SEMI E132.2-0422
SPECIFICATION FOR PROTOCOL BUFFERS FOR EQUIPMENT CLIENT AUTHENTICATION AND AUTHORIZATION (ECA)

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 September 7, 2021. Available at www.semiviews.org and www.semi.org in April 2022.

1 Purpose

1.1 The purpose of this Specification is to provide an implementation mapping of SEMI E132 to gRPC®[1] and Protocol Buffers.

2 Scope

2.1 The scope of this Specification is the faithful representation of the messages and services to support SEMI E132 with gRPC and Protocol Buffers. It will not add new concepts to the SEMI E132 model. The only additions made are those needed to render useful Protocol Buffers messages.

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 None.

4 Referenced Standards and Documents

4.1 *SEMI Standards and Safety Guidelines*

SEMI E132 — Specification for Equipment Client Authentication and Authorization

SEMI E179 — Specification for Protocol Buffers Common Components

4.2 *IETF® Documents*[2]

RFC 4122 — A Universally Unique Identifier (UUID) URN Namespace[3]

RFC 7540 — Hypertext Transfer Protocol Version 2 (HTTP/2)[4]

RFC 8018 — PKCS #5: Password-Based Cryptography Specification Version 2.1[5]

RFC 8446 — The Transport Layer Security (TLS) Protocol Version 1.3[6]

4.3 *NIST® Documents*[7]

Federal Information Processing Standards (FIPS) – 180-4 — Secure Hash Standard[8]

[1] Trademark is owned by The Linux Foundation®

[2] 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.

[3] https://tools.ietf.org/html/rfc4122

[4] https://tools.ietf.org/html/rfc7540

[5] https://tools.ietf.org/html/rfc8018

[6] https://tools.ietf.org/html/rfc8446

[7] National Institute of Standards and Technology, 100 Bureau Drive, Stop 3460, Gaithersburg, MD 20899-3460, USA; Telephone: +1.301.975.6478, http://www.nist.gov. NIST trademark is owned by National Institute of Standards and Technology.

[8] https://doi.org/10.6028/NIST.FIPS.180-4

SUPERSEDED
Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

4.4 *CNCF® Documents*[9]

Wire format over HTTP2 – gRPC over HTTP2[10]

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

5 Terminology

5.1 *Abbreviations and Acronyms*

5.1.1 *CA* — Certificate Authority

5.1.2 *EDA* — Equipment Data Acquisition

5.1.3 *gRPC* — gRPC Remote Procedure Call

5.1.4 *HMAC* — hash-based message authentication code

5.1.5 *HTTP/2* — Hypertext Transfer Protocol version 2

5.1.6 *PBKDF2* — Password-Based Key Derivation Function 2

5.1.7 *PKCS™*[11] — Public-Key Cryptography Standards

5.1.8 *PRF* — pseudorandom function family

5.1.9 *SHA* — Secure Hash Algorithm

5.1.10 *SSL* — Secure Sockets Layer

5.1.11 *TLS* — Transport Layer Security

5.1.12 *TCP/IP* — Transmission Control Protocol/Internet Protocol

5.1.13 *UUID* — Universally Unique Identifier

5.2 *Definitions*

5.2.1 *plaintext* — unencrypted information that is input into a cryptographic algorithm. [SEMI E132]

5.2.2 *SHA–256* — member of the SHA-2 family of hash functions that generates a 246-bit hash value.

6 Resources, Organizations, and Standards

6.1 Important resources and organizations used in the generation and definition of this Standard are found in:

- *Protocol Buffers* — https://developers.google.com/protocol-buffers/
- *gRPC* — https://grpc.io/docs/

6.2 For an overview of Protocol Buffers and gRPC, refer to:

- SEMI E179 — Related Information 1, Technology Overview
- SEMI E179 — Related Information 2, Protocol Buffers and gRPC Implementation Considerations

7 Conventions

7.1 *Requirements Identification*

7.1.1 The following notation specifies the structure of requirement identifiers.

7.1.1.1 The following requirements prefix format is used at the beginning of requirement text. Refer to Table 1 for the format notation of the requirements prefix: [Esss.ss-RQ-nnnnn-nn]

[9] Cloud Native Computing Foundation c/o The Linux Foundation, 548 Market St PMB 57274 San Francisco, CA 94104-5401, USA; Telephone: +1.415.723.9709. CNCF trademark is owned by The Linux Foundation.

[10] https://github.com/grpc/grpc/blob/master/doc/PROTOCOL-HTTP2.md

[11] Trademark is owned by RSA Data Security, Inc.

SUPERSEDED

Documents
Document Sharing Hub

Full document details: SEMI E132.2-0422

7.1.1.2 To mark the end of the requirement text, the following suffix format is used: [/RQ]

7.1.1.3 Requirements in the body text are highlighted with a border and light green background (may appear gray in black and white printouts) as shown below.

[Esss.ss-RQ-nnnnn-nn] Requirement text. [/RQ]

Table 1 Requirement Identifiers

Format Notation	*Purpose*
Esss.ss	SEMI Standards Specification identifier. Examples: E87.00, E87.01, E134.00.
RQ	Indicates this is a requirement identifier.
Nnnnn	Unique five-digit number within this Specification. 90000–99999 are reserved for use by SEMI.
Nn	Two-digit number that indicates version level of the requirement. A value of .00 is used for the first version of a requirement.
/RQ	Indicates the end of a requirement.

7.1.1.4 Note that the requirements in Appendix 1 use the reserved requirement numbers noted in Table 1 because they are repeated in other SEMI Standards that use this convention.

7.1.2 Requirements in tables are delimited in one of two ways.

7.1.2.1 Where the requirement occupies an entire row in the table, the requirement identifier (RequirementID) is placed in column 1. No '[/RQ]' is used to mark the end of the requirements text in this case. The table may also contain rows that are not requirements. In this case, the column 1 cell is left blank.

7.1.2.2 Non-requirements text related to a requirement row may be included in an adjacent row. The relationship between the rows is indicated by a broken line separating the two.

7.1.2.3 Where the requirement occupies only one cell, the text in the cell includes the RequirementID prefix and suffix similar to requirements in the body text.

7.1.2.4 No cell in a table will contain both requirement and non-requirement text.

7.1.2.5 Cells that contain requirements are shaded light green.

7.1.2.6 Table 2 provides an example of requirements in a table. Note that in this example, the same requirement is presented in two alternative formats. In practice, mixing the two approaches in the same table is not typically done.

Table 2 Example Table with Requirements

RequirementID	*Statement*	*Additional Detail*
Esss.ss-RQ-nnnnn-nn	The light shall be the sole blue light.	There shall be no similar color in the light panel.
	The blue light is typically placed at the leftmost position in the light panel.	
	[Esss.ss-RQ-nnnnn-nn] The light shall be the sole blue light. There shall be no similar color in the light panel. [/RQ]	The blue light is typically placed at the leftmost position in the light panel.

7.1.3 Only text marked with the RequirementID is a requirement of this Specification.

7.1.3.1 Clarifications, examples, and related recommendations may be provided near a requirement, but are not part of the requirement.

7.1.3.2 Note that the word 'should' is used in some non-requirements text, where it denotes a recommendation or a best practice, not a requirement.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

7.1.3.3 Note that the words 'must' and 'will' are used in the Implementation Overview section to illustrate the logical result of requirements specified in the prerequisites and requirements section. Such statements should not be considered requirements.

7.1.4 Parent-child relationships of requirements are noted in Appendix 1, Table A1-1. Where a parent requirement includes conditions or selection criteria, these are passed to their children. For example, if a parent requirement is stated to apply only to photolithography equipment, any child requirements also apply only to such equipment. The condition need not be restated for each child requirement.

7.1.4.1 Compliance to a parent requirement includes compliance to all of its child requirements.

7.1.4.2 Some cells or rows in a table may contain headings that are assigned a RequirementID. These are parent requirements for other requirements in the table—typically for the rows that immediately follow that heading.

7.2 This Standard uses 'service definition' to refer to gRPC services defined in the .proto file (such as E132ConsumerReceiverSessionInformation) and 'interfaces' to refer to interfaces defined in SEMI E132 (such as SessionClient and SessionManager).

8 General Requirements for Protocol Buffers

8.1 The following is the top-level parent requirement of all others in this Specification.

[E132.02-RQ-00001-00] Compliance to this Specification shall require compliance to all other identified requirements in this Specification, including complementary files. [/RQ]

8.2 *Protocol Buffers File*

8.2.1 This Specification includes a .proto file, 'SEMIE132-02-0422.proto' file, that provides the normative definitions of the types and methods described here.

[E132.02-RQ-00002-00] Implementations shall use the Complementary File 'SEMIE132-02-0422.proto' file that accompanies this Specification. The Complementary File is an official part of and accompanies this Specification. The contents of the .proto file constitutes a core part of this Specification. [/RQ]

8.2.2 The .proto file can be compiled into your implementation-specific programming language, generating the appropriate classes for all specified types.

[E132.02-RQ-00003-00] Implementations shall integrate with Protocol Buffers v3.17.0 at a minimum. Implementations may use newer versions of Protocol Buffers if they are compatible with this stated version. [/RQ]

8.2.3 gRPC provides a SDK framework based on the EDA application's specific development language. Each of the development language-specific gRPC implementation have their own version number. A specific version is not required to comply with this Standard because they all support the gRPC wire format (refer to § 13).

NOTE 1: This Standard specifies a minimum version of Protocol Buffers that needs to be supported to ensure compatibility because Protocol Buffers is used as the gRPC payload.

8.2.4 This Specification also includes Various Materials. These Various Materials are not an official part, but accompany this Specification. Use of these Various Materials is optional and is not required for compliance with this Specification.

8.2.4.1 The file SEMIE132-02-0422-helper.proto is a marked up version of the .proto file for this Specification and is released as Supplemental Material. It has comments prefixed by 'INTERNAL –' that describe how the Protocol Buffers messages map to XML definitions. Most of these items were specified in complementary files for SEMI E132.1. These comments can be helpful to implementers familiar with the XML objects. They are not included in the official .proto file included with the SEMI Specification to keep that document readable.

NOTE 2: Implementers should not use the SEMIE132-02-0422-helper.proto file as its messages are the same as the ones defined in the official .proto file that accompanies this Specification.

8.3 *gRPC Connections*

8.3.1 EDA Applications with Client Sessions connect to the Equipment Server to perform interface discovery, security administration and session management.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E132.2-0422

[E132.02-RQ-00004-00] The EDA Application with Client Sessions shall open the gRPC connection to the Equipment Server and make gRPC method calls on the E132InterfaceDiscovery, E132SecurityAdministration and E132SessionManagement service definitions. [/RQ]

8.3.2 Two connection scenarios for Consumer Sessions are supported.

[E132.02-RQ-00005-00] The Equipment Server shall support both of the following connection scenarios for Consumer Sessions:
1) Consumer Session shares the gRPC connection opened to the Equipment Server by the EDA Application for the Client Session and the Consumer Session makes any gRPC calls
2) Equipment Server opens the gRPC connection to the EDA Application at the specified endpoint, and the Equipment Server makes any gRPC calls. [/RQ]

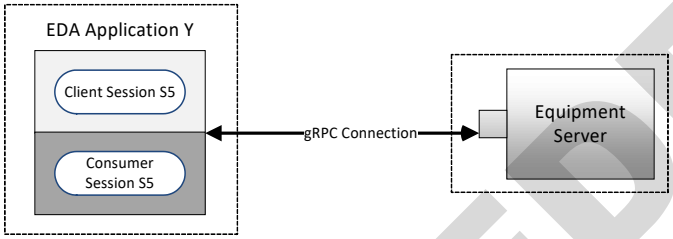

Figure 1
Consumer Session Shares gRPC Connection with Client Session

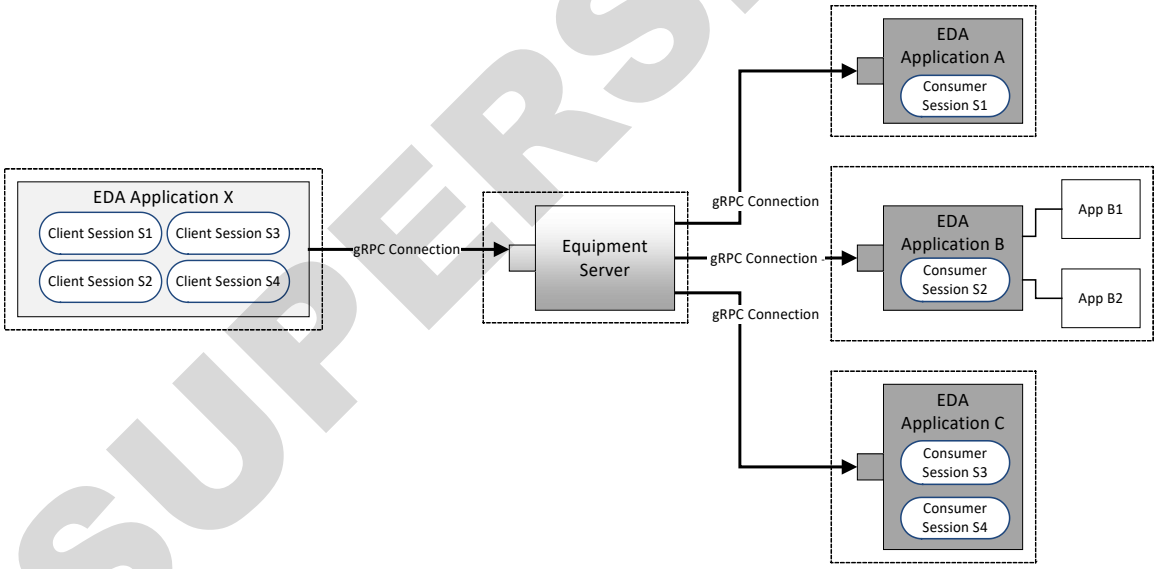

Figure 2
Equipment Server Opens gRPC Connections to EDA Applications

8.3.2.1 An implementation may choose to have the Equipment Server open a gRPC connection to the EDA Application running in the same software package as the Client Session. This is simply a variation of the second connection scenario (Equipment Server opens gRPC connection to the EDA Application).

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

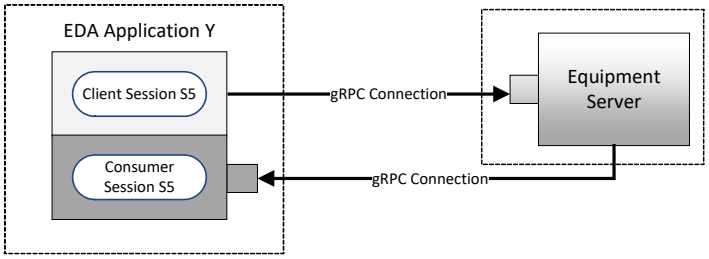

Figure 3
Equipment Server Opens gRPC Connection to EDA Application – Variation 1

8.3.2.2 The Equipment Server can send notification messages.

[E132.02-RQ-00006-00] The Equipment Server shall send notification messages using either the E132ConsumerReceiverSessionInformation service definition if the Equipment Server opened the gRPC connection to the EDA Application, or the E132SessionManagement service definition if the EDA Application opened the gRPC connection to the Equipment Server. [/RQ]

8.3.2.3 For security reasons, some equipment users may prefer to only support the scenario where EDA Applications connect to the Equipment Server, and not allow the Equipment Server to open connections to other machines. (They do not want to expose a gRPC port on EDA Application machines, and they do not want to open up ports on the firewall as it increases risk).

[E132.02-RQ-00007-00] The Equipment Server shall support the ability to enable or disable the Equipment Server's ability to establish communications with Consumer Sessions. [/RQ]

8.3.2.4 The Client Session can specify endpoint information as part of the EstablishSession() call.

- If hasConsumerSession argument is true and endpoint information is specified, the Equipment Server should establish the gRPC connection to the Consumer Session via the specified endpoint.
- If hasConsumerSession argument is true and no endpoint information is specified, the Consumer Session will share the gRPC connection to the Equipment Server opened by the EDA Application for the Client Session.
- If hasConsumerSession argument is false, the session will never be used to receive data.

[E132.02-RQ-00008-00] If the equipment is configured not to allow the Equipment Server to establish communications with Consumer Sessions, any EstablishSession() calls specifying endpoint information shall be rejected with an invalid endpoint error. [/RQ]

8.3.3 Configuration changes related to the ability for the Equipment Server to establish communications with the Consumer Session do not impact active sessions.

[E132.02-RQ-00009-00] Changing the configuration to disable the ability for the Equipment Server to establish communications with Consumer Sessions shall not change any pre-existing sessions in the ESTABLISHED state as defined in the SEMI E132 Session State Model. [/RQ]

NOTE 3: Persisted sessions configured with endpoint information will establish communications with the Consumer Session when it transitions from the Frozen to Established state (per SEMI E132 Session State Model), regardless of the configuration switch to disable the equipment server's ability to establish communications. After disabling the equipment server's ability to establish communications, the equipment user should close these sessions if they do not want the equipment server to establish communications to those Consumer Sessions.

9 SEMI E132 Authentication to Secure Communications Mapping

9.1 *Secure Socket Layers (SSL)/Transport Layer Security (TLS) Overview* — This section describes the use of SSL/TLS security protocol and supporting technology to implement the authentication scheme described in SEMI E132. SSL and TLS are both cryptographic protocols that provide authentication and data encryption between components operating over a network. SSL is the predecessor to TLS and has been deprecated by the IETF. SSL/TLS

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E132.2-0422

is a common phrase until more people become familiar with TLS. SSL/TLS is application protocol independent and transparently supports any higher layered transport protocols such as gRPC with Protocol Buffers.

9.1.1 gRPC has SSL/TLS integration built in. Refer to Related Information 2, § R2-3 SSL/TLS with gRPC for more information.

9.2 *Public Certificates Management*

9.2.1 Public key certificates are required to support secure communications. They are used to establish identities for the equipment and application software as part of the authentication process. Certificates are exchanged as part of the SSL/TLS handshake. This is done as part of the gRPC infrastructure when the software application establishes a connection to a secure port.

9.2.2 If secure communication is required, certificates will need to be installed on the Equipment Server and EDA Application(s). Certificate generation, distribution, maintenance and renewal are the responsibility of the equipment user (i.e., factory). Equipment suppliers are not required to provide certificates or Certificate Authority (CA) implementations. Refer to Related Information 2, § R2-2 Public Key Certificates for more information.

[E132.02-RQ-00010-00] If secure communications are used, the equipment shall reject all SEMI E132 EstablishSession requests if the Client Session does not have a valid ACLEntry defined, even if the EDA Application possesses a valid certificate issued by a trusted CA. [/RQ]

9.3 *Enabling/Disabling Secure Communications*

9.3.1 There may be some integration, development, and/or test environments in which it is not practical or feasible to install equipment and/or client certificates in order to communicate with the equipment. As a practical consideration for such environments only, the equipment should support disabling secure communication for the SEMI E132 functionality interface.

[E132.02-RQ-00011-00] The equipment shall support the ability to enable or disable secure communication. [/RQ]

[E132.02-RQ-00012-00] If an EDA Application with Client Sessions connects to the Equipment Server with secure communications, the Equipment Server shall require mutual client-server authentication for operations supported by the Equipment Server on that connection via SSL/TLS. [/RQ]

9.3.2 Mutual client-server authentication refers to both parties authenticating each other at the same time as part of the initial handshake – the server sends its certificate so the client can verify the server is who they claim to be. The client sends its certificate to the server so it can verify the client is who they claim to be. This is also known as Mutual TLS authentication (mTLS).

9.3.3 Users should be aware that connecting to the equipment over not secure communications opens up complete and unrestricted access to SEMI E132 data for any software application with network access to the equipment. Authenticating the calling component is only supported with secure communications.

[E132.02-RQ-00013-00] Changing the configuration to enable or disable secure communications shall not change the connection type of any pre-existing sessions in the ESTABLISHED state as defined in the SEMI E132 Session State Model. [/RQ]

9.3.4 If the equipment supports secure communications, it may choose to support both a secure and not secured port at the same time (each with a unique port number). Some software applications may want their data transmitted securely, and some may accept the trade-offs to transmit the data over a not secure connection.

[E132.02-RQ-00014-00] If a Client Session calls EstablishSession with an endpoint specifying a secure connection, the equipment shall establish the gRPC connection to that specified endpoint over a secure connection with mutual client-server authentication (with SSL/TLS). [/RQ]

9.3.5 Persisted session configuration includes if the connection to the specified endpoint is secure or not.

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

[E132.02-RQ-00015-00] Persisted sessions will remember if the connection to the specified endpoint is secure. When transitioning from FROZEN to ESTABLISHED as defined in the SEMI E132 Session State Model, if the equipment is to establish the gRPC connection, it shall establish it the same way, regardless of the equipment's current configuration setting to enable or disable secure communications. [/RQ]

NOTE 4: After disabling the Equipment Server's ability to use secure communications, if the equipment user does not want persisted sessions to continue to use secure communications, the session should be closed and established again without secure endpoint configuration.

9.3.6 If the equipment attempts to resume a persistent session from the Frozen state and the connection type cannot be supported, the Equipment Server treats it as if the Consumer Session is not reachable (as defined in the SEMI E132 Session State Model). Examples include:

- The persisted session is using an endpoint that does not accept the gRPC connection.
- The persisted session is using a secure connection and the EDA Application no longer supports it.

10 Mapping of SEMI E132 to Protocol Buffers

10.1 *Interactions with SEMI E179*

10.1.1 This Standard supports SEMI E179 for common Protocol Buffers elements that are not specific to a single standard and are available to multiple standards.

[E132.02-RQ-00016-00] SEMI E132.2 implementations shall use SEMI E179 for common Protocol Buffer components. [/RQ]

10.2 *SecurityAdmin Interface*

10.2.1 *Establishing an ACLEntry for the Security Admin* — An ACLEntry defining the Security Admin is available at time of equipment installation.

[E132.02-RQ-00017-00] The equipment shall define exactly one ACL entry for the Security Admin and it shall grant the reserved privilege 'urn:semi-org.auth:securityAdminPrivileges' as defined by SEMI E132. [/RQ]

10.3 *Session Identifier* and Client Identifier

10.3.1 The session identifier (SessionId) is defined in SEMI E132 and is generated by the equipment during the EstablishSession operation. It is an UUID that is guaranteed unique across space and time (refer to RFC 4122 for specific details on syntax and format of a UUID).

10.3.2 In Protocol Buffers messages, the SessionId is represented as a string data type, as it is more easily understood in its human-readable form.

10.3.3 The session identifier is specified in all requests to the Equipment Server over a SEMI E132 authenticated session. gRPC's metadata infrastructure is used to pass the SessionId for gRPC methods. This provides the most transparent way of sending the session identifier without affecting the input/output parameters of the individual unary methods. gRPC metadata take name-value pairs and stores them in the HTTP/2 header data.

NOTE 5: For more information about gRPC's metadata infrastructure, refer to SEMI E179 — Related Information 2, Protocol Buffers and gRPC Implementation Considerations.

[E132.02-RQ-00018-00] SEMI E132 SecurityAdmin, SessionManager and SessionClient operations that are implemented as gRPC methods (except for SessionManager.GetEquipmentInformation and SessionManager.EstablishSession) shall specify the SessionId with the gRPC metadata infrastructure. [/RQ]

10.3.4 The SessionId for methods in the E132SecurityAdministration can be sent in plaintext because the expectation is that this will be done over a secure connection (per § 9.3.3).

[E132.02-RQ-00019-00] When sending SessionId information with gRPC metadata for methods on the E132SecurityAdministration and E132SessionManagement service definitions,
The key for the SessionId name-value pair shall be 'sessionid'
The value for the SessionId name-value pair shall be the SessionId and be a string data type. [/RQ]

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E132.2-0422

10.3.5 To help protect the SessionId in outgoing messages from the Equipment Server (which may be sent over a not secured connection), it is hashed with the SHA-256 hash algorithm (as specified in the Secure Hash Standard). The value of SessionId when it is hashed is based on Base64 Binary type. The SessionId is first converted to UTF-8 prior to applying the hash, and the hashed value is stored in a string data type.

[E132.02-RQ-00020-00] When sending SessionId information with gRPC metadata on the E132ConsumerReceiverSessionInformation service definition,
The key for the SessionId name-value pair shall be 'sessionidhash'
The value for the SessionId name-value pair shall be the SessionId hashed with SHA-256 and be a string data type. [/RQ]

10.3.6 The client identifier (ClientId) is defined in SEMI E132 and is specified by the Client Session during the EstablishSession operation to identify the ACLEntry to authenticate.

10.3.7 The Consumer Session may be in a separate software application from the corresponding Client Session. It will require the ClientId so that it can respond to SessionPing messages from the Equipment Server properly. To help protect the ClientId in outgoing messages from the Equipment Server (which may be sent over a not secured connection), it is hashed with the SHA-256 hash algorithm (as specified in the Secure Hash Standard). The value of ClientId when it is hashed is based on Base64 Binary type. The ClientId is first converted to UTF-8 prior to applying the hash, and the hashed value is stored in a string data type.

[E132.02-RQ-00021-00] When sending ClientId information with gRPC metadata on the E132ConsumerReceiverSessionInformation service definition,
The key for the ClientId name-value pair shall be 'clientidhash'
The value for the ClientId name-value pair shall be the ClientId hashed with SHA-256 and be a string data type. [/RQ]

NOTE 6: Implementations where the Consumer Session shares the gRPC connection with the Client Session can communicate the ClientId between the two entities. The mechanism of transferring this information is an implementation detail.

10.3.8 Similarly, ID values (SessionId and ClientId) associated with messages sent from the Equipment Server through the SEMI E132 SessionClient interface are hashed with SHA-256 hash algorithm and stored in a string data type. For consistency, the ID value is hashed regardless if the Protocol Buffers message they are in are sent over secure communications. For example, if the SessionClosed notification message is sent to the Client Session over secure communications.

10.3.8.1 Table 3 outlines the SEMI E132 Protocol Buffers messages and fields that contain hashed ID values.

Table 3 Hashed ID Value Mapping to Protocol Buffer Message Fields

ID	*SEMI E132 Protocol Buffers Message and Field To Store Hashed Value*	*Comments*
ClientId	SessionPingAnswerType.IdHash	Used in response to SessionPing sent by the Equipment Server.
SessionId	SessionClosedNotificationType.SessionIdHash	Used in SessionClosed notification message.
SessionId	AclPasswordChangedNotificationType.SessionIdHash	Used in AclPasswordChanged notification message.

[E132.02-RQ-00022-00] ID values SessionId and ClientId ID specified in Protocol Buffer messages corresponding to the SEMI E132 SessionClient operations shall be hashed with SHA-256 and stored in the Protocol Buffers fields specified in Table 3. [/RQ]

10.4 *SEMI E132 Operation Error*s

10.4.1 *Common Error* — Table 4 below provides a mapping of the SEMI E132 defined errors to its common error code. The SEMI E132 mapped error codes below are in addition to any common error codes defined in SEMI E179, and any errors defined by the equipment supplier. These error codes are reported with the SEMI E179's ErrorType structure. Errors during the authentication message exchange are as defined by the secure communication protocol.

SUPERSEDED
Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

[E132.02-RQ-00023-00] SEMI E132 error codes shall be denoted in the ErrorType structure by defining the 'source' name of 'urn:semi-org:E132'. [/RQ]

Table 4 SEMI E132 Error Codes

SEMI E132 Error	*Common Error Code*
Operation not authorized	6000
Duplicate ACL entry found	6001
Unrecognized role	6002
Unrecognized privilege	6003
ACL entry not found	6004
Unrecognized session	6005
Max session limit exceeded	6006
Unrecognized equipment capability	6007
Duplicate equipment capability request	6008
Invalid version request	6009
Deleting role with principals assigned	6010
Unrecognized client	6011
The specified endpoint is invalid	6012
Invalid new SessionPing setting(s)	6013
Specified ACL entry already has active session	6014
Consumer Session not ready	6015
SessionId already in use	6016
Specified notificationRecipients configuration is invalid	6017
Could not connect to specified endpoint	6018
ACL password is not valid	6019
Invalid new maxSessions setting	6020

10.4.2 *Custom Errors* — Custom errors structures defined in the 'SEMIE132-02-0422.proto' file (such as UnauthorizedOperationError) do not need the source information explicitly specified because they are specific to SEMI E132.

10.4.3 *Unauthorized Operation Error* — Many operations may return an operation not authorized error (common error code 6000). For this error, SEMI E132 defines a specific UnauthorizedOperationError structure to provide additional context information such as a list of all the privileges that grant authorization for the requested operation.

10.5 *Using SEMI E179 Common Errors* — SEMI E179 defines common error codes. Implementations may return these error codes if they are applicable and the primary standard does not explicitly define the error condition. These error codes are reported with the SEMI E179's ErrorType structure and the error source is listed as SEMI E132.

10.5.1 Table 5 below provides a mapping of the SEMI E132 defined errors to specific SEMI E179 common error codes.

Table 5 SEMI E132 Error Codes Mapped to SEMI E179 Common Error Codes

SEMI E132 Error	*SEMI E179 Error Code*
System not ready	5004
System shutting down	5005
Specified configuration is invalid	5003

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E132.2-0422

[E132.02-RQ-00024-00] If SEMI E179 error codes are reported by the SEMI E132.2 implementation, 'source' name of the error shall be 'urn:semi-org:E132'. [/RQ]

10.6 *SEMI E132 Exceptions*

10.6.1 When working with a streaming gRPC method (e.g., E132SessionManagement.SessionUsageStream()), there is no message response that can indicate that there is an error with the method invocation. (e.g., SessionId specified in the gRPC metadata is unrecognized or missing when it is supposed to be provided).

10.6.1.1 In this situation, the gRPC method throws an RPC exception with a specific gRPC StatusCode and uses trailer gRPC metadata infrastructure to provide the error information. In the trailer gRPC metadata, the MetaData.Key is lower-case and the MetaData.Value is a string data type.

NOTE 7: For more information about how to throw gRPC exceptions, refer to SEMI E179 — Related Information 2, Protocol Buffers and gRPC Implementation Considerations.

10.6.2 This Standard defines specific gRPC exceptions to throw for SEMI E132 error conditions. The trailer gRPC metadata reports the SessionId information in the same format it was received from the header gRPC metadata. Only one of MetaData.Key 'sessionid' or 'sessionidhash' is specified in the trailer gRPC metadata.

- If the RPC exception is thrown from the E132SecurityAdministration or E132SessionManagement service definitions, the MetaData.Key 'sessionid' is used.
- If the RPC exception is thrown from on the E132ConsumerReceiverSessionInformation service definition, the MetaData.Key 'sessionidhash' is used.

10.6.2.1 If the SessionId information is missing from the header gRPC metadata when it is supposed to be present, the value for the corresponding SessionId entry in the trailer gRPC metadata is an empty string.

10.6.3 *Unrecognized Session Exception*

10.6.3.1 If the header gRPC metadata in a streaming gRPC method does not specify a valid SessionId, the implementing code cannot process it and throws an exception.

Table 6 gRPC Exception Details for Unrecognized Session Error Condition

gRPC Status Code	*StatusCode.Invalid_Argument*	
Trailer gRPC Metadata		
	MetaData.Key	*MetaData.Value*
	sessionid	Unrecognized SessionId specified (if specified and SessionId was sent as in plaintext)
	sessionidhash	Unrecognized SessionId specified (if specified and SessionId was sent as a hashed value)
	source	UnrecognizedSessionError's Source value
	code	UnrecognizedSessionError's Code value
	description	UnrecognizedSessionError's Description value
	extension	UnrecognizedSessionError's Extension value

[E132.02-RQ-00025-00] If a streaming gRPC method call is missing the SessionId or specifies an unrecognized SessionId in the gRPC metadata, the implementing code throws an RPC Exception with the gRPC Status Code and trailer gRPC Metadata as specified in Table 6. [/RQ]

10.6.4 *SessionId Already In Use Exception*

10.6.4.1 There can only be one instance of a streaming gRPC method call per SessionId. This helps ensure there is only one Client Session using the SessionId and at most one Consumer Session, and that there are not two different software applications trying to get the same streaming messages for the same SessionId. If the header gRPC metadata

SUPERSEDED

Documents Document Sharing Hub
Full document details: SEMI E132.2-0422

in a streaming gRPC method specifies a SessionId that already associated with an instance of the same streaming gRPC method, the implementing code cannot process it and throws an exception.

Table 7 gRPC Exception Details for SessionId Already In Use Error Condition

gRPC Status Code	*StatusCode.Invalid_Argument*	
Trailer gRPC Metadata		
	MetaData.Key	*MetaData.Value*
	sessionid	SessionId already in use (if SessionId was sent as in plaintext)
	sessionidhash	SessionId already in use (if SessionId was sent as a hashed value)
	source	SessionIdAlreadyInUseError's Source value
	code	SessionIdAlreadyInUseError's Code value
	description	SessionIdAlreadyInUseError's Description value
	extension	SessionIdAlreadyInUseError's Extension value

[E132.02-RQ-00026-00] The implementing code shall only allow one streaming gRPC method instance per SessionId. If the specified SessionId of a streaming gRPC method call is already associated with another instance of the same streaming gRPC method, the implementing code throws an RPC Exception with the gRPC Status Code and trailer gRPC Metadata as specified in Table 7. [/RQ]

10.6.5 *Unrecognized Client Exception*

10.6.5.1 The Equipment Server sends hashed ClientId information with gRPC metadata on streaming gRPC methods on the E132ConsumerReceiverSessionInformation service definition (refer to § 10.3.7). If the header gRPC metadata in a streaming gRPC method does not specify valid hashed ClientId, the implementing code cannot process it and throws an exception.

10.6.5.1.1 The trailer gRPC metadata reports the clientidhash information in the same format it was received from the header gRPC metadata.

10.6.5.1.2 If the clientidhash information is missing from the header gRPC metadata, the value for the corresponding clientidhash entry in the trailer gRPC metadata is an empty string.

10.6.5.1.3 Implementations may perform additional validation checks on the received hashed ClientId (e.g., check if Consumer Session should accept the data for the specified hashed ClientId). This Standard does not require any additional validation checks on the received hashed ClientId value.

SUPERSEDED

Documents
Document Sharing Hub
Full document details: SEMI E132.2-0422