Evaluating Publish/Subscribe Standards for Situational Awareness using Realistic Radio Models and Emulated Testbed

Topic 9: Experimentation, Analysis, Assessment and Metrics
Paper 15

Authors

Frank T. Johnsen and Trude H. Bloebaum
Norwegian Defence Research Establishment (FFI)
Norway

Norman Jansen
Fraunhofer FKIE
Germany

Gerome Bovet
Armasuisse
Switzerland

Marco Manso
PARTICLE, Lda.
Portugal

Andrew Toth and Kevin S. Chan
CCDC Army Research Lab (ARL)
USA

Point of contact

Frank T. Johnsen
Norwegian Defence Research Establishment (FFI)
P.O. Box 25, 2027 Kjeller, Norway
e-mail: frank-trethan.johnsen@ffi.no

Abstract

There is currently an ongoing initiative to improve the interoperability between nations and other partners during common missions through Federated Mission Networking (FMN). So far, the focus of the standardization and profiling work done in FMN has mostly been on static and deployed networks, where networking resources are stable and plentiful. There is however also a need for interoperability at the tactical edge, between mobile units that have limited and often disrupted communications. In a previous study, we compared different protocols for subscription based distribution of information. We concluded that the WS-Notification standard, which is currently used in NATO, has a too large overhead in lower capacity tactical networks, and that for instance the Message Queuing Telemetry Transport (MQTT) protocol could be used instead.

In this paper, we expand upon those findings by investigating the applicability of MQTT in tactical networks further. Here, we address one of the main shortcomings in the testbed used in our previous experiments by adding in new and more realistic radio models, which allow us to better assess the performance of MQTT in the tactical domain. Furthermore, we also expand our experiments evaluating MQTT for sensor networks (MQTT-SN) as well. The reason for adding MQTT-SN to the experiments is that this protocol is based on UDP rather than TCP.

This work has been performed in the context of the NATO STO/IST-150 «NATO Core Services profiling for Hybrid Tactical Networks» working group.

1 Introduction

There is currently an ongoing initiative to improve the interoperability between nations and other partners during common missions through Federated Mission Networking (FMN). The goal of this initiative is to enable so-called zero-day interoperability by establishing an increasingly mature framework for mission interoperability ahead of time. This framework includes all aspects of establishing a mission network, such as governance, procedures and also standardized technical services.

So far, the focus of the standardization and profiling work done in FMN has mostly been on static and deployed networks, where networking resources are stable and plentiful. Current directions of military operations are trending towards pushing decision making and collaboration at the tactical edge. Operations at the tactical edge are significantly different from the enterprise networked environment. The networks that support these tactical edge operations are often characterized as a disconnected intermittent connectivity and limited bandwidth (DIL) environment, or more recently a congested, contested operational environment. Current approaches within FMN are relevant for, and validated on, enterprise (perhaps wired) networks, but may not be applicable in environments with the aforementioned challenges present in the tactical domain.

Despite these challenges in the networking environment, operations must occur at much faster timescales and deal with increased uncertainty of information and operations, and as a collaborative effort between partners. There is thus a need for interoperability at the tactical edge, between mobile units that have limited and often disrupted communications. When there is a need of many-to-many information exchange based on the relevance of, or interest for, a given type of information, the subscription-based information exchange is a pattern that is well known also in these types of environments. In our previous study [4], we compared different protocols for subscription based distribution of information between a number of nodes. We concluded that the WS-Notification (WS-N) standard [9], which is currently used in NATO, has a too large overhead in lower capacity tactical networks, and that for instance the Message Queuing Telemetry Transport (MQTT) [10] protocol could be used instead.

In this paper, we expand upon those findings by investigating the applicability of MQTT in tactical networks further. Here, we address one of the main shortcomings in the testbed used in our previous experiments by adding in new and more realistic radio models, which allow us to better assess the performance of MQTT in the tactical domain. Furthermore, we also expand our experiments evaluating MQTT for sensor networks (MQTT-SN) as well. The reason for adding MQTT-SN to the experiments is that this protocol is based on UDP rather than TCP.

One can expect a variety of tactical services relevant to operations in this environment. For example, position location information is usually invoked as the primary shared situation awareness requirement in most operations. In this paper, we have considered Blue Force Tracking (BFT) as a representative service. We do note that other services such as sharing of video or imagery may demand more resources than typically available for these networks. One standing challenge is the optimization of multiple networked services for resource-constrained networks in these operational environments.

This work has been performed in the context of the NATO STO/IST-150 «NATO Core Services profiling for Hybrid Tactical Networks» research task group.

2 Testbed

Measuring the performance of a single BFT service in a lab environment will not indicate how multiple instances of the BFT service deployed together with tactical radio systems in military vehicles will perform in a realistic military scenario. This is the case, because typical lab experiments do not take the dynamic environment into account and are poorly scalable.

Instead, a whole combination of different systems (IT and communications systems) has to be taken into account. For the systems under test – i.e. BFT services – the original software (or virtualized versions) should be run in order to represent the real systems in as much detail as possible. Systems which cannot be virtualized, because the software is not publicly available (e.g., military radios), will be emulated by means of real-time radio simulators with realistic radio models (see Section 3).

We use a subset of the *Anglova scenario* [5] for our experiments. Specifically, we model a mechanized battalion with 24 military vehicles coordinated by the Coalition HQ. The battalion nodes are equipped with tactical radios that are used to exchange information. To drive the network emulation, we employ the Extendable Mobile Ad-hoc Network Emulator (EMANE) [11], which provides radio link emulation, signal propagation and mobility representation to the experiment. Advantages of this testbed approach are scalability and (to some degree) repeatability. Consider that the behavior of the applications may not be completely deterministic, since real software is running in real-time.

3 New radio models

During the first experiments with EMANE leveraging the standard Wi-Fi models used by the community, we noticed that the obtained results were not matching the performance of real tactical radios [1]. The Optimized Link State Routing (OLSR) routing tables as well as some performance metrics, such as throughput and latency between emulated nodes led us to the two following conclusions:

(1) The Wi-Fi models, although tunable, do not allow reproducing the latencies and throughput of real tactical radios. The obtained performance during the emulations is far too optimistic compared to the expected performance in a real deployment.
(2) The Anglova Vignette 2 with Company 1 scenario (24 nodes) is not challenging enough, as most of the time the topology tends to be a full-mesh, whereas multi-hop topologies would rather be more realistic.

The combination of these two drawbacks leads to the situation where experiments do not reflect reality, as even heavy protocols, which were not working under lab conditions with real radios, show high performance in the emulated environment. In order to obtain more realistic emulations, we started by reproducing Narrowband and Wideband tactical radios in EMANE. Their performance (throughput and latency) was measured under lab conditions with various Received Signal Strength Indicators (RSSIs). In a second step, and with the information in our possession regarding the Time-division multiple access (TDMA) schedules, we elaborated TDMA scheduling models in EMANE. As shown in [1], we were able to reproduce in quite high fidelity the performance of the real radios, including the adaptive rate changing the performance according to the channel quality.

As previously mentioned, the 24 nodes we used from the Anglova scenario do not produce a challenging network topology. This is due to the rather short distances between the nodes throughout the scenario. The emulated vehicles move in the form of clusters, which leads to the situation where full connectivity is achieved with only one-hop during most of the emulation. Such conditions are not challenging in terms of multi-hop topologies where performance is relative to the number of hops. We therefore adapted the Anglova scenario in order to generate more hops between the nodes [2]. This was achieved by decreasing the emulated output power to 5W (37dBm), which is often a tactical choice allowing lowering the possibility getting spotted by an enemy. Additionally, the locations of selected nodes were changed, so that during certain phases of the scenario, the topology also contains some chains. The average number of hops increased from 1.5 to around 2.5, whereas the maximum number of hops increased from 4 to 7. In this paper, we refer to this version as “Modified Anglova” and the original as “Anglova scenario”. We perform experiments with both versions of the scenario using the Wideband TDMA scheme developed by Switzerland.

4 Test applications and software

In our experiments we use the NATO Friendly Force Information (NFFI) data format in our BFT services. The reason for choosing the NFFI data format (described in draft STANAG 5527) is that it has been used with great success in many contexts, after it originally emerged to support interoperable friendly force tracking in the Afghan Mission Network. We consider it a good example of a representative standard payload for our experiment. The dissemination mechanisms we use are

WS-N, MQTT, and MQTT-SN, respectively. Each of these three standards provide the functionality necessary to distribute information from a provider to the interested consumers. It should be noted that WS-N consists of three standards; WS-BaseNotification, WS-BrokeredNotification and WS-Topics. For the work in this paper we use WS-Notification including the broker functionality described by WS-BrokeredNotificaton [9]. The BFT services were implemented by the Norwegian Defence Research Establishment (FFI).

WS-N is a part of the family of SOAP Web services standards. SOAP services promote interoperability, but the cost is increased overhead. Hence, it is not necessarily well suited for use in tactical networks where network capacity typically is low. As a consequence, we investigate two other publish/subscribe industry standards that can possibly provide the same functionality as WS-N, but with less overhead. Previously, we have compared WS-N with MQTT, and found MQTT to be more efficient [4]. In this paper, we continue our experiments using the above mentioned radio models, as well as adding on the UDP-based counterpart to MQTT, namely MQTT-SN.

We used a closed-source implementation of WS-N developed in-house at FFI. However, this implementation has been tested for interoperability at the NATO Coalition Warrior Interoperability eXercise (CWIX) in 2014, where it was shown that the functions used (subscribing to a topic, publishing to a topic, and notifying the subscribers of new data) in our experiments were indeed compliant with the standard [6].

For MQTT we used the open source VerneMQ broker which is freely available [7]. MQTT-SN is usually not supported natively by existing brokers, so we added MQTT-SN support to VerneMQ by installing the free, open source gateway solution from the Eclipse Paho project [8]. It should be noted that since MQTT-SN has to be offered via a gateway, this may negatively impact the performance of the protocol as opposed to if it were offered as a complete stand-alone solution.

5 Experiment execution

Overview of experiments:

Experiment series / Protocol	WS-Notification	MQTT	MQTT-SN
Anglova scenario, Swiss TDMA	Wideband radio	Wideband radio	Wideband radio
Modified Anglova, Swiss TDMA	Wideband radio	Wideband radio	Wideband radio

The experimental testbed used to conduct experiments is the Network Science Research Laboratory (NSRL) [12] established by the CCDC Army Research Laboratory (ARL). The NSRL provides network emulation capabilities and military relevant data and scenarios for the testing and evaluation of various networking oriented technologies and approaches. The facility has enabled collaboration between ARL researchers and those from other organizations. Additionally, infrastructure in the way of dynamic virtualization has been developed to assist in the execution of experiments in the NSRL. To enable repeatability and scalability of experimentation, ARL has also developed a platform called Dynamically Allocated Virtual Clustering Management System (DAVC). DAVC provides the capability to dynamically create and deploy virtual clusters of heterogeneous nodes as specified by Virtual Machines (VMs).

Experiments are completely reconfigurable through the DAVC interface, with minor modifications to parameters defined in custom scripts (e.g., nodes' location and radio signal path loss between nodes, as provided by Anglova).

Both the Anglova scenario and DAVC are releasable through NATO collaboration.

The Anglova scenario, incorporating WS-N, MQTT, and MQTT-SN broker messaging services, was setup in the NSRL environment. For that, WS-N and MQTT services were installed onto the VM template of the Anglova scenario to enable the publish/subscribe position location information services. The experiments use a **single broker topology**. The VM template is deployed to nodes during runtime of the scenario. This is illustrated in Figure 1.

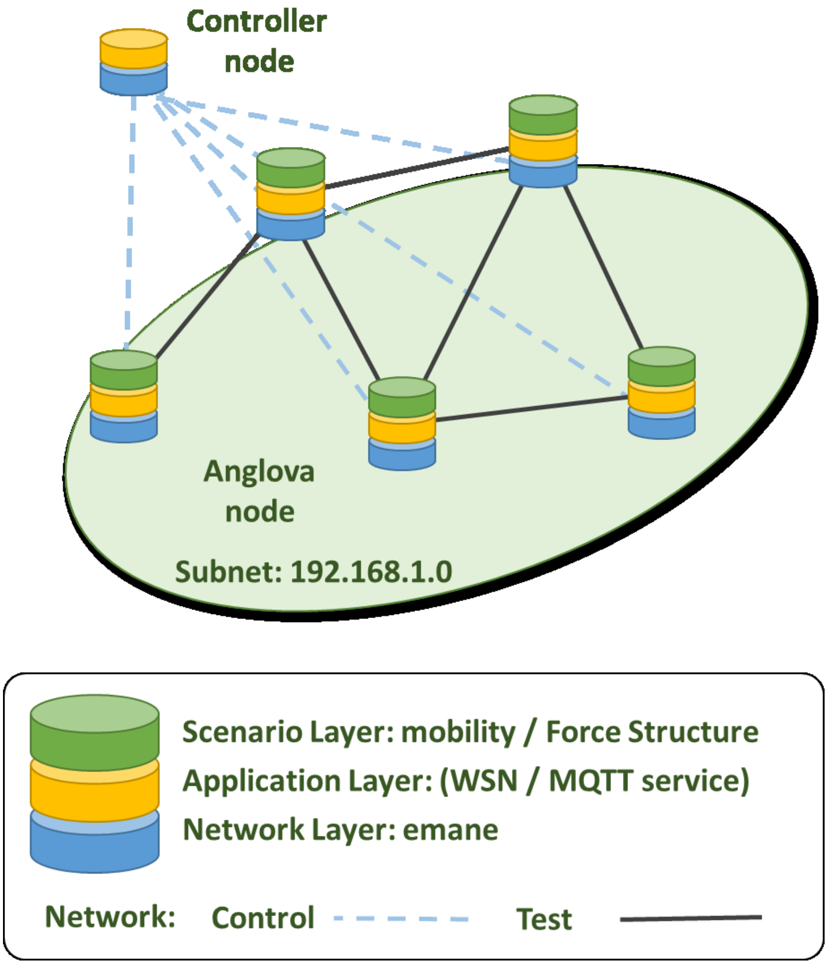

Figure 1: Architecture of network experiment including network emulation, application and scenario layers

For network emulation, we use the EMANE that provides – besides the emulation of the radio links – signal propagation and mobility representation to the experiment to create a more realistic environment. The mobility information was drawn from Anglova recorded data.

The emulation allows for various types of routing and radio models to be used; in this scenario we use Optimized Link State Routing (OLSR) [13] (OLSR 2016) V2 via the olsrd daemon on each virtual machine representing a node in the scenario with wireless links based on the Swiss TDMA Wideband model. The TDMA model was configured to emulate wideband tactical radios operating at 300 MHz with a 250 KHz bandwidth and 1 Mbit/s data rate. The TDMA model was configured with 8 frames with 24 slots, a slot overhead of 3 us, and slot duration of 5000 us. OLSR V2 was configured with a Hello Interval of 2 seconds, Hello Validity Time of 20 seconds, Topology Control Interval of 8 seconds, and Topology Control Validity time of 80 seconds.

In the initial set of experiments, we ran 20 minutes of the Anglova scenario vignette excerpt consisting of 24 nodes. The publishers on nodes 2 through 24, which sent node locations (i.e, NFFI messages) every 10 seconds, were started at the 1-minute mark, and stopped after 20 minutes at minute 21 in the scenario. We set up a DAVC cluster of 24 "Anglova" nodes and one controller node. The controller node is used as the orchestration node and is not represented in the experiment nor does it take part in the scenario. Node 1 for this experiment is arbitrarily established as the broker node (i.e., runs the WS-N, VerneMQ Broker, or VerneMQ broker with the MQTT-SN Gateway). It also has a subscriber service running on it (i.e., subscribes to and receives messages from all publishers). We note that the platform allows for any configuration of broker and subscriber services.

Additionally, to facilitate the execution of these experiments, we have created services that launch EMANE and the Anglova configuration. We also have Linux shell scripts that can start and stop the publisher services for both WS-N and MQTT as well as gathering generated pcap and log files used for analysis.

6 Analysis

The experiments described in this paper aim to test the performance of several different ways to distribute the information from BFT services (the system under test) in a realistic setup with emulated radio communications systems according to a realistic military scenario (Anglova). Two different scenario setups were used for the experiments. The first one uses the original Anglova scenario, but with the TDMA model described in Section 3 “New radio models” above. The second one also uses this TDMA model and additionally all other adaptations described in Section 3. These include decreasing the emulated output power to 5W and changing the positions of some of the units to generate more transmission hops. Thus, this second version of the scenario is even more challenging than the first one.

For the BFT service different protocol standards (WS-N, MQTT and MQTT-SN) have been evaluated.

For the analysis of the experiments, we used analyzing tools from the *Analyze and Test environment* (*AuT*) project of Fraunhofer FKIE, Germany. In [3], concepts and tools for analyzing complex military experiments in a virtualized testbed are described. These include a concept for capturing and processing monitoring data from C2IS applications used in distributed tactical networks, the specification of suitable metrics for military applications and the definition of different visualizations based on these metrics.

Our evaluation approach makes use of monitoring data from both the network layer as well as the application layer. For the network layer, the network traffic was logged via publicly available network logging tools (tcpdump). For the application layer, the application traffic was logged by the application service itself at different measuring points (e.g., after a message was received, after a message was processed by the application, etc.). This has been done via a logging interface which we defined by a JSON schema. For this purpose, we implemented the logging interface into the publisher and subscriber services. The JSON logs and tcpdumps are used to calculate packet and message losses.

6.1 WS-N with Anglova scenario

In this setup, we deploy a closed-source WS-N broker together with one WS-N subscriber on Node 1. Nodes 2 to 24 (23 nodes in total) each run a WS-N producer software publishing a NFFI message every 10 seconds. The measurements pertaining to network and application layers are presented next.

Network layer

By analyzing the network level log files (packet captures) the data volume produced by the WS-N could be obtained (see Table 1). This data volume contains all data from the different transmission layers (Ethernet, IP, TCP, HTTP). The WS-N-based communication produces 40 kbit of data per second. The message size of a WS-N message was 1863. The network logs show that there were 2468 TCP Duplicate Acknowledgements and that 2761 TCP retransmissions produced. 1761 of them were of the type «spurious»[1]. This problem often arises when using TCP in networks with a high bandwidth-delay product.

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
40 kbit/s	1863 bytes	2468	1761	2761

Table 1: Results from Experiments for WS-N, Anglova scenario (network layer)

Application layer

The application logs consist of logging entries of the senders (publishers) of NFFI messages and logging entries of the receiver (subscriber) of these messages. This approach allows us to calculate the overall transmission times of NFFI messages, which represent the age of the positions as observed by the user at the receiver node. The results were analyzed with help of analyzing tools of the AuT project and are shown in Figure 2. Figure 2 shows as a boxplot diagram the transmission times of all publishers. Note that the nodes in the Anglova scenario are named by numbers from 100 to 1450, whereas the nodes in the modified Anglova scenario (cf. Section 6.4 below) are renamed to 1, 2, ..., 24.

[1] Here, «spurious» means that a packet was unnecessarily retransmitted, because the respective acknowledgement arrived too late at the sender. Since the congestion control mechanism of TCP interprets «lost» (actually belated in this case) acknowledgements as buffer overflows, the congestion window is unnecessarily decreased, which leads to a reduced throughput.

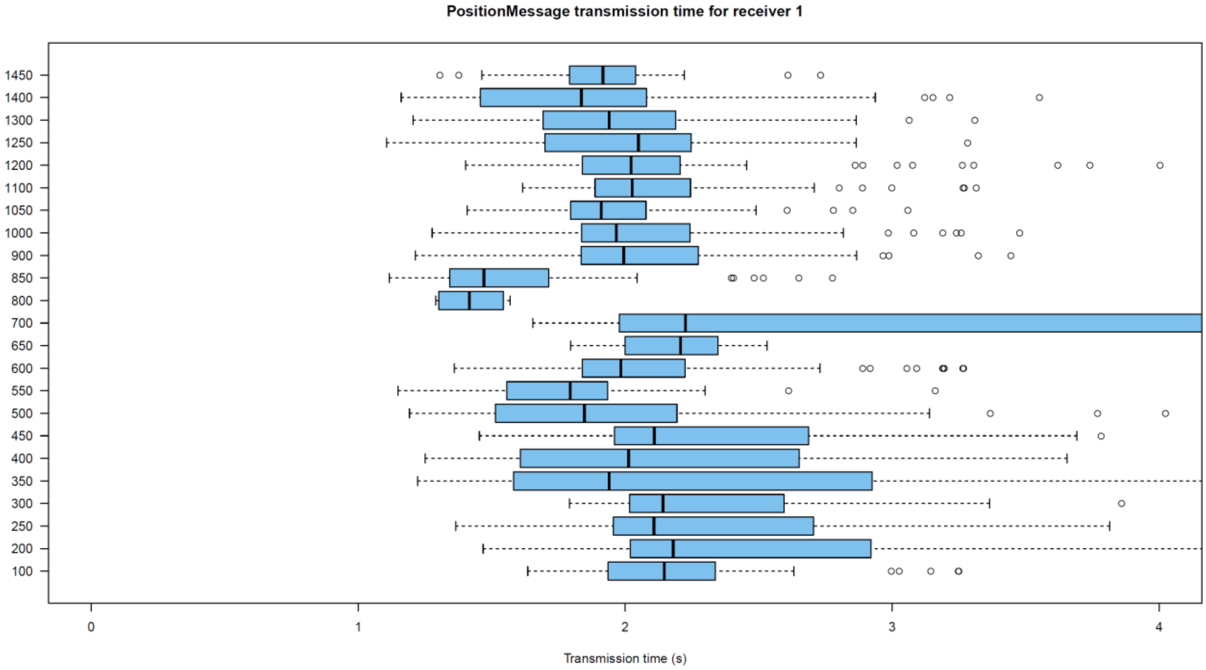

Figure 2: Transmission times of WS-N-based NFFI messages

In total 1697 messages were published, of which 22 (1.30%) were lost (cf. Table 2). The overall median of the transmission delay was 1.97 s (averaged over all messages from all publishers). The minimum delay was 1.11 s and the maximum delay was 86.78 s. As you can see in Figure 2, most messages were near the median delay, but there are higher values for some publishers, who have suffered from a poor connection to the other nodes at some time in the scenario.

messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
1697	22 (1.30 %)	1.11 s	1.97 s	210 s

Table 2: Results from Experiments for WS-N, Anglova scenario (application layer)

6.2 MQTT with Anglova scenario

In this setup, we deploy the VerneMQ broker together with one MQTT subscriber on Node 1. Nodes 2 to 24 (23 nodes in total) each run an instance of the MQTT producer software publishing a NFFI message every 10 seconds. For the MQTT publisher the Quality of Service class *QoS0*[2] was used. The measurements pertaining to network and application layers are presented next.

Network layer

The analysis of the network level log files (packet captures) results in the data shown in Table 3. The table shows that the MQTT-based traffic produced 31 kbit/s of data volume. The size (content) of each message was 880 Bytes (WS-N's size increase was due to extra overhead from using SOAP and XML). The network logs show that there were 1922 TCP Duplicate Acknowledgements. Furthermore, 4281 TCP retransmissions were produced, 1436 of them were of type «spurious» similar to the setup with WS-N.

[2] QoS0 gives *at most once* delivery semantics, whereas QoS1 gives *at least once* delivery semantics.

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
31 kbit/s	880 bytes	1922	1436	4281

Table 3: Results from Experiments for MQTT, Anglova scenario (network layer)

Application layer

In Figure 3 the average transmission times of the messages are shown for each publisher in a boxplot diagram.

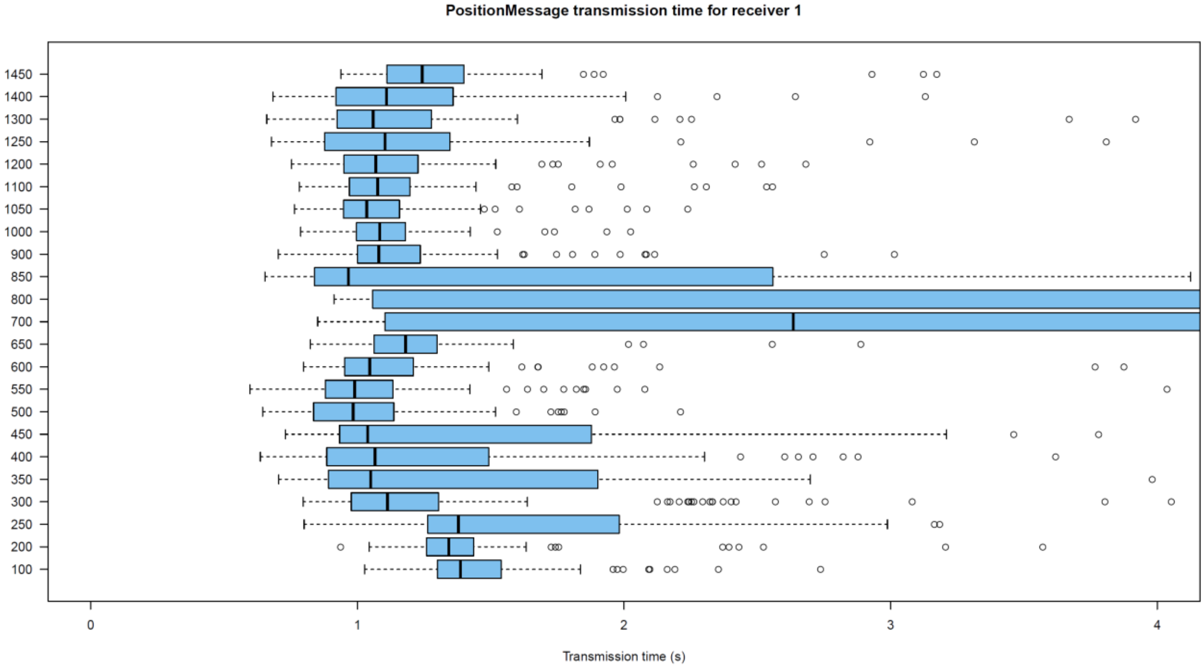

Figure 3: Transmission times of MQTT-based NFFI messages (whole diagram)

In total 2553 messages were published, from which 383 (15 %) were lost (see Table 4). The overall median of the transmission delay was 1.46 s (averaged over all messages from all publishers). The minimum delay was 0.60 s and the maximum delay was 225 s.

messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
2553	383 (15%)	0.60 s	1.46 s	225 s

Table 4: Results from Experiments for MQTT, Anglova scenario (application layer)

6.3 MQTT-SN with Anglova scenario

In this setup, we deploy the VerneMQ broker in conjunction with the open source MQTT-SN gateway solution from the Eclipse Paho project. Furthermore, one MQTT-SN subscriber on Node 1 is deployed. Nodes 2 to 24 (23 nodes in total) each run an instance of the MQTT-SN producer software publishing a NFFI message every 10 seconds. For the MQTT publisher the Quality of Service class *QoS0* was used. The measurements pertaining to network and application layers are presented next.

Network layer

The analysis of the network level log files (packet captures) results in the data shown in Table 5. The MQTT-SN-based traffic produced 13 kbit/s of data volume. The size (content) of each message was 894 bytes and thus similar as the message size of MQTT. Since MQTT-SN uses UDP, there are no TCP retransmissions.

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
13 kbit/s	894 bytes	NA	NA	NA

Table 5: Results from Experiments for MQTT-SN, Anglova scenario (network layer)

Application layer

In Figure 4 the average transmission times of the messages are shown for each publisher in a boxplot diagram.

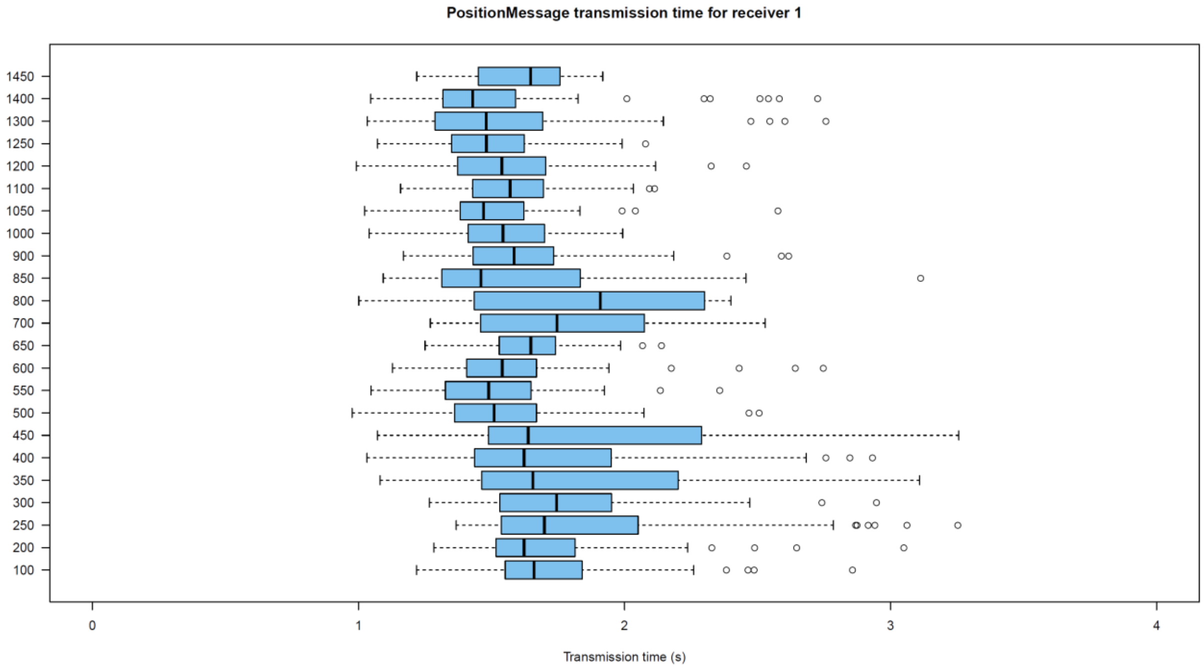

Figure 4: Transmission times of MQTT-based NFFI messages (whole diagram)

In total 2075 messages were published, of which 360 (17.35%) were lost (see Table 6). The overall median of the transmission delay was 1.60 s (averaged over all messages from all publishers). The minimum delay was 0.98 s and the maximum delay was 3.26 s. As you can see in Figure 4, most messages were near the median delay. In contrast to WS-N and MQTT, there are no high values for some publishers. This means that MQTT-SN (which is based on UDP) drops these messages at some point, while WS-N and MQTT still try to deliver them after more than 200 seconds.

messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
2075	360 (17.35%)	0.98 s	1.60 s	3.26 s

Table 6: Results from Experiments for MQTT-SN, Anglova scenario (application layer)

6.4 WS-N with Modified Anglova scenario

In this setup, we deploy the same services as in Section 6.1 (WS-N broker, one WS-N subscriber on Node 1, WS-N producer software on Nodes 2 to 24). The measurements pertaining to network and application layers are presented next.

Network layer

Table 7 shows the results from the network analysis. The WS-N-based communication produced a data volume of 39 kbit/s. The message size was 1863 bytes as in Section 6.1. The logs show that 2652 duplicate acknowledgments were produced and that 2741 TCP retransmissions were caused, from which 1821 were of type «spurious».

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
39 kbit/s	1863 bytes	2652	1821	2741

Table 7: Results from Experiments for WS-N, Modified Anglova scenario (network layer)

Application layer

In Figure 5 the average transmission times of the messages are shown for each publisher in a boxplot diagram.

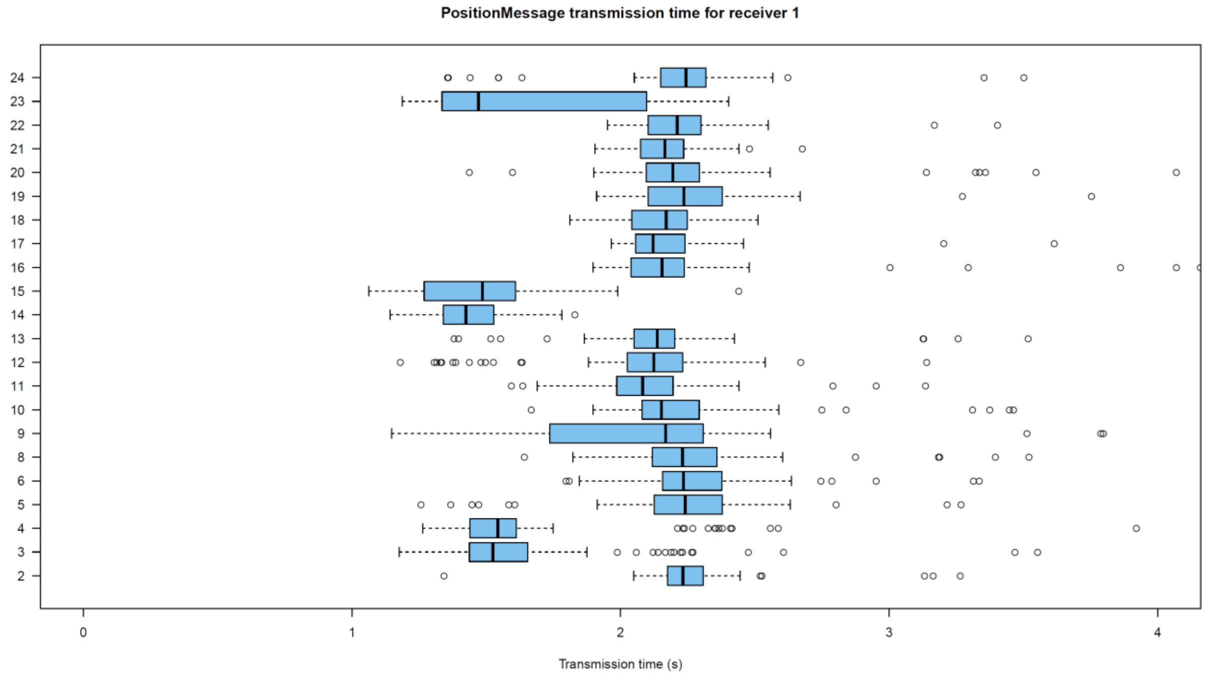

Figure 5: Transmission times of WS-N-based NFFI messages

In total 1725 messages were published, of which 22 (1.28 %) were lost (see Table 8). The overall median of the transmission delay was 2.02 s (averaged over all messages from all publishers). The minimum delay was 1.06 s and the maximum delay was 2.67 s. This means all messages were near the median delay.

messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
1725	22 (1.28%)	1.06 s	2.02 s	79 s

Table 8: Results from Experiments for WS-N, Modified Anglova scenario (application layer)

6.5 MQTT with Modified Anglova scenario

In this setup, we deploy the same services as in Section 6.2 (VerneMQ broker, one MQTT subscriber on Node 1, MQTT producer software on Nodes 2 to 24). For the MQTT publisher the Quality of Service class *QoS0* was used. The measurements pertaining to network and application layers are presented next.

Network layer

The results from the network analysis are shown in Table 9. The MQTT -based communication produced a data volume of 33 kbit/s. The message size was 880 bytes as in Section 6.2. The logs show that 2336 duplicate acknowledgments were produced and that 5091 TCP retransmissions were caused, from which 1999 were of type «spurious».

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
33 kbit/s	880 bytes	2336	1999	5091

Table 9: Results from Experiments for MQTT, Modified Anglova scenario (network layer)

Application layer

In Figure 6 the average transmission times of the messages are shown for each publisher in a boxplot diagram.

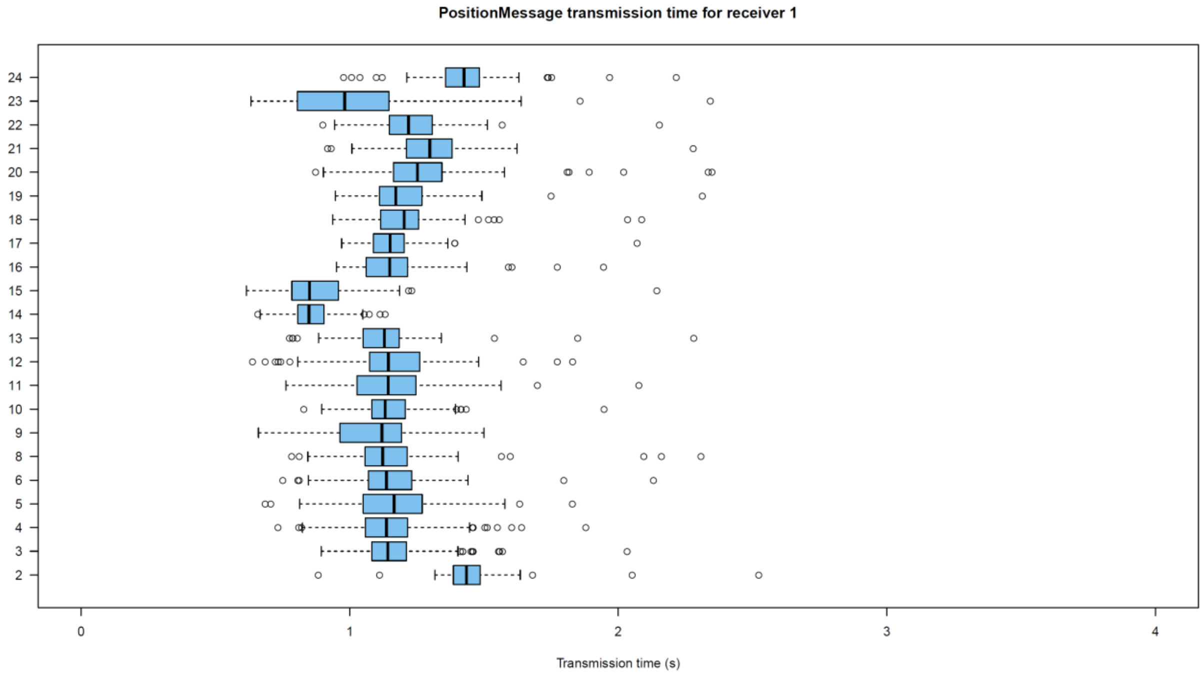

Figure 6: Transmission times of MQTT-based NFFI messages

In total 2490 messages were published, from which 367 (14.74 %) were lost (see Table 10). The overall median of the transmission delay was 1.15 s (averaged over all messages from all publishers). The minimum delay was 0.62 s and the maximum delay was 1.64 s. This means all messages were near the median delay.

messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
2490	367 (14.74%)	0.62 s	1.15 s	6.8 s

Table 10: Results from Experiments for MQTT, Modified adapted Anglova scenario (application layer)

6.6 MQTT-SN with Modified Anglova scenario

In this setup, we deploy the same services as in Section 6.3 (VerneMQ broker, MQTT-SN gateway, one MQTT subscriber on Node 1, MQTT-SN producer software on Nodes 2 to 24). For the MQTT publisher the Quality of Service class *QoS0* was used. The measurements pertaining to network and application layers are presented next.

Network layer

The results from the network analysis are shown in Table 11. The MQTT-SN-based communication produced a data volume of 14 kbit/s. The message size was 894 bytes as in Section 6.3. Since MQTT-SN is UDP-based, there are no TCP retransmissions.

data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retransmissions	TCP Retransmissions
14 kbit/s	894 bytes	NA	NA	NA

Table 11: Results from Experiments for MQTT-SN, Modified Anglova scenario (network layer)

Application layer

In Figure 7 the average transmission times of the messages are shown for each publisher in a boxplot diagram.

In total 2093 messages were published, of which 354 (16.91 %) were lost (see Table 12). The overall median of the transmission delay was 1.60 s (averaged over all messages from all publishers). The minimum delay was 0.90 s and the maximum delay was 2.17 s. This means all messages were near the median delay.

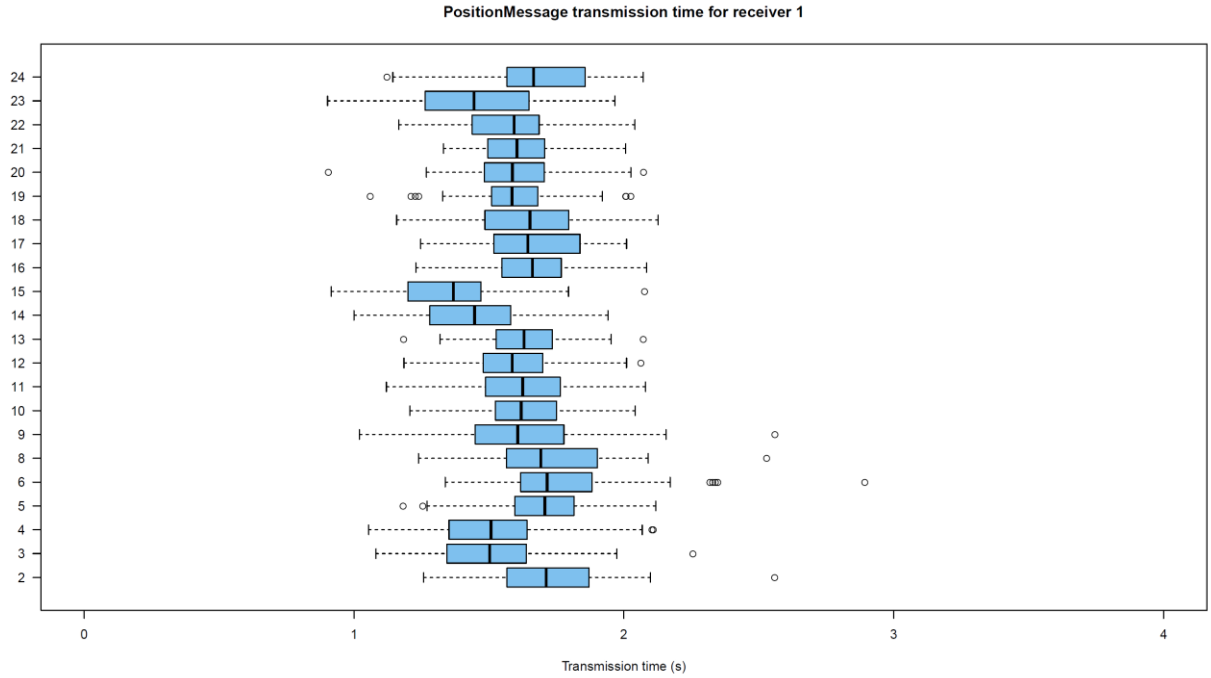

Figure 7: Transmission times of MQTT-based NFFI messages (whole diagram)

messages sent	**messages lost**	**delay (min)**	**delay (overall median)**	**delay (max)**
2093	354 (16.91%)	0.90 s	1.60 s	12 s

Table 12: Results from Experiments for MQTT-SN, Modified Anglova scenario (application layer)

6.7 Comparing Quality of Service settings in MQTT/MQTT-SN

In this setup, we will compare two Quality of Service settings in MQTT and MQTT-SN. For this purpose the MQTT publisher software was updated to use QoS1. The MQTT-related experiments above were conducted with QoS0. Besides this change of the publisher software, the same software as described above was used (VerneMQ broker, MQTT-SN gateway, one MQTT subscriber on Node 1, MQTT-SN producer software on Nodes 2 to 24). The measurements were conducted in the Modified Anglova scenario and are presented next.

Table 13 shows the results from network analysis. Results from the experiments with QoS0 (cf. Sections 6.2, 6.3, 6.5, and 6.6) are also listed in the Table for comparison reasons.

Experiment	**data volume per second**	**message size**	**TCP Duplicate ACK**	**TCP Spurious Retrans-missions**	**TCP Retrans-missions**
MQTT, QoS0, Modified Anglova	33 kbit/s	880 bytes	2336	1999	5091
MQTT-SN, QoS0, Modified Anglova	14 kbit/s	894 bytes	NA	NA	NA
MQTT, QoS1, Modified Anglova	38 kbit/s	893 bytes	3255	1981	10565
MQTT-SN, QoS1, Modified Anglova	13 kbit/s	910 bytes	NA	NA	NA

Table 13: Comparison of MQTT, MQTT-SN for QoS0 and QoS1 (network layer)

It can be seen from the table that MQTT produces about double the number of retransmission when used in reliable mode (QoS1). The produced data volume increased from 33 kbit/s to 38 kbit/s for MQTT with QoS1. The data for MQTT-SN remains the same when QoS1 was used.

In Figures 8 and 9, the average transmission times of the messages are shown for MQTT (QoS1) and MQTT-SN (QoS1) using the Modified Anglova scenario.

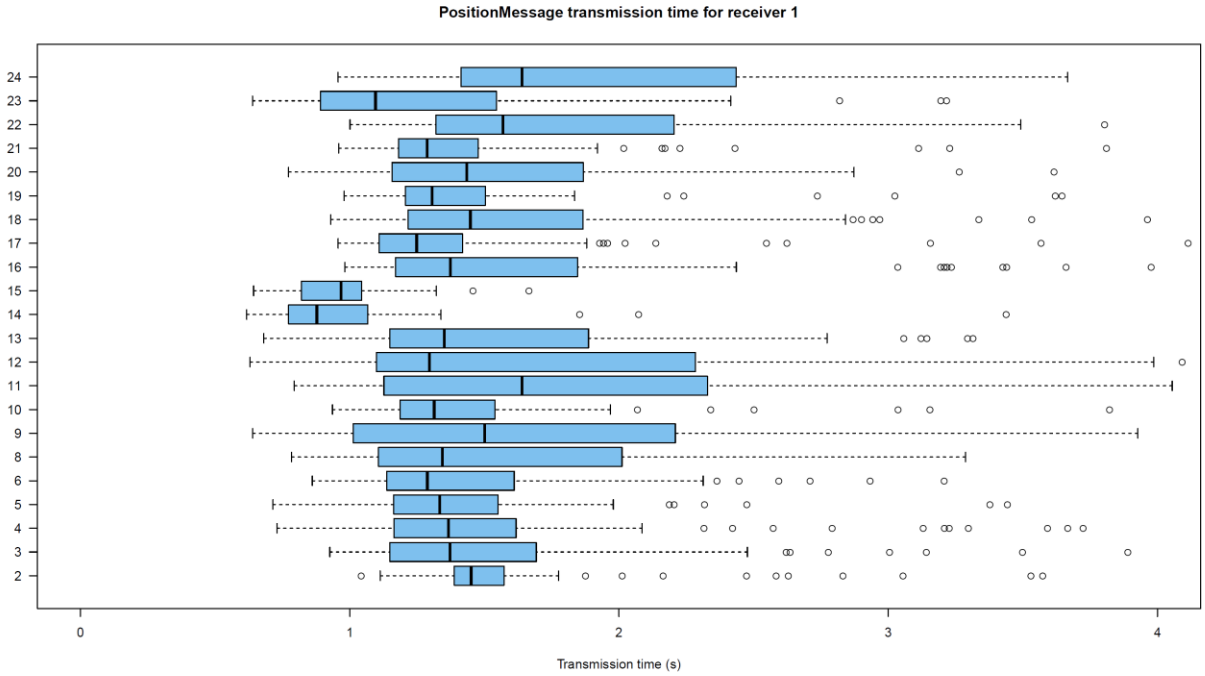

Figure 8: Transmission times of MQTT-based NFFI messages, QoS1, Modified Anglova

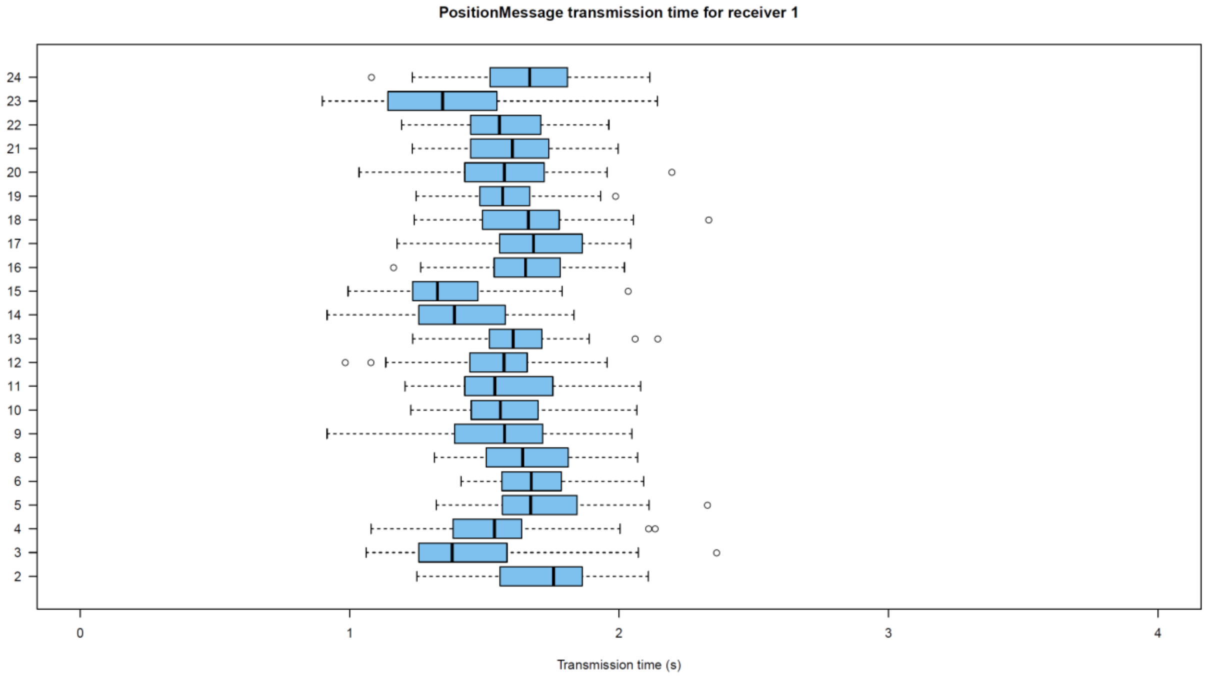

Figure 9: Transmission times of MQTT-SN-based NFFI messages, QoS1, Modified Anglova

An overview of the results is shown in Table 14. The results from the experiments with QoS0 (cf. Sections 6.2, 6.3, 6.5, and 6.6) are also listed in the table for comparison reasons.

Experiment	**messages sent**	**messages lost**	**delay (min)**	**delay (overall median)**	**delay (max)**
MQTT, QoS0, Modified Anglova	2490	367 (14.74%)	0.62 s	1.15 s	6.8 s
MQTT-SN, QoS0, Modified Anglova	2093	354 (16.91%)	0.90 s	1.60 s	12 s
MQTT, QoS1, Modified Anglova	2399	327 (13.63%)	0.62 s	1.34 s	62.43 s
MQTT-SN, QoS1, Modified Anglova	1663	8 (0.48%)	0.90 s	1.57 s	11.67 s

Table 14: Comparison of MQTT and MQTT-SN for QoS0 and QoS1 (application layer)

The results show that most values remain the same when MQTT or MQTT-SN are used with QoS1 instead of QoS0 (e.g. the transmission times). But the reliability improves significantly for MQTT-SN when QoS1 is used.

6.8 Comparison Analysis and Results

A comparison between results obtained with WS-N, MQTT and MQTT-SN in the two scenarios is presented next. The combined measurement results from Sections 6.1 - Section 6.7 used to support our analysis are presented in Tables 15 and 16.

Experiment	data volume per second	message size	TCP Duplicate ACK	TCP Spurious Retrans-missions	TCP Retrans-missions
Anglova, WS-N	40 kbit/s	1863 bytes	2468	1761	2761
Anglova, MQTT, QoS0	31 kbit/s	880 bytes	1922	1436	4281
Anglova, MQTT-SN, QoS0	13 kbit/s	894 bytes	NA	NA	NA
Modified Anglova, WS-N	39 kbit/s	1863 bytes	2652	1821	2741
Modified Anglova, MQTT, QoS0	33 kbit/s	880 bytes	2336	1999	5091
Modified Anglova, MQTT-SN, QoS0	14 kbit/s	894 bytes	NA	NA	NA
Modified Anglova, MQTT, QoS1	38 kbit/s	893 bytes	3255	1981	10565
Modified Anglova, MQTT-SN, QoS1	13 kbit/s	910 bytes	NA	NA	NA

Table 15: Overview of the Results from Experiments (network layer)

Experiment	messages sent	messages lost	delay (min)	delay (overall median)	delay (max)
Anglova, WS-N	1697	22 (1.30 %)	1.11 s	1.97 s	210 s
Anglova, MQTT, QoS0	2553	383 (15%)	0,60 s	1,46 s	225 s
Anglova, MQTT-SN, QoS0	2075	360 (17.35%)	0.98 s	1.60 s	3.26 s
Modified Anglova, WS-N	1725	22 (1.28%)	1.06 s	2.02 s	79 s
Modified Anglova, MQTT, QoS0	2490	367 (14.74%)	0.62 s	1.15 s	6.8 s
Modified Anglova, MQTT-SN, QoS0	2093	354 (16.91%)	0.90 s	1.60 s	12 s
Modified Anglova, MQTT, QoS1	2399	327 (13.63%)	0.62 s	1.34 s	62.43 s
Modified Anglova, MQTT-SN, QoS1	1663	8 (0.48%)	0.90 s	1.57 s	11.67 s

Table 16: Overview of the Results from Experiments (application layer)

From the evaluation of the experiments, it can be seen that:

- In overall (including the whole communications stack) MQTT-SN produces a data volume of about 13-14 kbit/s compared to about 31-38 kbit/s (MQTT) and about 39-40 kbit/s (WS-N).
- The message sizes of MQTT and MQTT-SN (about 900 bytes) are about half the size of WS-N (about 1850 bytes).
- MQTT caused notably more TCP retransmissions than WS-N, which is in contrast to former experiments with WiFi links, where the opposite could be observed. The causes for the high number of retransmissions has to be further analyzed in the future. When QoS1 was used with MQTT there were even much more retransmissions (double the size of QoS0).
- WS-N has less message losses (1.3 %) compared to MQTT (15 %) and MQTT-SN (17 %) if these are run with QoS0. The use of QoS1 with MQTT doesn’t increase the reliability significantly (still 14 %). But for MQTT-SN the reliability improves significantly by using QoS1 (packet loss 0.48 % vs. 17 %).
- The average delay is higher for WS-N than for MQTT or MQTT-SN. MQTT has the lowest delay.
- The transmission results of MQTT-SN do not contain messages with a very high delay (see e.g. Anglova, MQTT with 225 s). But the loss rate is slightly higher than MQTT when used with QoS0. It seems like MQTT-SN discards messages if the transmission takes too long. The use of QoS1 with MQTT-SN improves the reliability significantly (0.48% message loss) and thus leads to an even higher reliability than WS-N.

In summary, it could be seen that WS-N, MQTT and MQTT-SN behaved slightly different in the scenarios we used for the experiments. For QoS0, MQTT and MQTT-SN produced more message losses than WS-N, while WS-N produced higher delays for the transmission of messages. MQTT-SN was very reliable when used with QoS1. The delay remained the same for MQTT-SN when using QoS1 instead of QoS0.

Since we used a realistic radio model in conjunction with challenging tactical scenarios, TCP produced many «spurious» TCP retransmits. This indicates that TCP is not well suited for the kind of wireless networks used in this scenario. It has to be analyzed further why MQTT caused more TCP retransmissions than WS-N in this case, while we observed the opposite (MQTT causing half retransmission than WS-N) in former experiments with less challenging WiFi links.

One could expect that MQTT-SN, which is based on UDP, would be better suited for these kinds of scenarios and would have lower transmission delays. But our results show that the MQTT-SN implementation had an about 50 % higher transmission delay than MQTT when used with QoS0 and a slightly higher loss rate (17 % vs. 15 %). One has to keep in mind that MQTT-SN was deployed by using an additional MQTT/MQTT-SN gateway. Possibly some amount of the delay was caused by the processing times of this additional component. For QoS1 the transmission delays remained the same for MQTT-SN and increased for MQTT. Regarding the reliability of message delivery, MQTT didn’t benefit notably from using QoS1, while MQTT-SN benefits from this QoS setting significantly.

For BFT the transmission delay is most important. Since newer positions are transmitted periodically every 10 seconds, the transmission of outdated position messages does not necessarily increase the user experience. Thus, for this kind of services the higher reliability of WS-N is not essential for the choice of the middleware and MQTT has performed best in the two scenarios we have investigated.

For other services which rely on a reliable delivery of messages, the use of MQTT-SN with QoS1 could be considered, because MQTT-SN with QoS1 was the most reliable middleware in our experiments. Additionally, the delay was lower than for WS-N.

Furthermore, the results from the network analysis showed that MQTT-SN produces less than half the amount of network data per second than MQTT and WS-N. We expect that MQTT-SN is better suited for resource constrained devices and could be superior in networks with very limited data rates. But further experiments are needed to prove these assumptions.

7 Conclusions

In this paper, we have investigated the three industry standards WS-N, MQTT, and MQTT-SN in a comparative study using the Anglova scenario (both the original, and a modified version created by Switzerland) using Swiss Wideband TDMA models. We used EMANE and DAVC as our testbed, hosted and operated by ARL, USA. The service we used was BFT with NFFI data, as implemented by FFI, Norway. Fraunhofer FKIE, Germany provided the analysis tools that were used to evaluate our results.

In our experiments, we considered that for the BFT service transmission delay is the most important metric. Since newer positions are transmitted periodically every 10 seconds, the transmission of outdated position messages does not increase the user experience. Hence, for MQTT it makes sense to use QoS0 to reduce overhead for such messages, as reliability is not needed.

We found that MQTT-SN produces a data volume of about 13-14 kbit/s compared to about 31-38 kbit/s (MQTT) and about 39-40 kbit/s (WS-N). The message sizes of MQTT and MQTT-SN are about half the size of WS-N, which makes sense since WS-N has a SOAP message layer that MQTT does not. MQTT caused notably more TCP retransmissions than WS-N, which is in contrast to former experiments with WiFi links, where the opposite could be observed. The causes for the high number of retransmissions has to be further analyzed in the future.

WS-N has less message losses compared to MQTT and MQTT-SN if these are run with QoS0. The use of QoS1 with MQTT doesn't increase the reliability significantly. But, for MQTT-SN, the reliability improves significantly by using QoS1. This makes sense since the underlying TCP in MQTT can be expected to provide some reliability, unlike UDP in MQTT-SN, which requires the additional handshaking of QoS1 to increase its reliability. The loss rate is slightly higher than MQTT when used with QoS0. It seems like MQTT-SN discards messages if the transmission takes too long. The use of QoS1 with MQTT-SN improves the reliability significantly and thus leads to an even higher reliability than WS-N.

Of specific importance to our BFT service, was, as mentioned, the delay. The average delay is higher for WS-N than for MQTT or MQTT-SN. MQTT has the lowest delay. Hence, we can conclude that for BFT services, MQTT can be a better choice than WS-N in Wideband tactical networks with similar characteristics to what we evaluated here.

Acknowledgments

We thank Mr. Chien Hsieh (ICF) for performance enhancements to DAVC and for installing and configuring the TDMA-based EMANE radio models.

References

[1] A. Nikodemski, J.-F. Wagen, F. Buntschu, C. Gisler et G. Bovet, «Reproducing measured manet radio performances using the emane framework,» IEEE Communications Magazine, vol. 56, pp. 155-155, October 2018.

[2] J.-F. Wagen, V. Adalid, G. Waeber, F. Buntschu et G. Bovet, «Performance Profiling of Radio Models and Anglova Based Scenarios,» 2019 International Conference on Military Communications and Information Systems (ICMCIS 2019), Budva, Montenegro, 2019.

[3] M. Hirsch, A. Becker, F. Angelstorf, and F. Noth. «Performance Analysis of C2IS in Distributed Tactical Networks,» 2019 International Conference on Military Communications and Information Systems (ICMCIS 2019), Budva, Montenegro, 2019.

[4] Marco Manso, Norman Jansen, Kevin Chan, Andrew Toth, Trude H. Bloebaum and Frank T. Johnsen. «Mobile Tactical Force Situational Awareness: Evaluation of Message Broker

Middleware for Information Exchange,» 23rd International Command and Control Research and Technology Symposium (ICCRTS), Pensacola, FL, USA, 2018.

[5] N. Suri, A. Hansson, J. Nilsson, P. Lubkowski, K. Marcus, M. Hauge, K. Lee, B. Buchin, L. Misirlioglu, and M. Peuhkuri. «A Realistic Military Scenario and Emulation Environment for Experimenting with Tactical Communications and Heterogeneous Networks,» 2016 International Conference on Military Communications and Information Systems (ICMCIS 2016), Brussels, Belgium, 2016.

[6] Trude H. Bloebaum and Frank T. Johnsen. «CWIX 2014 core enterprise services experimentation,» FFI-report 2014/01510. https://www.ffi.no/no/Rapporter/14-01510.pdf

[7] VerneMQ homepage. «VerneMQ – A MQTT broker that is scalable, enterprise ready, and open source,» accessed 2019-03-18, https://vernemq.com/

[8] Eclipse, «MQTT-SN Transparent Gateway,» accessed 2019-03-18, https://www.eclipse.org/paho/components/mqtt-sn-transparent-gateway/

[9] OASIS. «Web Services Brokered Notification 1.3 (WSBrokeredNotification),» OASIS Standard, 1 October 2006. http://docs.oasis-open.org/wsn/wsn-ws brokered notification-1.3-spec-os.pdf

[10] OASIS. «MQTT Version 3.1.1,» OASIS Standard 29 October 2014. http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf

[11] U.S. Naval Research Laboratory. «Extendable Mobile Ad-hoc Network Emulator (EMANE),» accessed 2019-03-18, https://www.nrl.navy.mil/itd/ncs/products/emane

[12] CCDC Army Research Laboratory. «Network Science Research Laboratory,» accessed 2019-06-13, https://www.arl.army.mil/www/default.cfm?page=2485

[13] OLSR «Optimized Link State Routing Protocol» accessed 2019-06-13, http://www.olsr.org