Network Working Group   B. Aboba
Request for Comments: 2975   Microsoft Corporation
Category: Informational   J. Arkko
                                                           Ericsson
                                                      D. Harrington
                                              Cabletron Systems Inc.
                                                       October 2000

                   Introduction to Accounting Management

Status of this Memo

Copyright Notice

Abstract

   The field of Accounting Management is concerned with the collection
   of resource consumption data for the purposes of capacity and trend
   analysis, cost allocation, auditing, and billing. This document
   describes each of these problems, and discusses the issues involved
   in design of modern accounting systems.

   Since accounting applications do not have uniform security and
   reliability requirements, it is not possible to devise a single
   accounting protocol and set of security services that will meet all
   needs. Thus the goal of accounting management is to provide a set of
   tools that can be used to meet the requirements of each application.
   This document describes the currently available tools as well as the
   state of the art in accounting protocol design. A companion
   document, RFC 2924, reviews the state of the art in accounting
   attributes and record formats.

Table of Contents

1. Introduction

   The field of Accounting Management is concerned with the collection
   of resource consumption data for the purposes of capacity and trend
   analysis, cost allocation, auditing, and billing. This document
   describes each of these problems, and discusses the issues involved
   in design of modern accounting systems.

   Since accounting applications do not have uniform security and
   reliability requirements, it is not possible to devise a single
   accounting protocol and set of security services that will meet all
   needs. Thus the goal of accounting management is to provide a set of
   tools that can be used to meet the requirements of each application.
   This document describes the currently available tools as well as the
   state of the art in accounting protocol design. A companion
   document, RFC 2924, reviews the state of the art in accounting
   attributes and record formats.

1.1. Requirements language

   In this document, the key words "MAY", "MUST, "MUST NOT", "optional",
   "recommended", "SHOULD", and "SHOULD NOT", are to be interpreted as
   described in [6].

1.2. Terminology

   This document frequently uses the following terms:

   Accounting
            The collection of resource consumption data for the
            purposes of capacity and trend analysis, cost allocation,
            auditing, and billing. Accounting management requires that
            resource consumption be measured, rated, assigned, and
            communicated between appropriate parties.

   Archival accounting
            In archival accounting, the goal is to collect all
            accounting data, to reconstruct missing entries as best as
            possible in the event of data loss, and to archive data for
            a mandated time period. It is "usual and customary" for
            these systems to be engineered to be very robust against
            accounting data loss. This may include provisions for
            transport layer as well as application layer
            acknowledgments, use of non-volatile storage, interim
            accounting capabilities (stored or transmitted over the
            wire), etc. Legal or financial requirements frequently
            mandate archival accounting practices, and may often
            dictate that data be kept confidential, regardless of
            whether it is to be used for billing purposes or not.

   Rating   The act of determining the price to be charged for use of a
            resource.

   Billing   The act of preparing an invoice.

   Usage sensitive billing
            A billing process that depends on usage information to
            prepare an invoice can be said to be usage-sensitive. In
            contrast, a process that is independent of usage
            information is said to be non-usage-sensitive.

   Auditing The act of verifying the correctness of a procedure. In
            order to be able to conduct an audit it is necessary to be
            able to definitively determine what procedures were
            actually carried out so as to be able to compare this to

the recommended process. Accomplishing this may require
security services such as authentication and integrity
protection.

Cost Allocation
          The act of allocating costs between entities. Note that
          cost allocation and rating are fundamentally different
          processes. In cost allocation the objective is typically
          to allocate a known cost among several entities. In rating
          the objective is to determine the amount to be charged for
          use of a resource. In cost allocation, the cost per unit
          of resource may need to be determined; in rating, this is
          typically a given.

Interim accounting
          Interim accounting provides a snapshot of usage during a
          user's session. This may be useful in the event of a
          device reboot or other network problem that prevents the
          reception or generation of a session summary packet or
          session record. Interim accounting records can always be
          summarized without the loss of information. Note that
          interim accounting records may be stored internally on the
          device (such as in non-volatile storage) so as to survive a
          reboot and thus may not always be transmitted over the
          wire.

Session record
          A session record represents a summary of the resource
          consumption of a user over the entire session. Accounting
          gateways creating the session record may do so by
          processing interim accounting events or accounting events
          from several devices serving the same user.

Accounting Protocol
          A protocol used to convey data for accounting purposes.

Intra-domain accounting
          Intra-domain accounting involves the collection of
          information on resource usage within an administrative
          domain, for use within that domain. In intra-domain
          accounting, accounting packets and session records
          typically do not cross administrative boundaries.

Inter-domain accounting
          Inter-domain accounting involves the collection of
          information on resource usage within an administrative

          domain, for use within another administrative domain. In
          inter-domain accounting, accounting packets and session
          records will typically cross administrative boundaries.

   Real-time accounting
          Real-time accounting involves the processing of information
          on resource usage within a defined time window. Time
          constraints are typically imposed in order to limit
          financial risk.

   Accounting server
          The accounting server receives accounting data from devices
          and translates it into session records. The accounting
          server may also take responsibility for the routing of
          session records to interested parties.

1.3. Accounting management architecture

   The accounting management architecture involves interactions between
   network devices, accounting servers, and billing servers. The
   network device collects resource consumption data in the form of
   accounting metrics. This information is then transferred to an
   accounting server. Typically this is accomplished via an accounting
   protocol, although it is also possible for devices to generate their
   own session records.

   The accounting server then processes the accounting data received
   from the network device. This processing may include summarization
   of interim accounting information, elimination of duplicate data, or
   generation of session records.

   The processed accounting data is then submitted to a billing server,
   which typically handles rating and invoice generation, but may also
   carry out auditing, cost allocation, trend analysis or capacity
   planning functions. Session records may be batched and compressed by
   the accounting server prior to submission to the billing server in
   order to reduce the volume of accounting data and the bandwidth
   required to accomplish the transfer.

   One of the functions of the accounting server is to distinguish
   between inter and intra-domain accounting events and to route them
   appropriately. For session records containing a Network Access
   Identifier (NAI), described in [8], the distinction can be made by
   examining the domain portion of the NAI. If the domain portion is
   absent or corresponds to the local domain, then the session record is
   treated as an intra-domain accounting event. Otherwise, it is
   treated as an inter-domain accounting event.

Intra-domain accounting events are typically routed to the local
billing server, while inter-domain accounting events will be routed
to accounting servers operating within other administrative domains.
While it is not required that session record formats used in inter
and intra-domain accounting be the same, this is desirable, since it
eliminates translations that would otherwise be required.

Where a proxy forwarder is employed, domain-based access controls may
be employed by the proxy forwarder, rather than by the devices
themselves. The network device will typically speak an accounting
protocol to the proxy forwarder, which may then either convert the
accounting packets to session records, or forward the accounting
packets to another domain. In either case, domain separation is
typically achieved by having the proxy forwarder sort the session
records or accounting messages by destination.

Where the accounting proxy is not trusted, it may be difficult to
verify that the proxy is issuing correct session records based on the
accounting messages it receives, since the original accounting
messages typically are not forwarded along with the session records.
Therefore where trust is an issue, the proxy typically forwards the
accounting packets themselves. Assuming that the accounting protocol
supports data object security, this allows the end-points to verify
that the proxy has not modified the data in transit or snooped on the
packet contents.

The diagram below illustrates the accounting management architecture:

```
           +------------+
           |            |
           |  Network   |
           |  Device    |
           |            |
           +------------+
                 |
 Accounting |
 Protocol   |
            |
              V
           +------------+                           +------------+
           |            |                           |            |
           |   Org B    | Inter-domain session records |  Org A   |
           |   Acctg.   |<--------------------------->|  Acctg.  |
           |Proxy/Server|    or accounting protocol   |  Server  |
           |            |                           |            |
           +------------+                           +------------+
                 |                                        |
                 |                                        |
 Transfer  | Intra-domain                                |
 Protocol  | Session records                             |
           |                                             |
              V                                             V
           +------------+                           +------------+
           |            |                           |            |
           |   Org B    |                           |   Org A    |
           |   Billing  |                           |   Billing  |
           |   Server   |                           |   Server   |
           |            |                           |            |
           +------------+                           +------------+
```
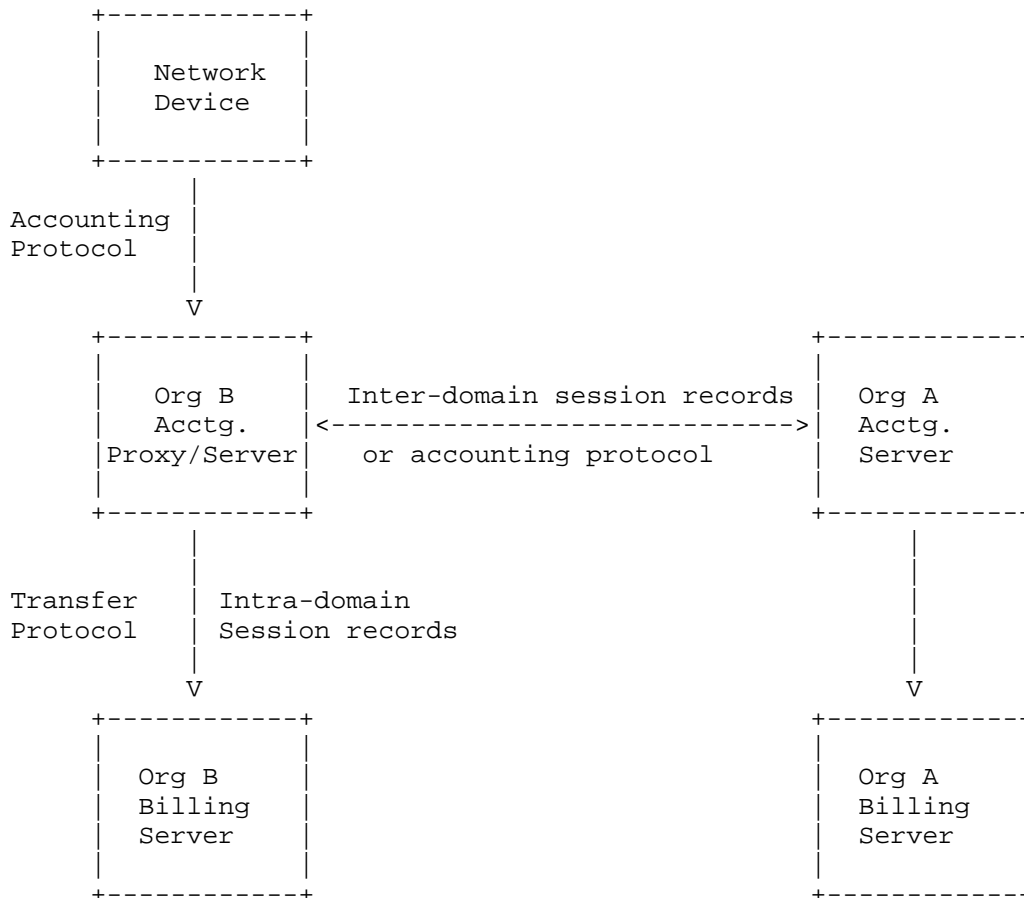
1.4. Accounting management objectives

   Accounting Management involves the collection of resource consumption
   data for the purposes of capacity and trend analysis, cost
   allocation, auditing, billing. Each of these tasks has different
   requirements.

1.4.1. Trend analysis and capacity planning

   In trend analysis and capacity planning, the goal is typically a
   forecast of future usage. Since such forecasts are inherently
   imperfect, high reliability is typically not required, and moderate
   packet loss can be tolerated. Where it is possible to use
   statistical sampling techniques to reduce data collection

   requirements while still providing the forecast with the desired
   statistical accuracy, it may be possible to tolerate high packet loss
   as long as bias is not introduced.

   The security requirements for trend analysis and capacity planning
   depend on the circumstances of data collection and the sensitivity of
   the data. Additional security services may be required when data is
   being transferred between administrative domains. For example, when
   information is being collected and analyzed within the same
   administrative domain, integrity protection and authentication may be
   used in order to guard against collection of invalid data. In
   inter-domain applications confidentiality may be desirable to guard
   against snooping by third parties.

1.4.2. Billing

   When accounting data is used for billing purposes, the requirements
   depend on whether the billing process is usage-sensitive or not.

1.4.2.1. Non-usage sensitive billing

   Since by definition, non-usage-sensitive billing does not require
   usage information, in theory all accounting data can be lost without
   affecting the billing process. Of course this would also affect
   other tasks such as trend analysis or auditing, so that such
   wholesale data loss would still be unacceptable.

1.4.2.2. Usage-sensitive billing

   Since usage-sensitive billing processes depend on usage information,
   packet loss may translate directly to revenue loss. As a result, the
   billing process may need to conform to financial reporting and legal
   requirements, and therefore an archival accounting approach may be
   needed.

   Usage-sensitive systems may also require low processing delay. Today
   credit risk is commonly managed by computerized fraud detection
   systems that are designed to detect unusual activity. While
   efficiency concerns might otherwise dictate batched transmission of
   accounting data, where there is a risk of fraud, financial exposure
   increases with processing delay. Thus it may be advisable to
   transmit each event individually to minimize batch size, or even to
   utilize quality of service techniques to minimize queuing delays. In
   addition, it may be necessary for authorization to be dependent on
   ability to pay.

Whether these techniques will be useful varies by application since
the degree of financial exposure is application-dependent. For
dial-up Internet access from a local provider, charges are typically
low and therefore the risk of loss is small. However, in the case of
dial-up roaming or voice over IP, time-based charges may be
substantial and therefore the risk of fraud is larger. In such
situations it is highly desirable to quickly detect unusual account
activity, and it may be desirable for authorization to depend on
ability to pay. In situations where valuable resources can be
reserved, or where charges can be high, very large bills may be rung
up quickly, and processing may need to be completed within a defined
time window in order to limit exposure.

Since in usage-sensitive systems, accounting data translates into
revenue, the security and reliability requirements are greater. Due
to financial and legal requirements such systems need to be able to
survive an audit. Thus security services such as authentication,
integrity and replay protection are frequently required and
confidentiality and data object integrity may also be desirable.
Application-layer acknowledgments are also often required so as to
guard against accounting server failures.

1.4.3. Auditing

With enterprise networking expenditures on the rise, interest in
auditing is increasing. Auditing, which is the act of verifying the
correctness of a procedure, commonly relies on accounting data.
Auditing tasks include verifying the correctness of an invoice
submitted by a service provider, or verifying conformance to usage
policy, service level agreements, or security guidelines.

To permit a credible audit, the auditing data collection process must
be at least as reliable as the accounting process being used by the
entity that is being audited. Similarly, security policies for the
audit should be at least as stringent as those used in preparation of
the original invoice. Due to financial and legal requirements,
archival accounting practices are frequently required in this
application.

Where auditing procedures are used to verify conformance to usage or
security policies, security services may be desired. This typically
will include authentication, integrity and replay protection as well
as confidentiality and data object integrity. In order to permit
response to security incidents in progress, auditing applications
frequently are built to operate with low processing delay.

1.4.4. Cost allocation

   The application of cost allocation and billback methods by enterprise
   customers is not yet widespread. However, with the convergence of
   telephony and data communications, there is increasing interest in
   applying cost allocation and billback procedures to networking costs,
   as is now commonly practiced with telecommunications costs.

   Cost allocation models, including traditional costing mechanisms
   described in [21]-[23] and activity-based costing techniques
   described in [24] are typically based on detailed analysis of usage
   data, and as a result they are almost always usage-sensitive.
   Whether these techniques are applied to allocation of costs between
   partners in a venture or to allocation of costs between departments
   in a single firm, cost allocation models often have profound
   behavioral and financial impacts. As a result, systems developed for
   this purposes are typically as concerned with reliable data
   collection and security as are billing applications. Due to
   financial and legal requirements, archival accounting practices are
   frequently required in this application.

1.5. Intra-domain and inter-domain accounting

   Much of the initial work on accounting management has focused on
   intra-domain accounting applications. However, with the increasing
   deployment of services such as dial-up roaming, Internet fax, Voice
   and Video over IP and QoS, applications requiring inter-domain
   accounting are becoming increasingly common.

   Inter-domain accounting differs from intra-domain accounting in
   several important ways. Intra-domain accounting involves the
   collection of information on resource consumption within an
   administrative domain, for use within that domain. In intra-domain
   accounting, accounting packets and session records typically do not
   cross administrative boundaries. As a result, intra-domain
   accounting applications typically experience low packet loss and
   involve transfer of data between trusted entities.

   In contrast, inter-domain accounting involves the collection of
   information on resource consumption within an administrative domain,
   for use within another administrative domain. In inter-domain
   accounting, accounting packets and session records will typically
   cross administrative boundaries. As a result, inter-domain
   accounting applications may experience substantial packet loss. In
   addition, the entities involved in the transfers cannot be assumed to
   trust each other.

Since inter-domain accounting applications involve transfers of
accounting data between domains, additional security measures may be
desirable. In addition to authentication, replay and integrity
protection, it may be desirable to deploy security services such as
confidentiality and data object integrity. In inter-domain
accounting each involved party also typically requires a copy of each
accounting event for invoice generation and auditing.

1.6. Accounting record production

Typically, a single accounting record is produced per session, or in
some cases, a set of interim records which can be summarized in a
single record for billing purposes. However, to support deployment
of services such as wireless access or complex billing regimes, a
more sophisticated approach is required.

It is necessary to generate several accounting records from a single
session when pricing changes during a session. For instance, the
price of a service can be higher during peak hours than off-peak.
For a session continuing from one tariff period to another, it
becomes necessary for a device to report "packets sent" during both
periods.

Time is not the only factor requiring this approach. For instance,
in mobile access networks the user may roam from one place to another
while still being connected in the same session. If roaming causes a
change in the tariffs, it is necessary to account for resource
consumed in the first and second areas. Another example is where
modifications are allowed to an ongoing session. For example, it is
possible that a session could be re-authorized with improved QoS.
This would require production of accounting records at both QoS
levels.

These examples could be addressed by using vectors or multi-
dimensional arrays to represent resource consumption within a single
session record. For example, the vector or array could describe the
resource consumption for each combination of factors, e.g. one data
item could be the number of packets during peak hour in the area of
the home operator. However, such an approach seems complicated and
inflexible and as a result, most current systems produce a set of
records from one session. A session identifier needs to be present
in the records to permit accounting systems to tie the records
together.

In most cases, the network device will determine when multiple
session records are needed, as the local device is aware of factors
affecting local tariffs, such as QoS changes and roaming. However,
future systems are being designed that enable the home domain to

control the generation of accounting records. This is of importance
in inter-domain accounting or when network devices do not have tariff
information. The centralized control of accounting record production
can be realized, for instance, by having authorization servers
require re-authorization at certain times and requiring the
production of accounting records upon each re-authorization.

In conclusion, in some cases it is necessary to produce multiple
accounting records from a single session. It must be possible to do
this without requiring the user to start a new session or to re-
authenticate. The production of multiple records can be controlled
either by the network device or by the AAA server. The requirements
for timeliness, security and reliability in multiple record sessions
are the same as for single-record sessions.

1.7. Requirements summary

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                   |                   |
|     Usage         |   Intra-domain    |   Inter-domain    |
|                   |                   |                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   | Robustness vs.    | Robustness vs.    |
|                   | packet loss       | packet loss       |
|  Capacity         |                   |                   |
|  Planning         | Integrity,        | Integrity,        |
|                   | authentication,   | authentication,   |
|                   | replay protection | replay prot.      |
|                   | [confidentiality] | confidentiality   |
|                   |                   | [data object sec.]|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Non-usage        | Integrity,        | Integrity,        |
|  Sensitive        | authentication,   | authentication,   |
|  Billing          | replay protection | replay protection |
|                   | [confidentiality] | confidentiality   |
|                   |                   | [data object sec.]|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   | Archival          | Archival          |
|  Usage            | accounting        | accounting        |
|  Sensitive        | Integrity,        | Integrity,        |
|  Billing,         | authentication,   | authentication,   |
|  Cost             | replay protection | replay prot.      |
|  Allocation &     | [confidentiality] | confidentiality   |
|  Auditing         | [Bounds on        | [data object sec.]|
|                   |  processing delay]| [Bounds on        |
|                   |                   | processing delay] |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   | Archival          | Archival          |
|  Time             | accounting        | accounting        |
|  Sensitive        | Integrity,        | Integrity,        |
|  Billing,         | authentication,   | authentication,   |
|  fraud            | replay protection | replay prot.      |
|  detection,       | [confidentiality] | confidentiality   |
|  roaming          |                   | [Data object      |
|                   |                   |  security and     |
|                   | Bounds on         |  receipt support] |
|                   |  processing delay | Bounds on         |
|                   |                   |  processing delay |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Key
[] = optional

2. Scaling and reliability

   With the continuing growth of the Internet, it is important that
   accounting management systems be scalable and reliable. This section
   discusses the resources consumed by accounting management systems as
   well as the scalability and reliability properties exhibited by
   various data collection and transport models.

2.1. Fault resilience

   As noted earlier, in applications such as usage-sensitive billing,
   cost allocation and auditing, an archival approach to accounting is
   frequently mandated, due to financial and legal requirements. Since
   in such situations loss of accounting data can translate to revenue
   loss, there is incentive to engineer a high degree of fault
   resilience. Faults which may be encountered include:

      Packet loss
      Accounting server failures
      Network failures
      Device reboots

   To date, much of the debate on accounting reliability has focused on
   resilience against packet loss and the differences between UDP, SCTP
   and TCP-based transport. However, it should be understood that
   resilience against packet loss is only one aspect of meeting
   archival accounting requirements.

   As noted in [18], "once the cable is cut you don't need more
   retransmissions, you need a *lot* more voltage." Thus, the choice of
   transport has no impact on resilience against faults such as network
   partition, accounting server failures or device reboots. What does
   provide resilience against these faults is non-volatile storage.

   The importance of non-volatile storage in design of reliable
   accounting systems cannot be over-emphasized. Without non-volatile
   storage, event-driven systems will lose data once the transmission
   timeout has been exceeded, and batching designs will experience data
   loss once the internal memory used for accounting data storage has
   been exceeded. Via use of non-volatile storage, and internally
   stored interim records, most of these data losses can be avoided.

   It may even be argued that non-volatile storage is more important to
   accounting reliability than network connectivity, since for many
   years reliable accounting systems were implemented based solely on
   physical storage, without any network connectivity. For example,

phone usage data used to be stored on paper, film, or magnetic media
and carried from the place of collection to a central location for
bill processing.

2.1.1. Interim accounting

Interim accounting provides protection against loss of session
summary data by providing checkpoint information that can be used to
reconstruct the session record in the event that the session summary
information is lost. This technique may be applied to any data
collection model (i.e. event-driven or polling) and is supported in
both RADIUS [25] and in TACACS+.

While interim accounting can provide resilience against packet loss,
server failures, short-duration network failures, or device reboot,
its applicability is limited. Transmission of interim accounting
data over the wire should not be thought of as a mainstream
reliability improvement technique since it increases use of network
bandwidth in normal operation, while providing benefits only in the
event of a fault.

Since most packet loss on the Internet is due to congestion, sending
interim accounting data over the wire can make the problem worse by
increasing bandwidth usage. Therefore on-the-wire interim accounting
is best restricted to high-value accounting data such as information
on long-lived sessions. To protect against loss of data on such
sessions, the interim reporting interval is typically set several
standard deviations larger than the average session duration. This
ensures that most sessions will not result in generation of interim
accounting events and the additional bandwidth consumed by interim
accounting will be limited. However, as the interim accounting
interval decreases toward the average session time, the additional
bandwidth consumed by interim accounting increases markedly, and as a
result, the interval must be set with caution.

Where non-volatile storage is unavailable, interim accounting can
also result in excessive consumption of memory that could be better
allocated to storage of session data. As a result, implementors
should be careful to ensure that new interim accounting data
overwrites previous data rather than accumulating additional interim
records in memory, thereby worsening the buffer exhaustion problem.

Given the increasing popularity of non-volatile storage for use in
consumer devices such as digital cameras, such devices are rapidly
declining in price. This makes it increasingly feasible for network
devices to include built-in support for non-volatile storage. This
can be accomplished, for example, by support for compact PCMCIA
cards.

Where non-volatile storage is available, this can be used to store
interim accounting data. Stored interim events are then replaced by
updated interim events or by session data when the session completes.
The session data can itself be erased once the data has been
transmitted and acknowledged at the application layer. This approach
avoids interim data being transmitted over the wire except in the
case of a device reboot. When a device reboots, internally stored
interim records are transferred to the accounting server.

2.1.2. Multiple record sessions

Generation of multiple accounting records within a session can
introduce scalability problems that cannot be controlled using the
techniques available in interim accounting.

For example, in the case of interim records kept in non-volatile
storage, it is possible to overwrite previous interim records with
the most recent one or summarize them to a session record. Where
interim updates are sent over the wire, it is possible to control
bandwidth usage by adjusting the interim accounting interval.

These measures are not applicable where multiple session records are
produced from a single session, since these records cannot be
summarized or overwritten without loss of information. As a result,
multiple record production can result in increased consumption of
bandwidth and memory. Implementors should be careful to ensure that
worst-case multiple record processing requirements do not exceed the
capabilities of their systems.

As an example, a tariff change at a particular time of day could, if
implemented carelessly, create a sudden peak in the consumption of
memory and bandwidth as the records need to be stored and/or
transported. Rather than attempting to send all of the records at
once, it may be desirable to keep them in non-volatile storage and
send all of the related records together in a batch when the session
completes. It may also be desirable to shape the accounting traffic
flow so as to reduce the peak bandwidth consumption. This can be
accomplished by introduction of a randomized delay interval. If the
home domain can also control the generation of multiple accounting
records, the estimation of the worst-case processing requirements can
be very difficult.

2.1.3. Packet loss

As packet loss is a fact of life on the Internet, accounting
protocols dealing with session data need to be resilient against
packet loss. This is particularly important in inter-domain
accounting, where packets often pass through Network Access Points

(NAPs) where packet loss may be substantial. Resilience against
packet loss can be accomplished via implementation of a retry
mechanism on top of UDP, or use of TCP [7] or SCTP [26]. On-the-wire
interim accounting provides only limited benefits in mitigating the
effects of packet loss.

UDP-based transport is frequently used in accounting applications.
However, this is not appropriate in all cases. Where accounting data
will not fit within a single UDP packet without fragmentation, use of
TCP or SCTP transport may be preferred to use of multiple round-trips
in UDP. As noted in [47] and [49], this may be an issue in the
retrieval of large tables.

In addition, in cases where congestion is likely, such as in inter-
domain accounting, TCP or SCTP congestion control and round-trip time
estimation will be very useful, optimizing throughput. In
applications which require maintenance of session state, such as
simultaneous usage control, TCP and application-layer keep alive
packets or SCTP with its built-in heartbeat capabilities provide a
mechanism for keeping track of session state.

When implementing UDP retransmission, there are a number of issues to
keep in mind:

    Data model
    Retry behavior
    Congestion control
    Timeout behavior

Accounting reliability can be influenced by how the data is modeled.
For example, it is almost always preferable to use cumulative
variables rather than expressing accounting data in terms of a change
from a previous data item. With cumulative data, the current state
can be recovered by a successful retrieval, even after many packets
have been lost. However, if the data is transmitted as a change then
the state will not be recovered until the next cumulative update is
sent. Thus, such implementations are much more vulnerable to packet
loss, and should be avoided wherever possible.

In designing a UDP retry mechanism, it is important that the retry
timers relate to the round-trip time, so that retransmissions will
not typically occur within the period in which acknowledgments may be
expected to arrive. Accounting bandwidth may be significant in some
circumstances, so that the added traffic due to unnecessary
retransmissions may increase congestion levels.

Congestion control in accounting data transfer is a somewhat
controversial issue. Since accounting traffic is often considered
mission-critical, it has been argued that congestion control is not a
requirement; better to let other less-critical traffic back off in
response to congestion. Moreover, without non-volatile storage,
congestive back-off in accounting applications can result in data
loss due to buffer exhaustion.

However, it can also be argued that in modern accounting
implementations, it is possible to implement congestion control while
improving throughput and maintaining high reliability. In
circumstances where there is sustained packet loss, there simply is
not sufficient capacity to maintain existing transmission rates.
Thus, aggregate throughput will actually improve if congestive back-
off is implemented. This is due to elimination of retransmissions
and the ability to utilize techniques such as RED to desynchronize
flows. In addition, with QoS mechanisms such as differentiated
services, it is possible to mark accounting packets for preferential
handling so as to provide for lower packet loss if desired. Thus
considerable leeway is available to the network administrator in
controlling the treatment of accounting packets and hard coding
inelastic behavior is unnecessary. Typically, systems implementing
non-volatile storage allow for backlogged accounting data to be
placed in non-volatile storage pending transmission, so that buffer
exhaustion resulting from congestive back-off need not be a concern.

Since UDP is not really a transport protocol, UDP-based accounting
protocols such as [4] often do not prescribe timeout behavior. Thus
implementations may exhibit widely different behavior. For example,
one implementation may drop accounting data after three constant
duration retries to the same server, while another may implement
exponential back-off to a given server, then switch to another
server, up to a total timeout interval of twelve hours, while storing
the untransmitted data on non-volatile storage. The practical
difference between these approaches is substantial; the former
approach will not satisfy archival accounting requirements while the
latter may. More predictable behavior can be achieved via use of
SCTP or TCP transport.

2.1.4. Accounting server failover

In the event of a failure of the primary accounting server, it is
desirable for the device to failover to a secondary server.
Providing one or more secondary servers can remove much of the risk
of accounting server failure, and as a result use of secondary
servers has become commonplace.

For protocols based on TCP, it is possible for the device to maintain
connections to both the primary and secondary accounting servers,
using the secondary connection after expiration of a timer on the
primary connection. Alternatively, it is possible to open a
connection to the secondary accounting server after a timeout or loss
of the primary connection, or on expiration of a timer. Thus,
accounting protocols based on TCP are capable of responding more
rapidly to connectivity failures than TCP timeouts would otherwise
allow, at the expense of an increased risk of duplicates.

With SCTP, it is possible to control transport layer timeout
behavior, and therefore it is not necessary for the accounting
application to maintain its own timers. SCTP also enables
multiplexing of multiple connections within a single transport
connection, all maintaining the same congestion control state,
avoiding the "head of line blocking" issues that can occur with TCP.
However, since SCTP is not widely available, use of this transport
can impose an additional implementation burden on the designer.

For protocols using UDP, transmission to the secondary server can
occur after a number of retries or timer expiration. For
compatibility with congestion avoidance, it is advisable to
incorporate techniques such as round-trip-time estimation, slow start
and congestive back-off. Thus the accounting protocol designer
utilizing UDP often is lead to re-inventing techniques already
existing in TCP and SCTP. As a result, the use of raw UDP transport
in accounting applications is not recommended.

With any transport it is possible for the primary and secondary
accounting servers to receive duplicate packets, so support for
duplicate elimination is required. Since accounting server failures
can result in data accumulation on accounting clients, use of non-
volatile storage can ensure against data loss due to transmission
timeouts or buffer exhaustion. On-the-wire interim accounting
provides only limited benefits in mitigating the effects of
accounting server failures.

2.1.5. Application layer acknowledgments

It is possible for the accounting server to experience partial
failures. For example, a failure in the database back end could
leave the accounting retrieval process or thread operable while the
process or thread responsible for storing the data is non-functional.
Similarly, it is possible for the accounting application to run out
of disk space, making it unable to continue storing incoming session
records.

In such cases it is desirable to distinguish between transport layer
acknowledgment and application layer acknowledgment. Even though
both acknowledgments may be sent within the same packet (such as a
TCP segment carrying an application layer acknowledgment along with a
piggy-backed ACK), the semantics are different. A transport-layer
acknowledgment means "the transport layer has taken responsibility
for delivering the data to the application", while an application-
layer acknowledgment means "the application has taken responsibility
for the data".

A common misconception is that use of TCP transport guarantees that
data is delivered to the application. However, as noted in RFC 793
[7]:

   An acknowledgment by TCP does not guarantee that the data has been
   delivered to the end user, but only that the receiving TCP has taken
   the responsibility to do so.

Therefore, if receiving TCP fails after sending the ACK, the
application may not receive the data. Similarly, if the application
fails prior to committing the data to stable storage, the data may be
lost. In order for a sending application to be sure that the data it
sent was received by the receiving application, either a graceful
close of the TCP connection or an application-layer acknowledgment is
required. In order to protect against data loss, it is necessary that
the application-layer acknowledgment imply that the data has been
written to stable storage or suitably processed so as to guard
against loss.

In the case of partial failures, it is possible for the transport
layer to acknowledge receipt via transport layer acknowledgment,
without having delivered the data to the application. Similarly, the
application may not complete the tasks necessary to take
responsibility for the data.

For example, an accounting server may receive data from the transport
layer but be incapable of storing it data due to a back end database
problem or disk fault. In this case it should not send an
application layer acknowledgment, even though a a transport layer
acknowledgment is appropriate. Rather, an application layer error
message should be sent indicating the source of the problem, such as
"Backend store unavailable".

Thus application-layer acknowledgment capability requires not only
the ability to acknowledge when the application has taken
responsibility for the data, but also the ability to indicate when
the application has not taken responsibility for the data, and why.

2.1.6. Network failures

   Network failures may result in partial or complete loss of
   connectivity for the accounting client. In the event of partial
   connectivity loss, it may not be possible to reach the primary
   accounting server, in which case switch over to the secondary
   accounting server is necessary. In the event of a network partition,
   it may be necessary to store accounting events in device memory or
   non-volatile storage until connectivity can be re-established.

   As with accounting server failures, on-the-wire interim accounting
   provides only limited benefits in mitigating the effects of network
   failures.

2.1.7. Device reboots

   In the event of a device reboot, it is desirable to minimize the loss
   of data on sessions in progress. Such losses may be significant even
   if the devices themselves are very reliable, due to long-lived
   sessions, which can comprise a significant fraction of total resource
   consumption. To guard against loss of these high-value sessions,
   interim accounting data is typically transmitted over the wire. When
   interim accounting in-place is combined with non-volatile storage it
   becomes possible to guard against data loss in much shorter sessions.
   This is possible since interim accounting data need only be stored in
   non-volatile memory until the session completes, at which time the
   interim data may be replaced by the session record. As a result,
   interim accounting data need never be sent over the wire, and it is
   possible to decrease the interim interval so as to provide a very
   high degree of protection against data loss.

2.1.8. Accounting proxies

   In order to maintain high reliability, it is important that
   accounting proxies pass through transport and application layer
   acknowledgments and do not store and forward accounting packets.
   This enables the end-systems to control re-transmission behavior and
   utilize techniques such as non-volatile storage and secondary servers
   to improve resilience.

   Accounting proxies sending a transport or application layer ACK to
   the device without receiving one from the accounting server fool the
   device into thinking that the accounting request had been accepted by
   the accounting server when this is not the case. As a result, the
   device can delete the accounting packet from non-volatile storage
   before it has been accepted by the accounting server. The leaves the

accounting proxy responsible for delivering accounting packets. If
the accounting proxy involves moving parts (e.g. a disk drive) while
the devices do not, overall system reliability can be reduced.

Store and forward accounting proxies only add value in situations
where the accounting subsystem is unreliable. For example, where
devices do not implement non-volatile storage and the accounting
protocol lacks transport and application layer reliability, locating
the accounting proxy (with its stable storage) close to the device
can reduce the risk of data loss.

However, such systems are inherently unreliable so that they are only
appropriate for use in capacity planning or non-usage sensitive
billing applications. If archival accounting reliability is desired,
it is necessary to engineer a reliable accounting system from the
start using the techniques described in this document, rather than
attempting to patch an inherently unreliable system by adding store
and forward accounting proxies.

2.1.9. Fault resilience summary

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                                    |
|   Fault           |   Counter-measures                 |
|                   |                                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                                    |
|   Packet          |   Retransmission based on RTT      |
|   loss            |   Congestion control               |
|                   |   Well-defined timeout behavior    |
|                   |   Duplicate elimination            |
|                   |   Interim accounting*              |
|                   |   Non-volatile storage             |
|                   |   Cumulative variables             |
|                   |                                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                                    |
|   Accounting      |   Primary-secondary servers        |
|   server & net    |   Duplicate elimination            |
|   failures        |   Interim accounting*              |
|                   |   Application layer ACK & error msgs. |
|                   |   Non-volatile storage             |
|                   |                                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   |                                    |
|   Device          |   Interim accounting*              |
|   reboots         |   Non-volatile storage             |
|                   |                                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Key
* = limited usefulness without non-volatile storage

Note: Accounting proxies are not a reliability
enhancement mechanism.

2.2. Resource consumption

In the process of growing to meet the needs of providers and
customers, accounting management systems consume a variety of
resources, including:

    Network bandwidth
    Memory
    Non-volatile storage
    State on the accounting management system
    CPU on the management system and managed devices

In order to understand the limits to scaling, we examine each of
these resources in turn.

2.2.1. Network bandwidth

Accounting management systems consume network bandwidth in
transferring accounting data. The network bandwidth consumed is
proportional to the amount of data transferred, as well as required
network overhead. Since accounting data for a given event may be 100
octets or less, if each event is transferred individually, overhead
can represent a considerable proportion of total bandwidth
consumption. As a result, it is often desirable to transfer
accounting data in batches, enabling network overhead to be spread
over a larger payload, and enabling efficient use of compression. As
noted in [48], compression can be enabled in the accounting protocol,
or can be done at the IP layer as described in [5].

2.2.2. Memory

In accounting systems without non-volatile storage, accounting data
must be stored in volatile memory during the period between when it
is generated and when it is transferred. The resulting memory
consumption will depend on retry and retransmission algorithms.
Since systems designed for high reliability will typically wish to
retry for long periods, or may store interim accounting data, the
resulting memory consumption can be considerable. As a result, if
non-volatile storage is unavailable, it may be desirable to compress
accounting data awaiting transmission.

As noted earlier, implementors of interim accounting should take care
to ensure against excessive memory usage by overwriting older interim
accounting data with newer data for the same session rather than
accumulating interim data in the buffer.

2.2.3. Non-volatile storage

Since accounting data stored in memory will typically be lost in the
event of a device reboot or a timeout, it may be desirable to provide
non-volatile storage for undelivered accounting data. With the costs
of non-volatile storage declining rapidly, network devices will be
increasingly capable of incorporating non-volatile storage support
over the next few years.

Non-volatile storage may be used to store interim or session records.
As with memory utilization, interim accounting overwrite is desirable
so as to prevent excessive storage consumption. Note that the use of
ASCII data representation enables use of highly efficient text
compression algorithms that can minimize storage requirements. Such

compression algorithms are only typically applied to session records
so as to enable implementation of interim data overwrite.

2.2.4. State on the accounting management system

In order to keep track of received accounting data, accounting
management systems may need to keep state on managed devices or
concurrent sessions. Since the number of devices is typically much
smaller than the number of concurrent sessions, it is desirable to
keep only per-device state if possible.

2.2.5. CPU requirements

CPU consumption of the managed and managing nodes will be
proportional to the complexity of the required accounting processing.
Operations such as ASN.1 encoding and decoding,
compression/decompression, and encryption/decryption can consume
considerable resources, both on accounting clients and servers.

The effect of these operations on accounting system reliability
should not be under-estimated, particularly in the case of devices
with moderate CPU resources. In the event that devices are over-
taxed by accounting tasks, it is likely that overall device
reliability will suffer.

2.2.6. Efficiency measures

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   Resource       |  Efficiency measures           |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   Network        |  Batching                      |
|   Bandwidth      |  Compression                   |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   Memory         |  Compression                   |
|                  |  Interim accounting overwrite  |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   Non-volatile   |  Compression                   |
|   Storage        |  Interim accounting overwrite  |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   System         |  Per-device state              |
|   state          |                                |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  |                                |
|   CPU            |  Hardware assisted             |
|   requirements   |     compression/encryption     |
|                  |                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

2.3. Data collection models

   Several data collection models are currently in use today for the
   purposes of accounting data collection. These include:

      Polling model
      Event-driven model without batching
      Event-driven model with batching
      Event-driven polling model

2.3.1. Polling model

   In the polling model, an accounting manager will poll devices for
   accounting information at regular intervals. In order to ensure
   against loss of data, the polling interval will need to be shorter
   than the maximum time that accounting data can be stored on the
   polled device. For devices without non-volatile stage, this is
   typically determined by available memory; for devices with non-
   volatile storage the maximum polling interval is determined by the
   size of non-volatile storage.

   The polling model results in an accumulation of data within
   individual devices, and as a result, data is typically transferred to
   the accounting manager in a batch, resulting in an efficient transfer
   process. In terms of Accounting Manager state, polling systems scale
   with the number of managed devices, and system bandwidth usage scales
   with the amount of data transferred.

   Without non-volatile storage, the polling model results in loss of
   accounting data due to device reboots, but not due to packet loss or
   network failures of sufficiently short duration to be handled within
   available memory. This is because the Accounting Manager will
   continue to poll until the data is received. In situations where
   operational difficulties are encountered, the volume of accounting
   data will frequently increase so as to make data loss more likely.
   However, in this case the polling model will detect the problem since
   attempts to reach the managed devices will fail.

   The polling model scales poorly for implementation of shared use or
   roaming services, including wireless data, Internet telephony, QoS
   provisioning or Internet access. This is because in order to
   retrieve accounting data for users within a given domain, the
   Accounting Management station would need to periodically poll all
   devices in all domains, most of which would not contain any relevant
   data. There are also issues with processing delay, since use of a
   polling interval also implies an average processing delay of half the
   polling interval. This may be too high for accounting data that
   requires low processing delay. Thus the event-driven polling or the
   pure event-driven approach is more appropriate for usage sensitive
   billing applications such as shared use or roaming implementations.

   Per-device state is typical of polling-based network management
   systems, which often also carry out accounting management functions,
   since network management systems need to keep track of the state of
   network devices for operational purposes. These systems offer
   average processing delays equal to half the polling interval.

2.3.2. Event-driven model without batching

   In the event-driven model, a device will contact the accounting
   server or manager when it is ready to transfer accounting data. Most
   event-driven accounting systems, such as those based on RADIUS
   accounting, described in [4], transfer only one accounting event per
   packet, which is inefficient.

   Without non-volatile storage, a pure event-driven model typically
   stores accounting events that have not yet been delivered only until
   the timeout interval expires. As a result this model has the
   smallest memory requirements. Once the timeout interval has expired,
   the accounting event is lost, even if the device has sufficient
   buffer space to continue to store it. As a result, the event-driven
   model is the least reliable, since accounting data loss will occur
   due to device reboots, sustained packet loss, or network failures of
   duration greater than the timeout interval. In event-driven
   protocols without a "keep alive" message, accounting servers cannot
   assume a device failure should no messages arrive for an extended
   period. Thus, event-driven accounting systems are typically not
   useful in monitoring of device health.

   The event-driven model is frequently used in shared use networks and
   roaming, since this model sends data to the recipient domains without
   requiring them to poll a large number of devices, most of which have
   no relevant data. Since the event-driven model typically does not
   support batching, it permits accounting records to be sent with low
   processing delay, enabling application of fraud prevention
   techniques. However, because roaming accounting events are
   frequently of high value, the poor reliability of this model is an
   issue. As a result, the event-driven polling model may be more
   appropriate.

   Per-session state is typical of event-driven systems without
   batching. As a result, the event-driven approach scales poorly.
   However, event-driven systems offer the lowest processing delay since
   events are processed immediately and there is no possibility of an
   event requiring low processing delay being caught behind a batch
   transfer.

2.3.3. Event-driven model with batching

   In the event-driven model with batching, a device will contact the
   accounting server or manager when it is ready to transfer accounting
   data. The device can contact the server when a batch of a given size
   has been gathered, when data of a certain type is available or after
   a minimum time period has elapsed. Such systems can transfer more
   than one accounting event per packet and are thus more efficient.

An event-driven system with batching will store accounting events
that have not yet been delivered up to the limits of memory. As a
result, accounting data loss will occur due to device reboots, but
not due to packet loss or network failures of sufficiently short
duration to be handled within available memory. Note that while
transfer efficiency will increase with batch size, without non-
volatile storage, the potential data loss from a device reboot will
also increase.

Where event-driven systems with batching have a keep-alive interval
and run over reliable transport, the accounting server can assume
that a failure has occurred if no messages are received within the
keep-alive interval. Thus, such implementations can be useful in
monitoring of device health. When used for this purpose the average
time delay prior to failure detection is one half the keep-alive
interval.

Through implementation of a scheduling algorithm, event-driven
systems with batching can deliver appropriate service to accounting
events that require low processing delay. For example, high-value
inter-domain accounting events could be sent immediately, thus
enabling use of fraud-prevention techniques, while all other events
would be batched. However, there is a possibility that an event
requiring low processing delay will be caught behind a batch transfer
in progress. Thus the maximum processing delay is proportional to
the maximum batch size divided by the link speed.

Event-driven systems with batching scale with the number of active
devices. As a result this approach scales better than the pure
event-driven approach, or even the polling approach, and is
equivalent in terms of scaling to the event-driven polling approach.
However, the event-driven batching approach has lower processing
delay than the event-driven polling approach, since delivery of
accounting data requires fewer round-trips and events requiring low
processing delay can be accommodated if a scheduling algorithm is
employed.

2.3.4. Event-driven polling model

In the event-driven polling model an accounting manager will poll the
device for accounting data only when it receives an event. The
accounting client can generate an event when a batch of a given size
has been gathered, when data of a certain type is available or after
a minimum time period has elapsed. Note that while transfer
efficiency will increase with batch size, without non-volatile
storage, the potential data loss from a device reboot will also
increase.

Without non-volatile storage, an event-driven polling model will lose
data due to device reboots, but not due to packet loss, or network
partitions of short-duration. Unless a minimum delivery interval is
set, event-driven polling systems are not useful in monitoring of
device health.

The event-driven polling model can be suitable for use in roaming
since it permits accounting data to be sent to the roaming partners
with low processing delay. At the same time non-roaming accounting
can be handled via more efficient polling techniques, thereby
providing the best of both worlds.

Where batching can be implemented, the state required in event-driven
polling can be reduced to scale with the number of active devices.
If portions of the network vary widely in usage, then this state may
actually be less than that of the polling approach. Note that
processing delay in this approach is higher than in event-driven
accounting with batching since at least two round-trips are required
to deliver data: one for the event notification, and one for the
resulting poll.

2.3.5. Data collection summary

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 |                 |                     |
|     Model       |      Pros       |       Cons          |
|                 |                 |                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Polling       | Per-device state| Not robust          |
|                 | Robust against  |  against device     |
|                 |   packet loss   |  reboot, server     |
|                 | Batch transfers |  or network         |
|                 |                 |  failures*          |
|                 |                 | Polling interval    |
|                 |                 |  determined by      |
|                 |                 |  storage limit      |
|                 |                 | High processing     |
|                 |                 |  delay              |
|                 |                 | Unsuitable for      |
|                 |                 |  use in roaming     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Event-driven,   | Lowest processing| Not robust         |
|   no batching   |  delay          |  against packet     |
|                 | Suitable for    |  loss, device       |
|                 |  use in roaming |  reboot, or         |
|                 |                 |  network            |
|                 |                 |  failures*          |
|                 |                 | Low efficiency      |
|                 |                 | Per-session state   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Event-driven,   | Single round-trip| Not robust         |
|   with batching |  latency        |  against device     |
|       and       | Batch transfers |  reboot, network    |
|    scheduling   | Suitable for    |  failures*          |
|                 |  use in roaming |                     |
|                 | Per active device|                    |
|                 |  state          |                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Event-driven    | Batch transfers | Not robust          |
|   polling       | Suitable for    |  against device     |
|                 |  use in roaming |  reboot, network    |
|                 | Per active device|  failures*         |
|                 |  state          | Two round-trip      |
|                 |                 |  latency            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Key
* = addressed by non-volatile storage

3. Review of Accounting Protocols

   Accounting systems have been successfully implemented using protocols
   such as RADIUS, TACACS+, and SNMP. This section describes the
   characteristics of each of these protocols.

3.1. RADIUS

   RADIUS accounting, described in [4], was developed as an add-on to
   the RADIUS authentication protocol, described in [3]. As a result,
   RADIUS accounting shares the event-driven approach of RADIUS
   authentication, without support for batching or polling. As a
   result, RADIUS accounting scales with the number of accounting events
   instead of the number of devices, and accounting transfers are
   inefficient.

   Since RADIUS accounting is based on UDP and timeout and retry
   parameters are not specified, implementations vary widely in their
   approach to reliability, with some implementations retrying until
   delivery or buffer exhaustion, and others losing accounting data
   after a few retries. Since RADIUS accounting does not provide for
   application-layer acknowledgments or error messages, a RADIUS
   Accounting-Response is equivalent to a transport-layer acknowledgment
   and provides no protection against application layer malfunctions.
   Due to the lack of reliability, it is not possible to do simultaneous
   usage control based on RADIUS accounting alone. Typically another
   device data source is required, such as polling of a session MIB or a
   command-line session over telnet.

   RADIUS accounting implementations are vulnerable to packet loss as
   well as application layer failures, network failures and device
   reboots. These deficiencies are magnified in inter-domain accounting
   as is required in roaming ([1],[2]). On the other hand, the event-
   driven approach of RADIUS accounting is useful where low processing
   delay is required, such as credit risk management or fraud detection.

   While RADIUS accounting does provide hop-by-hop authentication and
   integrity protection, and IPSEC can be employed to provide hop-by-hop
   confidentiality, data object security is not supported, and thus
   systems based on RADIUS accounting are not capable of being deployed
   with untrusted proxies, or in situations requiring auditability, as
   noted in [2].

   While RADIUS does not support compression, IP compression, described
   in [5], can be employed to provide this. While in principle
   extensible with the definition of new attributes, RADIUS suffers from
   the very small standard attribute space (256 attributes).

3.2. TACACS+

   TACACS+ offers an accounting model with start, stop, and interim
   update messages. Since TACACS+ is based on TCP, implementations are
   typically resilient against packet loss and short-lived network
   partitions, and TACACS+ scales with the number of devices. Since
   TACACS+ runs over TCP, it offers support for both transport layer and
   application layer acknowledgments, and is suitable for simultaneous
   usage control and handling of accounting events that require moderate
   though not the lowest processing delay.

   TACACS+ provides for hop-by-hop authentication and integrity
   protection as well as hop-by-hop confidentiality. Data object
   security is not supported, and therefore systems based on TACACS+
   accounting are not deployable in the presence of untrusted proxies.
   While TACACS+ does not support compression, IP compression, described
   in [5], can be employed to provide this.

3.3. SNMP

   SNMP, described in [19],[27]-[41], has been widely deployed in a wide
   variety of intra-domain accounting applications, typically using the
   polling data collection model. Polling allows data to be collected
   on multiple accounting events simultaneously, resulting in per-device
   state. Management applications are able to retry requests when a
   response is not received, providing resiliency against packet loss or
   even short-lived network partitions. Implementations without non-
   volatile storage are not robust against device reboots or network
   failures, but when combined with non-volatile storage they can be
   made highly reliable.

   SMIv1, the data modeling language of SNMPv1, has traps to permit
   trap-directed polling, but the traps are not acknowledged, and lost
   traps can lead to a loss of data. SMIv2, used by SNMPv2c and SNMPv3,
   has Inform Requests which are acknowledged notifications. This makes
   it possible to implement a more reliable event-driven polling model
   or event-driven batching model. However, we are not aware of any
   SNMP-based accounting implementations currently built on the use of
   Informs.

3.3.1. Security services

   SNMPv1 and SNMPv2c support per-packet authentication and read-only
   and read-write access profiles, via the community string. This
   clear-text password approach provides only trivial authentication,
   and no per-packet integrity checks, replay protection or
   confidentiality. View-based access control [40] can be supported
   using the snmpCommunityMIB, defined in [11], and SNMPv1 or SNMPv2c

   messages. The updated SNMP architecture [rfc2571] supports per-
   packet hop-by-hop authentication, integrity and replay protection,
   confidentiality and access control.

   The SNMP User Security Model (USM) [38] uses shared secrets, and when
   the product of the number of domains and devices is large, such as in
   inter-domain accounting applications, the number of shared secrets
   can get out of hand. The localized key capability in USM allows a
   manager to have one central key, sharing it with many SNMP entities
   in a localized way while preventing the other entities from getting
   at each other's data. This can assist in cross-domain security if
   deployed properly.

   SNMPv3 does not support end-to-end data object integrity and
   confidentiality; SNMP proxy entities decrypt and re-encrypt the data
   they forward. In the presence of an untrusted proxy entity, this
   would be inadequate.

3.3.2. Application layer acknowledgments

   SNMP uses application-layer acknowledgment to indicate that data has
   been processed. SNMP Responses to get, get-next, or get-bulk
   requests return the requested data, or an error code indicating the
   nature of the error encountered.

   A noError SNMP Response to a SET command indicates that the requested
   assignments were made by the application. SNMP SETs are atomic; the
   command either succeeds or fails. An error-response indicates that
   the entity received the request, but did not succeed in executing it.

   Notifications do not use acknowledgements to indicate that data has
   been processed. The Inform notification returns an acknowledgement
   of receipt, but not of processing, by design. Since the updated SNMP
   architecture treats entities as peers with varying levels of
   functionality, it is possible to use SETs in either direction between
   cooperating entities to achieve processing acknowledgements.

   There are eighteen SNMP error codes. The design of SNMP makes
   service-specific error codes unnecessary and undesirable.

3.3.3. Proxy forwarders

   In the accounting management architecture, proxy forwarders play an
   important role, forwarding intra and inter-domain accounting events
   to the correct destinations. The proxy forwarder may also play a
   role in a polling or event-driven polling architecture.

The functionality of an SNMP Proxy Forwarder is defined in [39]. For
example, the network devices may be configured to send notifications
for all domains to the Proxy Forwarder, and the devices may be
configured to allow the Proxy Forwarder to access all MIB data.

The use of proxy forwarders may reduce the number of shared secrets
required for inter-domain accounting. With Proxy Forwarders, the
domains could share a secret with the Proxy Forwarder, and in turn,
the Proxy Forwarder could share a secret with each of the devices.
Thus the number of shared secrets will scale with the sum of the
number of devices and domains rather than the product.

The engine of an SNMP Proxy Forwarder does not look inside the PDU of
the message except to determine to which SNMP engine the PDU should
be forwarded or which local SNMP application should process the PDU.
The SNMP Proxy Forwarder does not modify the varbind values; it does
not modify the varbind list except to translate between SNMP
versions; and it does not provide any varbind level access control.

3.3.4. Domain-based access controls in SNMP

Domain-based access controls are required where multiple
administrative domains are involved, such as in the shared use
networks and roaming associations described in [1]. Since the same
device may be accessed by multiple organizations, it is often
necessary to control access to accounting data according to the
user's organization. This ensures that organizations may be given
access to accounting data relating to their users, but not to data
relating to users of other organizations.

In order to apply domain-based access controls, in inter-domain
accounting, it is first necessary to identify the data subset that is
to have its access controlled. Several conceptual abstractions are
used for identifying subsets of data in SNMP. These include engines,
contexts, and views. This section describes how this functionality
may be applied in intra and inter-domain accounting.

3.3.4.1. Engines

The new SNMP architecture, described in [27], added the concept of an
SNMP engine to improve mobility support and to identify which data
source is being referenced. The engine is the portion of an SNMP
entity that constructs messages, provides security functions, and
maps to the transport layer. Traditional agents and traditional
managers each contain an SNMP engine. engineID allows an SNMP engine
to be uniquely identified, independent of the address where it is
attached to the network.

A securityEngineID field in a message identifies the engine which
provides access to the security credentials contained in the message
header. A contextEngineID field in a message identifies the engine
which provides access to the data contained in the PDU.

The SNMPv3 message format explicitly passes both. In SNMPv1 and
SNMPv2c, the data origin is typically assumed to be the
communications endpoint (SNMP agent). SNMPv1 and SNMPv2c messages
contain a community name; the community name and the source address
can be mapped to an engineID via the snmpCommunityTable, described in
[11].

3.3.4.2. Contexts

Contexts are used to identify subsets of objects, within the scope of
an engine, that are tied to instrumentation. A contextName refers to
a particular subset within an engine.

Contexts are commonly tied to hardware components, to logical
entities related to the hardware components, or to logical services.
For example, contextNames might include board5, board7, repeater1,
repeater2, etc.

An SNMP agent populates a read-only dynamic table to tell the manager
what contexts it recognizes. Typically contexts are defined by the
agent rather than the manager since if the manager defined them, the
agent would not know how to tie the contexts to the underlying
instrumentation. It is possible that MIB modules could be defined to
allow a manager to assign contextNames to a logical subset of
instrumentation.

While each context may support instances of multiple MIB modules,
each contextName is limited to one instance of a particular MIB
module. If multiple instances of a MIB module are required per
engine, then unique contextNames must be defined (e.g. repeater1,
repeater2). The default context "" is used for engines which only
support single instances of MIB modules, and it is used for MIB
modules where it only makes sense to have one instance of that MIB
module in an engine and that instance must be easy to locate, such as
the system MIB or the security MIBs.

SNMPv3 messages contain contextNames which are limited to the scope
of the contextEngineID in the message. SNMPv1 and SNMPv2c messages
contain communities which can be mapped to contextNames within the
local engine, or can be mapped to contextNames within other engines
via the snmpCommunityTable, described in [11].

3.3.4.3. Views

   Views are defined in the View-based Access Control Model. A view is
   a mask which is used to determine access to the managed objects in a
   particular context. The view identifies which objects are visible,
   by specifying OIDs of the subtrees included and excluded. There is
   also a mechanism to allow wildcards in the OID specification.

   For example, it is possible to define a view that includes RMON
   tables, and another view that includes only the SNMPv3 security
   related tables. Using these views, it is possible to allow access to
   the RMON view for users Joe and Josephine (the RMON administrators),
   and access to the SNMPv3 security tables for user Adam (the SNMP
   security Administrator).

   Views can be set up with wildcards. For a table that is indexed
   using IP addresses, Joe can be allowed access to all rows in given
   RMON tables (e.g. the RMON hostTable) that are in the subnet
   10.2.x.x, while Josephine is given access to all rows for subnet
   10.200.x.x.

   Views filter at the name level (OIDs), not at the value level, so
   defining views based on the values of non-index data is not
   supported. In this example, were the IP address to have been used
   merely as a data item rather than an index, it would not be possible
   to utilize view-based access control to achieve the desired objective
   (delegation of administrative responsibility according to subnet).

   View-based access control is independent of message version. It can
   be utilized by entities using SNMPv1, SNMPv2c, or SNMPv3 message
   formats.

3.3.5. Inter-domain access-control alternatives

   As the number of network devices within the shared use or roaming
   network grows, the polling model of data collection becomes
   increasingly impractical since most devices will not carry data
   relating to the polling organization. As a result, shared-use
   networks or roaming associations relying on SNMP-based accounting
   have generally collected data for all organizations and then sorted
   the resulting session records for delivery to each organization.
   While functional, this approach will typically result in increased
   processing delay as the number of organizations and data records
   grows.

   This issue can be addressed in SNMP using the event-driven, event-
   driven polling or event-driven batching approaches. Traps and
   Informs permit SNMP-enabled devices to notify domains that have

accounting data awaiting collection. SNMP Applications [39] defines
a standard module for managing notifications.

To use the event-driven approaches, the device must be able to
determine when information is available for a domain. Domain-
specific data can be differentiated at the SNMP agent level through
the use of the domain as an index, and the separation of data into
domain-specific contexts.

3.3.5.1. Domain as index

View-based access control [40] allows multiple fine-grained views of
an SNMP MIB to be assigned to specific groups of users, such that
access rights to the included data elements depend on the identity of
the user making the request.

For example, all users of bigco.com which are allowed access to the
device would be defined in the User-based security MIB module (or
other security model MIB module). For simplicity in administering
access control, the users can be grouped using a vacmGroupName, e.g.
bigco. A view of a subset of the data objects in the MIB can be
defined in the vacmViewFamilyTreeTable. A vacmAccessTable pairs
groups and views. For messages received from users in the bigco
group, access would only be provided to the data permitted to be
viewed by bigco users, as defined in the view family tree. This
requires that each domain accessing the data be given one or more
separate vacmGroupNames, an appropriate ViewTable be defined, and the
vacmAccessTable be configured for each group.

Views filter at the name (OID) level, not at the data (value) level.
When using views to filter by domain it is necessary to use the
domain as an index. Standard view-based access control is not
designed to filter based on the values on non-indexed fields.

For example, a table of session data could be indexed by record
number and domain, allowing a view to be defined that could restrict
access to bigco data to the administrators of the bigco domain.

An advantage of using domains as an index is that this technique can
be used with SNMPv1 and SNMPv2c agents as well as with SNMPv3 agents.
A disadvantage is that the MIB modules must be specifically designed
for this purpose. Since existing MIB modules rarely use the domain
as an index, domain separation cannot be enabled within legacy MIB
modules using this technique.

SNMP does support a RowPointer convention that could be used to
define a new table, indexed by domain, which holds tuples between the
domain and existing rows of data. This would introduce issues of

synchronization between tables.

3.3.5.2. Contexts

ContextNames can be used to differentiate multiple instances of a MIB
module within an engine.

Individual domains, such as bigco.com, could be mapped to logical
contexts, such as a bigco context. The agent would need to create
and recognize new contexts and to know which instrumentation is
associated with the logical context. The agent needs to collect
accounting data by domain and make the data accessible via distinct
contexts, so that access control can be applied to the context to
prevent disclosure of sensitive information to the wrong domain. The
VACM access control views are applied relative to the context, so an
operation can be permitted or denied a user based on the context
which contains the data.

Domain separation is handled by using contextName to differentiate
multiple virtual tables. For example, if accounting data has been
collected on users with the bigco.com and smallco.com domains, then a
separate virtual instance of the accounting session record table
would exist for each domain, and each domain would have a
corresponding contextName. When a get-bulk request is made with a
contextName of bigco, then data from the virtual table in the bigco
context, i.e. corresponding to the bigco.com domain, would be
returned.

There are a number of design approaches to creating new contexts and
associating the contexts with appropriate instrumentation, most
notably a sub-agent approach and a manager-configured MIB approach.

AgentX [51], which standardizes a registration protocol between sub-
agents and master agents to simplify SNMP agent implementation,
allows for the creation and recognition of new contextNames when a
subagent registers to provide support for a particular MIB subtree
range. The sub-agent knows how to support a particular
functionality, e.g. instrumentation exposed via a range of MIB
objects. Based on values detected in the data, such as
source=bigco.com, the sub-agent could determine that a new domain
needed to be tracked and create the appropriate context for the
collection of the data, plus the appropriate access control entries.
The determination could be table-driven, using MIB configuration.

A manager-driven approach could use a MIB module to predefine
contextNames corresponding to the domains of interest, and to
indicate which objects should be collected, how to differentiate to
which domain the data should be applied based on a specified

condition, and what access control rules apply to the context.

Either technique could associate existing MIB modules to domain-
specific contexts, so domain separation can be applied to MIB modules
not specifically designed with domain separation in mind. Legacy
agents would not be designed to do this, so they would need to be
updated to support inter-domain separation and VACM access control.

The use of contextNames for inter-domain separation represents new
territory, so careful consideration would be needed in designing the
MIB modules and applications to provide domain to context and context
to instrumentation mappings, and to ensure that security is not
weakened.

3.3.6. Outstanding issues

There are issues that arise when using SNMP for transfer of bulk
data, including issues of latency, network overhead, and table
retrieval, as discussed in [49].

In accounting applications, management stations often must retrieve
large tables. Latency can be high, even with the get-bulk operation,
because the response must fit into the largest supported packet size,
requiring multiple round-trips. Transfers may be serialized and the
resulting latency will be a combination of multiple round-trip times,
possible timeout and re-transmission delays and processing overhead,
which may result in unacceptable performance. Since data may change
during the course of multiple retrievals, it can be difficult to get
a consistent snapshot.

For bulk transfers, SNMP network overhead can be high due to the lack
of compression, inefficiency of BER encoding, the transmission of
redundant OID prefixes, and the "get-bulk overshoot problem". In
bulk transfer of a table, the OIDs transferred are redundant: all OID
prefixes up to the column number are identical, as are the instance
identifier postfixes of all entries of a single table row. Thus it
may be possible to reduce this redundancy by compressing the OIDs, or
by not transferring an OID with each variable.

The "get-bulk overshoot problem", described in reference [50], occurs
when using the get-bulk PDU. The problem is that the manager
typically does not know the number of rows in the table. As a
result, it must either request too many rows, retrieving unneeded
data, or too few, resulting in the need for multiple get-bulk
requests. Note that the "get-bulk overshoot" problem may be
preventable on the agent side. Reference [41] states that an agent
can terminate the get-bulk because of "local constraints" (see items
1 and 3 on pages 15/16 of [41]). This could be interpreted to mean

that it is possible to stop at the end of a table.

3.3.6.1. Ongoing research

To address issues of latency and efficiency, the Network Management
Research Group (NMRG) was formed within the Internet Research Task
Force (IRTF). Since the NMRG work is research and is not on the
standards track, it should be understood that the NMRG proposals may
never be standardized, or may change substantially during the
standardization process. As a result, these proposals represent
works in progress and are not readily available for use.

The proposals under discussion in the IRTF Network Management
Research Group (NMRG) are described in [46]. These include an SNMP-
over-TCP transport mapping, described in [47]; SNMP payload
compression, described in [48]; and the addition of a "get subtree"
PDU or the subtree retrieval MIB [50].

The SNMP-over-TCP transport mapping may result in substantial latency
reductions in table retrieval. The latency reduction of an SNMP-
over-TCP transport mapping will likely manifest itself primarily in
the polling, event-driven polling and event-driven batching modes.

Payload compression methods include compression of the IP packet, as
described in [5] or compression of the SNMP payload, described in
[48].

Proposed improvements to table retrieval include a subtree retrieval
MIB and the addition of a get-subtree PDU. The subtree retrieval MIB
[50] requires no changes to the SNMP protocol or SNMP protocol
engine, so it can be implemented and deployed more easily than a
change to the protocol. The addition of a get-subtree PDU implies
changes to the protocol and to the engines of all SNMP entities which
would support it. Since it may be possible to address the "get-bulk
overshoot problem" without changes to the SNMP protocol, the
necessity of this modification is controversial.

Reference [49] also discusses file-based storage of SNMP data, and
use of an FTP MIB, to enable storage of SNMP data in non-volatile
storage, and subsequent bulk transfer via FTP. This approach would
require implementation of additional MIB modules as well as FTP, and
requires separate security mechanisms such as IPSEC to provide
authentication, replay, integrity protection and confidentiality for
the data in transit. The file-based transfer approach has an
important benefit - compatibility with non-volatile storage.

Issues of legacy support exist with the NMRG proposals. Devices
which do not implement the new functionality would need to be
accommodated. This is especially problematic for proxy forwarders,
which may need to act as translators between new and legacy entities.
In these situations, the overhead of translation may offset the
benefits of the new technologies.

3.3.6.2. On-going security extension research

In order to simplify key management and enable use of certificate-
based security in SNMPv3, a Kerberos Security Model (KSM) for SNMPv3
has been proposed in [44]. This memo is not on the standards track,
and therefore is not yet readily available for use.

Use of Kerberos with SNMPv3 requires storage of a key on the KDC for
each device and domain, while dynamically generating a session key
for conversations between domains and devices. In terms of stored
keys, the KSM approach scales with the sum of devices and domains; in
terms of dynamic session keys, it scales as the product of domains
and devices.

As Kerberos is extended to allow initial authentication via public
key, as described in [42], and cross-realm authentication, as
described in [43], the KSM inherits these capabilities. As a result,
this approach may have potential to reduce or even eliminate the
shared secret management problem. However, it should also be noted
that certificate-based authentication can strain the limits of UDP
packet sizes supported in SNMP implementations, so that alternate
transport mappings may be required to support this.

An IPSEC-based security model for SNMPv3 has been discussed.
Implementation of such a security model would require the SNMPv3
engine to be able to retrieve the properties of the IPSEC security
association used to protect the SNMPv3 traffic. This would include
the security services invoked, as well as information relating to the
other endpoint, such as the authentication method and presented
identity and certificate. To date such APIs have not been widely
implemented, and in addition, most IPSEC implementations only support
machine certificates, which may not provide the required granularity
of identification. Thus, an IPSEC-based security model for SNMPv3
would probably take several years to come to fruition.

3.3.7. SNMP summary

Given the wealth of existing accounting-related MIB modules, it is
likely that SNMP will remain a popular accounting protocol for the
foreseeable future.

Support for notifications makes it possible to implement the event-
driven, event-driven polling and event-driven batching models. This
makes it possible to notify domains of available data rather than
requiring them to poll for it, which is critical in shared use
networks and roaming.

Given the SNMPv3 security enhancements, it is desirable for SNMP-
based intra-domain accounting implementations to upgrade to SNMPv3.
Such an upgrade is virtually mandatory for inter-domain applications.

In inter-domain accounting, the burden of managing SNMPv3 shared
secrets can be reduced via the localized key capability or via
implementation of a Proxy Forwarder. In the long term, alternative
security models such as the Kerberos Security Model may further
reduce the effort required to manage security and enable streamlined
inter-domain operation.

SNMP-based accounting has limitations in terms of efficiency and
latency that may make it inappropriate for use in situations
requiring low processing delay or low overhead. This includes usage
sensitive billing applications where fraud detection may be required.
These issues can be addressed via proposals under discussion in the
IRTF Network Management Research Group (NMRG). The experimental SNMP
over TCP transport mapping may prove helpful at reducing latency.
Depending on the volume of data, some form of compression may also be
worth considering. However, since these proposals are still in the
research stage, and are not on the standards track, these
capabilities are not readily available, and the specifications could
change considerably before they reach their final form.

SNMP supports separation of accounting data by domain, using either
of two general approaches with the VACM access control model. The
domain as index approach can be used if the desired MIB module
supports domain indexing, or it can implemented using an additional
table. The domain-context approach can be used in agents which
support dynamic logical contexts and a domain-to-context and
context-to-instrumentation mapping mechanism. Either approach can be
supported using SNMPv1, SNMPv2c, or SNMPv3 messages, by utilizing the
snmpCommunitytable [11] to provide a community-to-context mapping.

4. Review of Accounting Data Transfer

In order for session records to be transmitted between accounting
servers, a transfer protocol is required. Transfer protocols in use
today include SMTP, FTP, and HTTP. For a review of accounting
attributes and record formats, see [45].

Reference [49] contains a discussion of alternative encodings for SMI
data types, as well as alternative protocols for transmission of
accounting data. For example, [49] describes how MIME tags and XML
DTDs may be used for encoding of SNMP messages or SMI data types.
This enables data from SNMP MIBs to be transported using any protocol
that can encapsulate MIME or XML, including SMTP and HTTP.

4.1. SMTP

To date, few accounting management systems have been built on SMTP
since the implementation of a store-and-forward message system has
traditionally required access to non-volatile storage which has not
been widely available on network devices. However, SMTP-based
implementations have many desirable characteristics, particularly
with regards to security.

Accounting management systems using SMTP for accounting transfer will
typically support batching so that message processing overhead will
be spread over multiple accounting records. As a result, these
systems result in per-active device state. Since accounting systems
using SMTP as a transfer mechanism have access to substantial non-
volatile storage, they can generate, compress if necessary, and store
accounting records until they are transferred to the collection site.
As a result, accounting systems implemented using SMTP can be highly
efficient and scalable. Using IPSEC, TLS or Kerberos, hop-by-hop
security services such as authentication, integrity protection and
confidentiality can be provided.

As described in [13] and [15], data object security is available for
SMTP, and in addition, the facilities described in [12] make it
possible to request and receive signed receipts, which enables non-
repudiation as described in [12]-[17]. As a result, accounting
systems utilizing SMTP for accounting data transfer are capable of
satisfying the most demanding security requirements. However, such
systems are not typically capable of providing low processing delay,
although this may be addressed by the enhancements described in [20].

4.2. Other protocols

File transfer protocols such as FTP and HTTP have been used for
transfer of accounting data. For example, Reference [9] describes a
means for representing ASN.1-based accounting data for storage on
archival media. Through the use of the Bulk File MIB, accounting
data from an SNMP MIB can be stored in ASN.1, bulk binary or Bulk
ASCII format, and then subsequently retrieved as required using the
FTP Client MIB.

Given access to sufficient non-volatile storage, accounting systems
based on record formats and transfer protocols can avoid loss of data
due to long-duration network partitions, server failures or device
reboots. Since it is possible for the transfer to be driven from the
collection site, the collector can retry transfers until successful,
or with HTTP may even be able to restart partially completed
transfers. As a result, file transfer-based systems can be made
highly reliable, and the batching of accounting records makes
possible efficient transfers and application of required security
services with lessened overhead.

5. Summary

As noted previously in this document, accounting applications vary in
their security and reliability requirements. Some uses such as
capacity planning may only require authentication, integrity and
replay protection, and modest reliability. Other applications such
as inter-domain usage-sensitive billing may require the highest
degree of security and reliability, since in these cases the transfer
of accounting data will lead directly to the transfer of funds.

Since accounting applications do not have uniform security and
reliability requirements, it is not possible to devise a single
accounting protocol and set of security services that will meet all
needs. Rather, the goal of accounting management should be to
provide a set of tools that can be used to construct accounting
systems meeting the requirements of an individual application. As a
result, it is important to analyze a given accounting application to
ensure that the methods chosen meet the security and reliability
requirements of the application.

Based on an analysis of the requirements, it appears that existing
deployed protocols are capable of meeting the requirements for
intra-domain capacity planning and non-usage sensitive billing. In
these applications efficient transfer of bulk data is useful although
not critical. Thus, it is possible to use SNMPv3 to satisfy these
requirements, without the NMRG extensions. These include TCP
transport mapping, sub-tree retrieval, and OID compression.

In inter-domain capacity planning and non-usage sensitive billing,
the security and reliability requirements are greater. As a result,
no existing deployed protocol satisfies the requirements. For
example, existing protocols lack data object security support and
extensions to improve scalability of inter-domain authentication are
needed, such as the Kerberos Security Model (KSM) for SNMPv3.

For usage sensitive billing, as well as cost allocation and auditing
applications, the reliability requirement are greater. Here
transport layer reliability is required to provide robustness against
packet loss, as well as application layer acknowledgments to provide
robustness against accounting server failures. SNMP operations with
the exception of InforRequest provide application layer
acknowledgments, and the TCP transport mapping proposed by NMRG
provides robustness against packet loss. Inter-domain operation can
benefit from data object security (which no existing protocol
provides) as well as inter-domain security model enhancements (such
as the KSM).

Where high-value sessions are involved, such as in roaming, Mobile
IP, or telephony, it may be necessary to put bounds on processing
delay. This implies the need to reduce latency. As a result, the
NMRG extensions are required in time sensitive billing applications,
including TCP transport mapping, get-subtree capabilities and OID
compression. High reliability is also required in this application,
implying the need for application layer as well as transport layer
acknowledgments. SNMPv3 with the NMRG extensions and security
scalability improvements such as the KSM can satisfy the requirements
in intra-domain use.

However, in inter-domain use, additional security precautions such as
data object security and receipt support are required. No existing
protocol can meet these requirements. A summary is given in the
table on the next page.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |                 |                       |
|   Usage       |   Intra-domain  |  Inter-domain         |
|               |                 |                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |                 |                       |
|   Capacity    | SNMPv3 &        |  SNMPv3 &<*           |
|   Planning    | RADIUS #%@      |                       |
|               | TACACS+ @       |                       |
|               |                 |                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |                 |                       |
|   Non-usage   | SNMPv3 &        |  SNMPv3 &<*           |
|   Sensitive   | RADIUS #%@      |                       |
|   Billing     | TACACS+ @       |                       |
|               |                 |                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |                 |                       |
|   Usage       |                 |                       |
|   Sensitive   |                 |                       |
|   Billing,    | SNMPv3 &>$      |  SNMPv3 &<>*$         |
|   Cost        | TACACS+ &$@     |                       |
|   Allocation &|                 |                       |
|   Auditing    |                 |                       |
|               |                 |                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               |                 |                       |
|   Time        |                 |                       |
|   Sensitive   | SNMPv3 &>$      |   No existing         |
|   Billing,    |                 |   protocol            |
|   fraud       |                 |                       |
|   detection,  |                 |                       |
|   roaming     |                 |                       |
|               |                 |                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
Key
# = lacks confidentiality support
* = lacks data object security
% = limited robustness against packet loss
& = lacks application layer acknowledgment (e.g. SNMP InformRequest)
$ = requires non-volatile storage
@ = lacks batching support
< = lacks certificate support (KSM, work in progress)
> = lacks support for large packet sizes (TCP transport mapping,
    experimental)
```

6. Security Considerations

    Security issues are discussed throughout this memo.

7. Acknowledgments

    The authors would like to thank Bert Wijnen (Lucent), Keith
    McCloghrie (Cisco Systems), Jan Melen (Ericsson) and Jarmo Savolainen
    (Ericsson) for useful discussions of this problem space.

8. References

    [1] Aboba, B., Lu J., Alsop J., Ding J. and W. Wang, "Review of
         Roaming Implementations", RFC 2194, September 1997.

    [2] Aboba, B. and G. Zorn, "Criteria for Evaluating Roaming
         Protocols", RFC 2477, January 1999.

    [3] Rigney, C., Rubens, A., Simpson, W. and S. Willens, "Remote
         Authentication Dial In User Service (RADIUS)", RFC 2138, April,
         1997.

    [4] Rigney, C., "RADIUS Accounting", RFC 2139, April 1997.

    [5] Shacham, A., Monsour, R., Pereira, R. and M. Thomas, "IP Payload
         Compression Protocol (IPComp)", RFC 2393, December 1998.

    [6] Bradner, S., "Key words for use in RFCs to Indicate Requirement
         Levels", BCP 14, RFC 2119, March 1997.

    [7] Information Sciences Institute, "Transmission Control Protocol",
         RFC 793, September 1981.

    [8] Aboba, B. and M. Beadles, "The Network Access Identifier",
         RFC 2486, January 1999.

    [9] McCloghrie, K., Heinanen, J., Greene, W. and A. Prasad,
         "Accounting Information for ATM Networks", RFC 2512, February
         1999.

    [10] McCloghrie, K., Heinanen, J., Greene, W., and A. Prasad,
         "Managed Objects for Controlling the Collection and Storage of
         Accounting Information for Connection-Oriented Networks", RFC
         2513, February 1999.

    [11] Frye, R., Levi, D., Routhier, S. and B. Wijnen, "Coexistence
         between Version 1, Version 2, and Version 3 of the Internet-
         standard Management Framework", RFC 2576, March 2000.

   [12] Fajman, R., "An Extensible Message Format for Message
        Disposition Notifications", RFC 2298, March 1998.

   [13] Elkins, M., "MIME Security with Pretty Good Privacy (PGP)", RFC
        2015, October 1996.

   [14] Vaudreuil, G., "The Multipart/Report Content Type for the
        Reporting of Mail System Administrative Messages", RFC 1892,
        January 1996.

   [15] Galvin, J., Murphy, S., Crocker, S. and N. Freed, "Security
        Multiparts for MIME: Multi-part/Signed and
        Multipart/Encrypted", RFC 1847, October 1995.

   [16] Crocker, D., "MIME Encapsulation of EDI Objects", RFC 1767,
        March 1995.

   [17] Borenstein, N. and N. Freed, "MIME (Multipurpose Internet Mail
        Extensions) Part One: Mechanisms for Specifying and Describing
        the Format of Internet Message Bodies", RFC 1521, December 1993.

   [18] Rose, M.T., The Simple Book, Second Edition, Prentice Hall,
        Upper Saddle River, NJ, 1996.

   [19] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction
        to Version 3 of the Internet-standard Network Management
        Framework", RFC 2570, April 1999.

   [20] Klyne, G., "Timely Delivery for Facsimile Using Internet Mail",
        Work in Progress.

   [21] Johnson, H. T., Kaplan, R. S., Relevance Lost: The Rise and Fall
        of Management Accounting, Harvard Business School Press, Boston,
        Massachusetts, 1987.

   [22] Horngren, C. T., Foster, G., Cost Accounting: A Managerial
        Emphasis. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

   [23] Kaplan, R. S., Atkinson, Anthony A., Advanced Management
        Accounting, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

   [24] Cooper, R., Kaplan, R. S., The Design of Cost Management
        Systems. Prentice Hall, Englewood Cliffs, New Jersey, 1991.

   [25] Rigney, C., Willats, S. and P. Calhoun, "RADIUS Extensions", RFC
        2869, June 2000.

   [26] Stewart, R., et al., "Simple Control Transmission Protocol", RFC
        2960, October 2000.

   [27] Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for
        Describing SNMP Management Frameworks", RFC 2571, April 1999.

   [28] Rose, M., and K. McCloghrie, "Structure and Identification of
        Management Information for TCP/IP-based Internets", STD 16, RFC
        1155, May 1990.

   [29] Rose, M. and K. McCloghrie, "Concise MIB Definitions", STD 16,
        RFC 1212, March 1991.

   [30] Rose, M., "A Convention for Defining Traps for use with the
        SNMP", RFC 1215, March 1991.

   [31] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Structure of
        Management Information Version 2 (SMIv2)", STD 58, RFC 2578,
        April 1999.

   [32] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Textual
        Conventions for SMIv2", STD 58, RFC 2579, April 1999.

   [33] McCloghrie, K., Perkins, D. and J. Schoenwaelder, "Conformance
        Statements for SMIv2", STD 58, RFC 2580, April 1999.

   [34] Case, J., Fedor, M., Schoffstall, M. and J. Davin, "Simple
        Network Management Protocol", STD 15, RFC 1157, May 1990.

   [35] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser,
        "Introduction to Community-based SNMPv2", RFC 1901, January
        1996.

   [36] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Transport
        Mappings for Version 2 of the Simple Network Management Protocol
        (SNMPv2)", RFC 1906, January 1996.

   [37] Case, J., Harrington D., Presuhn R. and B. Wijnen, "Message
        Processing and Dispatching for the Simple Network Management
        Protocol (SNMP)", RFC 2572, April 1999.

   [38] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM)
        for version 3 of the Simple Network Management Protocol
        (SNMPv3)", RFC 2574, April 1999.

   [39] Levi, D., Meyer, P. and B. Stewart, "SNMPv3 Applications", RFC
        2573, April 1999.

[40] Wijnen, B., Presuhn, R. and K. McCloghrie, "View-based Access
     Control Model (VACM) for the Simple Network Management Protocol
     (SNMP)", RFC 2575, April 1999.

[41] Case, J., McCloghrie, K., Rose, M. and S. Waldbusser, "Protocol
     Operations for Version 2 of the Simple Network Management
     Protocol (SNMPv2)", RFC 1905, January 1996.

[42] Tung, B., Neuman, C., Hur, M., Medvinsky, A., Medvinsky, S.,
     Wray, J. and J. Trostle, "Public Key Cryptography for Initial
     Authentication in Kerberos", Work in Progress.

[43] Tung, B., Ryutov, T., Neuman, C., Tsudik, G., Sommerfeld, B.,
     Medvinsky, A. and M. Hur, "Public Key Cryptography for Cross-
     Realm Authentication in Kerberos", Work in Progress.

[44] Hornstein, K. and W. Hardaker, "A Kerberos Security Model for
     SNMPv3", Work in Progress.

[45] Brownlee, N. and A. Blount, "Accounting Attributes and Record
     Formats", RFC 2924, September 2000.

[46] Network Management Research Group Web page,
     http://www.ibr.cs.tu-bs.de/projects/nmrg/

[47] Schoenwaelder, J.,"SNMP-over-TCP Transport Mapping", Work in
     Progress.

[48] Schoenwaelder, J., "SNMP Payload Compression", Work in Progress.

[49] Sprenkels, R., Martin-Flatin, J.,"Bulk Transfers of MIB Data",
     Simple Times, http://www.simple-times.org/pub/simple-
     times/issues/7-1.html, March 1999.

[50] Thaler, D., "Get Subtree Retrieval MIB", Work in Progress.

[51] Daniele, M., Wijnen, B., Ellison, M. and D. Francisco, "Agent
     Extensibility (AgentX) Protocol Version 1", RFC 2741, January
     2000.

9. Authors' Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA

Phone: +1 425 936 6605
EMail: bernarda@microsoft.com

Jari Arkko
Oy LM Ericsson Ab
02420 Jorvas
Finland

Phone: +358 40 5079256
EMail: Jari.Arkko@ericsson.com

David Harrington
Cabletron Systems Inc.
P.O.Box 5005
Rochester NH 03867-5005
USA

Phone: +1 603 337 7357
EMail: dbh@cabletron.com

10. Intellectual Property Statement

   The IETF takes no position regarding the validity or scope of any
   intellectual property or other rights that might be claimed to
   pertain to the implementation or use of the technology described in
   this document or the extent to which any license under such rights
   might or might not be available; neither does it represent that it
   has made any effort to identify any such rights. Information on the
   IETF's procedures with respect to rights in standards-track and
   standards-related documentation can be found in BCP-11. Copies of
   claims of rights made available for publication and any assurances of
   licenses to be made available, or the result of an attempt made to
   obtain a general license or permission for the use of such
   proprietary rights by implementors or users of this specification can
   be obtained from the IETF Secretariat.

   The IETF invites any interested party to bring to its attention any
   copyrights, patents or patent applications, or other proprietary
   rights which may cover technology that may be required to practice
   this standard. Please address the information to the IETF Executive
   Director.

11. Full Copyright Statement

Acknowledgement