An evaluation of service discovery protocols in the internet of things

Conference Paper · April 2017

DOI: 10.1145/3019612.3019698

CITATIONS
17

READS
539

3 authors:

Christian Cabrera
Trinity College Dublin
19 PUBLICATIONS **108** CITATIONS
SEE PROFILE

Andrei Palade
Trinity College Dublin
22 PUBLICATIONS **658** CITATIONS
SEE PROFILE

Siobhán Clarke
Trinity College Dublin
182 PUBLICATIONS **3,413** CITATIONS
SEE PROFILE

An Evaluation of Service Discovery Protocols in the Internet of Things

Christian Cabrera, Andrei Palade, and Siobhán Clarke
Distributed Systems Group, School of Computer Science and Statistics
Trinity College Dublin
[cabrerac, paladea, Siobhan.Clarke]@scss.tcd.ie

ABSTRACT

The IoT environment surfaces challenging requirements for service discovery, such as: services heterogeneity, mobility, scalability, security, QoS support and context management. Different protocols have been proposed to facilitate service discovery, but it is difficult to assess how well these protocols meet the IoT requirements. This paper presents an evaluation of commonly used service discovery protocols for the IoT, CoAP-SD, DNS-SD, mDNS-SD, and DDS-SD, performed against both qualitative and quantitative metrics, on a physical experimental setup. The results show the limitations and strengths of the protocols, and future research directions are discussed.

Keywords

Internet of Things, Service Oriented Computing, Service Discovery, IoT Protocols, Evaluation, CoAP-SD, DNS-SD, mDNS-SD, DDS-SD.

1. INTRODUCTION

The Internet of Things offers services to a large number of applications in numerous domains. Different protocols that offer service discovery capabilities have been proposed as standards for the IoT [4], and include support for service description, registration, discovery and resolution. However, to operate effecitively in an IoT environment, they must also address non-functional requirements such as heterogeneity, scalability, mobility, security, QoS support, and context management [7, 26, 29]. Recent surveys have summarised these IoT protocols' features [4, 16], but no study has assessed how well the protocols meet IoT characteristics. This paper presents a qualitative and quantitative evaluation of commonly used service discovery protocols for the IoT: DNS-SD [11], mDNS-SD [12], CoAP-SD [32], and DDS Service Discovery [1].

The evaluation provides a detailed perspective of how the protocols meet both functional and non-functional requirements. We propose a set of qualitative and quantitative metrics which capture the essential requirements of IoT. These metrics are independent of the selected protocols, and can be used to evaluate any service discovery protocol from an IoT point of view. The experimental setup in this evaluation consists of a physical deployment of heterogeneous IoT devices and a dataset of around 60000 atomic descriptions related to weather services. This data was extracted from the LinkedSensorData[1] dataset which contains expressive descriptions of around 20000 weather stations in the United States. This paper outlines the limitations and strengths of the evaluated protocols and discusses future research directions based on the results.

The rest of the paper is organised as follows. Section 2 summarises related work. Section 3 presents the IoT requirements for service discovery and the metrics used in this evaluation. Section 4 introduces the selected protocols, while Section 5 details the data and infrastructure used in the experiments. Section 6 presents the results of the evaluation and Section 7 concludes the paper with the evaluation results and a discussion about future research directions.

2. RELATED WORK

Previous works on service discovery for IoT survey current service discovery protocols along with their capabilities. Al Fuqaha et al. [4] presents a list of IoT common standards including efforts led by different organizations. This list describes the components, features, functionalities and implementations of protocols which offer service discovery capabilities such as CoAP, DNS-SD, mDNS-SD and DDS. Datta et al. [16] presents the features of CoAP and DNS-SD, proposes an architecture for resource discovery and highlights the future standardisation aspects for interoperable discovery among different IoT platforms. This study does not present an implementation and evaluation of the architecture or any discussion about how they work at the resource level. Cirani et al. [13] proposes a scalable architecture for service discovery in IoT based on CoAP and DNS-SD. This work includes a description of how each protocol is implemented, introduces a set of performance metrics and presents an evaluation of the proposed architecture. How-

[1]LinkedSensorData - http://wiki.knoesis.org/index.php/LinkedSensorData

ever, metrics to evaluate IoT features such as heterogeneity, mobility, QoS support and context management are not included.

A number of studies evaluate each service discovery protocol individually, or in scenarios which are different or included in the IoT. Gligoric et al. [19] and Potsch el al. [27] evaluate the performance of CoAP for M2M communication in WSNs, Colitti et al.[14] compares CoAP and HTTP in terms of performance and energy consumption in WSNs, and Al-Mejibli et al. [5] compares transmission time of traditional protocols such as Jini, UPnP, Bonjour, and SLP. Villaverde et al.[36] evaluates the network overhead introduced by CoAP and DNS-SD for M2M communications in a resource-constrained environment. They also include a description of how the protocols work and meet interoperability, scalability, reliability and human interaction requirements in M2M communication environments. Rahman et al.[28] investigated how current IoT protocols address security issues, and included an analysis of CoAP highlighting its features and open research challenges.

3. SERVICE DISCOVERY REQUIREMENTS AND EVALUATION METRICS

The requirements and challenges behind the realisation of the IoT vision has been well documented (e.g., scalability, interoperability, mobility, availability, performance, reliability, security and privacy [4, 29, 8]). Service discovery protocols must consider these requirements in order to support development of service-based applications in an IoT environment. Table 1 outlines the requirements considered in this work. The rest of this section introduces the metrics used for evaluating the service discovery protocols presented in this paper.

Table 1: Service discovery requirements.

FEATURE	REQUIREMENT
Heterogeneity	Service discovery protocols should offer an interoperable solution to manage different services regardless of their providers, description formats or technologies.
Mobility	Service discovery protocols should handle dynamic scenarios providing components which guarantee effective and efficient registration, unregistration of mobile services.
Scalability	The registration, unregistration and discovery functionalities should offer a good response time regardless of the number of services in the environment.
Security	Service discovery protocols should prevent, detect and act against fraudulent service providers and customers.
QoS Support	Service discovery protocols should offer the ability to express quality parameters in the service description as well as use this information in the discovery process.
Context Management	Service discovery protocols should use context information to perform the discovery in order to offer relevant services to the user requests.

Table 2 defines a set of metrics based on the service discovery requirements outlined in Table 1. This set of qualitative metrics will be used to assess how each protocol interoperates with other protocols, what security mechanisms are provided, and how QoS parameters and context information are expressed and used in service discovery.

We also define metrics to quantify the performance of the protocols in practical scenarios. Heterogeneity metrics are used to determine the accuracy of service searching with heterogeneous service descriptions (i.e., precision and recall).

Table 2: Evaluation metrics.

QUALITATIVE METRICS	
FEATURE	**METRIC**
Heterogeneity	How is the interoperability with other service discovery protocols?
Security	Does the protocol offer security mechanisms?
QosSupport	Does the protocol allows QoS parameters in services description?
	Does the protocol use QoS parameters in service searching?
Context Management	Does the protocol allows context information in services description?
	Does the protocol use context information in service searching?
QUANTITATIVE METRICS	
FEATURE	**METRIC**
Heterogeneity	Precision in searching with syntactic and semantic differences in services descriptions (P): $P = \frac{truePos}{truePos + falsePos}$
	Recall in searching with syntactic and semantic differences in services descriptions (R): $R = \frac{truePos}{truePos + falseNeg}$
Mobility	Registration response time (ms) with different number of services in the environment.
	Unregistration response time (ms) with different number of services in the environment.
Scalability	Service discovery response time (ms) with different number of services in the environment for centralised protocols.
	Advertisement response time (ms) in proactive discovery with different number of services in the environment and different number of nodes in the network for distributed protocols.
	Unadvertisement response time (ms) in proactive discovery with different number of services in the environment and different number of nodes in the network for distributed protocols.

The metrics for mobility measure how long a service takes to be available or unavailable for discovery when it appears or leaves from the network. That is the response time for service registration, and unregistration with different number of services in the environment. Scalability metrics are used to determine the response time of reactive and proactive service discovery in centralised and distributed approaches respectively.

4. SELECTED PROTOCOLS

The selected protocols have been proposed by different organizations such as Object Management Group (OMG) or Internet Engineering Task Force (IETF). For instance, DNS-based Service Discovery (DNS-SD) [11], Multicast DNS Service Discovery (mDNS-SD) [12], Constrained Application Protocol (CoAP-SD) [32] and Data Distribution Service protocol which has a discovery module (DDS-SD) [1], provide service description, registration, unregistration, discovery and resolution capabilities. These protocols are considered standards for the IoT [4], are commonly used in current research and industry works, and are supported by well known organizations. The rest of this section introduces each protocol.

4.1 CoAP-SD

CoAP [32] is a centralised protocol for IoT applications created by the IETF Constrained RESTful Environments (CoRE)[2] working group. It defines a web transfer protocol based on REST where applications' endpoints can interact using a request/response model through traditional HTTP methods. This protocol supports discovery of services in a client/server model, CoAP-SD. The client can send registration, unregistration and discovery requests which are consumed by the CoAP server. The service descriptions contain an URI, a service type and, optionally, a list of client-defined key-value

[2]CoRE Group - https://datatracker.ietf.org/wg/core/charter/

attributes. The client uses the URI from the service description to access the service.

CoAP has been used in IoT environments because it provides REST support through a lightweight UDP protocol [9, 10, 20, 22, 25]. Multiple flavours of CoAP implementations are available on the CoAP website[3] with support for different types of platforms (e.g., constrained devices, sever side, smart phones).

4.2 DNS-SD

DNS-SD is a centralised service discovery protocol defined by IETF [11] which structures the DNS resource records to facilitate service discovery. The service providers can register services in a DNS-server through a registration request which includes the service description. This description contains the service name, type, domain and a TXT record where more information can be added in a key-value format. This protocol follows a request-reply approach, where the service consumers query a DNS-SD server using the service name or type. The server replies with service information (e.g., service endpoint) that matches the query.

Previous IoT solutions used DNS-SD for service discovery [21, 23, 33, 34, 35] because of its portability across different platforms. Avahi[4] is a well-known implementation of DNS-SD.

4.3 mDNS-SD

mDNS-SD [12] is a distributed protocol defined by IETF which requires minimal configuration, works on infrastructure-less environments and is resilient to network failure. This protocol uses the same service description structure as DNS-SD. It proactively discovers, advertises and unadvertises new services that become available in the environment. The resolution can be done after the discovery phase like in DNS-SD.

Previous studies used mDNS-SD in resource-constrained environments, where minimal configuration of devices is required[17, 18, 24, 34, 35]. Avahi also implements mDNS-SD.

4.4 DDS-SD

DDS [2] is a M2M standard developed by the Object Management Group (OMG)[5] with a number of commercials and open-source implementations such as RTI[6], and OpenDDS[7].

This protocol introduces a Data-Centric Publish Subscribe (DCPS) technique for real-time systems, and it uses a participant entity to create publishers, subscribers and topics. The publisher disseminates information about a topic through a Data Writer, and the subscriber receives the data about a topic through a Data Reader. DDS participants use the Participant Discovery Protocol to discover each other in the network, and the Endpoint Discovery Protocol to exchange information about their endpoints [1].

This protocol is used in current research as well as in industry and military applications[6, 15, 31, 37] because it offers real time features ensuring end-to-end quality of service with a rich set of QoS policies.

[3]CoAP - http://coap.technology/impls.html
[4]Avahi - http://avahi.org
[5]OMG - http://www.omg.org/
[6]RTI DDS - https://www.rti.com/products/dds/
[7]OpenDDS - http://opendds.org/

5. EXPERIMENTAL SETUP

This section introduces the experimental setup employed to evaluate each protocol. It describes the data used in the experiments, and the infrastructure used during evaluation, where the protocols were installed and how each experiment was performed.

5.1 Data Test Definition

The test data for this evaluation is a set of 60000 weather service descriptions. This data was generated using Linked-SensorData dataset which contains expressive descriptions of 20000 weather stations in the United States. Each record has a station identifier, a station location (represented by latitude, longitude and altitude), and a list of services that the station provides. Extra metadata, such as QoS parameters, types and units for service outputs and inputs, is added to each record. Table 3 describes the structure of a service description.

Table 3: Service descriptions structure.

FIELD	DESCRIPTION
ID	Service identification. It is formed by the station ID and the service type separated by underline. For example: 3CLO3_WinDirection
URI	Service URI. It is formed by the prefix http://surf and the service ID. For example: http://surf/3CLO3_WinDirection
Service Type	The sensor type which is extracted from the weather station description. For example WindDirection.
Description	Information about the service goal. For example: This service monitors WindDirection
Response Time	The time to get a response from the service. It is generated following a normal distribution with mean 500 and standar deviation 100. For example: 463 ms
Reliability	The service reliability. It is generated following a normal distribution with mean 0.8 and standar deviation 0.075. For example: 0.668
Latitude	The latitude of the service location. It is extracted from the station description. For example: -124.00
Longitude	The longitude of the service location. It is extracted from the station description. For example: 46.22
Altitude	The longitude of the service location. It is extracted from the station description. For example: 20
Unit	The unit for the service output. For example: degrees
Input	The service output type. For example: string
Output	The service input type. For example: double

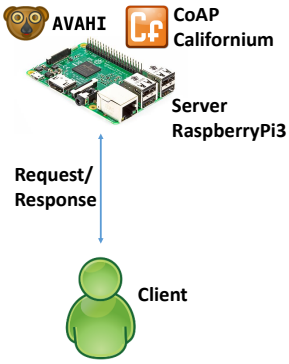
We define three data sets to represent heterogeneous descriptions using the 60000 descriptions generated. The first set is homogeneous (i.e., two services that are equivalent have the same syntactic description) and is correct (i.e., two services that provide the same reading have the same type). The second set has syntactic differences between their services. For example, service A and B provide air temperature readings, but the type of A is *AirTemperature* and the type of B is *AirTemp*. The third set has syntactic and semantic differences. For example, service A and B provide temperature readings, but the type of A is *AirTemperature* and the type of B is *TemperaturaDelAire* (Spanish form) as well as the unit of A is $^{\circ}C$ and the unit of B is $^{\circ}F$. The second and third set also have some mismatch data to simulate human errors. For example, service C provides wind direction readings, but its type is *AirTemperature*.

5.2 Experimental Architecture

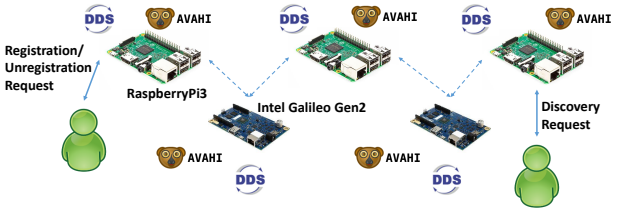
We define two architectures for our experiments: one for centralised protocols (i.e., DNS-SD and CoAP-SD), and other for the distributed approaches (i.e., mDNS, DDS-SD).

Figure 1(a) presents the architecture for centralised protocols. This architecture consists of a server which responds to client requests. The protocol is installed in a gateway (i.e., Raspberry Pi3 running Raspbian), and the requests are performed from a Java program that uses the existing implementations of DNS-SD[8] and CoAP-SD [9]. The client selects the services to register from the data previously defined. We change the number of services in the directory to evaluate scalability. We run experiments with 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800 and 2000 services. Each experiment goes through 100 iterations. Figure 1(b) shows the architecture for distributed approaches. This architecture comprises a set of nodes which are connected in a WiFi MANET. The nodes can be Raspberry Pi3 running Raspbian or Intel Galileo boards running Ubilinux. Each node implements the evaluated protocol and a Java program performs the advertisements or queries over the network using the protocol implementations for mDNS[10] and DDS[11]. We change the number of services in the directory to evaluate scalability and increase the number of nodes in the network to evaluate the response time with different number of services and hops. We run the experiments with 5, 10, 15 and 20 nodes in the network. Each experiment is repeated 100 times.

We apply the precision and recall metrics using the datasets previously defined. The first time the client selects services from the homogeneous dataset, registers the selected services and performs the discovery. This process is repeated 100 times. Afterwards, the client selects a subset of services from the syntactic heterogeneous dataset, registers this subset of services and the selected services from the first data set and performs the discovery. This process is repeated 100 times. Finally, the client selects a subset of services from the semantic heterogeneous dataset, registers this subset and the selected services in the steps before, and performs the discovery. This process is repeated 100 times.



(a) Centralised architecture



(b) Distributed architecture

Figure 1: Experimental setup architectures.

6. EVALUATION RESULTS

Table 4 shows the results of the qualitative evaluation. For heterogeneity, CoAP-SD has the best interoperability with other protocols (i.e., DNS-SD, mDNS-SD and HTTP) which allows the discovery and use of IoT and traditional services (e.g., web services). DNS-SD and mDNS-SD can cooperate to achieve automated service discovery using mDNS-SD to announce new services and DNS-SD to attend user queries. DDS-SD does not interoperate with other protocols.

CoAP-SD is the only protocol that offers built-in security mechanisms through the Datagram Transport Layer Security [30]. DTLS is a protocol for UDP which provides privacy for datagrams, and it is designed to prevent eavesdropping, tampering, and message forgery. DDS-SD does not provide security features but the OMG defines a security model to support DDS security [3]. This model provides data confidentiality and integrity, authentication and authorization of writers and readers, and non-repudiation of data. DNS-SD and mDNS-SD do not provide any security capabilities.

All the protocols support QoS parameters specification in their service descriptions. DNS-SD and mDNS-SD use TXT records, CoAP-SD uses a set of key-value properties that each user defines, and DDS-SD through a predefined set of QoS parameters that each participant can use in its description. The difference in QoS management is how each protocol uses this information at discovery time. DDS-SD provides a set of QoS policies which allows the development of solutions for critical scenarios such as military applications. These policies are part of the participant descriptions and they are used in the Participant Discovery Protocol to match two or more participants which can exchange information about their writers and readers. The client in CoAP-SD must define the QoS as a query parameter using the format *?parameter=value*. DNS-SD and mDNS-SD do not use QoS information in the discovery. They provide this information to the client after service resolution. If the user wants to filter services by QoS, human intervention and programming effort after discovery are needed. The results for context management shows that CoAP supports the expression and use of context parameters during discovery. This information can be expressed using a key-value format in the service descriptions and the query can contain context parameters using the format *?parameter=value*. DNS-SD and mDNS have support for context in the service description using the

[8]DNS-SD - http://www.java2s.com/Code/Jar/d/Downloaddnssdjar.htm
[9]CoAP Californium - http://www.eclipse.org/californium/
[10]JmDNS - http://jmdns.sourceforge.net/
[11]DDS - https://community.rti.com/downloads/rti-connext-dds-raspberry-pi

Table 4: Results for qualitative evaluation.

QUALITATIVE METRICS		PROTOCOLS			
FEATURE	**METRIC**	**CoAP-SD**	**DNS-SD**	**mDNS-SD**	**DDS-SD**
Heterogeneity	Interoperability with other service discovery protocols?	It can be used together with DNS-SD, mDNS, and HTTP which enables a link with the traditional Web.	This protocol can be used together with mDNS.	This protocol can be used together with DNS-SD.	It does not have interoperability with other protocols.
Security	Does the protocol offer security mechanisms?	It uses the Datagram Transport Layer Security (DTLS) protocol to address security issues in IoT	The protocol itself does not provide security mechanisms.	The protocol itself does not provide security mechanisms.	OMG defined a beta specification to support security in DDS.
QosSupport	Does the protocol allow QoS parameters in services' description?	QoS parameters can be expressed in service descriptions.	QoS parameters can be expressed in the TXT records.	QoS parameters can be expressed in the TXT records.	DDS defines a set of QoS parameters which are used to describe QoS of participants.
	Does the protocol use QoS parameters in service searching?	QoS properties can be used as a parameter in the searching.	QoS properties can not be used as a parameter in the searching.	QoS properties can not be used as a parameter in the searching.	QoS properties are used to determine if two or more participants match.
Context Management	Does the protocol supports context information in services' description?	Context parameters can be expressed in service descriptions.	Context parameters can be expressed in the TXT records.	Context parameters can be expressed in the TXT records.	The protocol does not provide context support.
	Does the protocol use context information in service searching?	Context parameters can be used as a parameter in the searching.	Context parameters can not be used as a parameter in the searching.	Context parameters can not be used as a parameter in the searching.	The protocol does not provide context support.

TXT records. This information can be used after the service resolution, which implies human intervention after service discovery. DDS-SD does not provide context support.

The rest of this section presents the results for the quantitative evaluation. Figure 2 shows the precision of the retrieved services from the heterogeneous datasets defined in the Section 5.1. Syntactic and semantic heterogeneity affect the precision in all protocols. More than 50% of retrieved services are not relevant to the query used when the services have syntactic differences in their descriptions, and more than 60% when the services have semantic differences. This is because the main attribute to search a service is the service type which means that if the type is wrong but it meets the query, then the service is retrieved. Search recall is also affected by the service heterogeneity (Figure 3). Almost 20% of relevant services were not retrieved when they have syntactic differences, and more than 30% were not retrieved when the services have semantic differences. This is because the protocols use exact string matching between the service type attribute and the query parameter.

Figure 4 shows the response time to register a new service in each protocol. DDS-SD has the best performance followed by CoAP-SD. The DNS based protocols have a higher response time because they perform domain resolution in the registration process in order to create the DNS hierarchical structure, whereas DDS-SD and CoAP-SD only create a new entry in the registry. DNS-SD has a slower performance than mDNS-SD, however mDNS-SD is less reliable because its response time varies considerably. This variation is due to the interaction between jmDNS and Avahi. In our experiments, we observed that Avahi spends time interpreting registration requests from jmDNS. Although DNS-SD has a higher latency, its response time does not have big variations. The response time in all protocols is not affected by the amount of services, however each protocol presents a limit in the number of services that they can register. For example, CoAP-SD's limit is 2000 services in the registry. For DNS-SD, mDNS-SD and DDS-SD the number of services is restricted by the resources where the protocol is running. For DNS-SD and mDNS-SD this limit is configurable in Avahi. The integration of directories with more storage capacity can remove this limitation.

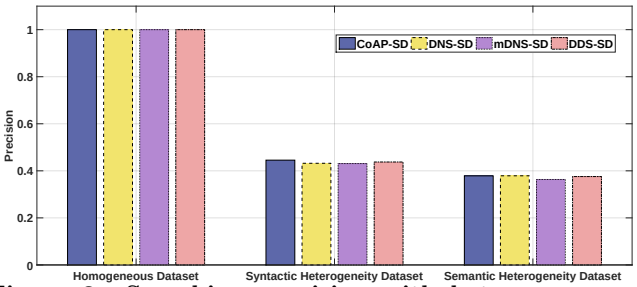


Figure 2: Searching precision with heterogeneous service descriptions.

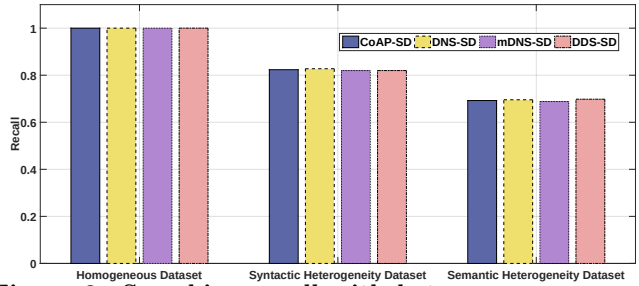


Figure 3: Searching recall with heterogeneous service descriptions.

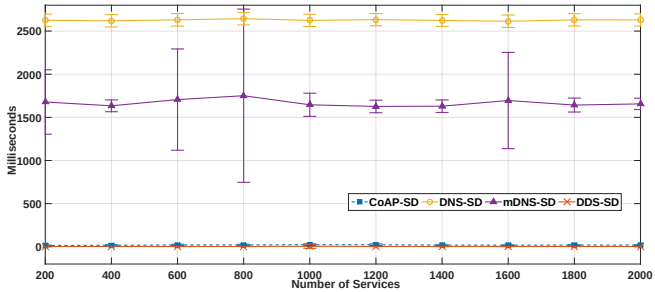


Figure 4: Service registration response time with different number of services.

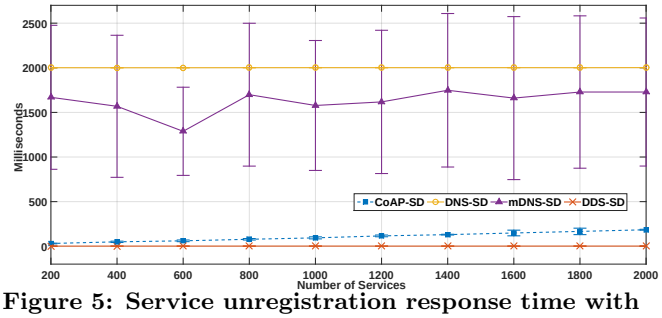


Figure 5: Service unregistration response time with different number of services.

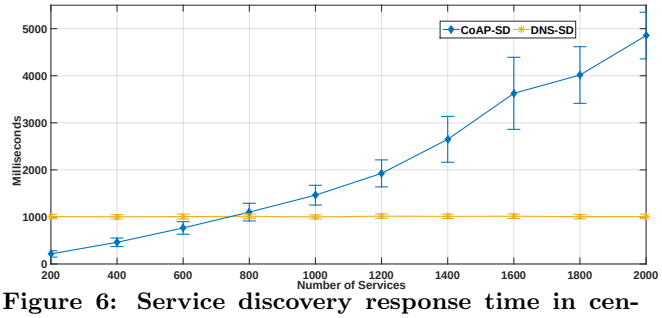


Figure 6: Service discovery response time in centralised protocols with different number of services.

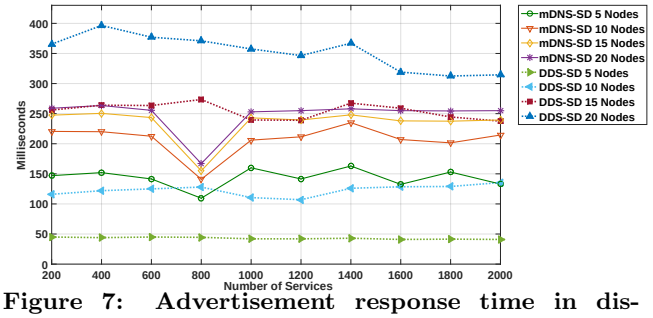


Figure 7: Advertisement response time in distributed protocols with different number of services.

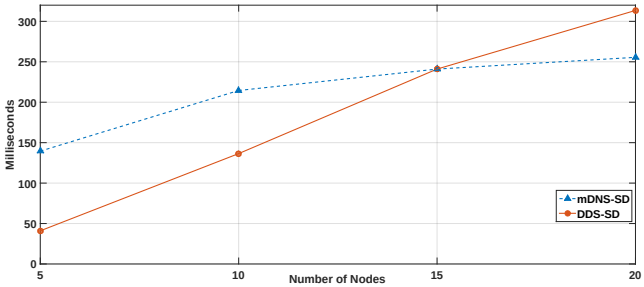


Figure 8: Advertisement propagation rate in distributed protocols with 2000 services.

The unregistration response time is shown in Figure 5. DDS-SD has the best performance. CoAP-SD has similar results as DDS-SD, but its response time is affected by the amount of services in the registry because the protocol searches for the service to delete in the centralised directory. The DNS based protocols have the worst performance because of the domain resolution. However, the latency is not affected by the number of services because of the hierarchical structure of the registry. mDNS-SD has a big variation in its response time as in the registration process.

Figure 6 presents the discovery response time in centralised protocols. CoAP-SD has a better performance than DNS-SD before 750 services in the directory. Then, DNS-SD keeps the response time constant regardless the number of services in the directory. CoAP-SD does not scale because it has to perform the search over the complete directory and each time the number of entries in the directory is bigger. DNS-SD uses asynchronous service discovery where the client is notified when a new service that meets the query becomes available. This asynchronous service discovery requires a listener implementation in the client side which implies more human intervention and programming effort than the synchronous request in CoAP-SD.

Figure 7 compares the proactive service discovery in mDNS-SD and DDS-SD with different number of nodes in the network and different number of services in the environment. The advertisement in mDNS-SD and DDS-SD is not affected by the number of services in the network as their performance is constant. Both protocols are affected by the number of nodes in the network. Figure 8 shows the growth rate in advertisement propagation in each protocol for different number of nodes with 2000 services. mDNS-SD is more efficient at propagating services after 15 nodes in the network because DDS-SD checks the service definition and its structure when it is advertised, to determine if it was seen before.

Figure 9 shows the unadvertisement response time in distributed protocols to complete the proactive discovery evaluation. This figure compares the time to unadvertise a service in each protocol with different numbers of nodes in the network and different numbers of services in the environment. The unadvertisement in mDNS-SD and DDS-SD is not affected by the number of services in the network. Both protocols are affected by the number of nodes in the network. DDS-SD has a better performance than mDNS-SD regardless the number of nodes. Figure 10 shows the growth rate in the unadvertisement propagation time for different number of nodes with 2000 services. DDS-SD has a better performance because it does not check the service definition unlike in the service advertisement.

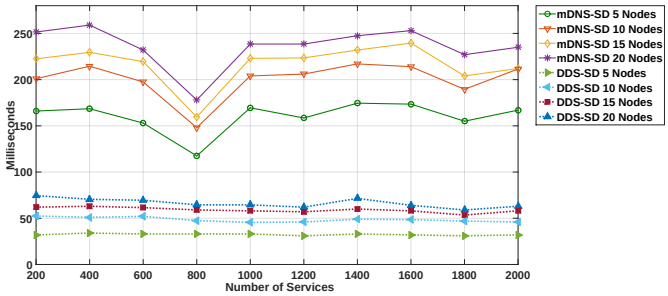


Figure 9: Unadvertisement response time in distributed protocols with different number of services.

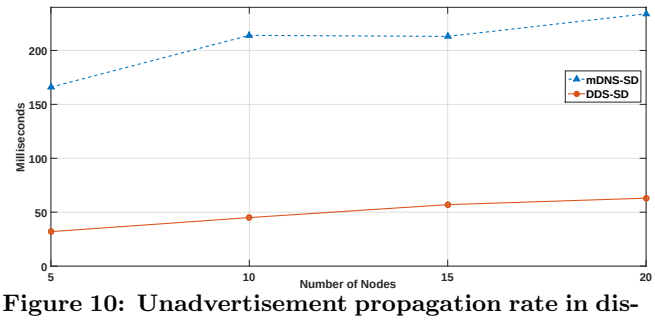


Figure 10: Unadvertisement propagation rate in distributed protocols with 2000 services.

7. CONCLUSIONS AND FUTURE WORK

This paper presents an evaluation of current service discovery protocols for IoT. The evaluation applied both qualitative and quantitative metrics, which were defined according to IoT requirements. The presented protocols were evaluated on a physical experimental setup.

Interoperability between the evaluated protocols is limited because each protocol defines its own service representation. While we assume the definition of standards to represent IoT services will improve the interoperability between IoT protocols and technologies. This representation must support the expressiveness, use and management of service attributes. The evaluated protocols allow the expression of context and QoS attributes, but, the use of this information in service discovery is limited. IoT requires context and QoS driven protocols in order to exploit its pervasiveness and offer optimal responses to user requests. The use of semantic technologies will allow the automatic use and management of context and QoS information in service discovery.

Search effectiveness metrics were impacted by service heterogeneity in all the protocols. Although search precision is important because IoT applications require correct services in short time, it was significantly affected because the protocols are limited to use exact and independent string matching between service attributes and query parameters. Future protocols must allow the expression, management and use of semantic relations between services to improve the search precision through smart service discovery. It requires lightweight semantic models and efficient search mechanisms.

The evaluated protocols offer good performance for the different service discovery phases. DDS-SD and CoAP-SD have low response time for service registration and unregistration. The response time is acceptable for service discovery in CoAP-SD with less than 800 services in the directory. The advertisement propagation time growth rate is low for mDNS-SD as well as in DDS-SD for unadvertisement. However, one main challenge for future protocols is the trade off between using semantic information to improve the discovery and keep the response time low. In addition, while not possible to evaluate with these protocols because of their scalability limitations, any assessment of a service discovery protocol should include consideration of thousands, if not millions, of services.

8. ACKNOWLEDGMENTS

This work is supported by the Science Foundation Ireland (SFI) under the project named SURF: Service-centric networking for Urban-scale Feedback systems.

9. REFERENCES

[1] The Real-time Publish-Subscribe Protocol (RTPS) DDS Interoperability Wire Protocol Specification. Technical report, Object Management Group, 2014. http://www.omg.org/spec/DDSI-RTPS/2.2/.

[2] Data Distribution Service. Technical report, Object Management Group, 2015. http://www.omg.org/spec/DDS/1.4/.

[3] DDS Security. Technical report, Object Management Group, 2016. http://www.omg.org/spec/DDS-SECURITY/.

[4] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash. Internet of Things: A Survey on Enabling Technologies, Protocols and Applications. *IEEE Communications Surveys & Tutorials*, 17(4):2347–2376, 2015.

[5] I. Al-Mejibli and M. Colley. Evaluating transmission time of service discovery protocols by using ns2 simulator. In *Wireless Advanced (WiAD), 2010 6th Conference on*, pages 1–6, 2010.

[6] K. An, A. Gokhale, D. Schmidt, S. Tambe, P. Pazandak, and G. Pardo-Castellote. Content-based Filtering Discovery Protocol (CFDP): Scalable and Efficient OMG DDS Discovery Protocol. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, pages 130–141. ACM, 2014.

[7] L. Atzori, A. Iera, and G. Morabito. The Internet of Things: A survey. *Computer Networks*, 54(15):2787–2805, 2010.

[8] E. Borgia. The Internet of Things vision: Key features, applications and open issues. *Computer Communications*, 54:1–31, 2014.

[9] T. A. Butt, I. Phillips, L. Guan, and G. Oikonomou. Adaptive and Context-Aware Service Discovery for The Internet of Things. *Internet of Things, Smart Spaces, and Next Generation Networking*, 8121:36–47, 2013.

[10] M. Castro, A. J. Jara, and A. F. Skarmeta. Enabling end-to-end CoAP-based communications for the Web of Things. *Journal of Network and Computer Applications*, 59:230–236, 2016.

[11] S. Cheshire and M. Krochmal. DNS-Based Service Discovery. RFC 6763, Internet Engineering Task Force, 2013. http://www.rfc-editor.org/rfc/rfc6763.txt.

[12] S. Cheshire and M. Krochmal. Multicast DNS. RFC 6762, Internet Engineering Task Force, 2013. http://www.rfc-editor.org/rfc/rfc6762.txt.

[13] S. Cirani, L. Davoli, G. Ferrari, R. Leone, P. Medagliani, M. Picone, and L. Veltri. A Scalable and Self-Configuring Architecture for Service Discovery in the Internet of Things. *IEEE Internet of Things Journal*, 1(5):508–521, 2014.

[14] W. Colitti, K. Steenhaut, N. De Caro, B. Buta, and V. Dobrota. Evaluation of constrained application protocol for wireless sensor networks. In *2011 18th IEEE Workshop on Local & Metropolitan Area Networks (LANMAN)*, pages 1–6. IEEE, 2011.

[15] A. Corradi, L. Foschini, and L. Nardelli. A DDS-compliant infrastructure for fault-tolerant and scalable data dissemination. In *The IEEE symposium on Computers and Communications*, pages 489–495. IEEE, 2010.

[16] S. K. Datta, R. P. F. Da Costa, and C. Bonnet. Resource discovery in Internet of Things: Current trends and future standardization aspects. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pages 542–547. IEEE, 2015.

[17] B. Djamaa and M. Richardson. Towards Scalable DNS-Based Service Discovery for the Internet of Things. In *Ubiquitous Computing and Ambient Intelligence. Personalisation and User Adapted Services*, volume 8867, pages 432–435. Springer International Publishing, 2014.

[18] R. Droms and T. P. Donahue. Dynamic DNS-Based Service Discovery, 2016.

[19] N. Gligoric, T. Dimcic, D. Drajic, S. Krco, I. Dejanovic, N. Chu, and A. Obradovic. CoAP over SMS: Performance evaluation for machine to machine communication. In *2012 20th Telecommunications Forum (TELFOR)*, pages 1–4. IEEE, 2012.

[20] A. J. Jara, P. Lopez, D. Fernandez, J. F. Castillo, M. A. Zamora, and A. F. Skarmeta. Mobile digcovery: discovering and interacting with the world through the Internet of things. *Personal and Ubiquitous Computing*, 18(2):323–338, 2013.

[21] A. J. Jara, P. Martinez-Julia, and A. Skarmeta. Light-Weight Multicast DNS and DNS-SD (lmDNS-SD): IPv6-Based Resource and Service Discovery for the Web of Things. In *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 731–738. IEEE, 2012.

[22] X. Jin, K. Hur, S. Chun, M. Kim, and K.-H. Lee. Automated mashup of CoAP services on the Internet of Things. In *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, pages 262–267. IEEE, 2015.

[23] D. Kaiser and M. Waldvogel. Efficient Privacy Preserving Multicast DNS Service Discovery. In *2014 IEEE Intl Conf on High Performance Computing and Communications, 2014 IEEE 6th Intl Symp on Cyberspace Safety and Security, 2014 IEEE 11th Intl Conf on Embedded Software and Syst (HPCC,CSS,ICESS)*, pages 1229–1236. IEEE, 2014.

[24] R. Klauck and M. Kirsche. Bonjour Contiki: A Case Study of a DNS-Based Discovery Service for the Internet of Things. In *Ad-hoc, Mobile, and Wireless Networks*, volume 7363, pages 316–329. Springer Berlin Heidelberg, 2012.

[25] M. Kovatsch, M. Lanter, and Z. Shelby. Californium: Scalable cloud services for the Internet of Things with CoAP. In *2014 International Conference on the Internet of Things (IOT)*, pages 1–6. IEEE, 2014.

[26] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials*, 16(1):414–454, 2014.

[27] T. Potsch, K. Kuladinithi, M. Becker, P. Trenkamp, and C. Goerg. Performance Evaluation of CoAP Using RPL and LPL in TinyOS. In *2012 5th International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5. IEEE, 2012.

[28] R. A. Rahman and B. Shah. Security analysis of IoT protocols: A focus in CoAP. In *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*, pages 1–7. IEEE, 2016.

[29] M. A. Razzaque, M. Milojevic-Jevric, A. Palade, and S. Clarke. Middleware for Internet of Things: A Survey. *IEEE Internet of Things Journal*, 3(1):70–95, 2016.

[30] E. Rescorla and N. Modadugu. Datagram Transport Layer Security Version 1.2. RFC 6347, Internet Engineering Task Force, 2012. http://www.rfc-editor.org/rfc/rfc6347.txt.

[31] J. Sanchez-Monedero, J. Povedano-Molina, J. M. Lopez-Vega, and J. M. Lopez-Soler. Bloom filter-based discovery protocol for DDS middleware. *Journal of Parallel and Distributed Computing*, 71(10):1305–1317, 2011.

[32] Z. Shelby, K. Hartke, and C. Bormann. The Constrained Application Protocol (CoAP). RFC 7252, Internet Engineering Task Force, 2014. http://www.rfc-editor.org/rfc/rfc7252.txt.

[33] A. Siljanovski, A. Sehgal, and J. Schonwalder. Service discovery in resource constrained networks using multicast DNS. In *2014 European Conference on Networks and Communications (EuCNC)*, pages 1–5. IEEE, 2014.

[34] M. Stolikj, P. J. L. Cuijpers, J. J. Lukkien, and N. Buchina. Context based service discovery in unmanaged networks using mDNS/DNS-SD. In *2016 IEEE International Conference on Consumer Electronics (ICCE)*, pages 163–165. IEEE, 2016.

[35] M. Stolikj, R. Verhoeven, P. J. L. Cuijpers, and J. J. Lukkien. Proxy support for service discovery using mDNS/DNS-SD in low power networks. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6. IEEE, 2014.

[36] B. C. Villaverde, R. De Paz Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch. Service Discovery Protocols for Constrained Machine-to-Machine Communications. *IEEE Communications Surveys & Tutorials*, 16(1):41–60, 2014.

[37] G. Yoon, J. Choi, H. Park, and H. Choi. Topic naming service for DDS. In *2016 International Conference on Information Networking (ICOIN)*, pages 378–381. IEEE, 2016.