Network Working Group   Bernard Aboba
INTERNET-DRAFT   Microsoft Corporation
Category: Informational   Jari Arkko
<draft-aboba-acct-02.txt>   Ericsson
2 October 1999

Introduction to Accounting Management

1. Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

To view the list Internet-Draft Shadow Directories, see http://www.ietf.org/shadow.html.

The distribution of this memo is unlimited. It is filed as <draft-aboba-acct-01.txt>, and expires April 1, 2000. Please send comments to the authors.

2. Copyright Notice

3. 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, draft-brownlee-accounting-attributes-0x.txt, reviews the state of the art in accounting attributes and record formats.

4. Table of Contents

5. 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, draft-brownlee-accounting-attributes-0x.txt, reviews the state of the art in accounting attributes and record formats.

5.1. 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. 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.

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 owed. In cost allocation, the cost per unit of resource may need to be determined; in rating, this is typically a given.

Interim accounting
An interim accounting packet provides a snapshot of usage during a user's session. It is typically implemented in order to provide for partial accounting of a user's session in the event of a device reboot or other network problem that prevents the reception of a session summary packet or session record.

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.

5.2. 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. 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.

The diagram below illustrates the accounting management architecture:

```
     +------------+
     |            |
     |   Network  |
     |   Device   |
     |            |
     +------------+
           |
Accounting |
Protocol   |
           |
           V
     +------------+                             +------------+
     |            |                             |            |
     |   Org B    |      Inter-domain           |  Org A     |
     |   Acctg.   |<----------------------------->|  Acctg.    |
     |   Server   |      session records        |  Server    |
     |            |                             |            |
     +------------+                             +------------+
           |                                          |
           |  Intra-domain                            |
Transfer   |  session records                         |
Protocol   |                                          |
           |                                          |
           V                                          V
     +------------+                             +------------+
     |            |                             |            |
     |  Org B     |                             |  Org A     |
     |  Billing   |                             |  Billing   |
     |  Server    |                             |  Server    |
     |            |                             |            |
     +------------+                             +------------+
```

5.3. Accounting management objectives

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

5.3.1. 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 may be tolerable.

In certain cases, it may be desirable to use statistical sampling techniques to reduce data collection requirements while still providing the forecast with the desired statistical accuracy. Such a sampling process may 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.

5.3.2. Billing

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

5.3.2.1. 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 wholesale data loss would also affect other tasks such as trend analysis or auditing, so that this would still be intolerable.

5.3.2.2. 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 meet requirements arising from financial reporting standards, or legal requirements, and therefore an archival accounting approach may be required.

Usage-sensitive systems may also have additional requirements relating to 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 it is desirable to minimize financial risk, a different approach may be required.

Since financial exposure increases with processing delay, it may be necessary to transmit each event individually or to minimize batch size, to require positive acknowledgment before providing service, or even to utilize quality of service techniques to minimize queuing delays.

The degree of financial exposure is application-dependent. For dialup Internet access from a local provider, charges are low and therefore the risk of loss is small. However, in the case of dialup 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. Thus security services such as authentication and integrity protection are frequently used, and confidentiality and non-repudiation may also be desirable.

5.3.3. 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 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 integrity protection and authentication of accounting data, as well as confidentiality and possibly data object security. In order to permit response to security incidents in progress, auditing applications frequently are built to operate with low processing delay.

5.3.4. 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.

5.4. 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 dialup roaming, Internet fax, Internet telephony 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 entitities 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 and integrity protection, it may be desirable to deploy security services such as confidentiality, replay protection, data object integrity, or non-repudiation. In inter-domain accounting each involved party also typically requires a copy of each accounting event for invoice generation and auditing.

5.5. Requirements summary

Usage	Intra-domain	Inter-domain
Capacity Planning	Robustness against moderate packet loss Integrity, authentication, replay protection [confidentiality]	Robustness against high packet loss Integrity, authentication, replay protection confidentiality
Non-Usage Sensitive Billing	Robustness against packet loss not required Integrity, authentication, replay protection [confidentiality]	Robustness against packet loss not required Integrity, authentication, replay protection confidentiality
Usage Sensitive Billing, Cost Allocation & Auditing	Archival accounting Integrity, authentication, replay protection [confidentiality] [Bounds on processing delay]	Archival accounting Integrity, authentication, replay protection confidentiality [non-repudiation] [Bounds on processing delay]

Key
[] = optional

6. 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 models.

6.1. 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 focussed on resilience against packet loss and the differences between UDP and TCP-based transport. However, it should be understood that resilience against packet loss is only one aspect of the program required to meet archival accounting requirements.

As noted in [43], "once the cable is cut you don't need more retransmissions, you need a *lot* nore voltage." Thus, the choice of UDP or TCP 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 such storage, session-oriented 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.

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.

6.1.1. 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+ [26].

While interim accounting can provide resilience against packet loss, server failures, short-duration network failures, or device reboot, its applicability is limited. Interim accounting 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 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 thereby worsening the buffer exhaustion problem.

Given the decreasing cost of non-volatile memory, it may be preferable to store interim accounting data in non-volatile storage. Stored interim events are then replaced by session data when the session completes, and the session data can itself be erased once the data has been transmitted. This approach avoids interim data being transmitted over the wire, except in the case of a device reboot.

6.1.2. 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. 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 transport may 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 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 as well as application-layer keep alive packets 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 modelled. For example, it is almost always preferrable 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 acknowledgements 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 backoff in accounting applications can result in data loss due to buffer exhaustion.

However, there are very persuasive arguments 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 backoff is implemented. This is due to elimination of retransmissions and 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. Furthermore, systems implementing non-volatile storage allow for backlogged accounting data to be placed in permanent storage pending transmission so that buffer exhaustion resulting from congestive backoff need not be a major concern.

Since UDP is not really a transport protocol, UDP-based accounting protocols such as [4] often do not prescribe timeout behavior. Thus one implementations may exhibit widely different behavior. For example, one implementation may drop accounting data after 3 constant duration retries to the same server, while another may implement exponential backoff to a given server, then switch to another server, up to a total timeout interval of 12 hours, while storing the untransmitted data on non-volatile storage. The difference between these approaches is like night and day. For example, the former approach will not satisfy archival accounting requirements while the latter may.

6.1.3. Accounting server failures

In the event of an accounting server failure, it may not be possible for a device to transmit accounting data to its primary accounting server. For protocols using TCP, opening of a connection to the secondary accounting server can occur after a timeout or loss of the primary connection, or it can occur on expiration of a timer. For protocols using UDP, transmission to the secondary server can occur after a number of retries or timer expiration. In either case, it is possible for the primary and secondary accounting servers to receive the same record, so that elimination of duplicates is required.

Since accounting server failures can result in data accumulation on accounting clients, use of non-volatile storage can ensure against the loss of such data due to transmission timeouts or buffer exhaustion.

On-the-wire interim accounting provides only limited benefits in mitigating the effects of accounting server failures.

6.1.4. 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 switchover 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.

6.1.5. 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. Sending interim accounting data over the wire is typically implemented to guard against loss of these high-value sessions. When interim accounting 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.

6.1.6. Fault resilience summary

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

Key
* = limited usefulness without non-volatile storage

6.2. 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 of accounting management systems, we examine each of these resources in turn.

6.2.1. Network bandwidth

Accounting management systems consume network bandwidth in the transferring of 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 [50].

6.2.2. 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.

6.2.3. 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 flash RAM declining rapidly, it is likely that network devices will be capable of incorporating non-volatile storage within the next few years.

As described in [11], non-volatile storage may be used to store interim or session records in a standard ASCII format. 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.

6.2.4. 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.

6.2.5. 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.

6.2.6. Efficiency measures

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

6.3. 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

6.3.1. 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, most of which would not hold any relevant data. There are also issues with latency, since use of a polling interval also implies an average latency of half the polling interval, which may be too high for accounting data that requires low processing delay. Thus the event-driven polling or even the pure event-driven approach will be more appropriate for 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 latencies equal to half the polling interval.

6.3.2. 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 vast 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 pure event-driven approach scales poorly. However, event-driven systems offer the lowest latency since events are processed immediately and there is no possibility of an event requiring low latency being caught behind a batch transfer.

6.3.3. 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.

Through implementation of a scheduling algorithm, event-driven systems with batching can deliver appropriate service to accounting events that require low latency. 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 latency will be caught behind a batch transfer in progress. Thus the maximum latency 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 to the event-driven polling approach. However, it has lower latency than the event-driven polling approach, since delivery of accounting data requires fewer round-trips and events requiring low latency can be accomodated if a scheduling algorithm is employed.

6.3.4. 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 latency 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.

6.3.5. Data collection summary

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

Key
* = addressed by non-volatile storage

7. Review of existing accounting protocols

Accounting systems have been successfully implemented using protocols such as RADIUS, TACACS+, and SNMP. This section describes the characteristic of each of these protocols, as well as transfer protocols such as HTTP, FTP, and SMTP. For a review of accounting attributes and record formats, see [45].

7.1. Accounting protocols

7.1.1. 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. In addition, 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. 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 network failures and device reboots. These deficiencies are magnified when RADIUS accounting is applied in inter-domain accounting as is required in roaming, as noted in [1] and [2]. On the other hand, the event-driven approach of RADIUS accounting is well suited to handling of accounting events which require low processing delay, such as is required for 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 non-repudiation, as noted in [2].

While RADIUS does not suppport compression, IP compression, described in [50], 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).

7.1.2. TACACS+

TACACS+ as defined in [26] 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 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. TACACS+ does not support non-repudiation. While TACACS+ does not suppport compression, IP compression, described in [50], can be employed to provide this.

7.1.3. SNMP

7.1.3.1. SNMP overview

SNMP, described in [27]-[41], has been widely deployed in a wide variety of intra-domain accounting applications, typically using the polling data collection model. Since polling allows data to be collected on multiple accounting events simultaneously, this model results in per-device state. Since the management agent is able to retry requests when a response is not received, such systems are resilient against packet loss or even short-lived network partitions. While implementations without non-volatile storage can only store accounting events up to the limits of their memory, and thus are not robust against device reboots or network failures, when combined with non-volatile storage, they can be made highly reliable. With SNMP v2 it is possible support confirmed notifications, so as to implement an event-driven polling model or even an event-driven batching model. However, we are not aware of any SNMP-based accounting implementations built on these models.

7.1.3.2. NMRG extensions

As discussed in [49], there are a number of efficiency and latency issues that arise when using SNMP for accounting. In such applications it is often necessary for management stations to retrieve large tables. In such situations, the latency can be quite high, even with the get-bulk operation. This is because the response must fit into the largest supported packet size, requiring multiple round-trips. Unless multiple threads are employed, the transfers will be serialized and the resulting latency will be a combination of multiple round-trip times, timeout and re-ransmission delays and processing overhead, resulting in unacceptable performance.

In addition, it is noted that SNMP is inefficient for transfer of accounting data, due to lack of compression, use of BER encoding, transmission of redundant OIDs prefixes, and the "get bulk overshoot" problem. Since the data may change during the course of the retrieval, it can be difficult to get a consistent snapshot.

As a result, this article recommends a number of changes to SNMP, which are now under discussion within the IRTF Network Management Research Group (NMRG), described in [46]. These include an SNMP-over-TCP transport mapping, described in [47]; SNMP payload compression, described in [48]; and addition of a "get subtree" protocol operation. Taken together, we will refer to these changes as the "NMRG extensions."

Reference [49] also discusses file-based storage of SNMP data, as described in [43], and the FTP MIB, described in [44]. Together these MIBs enable storage of SNMP data in non-volatile storage, and subsequent transfer via SNMP. It is noted that this approach requires implementation of additional MIBs as well as FTP, and requires separate security mechanisms such as IPSEC to provide integrity protection and confidentiality for the data in transit. However, the the file-based transfer approach also has an important benefit, which is compatibility with non-volatile storage.

While the NMRG extensions are attractive in the long-term, they represent signicant changes to SNMP and so will take quite a while to standardize and become widely available. While an SNMP over TCP transport mapping is easily implemented, it does require SNMP agents to listen on TCP ports 161 and 162. Addition of a GetSubtree PDU implies changes to every agent that the management station will interact with.

7.1.3.3. SNMP v3

While SNMP v1 and v2 did not incorporate security services, with SNMPv3, it is possible to incorporate view-based access controls, described in [40], as well as user-based security, described in [38]. As a result, SNMP v3-based accounting implementations can provide for hop-by-hop authentication, integrity and replay protection, confidentiality and access-control. Though data-objct security and non-repudiation are not supported in the protocol, it may be possible to support these capabilities through addition of appropriate MIB variables.

SNMP v3 also includes additional functionality useful in inter-domain accounting. Where multiple administrative domains are involved, such as in the shared use networks and roaming associations described in [1], domain-based access controls are required. Since in shared use networks 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. This implies that access rights will depend not only on the view, but on the identity of the user described in the data element. Through use of the contextEngineID, it is possible to support multiple instances of an SNMP MIB, one for each accessing organization. This permits access to be controlled on a per-domain basis, using the view-based access control model described in [40]. For example, when a contextEngineID of bigco.com is used, access would only be provided to data on bigco.com users. Note that this requires that view-based access control be separately set up for each context and that each domain accessing the data be given a separate userName. Note that because use of contextEngineID does not require changes to MIBs, all existing MIBs running on SNMP v3 will inherit domain-handling capabilities. This is very attractive since few existing MIBs use the domain as an index, allowing domain data to be separated out.

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 v3 through use of contextEngineID and the SNMP notification tables, using the event-driven polling approach. This permits SNMP v3-enabled devices to notify domains that have accounting data awaiting collection.

Note that while SNMP v3 has many features enhancing its suitability for shared use or roaming applications, it may be difficult to make use of these enhanced capabilities where there are still legacy devices implementing SNMP v1 or v2. In order to support legacy devices, an SNMP proxy will be required. However, since contextEngineID is only supported in SNMP v3, unless the legacy devices have implemented a MIB that separates out data for individual domains via an index, an SNMP v3 proxy receiving a request for data in a given domain cannot easily translate this into an equivalent SNMP v1 or v2 request.

The same issues of legacy support exist with the NMRG extensions. A proxy receiving a "get subtree" request going to a non-NMRG capabledevices would need to translate the "get subtree" PDU into multiple getbulk requests. Similarly, unless the devices support TCP

transport, deployment of an NMRG-capable proxy will not provide much benefit, since the proxy will need to fall back to UDP-based getnext or getbulk operations. This will result in multiple round-trips and high latency and the risk of inconsistent tables would remain. In addition, existing proxies are built to merely pass on operations so that new proxy code would be needed to support these translations.

Where 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 the SNMP v3 USM allows a manager to have one central key, sharing it with many agents in a localized way while preventing the agents from getting at each other's data. This can assist in cross-domain security if deployed properly.

Another solution is to implement a proxy for the purposes of shared secret reduction. In such a scheme, the domains will share a secret with the proxy, and the proxy will 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.

A Kerberos Security Model (KSM) for SNMP v3 is described in [51]. This approach is an individual submission not yet part of any IETF WG effort. It 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. Thus, in terms of stored keys the KSM approach scales with the sum of devices and domains, whereas 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 [52], and cross-realm authentication, as described in [53] the KSM will inherit these capabilities. As a result, this approach may have potential to reduce or even eliminate the shared secret management problem in the long-term.

7.1.3.4. SNMP v3 Summary

Given the wealth of existing accounting-related MIBs, it is likely that SNMP will remain a popular accounting protocol for the foreseeable future. Given the SNMP v3 enhancements, it is desirable for SNMP-based intra-domain accounting implementations to upgrade to SNMP v3. Such an upgrade is virtually mandatory for inter-domain applications.

With SNMP v3, it is now possible to provide hop-by-hop security services. Through use of contextEngineID, it is possible for SNMP v3 to provide per-domain access controls that are backward compatible with existing MIBs. Through use of the SNMP v3 notify tables, it is possible to implement an event-driven polling model, making it possible to notify domains of available data rather than requiring them to poll for it.

This is critical in shared use or roaming implementations.

In inter-domain accounting, management of SNMP v3 shared secrets can be assisted by the localized key capability or via implementation of a proxy. In the long term, alternative security models such as the Kerberos Security Model may further reduce the effort required to manage security.

As noted in [49], 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. These issues can be addressed via addition of extensions currently under discussion in the IRTF Network Management Research Group (NMRG). Compatibility with non-volatile storage can be achieved via implementation of the MIBs described in [43]-[44].

Note that since few current MIBs support the domain as an index, it can be difficult for an SNMP proxy to simulate the contextEngineID capability on legacy devices. Thus where legacy devices remain it may be necessary to collect data from the devices and sort it by domain, resulting in high processing delay. Elimination of this issue would require upgrading all devices to SNMP v3, as well as implementation of the NMRG extensions. Since SNMP v3 is not widely deployed today and the NMRG extensions are still under development, this "all or nothing" approach is typically not viable in the short to medium term.

7.2. 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.

7.2.1. SMTP-based accounting record transfer

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]-[18]. 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].

7.2.2. Other transfer mechanisms

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, described in [43], 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 described in [44].

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.

8. 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 while 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 the analysis given previously, it appears that existing   protocols are capable of meeting the security requirements for capacity planning and non-usage sensitive billing applications. For usage sensitive billing, as well as cost allocation and auditing applications, new work is required to support file-based storage and transfer of bulk data. Where high-value sessions are involved, such as in roaming, Mobile IP, or telephony, fraud detection support may require low processing delay. Currently, no existing protocol simultaneously meets the requiremens for high security and reliability, as well as low processing delay. A summary is given below:

Usage	Intra-domain	Inter-domain
Capacity Planning or Non-Usage Sensitive Billing	SNMP v1, v2, v3 RADIUS accounting TACACS+ accounting	SNMP v3
Usage Sensitive Billing, cost allocation & auditing	Non-volatile storage SNMP v3 w/NMRG extensions TACACS+ accounting	Non-volatile storage SNMP v3 w/NMRG extensions
Time Sensitive Billing, fraud detection, roaming	Non-volatile storage Low overhead, event driven batching Authenticity and privacy support No existing protocol	Non-volatile storage Low overhead, event driven batching Data object security and receipt support No existing protocol

9. Acknowledgements

The authors would like to thank Bert Wijnen (IBM), Jan Melen (Ericsson) and Jarmo Savolainen (Ericsson) for many useful discussions of this problem space.

10. 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., Willens, S., "Remote Authentication Dial In User Service (RADIUS)", RFC 2138, April, 1997.

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

[5] Gray, J., Reuter, A., Transaction Processing: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, California, 1993.

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

[7] Crocker, D., Overrell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

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

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

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

[11] Aboba, B., Lidyard, D., "The Accounting Data Interchange Format (ADIF)", Internet draft (work in progress), draft-ietf-roamops-actng-05.txt, November 1998.

[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., et al. "Security Multiparts for MIME: Multipart/Signed and Multipart/Encrypted", RFC 1847, October 1995.

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

[17] Shih, C., Jansson, M., Drummond,R., "MIME-based Secure EDI", Internet draft (work in progress), draft-ietf-ediint-as1-09.txt, December 1998.

[18] Shih, C., Jansson, M., Drummond, R., Yarbrough, L. "Requirements for Inter-operable Internet EDI", Internet draft (work in progress), draft-ietf-ediint-req-06.txt, December 1998.

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

[20] Joffe, N., Wing, D., Masinter, L., "SMTP Service Extension for Immediate Delivery", Internet draft (work in progress), draft-ietf-fax-smtp-session-04.txt, August 1998.

[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., Willens, S., Calhoun, P., "RADIUS Extensions", draft-ietf-radius-ext-03.txt, Internet Draft (work in progress), January 1999.

[26] Carrel, D., Grant, L., "The TACACS+ Protocol Version 1.78", Internet draft (work in progress), draft-grant-tacacs-02.txt, January 1997.

[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", RFC 1155, May 1990.

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

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

[31] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1902, January 1996.

[32] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1903, January 1996.

[33] Case, J., McCloghrie, K., Rose, M., and S. Waldbusser, "Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)", RFC 1904, January 1996.

[34] Case, J., Fedor, M., Schoffstall, M., and J. Davin, "Simple Network Management Protocol", 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] Rose, M.T., The Simple Book, Second Edition, Prentice Hall, Upper Saddle River, NJ, 1996.

[43] Stewart, B., "Bulk File MIB", Internet draft (work in progress), draft-stewart-bulk-file-mib-00.txt, November 1998.

[44] Stewart, B., "FTP Client MIB", Internet draft (work in progress), draft-stewart-ftp-client-mib-00.txt, November 1998.

[45] Brownlee, N., "Accounting Attributes and Record Formats", Internet draft (work in progress), draft-brownlee-accounting-attributes-00.txt, May 1999.

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

[47] Schoenwaelder, J.,"SNMP-over-TCP Transport Mapping", Internet draft (work in progress), draft-irtf-nmrg-snmp-tcp-01.txt, June 1999.

[48] Schoenwaelder, J.,"SNMP Payload Compression", Internet draft (work in progress), draft-irtf-nmrg-snmp-compression-00.txt, June 1999.

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

[51] Hornstein, K., Hardaker, W., "A Kerberos Security Model for SNMP v3", Internet draft (work in progress), draft-hornstein-snmpv3-ksm-00.txt, June 1999.

[52] Tung, B., Neuman, C., Hur, M., Medvinsky, A., Medvinsky, S., Wray, J., Trostle, J., "Public Key Cryptography for Initial Authentication in Kerberos", Internet draft (work in progress), draft-ietf-cat-kerberos-pk-init-09.txt, June 1999.

[53] Tung, B., Ryutov, T., Neuman, C., Tsudik, G., Sommerfeld, B., Medvinsky, A., Hur, M., "Public Key Cryptography for Cross-Realm Authentication in Kerberos", Internet draft (work in progress), draft-ietf-cat-kerberos-pk-cross-04.txt, June 1999.

11. Author's Addresses

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Phone: 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

12. Full Copyright Statement

13. Expiration Date

This memo is filed as <draft-aboba-acct-02.txt>, and expires April 1, 2000.