An M2M gateway-centric architecture for autonomic healing and optimizing of Machine-to-Machine overlay networks

Amine Dhraief, Abdelfettah Belghith, Hassan Mathkour, Khalil Drira

An M2M gateway-centric architecture for autonomic healing and optimizing of Machine-to-Machine overlay networks

Amine Dhraief[1], Abdelfettah Belghith, Hassan Mathkour[2], and Khalil Drira[3]

[1]*HANA Research Laboratory, University of Manouba, Tunisia*
[2]*College of Computer and Information Sciences, King Saud University, Saudi Arabia*
[3]*LAAS-CNRS, University of Toulouse, France*

Abstract

The Machine-to-Machine (M2M) technology, currently under standardization at both the ETSI and 3GPP, is expected to be one of the most promising revenue-generating services. However, to ensure the wide spread of this technology, M2M communications should be secure, fault-tolerant and self-managed. In this work, we add to the M2M gateway (an aggregator node in the M2M architecture) the self-healing and self-optimizing autonomic capabilities. We couple at the M2M gateway level the Host Identity Protocal (HIP) with the Reachability Protocol (REAP). REAP enables a self-healed M2M communication as it detects possible failures and seamlessly rehomes an M2M established session to a new working overlay path. Furthermore, we modify REAP to ensure self-optimized M2M communications. REAP continuously monitors M2M overlay paths and always selects the best available ones in terms of RTT. We implement our solution on the OMNeT++ network simulator. Results show that M2M sessions effectively resume after an outage affecting their currently used M2M overlay paths. Results also highlight that M2M sessions autonomically select the best available M2M overlay paths.

1 Introduction

Embedded systems such as sensors, smart meters and smart cards are experiencing a tremendous proliferation. Several market forecast predict that the number of these devices will soon outnumber the people on earth. According to the Wireless World Research Forum (WWRF), by 2017 we will have 7 trillion wireless devices serving 7 billion people [59]. Juniper Networks predicts that in 2015, the number of connections between embedded equipments will reach over 500 millions [34]. Machine-to-machine (M2M) communication is considered to be an adequate framework to handle the communication between these embedded systems

and their corresponding applications. M2M communication is a novel communication technology under standardization at both the European Telecommunications Standardization Institute (ETSI) [22, 21] and the 3rd Generation Partnership Project (3GPP) [58]. M2M communication is based on an autonomous communication between sensors/actuators and correspondent application over the Internet. The M2M architecture introduces a new level of indirection between the sensors/actuators and the application namely the M2M gateway. The M2M gateway aggregates data packets received form sensors and sends them to the M2M application. It generally communicates with M2M devices via short range communication technologies.

The telecommunication industry is energetically supporting the spreading of the M2M technology as it is expected to be one of the most promising revenue-generating service. Nonetheless, from a standardization point of view, the M2M paradigm is still in its infancy [41]. Both ETSI and 3GPP standards do not provide a secure, fault-tolerant and self-managed M2M architecture, which is a sine qua none condition to the healthy and sustained development of the M2M market [3, 2, 22, 21]. If neither security nor reliability is provided for M2M communications, this newly emerging paradigm will not be widely adopted. As M2M communication do not intrinsically require human intervention, they should be self-managed and fault-tolerant. Besides, as machines are generally low cost and unattended equipments, they are exposed to several attacks [11]. Furthermore, M2M communications are expected to be primary used for monitoring and telemetry applications such as in Advanced Metering Infrastructure (AMI) of the smart grids [23]. Fadlullah et al. forecasts in [23] that AMI is the most promising M2M market growth. However M2M communications over AMI must be secure, fault-tolerant and self-managed to reach a healthy and sustained M2M market expansion.

Several researcher have already pointed out this problem. Rongxing et al. noted in [41] that M2M communications reliability and security have not been well investigated. Geng et al. stated in [26] that securing M2M communication will be of paramount concern. He also indicates that "zero-touch" manageability is a serious challenge to an M2M network. Zhang et al. listed in [65] the challenges raised by M2M communications. Among the presented challenges, he stressed on the security and the self-organization issues. Hence, providing a secure, fault-tolerant and self-managed M2M communication is no more an option, it is a crucial necessity. For this purpose, we propose to build a secure and autonomic M2M overlay network over the Internet.

A previous work [17] proposed an M2M overlay network over the Internet based on the Host Identity Protocol (HIP) [43, 44], named HBMON (HIP-based M2M Overlay Network). This previous work have addressed the formation and the maintenance of the overlay. Other works have already focused either on building overlay M2M devices networks or on enabling autonomic properties in M2M/IoT architectures but to the best of our knowledge none of these works uses an autonomic architecture to build and manage an overlay network. Wan et al. consider in [62] that M2M networks are, at the present stage, the main pattern for the Inter-

net of Things (IoT). They review in their work the main features of M2M technologies and stress on their autonomous decision-making and control capabilities. [63] studies the node placement problem in an M2M devices/IoT internet-based overlay network. Ashraf et al. survey in [6] different autonomic-based security approach for the IoT. In [7], the author propose an autonomic framework for device communication in an IoT context. Jara et al. propose in [32] to add a secure mobility management support to the M2M devices in the HIMALIS (Heterogeneity Inclusion and Mobility Adaptation through Locator ID Separation) architecture. It adds the self-protecting (security) and self-healing (mobility) autonomic properties. The proposed architecture targets to extend the Internet of Things through the usage of two distinct levels of indirections: naming and addressing. It also relies on several registries which map the device names to their addresses. Kirsche et al. propose in [37] to use the Extensible Messaging and Presence Protocol (XMPP) in order to simplify the interconnection of M2M devices and actuators.

In this paper, we propose to add the autonomic management of the overlay at the M2M gateway level. We focus on the self-healing and self-optimization autonomic properties. We enable at the M2M gateway level the REAP protocol, a failure detection and locator pair exploration protocol for IPv6 multihoming nodes [5]. Thus, in our overlay, M2M gateways are able to autonomically detect failures of the overlay links and recover from them. Furthermore, M2M gateways are able to monitor the available overlay paths and dynamically select the best path in term of Round Trip Time (RTT). We implement our solution on the OMNeT++ network simulator. Results show that our solution is able to detect overlay link failures and recover from them. It is also able to self-optimize the selection of the overlay paths.

The remaining of this paper is organized as follows. Section 2 highlights the motivation of our work, the challenges that face the deployment of such solution and the fundamental properties to be satisfied. Section 3 presents the key building blocks of our solution. Section 4 gives an overview of HBMON [17, 13, 14]. Section 5 focuses on our contribution; namely the self-healing and optimizing of the HIP-based M2M overlay network. Section 6 presents our simulation results. Section 7 review some related work. Finally, section 8 concludes the paper.

2 Problem Statement

In the following we first detail in section 2.1 the motivation for building an M2M overlay network. Then, we highlight in section 2.2 the constraints that face the deployment of an M2M overlay network. Finally, we present in section 2.3 the fundamental properties of such solution.

2.1 Motivations

M2M technology targets a wide range of applications such as: smart grids, remote maintenance and control, healthcare, security and public safety and vehicular telematics. This large spectrum of uses cases requires a secure, reliable and

fault-tolerant end-to-end communications [26]. For example, for both healthcare (such as monitoring vital signs) and remote maintenance and control applications (such as industrial automation and metering), providing a reliable and fault-tolerant communications is extremely important. Nonetheless, standard Internet protocols do not include efficient failure detection and recovery mechanisms. Labovitz et al. demonstrated in [39] that Border Gateway Protocol (BGP) fault-recovery mechanisms may require several minutes before a consistent convergence of routes. Therefore, we cannot solely rely on current Internet protocols to ensure reliable M2M communications.

An overlay network is a private virtual network built on the top an existing network (which is usually Internet) in order to add a network service not available in the underlying network [4]. At the early stages of its development, Internet (previously named *catenet*[51, 10]) was conceived as concatenation of different scattered networks (ARPANET, MILNET, MINET, SATNET, TELENET). To enable packet switching over these network, Internet predecessor was designed as an overlay over the telephony network. Peer-to-peer (p2p) networks [42], Multicast networks [31] and Content Distributed Networks (CDN) [56] are some examples of current overlay networks.

Overlay networks do not require any change in the current Internet infrastructure, they only add additional servers. Thus, overlay network can be incrementally deployed over the Internet [31]. Moreover, the overlay paradigm breaks the end-to-end principal. Instead of "*keep-it simple in the middle, intelligent at the edge*" [55], overlay networks move the intelligence toward the middle. Indeed, overlay networks rely on middle-boxes (such as overlay router) connected through logical links referred as overlay links. Middle-boxes translates on-demand overlay links into Internet paths.

Consequently, we target to build an M2M overlay network over the existing Internet architecture which will ensure a secure and fault-tolerant M2M communications.

2.2 Constraints

The deployment of an M2M overlay network is subject to several constraints mainly due to the intrinsic nature an M2M network. in the following we detail them.

2.2.1 Resources constraints

M2M devices are expected to be largely deployed in our surrounding environment and this will necessitate very low-cost devices. Consequently, M2M devices will have very limited computation, storage and power capabilities. M2M gateways aggregates the data received from several M2M devices before communicating it to the distant M2M Application. Therefore, M2M gateways will benefit from larger computation, storage and power capabilities than M2M devices. From the above statement, we conclude that we should drastically limit M2M device role in the

management of the M2M overlay network. On the other hand, M2M gateways should have a central role in building and managing the M2M overlay network.

2.2.2 Scalability

In order to build a scalable M2M overlay network, M2M gateway should not be dependent on any specific configuration enabled at the upstream Internet Service Provider (ISP) side. An ISP should not give a specific support to the M2M gateway, M2M gateway incoming packets should handled as any packet coming from an Internet router. Otherwise, this will increase the cost of deploying an M2M solution and small and mid-size businesses would not benefit from M2M technology.

2.2.3 Unreliable communications

M2M devices exclusively use wireless communication technologies to send sensed data to their corresponding M2M gateways. The mass deployment of M2M devices will obviously lead to the increase of the wireless channel noise and fluctuations. And as an immediate consequence, M2M communication first-hop reliability will be degraded. Moreover, in order to maintain an M2M overlay network, M2M gateway should exchange periodic control messages with distant M2M application. Therefore, such communication should be fault-tolerant. Without and adequate support, communication within the M2M overlay are subject to failures which can lead to the loss of connectivity between the overlay members.

2.3 Properties

We target to build an autonomic M2M overlay network. An autonomic M2M overlay network is formed by a federation of heterogeneous machines self-formed without any managing authorities nor specific infrastructures [36]. Such a network should be able to self-manage itself and take its own decision according to information gathered from its environment. For this purpose, our targeted M2M overlay network should provide some fundamental autonomic properties, namely self-properties [35, 25], to efficiently handle M2M communications. Each of them is described below:

2.3.1 Self-configuring

A self-configured M2M network is able to dynamically adapts itself to the deployment of new equipments or changes in its environment. The self-configuring property includes overlay membership management (join and leave operations) [61], overlay resource and service discovery [28] and overlay information retrieval [57].

2.3.2 Self-healing

M2M overlay network members should be able to evaluate their current state and perform corrective actions accordingly. In this context, M2M devices should monitor their available overlay paths to detect possible failures and recover from them [66]. This is the most important property that should be satisfied by our M2M overlay network since we essentially target to build a reliable and fault-tolerant M2M communications.

2.3.3 Self-optimizing

Current M2M gateways are usually multihomed gateways. For example the Cisco 819 4G LTE M2M Gateway natively supports multihoming to ensure a highly available access to the M2M devices [**?**]. Therefore, M2M gateways should proactively monitor the available paths in order to ensure an optimal path selection with no human intervention.

2.3.4 Self-protecting

M2M devices are usually deployed in highly distributed insecure networks. M2M overlay members should protect themselves form physical attacks, compromise of credentials, configuration attacks, protocol attacks and attacks on the core network [11].

3 Key building blocks

In [17], authors proposed an M2M overlay network based on the Host Identity Protocol (HIP). They focused on the self-configuring and the self-protecting autonomic properties. In this work, we target to enhance this solution with the self-healing and self-optimizing autonomic properties. In the following, we first detail the M2M high-level architecture as defined by the ETSI in [22]. Then we detail the Host Identity Protocol (HIP) [43] and the reachability protocol (REAP) [5]. We target to use REAP alongside with HIP at the M2M gateway level to provide self-healing and self-optimizing autonomic properties.

3.1 M2M high-level architecture

The European Telecommunications Standards Institute (ETSI) defines in [22] a high level architecture for M2M. This architecture is divided into two main domains: (i) The device and gateway domain and (ii) the network domain (see Fig. 1).

The device and gateway domain consists of several M2M devices connected to an M2M gateway. An M2M device is typically a sensor or meter that collect data from its surrounding and sends it (in a single-hop or multi-hop) to the M2M gateway. According to the targeted M2M application several radio technologies

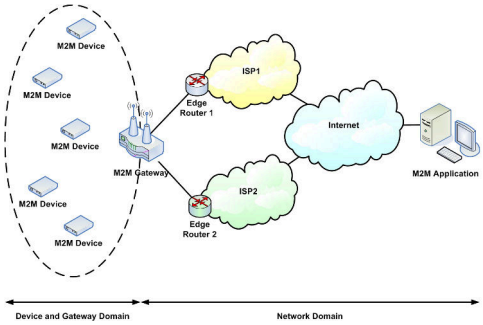

Figure 1: M2M high level architecture

can be used to connect an M2M device to an M2M gateway. For example, for human body monitoring and health-care applications, M2M device are equipped with IEEE 802.15.6 interface which is a low-power and low-data rate wireless technology [1]. For smart metering and home automation applications, M2M device can use a wide spectrum of available radio technology. For high data rate communication, M2M devices can use WiFi or UWB technologies; whereas for low-data rate communication M2M devices can use 6LowPan or Zigbee technologies [65]. The set of the M2M devices constitutes an M2M area network. The M2M gateway connects the M2M area network to the Internet via an access network. In order to improve their reliability, M2M gateways are usually multihomed middelboxes. They are at least connected with two different upstream Internet Service Providers to distant M2M applications. At the M2M applications level, information gathered from the different M2M devices are processed. The network domain includes the access network, the core network (usually Internet) and the distant M2M applications.

3.2 The Host Identity Protocol (HIP)

A well-known problem in the current Internet architecture is the overloading of the semantic of IP addresses: IP addresses have a dual role, they are simultaneously used as endpoint identifier (as seen by the transport layer) and endpoint locator (as seen by the IP layer) [30]. To ensure the scalability of the routing system and to prevent routes disaggregation, the Rekhter Law [53] stipulates that *"'Addressing can follow topology or topology can follow addressing; choose one."'*. This law forces endpoint locators to be topologically correct; whereas, endpoint identifiers are usually not congruent with the Internet topology [15]. Furthermore, with the democratization of the Internet Protocol version 6 (IPv6), endpoints can concurrently have multiple global IPv6 addresses. They are therefore considered as multi-attached or multihomed end-host whether they received their IPv6 prefix from the same ISP (multi-attached) or from different ISP (multihomed). Without an adequate support, such end-host sessions are halted after a failure in the currently used path or an ISP renumbering operation [20, 18]. Finally, we are wit-

nessing a tremendous proliferation of smart-phones and laptop. These nodes are equipped with several wireless access technologies and are able to perform a vertical or a horizontal handover while having running sessions. Without an adequate support, such end-host session are halted after changing their point of attachment to the network [16].

Several protocols have already been proposed to efficiently manage end-host mobility and multihoming such as Shim6 [49], Mobile IP protocol family (MIP, MIPv6, HMIP,...) [33] and the host identity protocol (HIP) [43]. Shim6 is a host-centric multihoming protocols, it enables multihoming features at the network layer. Shim6 provides session survivability upon any change that may affect the currently used IP address. These changes are usually due to failures affecting the used path or ISP renumbering operations. Mobile IP protocol family provide mobility functions at the network layer. They aim at preserving session survivability when the mobile node performs a layer 3 handover, i.e, it changes it attachment to a new access router. HIP provides both mobility and multihoming functions at the host level. Furthermore, in order to be compatible with the current Internet routing system, Shim6, Mobile IP protocol family and HIP use a topologically correct IP address as endpoint locator. These protocols however use different endpoint identifier. While Shim6 and Mobile IP protocol family use one of the available IP address as identifier, HIP introduces a new cryptographic name space called the Host Identity Tag (HIT). HIP also introduces a proxy element in the network architecture, the rendezvous server which holds a secure binding between end-hosts IP addresses and their HITs. Thus, HIP provides a native secure identity to mobile and multihomed nodes.

As highlighted in [12, 27], securing M2M communication is no more an option as M2M networks are inherently insecure networks prone to attacks. Moreover, as M2M devices are usually low-cost devices with limited computation capabilities, a potential mobility and multihoming protocol to be embedded on M2M devices should have a very low computational overhead. Nikander et al., showed in [48] that HIP natively integrates security, mobility and multihoming. Henderson studied in [29] the computational overhead of HIP and demonstrated that HIP can easily be deployed on 266 MHz Pentium II-based laptops. Therefore, HIP is a potential candidate to be deployed on resource-constrained multihomed and mobile M2M nodes.

HIP insert a shim layer in the TCP/IP stack between the IP and transport layer. This new layer rewrites the IP address into a HIT and vice versa. In order to perform this rewriting procedure, HIP establishes a context between any two communicating HIP-enabled hosts. The HIP context holds a binding between the HIT and its corresponding(s) IP address(es). This context is established after a four-way handshake (see Fig.2) between an Initiator and a Responder. The Initiator triggers the establishment of the context by sending a I1 message. The Responder replies with R1 message holding a cryptographic puzzle in order to is to protect the Responder from denial-of-service attacks. Upon receiving the R1 message, the Initiator solves the puzzle and includes the solution in I2 message. After checking the

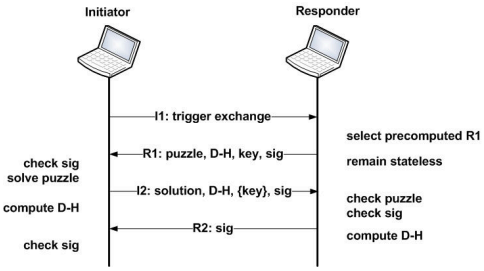

Figure 2: HIP context establishment

solution of the puzzle, the Responder confirms the establishment of the HIP context through an R2 message. Moreover, HIP introduces in the network architecture a registrar element: the Rendez-vous Server (RVS) [40]. This network function help to maintain a binding between IP addresses and their corresponding HIT. HIP nodes can register with an RVS and update their registration if any change has occurred on their available address sets.

3.3 The reachability protocol: REAP

Multihomed terminals are configured with a least two IP addresses each one associated with a distinct Internet Service Provider (ISP). These terminals are then reachable via different paths [15]. A multihomed terminal can spread its outgoing traffic among the available paths by applying a load sharing or balancing scheduling technique. However, such a scheduling technique has a negative impact on TCP. In fact, TCP segments sent on paths with lower delays may result in an out-of-order TCP segments. Upon receiving an out-of-order segment, destination's TCP immediately sends a TCP duplicate acknowledgement. Three duplicates acknowledgements results into the reduction of the TCP congestion window. Therefore, TCP erroneously concludes that these duplicates acknowledgments are due to packet losses and enters in a congestion avoidance phase. Hence, multihomed terminal usually consider one path as primary and the alternate paths as backups. If a failure occurs in the primary path, multihomed terminals migrate their ongoing session to a backup path [19, 20]. For this purpose, the IETF has standardized a protocol for failure detection and locator pair exploration protocol for IPv6 multihoming terminals named the reachability protocol (REAP) [5]. The IETF has designed this protocol for the specific use of the Shim6 protocol.

REAP relies during its functioning on two timers: the send timer and the keepalive timer. REAP assumes that communicating nodes have a prior knowledge of their locators. At the initiator side, REAP starts the send timer whenever a node sends a packets (step 1 in Fig. 3). If this node has not received any packet until the send timer expires, it performs a full reachability exploration procedure. Upon receving the data packet, the responder starts the keepalive timer. If the node has not sent any packet until the keepalive timer expiry, then it sends a REAP keepalive

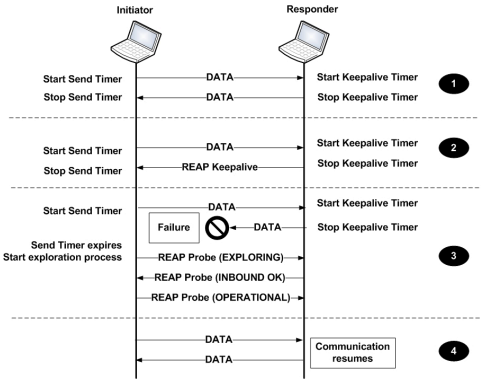

Figure 3: The reachability protocol: REAP

message to its corresponding peer (step 2 in Fig. 3). If the corresponding peer receives a keepalive message, then it stop the send timer and starts the keepalive one. The REAP specification recommends that the keepalive timer should be equal to the send timer divided by three. These two timers are mutually exclusive. In other word, the node is either expecting to receive a payload or preparing to send data. So the send timer is stopped when a payload or keepalive message is received and the keepalive timer is stopped when a payload is generated.

When REAP detects a failure, it starts a full reachability exploration procedure in order to find a new bidirectional working address pair using probe messages (step 3 in Fig. 3). REAP associates a state to each probe indicating the status of the communication. REAP defines three states: OPERATIONAL, INBOUNDOK and EXPLORING. The OPERATIONAL state indicates that communicating peers consider that their ongoing session does not suffer from any failure. The INBOUNDOK state reflect the case where a peer considers that its communication has apparently no problem, but its correspondent one has discovered a failure. The EXPLORING state indicates that a peer has just discovered a failure.

REAP failure recovery procedure is as follows. First, REAP creates a list of all possible pair of addresses by combining the local locator list and the peer locator list and sorts this list according to some priority specified by the user. Then, it switches its state to Exploring and sends four probes successively, a probe every 0.5 second. If it does not receive any probe, it retransmits a probe, but this time the retransmission is controlled by a back-off timer. If a node in the OPERATIONAL state receives a probe having EXPLORING state, it concludes that its correspondent peer has not received its outgoing traffic. This peer then sends a probe having an INBOUNDOK state. A peer in the EXPLORING state and receiving an INBOUNDOK probe conclude that its correspondent peer has received its probe and also that the probed locator pair address is bidirectionally reachable. Thus, it sends a probe having an OPERATIONAL state and the communication can be resumed (step 3 in Fig. 3).

4 HBMON: The HIP-based M2M overlay network

A previous work proposed a HIP-based M2M overlay network called HBMON [17, 13, 14]. In the following, we highlight the salient features of HBMON.

The HBMON is a set of HIP-enabled M2M devices associated with HIP Rendezvous Servers (RVSs) and having running sessions with distant correspondent nodes. The HIP RVS embeds M2M gateway functionalities. It uses a modified verion of the HIP base exchange mechanism in order to define, join and distribute information about our M2M overlay. Theses modifications enable the self-configuring and the self-protecting autonomic properties.

If an M2M device (the initiator) wants to create a new HBMON with a given correspond node (the responder), it sends to this responder a I1 HIP message containing a new field named REQUEST-HBMON. The responder acknowledges the reception of the I1 message by sending an R1 message including a new field named ACK-HBMON. Once it receives a positive acknowledgement, the initiator triggers the discovery of the nearest M2M gateway (HIP RVS). All the M2M gateways have pre-defined IPv6 anycast address. The initiator sends a new HIP signalling message called RVS-Discovery-Request to this specific anycast address. When the initiator receives the R1 packet, it sends an RVS-Discovery-Request packet to a pre-defined anycast address to discover the nearest RVS. A responding M2M gateway answers the initiator with a RVS-Discovery-Response. Similarly, the responder performs the same M2M gateway discovery process. After the discovery of the M2M gateway, the initiator sends a I2 HIP message to the responder including its set of locators, its HIT, the IPv6 address of its M2M gateway and a HBMON_Tag. The responder acknowledges the reception of the I2 message by sending an R2 message including its set of locators, its HIT and the IPv6 address of its M2M gateway. After that, both the initiator and the responder sends a new HIP message named HBMON-CONTEXT to their respective M2M gateways. The HBMON-CONTEXT message includes the following fields (HBMON_Tag, I_HIT, R_HIT, I_IP, R_IP, RVS_IP). HBMON_Tag is an identification of the current context. This context tag -generated by the initiator- should be included in all HBMON packets. I_HIT and R_ HIT are the Host Identity Tag of the initiator and the responder. I_IP and R_ IP are the set of the available IP addresses of the initiator and the responder. RVS_IP is the IP address of the currently used RVS. Finally, The HBMON context is stored by all HBMON members and similarly to the HBMON-CONTEXT message it includes the following records (HBMON_Tag, I_HIT, R_HIT, I_IP, R_IP, RVS_IP). If an M2M device (the initiator) wants to join an existent HBMON, it sends a HBMON-JOIN-Request message with the desired HBMON_Tag to the anycast address of the M2M gateways. If the M2M device is authorized to join this overlay, an M2M gateway replies with a HBMON-JOIN-Response. Fig. 4 presents the HBMON definition procedure.

In the proposed architecture, the M2M gateway acts as an IPv6 router. It periodically broadcasts an ICMPv6 Router Advertisement (RA) message to the M2M devices registered with it. The RA message includes a private IPv6 prefix dedicated

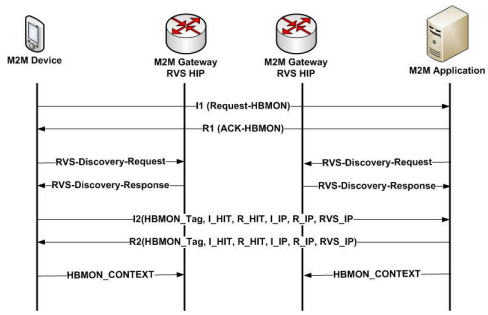

Figure 4: HBMON definition

to the HBMON members. When an M2M device joins a HBMON, it configures a new private IPv6 address upon receiving the RA message It then performs a Duplicated Address Detection procedure [45] to detect if another M2M device member of the overlay has previously configured the same IPv6 address. Finally, the M2M device updates it recording in the M2M gateway with this new address.

From an autonomic networking perspectives, HBMON enabled the self-configuration and self-protection properties. In the HBMON, self-configuration properties is provided by the registration functionality of the HIP protocol which allows M2M devices to autonomically register themselves with a rendezvous server and distribute overlay information between overlay members. The self-protection properties main goal is to give the system the possibility to protect itself from intrusion and any hostile behaviour. The cryptographic namespace HIT [48, 47] with the private addresses used within the overlay are the features used by the M2M devices in the HBMON to protect themselves from attacks.

5 M2M gateway-centric architecture for the autonomic management of the HBMON

The previous work [17] focused on the organization and the membership management of the M2M device within the overlay. We also proposed a novel IPv6 address assigning method in order to configure the overlay members with private IPv6 addresses [13, 14]. This solution already ensures the self-configuring and the self-protecting properties of the autonomic management of our M2M overlay network. In the following we propose to enable at the M2M gateway level the remaining autonomic properties (self-healing and self-optimization).

5.1 Gateway-centric vs. device-centric architectures

The self-healing and self-optimization autonomic properties can be enabled either at the M2M device level or at the M2M gateway level. Enabling these properties at the M2M device level is in concordance with the end-to-end principle, one of the pillars of the current Internet architecture [55]. Nonetheless, M2M devices

have a very limited computational, storage and power capabilities. Thus, handling the self-healing and self-optimizing autonomic properties will be an unacceptable overhead for them. Furthermore, enabling such networking services at the M2M device will increase wireless channel fluctuation and noise as it usually requires extra signalling traffic. On the other hand, M2M gateway have a central role in the M2M network architecture. In [65], Yan et al. proposed a home M2M architecture where all of the networking related functionalities are implemented at the gateway level. In [26], Geng et al. consider M2M gateways as an aggregation and a platform for value-added services. Hence, we propose to enable the self-healing and the self-optimizing autonomic properties at the M2M gateway level. Both autonomic properties are fulfilled through the coupling of HIP with REAP at the M2M gateway level.

5.2 M2M gateway protocol stack

We propose the following protocol stack for the M2M gateway depicted by Fig. 5. The network layer of the M2M gateway includes two sub-layers: a routing sub-layer and an autonomic management sub-layer. The routing layer is a regular IPv6 networking layer; whereas, the autonomic management sub-layer is responsible for the self-healing and the self-optimization capabilities.

To design a resilient M2M overlay network, we use the REAP protocol alongside with HIP at the autonomic management sub-layer to: (i) monitor the existing overlay paths, and (ii) detect failures and recover to a new working path.

At the autonomic management sub-layer, we implement a bi-directional communication between REAP and HIP (see (1) in Fig. 5). In fact , in our M2M overlay network, several overlay paths might exist between the gateway and the corresponding M2M applications. Each path is bounded to a different network interface of the M2M gateway. This path diversity is highly recommended for specific M2M fault-tolerant system such as M2M health-care applications [50]. In order to actively monitor the set of the available overlay paths, REAP retrieves from HIP, for a given HBMON context tag, the set of the available IP addresses. For this purpose, REAP needs to access the information stored by HIP in the HBMON context. Similarly, for a given HBMON context tag, HIP retrieves from REAP the best available overlay path. In fact, the available overlay paths cross different ISPs having different network characteristics (RTT, jitter, errors,...). Moreover, an overlay path can experience for a period of time a degradation of its quality of service (QoS) due to burst traffics and congestion. Meanwhile, this overlay path can be used by an M2M communication requiring a certain QoS level. Without an adequate support, this running M2M session will be affected by the deterioration of this overlay path. Therefore, we enhance the REAP exploring mechanism to offer to a given M2M running communication (identified by a HBMON Context tag) always the best available overlay path.

In order to provide a reliable M2M communication while ensuring session survivability, HIP communicates with the routing sub-layer (see (2) in Fig. 5). Finally,

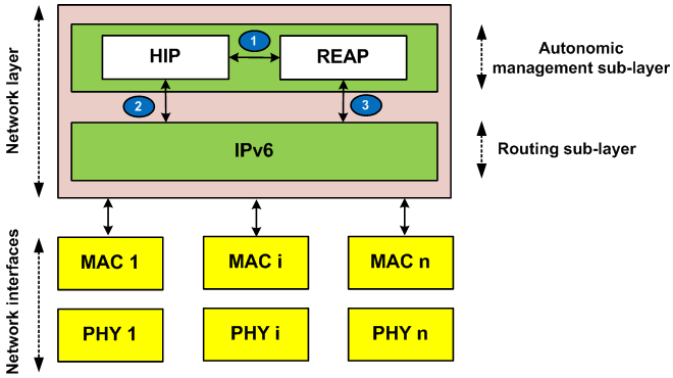

Figure 5: M2M gateway protocol stack

REAP sends its probe and keepalive packets via the routing sub-layer (see (3) in Fig. 5).

5.3 Gateway-centric self-healing of the HBMON

As we previously sated in sections 2.2.1 and 5.1, M2M devices have a very low computational, storage and power capabilities, and consequently HBMON self-healing functions should be deployed at the M2M gateway level. M2M gateways are therefore responsible for the monitoring of the currently used overlay path for a given HBMON context tag. For this purpose, we introduce new parameters in HIP messages namely ”PROBE” and ”KEEP ALIVE” in order to couple REAP with HIP. The ”PROBE” message is exchanged between M2M peer’s gateways when a failure is detected and the ”KEEP ALIVE” message is used to monitor unidirectional communications. We also append HIP with two REAP timers, namely the send and the keepalive timers. If an M2M gateway’s send timer expires without receiving any incoming packets, the M2M gateway assumes that a failure has affected this currently used overlay path and starts exploring the remaining available overlay paths. In unidirectional communications, the M2M gateway has to periodically inform its corresponding gateways that the currently used overlay link is working through the keepalive timer. When the keepalive timer expires, the M2M gateway sends a keepalive message.

If REAP detects a failure through the expiry of the send timer, REAP starts the overlay paths explorations. During this exploration, REAP sends probe exploring messages on each available overlay path bound to a given M2M session. The corresponding M2M gateway receiving the probe exploring message replies with a probe Inbound OK message indicating the status of the probed overlay path. Upon receiving a probe message with the status inbound OK, REAP replies with a probe operational message and switch the ongoing communication to this newly operational overlay link.

In order to explain the functioning of the HBMON self-healing procedure, we

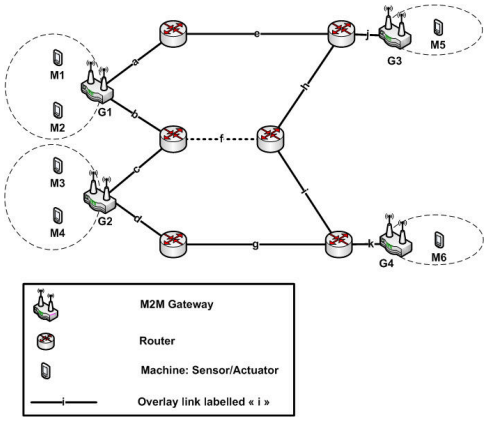

Figure 6: Gateway-centric self-healing of the HBMON

consider the example depcited by Fig. 6 and we assume the following:

- Two HBMON contexts C1 and C2 are established respectively between (M1,M5) and (M3,M6)
- C1 is stored in G1 and G3, while C2 is stored in G2 and G4
- C1 is currently using the overlay path (b,f,h,j) between the two M2M gateways G1 and G3
- C2 is currently using the overlay path (c,f,i,k) between the two M2M gateways G2 and G4
- An outage affects the overlay link (f)

REAP is triggered respectively at the M2M gateways G1 and G2 in order to find a new working overlay path. The M2M gateway G1 sends two REAP probe exploring messages on the overlay paths (a,e,j) and (b,f,h,j). Similarly, the M2M gateway G2 sends two REAP probe exploring messages on (c,f,i,k) and (d,g,k). The M2M gateways G3 and G4 answer with a REAP Inbound OK message on (a,e,j) and (d,g,k). Finally, G1 and G2 respond with a REAP operational message on these two paths and the M2M communications are rehomed to these new working overlay paths.

Therefore, by coupling REAP with HIP a the M2M gateway level, an M2M session can resume after an outage affecting a currently used overlay path. This failure recovery is completely transparent to the established M2M session. We can rely on regular routing protocols to detect such overlay path failure but this may require several minutes before a consistent convergence of routes as demonstrated in [39].

5.4 Gateway-centric self-optimization of the HBMON

Several overlay path migh exisit between two communicating M2M gateway having different RTT. We add to to the M2M gateway the self-optimizing capability by selecting the best available overlay path in term of RTT for a given destination.

Instead of triggering the REAP exploring process at the expiry of the send timer, we modify REAP in order to continuously monitor the available overlay paths and infer their respective RTT. Fro a given M2M session, REAP simultaneously explores all the available overlay paths between two given M2M gateways. For a given overlay path, REAP sends a probe request message and measure the elapsed time between the sending of this probe and the reception of the probe response. If a currently used overlay path experiences a degradation of its RTT, REAP proposes to HIP a new destination/source address pair of an overlay path having a lower RTT.

If we frequently perform the inferring of the RTT and the overlay paths switching, we can cause overlay paths oscillation, known as route flapping [60]. To avoid route flapping, we add a new timer, namely probe timer which defines the time between two consecutive path exploration process. We set up the probe timer to 3 seconds. This value is not supported by any analytical studies. Nonetheless, we believe that setting the probe timer to a lower value would potentially increase congestions if several HBMON contexts need to start path exploration. Moreover, authors in [8] proposed to set the reap send timer to 3 seconds for the same reason. Recall that as we previously explained in 3.3, the expiry of the send timer triggers the full reachability exploration procedure. Consequently, our HIP-based M2M overlay network is self-optimized as it always benefits from the best available overlay path in term of RTT.

6 Self-healing and optimizing signalling cost analysis

In this section, we propose an analytical model to assess the signalling cost of our self-healing and self-optimizing strategies. To do so, we consider theses assumptions:

- An M2M device can simultaneously have several running M2M sessions.
- An M2M session is bound to a HBMON context defining several overlay paths.
- An overlay path is composed of set of overlay links.
- All overlay links fail independently
- The time to fail and the repair time of a link are memoryless, exponentially distributed following a random process with constant means $MTTF_i$ and $MTTR_i$ [46, 54].

Tab.I gives notations that will be used in our signalling cost analysis.

6.1 self-healing signalling cost analysis

6.1.1 Overlay path failure probability

In this section we estimate the probability of an overlay path failure. We define A_i the steady-state availability of an overlay link i as:

$$A_i = \frac{MTTF_i}{MTTF_i + MTTR_i} = \frac{\mu_i}{\mu_i + \lambda_i}$$

The mean failure rate of an overlay link λ_i is measured in units of *Failure in time (FIT)*. 1 FIT is equivalent to 1 failure in 10^9 hours. A the steady-state availability of an overlay path composed of l_{rr} overlay links is defined as:

$$A = \prod_{i=1}^{l_{rr}} A_i$$

Therefore, the probability that an overlay path is in the failed state can be calculated as:

$$U = 1 - A = 1 - \prod_{i=1}^{l_{rr}} A_i = 1 - \prod_{i=1}^{l_{rr}} \frac{\mu_i}{\mu_i + \lambda_i}$$

6.1.2 REAP update exploring cost

For a given M2M session, once REAP detects a failure, it sends a REAP probe exploring message on all the overlay paths bounded to this session (N_{SPath}). Several M2M sessions ($N_{session}$) may share the same overlay path . Thus, the failure will trigger a REAP exploration process on all these overlay paths bound to these sessions. A REAP update exploring cost includes the transmission cost and processing cost at the M2M gateway for all the involved M2M sessions.

$$\Phi_{SH}^{Probe\_Exploring} = N_{session} * N_{SPath} * \frac{\Psi_{SH}^{Probe\_Exploring} + \gamma_r}{T_s}$$

where the transmission cost between two M2M gateways of a REAP exploring message is equal to:

$$\Psi_{SH}^{Probe\_Exploring} = l_{rr} * \delta$$

According to little theorem, T_s can be expressed as $N_{session}/\lambda_{sa}$. Therefore, the signalling cost of the REAP probe exploring message is:

$$\Phi_{SH}^{Probe\_Exploring} = \lambda_{sa} * N_{SPath} * (l_{rr} * \delta + \gamma_r)$$

6.1.3 REAP probe Inbound OK

M2M gateways receiving a REAP probe exploring message will reply will REAP update Inbound OK on all overlay paths bound to a given M2M session. Upon receiving a REAP probe Inbound OK message, the M2M gateway replies with a REAP probe operational message indicating for each M2M session involved in this process the new working overlay path. The REAP probe Inbound OK signalling cost is is equal to:

$$\Phi_{SH}^{Probe\_InboundOK} == N_{session} * (N_{SPath} - 1) * \frac{\Psi_{SH}^{Probe\_InboundOK} + \gamma_r}{T_s}$$

where the transmission cost between two M2M gateways of a REAP probe Inbound OK message is equal to:

$$\Psi_{SH}^{Probe\_InboundOK} = \Psi_{SH}^{Probe\_Exploring} = l_{rr} * \delta$$

The signalling cost of the REAP probe Inbound OK message is:

$$\Phi_{SH}^{Probe\_InboundOK} = \lambda_{sa} * (N_{SPath} - 1) * (l_{rr} * \delta + \gamma_r)$$

6.1.4 REAP probe operational

The REAP probe operational message is sent only on the selected overlay path. Its signalling cost is calucalted as:

$$\Phi_{SH}^{Probe\_Operational} = \lambda_{sa} * (l_{rr} * \delta + \gamma_r)$$

6.1.5 Total signalling cost of HBMON self-healing

The total signalling cost of HBMON self-healing is the sum of all signalling packets cost (see Eq. 1) multiplied by the probability of an overlay path failure U

$$\Phi_{SH}^{Tot} = U * (\Phi_{SH}^{Update\_Exploring} + \Phi_{SH}^{Update\_InboundOK} + \Phi_{SH}^{Update\_Established})$$

$$\Phi_{SH}^{Tot} = 2 * U * N_{SPath} * \lambda_{sa} * (l_{rr} * \delta + \gamma_r) \quad (1)$$

6.2 Self-optimizing signalling cost analysis

6.2.1 REAP probe request/response cost

REAP continuously monitor the available overlay path to estimate their respective RTTs. This estimation is based on the exchange of REAP Probe Request/REAP Probe Response signalling messages. These two messages have the same signalling cost which can be calculated as follows:

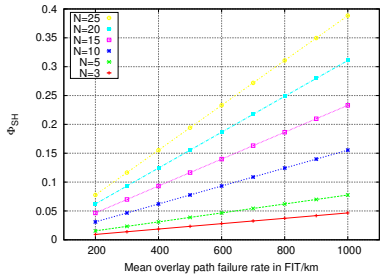

Figure 7: Impact of the number of overlay path on the self-healing signalling cost

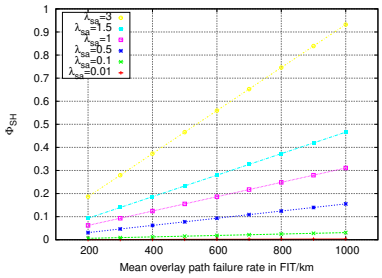

Figure 8: Impact of the session arrival rate on the self-healing signalling cost

$$\Phi_{SO}^{Probe\_Request} = \Phi_{SO}^{Probe\_Response} = N_{session} * N_{SPath} * \frac{\Psi_{SH}^{Probe\_Request} + \gamma_r}{T_s}$$

where the transmission cost between two M2M gateways of a REAP Probe_Request/REAP Probe_Response message is equal to:

$$\Psi_{SH}^{Probe\_Request} = \Psi_{SH}^{Probe\_Response} = l_{rr} * \delta$$

As T_s is equal to $N_{session}/\lambda_{sa}$. Therefore, the signalling cost of REAP Probe_Request/REAP Probe_Response is:

$$\Phi_{SO}^{Probe\_Request} = \Phi_{SO}^{Probe\_Response} \lambda_{sa} * N_{SPath} * (l_{rr} * \delta + \gamma_r)$$

6.3 Results

In the following, we evaluate the total signalling cost of the HBMON self-healing procedure while varying the overlay path failure rate from 200 FIT/km to 1000 FIT/km and the self-optimizing procedure while varying the number of overlay paths and the session arrival rate. For all numerical calculations, we use the same parameter values used in [52] and [64]: $l_{rr} = 35, \lambda_{sa} = 0.01, \delta = 0.2, \gamma_r = 30$

In Fig. 7 we measure the overall signalling cost of the self-healing procedure for different number of overlay path per M2M sessions (3,5,10,15,20,25). We can see that even for a high number of overlay path (25) per M2M session, the signalling cost of the HBMON self-healing procedure still reasonable compared to the HBMON definition signalling cost previously evaluated in [18]. In Fig. 8 we plot the total signalling cost of the HBMON self-healing procedure for different session arrival rates per seconds (λ_{sa}) (0.01,0.1,0.5,1,1.5,3) and for a fixed number of overlay paths per M2M sessions ($N_{SPath} = 10$). Similarly to the results obtained in Fig. 7, even for high M2M session rate arrival per seconds, the self-healing procedure does not introduce any signalling storm on HBMON and has a very low impact on the network load.

In Fig. 7 we measure the signalling cost of the self-optimizing procedure while varying the session arrival rate and the number of overlay paths. The measured signalling cost of our self-optimizing procedure has the same order of magnitude of well-established IETF protocols signalling cost evaluated in [52, 24].

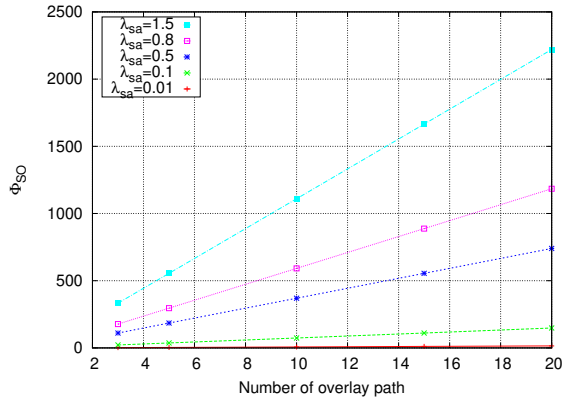

Figure 9: Impact of the number of overlay path on the self-optimizing signalling cost

7 Performance evaluation

In this section we present a performanc evaluation of our M2M gateway-centric architecture for autonomic healing and optimizing of M2M overlay networks. To evaluate our proposal, we use the OMNeT++ simulator coupled with the HIPSim++ framework[9]. We implement the protocol stack for the M2M gateway depicted by Fig. 5 in the HIPSim++ framework.

7.1 Simulation set up and evaluation metrics

We set up an M2M device connected to Internet via a mutlihomed M2M gateway. The M2M gateway has four available overlay paths having the following RTTs: 50ms, 100ms, 150ms and 200ms. The correspondent node is an M2M application. We set all the wireless accesses to 802.11b at 11 Mbit/s. Between the M2M application and the M2M device we use two types of traffic: the first one is an UDP flow having the following characteristics: 20 Bytes the packet length and 40 ms the inter-packet interval, the second traffic is TCP flows, namely an FTP application with hight data rate traffic. A failure affecting the currently used overlay path occurs after 20 s from the beginning of the communication and lasts twice as the send timer.

In order to evaluate our solution, we first focus on the application recovery time metric (ART). The ART was defined by La Oliva et al. in [38]. It measure the latency between the last packet received/sent before the outage and the first packet

received/sent after the outage. The ART highlight the self-healing capabilities of our solution. After that, we evaluate the impact of our architecture on the instant throughput of an M2M session.

7.2 Results

7.2.1 Self-healing evaluation

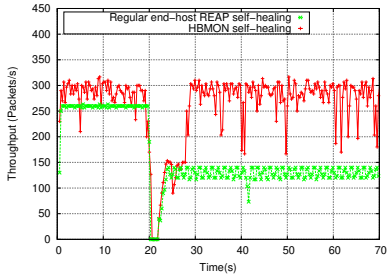

Figure 10: TCP self-healing with regular REAP and HBMON

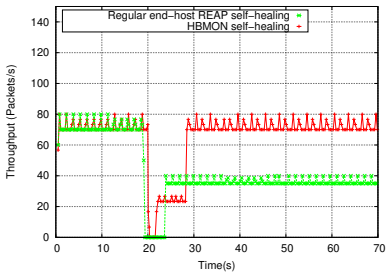

Figure 11: UDP self-healing with regular REAP and HBMON

First of all, we compare our solution with the regular REAP self-healing capabilities. Recall that regular REAP is deployed at the M2M device side while in our case REAP is deployed at the M2M gateway side. For this purpouse, we measure the instant throughput for both TCP and UDP traffic, depicted in Fig.10 and Fig.11 respectively. After 20s from the start of the simulation, the currently used overlay path (having the lowest RTT 50ms) experience a failure. The obtained result show that our self-healing strategy detects and recover from the failure more rapidly than the case of regular REAP. In fact, as the wireless segment is shared between 7 M2M devices and the REAP probe exploring message is simultaneously sent by all these M2M devices. This contention induces a delay in the failure recovery process. In addition, in the regular REAP case, once a new working overlay path is found, the current M2M sessions are simply rehomed to this path. In our case, the M2M gateway continuously monitors all the available overlay paths, and therefore, as soon as the previous path recovers from its failure, HBMON rehome ongoing M2M sessions to it as it has the lowest RTT.

We evaluate in the following the self-healing capabilities of our solution. We measure the ART of both UDP traffic and the TCP traffic while varying the REAP send timer. Results are presented by Fig. 12, the x-axis is the send timer value while the y-axis is the measured ART. Results show that for an UDP application, the ART time increases linearly while we increase the send timer value; whereas, for TCP application the ART experiences several plateaus. After failure recovery, UDP application immediately sends data packets to the newly selected overlay path. Even if a new overlay path is selected, TCP does not send immediately its data segments, it has to wait until the TCP Retransmission Timeout (RTO) timer

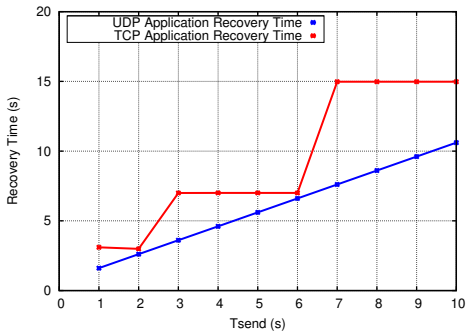

Figure 12: Application recovery time for TCP and UDP traffic

expiry. It adjusts the RTO timer as if it has experience of a congestion phase which explains the plateaus in Fig. 12. We conclude form the obtained results that our solution effectively detects failures and the established M2M session resumes after failure recovery for both TCP and UDP traffic. The TCP recovery lasts longer than the UDP one as TCP does not distinguish between a failure recovery process and the congestion in the currently used path [16]. To resolve this, we couple the RTO with REAP by a cross-layer design. This optimisation has been proposed by La Oliva et al. in [38]. When REAP finishes the exploration process and detects a new working overlay path, it cancels the RTO so that TCP can immediately retransmits the buffered TCP segments.

7.2.2 Self-optimization evaluation

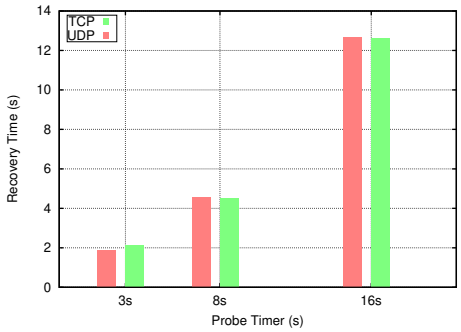

Figure 13: Probe Timer impact on the recovery of TCP and UDP traffic

We evaluate in this section the self-optimization capability of our solution. We modify REAP to actively monitor the available paths in order to offer the ongoing M2M communication the best available overlay path in term of RTT.

We focus on the following scenario: the currently used overlay path has an RTT of 50ms and a transient failure affects this path after 20s of the beginning of the M2M communication, the failure lasts the double of the probe timer.

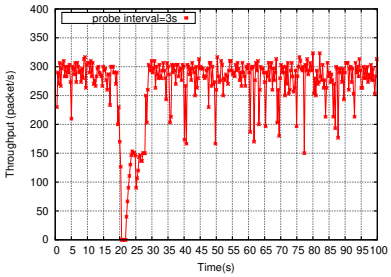

Figure 14: TCP recovery time

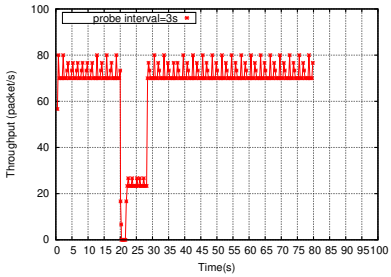

Figure 15: UDP recovery time

We firstly evaluate the impact of the probe timer on the recovery of both UDP and TCP traffic. Fig. 13 illustrates the TCP/UDP recovery time for different values of the probe timer. We see that both TCP and UDP traffics have almost the same recovery time value whenever we vary the probe timer. In addition, the recovery time increases whenever we increase the probe timer.

Before the occurrence of the failure (20 s), the M2M gateway has already inferred the RTT of each overlay path (50 ms, 100 ms, 150 ms, 200 ms). These measurements are done respectively each 3s, 8s and 16s (the different proposed values of the probe timer). As soon as the failure has been detected by REAP, the M2M gateway has to wait until the inferring the RTT of the available overlay paths ends. For a probe timer equal to 3 s, the last RTT measurement has be performed 18 s after the beginning of the simulation. After the failure, the RTT inferring will be ready after 21 s of the beginning of the simulation. Thus, the recovery can be performed only after 21 s. It is in fact performed at 21.88 s for the case of the UDP traffic and 22.12 s for TCP one, which represents a recovery time equal respectively to 1.88 s and 2,12s. Similarly for a probe timer equal to 8s (16 s respectively), the recovery can be performed only after 24 s (32 s respectively). Thus, whenever we increase the probe timer, the recovery time increases regardless of the nature of the traffic (TCP or UDP).

We fix in the following the probe timer to 3 s and evaluate the instant throughput of both a TCP and UDP traffics. Fig. 14 shows the obtained results for a TCP session. The x-axis is the time in second and the y-axis is the instant throughput. The obtained results show that during the first 20 s, the throughput reaches its maximum because the used path has the minimum RTT (50 ms). After the failure recovery, REAP detects a new working overlay path having the second best RTT (100 ms). As soon as the best overlay path (50 ms) recovers forms its failure, M2M communication switches to this new path and the throughput reaches again its maximum value. Fig. 15 shows the obtained results for a running UDP session and a probe timer set to 3 s. The obtained results show the same behaviour as for the TCP case in Fig. 14. After the outage, the UDP session is rehomed to a new working overlay path (100 ms). As soon as the new overlay path (50ms) become ready, the UDP session is rehomed to this newly available path, and the throughput reaches again its maximum value.

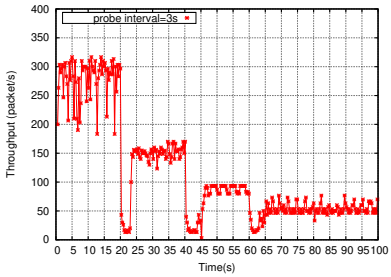

Figure 16: TCP dynamic path selection

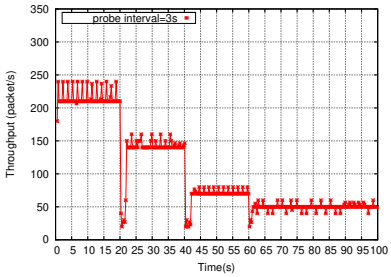

Figure 17: UDP dynamic path selection

In a last scenario, we explore the self-optimization capability of our solution by modifying the load of the currently used overlay path. The M2M communication starts in the overlay path having the lowest RTT. A congestion appears in this path, so the TCP ongoing connection experiences packet losses, TCP reduces its congestion window which impact the instant throughput of the M2M communication. Our solution detects the quality degradation of the path and switches the communication to the second best path in term of RTT. Results presented by Fig. 16 and Fig. 17 shows this dynamic selection of the most stable path. During the first 20 s, the M2M communication flows via the path having the lowest RTT (50 ms). We inject in this path an aggressive UDP traffic, creating therefore a congestioned path. Our solution detects the degradation of the RTT of this path and its fluctuations. It switches the ongoing communication to the second path. We repeat the same scenario on this second path. Our solution switches one more time the communication to a third path and finally to the last one until it finds a stable path in term of RTT and packet loss.

From Fig. 14, Fig. 15, Fig. 16 and Fig. 17 we clearly see that we build a self-optimized solution. It is able to detect failure in the currently used overlay path, select a new working path and monitor the remaining paths.

8 Conclusion

The M2M technology is currently under standardization at both ETSI and 3GPP and actively supported by the telecommunication industry. M2M technology is considered to be the “killer service” which will fill the revenue gap caused by the constant decrease of the voice service. In order to have secure and private M2M communications, a previous work defined an M2M overlay network based on the Host Identity Protocol (HIP) named HBMON. From an autonomic networking perspective, our M2M overlay network is already self-configured and self-protected. In this work we added the self-healing and the self-optimizing autonomic capabilities to our M2M overlay network. To do so, we coupled HIP with the Reachability protocol (REAP) at the M2M gateway level. We implemented and evaluated our

proposal on the OMNeT++ network simulation using both TCP and UDP traffic. We have demonstrated that our solution effectively integrates the self-healing and the self-optimized capabilities. We have highlighted that TCP timers impact the self-healing capabilities of our solution as TCP does not distinguish between a failure affecting a currently used overlay path and a congestion episode. We therefore recommend to couple the RTO TCP timers with REAP ones in a cross layer way to accelerate the M2M session recovery process.

9 Acknowledgement

The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group No. RGP-1435-090

References

[1] Ieee standard for local and metropolitan area networks - part 15.6: Wireless body area networks. *IEEE Std 802.15.6-2012*, pages 1–271, 2012.

[2] 3GPP TR 22.888 V12.0.0. Study on enhancements for Machine-Type Communications (MTC), March 2012.

[3] 3GPP TS 22.368 V12.2.0. Service Requirements for Machine-Type Communications, March 2013.

[4] David Andersen, Hari Balakrishnan, Frans Kaashoek, and Robert Morris. Resilient overlay networks. *SIGCOMM Comput. Commun. Rev.*, 32(1):66–66, January 2002.

[5] J. Arkko and I. van Beijnum. Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming. RFC 5534 (Proposed Standard), June 2009.

[6] Qazi Mamoon Ashraf and Mohamed Hadi Habaebi. Autonomic schemes for threat mitigation in internet of things. *Journal of Network and Computer Applications*, 49(0):112 – 127, 2015.

[7] Q.M. Ashraf, M.H. Habaebi, G.R. Sinniah, M.M. Ahmed, S. Khan, and S. Hameed. Autonomic protocol and architecture for devices in internet of things. In *Innovative Smart Grid Technologies - Asia (ISGT Asia), 2014 IEEE*, pages 737–742, May 2014.

[8] Sébastien Barré and Olivier Bonaventure. Improved path exploration in shim6-based multihoming. In *SIGCOMM 2007 Workshop "IPv6 and the Future of the Internet"*, Kyoto, Japan, Aug 2007.

[9] László Bokor, Szabolcs Nováczki, László Tamás Zeke, and Gábor Jeney. Design and evaluation of host identity protocol (hip) simulation framework for inet/omnet++. In *Proceedings of the 12th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '09, pages 124–133, New York, NY, USA, 2009. ACM.

[10] Vint Cerf. The catenet model for internetworking. IEN, July 1978.

[11] Inhyok Cha, Y. Shah, A.U. Schmidt, A. Leicher, and M.V. Meyerstein. Trust in m2m communication. *Vehicular Technology Magazine, IEEE*, 4(3):69–75, 2009.

[12] Yunjeong Choi, Inshil Doh, Seung-Soo Park, and Ki-Joon Chae. Security based semantic context awareness system for m2m ubiquitous healthcare service. In Youn-Hee Han, Doo-Soon Park, Weijia Jia, and Sang-Soo Yeo, editors, *Ubiquitous Information Technologies and Applications*, volume 214 of *Lecture Notes in Electrical Engineering*, pages 187–196. Springer Netherlands, 2013.

[13] A. Dhraief, M.A. Ghorbali, T. Bouali, and A. Belghith. Mobility management in the hip-based m2m overlay network. In *Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2013 IEEE 22nd International Workshop on*, pages 50–55, June 2013.

[14] A. Dhraief, M.A. Ghorbali, T. Bouali, A. Belghith, and K. Drira. Simultaneous mobility management in the hip-based m2m overlay network. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International*, pages 219–224, July 2013.

[15] Amine Dhraief and Abdelfettah Belghith. Multihoming support in the internet: A state of the art. In *International Conference on Models of Information and Communication Systems (MICS 2010)*, 2010.

[16] Amine Dhraief and Abdelfettah Belghith. An experimental investigation of the impact of mobile ipv6 handover on transport protocols. *The Smart Computing Revue, KAIS*, 2(1), 2012.

[17] Amine Dhraief, Mohamed Amine Ghorbali, Tarek Bouali, Abdelfettah Belghith, and Khalil Drira. HBMON: A HIP-Based M2M Overlay Network. In *The 2012 Third International Conference on the Network of the Future (NoF 2012)*, 2012.

[18] Amine Dhraief, Issam. Mabrouki, and Abdelfettah Belghith. A service-oriented framework for mobility and multihoming support. In *Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean*, pages 489 –493, march 2012.

[19] Amine Dhraief and Nicolas Montavont. Rehoming decision algorithm: Design and empirical evaluation. In *Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 02*, CSE '09, pages 464–469, Washington, DC, USA, 2009. IEEE Computer Society.

[20] Amine Dhraief, Tanguy Ropitault, and Nicolas Montavont. Mobility and multihoming management and strategies. In IFIP, editor, *14th Eunice Open European Summer School 2008*. RSM Dept (Institut TELECOM ;TELECOM Bretagne), 2008.

[21] ETSI TS 102 689 V2.1.1. Machine-to-Machine communications (M2M); M2M service requirements, July 2013.

[22] ETSI TS 102 690 V1.2.1. Machine-to-Machine communications (M2M); Functional architecture, June 2013.

[23] Z.M. Fadlullah, M.M. Fouda, N. Kato, A. Takeuchi, N. Iwasaki, and Y. Nozaki. Toward intelligent machine-to-machine communications in smart grid. *Communications Magazine, IEEE*, 49(4):60–65, 2011.

[24] Shaojian Fu, Mohammed Atiquzzaman, Liran Ma, and Yong-Jin Lee. Signaling cost and performance of sigma: A seamless handover scheme for data networks. *Wireless Communications and Mobile Computing*, 5(7):825–845, 2005.

[25] A. G. Ganek and T. A. Corbi. The dawning of the autonomic computing era. *IBM Systems Journal*, 42(1):5 –18, 2003.

[26] Geng Wu and Talwar, S. and Johnsson, K. and Himayat, N. and Johnson, K.D. M2M: From mobile to embedded internet. *Communications Magazine, IEEE*, 49(4):36 –43, april 2011.

[27] Jorge Granjal, Edmundo Monteiro, and JorgeS Silva. Security issues and approaches on wireless m2m systems. In Shafiullah Khan and Al-Sakib Khan Pathan, editors, *Wireless Networks and Security*, Signals and Communication Technology, pages 133–164. Springer Berlin Heidelberg, 2013.

[28] Qiang He, Jun Yan, Yun Yang, R. Kowalczyk, and Hai Jin. A decentralized service discovery approach on peer-to-peer networks. *Services Computing, IEEE Transactions on*, 6(1):64–75, 2013.

[29] T. R. Henderson, J. M. Ahrenholz, and J. H. Kim. Experience with the host identity protocol for secure host mobility and multihoming. *2003. WCNC 2003. 2003 IEEE Wireless Communications and Networking*, 3:2120–2125, March 2003.

[30] R. Jain. Internet 3.0: Ten problems with current internet architecture and solutions for the next generation. In *Military Communications Conference, 2006. MILCOM 2006. IEEE*, pages 1–9, 2006.

[31] John Jannotti, David K. Gifford, Kirk L. Johnson, M. Frans Kaashoek, and James W. O'Toole, Jr. Overcast: reliable multicasting with on overlay network. In *Proceedings of the 4th conference on Symposium on Operating System Design & Implementation - Volume 4*, OSDI'00, pages 14–14, Berkeley, CA, USA, 2000. USENIX Association.

[32] Antonio J. Jara, Ved P. Kafle, and Antonio F. Skarmeta. Secure and scalable mobility management scheme for the internet of things integration in the future internet architecture. *Int. J. Ad Hoc Ubiquitous Comput.*, 13(3/4):228–242, July 2013.

[33] D. Johnson, C. Perkins, and J. Arkko. Mobility Support in IPv6. RFC 3775 (Proposed Standard), June 2004. Obsoleted by RFC 6275.

[34] Juniper Networks. Machine-to-Machine (M2M) The Rise of the Machines. white paper, 2011.

[35] J.O. Kephart. Research challenges of autonomic computing. In *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, pages 15 – 22, may 2005.

[36] J.O. Kephart and D.M. Chess. The vision of autonomic computing. *Computer*, 36(1):41 – 50, jan 2003.

[37] M. Kirsche and R. Klauck. Unify to bridge gaps: Bringing xmpp into the internet of things. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 455–458, March 2012.

[38] Antonio La Oliva, Marcelo Bagnulo, Alberto García-Martínez, and Ignacio Soto. Performance analysis of the reachability protocol for ipv6 multihoming. In *Proceedings of the 7th international conference on Next Generation Teletraffic and Wired/Wireless Advanced Networking*, NEW2AN '07, pages 443–454, Berlin, Heidelberg, 2007. Springer-Verlag.

[39] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed internet routing convergence. *SIGCOMM Comput. Commun. Rev.*, 30(4):175–187, August 2000.

[40] J. Laganier and L. Eggert. Host Identity Protocol (HIP) Rendezvous Extension. RFC 5204 (Experimental), April 2008.

[41] Rongxing Lu, Xu Li, Xiaohui Liang, Xuemin Shen, and Xiaodong Lin. Grs: The green, reliability, and security of emerging machine to machine communications. *Communications Magazine, IEEE*, 49(4):28–35, 2011.

[42] Eng Keong Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim. A survey and comparison of peer-to-peer overlay network schemes. *Communications Surveys Tutorials, IEEE*, 7(2):72–93, 2005.

[43] R. Moskowitz and P. Nikander. Host Identity Protocol (HIP) Architecture. RFC 4423 (Informational), May 2006.

[44] R. Moskowitz, P. Nikander, P. Jokela, and T. Henderson. Host Identity Protocol. RFC 5201 (Experimental), April 2008. Updated by RFC 6253.

[45] T. Narten, E. Nordmark, W. Simpson, and H. Soliman. Neighbor Discovery for IP version 6 (IPv6). RFC 4861 (Draft Standard), September 2007. Updated by RFC 5942.

[46] Wenda Ni, Jing Wu, Changcheng Huang, and Michel Savoie. Analytical models of flow availability in two-layer networks with dedicated path protection. *Optical Switching and Networking*, 10(1):62 – 76, 2013. Advances in Optical Networks Control and Management.

[47] P. Nikander, A. Gurtov, and T. R. Henderson. Host identity protocol (hip): Connectivity, mobility, multi-homing, security, and privacy over ipv4 and ipv6 networks. *Commun. Surveys Tuts.*, 12(2):186–204, April 2010.

[48] Pekka Nikander, Jukka Ylitalo, and Jorma Wall. Integrating security, mobility, and multi-homing in a hip way. In Internet Society, editor, *Proc. of Network and Distributed Systems Security Symposium (NDSS'03)*, pages 87–98, Feburuary 2003.

[49] E. Nordmark and M. Bagnulo. Shim6: Level 3 Multihoming Shim Protocol for IPv6. RFC 5533 (Proposed Standard), June 2009.

[50] RoyC. Park, Hoill Jung, Dong-Kun Shin, Gui-Jung Kim, and Kun-Ho Yoon. M2m-based smart health service for human ui/ux using motion recognition. *Cluster Computing*, pages 1–12, 2014.

[51] Louis Pouzin. A proposal for interconnecting packet switching networks. In Brunel University, editor, *Proceedings of EUROCOMP*, pages 1023–1036, May 1974.

[52] Abu S Reaz, Pulak K Chowdhury, Mohammed Atiquzzaman, and William Ivancic. Signalling cost analysis of sinemo: seamless end-to-end network mobility. In *Proceedings of first ACM/IEEE international workshop on Mobility in the evolving internet architecture*, MobiArch '06, pages 37–42, New York, NY, USA, 2006. ACM.

[53] Y. Rekhter and T. Li. An Architecture for IP Address Allocation with CIDR. RFC 1518 (Historic), September 1993.

[54] S. Sahhaf, W. Tavernier, D. Colle, M. Pickavet, and P. Demeester. Availability analysis of resilient geometric routing on internet topology. In *Design of Reliable Communication Networks (DRCN), 2014 10th International Conference on the*, pages 1–8, April 2014.

[55] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Trans. Comput. Syst.*, 2(4):277–288, November 1984.

[56] Stefan Saroiu, Krishna P. Gummadi, Richard J. Dunn, Steven D. Gribble, and Henry M. Levy. An analysis of internet content delivery systems. *SIGOPS Oper. Syst. Rev.*, 36(SI):315–327, December 2002.

[57] Chunqiang Tang, Zhichen Xu, and Sandhya Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, pages 175–186, New York, NY, USA, 2003. ACM.

[58] Technical Specification Group Services and System Aspects. Study on facilitating machine to machine communication in 3GPP systems. 3GPP TR, March 2007.

[59] Mikko A. Uusitalo. Global vision for the future wireless world from the wwrf. *Vehicular Technology Magazine, IEEE*, 1(2):4 –8, 2006.

[60] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1):1 – 16, 2000.

[61] Spyros Voulgaris, Daniela Gavidia, and Maarten Steen. Cyclon: Inexpensive membership management for unstructured p2p overlays. *Journal of Network and Systems Management*, 13(2):197–217, 2005.

[62] Jiafu Wan, Hehua Yan, Qiang Liu, Keliang Zhou, Rongshuang Lu, and Di Li. Enabling cyber-physical systems with machine-to-machine technologies. *Int. J. Ad Hoc Ubiquitous Comput.*, 13(3/4):187–196, July 2013.

[63] Yuxin Wan, Junwei Cao, Kang He, Huaying Zhang, Peng Yu, Senjing Yao, and Keqin Li. Node placement analysis for overlay networks in iot applications. *IJDSN*, 2014, 2014.

[64] Jiang Xie and I.F. Akyildiz. An optimal location management scheme for minimizing signaling cost in mobile ip. In *Communications, 2002. ICC 2002. IEEE International Conference on*, volume 5, pages 3313–3317 vol.5, 2002.

[65] Yan Zhang, Rong Yu, Shengli Xie, Wenqing Yao, Yang Xiao, and M. Guizani. Home m2m networks: Architectures, standards, and qos improvement. *Communications Magazine, IEEE*, 49(4):44–52, 2011.

[66] Yan Zhang, Rong Yu, Shengli Xie, Wenqing Yao, Yang Xiao, and M. Guizani. Home M2M networks: Architectures, standards, and QoS improvement. *Communications Magazine, IEEE*, 49(4):44 –52, april 2011.

Parameter	Definition
$N_{session}$	Avg. number of sessions
λ_{sa}	Avg. session arrival rate per second
T_s	Avg. session duration $T_s = N_{session}/\lambda_{sa}$(little theorem)
N_{SPath}	Avg. number of paths per sessions
δ	Per-hop message transmission cost over wired link
γ_r	Processing cost in a M2M gateway
l_{rr}	Avg. number of wired link between two M2M gateways
λ_i	The mean failure rate for an overlay link i
μ_i	The mean repair rate for an overlay link i
$MTTF_i$	The mean time to fail of an overlay link i. $MTTF_i = 1/\lambda_i$
$MTTR_i$	The mean time to repair of an overlay link i. $MTTR_i = 1/\mu_i$
A_i	The steady-state availability of an overlay link i
A	The steady-state availability of an overlay path
U	The unavailability of an overlay path
Self-healing signalling cost	
Φ_{SH}^{Tot}	Self-healing signalling cost in unit time
$\Phi_{SH}^{Probe\_Exploring}$	Probe (exploring) signalling cost in unit time
$\Psi_{SH}^{Probe\_Exploring}$	Probe (exploring) transmission cost
$\Phi_{SH}^{Probe\_InboundOK}$	Probe (Inbound OK) signalling cost in unit time
$\Psi_{SH}^{Probe\_InboundOK}$	Probe (Inbound OK) transmission cost
$\Phi_{SH}^{Probe\_Operational}$	Probe (Operational) signalling cost in unit time
$\Psi_{SH}^{Probe\_Operational}$	Probe (Operational) transmission cost
Self-optimizing signalling cost	
Φ_{SO}^{Tot}	Self-optimizing signalling cost in unit time
$\Phi_{SH}^{Probe\_Request}$	Probe request signalling cost in unit time
$\Psi_{SH}^{Probe\_Request}$	Probe request transmission cost
$\Phi_{SH}^{Probe\_Response}$	Probe response signalling cost in unit time
$\Psi_{SH}^{Probe\_Response}$	Probe response transmission cost

Table 1: Notations