Security and Trust for M2M Communications

Inhyok Cha, Yogendra Shah, Andreas U. Schmidt, Andreas Leicher, Mike Meyerstein

Abstract—**The advent of M2M communication brings with it numerous challenges, including those in security, that traditional communication models have not yet addressed. Particularly, due to their inherent un-guarded, low cost and mass-deployed nature, M2M devices, and wireless communication architectures and solutions for such devices, would invite new threats in security. These threats may not be fully addressed by use of security technologies and methods adopted in existing wireless devices, cellular networks or WLANs. This paper attempts to put forward approaches which would address these new threats.**

***Index Terms*—M2M, security, trust, wireless**

I. Introduction

Machine-to-machine (M2M) communication is considered as one of the next frontiers in wireless communications. Freed from the traditional constraint that terminals communicating wirelessly with networks should be largely 'manned' by humans, communication from and to M2M devices is expected to open up a large number of possibilities in terms of new use cases, services, and applications. M2M will also bring benefits for the general masses and market opportunities for various related stakeholders, such as manufacturers of M2M devices and components, service providers, and communication network operators.

Considering the large number of M2M devices expected to be deployed, in a highly distributed network, global enforcement of security will not be practically feasible due to the low cost of many of these devices and cost of implementation. As the conventional centralized IT security network model, protected by a firewall, becomes challenged by the need for a dispersed model, de-centralized methods for establishing security are being explored. The growing trend towards de-centralized systems produces numerous situations in which enforcement, by practical necessity, has to be complemented by controlled risk. The principles of enforcement embraced by traditional concepts of access control and policy enforcement are being supplemented by a paradigm shift to incorporate "trust." An entity can be "trusted" if it predictably and observably behaves in the expected manner for its intended purpose. By delegating parts of the enforcement tasks to trusted elements dispersed in a system, transitive trust relationships can be established. This is the most advanced evolution of the organizational method of separation of duties within IT security. This evolved security model, which balances trust and enforcement, results in a useful, practical and scalable approach to IT security and, is suitable for M2M communications.

M2M devices will ultimately connect to core network services through a variety of means, from direct broadband or capillary wireless networks, to wired networks. Capillary networks used by M2M systems are made of a variety of links, either wireless or wired. Connectivity to these wireless and wired networks is an essential part of the M2M communications network. There is a need to be able to integrate a variety of application-specific technologies into a complete end-to-end solution to be offered by service providers. Customers may choose one or another capillary network technology, based on their requirements. The case for addressing security, which blends the trust and enforcement model from a comprehensive end-to-end system perspective, is a critical factor for the overall success of the M2M market.

II. M2M use cases

Various standards organizations have identified a number of use cases for M2M communications. This section provides more detail on some of the use cases, covering the most important user requirements in order to clarify the security requirements on M2M systems.

All of these use cases, or applications, have some common security requirements. Since the devices are typically unmanned, and a high value is placed on the information handled and communicated by these devices, the security of the information and trustworthy operation of these devices is of paramount importance, as is the ability to manage over the air.

A. Use Case 1: Traffic Cameras

Traffic cameras with cellular connectivity may be installed in locations such as motorway overpasses or remote stretches of roadway. Cameras may also require simultaneous secure local WLAN connectivity to the next camera down the road, e.g., when measuring average speed.

B. Use Case 2: Metering

Almost every house and office building has a gas, water, and electricity meter that measures the use of these utilities. When it comes to measuring the usage, the process is time inefficient and, therefore, a costly affair. Automatic meter reading is the technology of automatically collecting data from energy meters, and transferring that data to a central database for analysis and/or billing. Additionally, as the adoption of smart metering technologies grows, the need to monitor usage on a real-time basis becomes more and more compelling as a business case to optimize use of a smart grid for both energy usage, and for individual households to contribute to the electrical grid.

C. Use Case 3: Vending Machines

Vending machines are an efficient and cost-effective method of distributing retail goods and it's important that the service provider keep track of stock levels, and whether the machines are working properly. Vending machines are subject to regular attacks on their contents. This increases the threat to other items of value in the machine, as well as to the collection of electronic payments. Additionally, in some advanced applications, there is a desire to push multi-media marketing to displays in the vending machines. Vending machine connectivity may come from a variety of connectivity options within the customer premises.

D. Use Case 4: Asset/Cargo Tracking

A remote asset/cargo tracking system allows owners or users of equipment to, for instance, monitor critical parameters, perform remote commands, or monitor movements. Asset and cargo tracking will often require that the M2M device be placed in areas where physical access is difficult. Such placements would be part of a service provider's attempt to protect it from the environment, and to resist theft and tampering of the M2M device. This placement, together with the fact that the M2M device may be mobile, can make it difficult and costly to physically access the M2M device.

III. Security Threats for M2M

M2M devices have unique characteristics and subscription and deployment contexts [1]. M2M devices are typically required to be small, low cost, inexpensive, able to operate unattended by humans for extended periods of time, and to communicate over the wireless WAN or WLAN. M2M devices are typically deployed without having to require much direct human intervention and, after deployment, they tend to require remote management of their functionality. They also require flexibility in terms of subscription management. In addition, in many use cases, it is likely that M2M devices will be deployed in very large quantities, and many of them will also be mobile, making it unrealistic or impossible for operators or subscribers to send personnel to manage or service them.

These requirements introduce a number of unique security vulnerabilities for the M2M devices and the wireless communication networks over which they communicate. These security vulnerabilities are described in the following categories:

1. **Physical Attacks** may include insertion of valid authentication tokens into a manipulated device, inserting and/or booting with fraudulent or modified software ("re-flashing"), and environmental/side-channel attacks, both before and after in-field deployment. These possibilities then require trusted 'validation' of the integrity of the M2M device's software and data, including authentication tokens.
2. **Compromise of Credentials** comprising brute force attacks on tokens and (weak) authentication algorithms, physical intrusion, or side-channel attacks, as well as malicious cloning of authentication tokens residing on the Machine Communication Identity Module (MCIM).
3. **Configuration Attacks** such as fraudulent software update/configuration changes, mis-configuration by the owner, subscriber or user, mis-configuration or compromise of the access control lists.
4. **Protocol Attacks on the Device.** These are directed against the device. Major examples are man-in-the-middle attacks upon first network access, denial-of-service (DoS) attacks, compromising a device by exploiting weaknesses of active network services, and attacks on OAM and its traffic.
5. **Attacks on the Core Network.** These are the main threats to the mobile network operator (MNO): Impersonation of devices, traffic tunnelling between impersonated devices, mis-configuration of the firewall in the modem/router/gateways, DoS attacks against the core network. They may also include changing the device's authorized physical location in an unauthorized fashion or attacks on the radio access network, using a rogue device.
6. **User Data and Identity Privacy Attacks** include eavesdropping for other user's or device's data sent over the UTRAN or EUTRAN; masquerading as other user/subscriber's device; user's network ID or other confidential data revealed to unauthorized third parties, etc.

Some of the vulnerabilities that are more specifically geared to the subscription aspects of the M2M devices have also been identified in [2], and interested readers are referred to that document.

IV. The Trusted Environment

To establish trust relationships in dispersed systems, it is essential that the systems contain security-relevant elements and capabilities which form the building blocks of the trust boundaries. These components include methods to extend the trust boundaries and convey trust to an external entity. A Trusted Environment (TRE) provides a hardware security anchor and root of trust, allowing for the construction of systems which combine characteristics of trust and enforcement.

A. Introduction of Trusted Environment

The TRE is a logically separate entity within the M2M device. It contains all necessary functions and resources to provide a trustworthy environment for the execution of software and storage of sensitive data. The TRE provides isolation of software and stored data by separating them from the M2M device, thus protecting them from unauthorized access. To protect the system behavior, the TRE provides a hardware security trust anchor. This part is secured against tampering by hardware security measures. In particular, it holds the Root of Trust (RoT) for secure operation. The RoT secures the internal system operation and is able to expose properties, or the system's identity, to external entities. Based on the RoT, the TRE is protected by a secure start-up process which ensures the TRE reaches a determined trustworthy state. This secure start-up process includes all components and programs that are executed during the system boot, and can be extended to the operating system and software, thus expanding the trust boundary provided by the RoT. A model for this extension process is the verification of every new component when it is loaded, by measuring its integrity at the time of its initialization. This method uniquely identifies every component, its state and configuration. The measurement result is then stored and integrity is protected by the TRE. Furthermore, this measurement can then be compared to reference values, and the verification entity can then decide to include this new component in the extended trust boundary. As verification is intended to take place locally, it relies on the assumption that the TRE is in a predefined state after a completed verification process. Validation, on the other hand, requires that a reporting entity transfers the results of verification to an external party. The external validator can then assess the device's system state. Validation makes the predictability of the TRE's functions observable and, thus, trustworthy. In addition to the assessment of the M2M device's

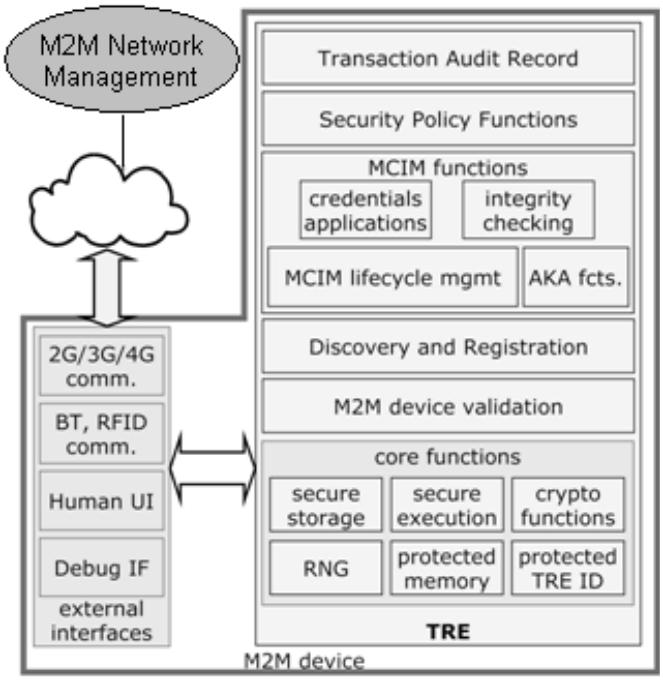

Fig. 1. Components and Interfaces of TRE in a M2M Device

trustworthiness, the TRE can provide protected functions for the authentication of the M2M device towards the operator's network. This can be achieved by storing the authentication data inside the TRE.

B. Requirements, Functionality and Interfaces

The TRE must provide cryptographic capabilities to perform security-related functions. These include symmetric and asymmetric encryption and decryption, hash value calculation and verification, random number generation, and digital signature generation and verification. In addition, the secure storage for keys, credentials, and authentication data must be provided by the TRE. The storage area can be inside the TRE or outside the TRE but security is protected by the TRE, e.g., by encrypting the data with a key stored inside the TRE. Furthermore, the TRE must be able to establish secure channels for the communication with other parts inside the M2M device, but outside the TRE. The interfaces which are needed for this communication are initialized in the secure start-up process, integrity- protected by the TRE, and, hence, can be assumed to operate correctly. Two categories of interfaces can be distinguished. Protected interfaces provide integrity protection and/or confidentiality of the data carried across them. This can be achieved by the use of security protocols or hardware interfaces. If security protocols are used, further functionality such as authentication of the entity with which the TRE communicates, and message authentication and/or confidentiality, can be provided. Unprotected interfaces facilitate communication between the TRE and general resources of the M2M device which is not assumed to be secured against tampering and/or eavesdropping. These unprotected interfaces can, nevertheless, give access to data which is cryptographically protected by the TRE, for instance when the TRE is in possession of pertinent key material, and cryptographically secures data stored in unsecure memory. Even unprotected interfaces can benefit from other security measures such as making the interface available only after the TRE checks the code of its counter-part resource across the interface, for example during a secure boot-up of the M2M device. Figure 1 shows the components and interfaces of the TRE in a M2M device.

V. Verification of Trustworthiness of M2M

The very specific practical requirements of, and security threats in, M2M application scenarios have two main aspects: a) unpredictable connectivity to the core network, and b) demand for high configurability and flexibility of the equipment. Both aspects arise at specification time of a M2M device and its interaction with the network, and must be taken into account early on, to enable a broad range of use cases with optimal cost-efficiency. We view fulfilling b) under the condition a), as the main obstacle to be overcome for the take-off of the M2M market.

Security is of the essence in the technological problem described above, and amounts to satisfying two concrete protection goals:

- Ensure that M2M device can reach and operate, locally, in a secure state without network connectivity.
- Enable establishment of assurance, locally and remotely, on the state of the M2M device, to assess its security properties and, hence, trustworthy operation.

Perhaps the most basic and important role is played by these protection goals when the state of a M2M device is to be changed in a controlled way, either by over-the-air (OAM) or local management. Both stakeholders, network operators and M2M device owners, have a clear mutual interest to have independent abilities to *validate* a M2M device's state in such a case.

Validation means the ability to technically assess the state of a system for all security-relevant properties. The argument for dedicated technical means to validate a M2M device is applicable to its whole lifecycle, from manufacture, initial deployment (validation for installation verification), maintenance (validation for diagnosis and success verification), through to OAM (validation of correct configuration change).

Means to validate a system in its operational phase are different from pre-deployment testing and certification, or formal security proofs on a design. They lead into the realm of trusted systems and trusted computing. We can distinguish three main variants. We name **autonomous validation** the model of closed systems, like smart cards, which arrive at a secure state merely by local means, and do not communicate state information to the exterior. As the other extreme, **remote validation** is an abstraction of what the Trusted Computing Group (TCG) has specified as *remote attestation*. Basically, an open (meaning mostly unrestricted in operational state changes) system only reports state information in a secure way in this second option. A broad spectrum of other variants lies in between the range marked by these extremes. We call this spectrum (not a single method) **semi-autonomous validation (SAV)**. There exists just one concrete example, at least on the level of technical specifications, namely, what the TCG's Mobile Phone Working Group has specified as *secure boot* [3].

Let us now try to get a stronger grip on the three notions, and discuss their pros and cons.

A. Autonomous Validation

Autonomous validation is a procedure by which the validation does not depend upon external entities, and local verification is assumed to have occurred before the M2M device will attempt communication with the exterior or other critical operation. The local verification process is assumed to be absolutely secure, as no direct evidence of the verification is provided to the outside world. The outside world makes the assumption that, due to the way in which all M2M devices are specified and implemented, a device which fails verification will be prevented from attaching itself to a network. Autonomous validation lays all enforcement duties on the implied trust in the device. Autonomous validation is applying a closed, immutable system model, essentially the trust model used for smart cards. The result of the local verification is a binary value "success" or "failure". Validation by an external, relying party is then an implicit process, e.g., during network attachment. A typical example is the release of an authentication secret, e.g., a cryptographic key, by a smart card.

Security resting only on devices has been broken in the past and is more likely to be broken as devices become open computing platforms. Autonomous validation delivers no information for advanced security requirements: in particular, if the device is partially compromised, the exterior cannot gain any knowledge about its state other than through inference from its behavior towards the network. Labeling of rogue devices can, therefore, be impossible, meaning that an exploit might proliferate without being noticed and cause significant damage to other stakeholders, such as network operators, before it can be contained.

Autonomous validation may be realized in such a way that verification is reactive to certain conditions, e.g., by not allowing certain functions, or by closing the device down and going to re-boot, depending on failure policy. This avoids network connection and seems advantageous. But such a feature is also a vector for denial-of-service (DoS) attacks. The device must not attach to the network in a compromised state and, thus, has little chance to revert to a secure state. Compromised devices could only be recovered by an in-field replacement which is to be considered costly, due to the distributed nature of M2M application scenarios. Remote management is also difficult. Specifically, there may be a loss of security in software download and installation since it potentially delivers values (software, secrets) to rogue devices. Thus, autonomous validation is prone to entailing out-of-band maintenance. For instance, failure of the update of software may lead to a state in which network connection is impossible. A lot of burdens and risk rests with the owner of such an M2M device. But the core network also bears additional burden since it has to keep track of the state of every autonomously validating device. That is, if their states change, for instance, by an externally forced update, this is only signaled through the next re-validation, which has no further informational content. It is the relying party's duty to update his database with the new device state. If multiple parties can force updates on a device, this may become complicated.

Finally, with autonomous validation, the freshness of the information on local verification is not, by itself, guaranteed. For this security property to be fulfilled, autonomous validation would have to take place automatically on every system state change, strictly speaking. As autonomous validation happens infrequently in practice, e.g., during network attachment, the M2M device's state may change significantly during its operation, in a manner unobservable by the relying party. Thus, an attacker may use this gap, for instance, to introduce malicious software. Autonomous validation is extremely prone to this kind of timing attack.

B. Remote Validation

In **remote validation**, the relying party directly assesses the validity of the device based on the evidence for the verification he receives. The local verification is only passive in this case, just measuring integrity values of the loaded and started components. The full Stored Measurement Log (SML) must be conveyed to the relying party. All policy decisions rest with the relying party. In a remote attestation, a TCG trusted platform exhibits a SML and Platform Configuration Register (PCR), signed by an Attestation Identity Key (AIK) to the relying party. The AIKs are ephemeral asymmetric key pairs, certified by a Privacy Certification Authority (PCA) which acts as an identity provider for the sole purpose of validation. More details on this process are found in [4]. The pseudonymity provided in remote attestation may not be sufficient in all cases. The TCG has additionally defined Direct Anonymous Attestation (DAA) [5, 6], which is based on zero-knowledge proofs [7].

Remote validation, as represented by remote attestation, poses practical problems with respect to scalability and complexity, as it lays the full computational load for validation on (central) access points to networks or services. The validation of an SML can be very costly for open platforms with a large number of software and hardware components in numerous versions and configurations. This also requires an enormous database of reference values, viz. Reference Integrity Measurements (RIMs), together with an infrastructure, to let stakeholders define the desired target configurations of devices. The same arguments make remote management of a device, i.e., the controlled and validated change of configuration, impractical with remote validation. Furthermore, run-time verifications are desirable with remote validation, as otherwise only the state after boot is exhibited to the relying party. The SML can be "withered" at time of validation. Thus, run-time verification becomes meaningless if it is not directly followed by validation, which would necessitate very frequent remote validations. Finally, remote validation of complex open devices compromises privacy, in spite of usage of a PCA, since the revealed SML might be almost unique to a device. A similar, economic argument is the possibility of discrimination by remote attestation, that is, the threat that only recent versions of software of major vendors enters into RIM databases, forcing users of other programs to switch to these or lose service access. Some of the disadvantages

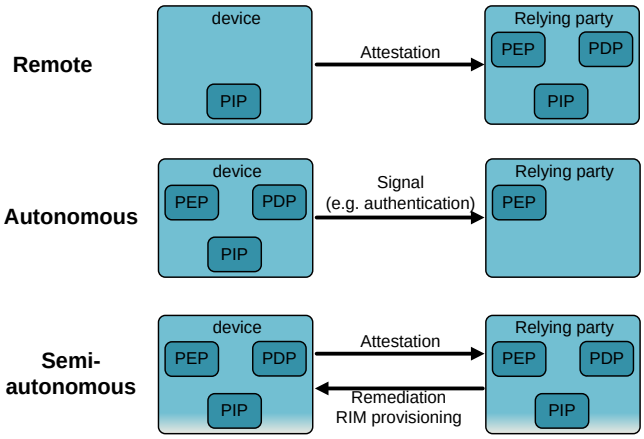

Fig. 2. Mapping Variants of Validation to Policy Enforcement.

might be alleviated by refined forms of remote attestation, such as semantic [8] or property-based attestation [9, 10], aiming at exhibiting the characteristics of components, rather than a concrete implementation. These options, however, need more research before they may become practical.

C. Semi-Autonomous Validation

Semi-autonomous validation is any procedure whereby the device validity is assessed during verification within itself, without depending on external entities, and policy decisions are made and enforced during this local verification. But in this case, the result of the verification and required evidence are signaled to the relying party, who can make decisions based on the content of the validation messages from the device and, optionally, additional information from a certifying or third party that provides security capabilities and architecture of the M2M device. The signaling from the device to the relying party must be protected to provide authentication, integrity, and confidentiality, if desired. Thus it must come from the TRE. A model case for semi-autonomous validation is secure boot, followed by signaling of the so called event structure [3] and indication of RIMs to the relying party. Semi-autonomous validation symmetrically distributes verification and enforcement tasks between device and relying party. Specifically, in secure boot, the former makes decisions at load time of components, while the latter can enforce decisions on the interactions permitted to the M2M device upon validation, based on the state evidence provided.

Semi-autonomous validation may be a promising avenue to a remedy for the disadvantages of the other two options. It can potentially transport the validation information more efficiently in the form of indicators of the RIMs used in verification. This can also be used to protect privacy, for instance, when such an indication designates a group of components with the same functionality and trustworthiness (such as versions). The interplay of enforcement in verification during validation on the part of the relying party, also opens options for remote management of devices.

On the path to technical realization of such opportunities, the Trusted Network Connect (TNC) working group of the TCG has introduced the concept of remediation [11], to obtain "support for the isolation and remediation of ARs [Access Requestors] which do not succeed in obtaining network access permission due to failures in integrity verification." (p. 24). This allows, in principle, "to bring the AR up to date in all integrity-related information, as defined by the current policy for authorization. Examples include O/S patches, AV [Antivirus] updates, firmware upgrades, etc." (p. 25). Concrete concepts for realization of remote management will have to rely on an infrastructure for the efficient representation and communication of RIM information. TCG MPWG has started to define such services for mobile devices [3], in particular to ingest RIMs for verification. The Infrastructure Working Group of the TCG is establishing a generic architecture and data structures for verification and validation [12]. More research and development is, however, needed to devise efficient and effective semi-autonomous validation on this path.

It is important to emphasize the role played by RIM certificates in semi-autonomous validation. RIM certificates are provided by a certification authority which has assessed, directly or by delegation, the corresponding component. Certification methods and bodies can be diverse and lead to different levels of operational trustworthiness. This leads to further flexibility for a semi-autonomous relying party who gets more fine-grained information on the device.

Semi-autonomous validation is also the only practical validation option for systems which are resource-limited so that, a) they lack the security qualities of a closed system needed to achieve autonomous validation, or b) lack the memory and/or communication capabilities to perform the extensive reporting needed for remote validation.

D. Validation and Enforcement

Validation and local verification are the central conceptual link between trusting a device and enforcing policies on its behavior. This is because, based on the results of validation, various policy decisions can be made, and verification, in turn, can incorporate enforcement during secure boot. We can map the concepts of validation, in the three variants described above, to the basic architecture of enforcement systems. Fig. 2 shows a simplified picture. A common trait of all variants is that the device needs a Policy Information Point (PIP) as minimal resource to support the validity decision by the relying party. The PIP has to perform the measurement of the device's state for this, and to securely record the results. Since validation is always performed for a purpose, there is, in all cases of practical relevance, a Policy Enforcement Point (PEP) present at the part of the relying party. Based on the attested information, it can enforce decisions such as granting network access. The richness of the latter information varies significantly between the validation variants.

In remote validation, the device has no other means to build trust with the relying party than to transmit the full SML to the relying party, plus information binding it to the device's state and protecting its authenticity, e.g., signed PCR values. The relying party's PIP has to contain a database of possible allowed device states including reference measurement values. Based on the attestation and the state reference information, the Policy Decision Point (PDP) at the relying party re-traces the SML, e.g., re-

calculates the digest values. The PEP obtains a graded result from this process, stating up to which position in the SML, the TS was in a good state. On this information, the PEP acts, for instance, by (dis-)allowing network access.

In autonomous validation, all functionality for measurement, verification, and enforcement during secure boot and runtime is localized in the device's PIP, PDP, and PEP, respectively. No explicit attestation statement is made to the relying party who has to rely on the implicit signal that can be inferred, e.g., from an authentication attempt. The relying party's PEP can enforce only policies based on the static information contained in this signal, e.g., system type or identity. Since validation information is not present, no validation-specific PIP and PDP are used at the relying party (however, a non-validation PIP and PDP can be constructed, in this case, based on TS identities and connection history – in effect, a traditional authentication, authorization, and accounting [AAA] system [13]).

Semi-autonomous validation allows for equally capable policy systems on both sides. The key to this is a codification of attestation data. It need not be transferred as a complex SML including measurement values of all components. It is replaced by a concise event log containing, essentially, references to RIMs, respectively, associated certificates (the precise content may depend on implementation requirements). This abstraction is made possible by the PDP in the device, which, at the time of verification, e.g., during secure boot, makes the association of component to target RIM. For that, it relies on an internal, protected, RIM database, whose management adds to the functional role of the PIP (beyond measurement). Attestation to codified RIMs allows interaction with the relying party in validation. The PDP of the relying party can use its own RIM database (provisioned by its PDP) to compare the attested state with fine granularity to a desired state. The PEP communicates the outcome to the device's TRE and can thus initiate, i) provisioning of new RIMs to the device, ii) unload of undesired components, iii) load of new, desired components, and by that finally, iv) updates of components. These processes are captured by the term remediation. To show the success of the remediation, the device needs, in principle, to revalidate only using the newly adjoined part of the event log. From the viewpoint of policy systems, RIMs add an essential piece to enable general policies for validation: A codified ontology on which conditions can be evaluated and decisions be taken.

VI. TRE Application to Subscription Management

A. Subscription Management

Unlike mobile phones, M2M devices operate largely without human intervention and protection. Also, unlike a mobile phone where the subscription is usually between one human owner/user who owns one phone to a network operator, in the case of M2M communication, the subscriber is usually an operator of a large number of M2M devices to provide service to their end-customers, and have a many device to one network operator subscription relationship. The situation is made more complex because, typically, the M2M subscriber may want to decide who the network operator will be for his machines AFTER the M2M devices are deployed in the field and left unattended. Such subscriber may also want to be able to change the subscription for his individual M2M devices from one network operator to another, again in the field, without direct servicing by human personnel.

Remote abilities to manage all or most aspects of WAN communication subscription for M2M devices, thus, becomes a highly desired feature and a likely requirement for fast uptake of M2M communication in the market place.

Since the 1990s, 2G and 3G cellular operators have grounded the subscription management of mobile phones by mandating the use of the UMTS Integrated Circuit Card (UICC). The UICC is a removable smart card specifically developed to house the software functionality of UMTS Subscriber Identity Module (USIM) for 3G UMTS cellular phones. It is well suited for subscription management for phones where the phone's owner/user can be expected to protect and manage the use of the removable UICC on his phone. Such assumptions, however, do not hold for most M2M devices.

The 3GPP Security Workgroup (SA3) has lately been describing in [2] an approach that allows remote management of M2M subscription, in the context of initial provisioning of services and also later change of subscriptions, all remotely performed. In this model, the TRE of an M2M device is defined so that it can function as both the secure protector of downloadable subscription credentials and secret key material, and secure executor of authentication and key agreement (AKA) functionality, based on the credentials and key material downloaded.

The following Figure 3 depicts the network architecture of the so-called architecture alternative 1 in [2].

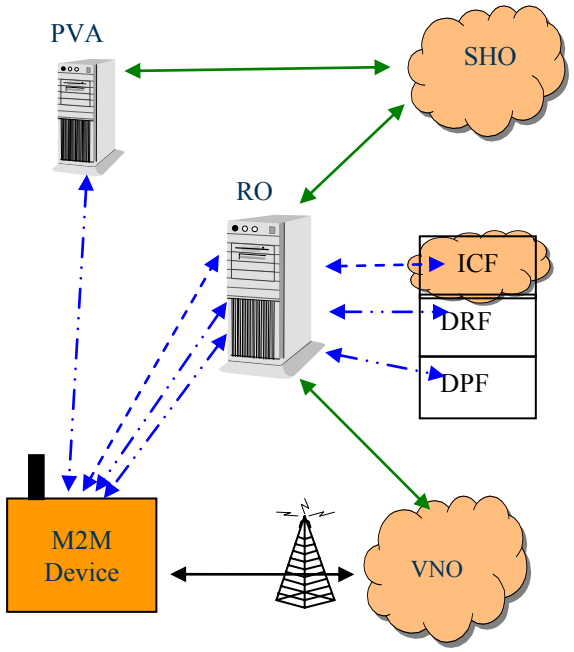

Fig. 3. Network authentication and MCIM provisioning in the M2ME, using 3G authenticated access (c.f. [2] Figure 5.1.3.7.1-1)

In this architecture, M2M devices can first obtain, via the air interface provided by a Visited Network Operator (VNO), a provisional IP connectivity provided by the Initial Connectivity Function (ICF) of a Registration Operator

(RO). The RO also provides a Discovery and Registration Function (DRF), which aids the M2M device to discover and register itself to the intended home network operator, called the Selected Home Operator (SHO). In order for the M2M device to be able to connect to the RO and obtain information about the SHO, it needs to be pre-provisioned with a temporary credential called the PCID (Provisional Connectivity ID).

After the M2M device finds and registers itself with the SHO, it then needs to download the SHO's subscription credential to a trusted environment (TRE) within itself. Before the SHO allows this download procedure, however, it first must validate the integrity of the M2M device. This is to be done by an entity named the Platform Validation Authority (PVA). After the PVA verifies the integrity of the M2M device, the SHO can use the RO's Download and Provisioning Function (DPF) to download and provision its own subscription credential and applications into the M2M device's TRE. The method in [2] also makes it possible for the M2M device's subscription to be changed from one operator to another, without requiring personnel to visit the M2M device in the field and replace the UICCs.

B. UICC vs TRE based Subscription Management

The authentication technologies for M2M device subscriptions can include a download capability for remote subscription management in the TRE in the M2M device, or in the UICC. The TRE provides a secure execution and storage environment for authentication applications that can be downloaded to the TRE. While UICCs are widely used and established in current mobile network environments, UICCs provide no options for the download of subscription identities. The possibility of using downloadable manageable identities (MIDs) for subscription management is a key benefit of TRE usage over UICC. If UICCs are to be equipped with more memory and more functions, they will become more costly and further hamper the take up of cost-conscious M2M applications, while a TRE could be built directly into the chipset of the M2M device. Being physically bound to the device, the TRE provides a secure solution for the subscription management for M2M devices. A traditional smart card has a single chip architecture, with little possibility for monitoring communications between different chips on the card. Smart card ICs are designed and implemented to prevent probing and reverse-engineering. Measures include scrambling of busses and of memory addresses, bonded passivation layers, permanently disabled test points, and self-generated programming voltage.

The smart card's standardized API consists of a restricted command set that has no hidden commands or access methods. It is trusted because of its own built-in security mechanisms and because of those of the underlying hardware platform. It is non-updateable. For applications such as USIM, the Ki and OTA keys are stored and accessed by the O/S in proprietary ways. The O/S cannot be made to reveal the values or memory locations of those data. Conventional GSM SIM cards did not allow adding applications to a live card. The advent of the Javacard [15] now allows applications on a multi-application UICC platform to be updated, deleted or added to an issued card, either remotely or locally. General concerns about the security of a multi-application Javacard, and the limited bandwidth implementation of using SMS as the bearer for messages to the UICC, restricts the Javacard's potential.

A UICC to be used in mass-market M2M applications would have to support the requirements of long life-time with long maintenance intervals, non-removability, remote download of operator's authentication application and remote change of operator. Therefore, the unauthorized removal of the UICC must be prevented, while still providing the possibility to replace the UICC in a service process. Another option is to fix the UICC in the M2M device and have the ability to download MIDs to it. The UICC must then be able to extract Ki objects and other sensitive data sent from a remote server, and store them in secured memory. Furthermore, M2M applications require the support for secure download protocols which do not require pre-shared keys and which can be used across IP bearers.

While UICCs can be protected from side channel attacks, especially power consumption analysis, during their design stage, the physical structure of the TRE would be such that it would not leak any information. Any noise occurring would contain no recoverable information, and no interfaces are provided to monitor the processing of the TRE. There are no standardized specifications, as such, that assure the degree of physical and logical tamper-resistance. It is up to the buyer to specify what he wants. The same might not be true of embedded TREs, if they use TCG-style technology and conform to protection profiles. There is never a guarantee that, for a given advertised UICC product, all or any of the possible counter-measures have been implemented. Network Operators can obtain un-published information about counter-measures from card vendors, and large-volume buyers can specify their own counter-measures, within the constraints of the silicon manufacturing process. Network operators trust the UICC because they are in charge of the specifications and they buy them directly from their limited set of approved vendors. It is necessary to establish an international accreditation scheme for TREs for the network operators to assess their security. This also helps to overcome the lack of confidence in the ability of terminal suppliers to implement a secure environment. Using an appropriate designed architecture, there could be only a small number of TRE vendors, reducing the number of TRE solutions. It is imaginable to have reputable chip manufacturers, with a proven track record in the security industry, manufacture and personalize the TREs.

As current OTA procedures and infrastructures are not secure enough for the downloading of MIDs, they need to be updated. This could cause backward compatibility problems. New protocols, allowing the use of an IP bearer for download protocols, can be implemented using an embedded TRE. The architecture mentioned in the previous section requires the network operators to pass their Ki around to many different entities. The Ki can be strongly encrypted for the TRE of the designated M2M device by the entity that generates it. In this case, it can be securely handled by all parties involved in the process of

downloading it to the M2M device.

VII. Conclusion

M2M communication applications and scenarios are growing and lead the way to new use and business cases. Due to the nature of M2M scenarios, which involve un-guarded, distributed devices, new security threats have emerged. The use case scenarios for M2M communications bring with them new requirements for security that also address the new requirement on flexibility, due to deployment scenarios of the M2M devices in the field. We believe that these new requirements require a paradigm shift. One important pillar of such a shift will be a new, more balanced mix of device-centric trust and traditional enforcement of security properties. Mixing and distributing trust building and enforcement tasks between device and network leads to scalable concepts which can be adapted flexibly to the technical tasks. The two most important building blocks for this are local state control, via secure boot, and conveying trust by semi-autonomous validation. By embracing these advanced concepts of security, the unique needs of the M2M market, that of remote management of the platform and for subscription management, can be met. The presented ideas enable new business opportunities aligned with the goal of achieving hardware-backed security.

References

[1] 3GPP. "3GPP Technical Report 22.868 v8.0.0 Study on Facilitation of Machine to Machine Communication in 3GPP Systems," March 2007.

[2] 3GPP. "3GPP Technical Report 33.812 draft version 1.3.0 Feasibility Study on Remote Management of USIM Application on M2M Equipment," February 2009.

[3] TCG. "TCG Mobile Trusted Module Specification Version 1.0. Revision 6;" "TCG Mobile Reference Architecture Specification Version 1.0. Revision 5," 2008.

[4] A. Leicher, N. Kuntze, and A.U. Schmidt, "Implementation of a Trusted Ticket System," in *Proceedings of the IFIP sec2009*. Boston, MA, USA, 2009, to be published..

[5] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in *Proc. 11th ACM Conf. on Computer and Communications Security,* 2004, pp. 132-145.

[6] J. Camenisch, "Better Privacy for Trusted Computing Platforms," in *Proc. 9th European Symposium On Research in Computer Security (ESORICS 2004),* 2004, pp. 73–88.

[7] D. Chaum, "Security without Identification: Transaction Systems to make Big Brother Obsolete," *Communications of the ACM,* vol. 28, no. 10, pp. 1030–1044, 1985.

[8] V. Haldar, D. Chandra, and M. Franz, "Semantic remote attestation: A virtual machine directed approach to trusted computing," in *USENIX Virtual Machine Research and Technology Symposium (VM '04)*, 2004, pp. 29-41.

[9] A.-R. Sadeghi, and Ch. Stüble, "Property-based attestation for computing platforms: caring about properties, not mechanisms," in *Proc. 2004 workshop on new security paradigms NSPW '04*, pp. 67–77, 2004.

[10] L. Chen, R. Landfermann, H. Löhr, M. Rohe, A.-R. Sadeghi, Ch. Stüble, and H. Görtz, "A protocol for property-based attestation," in *Proc. first ACM workshop on Scalable trusted computing (STC '06)* pp. 7-16, 2006.

[11] TCG, "TNC Architecture for Interoperability. Specification Version 1.3. Revision 6," 2006.

[12] TCG, "TCG Infrastructure Working Group. Architecture Part II - Integrity Management. Specification Version 1.0 Revision 1.0," 2008.

[13] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, and D. Spence, "Generic AAA Architecture" *IETF Network Working Group. RFC 2903*.

[14] TCG. "Trusted Computing Group Architecture Overview Version 1.4," 2006.

[15] Global Platform specifications, v 2.2, see www.globalplatform.org