved Privilege id that represents all Privileges that are supported by the equipment, with exception of the reserved SecurityAdmin Privilege defined later. The value of the id for this reserved Privilege is 'urn:semi-org:auth:allPrivileges'. All implementations shall support this reserved Privilege. If an equipment user's security policy permits the approach, an ACL entry can be created for any Principal or Role that assigns this reserved Privilege to that Principal or Role. Principals or Roles that are assigned this privilege are granted all privileges supported by the equipment, this does not include the special SecurityAdmin Privilege defined later. A PrivilegeAssignment ACL entry containing this privilege cannot include any other privileges supported by the equipment in the entry.

Table 10 Privilege Attribute Definitions

<i>Attribute Name</i>	<i>Definition</i>	<i>Form</i>
privilegeId	Identifies a specific privilege.	Text, shall be equal to either the reserved value 'urn:semi-org:auth:allPrivileges', or to the 'privilegeId' attribute of one of the DefinedPrivileges supported by the equipment (refer to Table 30).

10.2.10.3 This Specification also defines a reserved Privilege id that represents the privileges of the SecurityAdmin as described in ¶ 12.2.

10.2.10.3.1 The value of the id for this reserved Privilege is 'urn:semi-org:auth:securityAdminPrivileges'.

10.2.10.3.2 All implementations shall support this reserved Privilege.

10.2.10.3.3 A Principal can be assigned this privilege, either through a RoleAssignment or PrivilegeAssignment ACLEntry. Once assigned, that Principal is permitted access to all operations defined for the SecurityAdmin interface.

10.2.10.3.4 This privilege shall not be assignable to the reserved 'anyPrincipal' (either through a RoleAssignment or PrivilegeAssignment ACLEntry). Only Clients that have an explicit ACL Entry are permitted to access the SecurityAdmin interface.

10.2.10.3.5 The reserved allPrivileges identifier defined previously, which can be assigned to any number of Principals, does not include this SecurityAdmin privilege.

10.2.11 ACL Passwords

10.2.11.1 ACL passwords are used to ensure an entity is allowed to access the ACLEntry because it knows both the ID and its associated plaintext ACL password (which is a secret) or hashed password. For example, someone who happens to know that the equipment has a principal called 'admin' shouldn't be allowed to use it if they do not know the ACL password associated with that principal on that specific equipment.

10.2.11.2 The only way to properly protect ACL password information is to use secure communications between the Client and the Equipment Server. Otherwise a malicious actor could intercept the network traffic and try to reuse ACL password information. Refer to ¶ 13.3 for more details.

10.2.11.3 The plaintext ACL password can be null or an empty string. If the plaintext ACL password is a null value, an empty string shall be used when generating the passwordHash.

10.2.11.4 Implementations shall use a one-way hash function using the plaintext ACL password and a salt value to generate the passwordHash.

10.2.11.4.1 Plaintext ACL passwords are not transmitted between the Client and the Equipment Server.

10.2.11.5 A one-way hash function takes an arbitrary length of data and converts it to a fixed length binary sequence and it is difficult to get the original value from the binary sequence. Subordinate Standards shall specify the specific one-way hash function to use and how the salt value is used in the hashing process.

NOTE 7: Equipment Server implementations may choose to use the one-way hash and a second, different salt value to protect the passwordHash before it is saved to its nonvolatile storage. This is a good security practice to adopt as it protects the data sent by the Client if an attacker gets access to the equipment's storage system (i.e., the attacker will not know the value that the Client uses in the authentication process).

10.2.11.6 The equipment supports guest sessions when there is a PrivilegeAssignment or RoleAssignment ACLEntry for the reserved 'anyPrincipal'.

10.2.11.6.1 The reserved 'anyPrincipal' has an ACL Password so any Client establishing a guest session needs to know the corresponding ACL password to authenticate successfully.

10.2.11.6.2 Only principals with the reserved SecurityAdmin privilege can change the ACL password associated with the reserved 'anyPrincipal'.

10.2.11.7 This Specification defines a reserved Privilege id that represents the ability to change the ACL password.

10.2.11.7.1 The value of the id for this reserved Privilege is 'urn:semi-org.auth:changeAclPasswordPrivileges'.

10.2.11.7.2 All implementations shall support this reserved Privilege.

10.2.11.7.3 A principal assigned this privilege may change its ACL password with the ChangeACLPassword() operations defined in the SessionManager interface.

10.2.11.7.4 The ChangeACLPassword privilege cannot be assigned to the reserved 'anyPrincipal' ACLEntry.

10.2.11.7.5 The reserved SecurityAdmin privilege identifier includes the ChangeACLPassword privilege. A principal with the reserved SecurityAdmin privilege identifier can change the ACL password for any principal and the reserved 'anyPrincipal'.

10.2.11.7.6 The reserved allPrivileges privilege identifier (which can be assigned to any number of principals) does not include this ChangeACLPassword privilege.

11 Clients and Consumers

11.1 With a networked environment, it is possible for software applications to be on different machines and fulfil different functions for the equipment user.

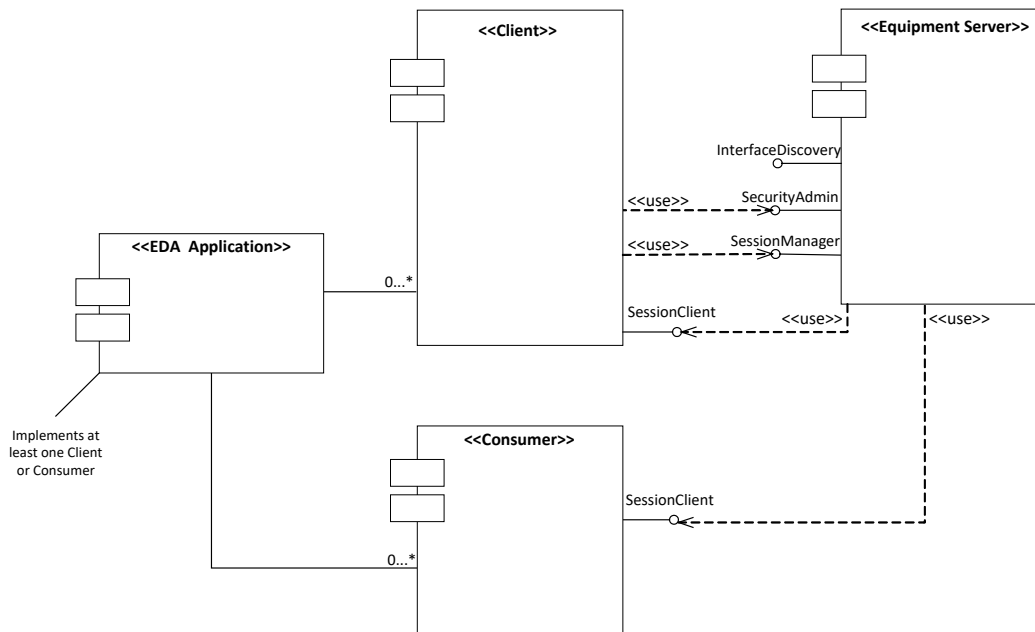


Figure 7
Session Interfaces Architecture

11.2 *Client and Consumer Conventions* — For this Specification, the following conventions are used:

11.2.1 *Client* — This application process initiates the connection to the Equipment Server.

- It is associated with one SessionId.
- It uses the SessionManager interface to perform management functionality (e.g., establishing a session). The Client can query if the Equipment Server is alive through the SessionManager interface.
- The Client can receive notification messages from Equipment Server by implementing the SessionClient interface.
- The Client can receive operational messages from the Equipment Server and responds to SessionPing messages sent from the Equipment Server by implementing the SessionClient interface. (When a Consumer is specified for the session, the Client does not receive operational messages.)

11.2.2 *Consumer* — The Equipment Server initiates the connection to this application process.

- It is associated with one SessionId.
- It does not do any management functionality or query if the Equipment Server is alive since it does not use the SessionManager interface.
- The Consumer always implements the SessionClient interface.
- Can receive notification messages.
- When present, the Consumer always receives operational messages from the Equipment Server and responds to SessionPing messages sent by the Equipment Server. (When the Consumer is specified for the session, the Client cannot receive operational messages.)

11.2.3 *Equipment Server* — This software application implements the InterfaceDiscovery, SecurityAdmin and SessionManager functionality. It uses the SessionClient interface to send notification and operational messages to the appropriate recipients (i.e., Clients and Consumers).

11.2.4 *EDA Application* — This software application implements one of the following scenarios:

- one or more Clients

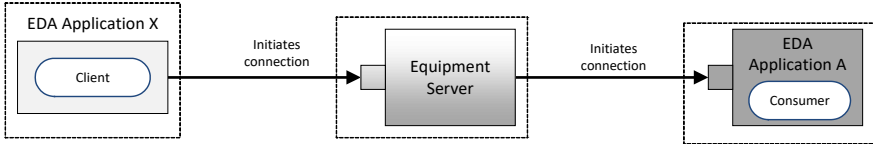
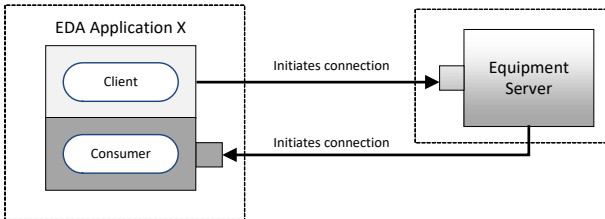
- one or more Consumers
- one or more Clients and one or more Consumers

11.2.5 *Security Admin Agent* — This is a special version of the Client that is allowed to use the SecurityAdmin interface via the security administration account.

NOTE 8: The Security Admin Agent may choose not to use the session management, notification or operational message functionality of the Client, but it requires a SessionId obtained from the SessionManager interface to use the SecurityAdmin interface.

11.3 *Client and Consumer Use Case Scenarios* — For this Specification, the following use cases are supported:

Table 11 Clients and Consumer Use Case Scenarios

Scenario	Architecture
1) No one gets operational messages	EDA Application X has the Client; it does not get operational messages. There is no Consumer. Equipment Server does not send SessionPing messages. 
2) Same entity gets operational messages	EDA Application X has the Client that gets operational messages. There is no Consumer. Equipment Server sends SessionPing messages to the Client. This scenario is only supported by implementations that use a protocol that supports bidirectional messages. 
3) Separate entity gets operational messages (Equipment initiates communications)	EDA Application X has the Client. EDA Application A has the Consumer that gets operational messages (through the specified endpoint). Equipment Server sends SessionPing messages to the Consumer. This scenario supports the Client changing the session's endpoint (where operational messages are sent for the Consumer).  Variation – EDA Application X has both the Client and the Consumer. The Equipment Server does not treat this variation any differently. 

11.4 *Common Errors* — Common errors independent of this Specification can be reported when they are applicable and not explicitly defined as SEMI E132-specific errors.

11.4.1 A Subordinate Standard shall identify how common errors are used with the implementation technology described in that Subordinate Standard.

11.4.2 Common errors defined in Table 15 shall be supported by all SEMI E132 operations. (For example, the EstablishSession() operation can return the common error ‘System shutting down’, even though it is not explicitly listed as an error parameter in Table 43, EstablishSession Parameter Definitions.)

Table 12 Common Errors Parameter Definitions

<i>Parameter</i>	<i>Description</i>	<i>Kind</i>	<i>Form</i>
error	System not ready.	error	Text, shall include the text ‘System not ready’.
error	System shutting down.	error	Text, shall include the text ‘System shutting down’.
error	Specified configuration is invalid.	error	Text, shall include the text ‘Specified configuration is invalid’, and a description of the invalid configuration.

12 Administration

12.1 Administration Overview

12.1.1 As the number of clients and resources to manage across a factory scales up, support for a centralized security administration model becomes a critical capability. This Specification will not describe how a factory can implement such a model for their network, but will specify standard mechanisms to enable this capability. This Specification describes a security administration interface to allow remote management of client credentials and authorizations, such as from a central location. Only concepts and behavior are described, it is expected that a separate technology adjunct standard will be created describing the specific message structure, protocols and algorithms that will used.

12.2 Security Admin

12.2.1 The equipment shall support at least one security administration account.

12.2.1.1 This Specification intentionally does not specify fix a security administration account name for security reasons.

12.2.1.2 If there is a generic security admin account provided by the equipment supplier, the initial security admin credentials shall be established at the time of equipment install via some method independent of this Specification. (This can be a local operator interface or provided by equipment vendor as a factory preset.) If the admin credentials are factory preset, then there shall be some means for end-user to re-establish its credentials.

12.2.1.3 This Specification requires that only the security administration accounts can utilize the services defined in the security administration interface (SecurityAdmin interface). Security admin privileges are granted via a reserved privilege id.

12.2.1.4 The equipment shall only support one active security admin session at a time.

12.3 Security Administration Interface

12.3.1 Figure 8 summarizes the standard operations available through the security admin interface to any Client authenticated as the security admin.

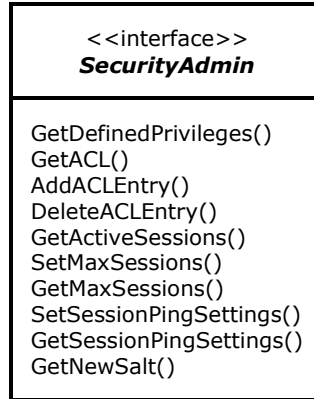


Figure 8
Security Administration Interface

12.3.2 Security Administration Operations

Table 13 Security Administration Operations

<i>Operation</i>	<i>Description</i>	<i>Type</i>	<i>Requestor/ Sender</i>	<i>Responder/ Receiver</i>
GetDefinedPrivileges	Requests for the list of all privileges defined on the Equipment Server.	RR	Security Admin Agent	Equipment Server
GetACL	Requests for the collection of ACL entries defined on the Equipment Server.	RR	Security Admin Agent	Equipment Server
AddACLEntry	Adds an ACL entry on the Equipment Server.	RR	Security Admin Agent	Equipment Server
DeleteACLEntry	Deletes an ACL entry on the Equipment Server.	RR	Security Admin Agent	Equipment Server
GetActiveSessions	Requests for information on all active sessions.	RR	Security Admin Agent	Equipment Server
SetMaxSessions	Sets maximum limit on number of non-admin sessions allowed.	RR	Security Admin Agent	Equipment Server
GetMaxSessions	Requests for the maximum limit of non-admin sessions currently set on the Equipment Server.	RR	Security Admin Agent	Equipment Server
SetSessionPingSettings	Request the SessionPing settings to be set for all current and future sessions.	RR	Security Admin Agent	Equipment Server
GetSessionPingSettings	Request the current values of the SessionPing settings to be returned.	RR	Security Admin Agent	Equipment Server
GetNewSalt	Returns a salt value generated by the Equipment Server	RR	Security Admin Agent	Equipment Server

12.3.2.1 *GetDefinedPrivileges* — This operation retrieves the set of privileges supported by the equipment manufacturer. The list shall include the reserved ‘urn:semi-org.auth:allPrivileges’ and ‘urn:semi-org.auth:securityAdminPrivileges’ privileges as defined in this Specification (refer to ¶ 10.2.10). This list shall also include the reserved ‘urn:semi-org.auth:changeAclPasswordPrivileges’ privilege as defined in this Specification (refer to ¶ 10.2.11.7).