The NATORM Group Communication Mechanism for Intrusion Tolerant System

Jong-Whoi Shin† *and Chong-Sun Hwang*†

†Department of Computer Science and Engineering, Korea University

Summary
The ITS (Intrusion Tolerant System) aims at guaranteeing the continuity of essential services despite errors brought about by attackers or arbitrary faults. In this work, the ITS has replicated group members communicating with each other by messages. In this situation, group communication should satisfy the reliability and total-ordered properties to maintain consistency among the replicated members. However, it requires additional communication overheads. To apply ITS in the real world, such an overhead should be minimized so as not to greatly affect the target service. As a solution to the problem, we propose a newly defined ITS group communication scheme, called the NATORM (NACK-based Total-ordered Reliable Multicast scheme), which requires little communication overhead and satisfies the properties for ITS. The NATORM is a cost-effective group communication mechanism, based on NACK messaging to notice which message is not received. The NATORM showed that it requires little communication overheads when the system environment is stable. To evaluate the suggested mechanisms, modeling and simulation have been performed and developments achieved.

Key words:
ITS (Intrusion Tolerant System), NATORM (NACK-based Total-ordered Reliable Multicast mechanism), Replicated members, Total-ordered.

1. Introduction

Intrusion Tolerant Systems (ITS) are emerging as an approach to guarantee various internet services by dealing with unknown attacks, which are not treated by conventional prevention and detection technologies. However, the ITS is accomplished by the combinations of fault tolerant and security technologies, its architecture and operations are very complex [1]. Because of this complexity, the ITS can degrade performance of the target service. Therefore, to deploy the ITS to the real world, performance degradation due to intrusion tolerant service should be minimized. One of the critical points concerning the performance of ITS is communication process. Fig. 1 shows the overall procedure of the intrusion tolerant service under consideration. As shown in the Fig. 1, communications among replicated group members entail considerable costs since most of the messages should be delivered to all members reliably and orderly. Moreover, communications among group members affect service latency directly. There are many studies on group communication mechanisms for ITS. As a representative work, the SecureRing protocol has the basic concept of delivering messages reliably by making a logical ring in the broadcast domain [2]. This protocol satisfies the reliability and total-ordered properties, but it uses the channel inefficiently because group members must wait until a token is available for sending a message. Group communication protocols in ITUA and Rampart make use of an echo-reliable multicast mechanism. To deliver messages, an exchange of *init* and *echo* control messages is required [3-5]. These multiple steps— *init-echo-message-nack* —for delivering messages require high costs. One of the well-known reliable multicast protocols, SRM (Scalable Reliable Multicast), is based on the NACK mechanism for reliable delivery and supports efficient delivery in large-scale networks [6]. However, it does not consider the order of message arrival and assumes a large distribution network. Therefore, it is not suitable for adaptation to ITS. In this paper, we propose the cost-effective group communication system, NATORM, designed to be used among replicated members. Using the NATORM, we can satisfy both reliability and total-ordered properties, plus the requirement for low communication overheads. Through modeling and simulation, we have shown that the proposed communication mechanism is less costly than other reliable multicast protocols.

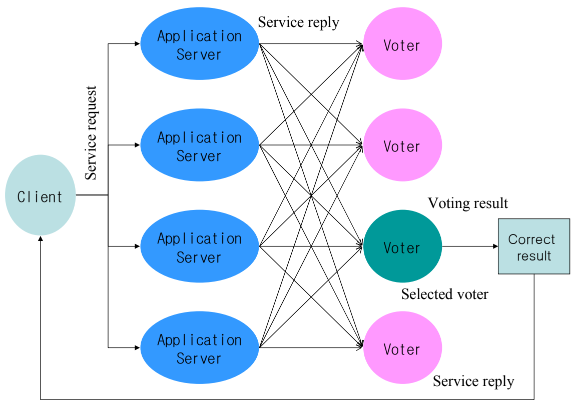


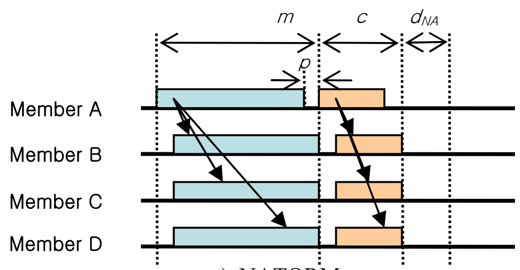
Fig. 1 Overall Procedure of the Intrusion Tolerant Service

Manuscript received September 5, 2006.
Manuscript revised September 25 , 2006.

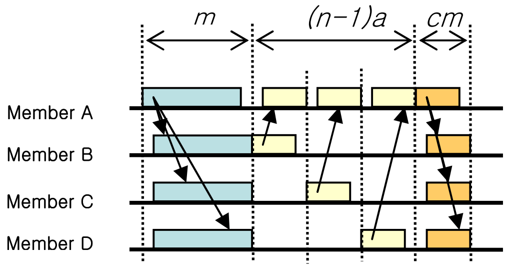
2. NATORM Mechanism

The NACK-based mechanism is suitable for ITS environments in terms of cost. Nevertheless, the NACK-based mechanism cannot guarantee the total-ordered property, while the NATORM solves this problem by adding the *confirm* control message after sending the message. The cost of adding *confirm* message is very cheap due to the fact that it is short—its length is variable according to the number of members.

2.1 NATORM in Normal Status



a) NATORM



b) ACK-based Group Communication

Fig. 2 NATORM and ACK-based Group Communication Mechanisms

m : duration for sending message
c : duration for sending *confirm* control message
d_{NA} : duration for waiting *NACK* control message
a : duration for sending *ACK* control message
cm : duration for sending *commit* control message
p : channel propagation delay
n : the number of members in group

$$T_{NATORM} = m + c + d_{NA} + 2 \cdot p \approx m + c + d_{NA} \quad (1)$$

$$T_{ACK} = m + (n-1) \cdot a + (n+1) \cdot p + cm \approx m + (n-1) \cdot a + cm \quad (2)$$

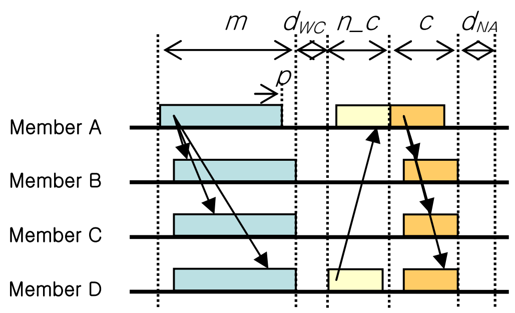
As shown in equations (1) and (2), from a cost perspective, the transmission time increases in proportion to the number of members *n,* under normal circumstances, since, in the case of the ACK-based mechanism, all receivers should send *ACK* to the sender as the number of members increases. On the other hand, the NATORM uses a fixed cost, regardless of the number of members. Consequently, if the number of group members is normal, the low-cost NATORM mechanism is more effective.

2.2 NATORM in Abnormal Status

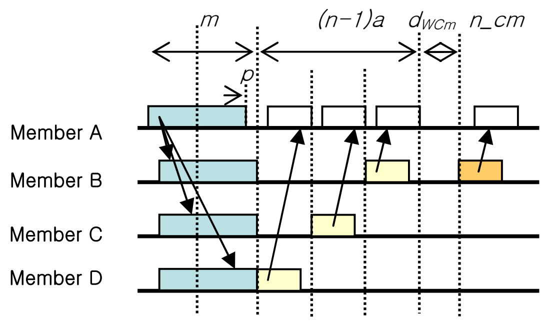
- *Error occurred in sender*

Fig. 3 presents an error occurring from the sender. As shown in Fig. 3a for NATORM, if an error occurs after sending a message, the sender cannot send the *confirm* message. In this case, one of the receivers requests a confirm message through the *confirm_NACK* control message. This action triggers transmission of *confirm* message from the sender. When the receiver has not received a message, though transmitting a message several times, the received message is discarded and its sender is eliminated from the group. In the case of the ACK-based mechanism (Fig. 3b), an error is found from the *commit* message, as with the *confirm* message of NATORM.

n_c : duration for sending *confirm_request* control message
n_cm : duration for sending *commit_request* control message
d_{WC}/d_{WCm} : duration for waiting *confirm/commit* control message



a) NATORM



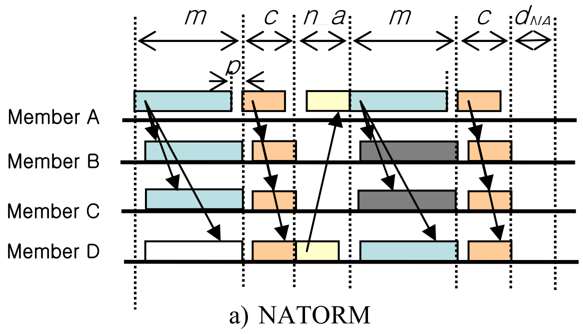
b) ACK-based Group Communication

Fig. 3 Status of error in sender

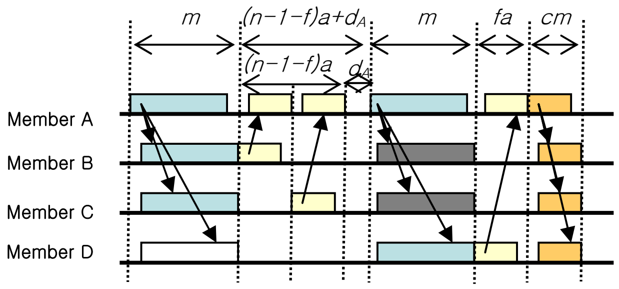
- *Error occurred in receiver*

For the NATORM mechanism (Fig. 4a), receivers can recognize the missing message by checking the sequence number in the succeeding *confirm* message. If the sequence number of a *confirm* message is out of order, the member sends the *NACK* packet to the sender directly. Once the sender receives the *NACK* control message, the sender resends the message using multicast, and finishes message transmission by sending the *confirm* message if the *NACK* control message is not sent again. As shown in Fig. 4b for the ACK-based mechanism, the sender can recognize the missing messages by *ACK* control message from receivers. When a sender receives all expected *ACK* packets, it sends the *commit* control packet. On the other hand, if *ACK* packets are not received from all expected receivers, then the sender waits for the expected *ACK*. When all *ACK* control messages have been received, the *commit* message is sent to complete message transmission. In both mechanisms, a maximum number of retransmission trials exists to avoid unlimited service delay and the retransmitted packet is simply dropped for members who have received it already.

n_a	: duration for sending *NACK* control message
d_A	: duration for waiting *ACK* control message
f	: the number of occurred fault
n_r	: the number of retransmission trials



a) NATORM



b) ACK-based Group Communication

Fig. 4 Status of error in receiver

$$T_{NATORM} = m + c + d_{NA} + n_r(m + n\_a + c) \quad (3)$$

$$T_{ACK} = m + (n-1)\cdot a + n_r(m + d_A) + cm \quad (4)$$

Equations (3) and (4) present costs, considering retransmission, based on equations (1) and (2). As regards cost, we found that the d_A value in (4) only increases in proportion to the retransmission number, while the n_a, c values for *NACK* and *confirm* messages in (3) increase in proportion to retransmission. In other words, while the NATORM cost is independent of the number of members, the ACK-based mechanism is proportional to number of member, since all receivers should send *ACK* control messages. Fig. 5 presents the NATORM algorithm.

```
Send_Group_Message
{
    send_data_to_group;
    wait_propagation_delay;
    send_confirm_message;
    wait_nack_duration;
}
Receive_Group_Message
{
  Switch(message_type)
  Case data_messsage:
    if sequence_number is out of order
    then notice_to_group_manager;
    else wait_confirm_duration;
      for until number_of_request_confirm_threshold
          if wait_confirm_duration timer expires
           //confirm message is not received
          then send_request_confirm;
                wait_confirm_duration;
           else break;
  Case confirm_Message:
      stop wait_confirm_duration timer;
      if sequence_number = old_sequence_number + 1
      then send_nack_message;
      else if sequence_number > old_sequence_number+1
      then notice_to_group_manager;
  Case NACK_Message:
     if number_of_retransmission> retransmission_threshold
     //retransmission up to the predefined number
     then notice_to_group_manager;
     else Send_Group_Message
}
```

Fig. 5 NATORM Algorithm

3. Simulation and Execution Analysis

In our simulated evaluation of the proposed mechanism, we analyzed the cost for one message transmission and one service transaction. We compared four multicast protocols: unreliable multicast, NATORM, ACK-based and echo-reliable multicast.

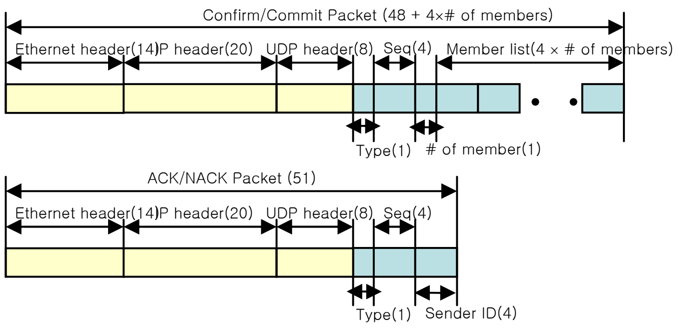


Fig. 6 Message Format

Fig. 6 shows the message format used for group communication. The message uses UDP and the data field is extended to use control message. And also, Table 1 shows the simulation parameters.

Table 1: Simulation Parameters

LAN	100Mbps, Ethernet	*init*, *echo*	60 (byte)
# of member	4, 7, 10 (members)	*confirm*, *commit*	4 member: 64, 7 member:76, 10member:88 (byte)
Request	72 (byte)	d_{NA}, d_A	2 (μs)
Service Reply	0~1500(byte)	*ACK*, *NATORM*	51(byte)

In this part, we describe 7 member case as the representative result. Fig. 7 shows the comparison of cost between the unreliable, NATORM, ACK-based and echo-reliable multicast protocols for seven members. The cost of NATORM is similar to the unreliable multicast protocol. Echo-reliable multicast requires an *init-echo* procedure before sending a message and is based on the NACK-based mechanism, but has no *confirm* control message, as in our proposed system. Fig. 8 shows the cost for one service transaction, with the following steps: service request from client, processing the request on application server, validating the reply on voter and returning the correct service reply to the client. Since several reliable deliveries are required in a transaction, the difference in the cost between each mechanism increases. The cost of NATORM is the lowest, except for the unreliable protocol. However, the unreliable mechanism does not guarantee the reliability of messages due to the lack of an acknowledgement.

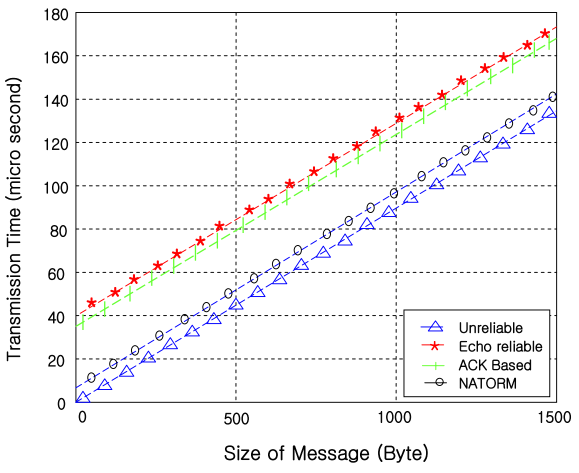


Fig. 7 Transmission time for one message

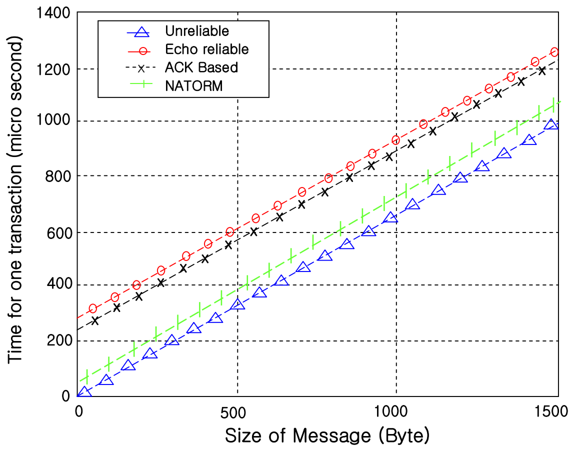


Fig. 8 Transaction time and Maximum throughput

4. Conclusion

To deploy the ITS in the real world, performance degradation due to intrusion tolerant service should be minimized. This requires relatively low costs and satisfactory ITS properties, such as reliability and total-ordered communication. Therefore, we propose the NATORM group communication mechanism as the most cost-effective mechanism in ITS. The NATORM is a NACK-based mechanism, which is suitable for ITS environments in terms of costs. Using the NATORM, we can satisfy both reliability and total-ordered properties, plus the requirement for low communication overheads. Through modeling and simulation, we have shown that the proposed communication mechanism is less costly than other reliable multicast protocols.

References

[1] K.S. Lee, C.T. Im, T.J. Lee, H.J. Kim and D.H. Lee, "SITIS: Scalable Intrusion Tolerance middleware for Internet Service Survivability", *2004 Pacific-Rim Conference on Multi-media(PCM 2004),* Nov. 2004.

[2] K. Kihlstrom, L. Moser, P.M. Mellia-Smith, "The SecureRing Protocols for Securing Group Communication", *Proceedings of the Hawai'i Int'l Conference on System Sciences*, Jan. 1998.

[3] Courtney Tod, Lyons James, et al., "Providing Intrusion Tolerance with ITUA", *Proceedings of the ICDSN 2002*, Jun. 2002.

[4] Cukier, M., et. al, "Intrusion Tolerance Approaches in ITUA", Supplement of the 2001 International Conference on Dependable Systems and Networks, Jun. 2001.

[5] Michael K. Reiter, "The Rampart Toolkit for Building High Integrity Services", *Theory and Practice in Distributed Systems*, LNCS 938, pp. 99-110. Springer, 1995.

[6] Floyd, S., Jacobson, V., Liu, C., McCanne, S., and Zhang, L., "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing", *IEEE/ACM Transactions on Networking*, Volume 5, Number 6, pp. 784-803, Nov. 1996.

Jong-Whoi Shin received M.S. degree in Computer Science and Technology from Korea University, South Korea, in 2001. He is currently studying for a PhD degree in the Department of Computer Science and Engineering, Korea University, South Korea. He is also working for the Korea Information Security Agency as a principal researcher. His research interests include security for mobile ad hoc wireless networks, ubiquitous sensor networks and intrusion tolerant systems.

Chong-Sun Hwang received the B.S. and M.S. degrees in mathematics from Korea University, South Korea, in 1966 and 1970, respectively, and a PhD degree in computer science and statistics from the University of Georgia, in 1978. From 1978 to 1980, he was an associate professor in University of South Carolina, Lander, USA. He has been a professor in the Department of Computer Science and Engineering, Korea University, South Korea. His research interests include distributed computing and mobile computing systems.