Security, privacy and device onboarding
The oneM2M approach (based on Release 2A)

François Ennesser & Wolfgang Granzow
oneM2M Security Working Group
francois.ennenner@gemalto.com
wgranzow@qti.qualcomm.com
The Internet ages: From computers to « anything »

**Internet of computers**
- Attended by human « owners »
- Comfortable, controlled environment
- Relatively fixed location
- Low latency broadband connection
- Few chipsets and OSs to secure
- Few Apps largely deployed
- Rather uniform lifetime
- Relatively powerful resources (computing, memory, energy supply)
- Billions of targets online
- Internet as entry point
- Frequent software security patches
- Ever decreasing cost of attacks
- « Virtual world » impact (information)

**Internet of Everything**
- Largely unattended by owners
- Harsh conditions, or physical exposure
- Potentially highly mobile
- Sporadic/constrained throughput/latency
- Diversity of embedded hard and soft
- Multitude of small deployments
- Lifetime from months to decades
- Constrained power, memory, processing
- 100s of billions of targets!
- More, weaker entry points
- Weaker, possibly unmaintained software
- Available and accessible
- Real world impact (physical safety)
IoT Architectures evolution from ICT to industry adoption

« IoT 1.0 »: Upstream **Sensor data acquisition** to **Big Data Analytics** in the Cloud
- Primarily concerned with exploiting *huge amount* of information
- Centralized, many clients to one server, predictable, asynchronous connections
- Addressed by traditional Cybersecurity
- **Privacy** as a main security driver

« IoT 2.0 »: Closed loop **autonomous system** with downstream **actuators control**
- Rather concerned with *processing time* for feedback loop
- Distributed, many-to-many, multi-roles, dynamic, real-time connections
- **critical infrastructures** require physical protection in addition to cybersecurity
- **Human safety** as a strong security driver

**Need to combine Physical safety with Cybersecurity**
Reactive « Patch as needed » virtual security approach applies at *software layer*
But « build it once for good » physical principles are required for *hardware design*
oneM2M security assumptions

• IoT application development requires field experience!
  • Not just Information & Communication Technology (ICT) expertise

• IoT Platform development integrates ICT expertise
  • Should expose underlying services to application

• IoT security countermeasures shall be derived by each stakeholder
  • From application specific risk assessment
  • Considering Privacy and Safety expectations in particular

• Multiple stakeholders that may not trust each others
  • Each stakeholder need to control its own isolated « secure environment »
    • Protect local sensitive information during storage and exchanges
    • And in use (during program execution and data manipulation)
    • Desired protection level conditions security implementation (Hardware + Software)

• Leveraging on common infrastructures and implementations
  • Solutions need to accommodate Trusted Third Parties
Security in oneM2M Release 2A

Expose security services to IoT applications

Enrolment services (RSPF / MEF)
Credentials Provisioning/Security Configuration of the M2M System

Secure communications services (SAEF / MAF)
Methods for Securing Information (PSK/PKI/Trusted Party)
Point-to-point and end-to-end solutions (TLS / DTLS)

Access Control & Authorization services
Requester Authentication
Information access Authorization
Static (ACL based) and Dynamic (token based) solutions
Privacy Policy Management

Device Configuration
TS-0022

Security Solutions
TS-0003

MEF & MAF interfaces
TS-0032
oneM2M Secure Environment and security levels

• « Secure Environment » concept abstracts the security implementation
  • Expose common services to applications, depending on implementation
  • Provide common interface for remote security administration, if needed

• oneM2M supported implementations distinguish 4 security levels
  • No security (!)
    • E.g. for devices otherwise protected from attackers, i.e. on trusted networks
  • Software only security (obfuscation, White box crypto etc.)
    • Always vulnerable to sufficiently motivated attacker
    • Acceptable when compromise is not critical
  • « Trusted Execution Environment » (TEE) relying on main CPU hardware features
    • Good barrier against software based attacks
    • Sufficient for remotely accessible, but not physically exposed devices
  • Tamper resistant hardware embedded Secure Element (eSE)
    • Required to protect secrets within devices physically exposed to attackers (SPA / DPA etc.)
    • E.g. to protect unattended devices against cloning
Onboarding is the procedure of bringing M2M Field Devices into operation in an M2M network. Procedures must cope with a large variety of field device types and Service Provider’s business models.

OneM2M has specified an “M2M Enrolment Function” (MEF) which enables stakeholders to setup their preferred onboarding and enrolment mechanisms in an interoperable way.

1. Field Device, provisioned with credentials, contacts MEF. MEF configures Field Device with parameters and credentials of an IN-CSE.
2. Field device sends registration request to IN-CSE.
3. IN-CSE authenticates the Field Device.

Optional, not in scope of oneM2M specifications.
M2M Enrolment Function (MEF)

- M2M Enrolment Function allows 3 types of Remote Security Provisioning Frameworks (RSPF)
  - Symmetric key authenticated RSPF
  - Certificate authenticated RSPF
  - GBA-authenticated RSPF; in this case the MEF is the Bootstrapping Server Function (BSF) of 3GPP Generic Bootstrapping Architecture (GBA)
- MEF can trigger the Field Device to execute a variety of procedures, including
  - Configuration of Field devices with registration parameters and authentication profiles applicable to the operational Security Frameworks (see next slide)
  - Provisioning of symmetric key credentials
  - Provisioning of certificates (certificate (re-)enrolment using EST and SCEP specified by IETF recommendations)
- MEF is operated by M2M Service Provider or trusted 3rd party (device manufacturer, underlying network operator)
Operational Security Frameworks

- Tie together credential management, configuration parameters, establishing security session (by TLS/DTLS handshake) and protecting the messages or data

  **Security Association Establishment Framework (SAEF):** Adjacent entities
  **End-to-End Security of Primitive (ESPrim):** Originator ↔ Hosting CSE
  **End-to-End Security of Data (ESData):** Data producer to data consumer

---

**Legend:**
- **SA**  Security Association
- **ADN** Application Dedicated Node
- **MN**  Middle Node
- **IN**  Infrastructure Node

---

MN-CSE can see and alter message. What if it is not trusted?

MN cannot see or alter messages
Operational Security Frameworks

• Tie together credential management, configuration parameters, establishing security session (by TLS/DTLS handshake) and protecting the messages or data

  Security Association Establishment Framework (SAEF): Adjacent entities
  End-to-End Security of Primitive (ESPrim): Originator ↔ Hosting CSE
  End-to-End Security of Data (ESData): Data producer to data consumer

Legend:
SA  Security Association
ADN Application Dedicated Node
MN  Middle Node
IN  Infrastructure Node

Protect using ESData

What if IN-CSE is not trusted with this app data

(opt) ESPrim

MN cannot see or alter messages

Protected using ESData.

IN-CSE cannot see or alter app data

IN-AE uses using ESData to extract app data
Message Security between adjacent devices

- Uses (Datagram) Transport Layer Security Protocols, TLS/DTLS Version 1.2
- Several Security Association Establishment Frameworks are supported:
  1) Authentication and session key establishment using **symmetric keys** shared by devices
  2) Authentication and session key establishment using **Certificates** provisioned to devices
  3) Authentication facilitated by an **M2M Authentication Function (MAF)** hosted by M2M-SP or third-party
     • The MAF authenticates the end-points (PSK or certificates) and facilitates establishing a symmetric key
E2E Protection of primitives ("ESPrim")

- Interoperable framework for securing oneM2M primitives
  - CSEs (forwarding the primitive) do not need to be trusted
  - ESPrim provides mutual authentication, confidentiality and integrity protection.
  - Protocol: JSON Web Encryption (JWE) using a symmetric key
    - Symmetric key can be established by pre-provisioning (using MEF), End-to-end Certificate-based Key Establishment (ESCertKE), or central authentication server (MAF)

Diagram:
- Entity A
- MN
- MN/IN
- Entity B
- MEF or MAF
- ESPrim, using JWE
- SA1
- SA2
- SA3

20-Sep-2017
3rd oneM2M Industry Day hosted by TSDSI
12
E2E Protection of selected data (“ESData”)

- Interoperable framework for protecting a selected data portion of a primitive
  - Data to be protected is called the **ESData Payload**.
  - Transited CSEs do not need to be trusted with that data.
  - ESData payload could typically compose all or part of an attribute value (e.g. **content** attribute value of a `<contentInstance>` resource) or a primitive parameter (e.g. a signed, self-contained access token communicated in a request primitive to obtain dynamic authorization).
- Protocol: JSON Web Encryption/Signature (JWE/JWS) or XML Encryption/Signature
Authorization using Access Control Lists

- Access control rules define **who** can do **what** under **which** circumstances
Dynamic Authorization

- **Dynamic Authorization**: Originator or Hosting CSE requesting authorization of Originator – provided by a Dynamic Authorization System (DAS) Server
  - Direct Dynamic Authorisation: Hosting CSE submits request to DAS, Originator not communicating with DAS Server
  - Indirect Dynamic Authorisation: Originator submits request to DAS Server using info provided by Hosting CSE. Similar to Open Authentication (OAuth) mechanism
  - DAS has multiple options for authorizing: Issue/update access control rules, assign Role(s) to the Originator, issue JSON Web Tokens (JWT)

### Direct Dynamic Authorisation

1. **Request**
2. **Originator (AE or CSE)**
3. **Response (with Token or dynamicACPInfo)**
4. **Response (‘success’)**
5. **Hosting CSE**
6. **AE**
7. **DAS Server**

1. Request
2. Originator (AE or CSE)
3. Response (with Token or dynamicACPInfo)
4. Response (‘success’)
5. Hosting CSE
6. AE
7. DAS Server

2017-Sep-3rd oneM2M Industry Day hosted by TSDSI
**Dynamic Authorization**

- **Dynamic Authorization**: Originator or Hosting CSE requesting authorization of Originator – provided by a Dynamic Authorization System (DAS) Server
  - Direct Dynamic Authorisation: Hosting CSE submits request to DAS, Originator not communicating with DAS Server
  - Indirect Dynamic Authorisation: Originator submits request to DAS Server using info provided by Hosting CSE. Similar to Open Authentication (OAuth) mechanism
  - DAS has multiple options for authorizing: Issue/update access control rules, assign Role(s) to the Originator, issue JSON Web Tokens (JWT)

**Indirect Dynamic Authorisation**

1. Request
2. Response (‘no privilege’, Token Request Info)
3. Request for Token
4. Response (Token or Token ID)
5. Request (Token or Token ID)
6. Request with parameters from step 5)
7. Response (with Token)
8. Response (‘success’)

When using Token ID:
6. Request with parameters from step 5)
Privacy Policy Manager (PPM)

• The PPM is a personal data management framework which converts the User’s privacy preferences into access control information in order to protect the User’s Personally Identifiable Information (PII) from access by unauthorized parties.

• Access control information consists of static and dynamic access control policies (ACP) and policies for issuing access Tokens

• Uses a “Terms and Condition’s Mark-up language” to derive consensus between the User’s privacy preferences and the ASP’s privacy policies

**Diagram:**

1. **User privacy preferences**
2. **ASP privacy policy**
3. Creates user-friendly customized ASP privacy policy for User
4. Customized privacy policy
5. Accept/decline
6. Creates access control policies (ACPs)
7. ASP’s AE requests PII
8. ACPs or dynamic authorization
9. PII

**KEY**
- When User and ASP Register w/ PPM
- When User subscribes for ASP service
- When ASP requests PII

**Application Service Provider (ASP)**

**M2M Device** (source of PII)

**CSE** (hosting PII)

**Privacy Policy Manager** (operated by M2M SP or trusted 3rd party)

**User**

**AE**
Time for questions

• For further reading: www.oneM2M.org