Järvinen, Juha; Marttinen, Aleksi; Järvinen, Risto; Carlén, Per; Luoma, Marko; Peuhkuri, Markus; Manner, Jukka

Enabling XEP-0258 Security Labels in XMPP

Published in:
MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)

DOI:
10.1109/MILCOM.2018.8599711

Published: 01/01/2018

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Järvinen, J., Marttinen, A., Järvinen, R., Carlén, P., Luoma, M., Peuhkuri, M., & Manner, J. (2018). Enabling XEP-0258 Security Labels in XMPP. In *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)* (pp. 661-666). (IEEE Military Communications Conference (MILCOM)). IEEE. https://doi.org/10.1109/MILCOM.2018.8599711

Enabling XEP-0258 Security Labels in XMPP

Juha Järvinen*, Aleksi Marttinen*, Risto Järvinen*, Per Carlén†, Marko Luoma*, Markus Peuhkuri* and Jukka Manner*

*Department of Communications and Networking, Aalto University
Postal Address: PO Box 15600, FI-00076 AALTO
Email: {Juha.Tapio.Jarvinen, Aleksi.Marttinen, Risto.Jarvinen, Marko.Luoma, Markus.Peuhkuri, Jukka.Manner}@aalto.fi

†Swedish Defence Materiel Administration (FMV)
Contractor from Peldakon
Knivsta, Sweden
Email: pc@peldakon.se

***Abstract*—Labeling makes data easier to process and search. This is vital while handling large amounts of data. By using labeling to indicate security classes, the number of different applications can be reduced as there would be no requirement to use separate application for each security class. The Extensible Messaging and Presence Protocol (XMPP) is an Instant Messaging (IM) protocol for Federated Networks. Extension XEP-0258 defines security labeling in XMPP, however not all systems support this draft standard.**

In this paper we present the processes and limitations of how different XMPP domains with different XMPP labeling capabilities can be joined together in a static or dynamic way. We have constructed an implementation to verify and validate the presented approach. In addition, we propose a way how non XEP-0258 applicable XMPP domains should be joined to mission networks, and hence security labeling would work more smoothly in the future. Finally, we show that the mangling of security labels does not significantly increase the anticipated delay of the communication.

I. Introduction

For numerous people, the labeling of content has become part of everyday life through social media applications. For instance, in Twitter and Instagram users can tag their content with so-called hashtags indicate to other users the (sub)topics of the shared content. Labeling makes the content more easily findable by others, and hence improves the popularity of the shared content and also the user experience of the users. For instance, a user can share her opinion of an ongoing sport event labelled with an appropriate hashtag. Simultaneously other users interested in the event can follow the label (hashtag) and their feed is fed with the updates of the event.

Data labeling is an active research topic in the critical communications framework. For instance the standard STANAG 4559 defines mechanisms to share intelligence and surveillance data between federated nations. In the context of STANAG 4559 the data may be motions imagery recorded by UAVs (Unmanned Aerial Vehicles), still images from satellites or tracking data. In this approach, intelligence and surveillance data may be labeled, for instance, with releasability and security classification labels. STANAG 4774 and 4778 define respectively the format for security labels and the profiles on how to integrate the security label into various protocols. When these labels are provided, Information Exchange Gateways (IEGs) can be used on the borders of the security domains, and the communication flows can be filtered based on the labels.

The instant messaging service is a key application in critical communications and may be an enabler of mission success. The Extensible Messaging and Presence Protocol (XMPP) is one popular approach for instant messaging in numerous critical infrastructure organizations. XMPP offers open standards for communications, and the community is actively developing new extensions (XEPs, XMPP Extension Protocols) for the framework. XEP-0258 defines the use of security labels in XMPP. XMPP profile defined in STANAG 4778 is based on XEP-0258 and thus is the preferred solution for the future. In the interim, a solution for inter-operability is required.

In this paper, we introduce our solution to enable communication between different XMPP domains, that use different security labeling systems, such as based on XEP-0258, STANAG 4778 or some proprietary mechanism. Nations may use different labeling mechanisms due to presence of legacy systems, de facto standards etc. Our solution allows the integration and inter-operation of multiple systems of varying compliancy. We have shown the system is capable of operating with four different systems. The measured increase in message transport delay is shown to be insignificant.

This article is organized as follows. We start with an overview of XMPP and Data labeling in Section II. Our solution for joining different Security labeling capable domains is presented in Section IV. We then discuss the future work of our solution in Section VI, and conclude the article with recommendations in Section VII.

II. Data labeling and XMPP

A. Data labeling

Data labeling is used to describe data in some formal way, e.g. to give a tag to it. Data can be anything such as text, videos, pictures or voice. Labeling is performed, for example, for machine-learning purposes and classification. Labels can be given by a human or a machine. For instance, people do tagging in the famous Instagram mobile application for their photos and videos in order for other users to find their photos. In the example presented in Fig 1, we have a dataset which is tagged with *priority* tags, and having '*low*' as a value. The used tags and their values always depend on the environment

where the labeled data is used. In this case, the labeled data is quite unusable in most contexts, except maybe in the ”pointless texts”.

```
<priority="low">Lorem Ipsum, Lorem Ipsum!</priority>
```

Fig. 1. An example of a labeling data in XML document.

In networking, especially in Federated Mission Networks (FMN), classification makes possible decreasing the amount of components (applications, systems and networks), when data of different security classes can be handled in the same system (see Fig. 2). Thus no more different clients are needed to handle, for example, emails, which belong to different security classes. This is one approach how cross-level labeling can be utilized in the Federated Networks. Of course, modifications must be implemented to the systems to ensure that they handle labels correctly. In addition, extra components such as data guards or IEGs have to be added to networks and elements. They are like label firewalls, that forwards or drops data according to, for example, clearances, and security policy.

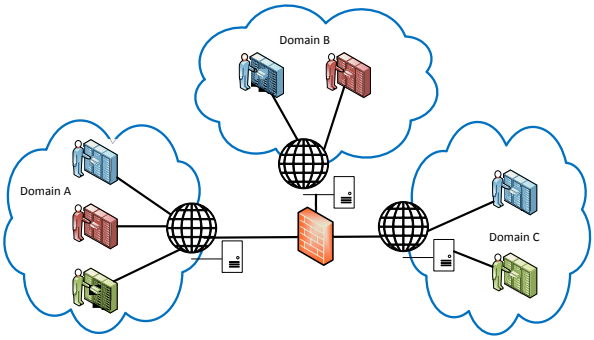

Fig. 2. A labeling example.

How to label data? The correctness of the labels is always critical. By giving an incorrect or inappropriate label to data, a receiver can miss crucial information which might be important for mission success. As we can see, labeling data by a human can cause errors. In contrast to that, labeling can be implemented automatically. For instance, for automatic video data labeling Yan et al. [1] developed some automated labeling methods within an active learning framework for decreasing the amount of human errors.

Also de Bie et al. have presented a similar model [2] that after obtaining a labeled dataset, machine learning models can be applied to the data so that new unlabeled data can be presented to the model, and a likely label can be guessed or predicted for that piece of unlabeled data. Kongsgård et al. [3] have represented a policy-based access control framework to label trusted information.

As we can see, there are quite many different automatic ways for labeling data. But how well are they able to perform their job? Sometimes a need for correctness is 100 %, other times much less is considered adequate.

As mentioned earlier, labeling is not in itself enough in a cross-domain environment; we, however, need guards which implement the policy of an organization, a mission etc; deciding which participants can send messages and which class etc. For example, Kongsgård et al. [4] build data-driven methods for guards which infer the words associated with classified content. Then when using ”wrong” words in too low classes, an alarm is given.

B. Extensible Messaging and Presense Protocol

The Extensible Messaging and Presense Protocol (XMPP) is a middleware protocol that started out as an Extensible Markup Language (XML) based instant messaging (IM) protocol. The basic protocol was collected into four RFCs (Request for Comments) [5]–[8] by the Internet Engineering Task Force (IETF) XMPP working group in 2002. This basic definition has then been widely extended by the XMPP Extension Protocol (XEP) standards [9]. Currently there are 156 XEPs, but most implementations only use a subset of those, based on the focus of the implementation.

Messages are sent as XML message structures called *stanzas*. Connections between servers may be encrypted with TLS. There is also XEP for end-to-end message encryption based on OpenPGP [10], [11].

The modifiability and flexibility has made XMPP widely applied in many applications for more than just messaging. For example, Jingle protocol extensions add peer-to-peer communications, like file-transfer, VoIP session setup and NAT-traversal. XMPP continues to be applied in a variety of new applications, such as Internet of Things (IoT) [12], [13], game matchmaking and in-game chat.

By design XMPP works as a Federated Network. This design is demonstrated in Fig. 3. Clients make a client-to-server (c2s) connection to their own domain-specific server. Clients *user1@example.com* and *user2@example.com* both connect to their respective domain server, example.com. Authentication, intra-domain message deliveries and offline services are performed by the local server.

When communicating with users outside of one’s own domain, the servers will form inter-domain server-to-server (s2s) connections on demand. For example, when *user1@example.com* wants to communicate with *jarvinen@example.fi*, the local server will form a connection to the foreign server. The connection points for s2s connections are found using DNS SRV records, and the connection will always be direct, without relays or proxies.

III. Security Labeling System in XMPP

Security labels in XMPP is defined in XEP-0258 [14]. It is a draft from 2013, and specifies how security label metadata is carried in XMPP. The standard defines a mechanism for carrying ESS Security Labels in XMPP. ESS Security Labels are specified in RFC 2634 [15]. Additionally, other format can be used as security labels. The main point is to use such a labeling structure that each participant understands; for example, the sanitization process of XMPP stanzas in an XMPP server does not clear ”unknown” markings, although

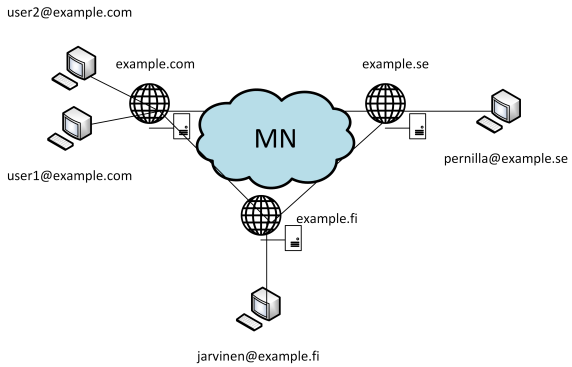

Fig. 3. The structure of a federated XMPP network.

its own client understand these markings. In this article, we assume that XEP-0258 is equal to ESS Security label format.

In XMPP, a normal Instant Message (IM) between two participants is defined as in Fig. 4. Security labels are added just after the written message in XML format, as in Fig. 5 a message with an ESS Security Label is presented. Finally, in Fig. 6 is presented another way to include security labeling in the stanza. The approach is an example of company proprietary method, which references a correct namespace but it is not using a schema in a correct way, and thus does not follow XEP-0258.

XEP-0258 defines how a server advertises its label catalogue to its connected clients, when the clients perform the Service Discovery (XEP-0030) [16] information request. For different services, for instance, IM and Multi-User Chat (MUC), this catalogue can differ from each other, but also in time.

```
<message to='pernilla@example.se'
         from='jarvinen@example.fi/swift'>
    <body>This is a message to Pernilla.</body>
</message>
```

Fig. 4. An example of a normal IM message in XMPP.

```
<message to='pernilla@example.se'
         from='jarvinen@example.fi/swift'>
    <body>This content is classified.</body>
    <securitylabel xmlns='urn:xmpp:sec-label:0'>
        <displaymarking fgcolor='black'
        bgcolor='red'>SECRET</displaymarking>
        <label><esssecuritylabel
            xmlns='urn:xmpp:sec-label:ess:0'>
            ZQOZAQQGEEo=
        </esssecuritylabel></label>
    </securitylabel>
</message>
```

Fig. 5. An example of a message with an ESS security label in XMPP.

The labeling format can be anything that clients and servers understand including the stanza structure and its informal message. In XMPP, it is very easy to add even weird control structures inside $< body/ >$ field, since the s2s connections are direct from end server to end server, at least in the Internet.

```
<x class="CONFIDENTIAL"
   x-securityBanner="ORG CONFIDENTIAL
   RELEASABLE TO EXERCISE"
   x-Marking="ORG CONFIDENTIAL
   RELEASABLE TO EXERCISE"
   owner="ORG" xmlns="urn:org:edu:ic:ism:v1"
   x-label_bg_color="#642400"
   x-label_fg_color="#FFFFFF"/>
```

Fig. 6. Another example of adding security labels into a message in XMPP: company proprietary.

In Mission Networks, connections may be intercepted by IEGs or proxies [17], which can act also as a message sanitizer and drop an entire message (stanza) or just the elements that are not understood. These kinds of special structures usually act very well among a few participants, but when the amount of structures increases, we need standards.

IV. Security labeling in XMPP communication

As stated earlier, our problem is that we have in the Mission Network single or multiple domains with different XMPP capabilities (servers, clients, different types of security labeling systems). This is due to individual nations having different implementations of varying age. Furthermore, some elements are *de facto*, although they do not fulfil all the required standards or specifications etc. In other words, the systems are very heterogeneous: there is no single vendor server and no single client type.

For example, we can have the following scenarios in the Mission Networks:

- A single XMPP server that is an open-source product and supports XEP-0258 (ESS). There are clients with different types of security labeling systems connected to it. *Scenario 1*.
- Multiple domains with different XMPP capabilities:
  - A client and a server, which both support XEP-0258. Both are open-source products. *Domain A*.
  - A commercial client, that supports company proprietary (*hardcoded*) security labels that are not in XEP-0258 format. In addition, the client becomes blocked if it gets an XEP-0258 advertisement from its server. An open-source server which does not support any XEP-0258 functionalities. *Domain B*.
  - A client which does not understand security labeling. An open-source server which supports XEP-0258 functionalities. *Domain C*.
  - A client which does not understand security labeling. A server, which does not understand XEP-0258 functionalities. *Domain D*.

In the following sections we go through the procedures.

A. Scenario 1

In this scenario there is a single XMPP server that is an open-source product and supports XEP-0258. There are clients with different types of security labeling systems connected to

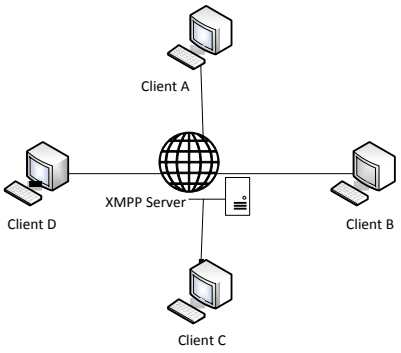

Fig. 7. A single domain.

it (Fig. 7). This is surely the easiest case, since there is only one XMPP server and we are able to modify the source code.

The procedure for both Instant Messaging and Multi-User Chat are similar, since everything happens under a single server.

1) Identifying participants.
2) If both a sender and receiver use XEP-0258, then we do nothing
3) If one participant supports XEP-0258 and another uses hardcoded security labels, then
   - No XEP-0258 advertisement in a login procedure for client which uses hardcoded values since some client may go in non-functioning state.
   - Replacing security labels in stanzas from the sender into correct format for the receiving client.
4) If the client does not support XEP-0258 or another security labeling system at all, then
   - A user types the security clearance in the beginning of the message, for example *[SECRET]*, that is wrapped out in the server and moved to the security labels of the receiver side. Or vice versa.

B. Scenario 2: Domains A, B and C

If we have a Mission Network with different domains (Fig. 8) with different XMPP security labeling features such as those described there (that either domain's server or client has some security awareness), the procedure is straightforward if we have a static table that indicates that a certain domain has certain users using certain client software. We can easily do mangling to XMPP stanzas where it is possible. In this scenario:

- All the stanza modifications happen in the servers of domains A or C. Not in B's server, since it does not know anything about security labels.
- If a stanza is heading to the C domain or coming from it, the procedure is similar as above: a user types a security clearance in the beginning of the message, for example *[SECRET]*, that is wrapped out in the server and moved to the security labels of the receiver side. Or vice versa.

This is the operation in a static system. If a user wants to send a message for the first time to a previously unknown

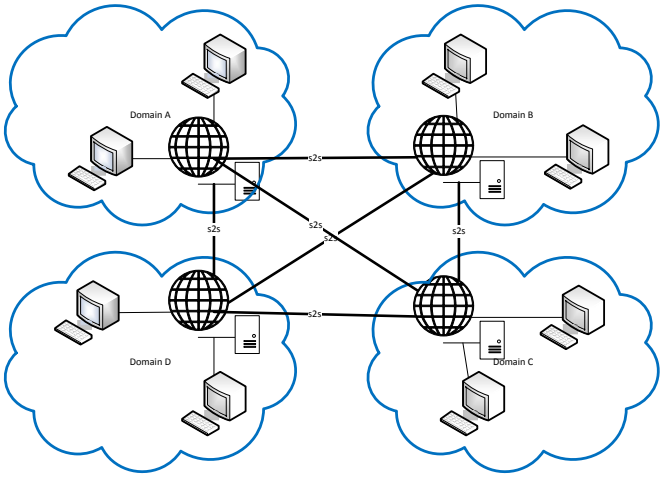

Fig. 8. Multiple domains.

domain, the system cannot know the client software and its security label capabilities. There can be default values, for example, the instantiation values of a Mission Network. More dynamical solution is to have all users participate in a certain MUC room all the time, thus making all the system components aware of the client capabilities. When the capabilities are known, we are able to mangle stanzas for every receiver.

C. Scenario 3: Domains A, B, C and D

In instant messaging, the domain D causes problems, since neither its server nor client support any security labeling functionalities. In this scenario, if the client in domain B wants to communicate with the client in domain D or vice versa, a user has to type a security clearance in the beginning of the message, for example *[SECRET]* – although there is a security labeling system in the client software in domain B. All the other processes are as in the previous scenario.

If in multi-user chat, a room is hosted in the XMPP server in domain B, all the users in domains B and D must type a security clearance in the beginning of the message. In addition, stanzas from domains A and C have to be modified to use the same security mechanism, that is different than in instant messaging.

D. Observations

As noticed here, a single homogeneous system is simplier: 1) all the clients and servers support the same XEP-0258 Security Labels in the XMPP system, or 2) all the labels are hardcoded in the client software and none of the servers support this certain XEP, they just forward stanzas between servers without any packet sanitation procedures.

The second scenario where there were three different domains was yet handled and functions quite easily.

The last scenario is the most challenging and requires extra work even from the user in certain domains. A good question arises here: is there any good reason to use any security labeling system in this kind of scenario.

E. Validation

We validated our procedures in all three test cases. The first one was similar as scenario 1 - a single domain (Fig. 7) with an open-source XMPP server with XEP-0258 support. There were two types of connected clients: 1) a client with XEP-0258 support and 2) a commercial client with a company proprietary (hardcoded) security labeling system.

We used the Prosody XMPP server, since it has XEP-0258 support. It is an open-source project and written with the Lua scripting language. All the added-on features are called modules and are written in the same language. Prosody has a module for a Layer 7 firewall called $mod\_firewall$. Our system uses the functions of this module. The firewall rules are based on the procedures presented earlier in this section. XEP-0258 security labels are transformed to a company proprietary labels and vice versa.

As a result, we observed that both clients were able to send messages correctly either using instant messaging or in a multi-user chat room. Without these rules this would have been impossible.

The second test case was similar to scenario 2, but only with domains A and B as in Fig. 9. In domain A we used clients with XEP-0258 support and the server was Prosody, like in the first scenario. In domain B, we had commercial clients with a hardcoded security labeling system. Openfire XMPP was used as the server. It is also open-source but without XEP-0258 support.

All the stanza modifications were performed in the XMPP server of domain A in a similar way as in the first test case. Rules were hardcoded with the knowledge that domain A clients have XEP-0258 support and domain B clients have a different system in use. Instant messaging was successful. In addition, we tested the functioning of MUC in a way that both servers had their own rooms. These room tests were also successful.

In the third test case we tested scenario 3 with three domains. In Domain C we used again Prosody as the XMPP server. Both IM and MUC tests between all domains were successful. There was no issues concerning where the MUC room was located.

Additionally, we tested a dynamic search of client software identities in an MUC room. The test was successful. In the future, the results of a such search will be used for an automatic stanza modification process as described earlier in section IV-B.

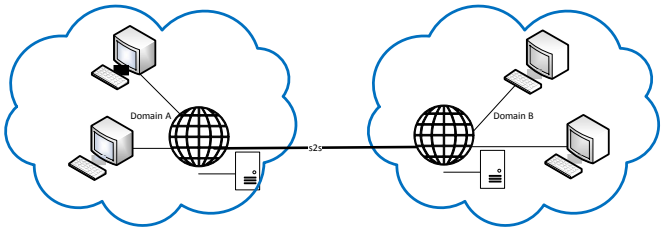

Fig. 9. Two domains.

Finally as a result we can mention that delay does not remarkably increase when using security label mangling between XEP-0258 and the company proprietary labels. This was measured in a test installation that was similar to that presented in Fig. 9. First, we tested a situation with two domains communicating with the same XEP-0258 mechanism. Label mangling was therefore not necessary. In the second test scenario, the label mangling was utilized from XEP-0258 to the company proprietary labels. Delay was measured between the Domain A server *in* and *out* ports. The results are presented in Fig. 10. The results show that labeling does not significantly increase the delay. For instance, the expected delay without the label mangling is 1.53 ms and with the mangling 1.58 ms.

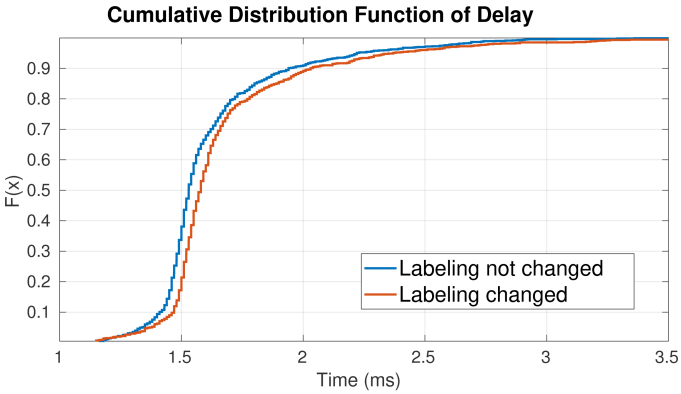

Fig. 10. The cumulative distribution function of delays.

V. Discussion

We found that a security label mangling system in a single domain with one XMPP server is quite easy to set up. In addition, it is possible to implement topologies with multiple XMPP domains and different types of server and client software. But as the number of combinations increases, the amount of required management work grows exponentially or the process is no longer possible.

The solution requires setup work for each different scenario. This limits the solution from being completely general. However, unconfigured XMPP domains may be kept unclassified; labeled data may only pass there if it is unclassified and all data received from there is considered unclassified.

Domain D can be thought to be also an Internet Relay Chat (IRC) [18] client that is connected to the XMPP network via a gateway. The procedure as stated earlier makes it possible for even the IRC client to use security labels. We have shown that this is technically possible, but it is a completely different question if the Mission Network instructions allow to use such approach.

In multiple domain missions, one solution to enable security labeling with XMPP servers, that do not directly support XEP-0258, is to use XMPP proxies on server-to-server (s2s) connections as presented in Fig. 11 [17]. In the proxy, all the stanza modifications can be done in one domain, that simplifies the XMPP security label mangling processes in the Mission

Network. Then there are no such limitations and no need for any guide book among participants, as mentioned earlier.

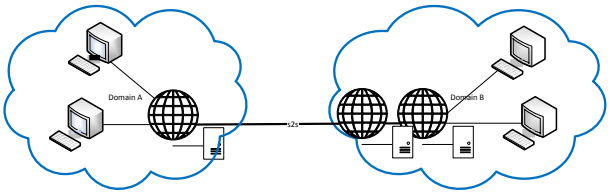

Fig. 11. XMPP proxy.

Another way of simplifying procedures is to use a hard-coded list that indicates whether a certain user is using a specific client software. In this case, we could just mangle the security labels according to participants. Such a hardcoding method can be replaced with a dynamic process, where the user identities of the MUC room are reviewed. A completely dynamic solution is not possible, since the label mangling server must always know initial information, such as the allowed client and server capabilities of the other end point.

VI. Future Work

Next we will try to implement a more dynamic and automatic version of a multiple domain as large as that discussed in Section IV-E, then we would have no need for hard-coded lists. The processes were described earlier could be verified on a large scale. Secondly, we will build and test proxies to mangle different security labels to the XEP-0258 notation on s2s connections as presented in Section V.

VII. Conclusion

In this paper we proposed processes for successfully implementing different security labeling mechanisms in XMPP both in static and in dynamic way in Instant Messaging and Multi-user Chat. We tested static processes and the tests showed that they work. With regards to the dynamic parts, we tested the basic idea, which was also successful. Although we have discussed and presented a mechanism to implement security label mangling system between different combinations, it does not decrease the criticality of the standardization. In the federated missions IEGs are integrated and the same labeling must be used, at least in the interoperability points. These processes are required if there are servers or clients in the Federated Network that utilize company proprietary labeling not in line with XEP-0258.

We noticed that a domain could co-operate with other XMPP domains without too much extra planning if either the server or the client has a support for XEP-0258 security labeling. As the number server–client combinations in the network grows, the complexity increases exponentially.

Finally, we proposed an XMPP proxy method to standardise security labels to XEP-0258 on s2s connections. This would ease the planning process of Mission Networks, since there would be no restrictions where, for example, MUC rooms can be located.

References

[1] R. Yan, Jie Yang, and A. Hauptmann. Automatically labeling video data using multi-class active learning. In *Proceedings Ninth IEEE International Conference on Computer Vision*, Oct 2003.

[2] Tijl De Bie, Thiago Turchetti Maia, and Antônio de Pádua Braga. Machine learning with labeled and unlabeled data. In *ESANN 2009, 17th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 22-24, 2009, Proceedings*, 2009.

[3] K. W. Kongsgård, N. A. Nordbotten, and S. Fauskanger. Policy-based labelling: A flexible framework for trusted data labelling. In *2015 International Conference on Military Communications and Information Systems (ICMCIS)*, May 2015.

[4] K. W. Kongsgård, N. A. Nordbotten, F. Mancini, R. Haakseth, and P. E. Engelstad. Data leakage prevention for secure cross-domain information exchange. *IEEE Communications Magazine*, 55(10), OCTOBER 2017.

[5] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Core. IETF RFC 3920, October 2015.

[6] P. Saint-Andre. Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence. IETF RFC 3921, October 2015.

[7] P. Saint-Andre. Mapping the Extensible Messaging and Presence Protocol (XMPP) to Common Presence and Instant Messaging (CPIM). IETF RFC 3922, October 2015.

[8] P. Saint-Andre. End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP). IETF RFC 3923, October 2015.

[9] XMPP - xmpp.org. [Online; accessed 2 June 2018].

[10] D. Schrmann F. Schmaus and V. Breitmoser. XEP-0373: OpenPGP for XMPP. Technical report, XMPP Standards Foundation, 2016.

[11] D. Schrmann F. Schmaus and V. Breitmoser. XEP-0374: OpenPGP for XMPP Instant Messaging. Technical report, XMPP Standards Foundation, 2016.

[12] A. Iivari, T. Väisänen, M. B. Alaya, T. Riipinen, and T. Monteil. Harnessing xmpp for machine-to-machine communications & pervasive applications. 10, 09 2014.

[13] S. Bendel, T. Springer, D. Schuster, A. Schill, R. Ackermann, and M. Ameling. A service infrastructure for the internet of things based on xmpp. In *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, March 2013.

[14] K. Zeilenga. XEP-0258: Security Labels in XMPP, April 2013.

[15] Paul E. Hoffman. Enhanced Security Services for S/MIME. RFC 2634, June 1999.

[16] R. Eatmon J. Hildebrand, P. Millard and P. Saint-Andre. XEP-0030: Service Discovery, October 2017.

[17] J. Järvinen, A. Marttinen, M. Luoma, M. Peuhkuri, and J. Manner. Architecture of xmpp proxy for server-to-server connections. In *2017 Military Communications and Information Systems Conference (MilCIS)*, Nov 2017.

[18] J. Oikarinen and D. Reed. Internet Relay Chat Protocol. Legacy RFC 1459, March 2013.