Preprint

The Modulated-Input Modulated-Output Model

Ira S. Moskowitz and *Myong H. Kang*

To Appear:
Proceedings of IFIP WG 11.3, Ninth Working Conference (Database Security), 13-16 August, 1995. (this version 7/6/95)

CONTACT:
Ira S. Moskowitz, Information Technology Division, Center for High Assurance Computer Systems, Mail Code 5543, Naval Research Laboratory, Washington, DC 20375.

Myong H. Kang, Information Technology Division, Center for High Assurance Computer Systems, Mail Code 5542, Naval Research Laboratory, Washington, DC 20375.

E-MAIL:
moskowitz@itd.nrl.navy.mil mkang@itd.nrl.navy.mil

The Modulated-Input Modulated-Output Model

*Ira S. Moskowitz & Myong H. Kang**
Information Technology Division, Mail Code 5540
Center for High Assurance Computer Systems
Naval Research Laboratory
Washington, D.C. 20375

Abstract

In this paper we discuss why message acknowledgements are an appropriate engineering approach to meet system functionality. The data replication problem in database systems is our motivation. We introduce a new queueing theoretic model, the MIMO model, that incorporates burstiness in the sending side and busy periods in the receiving side. Based on simulation results derived from this model, we show that the buffer requirements from M/M/1/N queues are too optimistic.

*moskowitz@itd.nrl.navy.mil mkang@itd.nrl.navy.mil

1 Introduction

In [6, 9] Kang and Moskowitz described a method for reliable, high performance communication between a low security class (Low) and a high security class (High) by means of the NRL Pump concept, or, more simply put, the Pump. Initially we developed the Pump as a communications interface between Low and High in NRL's replicated architecture database project SINTRA[1] [7, 8]. Since the writing of [6, 9], Kang, Moskowitz, and Lee have advanced the Pump concept to deal with multiple Lows and multiple Highs [10].

The Pump uses message ACKs for reliability of communication between Low and High. While these ACKs form a covert channel from High to Low, the probabilistic nature of the Pump ACKs keeps channel capacity within specified bounds without penalizing performance. Also, the Pump uses a "handshake" protocol so Low does not send a new message until the previous message has been ACKed[2]. Thus, the Pump also uses ACKs for flow control by slowing down message arrivals at an intermediary buffer between Low and High.

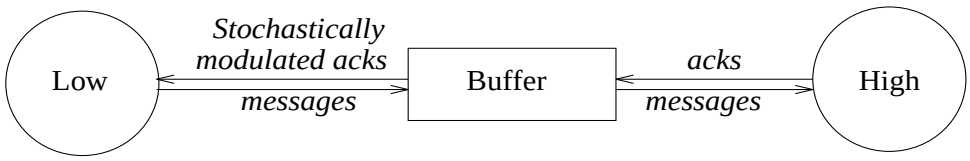

Figure 1: The Pump

McDermott presented work on the big buffer [13] as a Pump alternative. We are taking this opportunity to further contrast the Pump approach with his.

1.1 Why acknowledge?

This question might better be phrased as—Why not acknowledge? We believe that the burden of proof is upon those who advocate *not* using ACKs. The standard in computer communications is *to* use ACKs. Of course, when we add security to the picture we have the problem that ACKs can cause an illicit information flow. Thus, the standard use of ACKs must somehow be modified. The ACKs give us the reliability that our message

[1] Secure Information Through Replicated Architecture.

[2] The Pump uses timeout for the NAK.

has been safely received by High. Even if the transmission medium insured 100% error free transmission (fiber optics come close [2]), errors can occur in the systems themselves. How does Low know that High is ready to receive messages? How does Low know that a message arrived at High? These are just some of the reliability issues that concern us. Therefore, there must be some ACK/NAK passed from High back to Low.

The ACKs may also be used for flow control (as in the Pump). In this paper we will not discuss reliability issues in detail. Instead, we focus on one particular objection we have to ACK-free transmission; that of buffer management (size) under the paradigm of no ACKs. Buffer size is very important because of different "rates" between Low and High (High's ability/inability to receive messages can cause buffer overflow under the no-ACK paradigm).

1.2 Motivation: Replicated Architecture Database

The SINTRA approach uses physical separation and data replication as the primary protection mechanisms for a database system that provides both high assurance and multilevel security. Low level data is replicated in the high level DBMS so it will be available to high level users. Thus, messages must pass from Low DBMS to High DBMS, as shown in figure 2.

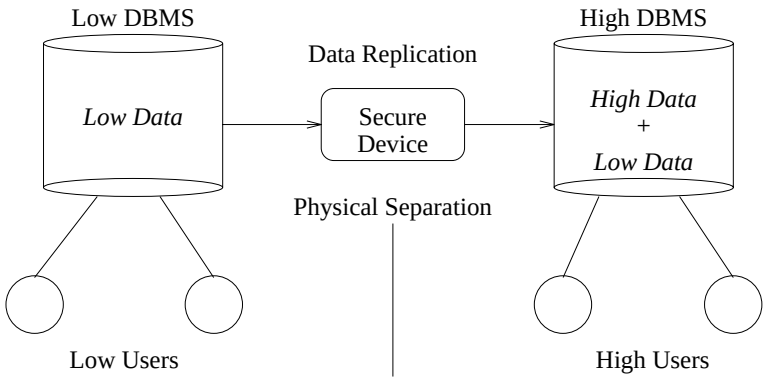

Figure 2: The SINTRA Architecture

DBMSs and Replication servers are reliable and recoverable. However, the "secure device" in figure 2 may or may not be reliable. Not satisfied with read-down[3] nor

[3]Similar to blind write-up, see [6, 9] for a discussion.

blind write-up (discussed in section 1.3), which do not provide reliability, we developed the Pump as a secure device. Since DBMS, replication server, and the Pump provide reliability and recoverability, the whole data replication path becomes reliable and recoverable.

Let us consider DBMSs (i.e., Low and High). Sometimes, Low DBMS has lots of data to send within a short period of time (e.g., sending large updates). During some other periods of time, Low DBMS has very little data to send (i.e., Low has bursty and normal periods). The same situation may apply to high DBMS (e.g., during some period of time many users are active, thus High may be in a busy state). Hence, High may have normal and busy periods[4]. To model this situation, as we will show, an M/M/1 queueing model is not adequate.

1.3 Blind write-up (no ACK transmission)

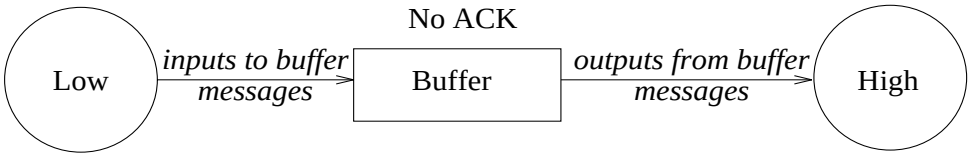

Figure 3: Blind Write-Up

Other approaches, e.g., [2, 13] or [1, section 5.3], also use an intermediary buffer between Low and High, but keep current High activity from being reflected down to Low ([2], [1, section 5.3] does allow a flow down in case of buffer saturation). Obviously, these approaches have no covert channels (unless the buffer is saturated often). We refer to these approaches as the "blind write-up" approaches since they do not let Low have ACKs. However, these models offer no assurance that a message was successfully transmitted and they do not offer flow control. McDermott's [13] (big) buffer sends ACKs down to Low when a message has been successfully placed in a buffer between Low and High. If the buffer is full, the new message will overwrite an old message. His model rests on the assumption that a large enough buffer can be chosen so that the probability of a message being overwritten before it has been serviced by High is negligible. Note that we still consider McDermott's method in the no-ACK (blind write-up) class because his buffer ACKs, unlike the Pump's, do not mimic true ACKs from High to Low. Goldschlag [2, section 3], [1, section 5.3] places a (big) buffer between

[4]An analogy is how an office building elevator services waiting people at 2:00 pm and at 5:30 pm.

Low and High, but a new message will overwrite an old message if the buffer saturates. However, downgraders can be added [2, section 4], to send a warning to Low if the buffer becomes full. Without this warning, Low does not moderate the message arrival rate at the buffer. The Pump, however, reduces the rate of arrivals from Low to the buffer so that Low's input rate roughly matches that of High's output (service) rate.[5] In contrast, these blind write-up models assume that (1) Low can successfully transmit a message to the buffer, and (2) a sufficiently large buffer can queue up enough messages going to High, so that no messages are lost; thus, blind write-up assumes messages successfully travel from Low to High. We strongly take exception to this. We feel that Low must know that its message got to High for reliability and flow control (which is itself a part of reliability).

McDermott [13] was the first, in the security community, to perform a serious queueing theoretic analysis of buffer management. He used an M/M/1 based model for his analysis. This model is not sufficiently rich to incorporate burstiness [5, 4, 12, 14, 16] and, aside from reliability issues, M/M/1 gives too optimistic estimates of sufficient buffer size. Note that Goldschlag also assumes that a properly sized buffer can be used. The communications community has done bursty analysis (on the transmitter end) by using a Markov Modulated Poisson Process (MMPP) [5], or Lo-Hi/On-Off type models [12, 16]. They have obtained closed form solutions for various terms of interest (such as mean queue size [16]). To a communications engineer, lost messages are not a problem because there is always some high layer (with respect to the ISO layering scheme) ACK to let the sender know that a message was lost or garbled and that it should be retransmitted. Since knowing if messages are lost (and therefore unrecoverable) rather than closed form solutions is our concern, the MMPP does not convey enough detail for our needs because it does not take High's behaviour into account. Therefore, we have extended the model.

The main contribution of this paper is our discrete extension of MMPP/G/1 type queues, to the Modulated-Input Modulated-Output (MIMO) model. Our studies of the MIMO model have only reinforced our belief that unacknowledged transmission from Low to High is unacceptable because the MIMO model shows that no matter what size buffer is used it is always possible to find an example where the Low rate is less than the High rate but the buffer will still fill. We assume that each message sent from Low to High is important. This need not be the case if we are sending video or voice, for example, but it is certainly the case for many types of data transfers, e.g.

[5] Input and output are relative to the buffer.

database updates, file transfers, etc.[6] Further, if our input rate is quite slow, there is no appreciable covert channel threat so the timing of acknowledgements is a non-issue (e.g., use the Pump or just send the raw ACK stream down to Low). Therefore, the Pump is a good answer for sending data messages from Low to High in a general situation.

2 Queuing theoretic models

For the rest of this paper, we will concentrate on only one of our objections to blind write-up; that of buffer management—what is the probability that a buffer of size N will overflow under realistic input/arrival and output/service assumptions. As before, this is not to say that the general reliability issues are not of interest to us. Rather, the goal of this paper is to introduce a new formal model so that questions of burstiness can be quantitatively studied.

The queueing models will describe how our message traffic behaves. We show that under our MIMO model, messages will be lost because there is nowhere to put them if they are sent too fast. Therefore, something must be done to moderate traffic flow.

TRANSMITTER $\rightarrow$ BUFFER $\rightarrow$ RECEIVER

Above is our transmission model. The transmitter (input) sends messages to the buffer. The buffer size is the number of messages that the buffer can hold. Every message is of the same size and all transmission times are zero. Messages *arrive* at the buffer from the transmitter. The receiver (output) *services* messages from the buffer. When we use the term rate, we mean the total number of messages divided by the total time for the arrival or servicing of the messages. Rates have units of messages per time unit. Messages arrive at the buffer from the transmitter at an input rate denoted by λ and the receiver services them at the output rate μ. The dimensionless quantity $\frac{\lambda}{\mu}$ is extremely important to queue size.

[6] This is not the case for data transfers that are constantly updated, such as sensor reports that are sent every few seconds.

2.1 M/M/1 model

As mentioned earlier, the first work on the write-up problem to apply queueing theory seriously is in [13], where an M/M/1/N queue is used for the model. This is a queue with a Poisson arrival process and exponential service times. The N signifies that the buffer is of finite size N, unlike the standard M/M/1 queue which has an infinite buffer. The average (exponentially distributed) interarrival time is $1/\lambda$ which uniquely determines the Poisson process. The average (exponentially distributed) service time is $1/\mu$. The term p_k is the steady state probability that there are k messages in the buffer. For an M/M/1 queue p_k exists if and only if $\frac{\lambda}{\mu} < 1$; this is the ergodic condition. The analysis in [13] is done only for $\frac{\lambda}{\mu} < 1$. However, for an M/M/1/N queue the condition on $\frac{\lambda}{\mu}$ can be relaxed to be any positive value. This is because $p_k = 0$ for $k > N$, and thus the ergodic condition is trivially met (e.g., finite birth-death process).

In [13], if a message arrived at a full buffer, the new message overwrote a message that was already in the buffer. Which message is overwritten is not specified and is left as a design parameter. Communications engineers study the probability of blocked or turned-away messages — mathematically, this is identical to overwritten messages, therefore we will just use the term "lost" messages. In [13], it is noted that for $\frac{\lambda}{\mu} < 1$ and bounded away from 1 that the probability of the buffer being full can be made as small as desired by choosing a very moderately sized buffer. We take no exception to the M/M/1/N analysis for $\frac{\lambda}{\mu} << 1$. However, we also find it instructive to look at unrestricted $\frac{\lambda}{\mu}$, first in terms of the mathematical analysis, second to consider buffer size, third to look at the assumptions in terms of physical reality, and fourth as motivation for our bursty analysis.

As noted, the (steady state) probability that the buffer is full is the same as the (steady state) probability of a message being lost. The derivation for this can be found in [15, 3] and is:

$$p_N = \frac{1 - \frac{\lambda}{\mu}}{1 - (\frac{\lambda}{\mu})^{N+1}} \, (\frac{\lambda}{\mu})^N \ , \ \text{for } \frac{\lambda}{\mu} > 0 \ .$$

Provided that $\frac{\lambda}{\mu} < 1$ we see that[7]

$$N = \frac{\log\left(\frac{p_N}{1-\frac{\lambda}{\mu}+p_N\frac{\lambda}{\mu}}\right)}{\log\frac{\lambda}{\mu}} .$$

Thus, as in [13], for $\frac{\lambda}{\mu} < .95$ a buffer of size less than 1000 will suffice to keep p_N extremely small. As long as $\frac{\lambda}{\mu} < 1$ two facts are true: (1) For fixed $\frac{\lambda}{\mu}$, p_N decreases as $N \to \infty$, and (2) p_N can be made as small as desired by letting N grow.

However, as $\frac{\lambda}{\mu} \to 1^-$ the necessary buffer size also grows for fixed p_N. Obviously, if we always input at a rate that is greater than or equal[8] to the output rate our system will either lose many messages (or grind to a halt). However, what if there are periods when this is true? Is our system robust enough to tolerate these bursts of input activity? If there is no feedback from High to Low, what is to prevent Low from temporarily sending messages faster than High can handle them? If our system is designed along an M/M/1/N type model we argue that we may be in serious trouble with respect to dropped/lost messages. So let us study the cases where $\frac{\lambda}{\mu} \geq 1$ might hold for some period of time. We analyze the situation where $\lambda/\mu \geq 1$, in the M/M/1/N model. This might not reflect the steady state behaviour of our Low-to-High system but it might represent either (1)—a large transient (bursty) period, large enough so that steady state probabilities are locally achieved, or (2)—the results from the accumulation of many bursty periods causing buffer congestion.

If the rates are equivalent then the buffer required to ensure no overwritten messages is in fact quite large. When $\lambda = \mu$, by using L'Hôpital's rule, we see that $N = \frac{1}{p_N} - 1$. So for $\lambda = \mu$ the buffer size, less one, is the inverse of the desired p_N. Therefore, a buffer of size 1000 will suffice if one is willing to accept a probability of a lost message of .001.

Let us analyze the situation where $\lambda/\mu > 1$. Note that for $\lambda > \mu$:

$$p_N = \frac{1-\frac{\lambda}{\mu}}{1-(\frac{\lambda}{\mu})^{N+1}}\,(\frac{\lambda}{\mu})^N = \frac{\frac{\lambda}{\mu}-1}{\frac{\lambda}{\mu}}\frac{1}{1-(\frac{\mu}{\lambda})^{N+1}} \;\geq\; 1-(\lambda/\mu)^{-1} > 0 .$$

Similar to the situations when $\frac{\lambda}{\mu} \leq 1$, we see that for $\frac{\lambda}{\mu} > 1$, p_N decreases as $N \to \infty$, but when $\frac{\lambda}{\mu} > 1$, p_N does not approach zero, the smallest p_N can ever be is $1-(\lambda/\mu)^{-1}$.

[7] For $\frac{\lambda}{\mu} > 1$, we will show later that N cannot be solved for arbitrary p_N.

[8] Since arrivals are probabilistic, not deterministic, equal rates can still cause problems.

For example, if $\frac{\lambda}{\mu} = 1.1$, then we will lose at least 9.1% of the transmitted messages on average. It is certainly not surprising that if we send more into the buffer than what can be serviced, messages will be lost.

We must make an effort to model burstiness. In those periods when Low is faster than High, how do we know that messages are not overwritten? Even if we accept the M/M/1 queue as being an accurate model of reality (which we do not), if the ratio λ/μ hovers around 1^-, the probability of a message being lost can be quite large. All of this aside, we are still faced with the fact that the M/M/1 queue does not take burstiness into account. The Poisson process of rate λ has as one of its main assumptions that the probability of exactly one arrival in the interval $[t, t + dt]$ is λdt [15]. This certainly does not imply deterministic arrival. It is not proper to call this burstiness; rather, this is a fixed probabilistic behaviour. Bursty behaviour does not obey this Poisson assumption in that a fixed λ cannot be assigned to the probability of exactly one arrival in the interval $[t, t + dt]$. Therefore, we propose an alternative to the M/M/1/ based models that incorporates burstiness.

2.2 The MIMO Model

We have discussed why the M/M/1 model does not model burstiness. We have shown the impact of the input rate being faster than the output rate—something that could happen for certain periods of the system. Therefore, we present a new model, which has the M/M/1 model as a special case. Our model is based upon current research in the computer communications community. The basic premise is that the system oscillates between certain bursty and non-bursty periods.

Our model is based on the MMPP [5]; this is a fundamental paper for dealing with burstiness. The MMPP only applies to the transmitter (Low in our case). Instead of message arrivals being a Poisson process (as in the M/M/1 model), the transmitter oscillates between two states, each state sends messages in a Poisson process, and the state changes are determined by a continuous-time Markov process with fixed mean sojourn times for each state. The two states represent the normal and the bursty behaviour of the transmission process.

For our purpose, we find the MMPP lacking because it does not take receiver (High) behaviour into account. Thus, we wish to also model High as oscillating between two states: a normal state and a busy state. The MMPP considers continuous state changes,

our model assumes that state changes only take place at certain constant time increments, called "frames".

Therefore, our state changes are given by a (discrete) Markov chain. We base our conclusions concerning buffer congestion on a computer simulation of the MIMO model.

2.2.1 Low: Modulate Input

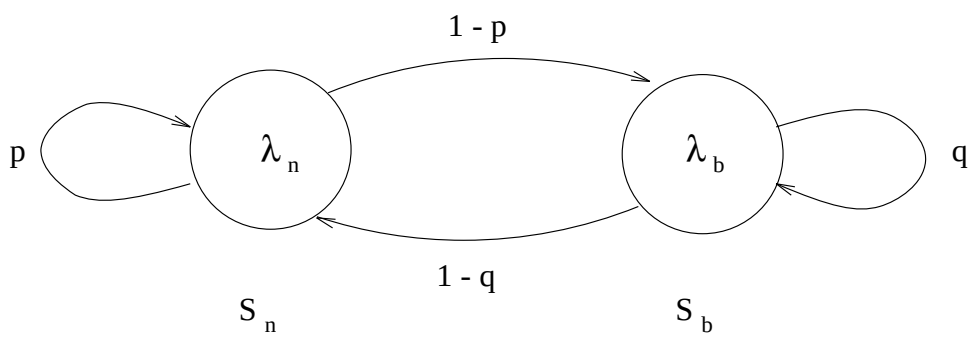

Figure 4: Low as a Markov chain

$$M_l = \begin{pmatrix} p & 1-p \\ 1-q & q \end{pmatrix}$$

Low's transmission is given by a 2-state Markov chain. In every frame (a fixed number of time units) we determine which state we are in according to the transition matrix M_l. In the first or normal state S_n, Low transmits messages that arrive, via a Poisson process of rate λ_n. In the second or bursty state S_b, arrivals are a Poisson process of rate λ_b.

The steady state probability for the normal state is $\Pi_n = \frac{1-q}{2-p-q}$. The steady state probability for the bursty state is $\Pi_b = \frac{1-p}{2-p-q}$. We see then that the (global) arrival rate, denoted by λ, is $\lambda = \Pi_n\lambda_n + \Pi_b\lambda_b$. Once the Markov chain has entered, in the steady state, S_n the number of frames that it stays in that state is given by a geometric random variable with parameter $1-p$. The mean of this geometric random variable is $\frac{1}{1-p}$, corresponding to the mean sojourn time in the continuous MMPP (similarly for S_b). Thus we see that the Low end is just a "discretized" MMPP with large mean sojourn times.

2.2.2 High: Modulated Output

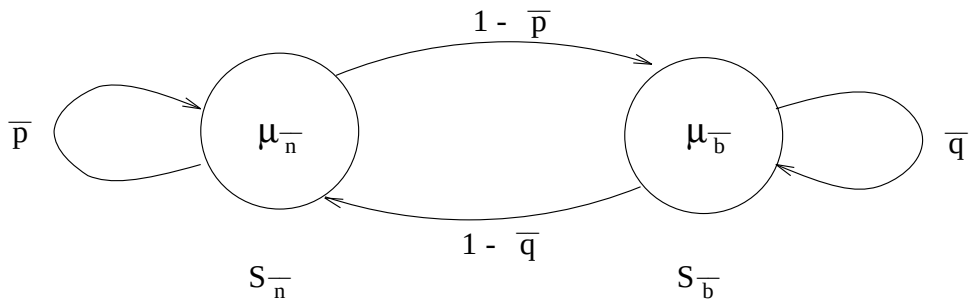

Figure 5: High as a Markov chain

$$M_h = \begin{pmatrix} \overline{p} & 1-\overline{p} \\ 1-\overline{q} & \overline{q} \end{pmatrix}$$

High's service times are similarly given by a 2-state Markov chain. In every frame (Low and High frames are synchronized), we determine which state we are in according to the transition matrix M_h. In the first or normal state $S_{\overline{n}}$, High services messages by an exponential distribution with parameter (inverse of the mean) $\mu_{\overline{n}}$. In the second or busy state $S_{\overline{b}}$, messages are exponentially serviced at a slower rate $\mu_{\overline{b}}$[9].

The steady state probability for the normal state is $\Pi_{\overline{n}} = \frac{1-\overline{q}}{2-\overline{p}-\overline{q}}$. The steady state probability for the bursty state is $\Pi_{\overline{b}} = \frac{1-\overline{p}}{2-\overline{p}-\overline{q}}$. Note that the (global) service rate is $\mu = \Pi_{\overline{n}}\mu_{\overline{n}} + \Pi_{\overline{b}}\mu_{\overline{b}}$.

2.3 Low & High together

In the MIMO model we assume that Low is transmitting to High and messages are queued in a buffer until High can service them, and that Low and High behave as the 2-state Markov chains described above[10]. Low and High are independent[11] in our model because Low's bursty and High's busy periods cannot be controlled. Therefore, the

[9] There is a slight idealization for mathematical simplicity because service times cannot change once they have been chosen. However, as long as the frame size is large in comparison to the mean service times, the difference is negligible.

[10] There is no requirement that $p + q = 1$ or $\overline{p} + \overline{q} = 1$. The relationship between the probabilities determines correlations between the states. In our simulations we used correlated states.

[11] Thus we have described the MIMO model as a queueing model, which we denote as Mi/Mo/1/N, for modulated input, modulated output/service, single server, and finite buffer of size N.

MIMO model consists of a four state Markov chain:

$S_{n,\overline{n}}$ Low normal, High normal
$S_{n,\overline{b}}$ Low normal, High busy
$S_{b,\overline{n}}$ Low bursty, High normal
$S_{b,\overline{b}}$ Low bursty, High busy

Where $S_{n,\overline{n}}$ is the state where Low is in its normal state, and High is in its normal state, etc.

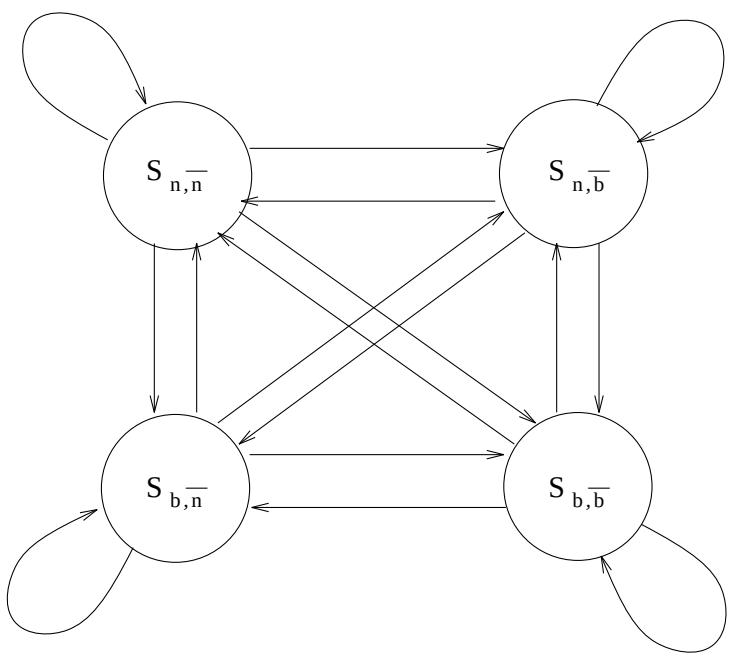

Figure 6: MIMO model

$$M = \begin{pmatrix} p\overline{p} & p(1-\overline{p}) & (1-p)\overline{p} & (1-p)(1-\overline{p}) \\ p(1-\overline{q}) & p\overline{q} & (1-p)(1-\overline{q}) & (1-p)\overline{q} \\ (1-q)\overline{p} & (1-q)(1-\overline{p}) & q\overline{p} & q(1-\overline{p}) \\ (1-q)(1-\overline{q}) & (1-q)\overline{q} & q(1-\overline{q}) & q\overline{q} \end{pmatrix}$$

The four states transit, every frame, via the matrix M.

We have the four steady state probabilities that are obtained by multiplying the Low

end and High end steady state probabilities together. This is valid because the Low and High ends are independent.

$$\Pi_{n,\overline{n}} = \left(\frac{1-q}{2-p-q}\right)\left(\frac{1-\overline{q}}{2-\overline{p}-\overline{q}}\right)$$
$$\Pi_{n,\overline{b}} = \left(\frac{1-q}{2-p-q}\right)\left(\frac{1-\overline{p}}{2-\overline{p}-\overline{q}}\right)$$
$$\Pi_{b,\overline{n}} = \left(\frac{1-p}{2-p-q}\right)\left(\frac{1-\overline{q}}{2-\overline{p}-\overline{q}}\right)$$
$$\Pi_{b,\overline{b}} = \left(\frac{1-p}{2-p-q}\right)\left(\frac{1-\overline{p}}{2-\overline{p}-\overline{q}}\right)$$

The term λ from the Low end and μ from the High end let us form, as in the M/M/1 queue, the ratio λ/μ. We will simulate the MIMO model under certain buffer sizes and see how many messages get lost in transmission when $\frac{\lambda}{\mu} < 1$.

2.4 Simulation results of the MIMO model

Below we present a table summarizing some of our simulation results of the MIMO model versus the M/M/1/N model. Our simulation was done on a Sparc20 using the MODSIM package. The theoretical steady state probabilities and mean arrival and service rates matched our simulation results for the same statistics. Thus, even though we do not have MIMO theoretical results for lost message percentages we feel that our simulations are accurate.

We present only a small subset of our runs. We use a frame size of 10. The runs that we discuss were chosen to show the danger of assuming results about lost messages by just studying the ratio of λ to μ. We do not claim that the MIMO model is an accurate representation of reality. What we do claim is that the M/M/1/N model is not a good model of reality if the system behaves in a bursty manner and that assumptions of lost messages can be misleading and dangerous. For a bursty system we feel that the MIMO model is a better model of reality than the M/M/1/N model.

The first table compares the M/M/1/N model against the MIMO model for a buffer of size N = 1000. We chose 1000 for the buffer size because that was an appropriate size proposed in [13]. In the following table $p = .70$, $q = .30$, and $\lambda_n = 1.0$. We let λ_b take on the values 31,51,61 to result in λ/μ taking the values of .50,.80,.95 , respectively. High was fixed at $\overline{p} = .50 = \overline{q}$, $\mu_{\overline{n}} = 30$, $\mu_{\overline{b}} = 10$.

$\frac{\lambda}{\mu}$	% lost in M/M/1	% lost in MIMO
.50	0^+%	.04%
.80	0^+%	7.5%
.95	0^+%	15.6%

Simulation: Buffer size 1000 High: $\overline{p} = .50 = \overline{q}$, $\mu_{\overline{n}} = 30$, $\mu_{\overline{b}} = 10$

The simulation gave us no messages lost for the M/M/1/N queue. However, all of the theoretical M/M/1/N lost percentages are infinitesimally small, but still non-zero. This is why we used the notation 0^+. We see, in the MIMO model, unsurprisingly, that as $\frac{\lambda}{\mu} \to 1^-$ the percent of lost messages increases. What is surprising is that for Low sending at half the rate that High is servicing, a non-trivial amount of messages can still be lost.

We increased the buffer size by an order of magnitude and still had lost messages for the MIMO model as shown below. We did have to increase λ/μ to .995 to get a significant amount of lost messages.

p	λ_n	q	λ_b	$\frac{\lambda}{\mu}$	% lost in M/M/1	% lost in MIMO
.70	2	.30	128	.995	0^+%	6.2%

Simulation: Buffer size 10,000 High: $\overline{p} = .50 = \overline{q}$, $\mu_{\overline{n}} = 60$, $\mu_{\overline{b}} = 20$.

In fact, no matter what size buffer is used we could always come up with a MIMO example with $\frac{\lambda}{\mu} < 1$, such that messages were still lost. Further, our simulations also showed that messages were lost for M/Mo/1/N and Mi/M/1/N queues when $\frac{\lambda}{\mu} < 1$. Of course one could insure, up to a high probability, that messages were not lost by basically using a buffer on the order of magnitude of the number of messages sent. But this is not a practical engineering solution. The point of this is: *the potential for bursty periods on the Low end and busy periods on the High end must be well understood before any claims of buffer management can be made if one does not have flow control.*

The MIMO model, because of its discrete nature, does not capture as much of the bursty behaviour as the continuous time MMPP. Further, there are some who argue that all of these models are far from reality because message traffic patterns have a fractal (self-similar) nature to them [11, 14]; thus the simple burstiness modelled in the MIMO model may be far from reality (but it is still better than M/M/1 which has no burstiness). Therefore, we feel flow control is necessary unless one can be sure that Low

is always slower than High. We feel the Pump, via its ACK stream flow control, will let Low send messages as fast as High can handle them, and because the Pump insures reliability (none of the blind write-up methods do), it is a good solution to the write-up problem.

3 Pump scales up, does blind write-up?

We have just been dealing with one source and one destination. What happens when there are multiple sources and multiple destinations in a network environment where the number and/or rates of inputs change dynamically? The bandwidth of communication links, transmission speed, and processing speed are all limited. In a network, if the load of data traffic offered to the network exceeds its capability, some of the load must be cut fairly.

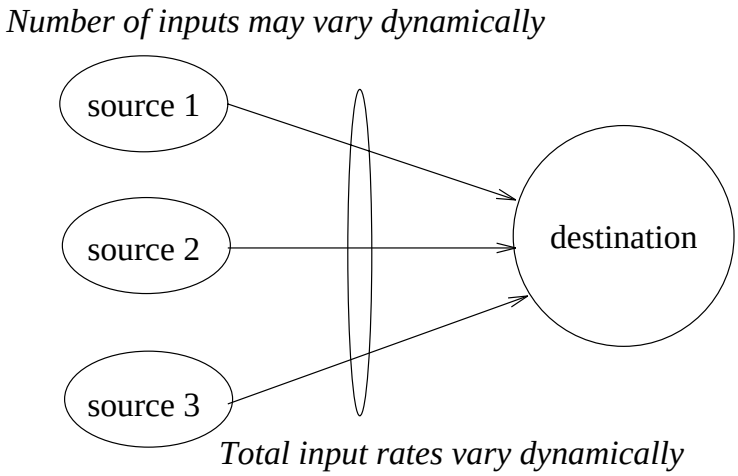

Figure 7: Simplified view of fairness problem

In a network, resources are often shared among several sessions. Services for other sessions can potentially be disrupted if too much resource is allocated to one particular session. Therefore, we must consider denial of service attacks.

The basic idea of the network Pump [10] is to dynamically control input rates by attempting to "slave" the input rates to a moving average of the output rates (service rates) by moderating the ACK rate to a source. In this way the network Pump can achieve fairness and prevent denial of service attacks without sacrificing performance. The simulation results in [10] support our claims.

The blind write-up method does not send any control information back to the sources. Hence, sources neither know the status of the destination (busy/alive?) nor can they intelligently control their input rates. We would be interested in seeing how non-flow controlled methods deal with fairness and denial of service in the network environment.

4 Conclusion

This paper focused on the consequences of sending messages from Low to High without flow control. We showed that the paradigm of "*if input rate is less than output service rate, an intermediary buffer can be used to insure message delivery*" is too optimistic.

Historically, (in the non-secure world) the M/M/1 queue has been used when we have unlimited buffer resources. In reality, buffer size is limited, so if there is burstiness from the input, ACKs can be used to prevent message loss. However, in the secure world we do not want to send ACKs from High to Low, but the consequences will be lost messages. Hence, determining the queue size is extremely important if there are no ACKs. The M/M/1/N model has been used to find the adequate buffer size under the no-ACK condition. However, the M/M/1/N model is far from reality. In this paper we introduced the MIMO model which takes burstiness into account. We showed that the queue size based on the M/M/1/N model is too optimistic through simulation results. The MIMO model, which has 4 states (2 Low and 2 High), might itself not accurately reflect real system behavior. But it is a sufficient model to show that the buffer size analysis based on the M/M/1/N model is too optimistic for the secure world. Even if there exist buffers of sufficient size to store all bursty messages, we still have the other very important reliability issues that ACKless based schemes do not sufficiently address. Therefore, the ACKs must be sent to Low.

5 Acknowledgements

We thank Craig Barnhart, Tony Ephremides, David Goldschlag, Carl Landwehr, John McDermott, Ruth Heilizer, Tobi Saulnier, and the anonymous referees for their helpful comments and suggestions.

References

[1] J.N. Froscher, D.M. Goldschlag, M.H. Kang, C.E. Landwehr, A. Moore, I.S. Moskowitz, and C.N. Payne. "Improving Inter-Enclave Information Flow for a Secure Strike Planning Application," Submitted for publication, 1995.

[2] D. Goldschlag. "Several Secure Store and Forward Devices," Submitted for publication, 1995.

[3] D. Gross and C.M. Harris. "Fundamentals of Queuing Theory," 2nd ed., John Wiley, NY, 1985

[4] R. Gusella. "Characterizing the variability of arrival processes with indexes of dispersion," IEEE Journal on Selected Areas in Communications, V. 9, No. 2, pp. 203-211, Feb. 1991.

[5] H. Heffes and D.M. Lucantoni. "A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance," IEEE Journal on Selected Areas in Communications, Vol. SAC-4, No. 6, pp. 856-868, Sept 1986.

[6] M. H. Kang and I. S. Moskowitz. "A Pump for rapid, reliable, secure communication," Proceedings of ACM Conf. Computer & Commun. Security '93, pp. 119 - 129, Fairfax, VA, 1993.

[7] Kang, M. H., Froscher, J. N., and Costich, O. "A practical transaction model and untrusted transaction manager for multilevel-secure database systems," In Database Security, VI: Status and Prospects, eds. Bhavani Thuraisingham and Carl Landwehr, North-Holland pp. 285-300, 1993.

[8] Kang, M. H. et al., "Achieving Database Security through Data Replication: The SINTRA Prototype," 17th National Computer Security Conference pp. 77-87, 1994.

[9] M. H. Kang and I. S. Moskowitz. "A data pump for communication," Submitted for publication, 1995.

[10] M. H. Kang, I. S. Moskowitz, and D.C. Lee. "A network version of the Pump," To appear: Proceedings of IEEE Symposium on Research in Security and Privacy, Oakland, CA, May 1995.

[11] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. "On the self-similar nature of ethernet traffic," Proceedings of SIGCOMM'93, 1993.

[12] S. Li and J.W. Mark. "Performance of voice/data integration on a TDM system," IEEE Transactions on Communications, V. COM-33, No. 12, pp. 1265-1273, Dec. 1985

[13] J. McDermott. "The b^2/c^3 problem: How big buffers overcome covert channel cynicism in trusted database systems," Biskup, J., M. Morgenstern, and C. E. Landwehr, eds. Database Security, VIII: Status and Prospects. IFIP Transactions A-60, Elsevier Science B.V., Amsterdam, pp. 111-122, 1994.

[14] V. Paxson and S. Floyd. "Wide area traffic: The failure of Poisson modeling," Proceedings of SIGCOMM'94, 1994 and IEEE/ACM Transactions on Networking, June 1995, V. 3, No. 3, pp. 226-244.

[15] T.H. Robertazzi. "Computer Networks and Systems: "Queuing Theory and Performance Evaluation,"Springer-Verlag, NY, 1990. .

[16] E.T. Saulnier and K.S. Vastola."A "HI-LO" Markov chain model for multimedia traffic in ATM networks," Proceedings of Globecom'92, Vol. 3, pp. 1450-1454, Dec. 1992.