Implementation of a Priority Service for SMTP

Michael Schmeing
FGAN FKIE/KOM
Neuenahrer Straße 20
53343 Wachtberg
Germany

schmeing@fgan.de

ABSTRACT

Current Military Message Handling Systems (MMHS) are based on the X.400 standard of the International Standardization Organization. Recent examinations have shown that the Simple Mail Transfer Protocol (SMTP) can satisfy most requirements for MMHS. For the only relevant requirement not satisfied, we have developed a solution. This paper describes the solution and an experimental implementation. Additionally, the results from a series of tests on a tactical narrow bandwidth testbed are described.

1 INTRODUCTION

In military operations, orders and reports are usually transferred in written form. For this purpose (and others) Military Message Handling Systems (MMHS) are used. Current MMHS are based on STANAG 4406 [1], which is basically a military profile of X.400 as defined in [2]. On the other hand, in the civilian domain the Internet based email system is used for electronic messaging. The email system is defined by the Internet Engineering Task Force (IETF). The basic standards for the email system are the "Simple Mail Transfer Protocol" (SMTP) [3] and the "Internet Message Format" [4].

Originally (i.e. in the early 1980's), the email system only offered a basic transport service for messages in the ASCII character set. The current version of the system, on the other hand, is capable of transmitting arbitrary data types using additional functionality for the transfer of the messages such as delivery status notification.

In 2004, an evaluation of the email system [5] has been conducted to determine whether it can replace X.400 as the base technology for a next-generation MMHS [6]. This evaluation yielded only one military requirement not yet covered by civilian standards: message priorities[1]. With a priority service, emails marked with a high priority get expedited handling at the cost of lower priority messages.

The rest of this paper is organized as follows: The following section gives a brief introduction to the SMTP. Section 3 describes an extension to the SMTP defining a priority service. An experimental implementation of the extension is explained in Section 4 and a narrow bandwidth testbed used to test the implementation is given in Section 5. In Section 6, the actual experiments conducted on the testbed as well as their results are presented.

[1] Work on another missing feature has been started by the IETF in the meantime. A third military requirement can only be addressed partially. For more details on these features, see [5].

Schmeing, M. (2006) Implementation of a Priority Service for SMTP. In *Dynamic Communications Management* (pp. 9-1 – 9-14). Meeting Proceedings RTO-MP-IST-062, Paper 9. Neuilly-sur-Seine, France: RTO. Available from: http://www.rto.nato.int/abstracts.asp.

2 INTRODUCTION TO SMTP

This section provides a brief overview over the "Simple Mail Transfer Protocol" as defined in Request For Comment (RFC) 2821 [3]. The SMTP is the protocol used to transfer emails between different components of the email system. It is organized as a client/server protocol, where the *SMTP client* initiates a connection to an *SMTP server* and sends *SMTP commands* to it. A server waits for connection requests from clients, executes their commands and sends the results back to the clients in the form of *SMTP replies*. An SMTP command consists of an *SMTP verb* and depending on the command a list of mandatory or optional parameters. The terms *command, client* and *server* are used as synonyms for "SMTP command", "SMTP client" and "SMTP server" respectively.

Usually, each SMTP command has exactly one SMTP verb. On the other hand, the `HELO` command is an example of a command with two SMTP verbs. This command is used to identify the client to the server and indicate whether the client supports the extension mechanism (using the verb `EHLO`) or not (using `HELO`).

Figure 1 shows an example of an SMTP session during which an email is sent to two recipients. In lines 1 to 6, the session is started: The server indicates that it is ready to start (line 1); the client communicates its identity to the server and the fact that it supports the extension mechanism (line 2); finally the server accepts the session with the client (line 3) and provides a list of all SMTP extensions it supports (lines 4 – 6).

Client connects to server

```
1  S: 220 mail.some.domain.com ESMTP ready
2  C: EHLO host.some.domain.com
3  S: 250-mail.some.domain.com Hello host.some.domain.com
4  S: 250-SIZE 31457280
5  S: 250-PIPELINING
6  S: 250 HELP
7  C: MAIL FROM:<c.brown@some.domain.com>
8  S: 250 <c.brown@some.domain.com> is syntactically correct
9  C: RCPT TO:<snoopy@another.domain.com>
10 S: 250 <snoopy@another.domain.com> verified
11 C: RCPT TO:<schroeder@another.domain.com>
12 S: 250 <schroeder@another.domain.com> verified
13 C: DATA
14 S: 354 Enter message, ending with "." on a line by itself
15 C: Date: Tue, 11 Apr 2006 08:16:05 +0200
16 C: From: Charlie Brown <c.brown@some.domain.com>
17 C: To: Snoopy <snoopy@another.domain.com>
18 C: Cc: Schroeder <schroeder@another.domain.com>
19 C: Subject: Test
20 C:
21 C: Some text ....
22 C: .
23 S: 250 OK
24 C: QUIT
25 S: 221 mail.fgan.de closing connection
```

Server closes connection

Key:
C: Line sent from client to server
S: Line sent from server to client

Figure 1: Example of an SMTP Session

In line 7, the client initiates an email transaction and defines c.brown@some.domain.com as the originator. The server accepts the transaction in line 8. The recipients of the email are defined by the client in lines 9 and 11 and accepted by the server in lines 10 and 12 respectively.

With the DATA command in line 13 the client instructs the server to prepare for the content of the email, which the server confirms in line 14. The actual content is shown in lines 15 to 21. As instructed by the server, the client sends a line consisting only of a dot to indicate the end of the content (line 22). In line 23 the server confirms that it has received the content and accepts responsibility to undertake every effort to deliver the email to its recipients. Afterwards, the session is closed in lines 24 and 25.

3 SMTP SERVICE EXTENSION FOR PRIORITY MESSAGE HANDLING

In Sec. 1, it has been explained that the major military requirement not satisfied by the current version of the email system is the requirement for priorities. This section introduces the "SMTP Service Extension for Priority Message Handling" developed at FGAN that can satisfy this requirement.

The basic service of this extension is expedited processing of emails according to their priority. Beside this requirement, it is possible for a policy governing use of priorities to attach arbitrary constraints to certain priority levels. Common examples are size limitations or maximum time frames within which an email of a certain priority must reach its intended recipients.

A server announces the fact that it supports the priority extension in its reply to an EHLO command from a client by inserting a line of the form 250-PRIORITY *policy-identifier*. Figure 2 shows that reply from the example in Fig. 1 with an added indicator that the priority extension is supported (l. 5a).

```
2   C: EHLO host.some.domain.com
3   S: 250-mail.some.domain.com Hello host.some.domain.com
4   S: 250-SIZE 31457280
5   S: 250-PIPELINING
5a  S: 250-PRIORITY http://some.domain.com/priority-policy.html
6   S: 250 HELP
```

Key:
C: Line sent from client to server
S: Line sent from server to client

Figure 2: EHLO SMTP Command with Priority Support

The straight forward approach to deploy priorities would be to attach them to the whole message. This could be achieved by either defining a new SMTP command or a new parameter to the MAIL FROM: command. One consequence of such an approach would be that the email would have to be transmitted using the same priority for all recipients. Nevertheless, one can imagine situations, where it is possible to send the email to some recipients with a lower priority than required by others. An example for such a recipient that does not need high priorities could be an external archiving system. With the straight forward approach, the email would either have to be sent twice, once for the primary recipient with high priority and once for the archiving system with low priority or the archiving system would receive it at an unnecessary high priority. In the first case, the network would be burdened with two basically redundant transmissions eating up data rate unnecessarily which may be unacceptable in tactical scenarios where data rate is often severely limited. The second alternative might interfere with other emails of the same priority or a priority between the one the archive would need and the one it received the email at on that part of the delivery path that is not shared with the primary recipient. Thus, these emails may be delayed unnecessarily.

To take advantage both of the multiple recipient feature shown in Fig. 1 and the fact that it is possible that different recipients have different priority requirements, the SMTP Service Extension for Priority Message Handling attaches the priorities to the individual recipients. Each email is to be treated according to the highest priority assigned to any recipient. This priority is called the priority of the email. Lower priorities can become effective, if the path to some of the recipients diverts from the path to others. Thus, the perceived priority of an email may decrease during the delivery for a specific recipient.

To illustrate this, assume, the email shown in Fig. 1 is to be transferred as displayed in Fig. 3: It is submitted to MTA 1 which sends it to MTA 2. MTA 2 has to send it to MTA 3 for the recipient address `snoopy@another.domain.com` and to MTA 4 for `schroeder@another.domain.com`. If `snoopy` had priority i and `schroeder` priority $j < i$, then MTAs 1, 2 and 3 would perceive priority i for the email while MTA 4 would perceive only the lower priority j for the email. Additionally assume that MTA2 first sends the email to MTA 3 and then to MTA4 (instead of sending it to both in parallel). After the email has been sent to MTA3, its priority decreases from i to j from the perspective of MTA 2 as the transfer to MTA 4 only requires this priority.

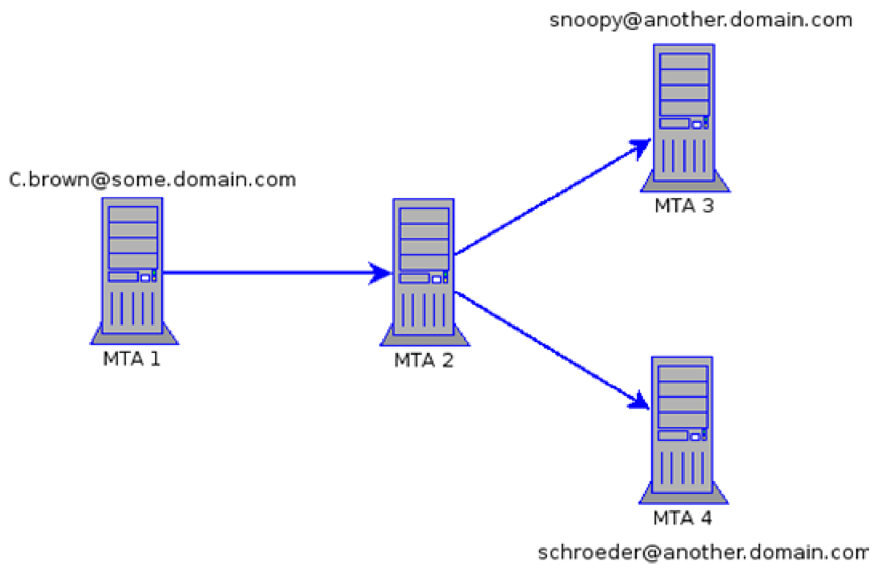

Figure 3: Example of email delivery path

An example of priorities being bound to individual recipients is shown in Fig. 4. It shows the two recipient definitions from lines 9 and 11 of Fig. 1 with added priorities and the respective replies from the server.

```
9  C: RCPT TO:<snoopy@another.domain.com> PRIORITY=FLASH
10 S: 250 Priority FLASH accepted for <snoopy@another.domain.com>
11 C: RCPT TO:<schroeder@another.domain.com> PRIORITY=ROUTINE
12 S: 250 Priority ROUTINE accepted for <schroeder@another.domain.com>
```

Key:
C: Line sent from client to server
S: Line sent from server to client

Figure 4: `RCPT TO:` SMTP Commands with Priorities

While it is generally left to the local policy to decide how many priority levels are required and what their respective constraints are, the priority extension mandates that all servers support recipients without added priority. This ensures servers with support for the priority extension remain backwards compatible with servers that don't. Servers must even be able to correctly handle emails where only some recipients have priorities added.

Unless local policy defines a different behaviour, only emails for recipients with no priorities attached may be send to servers that do not support the priority extension or have a different policy.

3.1 Example Policy

For testing purposes, an example policy has been defined that is described in this section. This policy is loosely oriented at the priority policy used by the old ACP 127 systems. It defines four priority level. They are in ascending order: `ROUTINE`, `PRIORITY`, `IMMEDIATE` and `FLASH`. Emails for recipients with no priority attached are given less priority than `ROUTINE` during processing. The policy defines that emails may be passed to servers without support for the priority extension or with different policies for recipients which have the two lower priorities (`ROUTINE` and `PRIORITY`) assigned to them, if it is otherwise not possible to deliver the emails successfully. If recipients with a priority of `IMMEDIATE` or `FLASH` can not be reached through servers supporting the priority extension and the example policy, this is to be treated as an unrecoverable error. Additionally, emails of priority `IMMEDIATE` are limited to 4096 bytes in size and emails of priority `FLASH` to 2048 bytes. Emails of priorities `ROUTINE` and `PRIORITY` are unlimited in size. In all cases external circumstances such as limited storage space on a server or other applicable policies may impose stricter constraints, which then take precedence over the constraints from the priority policy.

The “Deliver By SMTP Service Extension” [8] defines a service that allows to define the maximum time frame within which an email must reach its recipients after it has been submitted to the first server. Using this extension, the example policy states that emails with priority `ROUTINE` must reach their recipients within four hours. For `PRIORITY` this time frame is one hour, for `IMMEDIATE` 30 minutes and 10 minutes for `FLASH`. If an email can not be delivered within the time frame defined by the priority, this is to be considered an error. As support for this extension is very rare, support for the delivery time constraints has been declared mandatory only, if the Deliver By extension is present. Otherwise the time constraints are optional.

4 EXPERIMENTAL IMPLEMENTATION OF THE PRIORITY EXTENSION

This section describes the experimental implementation of the priority extension. First a few terms required for this section and the following are explained. Afterwards, the two main components of the implementation are described.

4.1 Some Terminology

A *Mail Transfer Agent* or MTA is a software component that is responsible for transporting emails to the next MTA on their route to the recipient or delivering them to the recipients mailbox. It usually contains both an SMTP server (for receiving emails) and an SMTP client (for sending them). The emails can be transmitted to the MTA from another MTA or from an MUA.

A *Mail User Agent* (MUA for short) is a component that is responsible for creating, sending, receiving and managing emails on behave of the user. It contains an SMTP client and often one or more clients for protocols such as POP3 or IMAPv4 to transfer the emails from the mailbox on the host of the provider to the computer of the user.

4.2 Mail Transfer Agent

The central component of the implementation is an MTA supporting the SMTP Service Extension for Priority Message Handling. Instead of creating a complete MTA, it has been decided to extend an existing one.

Several MTAs have been examined to identify the one best suited for extension. While extensibility was the main criterion for the selection, some mandatory requirements had to be fulfilled: The MTA had to be available in source code and under a license allowing modification and distribution of the modified version; it had to support IPv6 and it had to offer some SMTP extensions such as an extension to check the size of an email prior to its being sent [6] or to authenticate different components of the net against each other [7]. From the MTAs that offered the required functionality, `exim` was the one that appeared easiest to extend and has therefore been chosen.

`exim` stores emails in spool files after reception. If it is possible, arriving emails are sent to their next MTA or delivered to the recipient's mailbox immediately after reception. If this is not possible, emails are queued for later transmission. One common reason for queueing is that the maximum allowed number of outgoing connections has been reached. In such a case, a special process called *queue runner* is started at predefined intervals to try to send the waiting emails. Normally, emails would be sent in the order of arrival, but if the queue runner detects several emails that are to be sent to the same next MTA, they are all sent in the same SMTP connection.

For priority support, this algorithm must be changed in several ways. Most importantly, the first criterion for deciding which email to send has to be the priority: Emails with higher priorities have to be sent before those with lower priorities, even if the latter arrived earlier. Second, it is not possible to send several emails directly one after the other any more. Instead, the MTA has to check whether new emails with higher priorities have arrived during the transmission of the current one before the next one can be started. The priority extension does not require preemption of running email transactions to satisfy higher priority requirements by emails arriving during the transmission of another email but mandates the preemption of sessions between emails in that case.

Given the features offered by `exim`, all mandatory and a few optional requirements of the priority extension could be implemented. Just like the other examined MTAs, `exim` does not support the "Deliver By SMTP Service Extension" [8]. Therefore, the (optional) delivery time constraints could not be implemented.

RFC 2597 [12] describes a set of four Assured Forwarding (AF) classes for data packages at the network level. These classes are as well supported by the Network Adapter introduced in Sec. 5. To deploy this feature, the MTA assigns AF classes to emails with SMTP priorities in ascending order, i.e. from AF class 1 for `ROUTINE` emails to AF class 4 for emails with priority `FLASH`. Emails with no priority are sent as best effort.

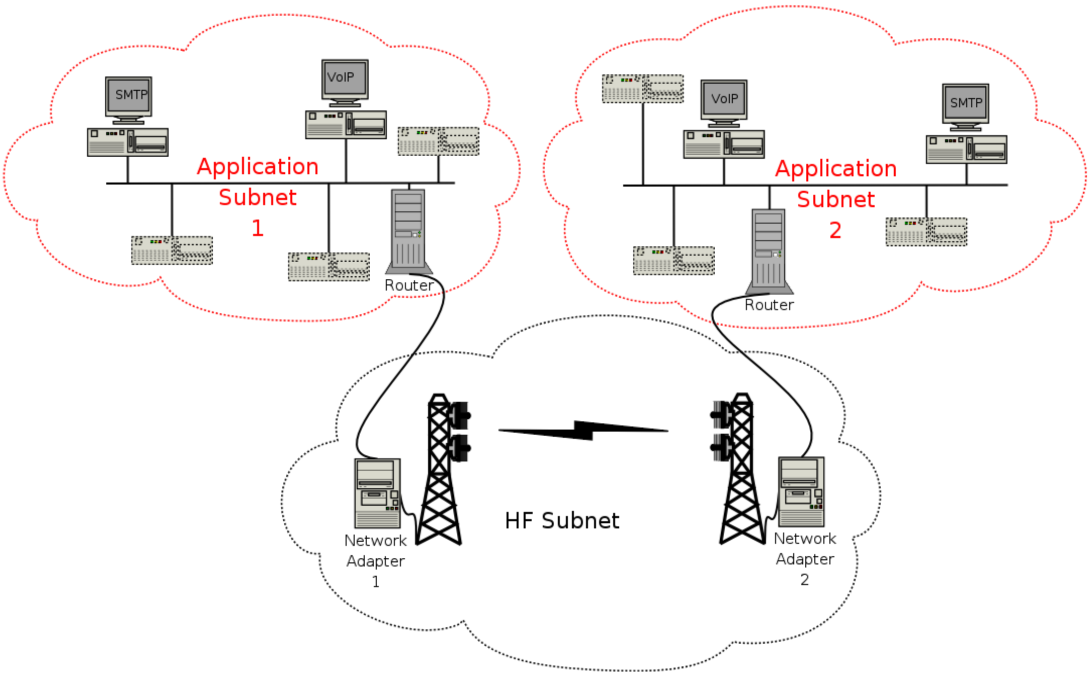

Figure 5: Layout of the low-data rate Testbed

4.3 Mail User Agent

To be able to send prioritized emails, a command line oriented partial MUA was implemented. The primary purpose of the MUA is to be used in tests of the MTA. For this reason, only the sending part of an MUA, i.e. the SMTP client has been implemented. Another consequence of this purpose is that the commandline parameters have been organized according to this task instead of following the commandline parameters of the program `sendmail`, still the most popular MTA, as many similar programs do.

As the program is designed to send one email at a time (although to an arbitrary number of recipients), no priority management is required: Whenever the program is started, it sends the email defined on its commandline to the defined recipients assigning the given priorities to them. Another limitation is that the size of the email is not checked against the limitations of the assigned priorities. This omission has been made to be able to test the handling of emails which are too large for one or more of the assigned priorities by the MTA[2].

For sending an email, basically two modes exist: The email can be composed by the program from information given on the commandline or a pre-formatted email can be send directly. In the latter case, the user has the responsibility to ensure that the email completely conforms to the email format.

When sending a pre-formatted email, the information on the commandline such as originator and recipients is used only for the SMTP dialogue. When the email is composed by the program on the other hand, the recipient and originator information on the commandline is as well used to fill the header fields of the email. Beside the originator and recipients information such as the subject or the date header[3] field can be specified. Additionally, the main part and the attachments can be specified on the commandline.

[2] It is planned for a later version of the MUA to implement this feature and make it (de)activable via the commandline.

[3] Providing the date on the commandline is optional. If it is omitted, the program generates a correct date header field for the current time.

As the MUA was only used on the same host as the MTA to which the emails were submitted, no priority management for the network was implemented.

5 LOW-DATA RATE TESTBED

This section describes the layout of the testbed for the priority extension. Figure 4 shows the general topology: Two wired "Application Subnets" are connected via an "HF Subnet". Each application subnet contains several nodes, each designed for a specific function. One of these nodes is an SMTP host on which the MTA and MUA described in the previous section are installed. Another example is a Voice-over-IP (VoIP) host, which was used to generate real-time background traffic in the form of prioritized packages. Additionally, one of the application subnets contained an uplink to the Intranet of the laboratory basically providing access to the Domain Name Service (DNS) name server. Such a DNS is required for SMTP to operate correctly. Each application subnet as well contains a router connecting the application subnet to the HF subnet. Actually, the routers are IPSec (IP-Security) gateways, but because the name server could not be included in the IPSec infrastructure, this functionality has been disabled.

The HF subnet consists of one Network Adapter (NA) per application subnet. The Network Adapters are responsible for managing the network traffic on the low-data rate link. A full description of the NA is available in [9]. One of the major means to manage network traffic is priority based packet handling. As has been explained in Sec. 4.2, the network priorities are the AF classes from [12]. The Drop Precedence defined in that standard is ignored in the current version of the NA.

The connection between the NAs can be realized in different ways:

- as a hardware HF Link;
- as a simulated HF Link using an HF simulator; or
- as a null-modem connection set to an appropriate data rate.

For these experiments, the last option has been chosen.

6 EXPERIMENTS AND RESULTS

This section describes the types of experiments conducted on the testbed and their results. The experiments are classified by which of the SMTP hosts have sent emails and whether any VoIP traffic has been created.

When designing the experiments, it has become clear that it is not possible to synchronize the clocks of the involved SMTP hosts over the HF link due to high resource requirements for such synchronization. For this reason, it was not possible to measure the time an email took from the submitting MTA to the receiving one. To compensate for this, when either of the SMTP hosts received a special email, called a start email, from the other host, it sends a reply email with the same Message-ID and priority back to the originating SMTP host. An SMTP host sending a start email logs the time at which the start email was sent as well as any reply emails it receives. The round-trip time of the emails could be determined from these two log files through the Message-Id of the start and reply emails.

The first set of experiments has been conducted without explicitly generated background traffic[4]. In this set of experiments, all have been conducted in two variants: In the first execution, only one of the SMTP hosts sent start emails and in the second both.

[4] Administrative background traffic such as name server queries or ICMP messages have been ignored because they could not be measured.

When examining the results from the individual executions of the experiments, it becomes obvious that during some of them, the round-trip times are rather constant while in others they are constantly increasing. As the latter are those that created the most data rate usage (i.e. the highest total number of emails) it became clear that during these executions the network, especially the low-data rate link was overloaded. Consequently, these are the executions that are of the highest interest to the evaluation of the effects of the priority extension.

As an example, Figs. 6 and 7 show the results of an experiment with no background traffic, where emails with three different priorities have been used. In the execution shown in Fig. 6, only one of the SMTP hosts has sent start emails. Here it can be clearly seen that the round-trip times for emails of a given priority remain within a certain corridor. The higher the priority, the smaller the corridor and the lower the absolute boundaries of the corridor.

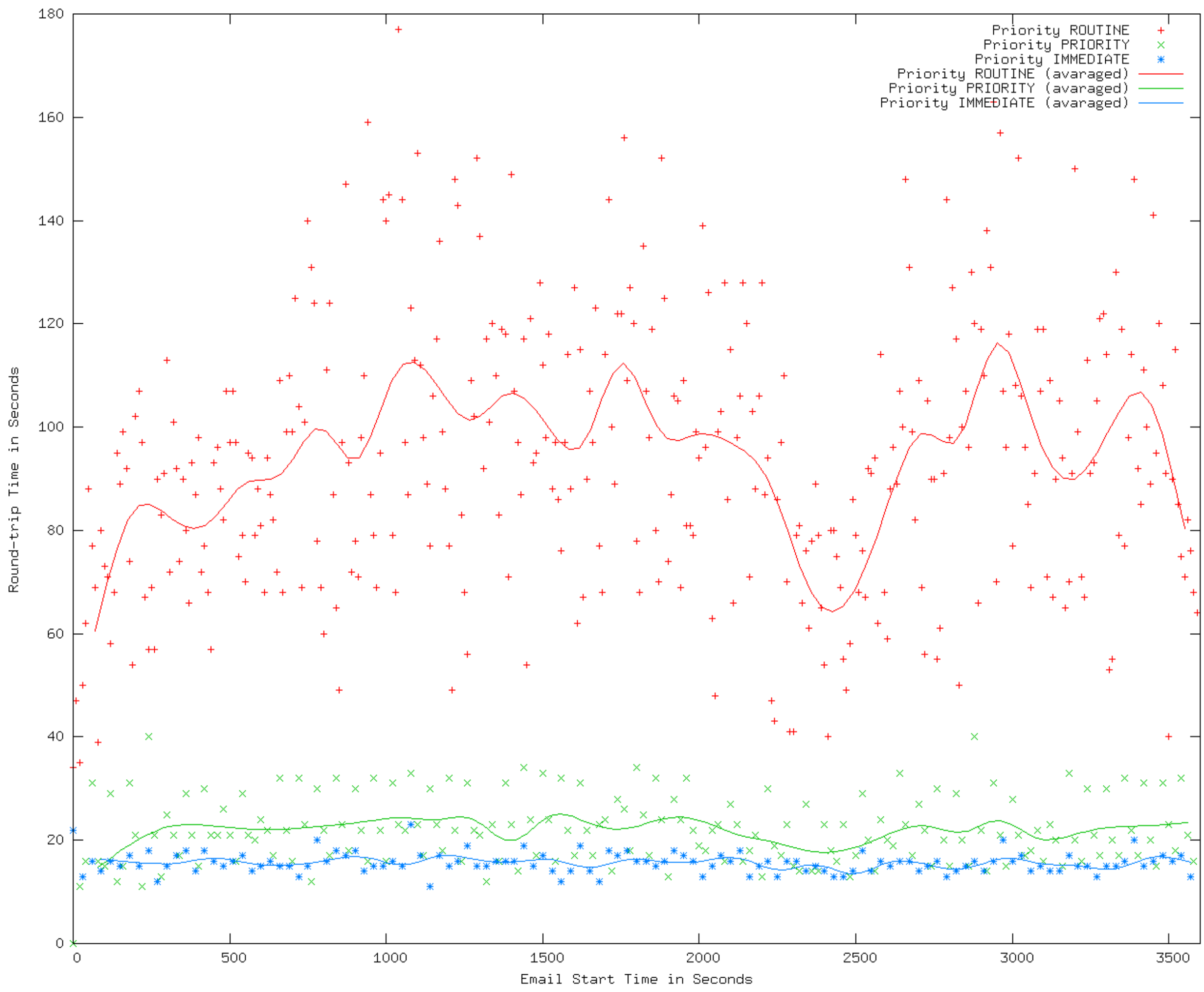

Figure 6: Three priorities w/o network overload

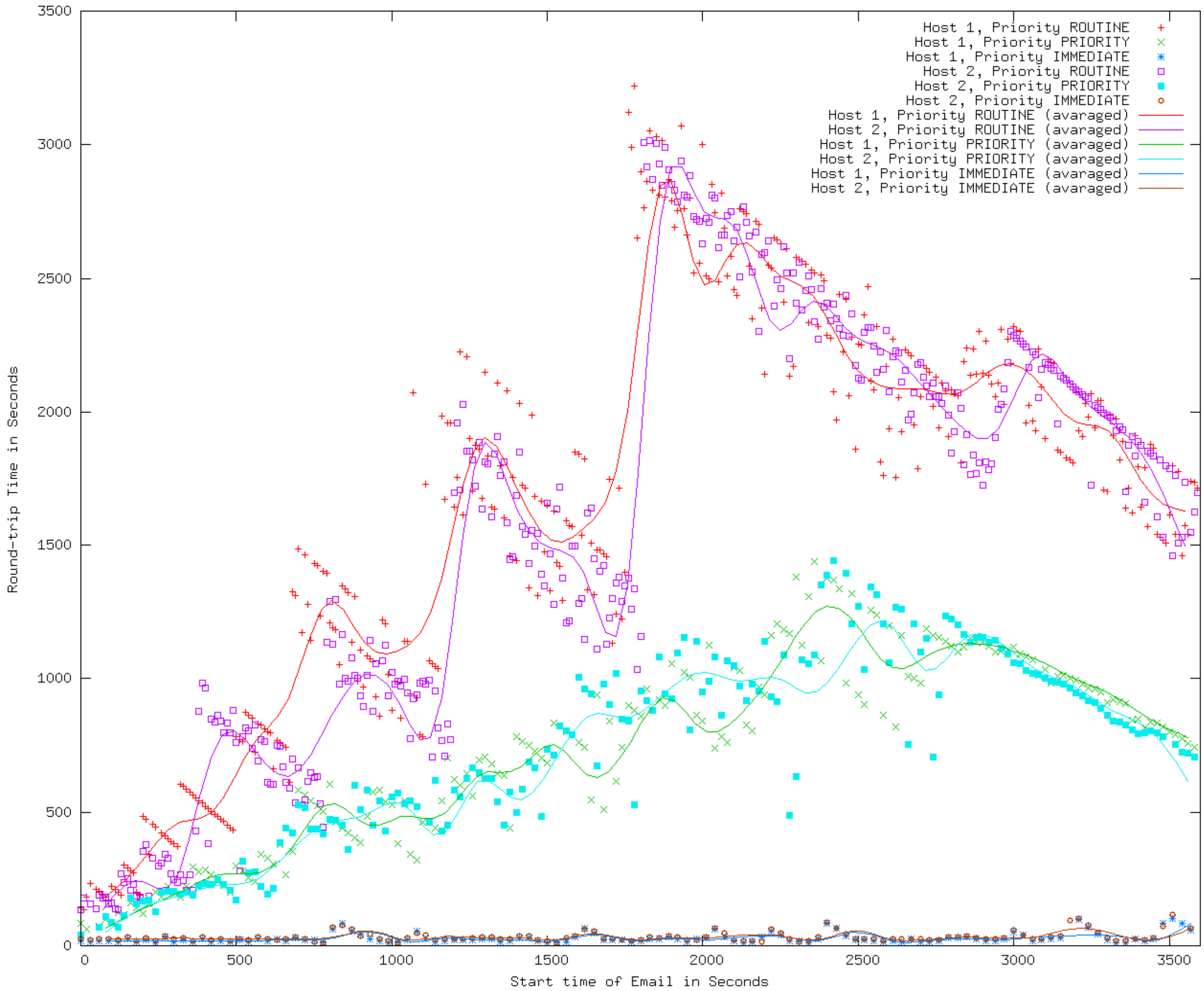

Figure 7: Three priorities with network overload

Figure 6 shows the execution of the same experiment where both SMTP hosts sent start emails. Here it can easily be seen that for the lower priorities (i.e. ROUTINE and PRIORITY), no corridor can be identified. Instead, the round-trip times are constantly increasing. All emails with highest priority however seem to have more or less the same round-trip time. Closer inspection of the results from the execution basically confirmed that impression: The round-trip times of these emails are within a rather narrow corridor well below the resolution of the diagram.

This experiment can be taken as a strong hint that priority handling can actually achieve its intended objective for SMTP based email.

The primary focus of the second set was to examine the influence of the background traffic at different priority levels on the email round-trip times. To generate this background traffic, a VoIP program (PC-Phone) developed at FGAN has been used. On a background of voice traffic at 1200 bits/sec or 2400 bits/sec and various priority levels start emails with two priorities (ROUTINE and IMMEDIATE) have been sent from one host.

All executions of the experiment with 1200 bits/sec voice traffic show similar results independent of the priority of the voice traffic. The same is true for the experiments with 2400 bits/sec voice background. As can be seen in Fig. 8, for 1200 bits/sec even with FLASH voice traffic no overload could be created in the given scenario. The results with a 2400 bits/sec voice traffic with otherwise unchanged conditions is shown in Fig. 9. These results show that it is basically the presence and resource requirements of realtime traffic that affects the transmission of non-realtime traffic rather than the priority of the realtime traffic.

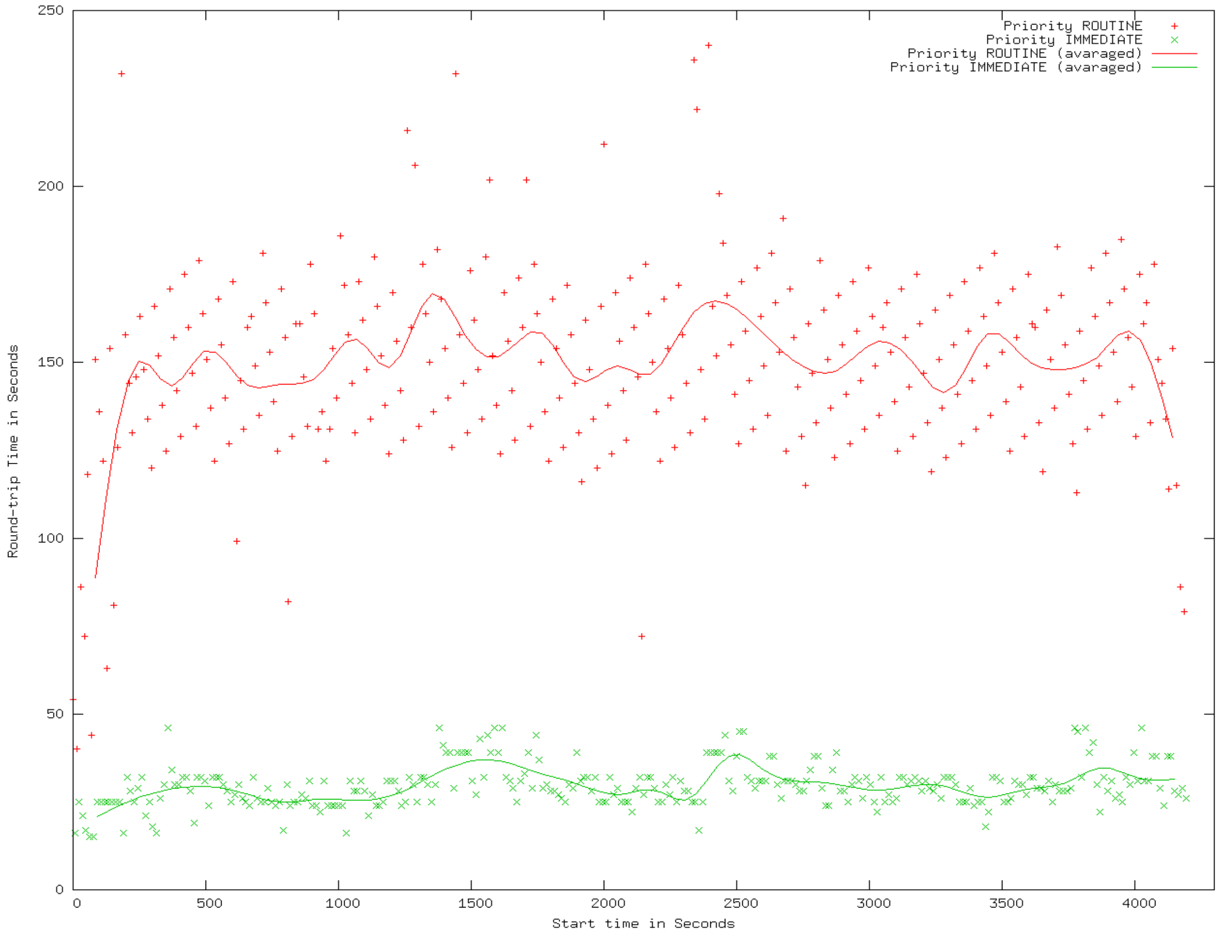

Figure 8: Two priorities with 1200 bits/sec FLASH voice background

Figure 9: Two priorities with 2400 bits/sec FLASH voice background

7 FUTURE WORK

After the tests on the testbed, it is planned to create an implementation of the priority extension suited to be used on a network simulator, e.g. ns-2. This would allow to test the extension in a completely controlled environment and thus eliminate external factors.

Development of the extension itself is currently continuing. While no fundamental changes are to be expected, the mechanism to negotiate supported policies and applied priority levels will be brought closer to the schema used in the *Session Initiation Protocol* (SIP) [10,11]. The ultimate goal of these activities is to achieve standardization of the extension by the IETF.

8 SUMMARY AND CONCLUSIONS

This paper has introduced the protocol used to send email in the Internet, the Simple Mail Transfer Protocol, and has given a summary of an evaluation concerning its military applicability. The result of this evaluation was that only one major requirement can not be satisfied: message priorities. Based on this evaluation, the paper has described an extension to the SMTP that implements this feature. The extension allows to assign priorities emails to enforce expedited delivery of emails with higher priorities.

Additionally, an experimental implementation of this extension based on an existing MTA has been presented together with a low-data rate testbed. Although the low-data rate link has only been a null-modem connection in the tests described in this paper, it is possible to replace that connection by either an HF simulator or an actual HF link.

On this testbed, a series of experiments has been conducted. The results of these tests have been summarized. They show, that prioritization can actually help emails with higher priorities to still reach their intended recipient(s) even if the network is under heavy load. Especially in such situations, emails with higher priorities are delivered quicker than those with lower priorities.

Another set of experiments has shown that the network priority of VoIP traffic is less relevant for the email delivery time than the actual amount of the traffic.

As a conclusion, the priority extension is considered to fulfill the expectation but further work is required to determine the actual benefit it yields. Nevertheless, this paper has shown that it is possible to extend the SMTP so that it can meet all military requirements. SMTP is thus a possible candidate for the next MMHS after STANAG 4406 [1]. Given the fact that with the priority extension the last missing feature for military use has entered civilian standardization processes even an SMTP version for military use can be at least consist of COTS standards only. It may even be possible to use COTS products.

9 REFERENCES

[1] NATO MMHS WG. Military Message Handling Systems. STANAG 4406, NATO, March 1999.

[2] CCITT Eighth Plenary Assembly. Recommendations X.400 – X.420: Data Communication Networks Message Handling Systems. ITU, November 1988, CCITT Blue Book Volume VIII.

[3] J. Klensin. Simple Mail Transfer Protocol. RFC 2821, IETF, April 2001.

[4] P. Resnick et. al. Internet Message Format. RFC 2822, IETF, April 2001.

[5] M. Schmeing, N. Haak. Applicability of Internet-email for Military High-grade Messaging. FKIE-Report 93, FGAN, February 2005.

[6] J. Klensin et. al. SMTP Service Extension for Message Size Declaration. RFC 1870, IETF November 1995.

[7] J. Myers. SMTP Service Extension for Authentication. RFC 2554, IETF, March 1999.

[8] D. Newman. Deliver By SMTP Service Extension. RFC 2852, IETF, June 2000.

[9] R. Göbel. Anforderungen an einen QoS-unterstützenden Netzadapter für schmalbandige Subnetztypen. FKIE-Report 38, FKIE, January 2001.

[10] J. Rosenberg et.al. SIP: Session Initiation Protocol. RFC 3261, IETF June 2002.

[11] H. Schulzrinne, J. Polk. Communications Resource Priority for the Session Initiation Protocol (SIP). RFC 4412, IETF, February 2006.

[12] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski. Assured Forwarding PHB Group. RFC 2597, IETF, June 1999.