. MQTT (“Message Queuing Telemetry Transport”) As mentioned, MQTT is an open pub/sub protocol [2] designed for constrained devices used in telemetry applications. However, it does not consider the case of SA devices; its extension for sensor networks, “MQTT-S”, will be described in Section IV. MQTT is designed in such a way that its implementation on the client’s side (i.e., the SA’s side) is very simple. All of the system complexities reside on the broker’s side. MQTT does not specify any routing or networking techniques; it assumes that the underlying network provides a point-to-point, sessionoriented, auto-segmenting data transport service with in-order delivery (such as TCP/IP) and employs this service for the exchange of messages.

MQTT is a topic-based pub/sub protocol that uses character strings to provide support of hierarchical topics. This also facilitates the subscription to multiple topics. For example, a temperature sensor located on floor “F2”, room “R248” could publish its data using the hierarchical topic “wsn/sensor/F2/R248/temperature”. The forward slash character “/” is used to separate each part of the topic. Wildcard characters can then be used to replace any part of the topic, e.g., the string “wsn/sensor/F2/+/temperature” could be employed to subscribe to data generated by all temperature sensors on floor F2. In this example the character “+” was used as wildcard for any pattern at the 4th level of the topic. MQTT supports basic end-to-end Quality of Service (QoS) [21]. Depending on how reliably messages should be delivered to their receivers, MQTT distinguishes between three QoS levels. QoS level 0 is the simplest one: it offers a besteffort delivery service, in which messages are delivered either once or not at all to their destination. No retransmission or acknowledgment is defined. QoS level 1 provides a more reliable transport: messages with QoS level 1 are retransmitted until they are acknowledged by the receivers. Consequently, QoS level 1 messages are certain to arrive, but they may arrive multiple times at the destination because of the retransmissions. The highest QoS level, QoS level 2, ensures not only the reception of the messages, but also that they are delivered only once to the receiving entities. It is up to the application to select the appropriate QoS level for its publications and subscriptions. For example, a temperaturemonitoring application could decide to use QoS level 0 for the publication of normal and regular measurement reports, but QoS level 1 for transferring alarm messages when the temperature exceeds a certain threshold. MQTT is a connection-oriented protocol in the sense that it requires a client to setup a connection with the broker before it can exchange publications and subscriptions with the broker. To this end, a “CONNECT” message is defined. It contains, among other connection parameters, a Client Id, which enables the broker to identify the connected client. This Client Id is used by the broker, for example to make sure that QoS level 1 and 2 publications are delivered correctly when the client reconnects after a network failure. The broker supervises the liveliness of the client/connection by a “keep-alive” timer, which defines the maximum time interval that may elapse between two messages received from that client. If during this time interval the client has no data-related messages to be transmitted, it will send a PING message to the broker, which is acknowledged by the broker. Thus the keep-alive timer enables the broker to detect the failure of either the client or the network link. A related and interesting MQTT feature is its support of the so-called “Will” concept. At connection time, a client could ask the broker to store a “Will” message together with a “Will” topic. The broker will only send this “Will” publication to the subscribers when it abnormally loses the connection with the client. Applications could use this feature to detect failures of devices and links.