Configuration and analysis of robust header compression in UMTS

Ana Carolina Minaburo Villar, Loutfi Nuaymi, Kamal Deep Singh, Laurent Toutain

Configuration and Analysis of Robust Header Compression in UMTS

Ana Minaburo, Loutfi Nuaymi, Kamal Deep Singh, Laurent Toutain
ENST-BRETAGNE, Campus Rennes, BP 78, Rennes, France
{anacarolina.minaburo, loutfi.nuaymi, kamal-deep.singh, laurent.toutain}@enst-bretagne.fr

Abstract.–**ROHC [1] is the new standard for header compression proposed by the IETF (Internet Engineering Task Force) to compress different protocol headers. ROHC needs to be configured to afford the changes in the characteristics of the UMTS radio link and to enable a better compression of the IPv6 flows. In this document we will introduce the characteristics and the behavior of ROHC. Also we study the compression parameters that are recommended as implementation dependent in the ROHC specification and we show how to configure them. This study can help to improve the efficiency and robustness of ROHC when it is used in scarce resources links, such as UMTS.**

I. INTRODUCTION

The third generation of mobile system, Realease 4 of UMTS [2] (Universal Mobile Telecommunication System) will have many IP based services. However, the use of IP will mean large overheads, which will take a significant amount of bandwidth that is already scarce in radio links. Hence, there is a need to compress the large headers and save the valuable radio resources. There are some schemes for header compression that work by compressing the header after removing the redundant information from it and thus saving the bandwidth when it is scarce.

Existing header schemes do not work well over cellular links due to high error rates and long round trip times. A viable solution thus, is required and represented by ROHC that has been proposed by the IETF (Internet Engineering Task Force) working group. ROHC aims at providing high compression efficiency with high robustness and can be used in the PDCP (Packet Data Convergence Protocol) [2] layer of the UMTS.

To investigate the behavior of ROHC for real-time applications in UMTS architecture we have made an implementation of ROHC for IPv6 and we simulate the error conditions and the long round trip time that prevail in the radio link. Based on our results we study how to configure ROHC for UMTS and we adjust some parameters, which are crucial for the performance of ROHC in terms of efficiency and robustness.

This paper first introduces the ROHC platform and our implementation, for more details refer to [1]. This paper then explains the ROHC compression schemes and parameters. Followed by that, a complete study of these parameters is done based on the implementation results.

II. ROHC PLATFORM

The main reason why header compression should be done is the fact that there is redundancy in the header information and between header fields of the same data stream. ROHC mechanism works by removing the redundancy and transfers only changing fields. It classifies the header fields into static and dynamic fields. Static fields are those that remain constant during the lifetime of the packet and dynamic fields are those that keep on changing but their change pattern may be known.

The ROHC compressor removes the redundant header fields and the redundant information in the packet flow. ROHC compression communicates changing fields most of the time. While sending the fields that change, it further achieves efficiency by using an encoding algorithm in which only the last significant bits are sent.

The ROHC compressor has three compression levels: Initialization and Refresh (IR), First Order (FO) and Second Order (SO). In IR compression level it tries to establish the static information and in FO compression level it establishes the change pattern of dynamic fields. In the last compression level, SO, it sends encoded values of Sequence Number (SN) and Timestamp (TS) forming the minimal size packets. With the use of this header format packet all header fields can be generated at the other end of the link using the previously established change pattern. When some updates or errors are there, the compressor goes back to the upper compression levels. It only returns to the SO compression level after retransmitting the updated information and establishing again the change pattern in the decompressor.

The decompressor works at the receiving end of the link and decompresses the headers based on the header fields' information of the context. Both the compressor and the decompressor use a context to store all the information about the header fields. To ensure correct decompression, the context should be synchronized all the time.

The decompressor has three states, the first is No Context (NC) that is when there is no context synchronization, and the second is Static Context (SC) that is reached only after the dynamic information in the context is lost. The third is Full Context (FC), reached when the decompressor has all the information about header fields. When in FC state, the decompressor moves to the initial states as soon as it detects context damage. It uses the k out of n rule by looking at the last n packets, if CRC failures have occurred for at least k packets then, it assumes context damage and transits backward to an initial state. The decompressor also sends feedback according to the operation modes. ROHC has three operation modes: Unidirectional (U), bi-directional Optimistic (O) and bi-

0-7803-7822-9/03/$17.00 © 2003 IEEE.

directional Reliable (R). The U-mode is used when the link is unidirectional or when feedback is not possible. For bi-directional links we can use the O-mode or the R-mode. The O-mode sends only negative feedbacks, optionally it can also send positive feedbacks but the R-mode uses both negative and positive feedbacks.

The Decompressor manages the operation mode in which the system will work through the use of mode transitions that allow it to change from one mode to another, based on the link characteristics and the performance requirements. The decompressor also uses some efficient schemes to correct the context when it gets damaged or the synchronization gets lost. The compressor also employs some schemes through which it ensures the correct transmission of the information to the decompressor. These schemes involve many parameters and they control the performance of ROHC in terms of compression efficiency and robustness.

In order to test the performance of ROHC over UMTS radio link we have developed the ROHC layer for IPv6 and we have studied the role of its schemes and its parameters. To study the behavior of ROHC in conditions that are common to the UMTS radio link, we have developed a simple error simulator, which generates error affecting the header and the payload of the packets. Our error simulator generates error in the received packets based on BER (*Bit Error Rate*) of a radio link [2]. The error model, which is present in the physical layer, then passes the packet to the upper layer. The packets are thus received with the known BER, whose value can be controlled.

In our ROHC IPv6 implementation, first the physical layer is established and then the negotiation takes place, establishing ROHC negotiation channel parameters that define the channel and are different from the compression parameters discussed later in this paper.

After negotiation, the context space is then allocated corresponding to different CIDs (*Context Identifier*), which differentiate each channel and all contexts. The contexts are initialized to the values specified in ROHC [1]. The compressor can operate in any of the defined standard modes but it starts always in U-mode. The compressor has two inputs one for receiving feedback packets and other for receiving packets to be compressed. It has only one output through which compressed packets are sent. The decompressor starts in NC State and has one input for receiving compressed packets and two outputs for sending feedback packets and decompressed packets respectively.

III ROHC COMPRESSION PARAMETERS AND SCHEMES

The value of the compression parameters of ROHC that determine the efficiency and robustness are not defined in ROHC specification and are not negotiated initially but are stated as implementation dependent. In order to analyze these parameters for the performance of ROHC for IPv6 flows over UMTS, in our implementation these parameters are made flexible and can be changed. The compression parameters see figure 1 and the schemes that use them, are as follows:

1. *L:* In U-mode and O-mode the ROHC compressor uses a confidence variable (L) in order to ensure the correct transmission of header information. This means sending the same header format packet of each compression level, at least L times for the first levels before transitting to SO compression level. If even a single packet reaches the decompressor the information gets communicated.
2. *Timer_1 (IR_TIMEOUT):* In U-mode, the compressor uses this timer to return to the IR compression level and periodically resends static information because it has no way to get any feedback about the errors which may occur at the decompressor.
3. *Timer_2 (FO_TIMEOUT):* The compressor also uses another timer in U-mode and this timer is used to go downward to FO compression level if the compressor is working in SO compression level.
4. *Sliding Window Width (SWW):* The compressor while compressing header fields like Sequence Number (SN) and Timestamp use W_LSB encoding that uses a Sliding Window of width equal to SWW.
5. *W_LSB* encoding is used to compress those header fields whose change pattern is known. When using this encoding, the compressor sends only the least significant bits. The decompressor uses these bits to construct the original value of the encoding fields. The number of bits required to send the changes depends on the *SWW* of the Sliding Window, which is maintained by the compressor. All the SN values, which the compressor believes have not reached the decompressor, are to be used and are kept in the Sliding Window. Thus, larger the *SWW* will require larger number of bits to be send in the header format packets because only one value in the window should have those LSB SN bits for correct decompression.
6. *k and n:* The ROHC decompressor uses a "*k out of n*" failure rule, where k is the number of packets received with an error in the last n transmitted packets. This rule is used in the state machine of the decompressor to assume the damage of context and move downwards to a state after sending a negative acknowledgment to the compressor, if bi-directional link is used. The decompressor does not assume context damage and stays in the current state until k packets arrive with error in the last n packets. The k_1, n_1 values are used to assume dynamic context damage and k_2, n_2 to assume static context damage.
7. *MODE*: There are two bits in the header format packets to define the operation mode in which the compressor and the decompressor work. There are only three-operation modes, U, O and R mode defined in [1].

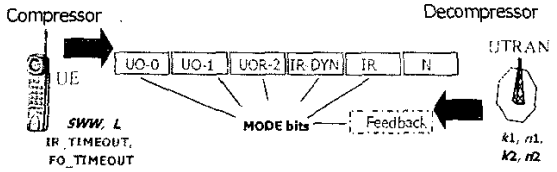

Figure 1. The Compression Parameters. They are localized in the compressor, in the decompressor and in the different header format packets. Note that at each end of the link both Compressor and Decompressor are present.

IV IMPLEMENTATION RESULTS

We have analyzed the results and investigated the role of some of the ROHC compression parameters for IPv6 flows. In order to evaluate efficiency, throughput and to estimate robustness, our implementation stores the statistics of each connection.

The parameters kept for each end of the link are: number of packets send and received, size of each packet send, size of header, size of compressed header; compressed header packet type; number of packets discarded due to ROHC, that we call it ROHC loss; and the number of packets discarded after ROHC, that we call it application loss. The header format packets of compressed packets are defined in [1], their use depend on the variation of the original header and the quality of the link.

We use IPv6/UDP/RTP video stream that sends packets in which the SN and TS field of the RTP header change with regular patterns. In our tests we have used the average payload size of 200 bytes and a RTT (*Round Trip Time*) of around 100ms. It may be noted that as IPv6 is used the UDP checksum becomes mandatory in our RTP application, adding 2 extra bytes to the ROHC compressed header.

A. Robustness

The BER of a radio link can generate consecutive errors in the compressed headers and can cause loss of context synchronization. A robust scheme should have error management and error detection schemes and it should also have fast recovery schemes to avoid prolonged losses due to relatively long RTT. ROHC uses CRC to detect these errors and discards the packets whenever CRC check fails. The errors in the links also cause the loss of packets carrying updating information. All these errors sometimes cause the loss of synchronization between the context of the compressor and the decompressor; this situation is known as context damage.

Context damage can occur in two cases. First, when CRC is unable to detect damage in the packet and then the information corrupts the context. Second, when an update takes place and could not reach the decompressor due to bursts of error.

The values of k and n parameters have a high influence on the robustness of ROHC because these parameters are used to detect context damage. In a contrary situation the errors in link can cause consecutive loss and the decompressor will incorrectly assume context damage depending on k. Detection of context damage whether correct or incorrect will make the decompressor go up to a state where it will unnecessarily discard large number of packets and will continue to do so until an update from the compressor arrives. Large number of packets will be lost when the value of RTT is relatively high. The result is worst in U-mode where feedbacks are not possible and packets will continue to get lost until the timer expires in the compressor to send updates.

The possible solution will be to take a large value of k and n but this implies more computational efforts and large storage space, as the compressor will need to know whether in the k packets out of last n packets, a CRC failure has occurred. A larger value of k will also mean that the decompressor will have to wait each time for k packets to know if context damage has occurred and this will cause all the packets to be lost during that time.

In figure 2 we have shown the results of our implementation for different values of k and BER in the three ROHC operation modes. As we can see in figure 2, the case is different in each operation mode. Though in all the modes the error rate is high for small values of k because when the value of k is small, the decompressor incorrectly assumes context damage and then starts discarding packets until an update packet arrives. This error rate decreases fast with the value of k because the probability of a number of consecutive packets being in error for a given BER, falls exponentially with the number of consecutive packets with error.

In U-mode the error becomes stable after a value of k; further increasing k does not make any difference. This happens because the decompressor in U-mode will have to wait until a timeout occurs, and whether it detects context damage early or late it makes no difference.

In O-mode and R-mode the error first decreases down to a minimum and then starts increasing with the value of k because as k increases the compressor has to wait for more CRC fails to occur before detecting a context damage and communicating it to the compressor through feedbacks. In a situation where BER is lower than 10-3, the error in O-mode does not rise much with the value of k because the probability of context getting damaged decreases with BER.

In R-mode the error is lowest and it does not rise much with the value of k because in R-mode the compressor always uses a 7 or 8 bit CRC in packets that update the context and hence, context damage is less likely in R-mode. In R-mode the compressor also uses some packets, which do not have any CRC bits, and they do not update the context, this also decreases the number of packets dropped and reduces the overall error as compared to other modes.

The values of k and n do not affect compression efficiency hence, these values should be chosen keeping only robustness in mind. In figure 2, we have taken a value of n equal to k+int(k/8) to allow for the case when erroneous packets pass through the CRC check, especially in header format packets with 3 bits CRC that detect errors only with a probability of 7/8. The value of k and n used to assume static context should be kept larger than that used for assuming dynamic context damage because probability of former is significantly low as compared to that of later.

Based on our results we suggest that in U-mode the value of k should be kept larger and it would be better if the decompressor does not move to the lower states even if context damage occurs and keeps trying to repair the context. In O-mode and R-mode the value of k may be choosen between 8 to 20 because the error is lowest between these values.

B. Compression efficiency vs. Robustness

Some parameters, which affect the robustness, also affect the compression efficiency. In some cases, compression efficiency is reduced when high robustness is achieved. It will be desirable to configure ROHC such that it performs at the best compression efficiency and with a high robustness. We

will use *Average Compressed header Length* (*ACL*) introduced by [3] as our benchmark for making comparisons for the compression efficiency.

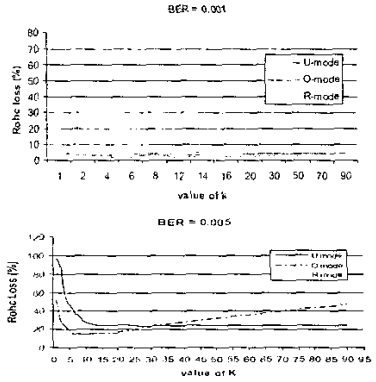

Figure 2. *Percentage loss of packets at ROHC layer for different values of BER, k and operation modes.*

In U/O mode the compressor maintains robust behavior by sending header format packets L times and in U-mode it also uses *IR_TIMEOUT* and *FO_TIMEOUT* to return to one of the initial compression levels and resends packets periodically with more information.

The values of *IR_TIMEOUT* and *FO_TIMEOUT* determine compression efficiency in U mode. Compression efficiency decreases when large headers are sent with more frequency i.e. when the value of *IR_TIMEOUT* and *FO_TIMEOUT* are low and the value of L is large. Thus, in order to have good compression efficiency a solution is to choose a value for *IR_TIMEOUT* and *FO_TIMEOUT* as large as possible and the value of L as small as possible. We assume that large values of L more than 10 will not be taken, as this will reduce compression efficiency significately. We show the effect of increasing *IR_TIMEOUT* and *FO_TIMEOUT* with L equals 5 in the figure 3.

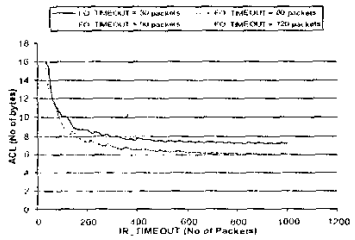

Figure 3: *Average compressed header length in U-mode with varying IR-TIMEOUT and FO_TIMEOUT.*

We have to determine the value for *IR_TIMEOUT* and *FO_TIMEOUT* to get the required robustness because, in U-mode the feedbacks are not possible and the probability of context damage, increases at high rate with time. The values of the two timers thus, depend on the probabilities of dynamic context getting damaged for *FO_TIMEOUT* and static context for *IR_TIMEOUT*. This probability of damage is lower for static context in comparison to that of dynamic context since, most of the time static information remains constant, the CRC used to detect error is larger (8 bits) and generally no update takes place for static values.

The compressor is designed to work in SO compression level most of the time thus, packets carrying just dynamic information are expected to pass through the channel most of the time. This makes them more vulnerable to error and moreover they carry a small CRC (3 bits) that may not be able to detect error sometimes and thus can cause dynamic context damage with significant probability as compared to static context damage. After comparing the cases for static context damage and dynamic context damage, we suggest that the values of timers be chosen such that dynamic information is refreshed much more often than static information.

We thus compare the robustness by looking at the values of error rate for different values of *BER*, *IR_TIMEOUT* and *FO_TIMEOUT*. We choose two values of *FO_TIMEOUT* that are 30 packets and 90 packets for our analyses, because it can be seen in the figure 3 that when *FO_TIMEOUT* is increased from 30 packets to 90 packets, the compression efficiency increases as *ACL* decreases but, increasing the value further from 90 does not decrease *ACL* significantly and a larger value of *FO_TIMEOUT* is not even desired because the probability of dynamic context damage is high and the compressor should keep going to FO compression level at an interval as small as possible. In figure 4 we show our implementation results and find that increasing the value of timeouts increases ROHC loss. However, decreasing those values not necessarily decreases ROHC loss always because it may mean that larger headers are sent more times and since large headers are more prone to errors, they will cause more loss of packets and thus, reduced robustness. Based on our results we suggest the value of *IR_TIMEOUT* to be taken between 200 and 400 packets if a value of *FO_TIMEOUT* be choosen around 90, and if the value of *FO_TIMEOUT* is choosen around 30 packets then even larger values of *IR_TIMEOUT* will maintain robustness as in figure 4.

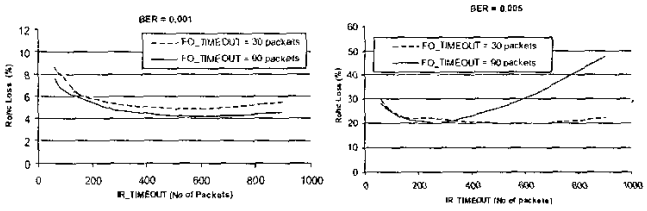

Figure 4: *Error rate at ROHC layer for different values of BER and timers in U-mode. When k and n equals 10 and 12 respectively*

In order to communicate the information efficiently in U/O mode, the compressor uses another technique in which it sends a particular header format packet L times. The probability of information being communicated thus increases while the value of L increases. The good value of L can make many cases of error diminish. For example if update is not communicated to the decompressor due to burst of error in the link then context damage happens and large number of packets are lost. The problem diminishes if a value of L is chosen which is sufficiently large that even if an occasional burst of error happens it is able to communicate the information.

Compression efficiency and throughput clearly decreases as larger headers are communicated more times. Thus value of L cannot be high. We have tested our implementation in U and O-mode with different values of L and *BER*, see figure 5. In U-mode when *BER* is high L could take the value of 5 to get the robustness against error also it can take the value of 9 but then the efficiency and the throughput will be affected. For a medium *BER*. L equals 3 can reduce the error and when *BER* is low L equals 1 is a good solution for efficiency and robustness. However when using O-mode the results are different and the vaues of L equal to 1 and equal to around 5 gives good performance. As when RTT is low, the error in the context can be quickly corrected by the feedback and the value for L may be taken as 1 but when RTT is long L may be taken as 4 to communicate the update with more confidence.

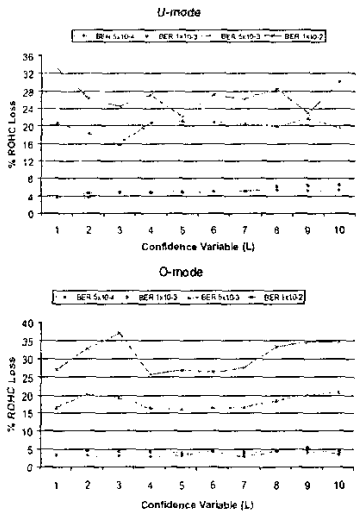

Figure 5. *Percentage loss of packets for different values of L. When k and n equals 10 and 12 respectively and IR_TIMEOUT equals 300 packets FO_TIMEOUT equals 40 packets.*

The *SWW* is another important parameter for compression efficiency as already shown by Wang et al. [3]. With a larger *SWW*, a ROHC compressor needs more bits to represent *LSB* bits as already seen in the previous section. Also a large *SWW* needs more storage space and more computing efforts. But, a larger *SWW* helps to improve robustness because even when there are consecutive lost packets equal to *SWW*–1, compression still works well.

In [3], Wang gives the effect of increasing *SWW* for U-mode and showed that in this mode ROHC works well if a *SWW* of 14 is taken for the case of IPv4/UDP/RTP. We continue to compare and analyze the effect of increasing *SWW* in O-mode and R-mode. The O-mode and R-mode are different from U-mode because they use feedback channel. In O-mode and R-mode, the decompressor can send the ACK with the SN value of the last successfully decompressed header. The compressor thus comes to know about the SN values that have reached the decompressor and shrinks the sliding window. However in O-mode, the ACKs are optional and normally are not sent hence, ACKs normally shrinks the sliding window only in R-mode.

In R-mode the number of bits available for SN in the SO packets is 6 in R-mode type-0 packet (R-0) and 7 in R-mode type-0 packet with CRC (R-0-CRC), unlike U/O mode packet type-0 (UO-0) that has only 4 bits. Hence, one might conclude that for R mode, the value of *SWW* can be chosen larger than 14 but, above a particular value of *SWW* the *ACL* will increase as it does in case of U/O-mode. However, in figure 6 we can see that increasing *SWW* window in R mode has no effect on compression efficiency because ACKs always shrink the sliding window and increasing *SWW* produces no effect. But, increasing *SWW* has effect on O-mode because ACKs are optional and generally not enabled, and in U-mode because ACKs are not possible.

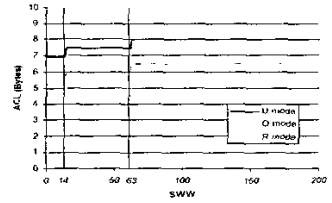

Figure 6. *ACL in different operation mode. Where L = 5 and IR_TIMEOUT equals 300 packets FO_TIMEOUT equals 40 packets.*

Thus, for R-mode a larger value of *SWW* should be chosen keeping only robustness in mind and that can be more than 14 without affecting compression efficiency.

We also find that the case of increasing *SWW* in U-mode and O-mode is different for IPv6 because U/O mode type-1 packets (UO-1) for IPv6, cannot use extension and to send more bits the compressor uses packets of type 2 (UOR-2) that have 6 bits for SN. As indicated in figure 6, the *ACL* increases if *SWW* is increased more than 63 because UOR-2 starts using extension to send SN because more than 6 bits are required and also extensions can be used in UOR-2 for IPv6.

V Conclusion

In a real situation, the compressed header packets over a radio channel such as UMTS will be confronted to a variable environment that sometimes could make compression fail. The use of correct compression parameters adapted to these changes will give better robustness and efficiency of ROHC. We have seen the different compression parameters that could affect the efficiency and robustness of ROHC. Based on our results we propose the values of ROHC parameters in order to improve compression performance in terms of compression efficiency and robustness for a radio link similar to that of UMTS.

References

[1] Bormann C., et Al., "RObust Header Compression (ROHC): Framework and four profiles: RTP, UDP, ESP and uncompressed", IETF RFC 3095, 2001.

[2] Holma, H., Toskala A., "WCDMA for UMTS", Wiley, England 2001.

[3] Wang, B., et. Al., "On Implementation and Improvement of Robust Header Compression in UMTS", IEEE 2002.