UNIT - V: Service layer protocols and Security:

Service Layer -oneM2M, ETSI M2M, OMA, BBF – Security in IoT Protocols – MAC 802.15.4 6LoWPAN, RPL, Application Layer.

Service Layer one M2M :

Functional Building Blocks This Common M2M Service Layer should be agnostic to underlying network technology (yet leveraging the unique features of these underlying networks), and it will use network layer services (such as security (encryption and authentication), QoS, policy, provisioning, etc.) through an adaptation layer/APIs. These Common M2M Service Layer functions include:

- **Provisioning:** On board new devices by creating new subscription in the common M2M service layer and network layer; activate/deactivate/suspend/resume network and service subscriptions.
- **Device Management:** Manage all aspects of the devices including configuration, firmware upgrades, application lifecycle management, device lock and wipe. Security: Generate relevant key material for secure communications; Authenticate devices before they can register and modify resources. Prevent unauthorized entities from sending IP packets to the devices. Provide a secure connection to the device

Application and Device Registrations: Application and devices will be able to register with the service layer entity for various services. Registration will involve authentication or verification of credentials and creation or allocation of resources within the server and the database. A profile with the capabilities of the device and the type of services allowed for the applications are created.

- **Resource Management:** Applications and devices will be able to create, update, and delete resource objects containing various attributes in the service layer. Entities will be able to discover resources.
- **Content push/pull Services:** Provide API for applications to perform unicast and multicast data push to specified devices within the specified time window. Push may be result of a notification that is triggered as a result of modification of a resource. Provide API for applications to pull data from one or more devices within the specified time window or specified periodicity or other policies that have been established.
- **Store and Forward Messaging:** Applications may request messages to be sent to one or more devices that may not be registered with the network at that time. In this case the communications management entity shall store and aggregate the messages and forward them to the devices at a later time when the devices wake up.

Protocol Translation: Translate protocols between application and device as needed. For example, applications may use HTTP while devices may use Constrained Application Protocol (CoAP) or Zigbee® Smart Energy 2.0 protocol.

- **Subscribe/Notification:** Application and devices should be able to subscribe to receive notifications upon certain events or when certain resources are updated. Events may be specified as rules on certain resource data.

- **Policy Framework:** Framework to establish and incorporate in the session orchestration, data aggregation and storage, the network provider and application provider policies. Examples include incorporating a location tag or time stamp on all data, policy restricting sessions only to certain hours of the day.

- **Location and Geo Fencing:** Provide device and network based location and location related services such as creating a geo-fence or identifying a group of devices within a region or adding a location tag to the device data.

Groups Management: Framework for creation of groups by specifying the members of the group through one of the identifiers of the device, adding additional members or removing members; setting group attributes.

- **Device Triggering:** Provide the capability to trigger the device to register with the network and an application through a secondary means such as an SMS. Provide information about the status of the device in the network.
- **Access Control:** Control the access to the data collected from the devices based on access restrictions specified by applications in terms which users or devices can access what resources.
- **Data Processing and Storage:** Provide temporary and permanent storage for data collected from devices. Process queries on data collected. Provide threshold and expression rules setting and execution on the various data collected from the devices. Notifications could be triggered based on the outcome of the rules testing.
- **Consumption Statistics/records:** Process queries regarding the usage of network resources by a device or a group of devices for billing reconciliation.
- **API Management:** Manage API usage, such as authentication and authorization of calls to APIs provided. The functions above have been identified as essential by the following verticals: Connected Vehicle; Smart Grid; eHealth; and Connected Home. The goal of this diligence was to determine which Common M2M service layer functions were necessary for their respective vertical segment and to then converge on those common functions as the components for a "Common M2M Service Layer" capability

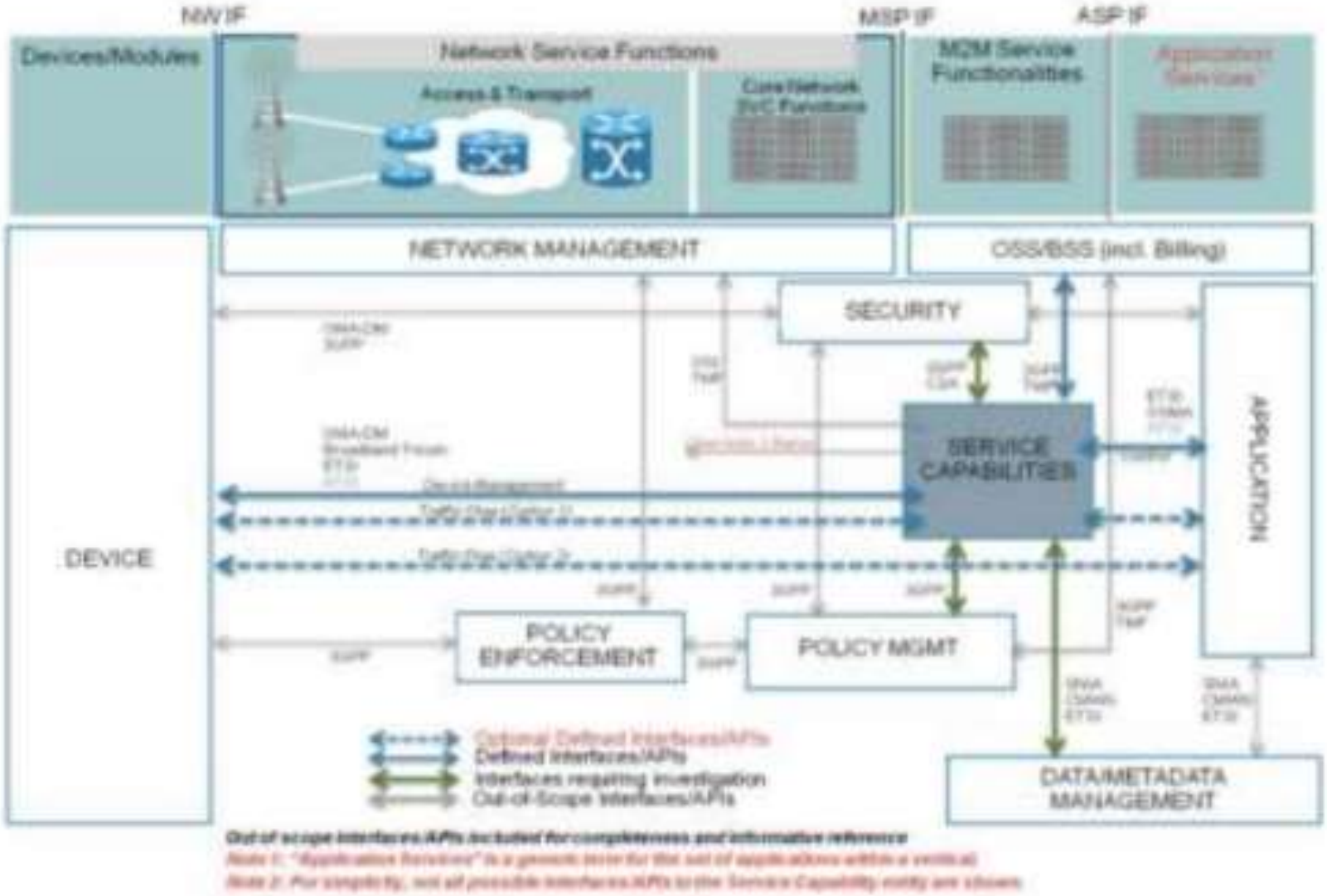
The functions proposed for a Common M2M Service Layer are:

Device Management

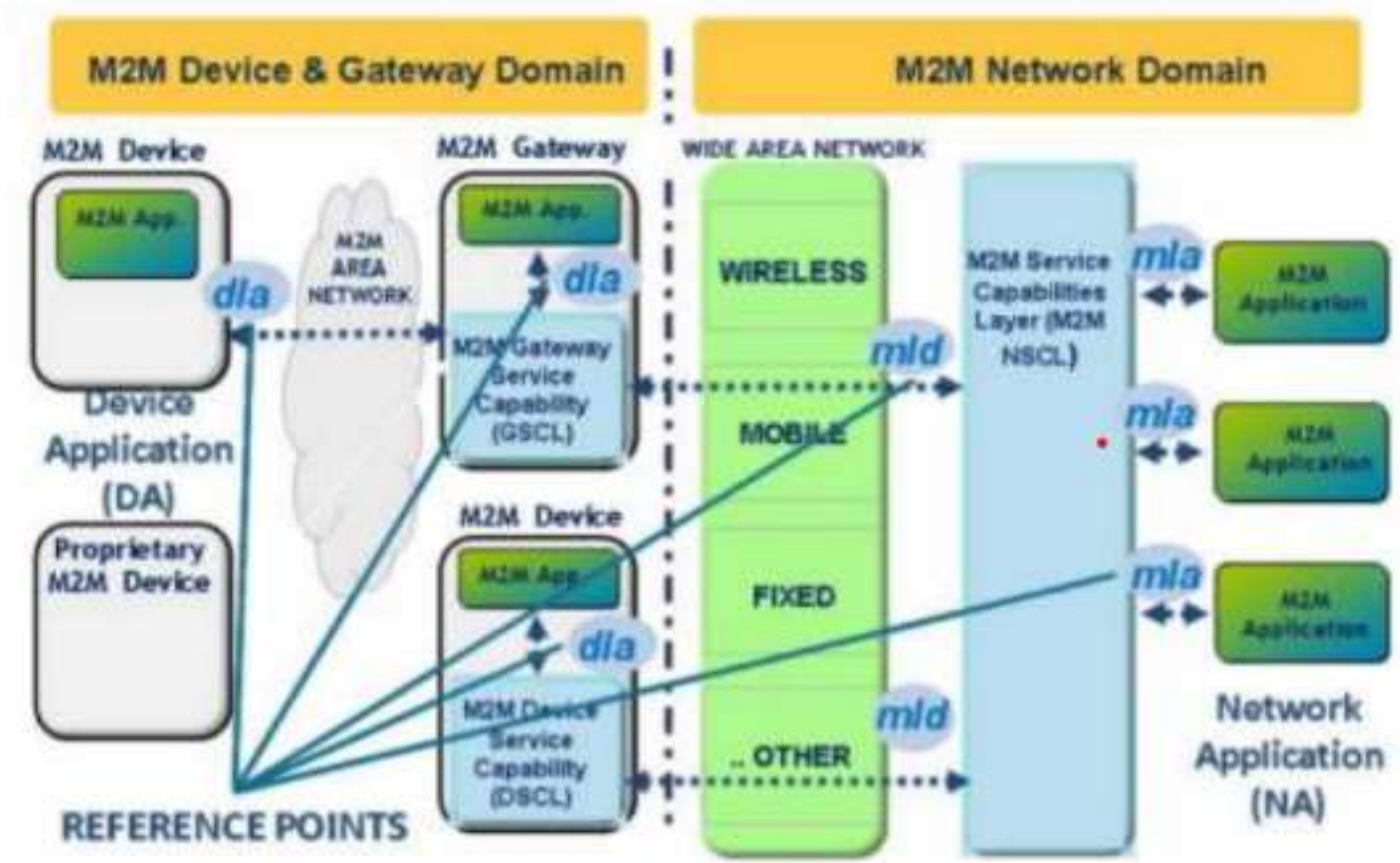
- Provision/Activate (Individual and bulk) and Bootstrap
- Suspend/Resume
- Configuration Management
- Firmware/Software Management
- Inventory Management
- Diagnostics (resource information, status) Policy & Resource Management
- Authentication and Registration (Identity Management)
- Establish communications session (Add/Delete/Modify)
- QoS/SLA for communication session
- Billing, Charging, and Rating rules
- Group Management
- Security Management (Data confidentiality, integrity, abuse prevention, privacy)

API Services

- Definition, Authentication/Authorization and Security
- Service to Device (Management, Establish/Teardown Communication Flows)
- Service to Policy/Resource Management (Rx Extensions for Group Management)
- Service to Data/Metadata Management (Storage/Retrieval)
- Service to Applications (Management, Communications Flows) Data/Metadata Management
- Data processing and append (location, timestamp)
- Data storage/retrieval

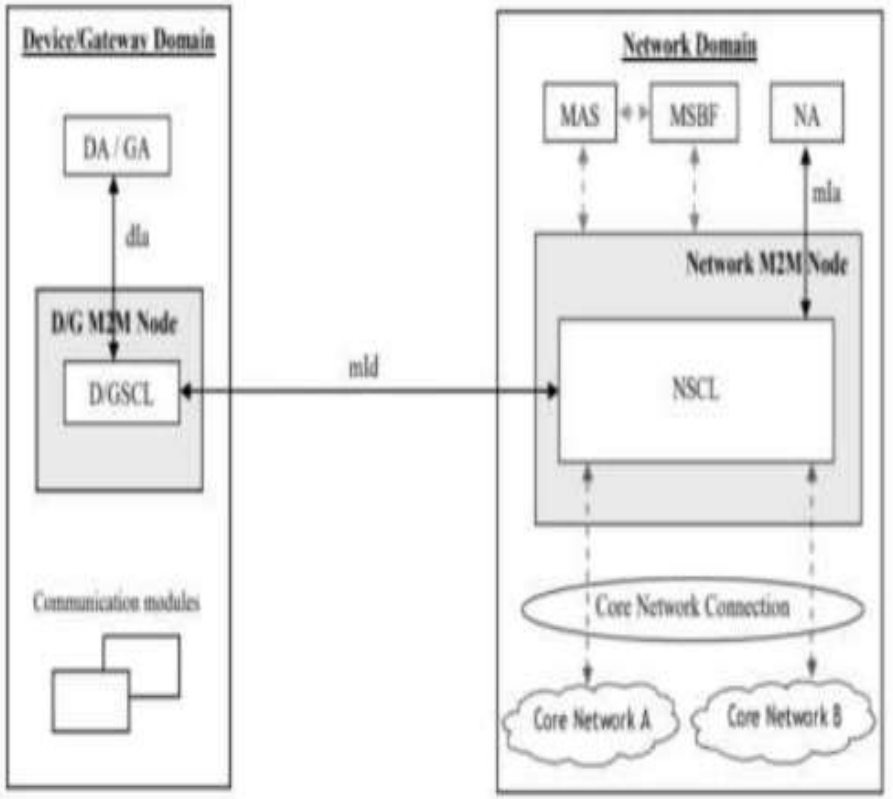

NW IF
MSP IF
ASP IF
Devices/Modules
Network Service Functions
Access & Transport
Core Network
M2M Service Functionalities
NETWORK MANAGEMENT
OSS/BSS (incl. Billing)
SECURITY
SERVICE CAPABILITIES
APPLICATION
DEVICE
POLICY ENFORCEMENT
POLICY MGMT
DATA/METADATA MANAGEMENT
Defined interfaces/APIs
Interfaces requiring investigation
Out-of-Scope Interfaces/APIs

ETSI M2M:


M2M Device & Gateway Domain
M2M Network Domain
M2M Device
M2M Gateway
WIDE AREA NETWORK
M2M App.
dla
M2M AREA NETWORK
M2M Gateway Service Capability (GSCL)
Device Application (DA)
Proprietary M2M Device
M2M Device
M2M App.
M2M Device Service Capability (DSCL)
REFERENCE POINTS
WIRELESS
MOBILE
FIXED
.. OTHER
mld
M2M Service Capabilities Layer (M2M NSCL)
mla
M2M Application
Network Application (NA)

M2M Service Capabilities Layer provides functions that are to be exposed on the reference points. M2M SCs can use
Core Network functionalities through a set of exposed interfaces (e.g. existing interfaces specified by 3GPP, 3GPP2,
ETSI TISPAN, etc.). Additionally, M2M SCs can interface to one or several Core Networks.
the following terms will be used are :

• NSCL: Network Service Capabilities Layer refers to M2M Service Capabilities in the Network Domain.
• GSCL: Gateway Service Capabilities Layer refers to M2M Service Capabilities in the M2M Gateway.
• DSCL: Device Service Capabilities Layer refers to M2M Service Capabilities in the M2M Device.
• SCL: Service Capabilities Layer refers to any of the following: NSCL, GSCL, or DSCL.
• D/G SCL: refers to any of the following: DSCL, GSCL.



The external reference points (mIa, mId, dIa) are mandated and are required for ETSI M2M compliance. M2M Node: is a logical representation of the M2M components in the M2M Device, M2M Gateway, or the M2M Core. An M2M Node shall include one SCL, and optionally an M2M Service Bootstrap function and an M2M Service Connection function. An M2M Node relies on a Secured Environment Domain, controlled by the M2M Service Provider associated with the SCL, to protect Sensitive Functions and Sensitive Data. A Device/Gateway M2M Node shall be instantiated upon pre-provisioning or executing an M2M Service Bootstrap procedure on the M2M Device/Gateway with an M2M Service Provider. Each Device/Gateway M2M Node may be instantiated with only one M2M Service Provider. M2M Applications: are respectively Device Application (DA), Gateway Application (GA) and Network Application (NA). DA could reside in an M2M Device which implements M2M Service Capabilities or alternatively reside in an M2M Device which does not implement M2M Service.

OMA

OMA Lightweight M2M (LwM2M) is a protocol from the Open Mobile Alliance for machine to machine (M2M) or Internet of things (IoT) device management and service enablement. It offers an approach for managing IoT devices and allows devices and systems from different vendors to co-exist in an IoT- ecosystem. Management: Remote Entity Management (REM) service capability shall be supported by all M2M SCLs. Both OMA-DM and BBF TR-069 are considered as the supporting enablers to be reused in ETSI M2M functional architecture to implement the REM service capability. An M2M system may choose to implement either or both of them. Other device management enablers may also be reused in a similar manner, but the details are out of scope. The mIa reference point shall support the RESTful interface procedures to allow the NREM to handle the request from M2M Network Applications for the purpose of the remote entity management The mId reference point shall support the RESTful interface procedures to allow the D/GREM to create and/or resources in the NREM for the purpose of the remote entity management thereafter. It shall also support corresponding OMA-DM [i.2]/TR-069 [i.3] protocol interfaces and procedures for managing M2M Devices/Gateways enabled by OMA-DM [i.2]/TR-069 [i.3].

A <mgmtObj> or <mgmtCmd> resource in the NSCL represents either: • high-level management functionalities (e.g. ETSI M2M specific data model) which shall be supported by the underlying Management Object(s) on the remote entity; or • low-level functionalities on the remote entity mapped from the data model as specified by existing device management technologies (e.g. OMA-DM, BBF TR-069 [i.3]). Through the manipulation of <mgmtObj> or <mgmtCmd> resources, ETSI M2M REM supports the following management functions at different layers of remote entities: • M2M application lifecycle management: installing, removing and upgrading applications in an M2M Device/M2M Gateway. • M2M service management: configuration management for the M2M Service Capabilities in the M2M Device/M2M Gateway. • M2M Area Network management: configuration management for the M2M Area Networks (namely Capillary Network). • M2M device management: configuration management of the M2M Device/M2M Gateway. Management Protocol OMA DM: The OMA DM is a management protocol of mobile devices which makes possible to remotely reach and control devices resources. This protocol is conceived for small devices such as Mobile phones, Smartphones, Tablets, robust laptops, mobile printers, PDAs (Personal DIGITAL Assistant), etc.

OMA DM is for sensitive devices, in other words, devices which have a limited storage memory or limited communication bandwidth, like a wireless connectivity 4. It is difficult for the server to manage mobiles because of their specific configuration and their particular characteristics which depend on each manufacturer. The OMA DM protocol proposed a standard framework making it possible forthe suppliers to define the description of the devices and to communicate it to the server. The server is based on this description to send operations. This description is called the management tree of the device. This tree organizes allthe managed objects of the device in which each node is addressed (in a single way) by an URI (Universal Resource Identifier). The server is so able, if he has the access right, to modify the management tree by sending operations Get, Add, Delete and Update. The OMA DM protocol consists in exchanging a sequence of XML messages between theserver and the client, more particularly the subset defined by SyncML (Synchronization Markup Language). SyncML contains a set of well-defined messages which are transmitted between a server and a client taking part in an operation of synchronization. The communication is carried out in two phases; the phase of initiation then the phase of management of the device.

Management protocol BBF TR-069BBF: TR-069 is a protocol for remote management of end-user devices. He offers to the user a set of administration services, control and diagnosis while avoiding the problems involved in material and software diversities. The standard TR-069 was developed for the automatic configuration of the devices with Internet access with the automatic configuration servers ACS (modem, router, Gateway, Set-Top Boxings, VoIP-telephone, etc.). Since management is done by recovery of the values of devices parameters, the latter are organized according to a hierarchical well-defined structure which is more or less common to all models of devices and manufacturers. This model is published into two formats:• XML format where each model contains a detailed description of devices as well as the modifications carried out.• Pdf version container of information readable by human. This last format is seldom used, since it requires a human intervention which is not always practical.

	OMA DM	BBF TR-069
Data model	Tree	XML file
Device type	Mobile Devices	Network Devices
Trigger	The server informs Device by SMS or WAP Push	The manager connects to the device embeded http server
Architecture Style	Uses the style of architecture REST	Non RESTful management command, RPC

Security in IoT Protocols Security is another aspect of IoT applications which is critical and can be found in all almost all layers of the IoT protocols. Threats exist at all layers including datalink, network, session, and application layers.

MAC 802.15.4 MAC 802.15.4 offers different security modes by utilizing the "Security Enabled Bit" in the Frame Control field in the header. Security requirements include confidentiality, authentic apapertion, integrity, access control mechanisms and secured Time-Synchronized Communications.

6LoWPAN lo6LoWPAN by itself does not offer any mechanisms for security. However, relevant documents include discussion of security threats, requirement and approach to consider in IoT network layer. For example, RFC 4944 discusses the possibility of duplicate EUI-64 interface addresses which are supposed to be unique [RFC4944]. RFC 6282 discusses the security issues that are raised due to the problems introduced in RFC 4944 [RFC6282]. RFC 6568 addresses possible mechanisms to adopt security within constrained wireless sensor devices [RFC6568]. In addition, a few recent drafts in [6Lo] discuss mechanisms to achieve security in 6loWPAN.

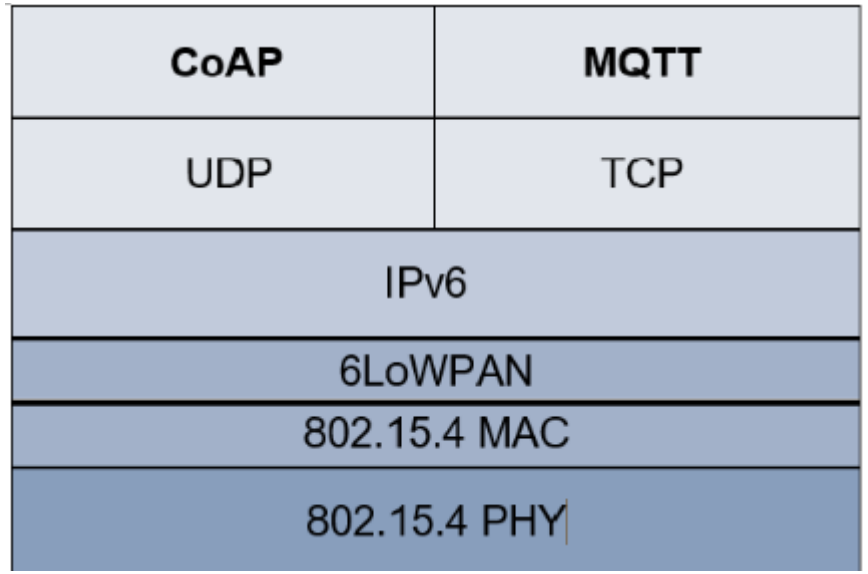
RPL RPL offers different level of security by utilizing a “Security” field after the 4-byte ICMPv6 message header. Information in this field indicates the level of security and the cryptography algorithm used to encrypt the message. RPL offers support for data authenticity, semantic security, protection against replay attacks, confidentiality and key management. Levels of security in RPL include Unsecured, Preinstalled, and Authenticated. RPL attacks include Selective Forwarding, Sinkhole, Sybil, Hello Flooding, Wormhole, Black hole and Denial of Service attacks.

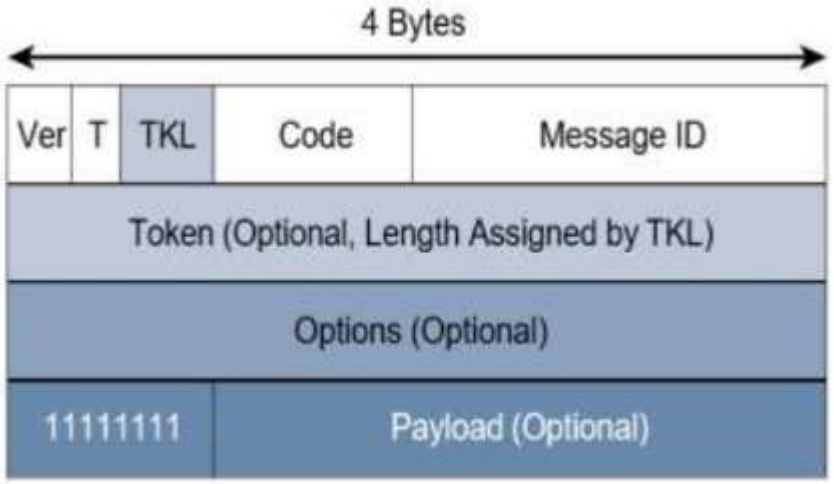
Application Layer Applications can provide additional level of security using TLS or SSL as a transport layer protocol. In addition, end to end authentication and encryption algorithms can be used to handle different levels of security as required.

IoT Application Layer Protocols When considering constrained networks and/or a large-scale deployment of constrained nodes, verbose based and data model protocols, may be too heavy for IoT applications. To address this problem, the IoT industry is working on new lightweight protocols that are better suited to large numbers of con- strained nodes and networks. Two of the most popular protocols are CoAP and MQTT. Figure highlights their position in a common IoT protocol stack.

Constrained Application Protocol (CoAP) resulted from the IETF Constrained RESTful Environments (CoRE) working group's efforts to develop a generic framework for resource-oriented applications targeting constrained nodes and networks. (For more information on the IETF CoRE working group The CoAP framework defines simple and flexible ways to manipulate sensors and actuators for data or device management. The IETF CoRE working group has published multiple standards-track specifications for CoAP, including the following: ■ RFC 6690: Constrained RESTful Environments (CoRE) Link Format ■ RFC 7252: The Constrained Application Protocol (CoAP) ■ RFC 7641: Observing Resources in the Constrained Application Protocol (CoAP) ■ RFC 7959: Block-Wise Transfers in the Constrained Application Protocol (CoAP) ■ RFC 8075: Guidelines for Mapping Implementations: HTTP to the Constrained Application Protocol (CoAP) The CoAP messaging model is primarily designed to facilitate the exchange of messages over UDP between endpoints, including the secure transport protocol Datagram Transport Layer Security (DTLS). (UDP is discussed earlier in this chapter.) The IETF CoRE working group is studying alternate transport mechanisms, including TCP, secure TLS, and WebSocket. CoAP over Short Message Service (SMS) as defined in Open Mobile Alliance for Lightweight Machine-to-Machine (LWM2M) for IoT device manage- ment is also being considered.

RFC 7252 provides more details on securing CoAP with DTLS. It specifies how a CoAP endpoint is provisioned with keys and a filtering list. Four security modes are defined: NoSec, PreSharedKey, RawPublicKey, and Certificate. The NoSec and RawPublicKey implementations are mandatory. From a formatting perspective, a CoAP message is composed of a short fixed-length Header field (4 bytes), a variable-length but mandatory Token field (0–8 bytes), Options fields if necessary, and the Payload field. Figure details the CoAP message format, which delivers low overhead while decreasing parsing complexity.





CoAP Message Format