37

A model for the detection of the message stream delay attack

O'Connell, S. and Patel, A.
Computer Networks and Distributed Systems Research Group
Department of Computer Science, University College Dublin,
Dublin 4. Ireland, tel : +353-1-7062476, fax : +353-1-2697262.
E-mail : sean.oconnell@sse.ie, apatel@ccvax.ucd.ie.

Abstract

The message stream delay attack consists of an intruder deliberately delaying one or more messages being transmitted in one or both directions between two communicating computer hosts. This paper provides a breakdown of the different forms that this attack can take and outlines the benefit to an intruder in carrying them out. Furthermore, the current status of the ISO security related standards with regard this network based attack are discussed.

In order to counteract this attack, a model is presented which will allow a computer host to determine whether a received message has been deliberately delayed by an intruder during its transmission i.e. has it been sent in real-time or not. The basis of this model, is to provide a precise definition as to what is meant by real-time communications. However, unlike previous definitions of real-time data transmission, this model does not assume the availability of synchronised clocks for a number of important reasons, which are also listed. Rather, this model allows the clock devices associated with the various communicating hosts to drift positively or negatively. Finally, several important results are described as to the effectiveness of the model to counteract the various types of delay attack.

Keywords

Network security, message delay attack, real-time model, clock-drift, service delay.

1 INTRODUCTION

Throughout this paper, it is assumed that there are a number of initiator hosts which request an association with a responder host, in order to exchange data with it. Figure 1. No restriction is being placed on the relationship between the communicating hosts i.e. whether client-server, peer-to-peer or master-slave dependencies exist. Furthermore, each initiator has a separate independent association with the responder host. A number of other important assumptions are also relevant here. These include:

S. K. Katsikas et al. (eds.), *Information Systems Security*

- all communicating hosts maintain a monotonically increasing physical clock which, when read, will return the local time.
- the system administration will periodically change a host's system time in order to take into account winter and summer time-shifts, etc.
- an intruder can interfere with the network traffic in transit over any of the communication links being used by a particular host.
- the underlying network will only deliver uncorrupted data between the initiator and responder host systems.

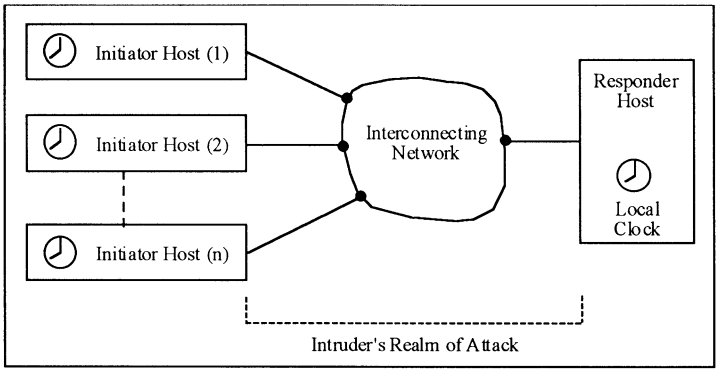

Figure 1 Data exchange between a number of communicating hosts.

2 THE MESSAGE STREAM DELAY ATTACK

The nature of this potential form of threat is to deliberately delay the transmission of one or more messages for a finite period of time, over some form of network link. There are three main forms of such a delay attack: single association delay, reordering through delay and suppress-forward. These are discussed below:

- **Single association delay** involves the intruder suppressing the transmission of a particular message. After a specific time interval has elapsed, the delayed message is re-issued into the network for subsequent reception by the destination host. The main benefits of this attack form include:

  a) Real-time operation prevented
  The initiator's application is no longer able to remotely operate on data in a real-time manner. For example, if a request to purchase oil stocks at a foreign source were suppressed and re-transmitted several hours later, the request could result in a purchase when the stock price had already changed significantly.
  b) Attack is undetectable
  If all the messages exchanged in both directions between the initiator and responder are delayed by the same time interval then this attack form is undetectable. The reason for this is that, in order to detect the delay in the first place, the average expected

network delay is required by both the initiator and responder. To determine this, either all possible network delays between the different hosts must be calculated, which would not be practical in a large environment such as the Internet (Wallich 1994) or the initial message exchange is accompanied with enough information to enable this network delay to be estimated. However, if the intruder equally delays the transmission of the initial messages, then a false network delay will be calculated. For example, if the actual network delay between the initiator and responder is 15 seconds, and both communicating parties understand it to be 50 seconds (as a result of the induced delays), then the intruder can delay a message for an extra 35 seconds (i.e. 50-15) without detection. This extra time will enable the intruder to perform an eavesdropping attack without detection.

- The nature of a **reordering through delay** attack involves the intruder selecting a point on the interconnecting network (called the network reference point) to carry out this attack. The transmission of a number of messages of different associations are delayed by varying amounts of time as they pass this point. In this manner, it is possible for a specific message sent by a particular initiator to be received, processed and replied to before any of the others are received at the same responder host. This attack is carried out by monitoring a specific association for any messages being transmitted (called the filtered messages). When one is detected, those message being sent on other associations that have not fully passed the network reference point are delayed by the amount of time it takes for the filtered message to be replied to. The delay experienced by the messages under attack will be proportional to the volume of filtered messages being issued and the amount of processing time required to service each such message at the responder host. The net effect of this attack is to reorder the transmitted messages such that those issued on behalf of a particular initiator host will be given a higher service priority over all other initiators. The main benefits of this attack include:

  a) Change in the speed of response
     The filtered messages will be guaranteed to be serviced before all other messages simultaneously issued. In this manner, the corresponding initiator will virtually receive replies from the responder as if it alone had the only association i.e. the speed by which responses will be returned will be increased. Consequently, the service speed provided to the other initiator hosts under attack will be reduced in the form of significant delays before a reply is received.
  b) Introduction of data inconsistencies
     If a number of outgoing messages require modifications to be made to the same piece of data then, changing the order in which these requests are received can affect the value of the accessed data to be maintained at the responder host. As a result, if two or more messages that operate on the same data are not commutative then this attack can cause inconsistencies in the value of the data maintained.

- Although analogous to the threat of a single association delay, the **suppress-forward** network attack (Gong 1992) is specific to the situation where synchronised clocks are being used. In the former method, each message is sent with an attached timestamp so that it will arrive (if not interfered with during transmission) within the time window

maintained at the receiver. The time window is used to determine if the message is recent. However, with the latter method, each message is sent with a genuine timestamp, but, the message will not be acceptable within the time window because, due to clock drift between the communicating hosts, etc. it will be regarded as being too early.

For example, if the time window is [17:05:00 to 17:15:00] (assuming a window tolerance of ±5 minutes) at real time 17:10:00, when a message arrives at the receiver with an attached timestamp of 17:18:00, then it will not be accepted, since it is too early to fall within the current time window. The incorrectly assigned timestamp is due to the fact that the initiator's local clock has failed and its time has 'gone ahead' of the local clock associated with the synchronising responder or the responder clock synchronisation process has failed and its local time has 'lapsed behind' the synchronised time associated with the other communicating initiator hosts.

In either case, the receiving host determines that the message must be post-dated and will reject it. An intruder can exploit this by being able to suppress the message transmission and then forward it to its original destination so that is appears 'recent' at the receiver (i.e. the message's timestamp falls within the receiver's current time window at the time of reception). For example, host A has become unsynchronised with the other hosts and, after a certain amount of time, it has gained an extra 10 minute difference from the actual real time (or the time maintained through distributed host synchronisation). When it sends a message with timestamp of 17:05:00, assuming a minimal network transmission delay (e.g. 1 second) and no intruder interference, it would arrive within the receiver's acceptance time window [16:50:01, 17:00:01], where the local correct time is actually 16:55:01. As a result, the message would not be accepted since its timestamp clearly does not fall within this time window.

However, an intruder can exploit this by being able to suppress the message transmission for up to 14 minutes and 59 seconds, and then forward it to the receiver. The local time would now be 17:10:00 and the time window [17:05:00, 17:15:00] thus allowing the message to be accepted. There are a number of obvious advantages to the intruder in carrying out this method of active network attack. These include:

a) Introduction of data inconsistencies
   If an initiator issues a message, and it is suppressed, then, after a time out period, the initiator will stop waiting for a reply. The initiator may assume that the message was lost and was therefore not serviced at all. However, when the intruder forwards the message onto the responder, it will be accepted as a 'recent' message, the change will be carried out to the specific data. If the initiator and responder hosts mirror their data contents then, under this attack, the initiator will have unintentionally introduced a data inconsistency at the responder's host site.
b) Real time operation is prevented
   By the very nature of suppressing the transmission of a message, the responder host is denied the ability to receive real-time data from a number of remote initiator sites. For instance, if a large amount of money involving currency conversion, is being exchanged between international banks, and the intruder manages to suppress the request for several hours, then the responder bank may have to use a lower exchange rate than would have been the case had the message not been suppressed.

3 STATUS OF THE ISO SECURITY STANDARDS

From the current list of standards given in (Omnicom 1995), ISO 7498-2 is the main standard being used by the designers of today's network security utilities and products. Although this standard recommends the use of timestamps to detect the message stream delay attack, it does not indicate:

- How timestamps for transmitted messages are to be used, generated and subsequently verified,
- What time scheme is to be adopted (e.g. political time or universal time),
- Whether clock synchronisation is necessary.

To address some of these issues the sub-committee ISO/IEC JTC1/SC21 have specified a service (i.e. Time Management Function) for the management of time in an open distributed environment. This document (N7961 1993), provides clear definitions for a number of important time and clock-related terms (e.g. the size of a timestamp). Furthermore, it describes three different clock synchronisation methods that are most commonly used to maintain a global time among a number of distributed hosts. However, N7961 fails to provide:

- A definition for real time communications, since this is closely linked to the detection of the message stream delay attack.
- An objective analysis of using clock synchronisation. For example, what are its practical limitations in a large distributed environment and the effective cost which will subsequently be incurred?

As this is the main ISO timestamp-related security document, it is clear that significant work is still required before the designers and manufacturers can incorporate an adequate mechanism to counteract the message stream delay attack into the new security products.

4 DEFINING THE MODEL

The threat of deliberately delaying a message is an important complex form of network security threat which must be counteracted, in order to avoid significant revenue losses being suffered by commercial organisations. This section presents a model, the aim of which is to enable communicating hosts to detect these different forms of message stream delay attack. However, in order to derive this model, a number of requirements must be met. These are briefly outlined below.

- The model should overcome exceptional conditions which may result in a received message loosing its timeliness. This may be caused by unexpected delay between the communication hosts due to a switch or router multiplexing the data traffic between different associations.
- The model should support the bi-directional exchange of messages and handle variable transmission delays for different sized messages. In addition, the use of synchronised clocks should be avoided. Clock synchronisation involves a number of interconnected

hosts trying to maintain a commonly agreed time value (with the aid of time servers and specialised protocols), which is subsequently used to set their individual clocks. There are a number of disadvantages associated with depending upon synchronised clocks. These include:

a) Although a significant number of clock synchronisation algorithms have been developed including those described in (Kopetz and Ochsenreiter 1987) and (Scheider 1986), the protocols and techniques (e.g. radio broadcast) used for distributing the time information are currently insecure. They are susceptible to various attacks including denial of service, replay and message integrity attacks. As a result, an intruder could easily induce a false time on one or more communicating hosts, thus removing the benefit of having synchronised time.
b) Clock synchronisation within an independent administrative domain can be assumed to be possible since a single administration can ensure that a common algorithm is operating normally on all its machines. However, in a large autonomously administered multi-domain distributed environment such an assumption cannot be made. Experiences with clock synchronisation in the DARPA networks show that even when time servers and clock synchronisation protocols are available, the hosts seldom maintain closely synchronised clocks (RFC-1128 1989). This is not due to the synchronisation algorithms used but, rather, system management and network disfunctionality. The results of a survey carried out over an extended period of time showed that, despite the fact that the most up-to-date algorithms were being used, the majority of the hosts end up being loosely synchronised with the Co-ordinated Universal Time (UTC) i.e. most clocks deviated with the time servers by between 1 - 13 minutes; 10% of the hosts actually differed from UTC by more than 13 minutes.
c) Most hosts that try to have synchronised clocks do so according to their geographical timezone or UTC time. The majority of these algorithms require the availability of a time reference which can act as a highly accurate source of time, for example, a caesium beam clock. However, these reliable clocks which tend to have a typical time loss resolution of 1μ sec per year (RFC-1129 1989), are very expensive and a number of them may be required in a large distributed environment.

4.1 Deriving the necessary equations

The main purpose of the model is to provide a meaning for real-time communications, which satisfies the above requirements. By real-time is generally meant that, when a message is sent to a remote host, it should be possible to determine whether it has arrived within a predetermined period of time. If not, then, it is said that the message was not sent in real-time. The aim of real-time is at best to prevent a network based intruder from being able to delay a message or at least to prevent a message from being delayed for more than a specific period of time without detection before arriving at the receiver. The approach taken to derive an alternative definition of real-time data exchange is to expand on an existing definition which is suitable for an environment where synchronised clocks are present. The reason for this is because a suitable definition was not found in the examined literature, where the dependency on synchronised clocks was removed. The definition being used in this paper is

called the Real-Time Consistency Property, while synchronised clocks are present. This is defined (Alford 1990) in the form of the following inequality equation:

$$T - \tau > \delta + \varepsilon \quad (1)$$

This equation assumes that the local clocks of any two communicating hosts are synchronised to within a time difference, ε. Furthermore, let δ represent the average end-to-end network delay when no failure occurs, T the time when the particular message is received and τ the local time at the sender which is attached to the message before transmission. If the value of τ does not satisfy this inequality, then the received message is treated as a late arrival.

The Real-Time Consistency Property, as described by equation (1), is dependant upon clock synchronisation being maintained between the communicating hosts and a generic over-estimate value being selected for the network delay, δ. However since δ is a gross over estimation of what it should be (thus giving an intruder the opportunity to delay a message more on high speed links than on slow ones) and a bound on the clock skew ε cannot be maintained when synchronised clocks are not present, this equation is inadequate. To overcome this, an alternative mathematical approach is required which allows the network delay to be variable and the hosts to maintain their own independent value of time, which is subject to clock drift, etc.

4.1.1 Calculating the service delay

Rather than considering the network delay, which only determines the time for a message to traverse a particular association, it is more accurate to consider the service delay which is denoted by Serv_Delay. This alternative value provides a summation to assemble and dissemble the exchanged messages, the network delay and the delay experienced while interfacing with the network from the host system. The approach used to calculate the service delay was adopted from the Network Time Protocol (NTP) specified in (RFC-1129 1989), which is a specialised protocol that aims to maintain synchronised clocks among a group of interconnected hosts. Although the method is based upon calculating the network delay, it is equally valid to re-apply the NTP equations to determine the service delay, provided the calculation is carried out within the service layer of the communications stack shared between the communicating hosts. For example, within an OSI environment, the Application layer would be appropriate. The NTP equation for calculating the network delay is given by:

$$\text{Network\_Delay} = \left(\frac{[t_4 - t_1] - [t_3 - t_2]}{2} \right) \quad (2)$$

where:

t_1 is the time at the initiator host when the message was submitted for transmission,
t_2 is the time at the responder when the message was received in full,
t_3 is the time at the responder when a reply is submitted for transmission,
t_4 is the time at the initiator host when the reply is subsequently received in full.

Furthermore, due to the fact that the service delay value will vary for different sized messages transmitted, it is important to initially calculate Serv_Delay for 1 bit of data, and

then multiply the respective delay value by the size of subsequently received messages in order to determine the actual service delay for a particular transmitted message. This bit representative value is denoted by Serv_Delay/bit and is calculated using the following equation which is adopted from equation (2):

$$\text{Serv\_Delay/bit} = \left(\frac{(t_2 - t_1)/Tx\_message(bits) + (t_4 - t_3)/Rcv\_message(bits)}{2} \right) \text{secs} / bit \qquad (3)$$

Deviations in the service delay value can be caused by multiplexing associations in the underlying network infrastructure, transmission and queuing delays at the initiator and/or responder due to the presence of multiple concurrent associations that are multitasking on one or both communicating hosts, etc. The possibility of these deviations occurring must be taken into account for the two communicating hosts and are denoted respectively as Δ Serv_Delay(I) and ΔServ_Delay(R). Due to the unpredictability of the network infrastructure it is not clear how these values may be calculated. However, an upper-bound approach is adopted, such that if this bound for a particular message transmission is exceeded for a particular association, then its timestamp is deemed invalid.

4.1.2 Handling the clock drift

Differences between the communicating hosts in terms of the values of their system clocks and the constant presence of relative clock drift must also be taken into account. The clock offset is denoted by Clock_offset and is defined for a particular association as the difference between the initiator's local clock value and the responder's local time at the instant the association is being established. This value may be positive or negative, depending on the time zones occupied by the various hosts. The clock offset can be calculated using the following NTP equation:

$$lock\_offset = \left(\frac{[t_2 - t_1] - [t_3 - t_4]}{2} \right) \text{ (secs)} \qquad (4)$$

where the values t_1, t_2, t_3 and t_4 are as before.

The relative clock drift denoted by ΔClock_offset, is defined as the maximum allowable time by which the value of Clock_offset is allowed to drift positively while an association exists. For example, the value of ΔClock_offset could be set to 1 second. This deviation may be caused by clock drift occurring at the initiator and/or the responder hosts. As a result, if the value of ΔClock_offset exceeds the maximum allowable deviation then, either the association must be terminated or a new clock offset value needs to be calculated (i.e. the association is re-initialised).

4.1.3 Representing the timestamping values

All processes communicating with one another in an open interconnected environment generally use the Abstract Syntax Notation No 1 (ASN.1), which is specified by ISO in (ISO

8824 1987), to represent the data being exchanged. Within this standard, it is recommended to send a timestamp or the local host time in either the form of a Generalised Time or Universal Time Code (UTC) data type. However, the method used to calculate the clock offset and service delay value requires that each timestamp (i.e. t_1, t_2, t_3, etc.) must be sent in the form of a seconds count with a high level of granularity. The granularity required is in the order of 1/1000 sec which based on the NTP time scale, that represents the local time as the number of seconds which has elapsed since 00:00 January 1st 1972. The two ASN.1 defined types for handling the distribution of time, are not suitable for supporting this time scale, since the Generalised Time ASN.1 type, although having the required granularity, represents time as a textual string (i.e. YYYYMMDDHHMMSS.SSS), while the UTC type only has a maximum granularity of 1 second. To overcome this issue, the definition of a timestamp, given by ISO in the Time Management document (N7961 1993), is adopted which recommends that each timestamp should be represented as a 64 bit string. Such timestamps will have a validity period of 600 years and a granularity of 1 nanosecond. As a result, the ASN.1 BIT STRING type is used to exchange the various timestamps. In addition, to avoid the use of a complex timestamping mechanism ASN.1 structure to carry the related data, including Serv_Delay and ΔServ_Delay(I or R), the BIT STRING type is selected to represent their values. This does not increase the level of security provided but, allows the amount of processing time the receiver must spend on extracting the data to be kept to a minimum, so that a timestamp received in real-time is not flagged as being invalid.

4.1.4 The definition of real-time communications

From equation (1) it is now possible to define the equations for determining whether a message was transmitted in real-time or not. For messages of q bits in size being sent by the initiator and subsequently received at the responder host, the equation is given by:

$$T(R) - T(M) \leq \text{Clock\_offset} + [\Delta\text{Clock\_offset} + \text{Serv\_Delay/bit} \times (q) + \Delta\text{Serv\_Delay(I)} + \Delta\text{Serv\_Delay(R)}] \quad (5)$$

and the timestamp acceptance window range for the message's timestamp, T(M), at the responder host is expressed in terms of its local system time, T(R), by the following inclusive limits:

$$T(R) - \text{Clock\_offset} - \varphi \leq T(M) \leq T(R) - \text{Clock\_offset} \quad (6)$$

where $\varphi = [\Delta\text{Clock\_offset} + \text{Serv\_Delay/bit} \times (q) + \Delta\text{Serv\_Delay(I)} + \Delta\text{Serv\_Delay(R)}]$

Furthermore, for replies being sent back from the responder host to the initiator, a separate equation is required. The reason for this is that equation (5) will not hold true in the case of a negative value for Clock_offset, and will not allow for the detection of a real-time message transmission in the case of a positive value for Clock_offset. As a result, in order to determine the real-time transmission of the replies sent by the responder, the following equation must be used instead:

$$T(M) - T(I) \geq \text{Clock\_offset} - [\Delta\text{Clock\_offset} + \text{Serv\_Delay/bit} \times (q) + \Delta\text{Serv\_Delay(I)} + \Delta\text{Serv\_Delay(R)}] \quad (7)$$

The timestamp acceptance window range for the reply's timestamp, T(M), is expressed by the following inclusive limits, where T(I) is the local system time at the initiator host when the reply is received:

$$T(I) + Clock\_offset - \varphi \leq T(M) \leq T(I) + Clock\_offset \quad (8)$$

The reason for the inequality sign in these equations is to allow for a maximum deviation in the clock offset and service delay to take place during a message exchange and still enable it to be acceptable at the receiving host. For purposes of clarity, the equations, (5) and (7), derived for detecting real-time communications are referred to hereafter as the Real-Time Conditional equations. To understand how these equations work, consider the case of a negative time difference between two communicating hosts, A and B, with the following configuration:

Host A:	**Host B**:
ΔServ_Delay(I) = 1 sec	ΔServ_Delay(R) = 1.5 sec
ΔClock_offset = 1 sec	ΔClock_offset = 1 sec
Serv_Delay/bit = 0.0006 sec	Serv_Delay/bit = 0.0006 sec

The clock offset between A and B is -100 secs i.e. the local host times are 600 secs and 500 secs respectively (assuming small values for the purpose of representing local times). When the initiator sends an message of 80 bytes to the responder with an associated timestamp value of 600.000 secs, if the message arrives at the responder latter than 503.548 secs (from equation (5)) it will be flagged as being invalid. An alternative way of examining this would be, when the message arrives at the responder, a timestamp acceptance window is calculated using equation (6). If the message's timestamp falls within the inclusive range of this window, then it is accepted. For example, if the message arrives at time 503.548 secs (assuming a maximum deviation in the service delay and clock drift parameters), then its timestamp would be accepted since it falls within the calculated timestamp window of [600.000, 603.548]. However, if the message was delayed during transmission for a further 1 second, the timestamp window calculated when the message is subsequently received at 504.548 secs would be [601.000, 604.548], a range within which the timestamp of 600.000 does not fall and therefore the message would be flagged as having an invalid timestamp.

When the responder sends back a reply of 160 bytes to the initiator with associated timestamp of 505.000 secs, if it is received before 608.098 secs, (using equation (7)) then the reply is accepted. Alternatively, if the received timestamp falls within the timestamp window calculated by the initiator (using equation (8)) upon receipt of the reply, then it is also accepted as being sent in real-time i.e. if the reply arrives back at time 608.098 secs, then the timestamp acceptance window for the message will be [505.000, 508.598]. As a result, the timestamp is still acceptable within this range. However, if the reply is delayed by 0.1 sec and, therefore, arrives instead at 608.198 secs, then the timestamp received with the reply will not be acceptable, since it will not fall within the range of the resultant new timestamp window i.e. [505.100, 508.698]. The above timestamp validation process also holds for situations between communicating hosts where the Clock_offset ≥ 0.

4.1.5 Restricting the possibility of a deliberate delay

In an ideal situation, the maximum time an intruder has for delaying a particular message is given by the summation of ΔClock_offset + Serv_Delay/bit × (q) + ΔServ_Delay(I) + Δ Serv_Delay(R) secs, which from the example above, is 3.5 + [Serv_Delay/bit × (q)] secs. In reality, however, the possibility to delay a message for such a period may not be so high because:

- Unpredictable variance may occur in the underlying network traffic and the workload on the communicating hosts will vary, which will adversely affect the service delay. Therefore, it is not predictable that the value of time indicated by ΔServ_Delay(I) or Δ Serv_Delay(R) can be used to delay a particular message. Furthermore, the value of Serv_Delay/bit × (q) is calculated on a per message basis and does not provide for any variance. As a result, its value is not over estimated to provide the possibility for the intruder to delay the message.
- There is a wide range of different manufactured clock devices for the computer hosts which will result in different graphs for their associated clock drift. Consequently, although an intruder may be able to delay a message for 1 sec (i.e. the value of DClock_offset) when the association is established, the longer the association remains open, the relative clock drift increases. Therefore the possibility of delaying messages for 1 sec is gradually reduced, eventually becoming zero as the lifetime of an association increases.

From the above discussion, it is evident that the maximum predictable delay an intruder could introduce into this scheme is the value of DClock_offset.

4.2 Handling exceptions

The main exception, to be handled by this model for real-time communications, is related to an invalid timestamp being received such that it fails the real-time check. This may be caused by unexpected clock drift, greater than normal service delay, etc. To recover from this problem at the responder host, a challenge is issued to the initiator, which will result in the service delay and clock offset values being re-calculated. This is done by re-sending the message which is not subject to real time checks at the responder. When the reply is returned, with values t_2 and t_3 attached, the initiator is able to re-calculate the service delay and clock offset values that will enable the Real-Time Conditional equations to be used for subsequent message exchanges i.e. the association is re-initialised. In order to inform the responder host of the new values of Clock_offset and Serv_Delay, these must be sent with the next outgoing message.

If in the event that a reply message fails the real-time check at the initiator host, then the next outgoing message is not timestamped to indicate to the responder that the association is being re-initialised. If after re-initialisation, the real-time check is still not satisfied, then the association is deemed unrecoverable and terminated immediately by the communicating hosts.

5 IMPLEMENTING THE MODEL

In order to test the model presented in this paper, the Real-Time Conditional equations were integrated into the Association Control Service Element (ACSE) and Remote Operations Service Element (ROSE) which form the main communications facilities of the Application Layer of the OSI stack. A sample Remote Operations (RO) based service was developed using these service elements which would allow messages of 0.1 k to 4k bytes in size to be interchanged over a number of associations that are active between an initiator and responder host. Furthermore, it is important that the scheme was tested over long distance network links which would involve a significant amount of international data traffic. Consequently, two machines were chosen to support the testing carried out. Test machine A (the initiator - SUN Sparc IPX) was based in Dublin, while the second machine B, (the responder - SUN Sparc 5), was located in Copenhagen; both hosts were interconnected via TCP/IP. Responder B was acting locally as a file server, which meant that its value of DServ_Delay was set higher (i.e. 2 secs) than A, which was set to 1 sec.

One of the tests carried out was to determine the impact the real-time checking procedure would have on the operational performance of an RO service when varying message sizes were being transmitted. The test was run twice, initially with the real-time checks turned off and secondly with them turned on. The amount of time required for a message to be issued to the responder and a subsequent reply received by the initiator was measured on both occasions. From this test, it is possible to estimate the impact this model will have on a data exchange service for varying message sizes. The impact in terms of the percentage deviation in the operational speed of the RO service is illustrated in Figure 2.

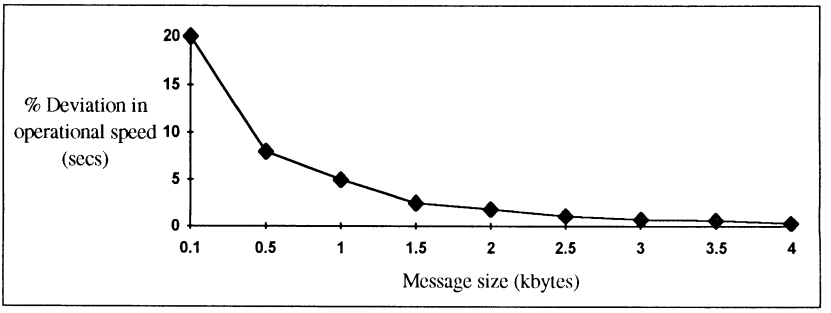

Figure 2 The recorded deviation in the operational speed of a test RO service.

From this graph it is clear that the inclusion of the real-time check can slow down the RO service speed from 20% to 0.3%. The actual impact encountered can be attributed to the small amount of processing time required to perform the check at the receiving host and the varying transmission, message scheduling delays incurred due to sending the timestamp information with the outgoing message. As a result, it can be determined from the above graph that the average throughput operational speed of the RO service is only reduced to 95.5%. A more comprehensive evaluation of the model and its use as part of a timestamping mechanism and protocol, which also allows for the detection of the replay and reordering attacks, is given in (O'Connell, 1995).

6 DISCUSSION AND CONLUSIONS

The most important and fundamental conclusion that can be drawn from this paper is that the use of synchronised clocks is no longer necessary, in order to accurately detect the deliberate delay of messages interchanged over a network. The reason for this is that a model of real-time communications which counteracts this form of attack without the use of synchronised clocks, is presented.

From the description and implementation of the model, there remain two forms of delay attacks which are not adequately counteracted by the model presented. The first is the two-way single association attack. One way to resolve this issue is to record all the service delays between the various communicating hosts. However, this is an impracticable solution in a working environment like the Internet, which consists of over 2.5 million interconnected machines (Wallich 1994), since the interconnecting transit delay values are dependent upon a large number factors including, the volume of data traffic being sent, the type of route chosen for each association, the number of networks to be crossed, etc. Further study of this problem is required.

The second threat involves carrying out a reordering through delay attack. However, the amount of time by which the intruder can delay the affected messages is limited by the use of the Real-Time Conditional equations. To adequately counteract this threat, a buffer is required to record the previously received timestamps in order to determine if subsequent messages have been reordered or not. This is further discussed in (O'Connell, 1995).

Despite these weaknesses, the model presented in this paper, provides a basis for understanding the meaning for real-time communications without the need for synchronised clocks. In addition, the use of over-estimates is avoided for determining the service delay, which is calculated on a per-association basis. Finally, although there is currently no ISO standard detailing the meaning of real-time communications, it is important that, in a world which is rapidly becoming computerised, research of this nature is carried out in concert with the international standards bodies in order to provide guidance in the formulation of emerging standards, and to identify any shortcomings in the exiting ones. Otherwise, the security solutions to be developed will be proprietary and therefore hinder the development of an open computerised international market place.

7 REFERENCES

Wallich P., (1984) Wire Pirates. *Scientific American,* March.

Gong L., (1992) A Security Risk of Depending on Synchronised Clocks. *Operating Systems Review*, Vol. 26, No. 1, pp. 49-53, January.

Omnicom, (1995) Catalogue of ISO Standards for OSI, LANs, etc. *Omnicom PPI Ltd*, May.

N7961, (1993) Open Systems Interconnection, Data Management and Open Distributed Processing, Time Management Function. *ISO/IEC JTC 1/SC21* WG4 Meeting, June.

Kopetz H. and Ochsenreiter, W., (1987) Clock Synchronisation in Distributed Real-Time Systems. *IEEE Transactions on Computers*, Vol. C-36, No. 8, August.

Scheider, F., (1986) A Paradigm for Reliable Clock Synchronisation. *Proceedings of the Advanced Seminar on Real Time Local Area Networks*, pp. 85-104, April.

RFC-1128, Mills D., (1989) Measured Performance of the Network Time Protocol in the

Internet System. *DARPA Network Working Group Report,* University of Delaware, US.
RFC-1129, Mills D., (1989) Internet Time Synchronisation - The Network Time Protocol. *DARPA Network Working Group*, University of Delaware, US.
Alford M., Ansart J., Hommel G., Lamport L., Liskov B., Mullery G. and Schneider F., (1990) Paradigms for Distributed Programs. Lecture Notes in Computer Science, Chapter 8, *Springer-Verlag,* Berlin, ISBN 0-38715-216-4.
ISO 8824, (1987) Specification of Abstract Syntax Notation One (ASN.1). *ISO,* Information Processing Systems - Open Systems Interconnection.
O'Connell S., (1995) The Development of a Timestamping Mechanism for Remote Operations in an OSI Environment, *M.Sc. Thesis*, Department of Computer Science, University College Dublin, Ireland.

8 BIOGRAPHIES

Sean O'Connell received his B.Sc. (Hons.) in Computer Science from University College Dublin (UCD) in 1991. He worked as research assistant with the Computer Networks and Distributed Systems Research Group (CNDSRG), located in UCD, until 1994. During this period he was involved in the AIM project SEISMED and various internal security related projects. He joined Broadcom Eireann Research Ltd. in 1994 where he was responsible for the research work carried out in several security projects in EURESCOM and RACE. Since April 1995, he has been working at Software and Systems Engineering, where he holds the position of Software Engineer in the Secure Messaging Group. His interests include cryptography, international communications standards, real-time systems, network management and security evaluation techniques. He has recently submitted a thesis for a M.Sc. in Computer Security to UCD.

Ahmed Patel received his M.Sc. and Ph.D. in Computer Science from Trinity College Dublin in 1977 and 1984, respectively. From 1978-82, he was responsible for developing the Irish Universities Data Network and from 1982-85 he developed EuroKom computer conferencing and electronic mail service used by R&D projects in Europe. At UCD he is a lecturer in Computer Science, and head of CNDSRG, and centre director of Teltec Ireland. He is involved in various multi-national R&D projects in ESPRIT, RACE/ACTS, INFOSEC, AIM, COST and the Irish national IT and Telecommunications programmes. His main research interests include network management, security, protocols, performance evaluation, intelligent networks, CSCW and open distributed processing systems. He has published many technical papers and co-authored two books on computer network security and one book on group communications. He is a member of the Editorial Advisory Board of the Computer Communications and Collaborative computing Journals.