A Reliable Communication Framework Using XMPP for the Internet of Things

Luan. Oliveira, Zair. Abdelouahab, Denivaldo. Lopes, Mario. Santos

***Abstract*—**Internet of Things (IoT) is a paradigm in which intelligent objects collaborate actively with other physical and virtual ones available in the Internet, and are characterized by a high degree of heterogeneity of devices and network protocols. However, several issues related to secure communications, including interoperability of devices, autonomous systems, require emphasis on applications for this scenario which may have a significant impact on many aspects of everyday life of the end user. To address these issues, some mechanisms to solve these problems are fundamental to. This paper presents a framework based on the XMPP protocol (eXtensible Messaging and Presence Protocol) for application development with a focus on secure and reliable communications in IoT environment. The Framework is based on the publish/subscribe model implemented with reliability for real-time communications with the Cloud. We use the security features provided by the XMPP such as TLS and SASL authentication. Based on a case study, we demonstrate the framework's features.

***Keywords*—**Internet of Things, Framework, XMPP, Security, reliability, TLS (SASL).

I. Introduction

THANKS to progress in the field of embedded devices, physical objects such as household appliances, industrial machinery, wireless sensors and actuators can now connect to the internet. Connecting all these objects to the Internet to achieve interoperability is is known in the literature as Internet of Things (IoT) [1]. These objects can be any device such as home appliances, tires, sensors, actuators, mobile phones, among others, which can be identified and connected to the Internet to exchange information and make decisions to achieve common goals [2].

These objects can participate in virtual processes, providing real-world data such as using sensors and allow to control and manipulate the real world as with actors. Current approaches to the IoT focus primarily on communication protocols to integrate "things" with standards protocol on the Internet considering limited computing and memory resources, and limited availability of bandwidth and energy [5].

The special focus is set on the integration of the things in the service layer. The work [4] exemplifies the communication in IoT as a platform for multiple services using an infrastructure with XMPP protocol for devices with limited resources. [7] describes an approach to access real-world objects using principles of RESTful Web of Things.

However, there are several scenarios of IoT such as connected cars, power management, health systems which have a much broader scope than the one considered by these cited works. They require user-centric services that involve real world objects, back-end systems and mobile devices to allow users to consume real-world data interacting with the physical environment in real time. Thus IoT basic communication infrastructure must support efficiently interactions of types (publish / subscribe) based services for mobile applications on one side and interactions of type request / response for the consumption of services on the other side. Furthermore, it should support a large number of device platform, including sensors.

The XMPP [6] stands out in this scenario as a standardized protocol of open source, which provides extensibility and open innovation than legacy technologies. This protocol is well evolved and is standardized by the IETF (Internet Engineering Task Force). It has over 10 years of development, and there are thousands of Jabber servers on the Internet and thousands of users (eg Google Talk) and applications developed using features of instant messaging in different fields of activity. XMPP is totally decentralized where everyone can use its server and manage it independently of the network, providing interoperability features, HTTP, files transfers, servers federation, services discovery and provides natively security features.

XMPP is designed as a real time communication protocol which supports low latency messaging. It uses a single IADB (Definition of Jabber Identifiers) to provide a framework for globally unique addresses similar to e-mail addresses identifiers. It also allows different types of messages (XML stanza) and enables two-way communication-based publish / subscribe and interactions of request / response, allowing data provisioning.

XMPP is highly extensible, allowing specifications of

Luan Oliveira. is with the Federal University of Maranhão, DEE/CCET, Campus do Bacanga, São Luis-MA 65085-580, Brazil. (phone:+559832728234; e-mail: luan.oliveira.c@gmail.com).

Mario. Santos. is with the Federal University of Maranhão, DEE/CCET, Campus do Bacanga, São Luis-MA 65085-580, Brazil. (e-mail: marioh90@gmail.com).

Zair. Abdelouahab. is with the Federal University of Maranhão, DEE/CCET, Campus do Bacanga, São Luis-MA 65085-580, Brazil. (e-mail: zair@dee.ufma.br).

Denivaldo. Lopes. is with the Federal University of Maranhão, DEE/CCET, Campus do Bacanga, São Luis-MA 65085-580, Brazil. (e-mail: dlopes@dee.ufma.br).

XEPs (XMPP Extension Protocols) such as XEP-0045 a multi user chat, XEP-0030 a service discovery protocol and XEP-0060 a publish-subscribe service.

This paper presents a framework based on XMPP to simplify the application development process in IoT and to provide a reliable communication mechanism between objects and the Cloud [14]. We show the architecture and the framework in Section 2 as well as a case study in Section 3. The latter is a real-time chat application using the Framework. Finally, section 4 presents concluding remarks and future works.

II. The Proposed Approach

The framework architecture is shown in Figure 1. The framework heart is composed of the XMPP protocol used as the communication system; all users including XMPP services and applications are clients can be identified by JID (e.g. usuario@servidor.com) . A second major block of components consists of the Connection Factory, Reliability Management, Security Management (TLS) and authentication (SASL).

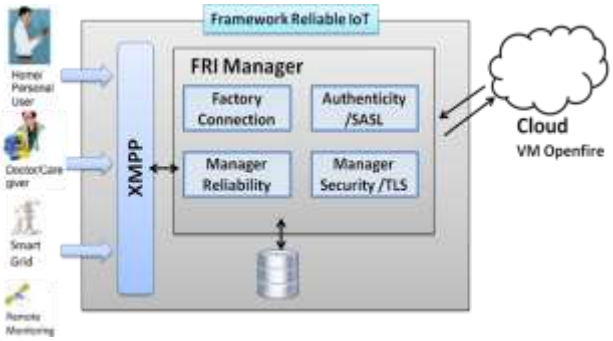

Fig 1. The Framework architecture

The connection Factory Component provides a run-time environment which creates a connection with Jabber servers. It chooses an available Jabber server to establish a connection transparently to the user.

The Reliability Manager component provides an environment which guarantees reliability is analyzed for multiple dynamically communications runtime in the post checking with successful receipt by the recipient. This mechanism verifies cases where the server is unavailable and / or unavailable recipient performing the treatment of these exceptions, requesting connection to the change of the connection factory to another server or creating exceptions for runtime applications.

The Security and Authentication Manager component provides communication using TLS (Transport Layer Security - RFC 2246 [10]) ensuring the integrity and confidentiality of data, encryption of XML streams as defined using the extension "STARTTLS" (RFC 2595 [13]). This extension allows to confirm the identity of the server, verify the user's identity (optional), and integrity using SASL (Simple Authentication and Security Layer - RFC 4422 [11] and RFC 2222 [12]) to guarantee authentication.

III. A Case Study

To demonstrate the feasibility of the approach, we implement an IoT-Reliable-Chat. In this implementation, we wants to show an example of the architecture as a generic model for reliable and secure communication. Thus, the use of the framework becomes feasible for any other application that uses this model of communication such as cases of applications for remote monitoring, smart grids, medical monitoring, among others.

A. A Use Case of IoT-Reliable-Chat

The IoT-Reliable-Chat demonstrates the real-time capabilities for exchanging messages using the implemented framework. The application is able to view messages exchanged with another application, view the status (available, unavailable, busy), use multichat with other user. The engine stores in its database connection information, recipient (servers - addresses) and use encryption and authentication (optional). The latter is achieved using the Security Manager component / TLS / SASL (section 2.1). Before performing communication, it checks the availability of the server, clients (recipients) and in case of success a communication is established. The verification of the communication reception is achieved by the Manager Reliability component (section 2.1) and when necessary it also checks the retransmission when necessary. In the case of error with the initial connection, the engine performs attempt to connect to other previously registered servers. In case of no success, an exception is generated for the application.

B. Implementation

To develop the framework, we have used the Smack library [3] for XMPP communication for the framework. This library is an Open Source XMPP instant messaging and presence. It is a pure Java library, which can be embedded within applications to create XMPP client and simple XMPP integrations such as sending notification messages and presence enabling devices. The implementation is carried out on Android and Windows OS (PC) using the OpenFire server [3] Eucalyptus cloud [8].

To demonstrate the applicability of the framework for different architectures, we use three different architectures: devices with high resources such as a PC, devices with limited resources such as a Smatphone and devices with simple resources such as RFID sensors or reader. In the latter case, a XMPP also provides ad-hoc connectivity in local WLANs, any sensors and actuators such as a light switch for example able to interact with XMPP can also be integrated into the system and if necessary its messages are encapsulated with through XMPP.

C.3.3 Framework Evaluation

We have evaluated the proposed framework using the following cases:

- Message Reception Verification: The Framework verifies that the message sent is received successfully by the recipient by notifying the application (XEP-0184). The sender of a message can request a notification that the message is delivered to the intended recipient. In the event of any failure, a retransmission is done.
- Analyze the absence of the application Primary Server: The framework is designed to use XMPP servers. If it can not connect to the default server when sending the message, it tries to restore connection during a certain amount of time defined by the application until the connection is establish and the message sending is realized.
- Analyze the lack of connection with the Application Primary Server: In case the primary server does not respond, another connection can be established with another available server.
- Sending messages with a secure mechanism: The framework provides functions to perform communication using TLS security protocols and authentication SASL, but not all messages are required to use this feature.
- Platform Independence: As the framework runs on the JVM [9], applications are said to be platform independent.

IV. Conclusion

This paper presents a framework for the Internet of Things to integrate real-world objects, back-end systems and mobile devices. The platform is based on XMPP as the communication protocol to support efficient and secure communication with reliability and security management mechanism. The platform simplifies the application development using framework classes able to abstract features of great importance in building IoT applications involving services running on mobile devices and the cloud. The potential of the approach is demonstrated with a generic case study showing the feasibility of the framework especially using the evaluation tests. The next steps will be focus on the use of devices information context, performance evaluation and scalability.

Acknowledgment

Financial supports of CAPES and FAPEMA are gratefully acknowledged.

References

[1] GARTNER. *Internet of Things definition*. Available at: http://www.gartner.com/itglossary/internet-of-things. Accessed on: 20/1/2015.

[2] L.Atzori, A. Iera, G. Morabito, *The Internet of Things: A survey*, Computer Networks Volume 54, Issue 15, 28 October 2010, , p. 2787–2805.

[3] REALTIME, *Ignite. Smack API*. Available at: http://www.igniterealtime.org/projects/smack, 2008. Accessed on: 19/1/2016.

[4] [4] T. C. de França, P. F. Pires, L. Pirmez, F. C. Delicato, and C. Farias. Web das coisas: *Conectando Dispositivos Físicos Ao Mundo Digital*, minicourse, Federal University of Rio de Janeiro, 2014.

[5] S. Bendel, et al. *A Service Infrastructure For The Internet Of Things Based On XMPP*. In: Pervasive Computing and Communications Workshops (PERCOM Workshops), IEEE, 2013. p. 385-388.

[6] P. Saint-Andre, *Extensible Messaging And Presence Protocol (XMPP): Core*, IETF RFC 6120, 2011.

[7] D. Guinard, V. Trifa, E. Wilde, *A Resource Oriented Architecture For The Web Of Things*. In: Internet of Things (IOT) Conference, IEEE, 2010. p. 1-8.

[8] EUCALYPTUS. Available at: http://www8.hp.com/br/pt/cloud/helion-eucalyptus-overview.html Accessed on 19/1/2016.

[9] *JAVA JVM*. Available at: https://www.oracle.com/java/technologies/index.html. Accessed on 19/1/2016.

[10] T. Dierks, C. Allen, W. Treese, P. Karlton, A. Freier, and P. Kocher, The *TLS Protocol Version 1.0*, [RFC 2246], January 1999.

[11] A. Melnikov and K Zeilenga., *Simple Authentication And Security Layer (SASL)*, [RFC 4422], June 2006.

[12] J. Myers, *Simple Authentication And Security Layer (SASL)*, [RFC 2222], October 1997.

[13] C. Newman, *Using TLS with IMAP, POP3 and ACAP*, [RFC 2595], June 1999.

[14] *Apps e IoT: qual é o papel da nuvem?* Available at: http://convergenciadigital.uol.com.br/cgi/cgilua.exe/sys/start.htm?infoid=38486&sid=97#.Vq_jhFL0wfY.l Accessed on 19/1/2016.