Network Working Group
Request for Comments: 2729
Category: Informational

P. Bagnall
R. Briscoe
A. Poppitt
BT

December 1999

Taxonomy of Communication Requirements for Large-scale Multicast Applications

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Abstract

The intention of this memo is to define a classification system for the communication requirements of any large-scale multicast application (LSMA). It is very unlikely one protocol can achieve a compromise between the diverse requirements of all the parties involved in any LSMA. It is therefore necessary to understand the worst-case scenarios in order to minimize the range of protocols needed. Dynamic protocol adaptation is likely to be necessary which will require logic to map particular combinations of requirements to particular mechanisms. Standardizing the way that applications define their requirements is a necessary step towards this. Classification is a first step towards standardization.

Table of Contents

1. Introduction

This taxonomy consists of a large number of parameters that are considered useful for describing the communication requirements of LSMAs. To describe a particular application, each parameter would be assigned a value. Typical ranges of values are given wherever possible. Failing this, the type of any possible values is given. The parameters are collected into ten or so higher level categories, but this is purely for convenience.

The parameters are pitched at a level considered meaningful to application programmers. However, they describe communications not applications - the terms '3D virtual world', or 'shared TV' might imply communications requirements, but they don't accurately describe them. Assumptions about the likely mechanism to achieve each requirement are avoided where possible.

While the parameters describe communications, it will be noticed that few requirements concerning routing etc. are apparent. This is because applications have few direct requirements on these second order aspects of communications. Requirements in these areas will have to be inferred from application requirements (e.g. latency).

The taxonomy is likely to be useful in a number of ways:

1. Most simply, it can be used as a checklist to create a requirements statement for a particular LSMA. Example applications will be classified [bagnall98] using the taxonomy in order to exercise (and improve) it

2. Because strictest requirement have been defined for many parameters, it will be possible to identify worst case scenarios for the design of protocols

3. Because the scope of each parameter has been defined (per session, per receiver etc.), it will be possible to highlight where heterogeneity is going to be most marked

4. It is a step towards standardization of the way LSMAs define their communications requirements. This could lead to standard APIs between applications and protocol adaptation middleware

5. Identification of limitations in current Internet technology for LSMAs to be added to the LSMA limitations memo [limitations]

6. Identification of gaps in Internet Engineering Task Force (IETF) working group coverage

This approach is intended to complement that used where application scenarios for Distributed Interactive Simulation (DIS) are proposed in order to generate network design metrics (values of communications parameters). Instead of creating the communications parameters from the applications, we try to imagine applications that might be enabled by stretching communications parameters.

2. Definition of Sessions

The following terms have no agreed definition, so they will be defined for this document.

Session
: a happening or gathering consisting of flows of information related by a common description that persists for a non-trivial time (more than a few seconds) such that the participants (be they humans or applications) are involved and interested at intermediate times. A session may be defined recursively as a super-set of other sessions.

Secure session
: a session with restricted access

A session or secure session may be a sub and/or super set of a multicast group. A session can simultaneously be both a sub and a super-set of a multicast group by spanning a number of groups while time-sharing each group with other sessions.

3. Taxonomy

3.1 Summary of Communications Parameters

Before the communications parameters are defined, typed and given worst-case values, they are simply listed for convenience. Also for convenience they are collected under classification headings.

3.2 Definitions, types and strictest requirements

The terms used in the above table are now defined for the context of this document. Under each definition, the type of their value is given and where possible worst-case values and example applications that would exhibit this requirement.

There is no mention of whether a communication is a stream or a discrete interaction. An attempt to use this distinction as a way of characterizing communications proved to be remarkably unhelpful and was dropped.

3.2.1 Types

Each requirement has a type. The following is a list of all the types used in the following definitions.

Application Benchmark

This is some measure of the processor load of an application, in some architecture neutral unit. This is non-trivial since the processing an application requires may change radically with different hardware, for example, a video client with and without hardware support.

Bandwidth Measured in bits per second, or a multiple of.

Boolean

Abstract Currency
An abstract currency is one which is adjusted to take inflation into account. The simplest way of doing this is to use the value of a real currency on a specific date. It is effectively a way of assessing the cost of something in "real terms". An example might be 1970 US$. Another measure might be "average man hours".

Currency - current local

Data Size

Date (time since epoch)

Enumeration

Fraction

Identifiers
A label used to distinguish different parts of a communication

Integer

Membership list/rule

Macro
A small piece of executable code used to describe policies

Time

3.2.2 Reliability

3.2.2.1 Packet Loss

Transactional

When multiple operations must occur atomically, transactional communications guarantee that either all occur or none occur and a failure is flagged.

Type:	Boolean
Meaning:	Transactional or Not transaction
Strictest Requirement:	Transactional
Scope:	per stream
Example Application:	Bank credit transfer, debit and credit must be atomic.
NB:	Transactions are potentially much more complex, but it is believed this is an application layer problem.

Guaranteed

Guarantees communications will succeed under certain conditions.

Type:	Enumerated
Meaning:	Deferrable - if communication fails it will be deferred until a time when it will be successful. Guaranteed - the communication will succeed so long as all necessary components are working. No guarantee - failure will not be reported.
Strictest Requirement:	Deferrable
Example Application:	Stock quote feed - Guaranteed
Scope:	per stream
NB:	The application will need to set parameters to more fully define Guarantees, which the middleware may translate into, for example, queue lengths.

Tolerated loss

This specifies the proportion of data from a communication that can be lost before the application becomes completely unusable.

Type:	Fraction
Meaning:	fraction of the stream that can be lost
Strictest Requirement:	0%
Scope:	per stream
Example Application:	Video - 20%

Semantic loss

The application specifies how many and which parts of the communication can be discarded if necessary.

Type:	Identifiers, name disposable application level frames
Meaning:	List of the identifiers of application frames which may be lost
Strictest Requirement:	No loss allowed
Scope:	per stream
Example Application:	Video feed - P frames may be lost, I frames not

3.2.2.2. Component Reliability

Setup Fail-over time

The time before a failure is detected and a replacement component is invoked. From the applications point of view this is the time it may take in exceptional circumstances for a channel to be set-up. It is not the "normal" operating delay before a channel is created.

Type:	Time
Strictest Requirement:	Web server - 1 second
Scope:	per stream
Example Application:	Name lookup - 5 seconds

Mean time between failures

The mean time between two consecutive total failures of the channel.

Type:	Time
Strictest Requirement:	Indefinite
Scope:	per stream
Example Application:	Telephony - 1000 hours

Fail over time during a stream

The time between a stream breaking and a replacement being set up.

Type:	Time
Strictest Requirement:	Equal to latency requirement
Scope:	per stream
Example Application:	File Transfer - 10sec

3.2.3. Ordering

Ordering type

Specifies what ordering must be preserved for the application

Type:	{ Enumeration timing, Enumeration sequencing, Enumeration causality }
Meaning:	Timing - the events are timestamped Global Per Sender none Sequencing - the events are sequenced in order of occurrence Global Per Sender none Causality - the events form a graph relating cause and effect Global Per Sender none
Strictest Requirement:	Global, Global, Global
Scope:	per stream
Example Application:	Game - { none, per sender, global } (to make sure being hit by bullet occurs after the shot is fired!)

3.2.4. Timeliness

Hard real- time

There is a meta-requirement on timeliness. If hard real-time is required then the interpretation of all the other requirements changes. Failures to achieve the required timeliness must be

reported before the communication is made. By contrast soft real-time means that there is no guarantee that an event will occur in time. However statistical measures can be used to indicate the probability of completion in the required time, and policies such as making sure the probability is 95% or better could be used.

Type:	Boolean
Meaning:	Hard or Soft realtime
Strictest Requirement:	Hard
Scope:	per stream
Example Application:	Medical monitor - Hard

Synchronicity

To make sure that separate elements of a session are correctly synchronized with respect to each other

Type:	Time
Meaning:	The maximum time drift between streams
Strictest Requirement:	80ms for human perception
Scope:	per stream pair/set
Example Application:	TV lip-sync value 80ms
NB:	the scope is not necessarily the same as the session. Some streams may no need to be sync'd, (say, a score ticker in a football match

Burstiness

This is a measure of the variance of bandwidth requirements over time.

Type:	Fraction
Meaning:	either: Variation in b/w as fraction of b/w for variable b/w communications or duty cycle (fraction of time at peak b/w) for intermittent b/w communications.
Strictest Requirement:	Variation = max b/w Duty cycle ~ 0
Scope:	per stream
Example Application:	Sharing video clips, with chat channel - sudden bursts as clips are swapped. Compressed Audio - difference between silence and talking
NB:	More detailed analysis of communication flow (e.g. max rate of b/w change or

Fourier Transform of the b/w requirement) is possible but as complexity increases usefulness and computability decrease.

Jitter

Jitter is a measure of variance in the time taken for communications to traverse from the sender (application) to the receiver, as seen from the application layer.

Type:	Time
Meaning:	Maximum permissible time variance
Strictest Requirement:	<1ms
Scope:	per stream
Example Application:	audio streaming - <1ms
NB:	A jitter requirement implies that the communication is a real-time stream. It makes relatively little sense for a file transfer for example.

Expiry

This specifies how long the information being transferred remains valid for.

Type:	Date
Meaning:	Date at which data expires
Strictest Requirement:	For ever
Scope:	per stream
Example Application:	key distribution - now+3600 seconds (valid for at least one hour)

Latency

Time between initiation and occurrence of an action from application perspective.

Type:	Time
Strictest Requirement:	Near zero for process control apps
Scope:	per stream
Example Application:	Audio conference 20ms
NB:	Where an action consists of several distinct sequential parts the latency budget must be split over those parts. For process control the requirement may take any value.

Optimum Bandwidth

Bandwidth required to complete communication in time

Type:	Bandwidth
Strictest Requirement:	No upper limit
Scope:	per stream
Example Application:	Internet Phone 8kb/s

Tolerable Bandwidth

Minimum bandwidth that application can tolerate

Type:	Bandwidth
Strictest Requirement:	No upper limit
Scope:	per stream
Example Application:	Internet phone 4kb/s

Required by time and tolerance

Time communication should complete by and time when failure to complete renders communication useless (therefore abort).

Type:	{
	Date - preferred complete time,
	Date - essential complete time
	}

Strictest Requirement:	Both now.
Scope:	per stream
Example Application:	Email - Preferred 5 minutes & Essential in 1 day
NB:	Bandwidth * Duration = Size; only two of these parameters may be specified. An API though could allow application authors to think in terms of any two.

Host performance

Ability of host to create/consume communication

Type:	Application benchmark
Meaning:	Level of resources required by Application
Strictest Requirement:	Full consumption
Scope:	per stream
Example Application:	Video - consume 15 frames a second
NB:	Host performance is complex since load, media type, media quality, h/w assistance, and encoding scheme all affect the

processing load. These are difficult to predict prior to a communication starting. To some extent these will need to be measured and modified as the communication proceeds.

Frame size

Size of logical data packets from application perspective

Type:	data size
Strictest Requirement:	6 bytes (gaming)
Scope:	per stream
Example Application:	video = data size of single frame update

Content size

The total size of the content (not relevant for continuous media)

Type:	data size
Strictest Requirement:	N/A
Scope:	per stream
Example Application:	document transfer, 4kbytes

3.2.5. Session Control

Initiation

Which initiation mechanism will be used.

Type:	Enumeration
Meaning:	Announcement - session is publicly announced via a mass distribution system Invitation - specific participants are explicitly invited, e.g. my email Directive - specific participants are forced to join the session
Strictest Requirement:	Directive
Scope:	per stream
Example Application:	Corporate s/w update - Directive

Start Time

Time sender starts sending!

Type:	Date
Strictest Requirement:	Now
Scope:	per stream
Example Application:	FTP - at 3am

End Time

Type:	Date
Strictest Requirement:	Now
Scope:	per stream
Example Application:	FTP - Now+30mins

Duration

(end time) - (start time) = (duration), therefore only two of three should be specified.

Type:	Time
Strictest Requirement:	- 0ms for discrete, indefinite for streams
Scope:	per stream
Example Application:	audio feed - 60mins

Active Time

Total time session is active, not including breaks

Type:	Time
Strictest Requirement:	equals duration
Scope:	per stream
Example Application:	Spectator sport transmission

Session Burstiness

Expected level of burstiness of the session

Type:	Fraction
Meaning:	Variance as a fraction of maximum bandwidth
Strictest Requirement:	=bandwidth
Scope:	per stream
Example Application:	commentary & slide show: 90% of max

Atomic join

Session fails unless a certain proportion of the potential participants accept an invitation to join. Alternatively, may be specified as a specific numeric quorum.

Type:	Fraction (proportion required) or int (quorum)
Strictest Requirement:	1.0 (proportion)
Example Application:	price list update, committee meeting
Scope:	per stream or session
NB:	whether certain participants are essential is application dependent.

Late join allowed ?

Does joining a session after it starts make sense

Type:	Boolean
Strictest Requirement:	allowed
Scope:	per stream or session
Example Application:	game - not allowed
NB:	An application may wish to define an alternate session if late join is not allowed

Temporary leave allowed ?

Does leaving and then coming back make sense for session

Type:	Boolean
Strictest Requirement:	allowed
Scope:	per stream or session
Example Application:	FTP - not allowed

Late join with catch-up allowed ?

Is there a mechanism for a late joiner to see what they've missed

Type:	Boolean
Strictest Requirement:	allowed
Scope:	per stream or session
Example Application:	sports event broadcast, allowed
NB:	An application may wish to define an alternate session if late join is not allowed

Potential streams per session

Total number of streams that are part of session, whether being consumed or not

Type:	Integer
Strictest Requirement:	No upper limit
Scope:	per session
Example Application:	football match mcast - multiple camera's, commentary, 15 streams

Active streams per sessions (i.e. max app can handle)

Maximum number of streams that an application can consume simultaneously

Type:	Integer
Strictest Requirement:	No upper limit
Scope:	per session
Example Application:	football match mcast - 6, one main video, four user selected, one audio commentary

3.2.6. Session Topology

Note: topology may be dynamic. One of the challenges in designing adaptive protocol frameworks is to predict the topology before the first join.

Number of senders

The number of senders is a result the middleware may pass up to the application

Type:	Integer
Strictest Requirement:	No upper limit
Scope:	per stream
Example Application:	network MUD - 100

Number of receivers

The number of receivers is a results the middleware may pass up to the application

Type:	Integer
Strictest Requirement:	No upper limit
Scope:	per stream
Example Application:	video mcast - 100,000

3.2.7. Directory

Fail-over timeout (see Reliability: fail-over time)

Mobility

Defines restrictions on when directory entries may be changed

Type:	Enumeration
Meaning:	while entry is in use while entry in unused never
Strictest Requirement:	while entry is in use
Scope:	per stream
Example Application:	voice over mobile phone, while entry is in use (as phone gets new address when changing cell).

3.2.8. Security

The strength of any security arrangement can be stated as the expected cost of mounting a successful attack. This allows mechanisms such as physical isolation to be considered alongside encryption mechanisms. The cost is measured in an abstract currency, such as 1970 UD$ (to inflation proof).

Security is an orthogonal requirement. Many requirements can have a security requirement on them which mandates that the cost of causing the system to fail to meet that requirement is more than the specified amount. In terms of impact on other requirements though, security does potentially have a large impact so when a system is trying to determine which mechanisms to use and whether the requirements can be met security will clearly be a major influence.

Authentication Strength

Authentication aims to ensure that a principal is who they claim to be. For each role in a communication, (e.g. sender, receiver) there is a strength for the authentication of the principle who has taken on that role. The principal could be a person, organization or other legal entity. It could not be a process since a process has no legal representation.

Type:	Abstract Currency
Meaning:	That the cost of hijacking a role is in excess of the specified amount. Each role is a different requirement.

```
Strictest Requirement: budget of largest attacker
Scope:                 per stream
Example Application:   inter-governmental conference
```

Tamper-proofing

This allows the application to specify how much security will be applied to ensuring that a communication is not tampered with. This is specified as the minimum cost of successfully tampering with the communication. Each non-security requirement has a tamper-proofing requirement attached to it.

Requirement: The cost of tampering with the communication is in excess of the specified amount.

```
Type:                  {
                         Abstract Currency,
                         Abstract Currency,
                         Abstract Currency
                       }
Meaning:               cost to alter or destroy data,
                       cost to replay data (successfully),
                       cost to interfere with timeliness.
Scope:                 per stream
Strictest Requirement: Each budget of largest attacker
Example Application:   stock price feed
```

Non-repudiation strength

The non-repudiation strength defines how much care is taken to make sure there is a reliable audit trail on all interactions. It is measured as the cost of faking an audit trail, and therefore being able to "prove" an untrue event. There are a number of possible parameters of the event that need to be proved. The following list is not exclusive but shows the typical set of requirements.

1. Time 2. Ordering (when relative to other events) 3. Whom 4. What (the event itself)

There are a number of events that need to be provable. 1. sender proved sent 2. receiver proves received 3. sender proves received.

```
Type:                  Abstract Currency
Meaning:               minimum cost of faking or denying an event
Strictest Requirement:  Budget of largest attacker
Scope:                 per stream
Example Application:   Online shopping system
```

Denial of service

There may be a requirement for some systems (999,911,112 emergency services access for example) that denial of service attacks cannot be launched. While this is difficult (maybe impossible) in many systems at the moment it is still a requirement, just one that can't be met.

Type:	Abstract Currency
Meaning:	Cost of launching a denial of service attack is greater than specified amount.
Strictest Requirement:	budget of largest attacker
Scope:	per stream
Example Application:	web hosting, to prevent individual hackers stalling system.

Action restriction

For any given communication there are a two actions, send and receive. Operations like adding to members to a group are done as a send to the membership list. Examining the list is a request to and receive from the list. Other actions can be generalized to send and receive on some communication, or are application level not comms level issues.

Type:	Membership list/rule for each action.
Meaning:	predicate for determining permission for role
Strictest Requirement:	Send and receive have different policies.
Scope:	per stream
Example Application:	TV broadcast, sender policy defines transmitter, receiver policy is null.
NB:	Several actions may share the same membership policy.

Privacy

Privacy defines how well obscured a principals identity is. This could be for any interaction. A list of participants may be obscured, a sender may obscure their identity when they send. There are also different types of privacy. For example knowing two messages were sent by the same person breaks the strongest type of privacy even if the identity of that sender is still unknown. For each "level" of privacy there is a cost associated with violating it. The requirement is that this cost is excessive for the attacker.

Type:	{ Abstract Currency, Abstract Currency, Abstract Currency, Abstract Currency }
Meaning:	Level of privacy, expected cost to violate privacy level for:- openly identified - this is the unprotected case anonymously identified - (messages from the same sender can be linked) unadvertised (but traceable) - meaning that traffic can be detected and traced to it's source or destination, this is a breach if the very fact that two specific principals are communicating is sensitive. undetectable
Strictest Requirement:	All levels budget of attacker
Scope:	per stream
Example Application:	Secret ballot voting system openly identified - budget of any interested party anonymously identified - zero unadvertised - zero undetectable - zero

Confidentiality

Confidentiality defines how well protected the content of a communication is from snooping.

Type:	Abstract Currency
Meaning:	Level of Confidentiality, the cost of gaining illicit access to the content of a stream
Strictest Requirement:	budget of attacker
Scope:	per stream
Example Application:	Secure email - value of transmitted information

Retransmit prevention strength

This is extremely hard at the moment. This is not to say it's not a requirement.

Type:	Abstract Currency
Meaning:	The cost of retransmitting a secure piece of information should exceed the specified amount.
Strictest Requirement:	Cost of retransmitting value of information
Scope:	per stream

Membership Criteria

If a principal attempts to participate in a communication then a check will be made to see if it is allowed to do so. The requirement is that certain principals will be allowed, and others excluded. Given the application is being protected from network details there are only two types of specification available, per user, and per organization (where an organization may contain other organizations, and each user may be a member of multiple organizations). Rules could however be built on properties of a user, for example does the user own a key? Host properties could also be used, so users on slow hosts or hosts running the wrong OS could be excluded.

Type:	Macros
Meaning:	Include or exclude users (list) organizations (list) hosts (list) user properties (rule) org properties (rule) hosts properties (rule)
Strictest Requirement:	List of individual users
Scope:	per stream
Example Application:	Corporate video-conference - organization membership

Collusion prevention

Which aspects of collusion it is required to prevent. Collusion is defined as malicious co-operation between members of a secure session. Superficially, it would appear that collusion is not a relevant threat in a multicast, because everyone has the same information, however, wherever there is differentiation, it can be exploited.

Type:	{ Abstract Currency, Abstract Currency, Abstract Currency

	}
Meaning:	time race collusion - cost of colluding key encryption key (KEK) sharing - cost of colluding sharing of differential QoS (not strictly collusion as across sessions not within one) - cost of colluding
Strictest Requirement:	For all threats cost attackers combined resources
Scope:	per stream
Example Application:	A race where delay of the start signal may be allowed for, but one participant may fake packet delay while receiving the start signal from another participant.
NB:	Time race collusion is the most difficult one to prevent. Also note that while these may be requirements for some systems this does not mean there are necessarily solutions. Setting tough requirements may result in the middleware being unable to create a valid channel.

Fairness

Fairness is a meta-requirement of many other requirements. Of particular interest are Reliability and Timeliness requirements. When a communication is first created the creator may wish to specify a set of requirements for these parameters. Principals which join later may wish to set tighter limits. Fairness enforces a policy that any improvement is requirement by one principal must be matched by all others, in effect requirements can only be set for the whole group. This increases the likelihood that requirements of this kind will fail to be met. If fairness if not an issue then some parts of the network can use more friendly methods to achieve those simpler requirements.

Type:	Level of variance of the requirement that needs to be fair. For example, if the latency requirement states within 2 seconds, the level of fairness required may be that variations in latency are not more than 0.1s. This has in fact become an issue in online gaming (e.g. Quake)
Meaning:	The variance of performance with respect to any other requirement is less than the specified amount.
Scope:	per stream, per requirement

Example Application:	Networked game, latency to receive positions of players must be within 5ms for all players.

Action on compromise

The action to take on detection of compromise (until security reassured).

Type:	Enumeration
Meaning:	warn but continue pause abort
Scope:	Per stream
Strictest Requirement:	pause
Example Application:	Secure video conference - if intruder alert, everyone is warned, but they can continue while knowing not to discuss sensitive matters (cf. catering staff during a meeting).

3.2.8.1. Security Dynamics

Security dynamics are the temporal properties of the security mechanisms that are deployed. They may affect other requirements such as latency or simply be a reflection of the security limitations of the system. The requirements are often concerned with abnormal circumstances (e.g. system violation).

Mean time between compromises

This is not the same as the strength of a system. A fairly weak system may have a very long time between compromises because it is not worth breaking in to, or it is only worth it for very few people. Mean time between compromises is a combination of strength, incentive and scale.

Type:	Time
Scope:	Per stream
Strictest Requirement:	indefinite
Example Application:	Secure Shell - 1500hrs

Compromise detection time limit

The average time it must take to detect a compromise (one predicted in the design of the detection system, that is).

Type:	Time
Scope:	Per stream
Strictest Requirement:	Round trip time
Example Application:	Secure Shell - 2secs

Compromise recovery time limit

The maximum time it must take to re-seal the security after a breach. This combined with the compromise detection time limit defines how long the system must remain inactive to avoid more security breaches. For example if a compromise is detected in one minute, and recovery takes five, then one minute of traffic is now insecure and the members of the communication must remain silent for four minutes after detection while security is re-established.

Type:	Time
Scope:	Per stream
Strictest Requirement:	1 second
Example Application:	Audio conference - 10 seconds

3.2.9. Payment & Charging

Total Cost

The total cost of communication must be limited to this amount. This would be useful for transfer as opposed to stream type applications.

Type:	Currency
Meaning:	Maximum charge allowed
Scope:	Per user per stream
Strictest Requirement:	Free
Example Application:	File Transfer: comms cost must be < 1p/Mb

Cost per Time

This is the cost per unit time. Some applications may not be able to predict the duration of a communication. It may be more meaningful for those to be able to specify price per time instead.

Type:	Currency per timeS
Scope:	Per user per stream
Strictest Requirement:	Free
Example Application:	Video Conference - 15p / minute

Cost per Mb

This is the cost per unit of data. Some communications may be charged by the amount of data transferred. Some applications may prefer to specify requirements in this way.

Type:	Currency per data size
Scope:	Per user per stream
Strictest Requirement:	Free
Example Application:	Email advertising - 15p / Mb

4. Security Considerations

See comprehensive security section of taxonomy.

5. References

[Bagnall98] Bagnall Peter, Poppitt Alan, Example LSMA classifications, BT Tech report, <URL:http://www.labs.bt.com/projects/mware/>

[limitations] Pullen, M., Myjak, M. and C. Bouwens, "Limitations of Internet Protocol Suite for Distributed Simulation in the Large Multicast Environment", RFC 2502, February 1999.

[rmodp] Open Distributed Processing Reference Model (RM-ODP), ISO/IEC 10746-1 to 10746-4 or ITU-T (formerly CCITT) X.901 to X.904. Jan 1995.

[blaze95] Blaze, Diffie, Rivest, Schneier, Shimomura, Thompson and Wiener, Minimal Key Lengths for Symmetric Ciphers to Provide Adequate Commercial Security, January 1996.

6. Authors' Addresses

Peter Bagnall
c/o B54/77 BT Labs
Martlesham Heath
Ipswich, IP5 3RE
England

EMail: pete@surfaceeffect.com
Home page: http://www.surfaceeffect.com/people/pete/

Bob Briscoe
B54/74 BT Labs
Martlesham Heath
Ipswich, IP5 3RE
England

Phone: +44 1473 645196
Fax: +44 1473 640929
EMail: bob.briscoe@bt.com
Home page: http://www.labs.bt.com/people/briscorj/

Alan Poppitt
B54/77 BT Labs
Martlesham Heath
Ipswich, IP5 3RE
England

Phone: +44 1473 640889
Fax: +44 1473 640929
EMail: apoppitt@jungle.bt.co.uk
Home page: http://www.labs.bt.com/people/poppitag/

7. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.