|  |
| --- |
| C:\Users\grayv\Desktop\oneM2M-Logo.gif |

|  |  |
| --- | --- |
| **oneM2M**  **Technical Report** | |
| Document Number | oneM2M-TR-0059-V-0.1.0 |
| Document Name: | oneM2M Services and Platforms Discovery |
| Date: | 2019-01-30 |
| Abstract: | The document is describing what services and platforms discovery scenarios are considered beneficial from a oneM2M standpoint and how these can be supported by oneM2M system. Based on the result of the technical report, it will identify possible advanced features and enhancements which the next oneM2M release(s) could support. |
| Template Version: January 2017 (Do not modify) | |

The present document is provided for future development work within oneM2M only. The Partners accept no liability for any use of this report.

The present document has not been subject to any approval process by the oneM2M Partners Type 1. Published oneM2M specifications and reports for implementation should be obtained via the oneM2M Partners' Publications Offices.

About oneM2M

The purpose and goal of oneM2M is to develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software, and relied upon to connect the myriad of devices in the field with M2M application servers worldwide.

More information about oneM2M may be found at: http//www.oneM2M.org

Copyright Notification

© 2017, oneM2M Partners Type 1 (ARIB, ATIS, CCSA, ETSI, TIA, TSDSI, TTA, TTC).

All rights reserved.

The copyright and the foregoing restriction extend to reproduction in all media.

Notice of Disclaimer & Limitation of Liability

The information provided in this document is directed solely to professionals who have the appropriate degree of experience to understand and interpret its contents in accordance with generally accepted engineering or other professional standards and applicable regulations. No recommendation as to products or vendors is made or should be implied.

NO REPRESENTATION OR WARRANTY IS MADE THAT THE INFORMATION IS TECHNICALLY ACCURATE OR SUFFICIENT OR CONFORMS TO ANY STATUTE, GOVERNMENTAL RULE OR REGULATION, AND FURTHER, NO REPRESENTATION OR WARRANTY IS MADE OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS. NO oneM2M PARTNER TYPE 1 SHALL BE LIABLE, BEYOND THE AMOUNT OF ANY SUM RECEIVED IN PAYMENT BY THAT PARTNER FOR THIS DOCUMENT, WITH RESPECT TO ANY CLAIM, AND IN NO EVENT SHALL oneM2M BE LIABLE FOR LOST PROFITS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. oneM2M EXPRESSLY ADVISES ANY AND ALL USE OF OR RELIANCE UPON THIS INFORMATION PROVIDED IN THIS DOCUMENT IS AT THE RISK OF THE USER.

Contents

1 Scope 5

2 References 5

2.1 Normative references 5

2.2 Informative references 5

3 Definitions, symbols and abbreviations 5

3.1 Definitions 5

3.2 Symbols 6

3.3 Abbreviations 6

4 Conventions 6

5 Introduction 6

6 Background and Motivation 6

6.1 State of the Art of Dynamic Discovery Mechanisms 6

6.1.1 Zeroconf 6

6.1.1.1 Overview 6

6.1.1.2 Technologies in ZeroConf Protocol 7

6.1.1.3 Services 7

6.1.1.4 Implementations 8

6.1.2 DNS 8

6.1.2.1 Overview 8

6.1.2.2 Structure 8

6.1.2.3 Mechanisms 9

6.1.2.4 Applications 9

6.1.3 UPnP 9

6.1.3.1 Overview 9

6.1.3.2 Mechanisms 9

6.1.3.3 Features 10

6.2 Challenges and Key Issues 10

6.2.1 Key Issues on oneM2M platforms discovery 11

6.2.2 Key Issues on oneM2M services discovery 11

7 Required Functions on oneM2M Architectural Framework 11

7.1 Introduction 11

7.2 Discovery of IoT Platforms 11

7.3 Discovery of IoT Services 11

7.4 Provisioning of IoT Services in Visited oneM2M Systems 11

8 Proposed Solutions 11

8.1 Solution A 11

8.1.1 Overview 11

8.1.2 Solution Description 12

8.2 Solution X 12

9 Conclusions 12

I. Annexes 12

III. History 13

# Scope

The document is describing what services and platforms discovery scenarios are considered beneficial from a oneM2M standpoint and how these can be supported by oneM2M system. Based on the result of the technical report, it will identify possible advanced features and enhancements which the next oneM2M release(s) could support.

# References

The following text block applies.

References are either specific (identified by date of publication and/or edition number or version number) or non- specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

## Normative references

Normative references are not applicable in the present document.

## Informative references

Clause 2.2 shall only contain informative references which are cited in the document itself.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] oneM2M Drafting Rules.

NOTE: Available at <http://www.onem2m.org/images/files/oneM2M-Drafting-Rules.pdf>.

# Definitions, symbols and abbreviations

Delete from the above heading the word(s) which is/are not applicable.

## Definitions

Clause numbering depends on applicability.

* **A definition shall not take the form of, or contain, a requirement.**
* **The form of a definition shall be such that it can replace the term in context. Additional information shall be given only in the form of examples or notes (see below).**
* **The terms and definitions shall be presented in alphabetical order.**

For the purposes of the present document, the [following] terms and definitions [given in ... and the following] apply:

Definition format

**<defined term>:** <definition>

If a definition is taken from an external source, use the format below where [N] identifies the external document which must be listed in Section 2 References.

**<defined term>**[N]: <definition>

**example 1:** text used to clarify abstract rules by applying them literally

NOTE: This may contain additional information.

## Symbols

Clause numbering depends on applicability.

For the purposes of the present document, the [following] symbols [given in ... and the following] apply:

Symbol format

<symbol> <Explanation>

<2nd symbol> <2nd Explanation>

<3rd symbol> <3rd Explanation>

## Abbreviations

For the purposes of the present document, the [following] abbreviations [given in ... and the following] apply:

Abbreviation format

<ABREVIATION1> <Explanation>

<ABREVIATION2> <Explanation>

<ABREVIATION3> <Explanation>

# Conventions

The key words "Shall", "Shall not", "May", "Need not", "Should", "Should not" in the present document are to be interpreted as described in the oneM2M Drafting Rules [i.1].

# Introduction

*Editor’s Note:*

# Background and Motivation

*Editor’s Note: The section summarizes existing technologies of service and platforms discovery that can be used in oneM2M. It also includes architectural gap analysis including key issues.*

## State of the Art of Dynamic Discovery Mechanisms

*Editor’s Note: The section provides an overview of the various discovery mechanisms, incl, Zeroconf, DNS, UPnP.*

### Zeroconf

#### Overview

Zero Configuration Networking makes it possible to connect two or more number of computers and connects them with crossover Ethernet cable for communicate usefully using IP without in the absence of configured information from either a user or infrastructure services such as conventional DHCP. It can be scaled up for communication at larger network. It was initiated to achieve the ease-of-use of AppleTalk, applied to IP, the ubiquitous platform-agnostic communications protocol which coexist gracefully with larger configured networks. Zero Configuration Networking is required for environments where administration is impractical or impossible, such as in the home or small office, embedded systems ‘plugged together’ as in an automobile, or to allow impromptu networks such as ‘‘between the devices of strangers on a train’’. Zeroconf is not a good option: medium or large networks, networks that need a high degree of security and control, large public access networks, and networks that have low bandwidth and high latency (such as some wireless networks). The long-term goal of Zero Configuration Networking is to enable the creation of entirely new kinds of networked products, products that today would simply not be commercially viable because of the inconvenience and support costs involved in setting up, configuring, and maintaining a network to allow them to operate. It is not necessary to simultaneously use Zeroconf protocols in all four areas (i.e. IP interface configuration, translation between host name and IP address, IP multicast address allocation, service discovery). For example, it may make sense on some networks to provide a DHCP server for configured IP interface configuration but perform translation between host name and IP address using a ZeroConf protocol. In deployment of ZeroConf on existing configuration network, the design consideration should be taken such as minimal impact on existing networks and application and no less secure than related current IETF-Standard protocols.

The ZeroConf network aims to meet following requirements:

* The benefits of ZeroConf protocols over existing configured protocols are an increase in the ease-of-use for end-users and a simplification of the infrastructure necessary to operate protocols
* ZeroConf is not limited to single IP subnet. If the Protocol is intended to span multiple IP subnets it MUST NOT use broadcasts or link-local addressing.
* ZeroConf protocols must adapt to changing network conditions. Zeroconf protocols must be able to resolve conflicts and return the network to a consistent state after changes in network topology or other events.

#### Technologies in ZeroConf Protocol

Zeroconf uses a number of underlying technologies. For address selection, link-local addressing replaces the DHCP server. This capability is built into Ipv6 addresses. However, traditional Ipv4 is used as a last resort except in a ZeroConf context. The multicast name service is used for name resolution in Zeroconf in order to allow a network device to select a domain name in the local name space and then announce it using a designated multicast IP address. Zeroconf uses multicast DNS for service discovery. In this model, every computer on the local network stores an individual listing of DNS resource records and then joins the respective DNS multicast group supporting the Zeroconf network. Finally, DNS-based service discovery is one of the largest aspects of the Zeroconf implementation. It is very implementation specific, but relies on a type of messaging in order to discover services and provide notifications of the available ones on a network.

Zeroconf is based on three technologies:

* Assigning network addresses for various devices
* Determining computer host names
* Locating network services

Service discovery protocols automatically detect connected network services and devices. These protocols include:

* Service Location Protocol
* Universal Description Discovery and Integration for Web services
* Bluetooth Service Discovery Protocol
* extensible Resource Descriptor Sequence

#### Services

**Obtaining an IP Address**

For a device to route messages on a network, it needs an IP address. This is relatively trivial if a DHCP server is available, but when this isn’t the case, ZeroConf uses link-local addressing to obtain one. The address assigned by link-local addressing is in the 169.254.0.0/16 block, which is only useful within the link (because routers won’t forward packets from devices in this address space). To obtain an address the device sends ARP requests to establish whether its desired IP address (chosen at random from the 169.254 range) is available.

This is a two-step process:

1. First, an ARP probe is sent asking for the MAC address of the machine with a given IP. This probe is sent a number of times to confirm that the given IP address is not in use (there will be no response if it isn’t).
2. If no reply is received, the device sends out an ARP announcement saying that it is now the machine with the given IP.

**Obtaining a Hostname**

Once a device has an IP address it can be contacted through the network, but IP addresses are volatile and for the device to be consistently discoverable it needs a hostname (which is less likely to change). Assigning as hostname is simple if you have a DNS server, but, if you don’t, zero-conf can assign hostnames using Multicast DNS. To claim a local hostname with mDNS, a device sends DNS messages to probe for the uniqueness of the hostname. Three queries are sent in 250ms intervals, and if no device reports using this hostname, the requesting device then sends an announce message to claim ownership. In the case of a conflict (where two devices believe they own the same hostname), lexicographic ordering is used to determine a winner. DNS hostnames must use a local top-level domain (.com, .org, .gov, etc.) to distinguish them from globally accessible hosts. Apple devices and many others use the local domain.

**Browsing for Services**

Once a device has an IP address and hostname, it can be contacted, but only if we know the name of the specific device we are looking for. If we don’t, DNS Service Discovery (also known as DNS-SD) can be used to search for services available on the local link.

#### Implementations

The few implementations of Zeroconf are following:

**Apple’s Bonjour:**

An implementation of Zeroconf’s service discovery protocol used in several operating systems such as Mac OS X and Microsoft Windows. It uses multicast DNS service records to locate devices such as printers, computers and many other services and utilized to discover services on a LAN and to allow users to set up a network without any configurations. Bonjour only works within a single broadcast domain, which is usually a small area if the DNS server is not appropriately configured. Bonjour does not have the capability to advertise services to the public Internet.

**Avahi:**

It facilitates service discovery on local networks. Avahi allows a new host on the local network to view other hosts and communicate with them (chat, find printers or shared files). The Avahi mDNS responder is actually completely implemented and has passed all tests in the Apple Bonjour conformance test suite. Moreover, it supports some unique features such as the correct mDNS reflection across LAN segments. It is implemented as a C library (‘‘avahi-core’’) which can be embedded into other applications.

**Windows CE 5.0:**

It configures the wireless network adapter to connect to an available wireless network. If there are two networks covering the same area, it is possible to configure a preferred network order and the device will try to contact each network in the order defined until it locates one that is active. It is also possible to limit association to only the configured, preferred networks.

**Other Implementations:**

Other link IPv4 implementations include ZCIP, CUPs, J-share and mono.zeroconf

### DNS

#### Overview

The Domain Name System (DNS) refers to hierarchical naming system for resources linked to the Internet. The main job of DNS is to provide translation of domain names to the IP addresses for identifying and locating a computing resource. DNS servers eradicate the need for humans to remember the numeric IP addresses of a computer or device on the Internet by providing a distributed directory service. The DNS provides a fault tolerant service as domain names are assigned by authoritative name servers for each domain.

#### Structure

The DNS comprises of a tree structure where each nodes contains a label and zero or more resource records. These resource records keep domain name related information. The domain name is composed of the label which is concatenated with the name of its parent and separated by a dot.

The DNS tree is divided into zones at the root. A zone may consists of one or more domains and sub-domains. A zone may be divided into additional sub-zones for better administration. In this case the authority delegates to a designated server for this new zone.

The domain name consists of labels that are separated by dots. For instance test.com. Right most label is called the top-level-domain. The hierarchy works from right to left as each label on the left indicates a sub-domain of the domain to the one on the right. Continuing the example, the label *test* describes the sub-domain of the domain *com*. Figure 6.1.2.2-1 shows the DNS for Internet.

A close up of a logo

Description automatically generated

Figure 6.1.2.2-1: Hierarchical Domain Name System for Internet

#### Mechanisms

The DNS comprises of a tree structure where each nodes contains a label and zero or more resource records. These resource records keep domain name related information. The domain name is composed of the label which is concatenated

The DNS is worked by various name servers. A name server is a node in a distributed database which allows DNS to resolve queries. The top of hierarchy is served by root name server. Each domain has at least one authoritative name server that provides information about the domain and the name servers of sub-domains.

A query is resolved by DNS resolvers that are responsible for querying domain names by sequence of queries initiating with the right most domain label. For proper working of DNS resolver, a host is configured with cache of the known addresses of the root servers. If there is no cache, the process begins by querying to one of root servers. Root servers in typical environment do not process the query themselves but instead forward it to authoritative name servers. For instance, a query to “www.test.com” is referred to *com* servers. The resolver will query to the referred servers and iteratively repeats until it finds an authoritative server. Figure 6.1.2.3-1 shows the iterative process of resolving the domain “www.test.com”.

Caching in address resolution helps keeping a balanced traffic burden on root servers. In practical, root name servers are involved in the request process for relatively small time.

A close up of a logo

Description automatically generated

Figure 6.1.2.3-1: DNS Resolver resolving the domain name in iterative way

#### Applications

There are several other applications of DNS beside translation of domain names to IP addresses. It is not necessary that domain name and IP address are in one-to-one relationship. A single domain name can resolve to multiple IP addresses to provide load balancing and fault tolerance to servers. Conversely, multiple domain names may resolve to a single IP address to provide virtual hosting.

Mail transfer agents use DNS to find the most suitable mail server to deliver email. The DNS is also used for storage of blocked email hosts. This is achieved by placing the IP address of subject host to the sub-domain of higher level domain name. That name is resolved to a record for positive or negative indication.

For example:

A blocked address 192.168.1.1 points to 1.1.168.192.block.example, which resolve to 127.0.0.1.

### UPnP

#### Overview

UPnP (Universal Plug & Play) technology allows devices to connect seamlessly and to simplify network implementation in the home and corporate environments. The UPnP architecture offers pervasive peer-to-peer network connectivity of PCs of all form factors, intelligent appliances, and wireless devices. UPnP consists of a set of related protocols built on open Internet standards, such as TCP/IP, HTTP, XML. UPnP devices are "plug and play" in that, when connected to a network, they automatically establish working configurations with other devices.

#### Mechanisms

**Step 0. Addressing**

Most internet routers have integrated UPnP technology. Whenever a new device supporting UPnP connects to a network it announces its information (protocol, port forwarding information) to the router. Router assigns IP address to the device and enables discovery.

**Step 1: Discovery**

When a device (ex: sensor) is added to the network, the UPnP discovery protocol allows that device to advertise its services to control points on the network. Similarly, when a control point (ex: pc, smartphone) is added to the network, the UPnP discovery protocol allows that control point to search for devices of interest on the network. Discovery architecture depicted below:

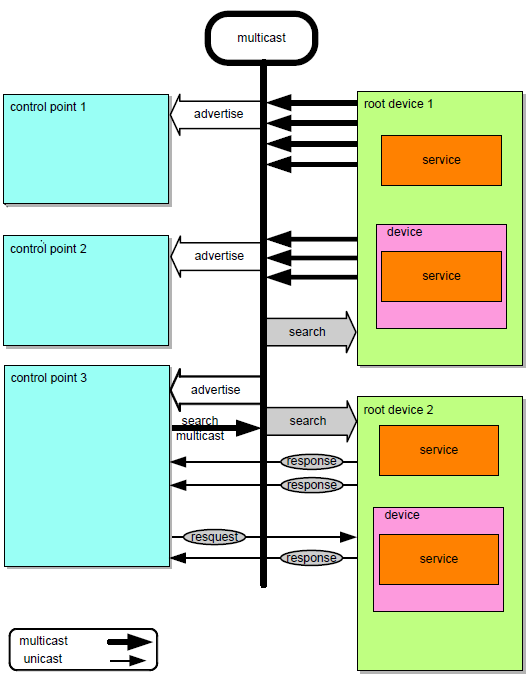


Figure 6.1.3.2-1: UPnP high-level procedures

After obtaining IP address, a device broadcasts messages advertising itself, its embedded devices, and its services.

When a new control point is added to the network, it is allowed to multicast a discovery message searching for interesting devices, services, or both. If any device matches search criteria, they will respond. In addition, a control point is allowed to unicast a discovery message to a specific IP address. This action presumes the control point already knows the device at this IP address is a UPnP device.

**Step 2: Description**

After a control point has discovered a device, the control point still knows very little about the device -- only the information that was in the discovery message. For the control point to learn more about the device and its capabilities, or to interact with the device, the control point shall retrieve a description of the device and its capabilities from the URL provided by the device in the discovery message. A UPnP device description includes vendor -specific

manufacturer information like the model name and number, serial number, manufacturer name, URLs to vendor-specific Web sites, etc. (details below). For each service included in the device, the device description lists the service type, service name, a URL for a service description, a URL for control, and a URL for eventing.

**Step 3: Control**

Given knowledge of a device and its services, a control point can ask those services to invoke actions and receive responses indicating the result of the action. Invoking actions is a kind of remote procedure call; a control point sends the action to the device's service, and when the action has completed (or failed), the service returns any results or errors.

**Step 4: Eventing**

A control point subscribes to notifications from a device.

**Step 5: Presentation**

Events are displayed on UI of a control point.

#### Features

UPnP supports the following features:

* Media and device independence. UPnP technology can run on any network technology including Wi-Fi, coax, phone line, power line, Ethernet and 1394
* Platform independence. Vendors can use any operating system and any programming language to build UPnP products
* Internet-based technologies. UPnP technology is built upon IP, TCP, UDP, HTTP, and XML, among others.
* UI Control. UPnP architecture enables vendor control over device user interface and interaction using the browser.
* Programmatic control. UPnP architecture enables conventional application programmatic control.
* Common base protocols. Vendors agree on base protocol sets on a per-device basis.
* Extendable. Each UPnP product can have value-added services layered on top of the basic device architecture by the individual manufacturers.

## Challenges and Key Issues

*Editor’s Note: The section introduces challenges and key issues.*

### Key Issues on oneM2M platforms discovery

### Key Issues on oneM2M services discovery

# Required Functions on oneM2M Architectural Framework

*Editor’s Note: The section provides an analysis of the oneM2M architectural framework needed for supporting oneM2M Platforms and Services discovery.*

## Introduction

*Editor’s Note: This section provides a general introduction on why oneM2M needs mechanisms for platforms and service discovery.*

## Discovery of IoT Platforms

*Editor’s Note: This section describes a general concept of oneM2M Platforms discovery.*

## Discovery of IoT Services

*Editor’s Note: This section describes a general concept of oneM2M services discovery.*

## Provisioning of IoT Services in Visited oneM2M Systems

*Editor’s Note: This section describes a general concept of provisioning of IoT services in visited oneM2M systems.*

*.*

# Proposed Solutions

*Editor’s Note: The section provides solutions to the required functions identified in the previous section.*

## Solution A

*Editor’s Note: Each Solution section references one or more Key Issues that it addresses and provides a brief solution description.*

### Overview

*Editor’s Note: This section provides a general description of the proposed solution. The solution should identify which issues it is addressing.*

### Solution Description

*Editor’s Note: This section provides a concise description of the solution which provides enough detail for further stage 2 development.*

## Solution X

# Conclusions

*Editor’s Note: This section provides a summary of the conclusions drawn during the study.*

# Annexes

Each annex **shall** start on a new page (insert a page break between annexes A and B, annexes B and C, etc.).

Use the **Heading 9** style for the title and the Normal style for the text.

Annex <A>:  
Title of annex *(style H9)*

<Text>

<PAGE BREAK>

# History

|  |  |  |
| --- | --- | --- |
| **Publication history** | | |
| V.1.1.1 | <dd Mmm yyyy> | <Milestone> |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

| **Draft history** (to be removed on publication) | | |
| --- | --- | --- |
| V0.0.1 | 2018-10-31 | Skeleton of the TR. |
| V0.1.0 | 2019-01-30 | Agreed contribution added |