Technische Universität München
Forschungs- und Lehreinheit Informatik / LRZ
Lehrstuhl für Technische Informatik - Rechnernetze

Univ.-Prof. Dr. Heinz-Gerd Hergering

Systementwicklungsprojekt

Aufbau eines Testbeds für das IP Accounting und dessen Integration in die Mobile Agent System Architecture (MASA)

Valentin Saca

Betreuer:

Igor Radisic
Dr. Holger Schmidt

Inhaltsverzeichnis

Inhaltsverzeichnis **3**

Abbildungsverzeichnis **5**

1 Einleitung **7**
- 1.1 Motivation 7
- 1.2 Aufgabenstellung 7
- 1.3 Das MASA System 7
- 1.4 Aufbau der Ausarbeitung 7

2 Traffic Accounting Grundlagen **9**
- 2.1 Ein Internet Accounting Modell 9
- 2.2 Aufbau eines Accounting Tools nach dem OSI Accounting Modell 9
- 2.3 Meter - das zentrale Element für das Internet Accounting 10

3 Internet Accounting mit Hilfe von NeTraMet **12**
- 3.1 Die NeTraMet Architektur 12
- 3.2 Voraussetzungen für den Einsatz von NeTraMet 12
  - 3.2.1 Hardware Ausstattung 12
  - 3.2.2 Software Voraussetzungen 13
- 3.3 Installation von NeTraMet 13
- 3.4 Starten des Meters 14
  - 3.4.1 Startoptionen des NeMaC's 14
  - 3.4.2 Startbeispiele für den NeMaC 14
- 3.5 SNMP in NeTraMet 15
- 3.6 Flows in NeTraMet 15
  - 3.6.1 Funktionsweise 16
  - 3.6.2 Beispiele von Flow Dateien 16
- 3.7 Rules in NeTraMet 17
  - 3.7.1 Welche Attribute können getestet werden ? 18
  - 3.7.2 Beispiel Rule-Dateien 19
- 3.8 Zugriffe und Rechte 19
- 3.9 Beispielszenario 20
- 3.10 Tools um NeTraMet 20

4 Der MASA TrafficDataProvider Agent **21**
- 4.1 Aufbau des TrafficDataProvider Agenten 21
- 4.2 Rule Editor 21
  - 4.2.1 Aktionen 21
- 4.3 Meter Configuration 22
  - 4.3.1 Aktionen 22
- 4.4 Generic Actions 23
  - 4.4.1 Aktionen 23
- 4.5 Bemerkungen zur TrafficDataProvider API 24

5 Zusammenfassung und Ausblick **25**

Anhang A : Optionen für NeTraMet und NeMaC **26**

Literaturverzeichnis **27**

Abbildungsverzeichnis

1 Domänenmodell 9
2 Accounting Modell Testbed 10
3 NeTraMet Architektur 12
4 TrafficDataProvider Testbed 20
5 TrafficDataProvider Architektur 21
6 Rule Editor 22
7 Meter Configuration 23
8 Generic Actions 24

1 Einleitung

1.1 Motivation

Anders als bei der Telefonrechnung, die eine genaue Aufschlüsselung und damit eine Zuweisung der Kosten auf verschiedene Kostenstellen zuläßt, ließ sich die Netzkommunikation bislang nur schwer abrechnen – meist pauschal über eine Kostenstelle. Gerade vor dem Hintergrund des rasant wachsenden Marktes von Nutzern IP basierender Dienste stehen heute zugleich Carrier und ISPs unter einem enormen Wettbewerbsdruck. Das Datenaufkommen im Internet und den Intranets verdoppelt sich nahezu halbjährlich, nicht zuletzt durch das Angebot immer neuer Dienste wie z.B. Videoconferencing oder Voice over IP.

Volumen- oder zeitbezogene Pauschaltarife, wie sie heute von den ISPs und Carriern angeboten werden, sind dabei kaum länger geeignet, die betriebswirtschaftlich geforderte Zuordnung von in Anspruch genommenen Netzdiensten an einzelne Nutzer bzw. Abteilungen zu leisten. Auch eine differenzierte Gewichtung einzelner Dienste ist notwendig. Diese erlaubt es z.B., zeitkritische und damit technisch aufwendige Dienste wie VoIP anders zu tariffieren als zeitunsensible aber volumenintensive Dienste wie FTP. Dieses Problem trifft gleichermaßen in großen Intranets zu. Die von der Zentrale zur Verfügung gestellte Ressource „Netz“ mit ihren Kosten sollte möglichst entsprechend der Nutzung auf die Kostenstellen verteilt werden können.

Hinter der Beobachtung des Datenverkehrs verbergen sich weitere Ziele der Internet-Anbieter. Man versucht das Verhalten der Nutzer zu verstehen bzw. zu beeinflussen. Der Nutzer kann sich dadurch selber besser analysieren und versuchen, seine Performance zu verbessern und seine Kosten zu senken. Die Messung wird mit Hilfe von sog. Accounting Anwendungen durchgeführt. Aus der Sicht des Providers dienen diese Messungen um zu sehen, ob sich der Benutzer innerhalb der vereinbarten zu transportierenden Datenmenge befindet. Solche Reports sind nicht ausreichend für die Einhaltung der Vereinbarungen, sie sind aber ein Hinweis, daß zusätzliche Maßnahmen zur Kontrolle notwendig sind. Die Accounting Anwendungen werden auch dafür eingesetzt, die Rentabilität eines Rechnernetzes zu messen. Man kann vor ineffizienter Netz Konfiguration/Nutzung mahnen, kann dadurch z.B. die Anwender ermuntern, große Datenmengen nicht während Spitzennutzungszeiten durchzuführen, sondern wenn die Anzahl der Netz-Nutzer klein ist.

Bisher fehlten für IP-Netze vielfach die notwendigen Tools, um auch in diesen Netzen den Datenverkehr so zu erfassen, daß flexible Kostenzuordnungen, oder anders gesagt Abrechnungsmodelle, entwickelt werden konnten. Ein solches Werkzeug, daß es ermöglicht, den Netzverkehr zu sammeln und auszuwerten, ist NeTraMet, auf das in dieser Arbeit näher eingegangen wird.

1.2 Aufgabenstellung

Ziel dieses Systementwicklungsprojektes ist es, ein Test-Bed aufzubauen, um die Eignung NeTraMets als Traffic Accounting Tool zu testen. Zusätzlich soll für das MASA-System ein Traffic Data Provider Accounting Agent aus dem bereits bestehenden DataProvider-Agent abgeleitet werden, der es ermöglicht, auf einem beliebigen Rechner im Agentensystem Traffic-Messungen, an Hand selbstdefinierter Accounting-Regeln, die über ein Formular erstellt werden können, durchzuführen. Der Report des gesammelten Traffics soll in einem bestimmten Format erstellt werden und zwar im IPDR-Format, als XML-Dokument.

1.3 Das MASA System

MASA ist eine plattformunabhängige Laufzeit- und Kommunikationsarchitektur für mobile Agenten. Auf einem zu managenden Endsystem wird ein Agentensystem (MASA) ausgeführt, das die Laufzeitumgebung für alle Agenten auf dem Endsystem darstellt. Die Kommunikation der Agenten untereinander geschieht über die Common Object Request Broker Architecture (CORBA). Zudem ist ein Webserver Teil eines jeden Agentensystems, welcher sowohl für die Kommunikation zur Steuerung der Agenten, als auch für das Agentensystem selbst eingesetzt wird. Genauere Informationen über MASA sind unter [KEMP_98] anzufinden.

1.4 Aufbau der Ausarbeitung

Nach einer kurzen Einführung in die Aufgabenstelltung und einer kurzen Beschreibung des MASA Systems im ersten Kapitel, werden im Abschnitt 2 die Traffic Accounting Grundlagen besprochen. Als erstes wird auf das Internet Accounting Modell eingegangen (Abs. 2.1), danach der Aufbau eines Accounting Tools vorgestellt (Abs. 2.2), wobei dann im darauffolgenden Abschnitt (Abs. 2.3) explizit auf die zentrale Komponente einer solchen Applikation - dem Meter - eingegangen wird. Der dritte Abschnitt befasst sich mit dem im Systementwicklungsprojekt verwendeten Tool - **NeTraMet**. Es wird auf die NeTraMet Architektur (Abs. 3.1), auf die Hardware- und Softwarevoraussetzungen (Abs. 3.2) und auf die Installation von NeTraMet (Abs. 3.3) eingegangen. Im Abschnitt 3.4 wird das Starten eines Meters erläutert, auf die Rolle von SNMP in NeTraMet wird im Abschnitt 3.5 eingegangen. In den verbleibenden Abschnitten des dritten Kapitels werden der Aufbau und die Verwendung der Rule Files und die für das Benutzen von NeTraMet notwendigen Zugriffsrechte (Abs. 3.8) behandelt. Ein Beispielszenario wird im Abschnitt 3.9 präsentiert und im darauffolgenden Abschnitt werden weitere Tools, die zusammen mit NeTraMet verwendet werden können, vorgestellt. Im vierten Teil der Ausarbeitung wird der Aufbau des neu realisierten Traffic Data Provider Agenten geschildert. Es wird in einzelnen Abschnitten auf die einzelnen Konfigurationsbereiche Generic Actions (Abs. 4.2), Rule Editor (Abs. 4.3) und Meter Configurator (Abs. 4.4) eingegangen. Abschliessend wird in der Zusammenfassung im fünften Teil der Ausarbeitung auf mögliche Erweiterungen des Traffic Data Provider Agenten und alternative Traffic Measurement Tools verwiesen.

2. Traffic Accounting Grundlagen

2.1 Ein Internet Accounting Modell

Für gewöhnlich hat der Netz-Administrator das größte Interesse am Internet Accounting, sei es weil er den Verkehr genauer überwachen möchte oder weil der Netzverkehr abgerechnet werden soll. Er verwaltet einen Teil des Internets und zwar seine administrative Domäne, für die er zuständig ist. Diese Domäne hat gut definierte Grenzen. Der Administrator ist interessiert an dem Traffic (Verkehr) innerhalb seiner Grenzen und an den, der seine Grenzen überschreitet. Innerhalb seines Bereiches wird er entweder von Endsystem zu Endsystem abrechnen wollen, oder eine pauschale Abrechnung von Abteilung zu Abteilung aufstellen. Für gewöhnlich wird der Administrator nicht an der Abrechnung von Endsystemen außerhalb seines Bereichs interessiert sein, sein hauptsächliches Interesse gilt der Abrechnung des Verkehrs zu den Domänen, die direkt an seiner angeschlossen sind. [RFC 1272]

Sei folgende Abbildung gegeben:

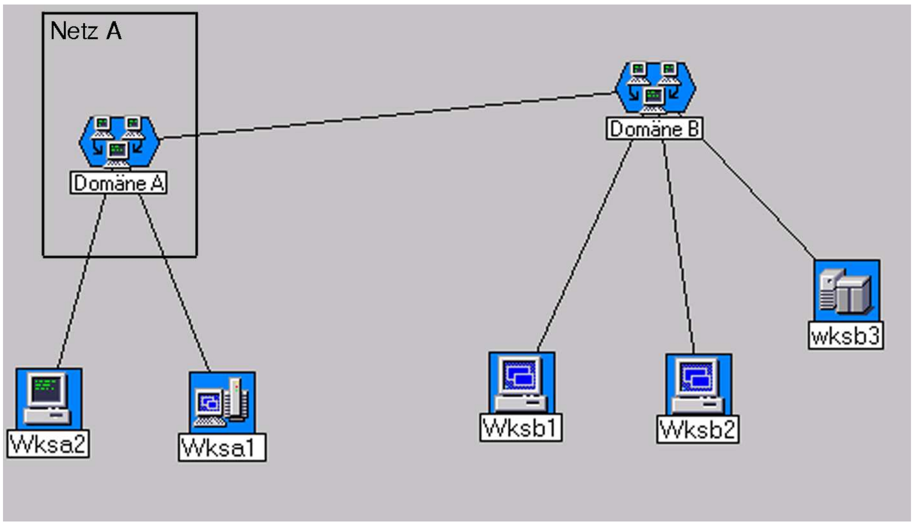

Abbildung 1: Domänenmodell nach [RFC 1272] beschrieben

Im RFC 1272 „Internet Accounting Background" wird die Abrechnung wie folgt erstellt: Sei A die Domäne unseres Administrators. Dieser wird jeder angrenzenden Domäne (wie z.B. Domäne B in Abb. 1) eine Abrechnung zukommen lassen für die Nutzung seiner Ressourcen und diese werden ihm ebenfalls eine zukommen lassen, für die Nutzung ihrer Ressourcen. Wenn er eine Rechnung vom Netz B erhält und diese an die Systeme seiner Domäne weiterleiten will, die diese Ressourcen tatsächlich benutzt haben, so ist es seine Pflicht die entsprechenden Daten zu sammeln, um zu sehen, wie sie diese Ressourcen des Netzes B in Anspruch genommen haben. Sollte die Domäne B auch die Ressourcen von A verwendet haben, so wird der Administrator an B den Teil der Rechnung weiterleiten, den B generiert hat.

Dieses rekursive Abrechnungsparadigma, so wie es im [RFC 1272] beschrieben wird, kann auch innerhalb von Domänen verwendet werden, die logisch in administrativen Subdomänen aufgeteilt sind.

Die wichtigste Anforderung des Internet Accounting Modells besagt, daß Abrechnungen sowohl für angrenzende Systeme, als auch für Endsysteme aufstellbar sein müssen. Abrechnungsinformationen für angrenzende Systeme können nicht aus den Abrechnungsinformationen der Endsysteme abgeleitet werden, da der Verkehr zwischen Endsystemen über verschiedene angrenzende Systeme laufen kann. Die Anforderung, auch Abrechnungen für angrenzende Zwischensysteme erstellen zu können, setzt voraus, daß das Internet Accounting Informationen benötigt, die nicht im IP-Header stehen. Benötigt man benutzerspezifische Informationen, so muß auf die darunter liegenden Protokolle (link layer) zugegriffen werden, für anwendungsspezifische Informationen, sind die oberen Protokolle (ab Level 4) relevant.

2.2 Aufbau eines Accounting Tools nach dem OSI Accounting Modell

Die meisten, Traffic Measurement Tools, wie auch NeTraMet, wurden nach dem OSI Accounting Modell aufgebaut und sind durch drei Entitäten definiert:

1) METER — führt nur die Messung durch

2) COLLECTOR — sammelt die Daten, bearbeitet/formatiert/speichert sie und ist für die Integrität und Sicherheit der Daten beim Speichern und Transportieren zuständig

3) APPLICATION — ist eine weiterverarbeitende Anwendung, die auf den gesammelten und bereits gespeicherten Daten operiert, wie z.B. ein Billing System

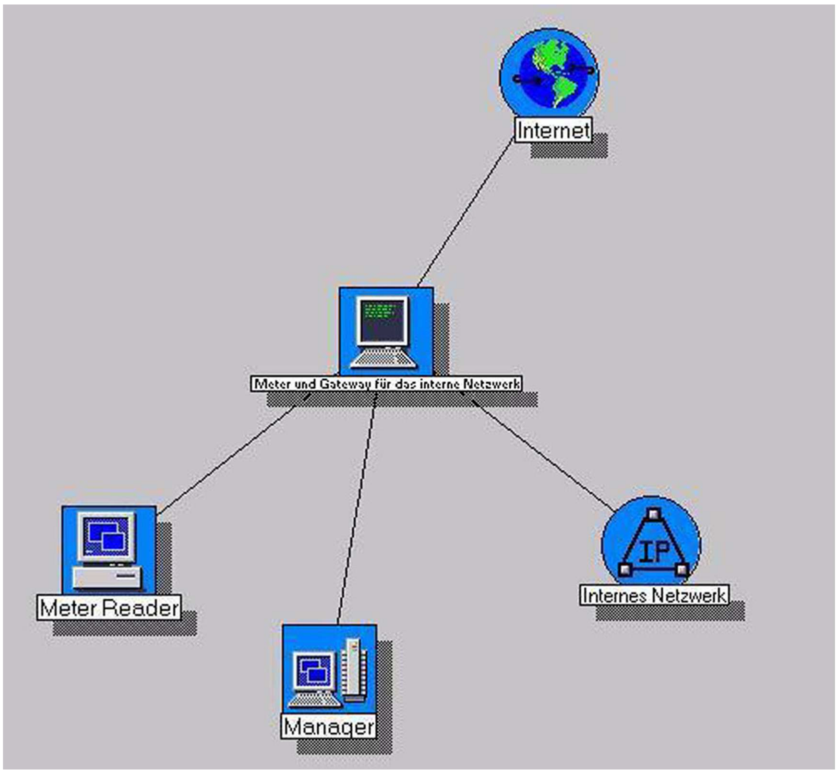

Abbildung 2: Accounting Modell Testbed

2.3 Meter - das zentrale Element für das Internet Accounting

Ein **Meter** ist ein Prozess, der ein Fluß von Paketen auf einem Kommunikationsmedium oder zwischen einem Paar von Medien untersucht. Ein solcher Fluss von Paketen nennt man einen **Flow**.

Die Meter-Aufzeichnungen beinhalten die Anzahl der Pakete, die zu einem Flow zwischen zwei Entitäten gehören. Die Zuordnung der Pakete zu Flows geschieht durch die Beachtung einer Reihe von Regeln, den sogenannten **Rules**. Meter können in allen der folgenden drei Umgebungen implementiert werden: dedizierte Monitore, in Routern oder in normalen Systemen.

Eines der wichtigsten Überlegungen beim Internet Accounting ist die Entscheidung, wo man den Meter am besten aufstellt. Ein wichtiges Kriterium dabei sind die Kosten. Der Accounting Overhead muß minimiert werden, es soll nur der Traffic über den Meter laufen, der auch abgerechnet wird, wobei nur wenige unrelevante Verbindungen darüber entstehen sollten. Bei dieser Entscheidung spielen Genauigkeit und Zuverlässigkeit eine wichtige Rolle. Je nach Zweck des Accountings sind entweder ganz genaue Abrechnungen gefragt, was allerdings mehr Overhead nach sich zieht, oder man versucht mit Hilfe der Statistik den Verkehr abzuschätzen. Will man nur das Netzverhalten verstehen, um entsprechend die Leistung und das Design zu verbessern, so kann man mit statistischen Werten arbeiten, sollen aber genaue Rechnungen (ähnlich den Telefonrechnungen) gestellt werden, so muß die Messung ganz genau erfolgen.

Meter sollten da aufgestellt werden, wo sie eine optimale Datenerfassung erreichen können. Router sind als Ort oft am besten für die Plazierung eines Meters geeignet . Diese Entscheidung hängt aber auch von der darunter liegenden Transporttechnik und von den Überwachungskriterien ab. Läuft die Verbindung über mehrere Router, was in der Praxis oft der Fall ist, so ist der Router nicht mehr der ideale Platz. Ist das Netz ein Broadcast Netz (z.B. bus-basiert), nicht erweitert (z.B. über Bridges und Router), dann ist die Plazierung nicht mehr so entscheidend. Im Normalfall werden dann die Messungen lokal auf den entsprechenden Hosts durchgeführt .

Meistens sind aber die Netze größer (erweitert), dann werden Meter auf oder neben Routern plaziert, um alle Pakete zu zählen. Der gesamte Verkehr fliest zwischen Hosts und Routern. Die Gründe einen Router als Messpunkt einzusetzen wären:

a. Minimierung der Kosten und des Overheads

b. Traffic Kontrolle

c. Abrechnung für Zwischensysteme (s.o. bei 1.2)

Es gibt folgende Meter Typen:

Netz Monitore	- messen nur innerhalb eines Netzes
Line Monitore	- messen nur auf einer Leitung
Router-integrale Meter	- messen alles was über den Router läuft
Router Spiders	-bestehen aus einer Menge von Linien Monitoren, die einen Router umgeben, den Verkehr auf all seinen Ports messen und die Ergebnisse koordinieren

3. Internet Accounting mit Hilfe von NeTraMet

NeTraMet ist eine freie Software, die unter der GNU General Public License erhältlich ist [NTM_SWD]. Dahinter verbirgt sich ein Meter für das Messen des Netzverkehrs. Das Tool ist die erste Implementierung der Realtime Traffic Flow Measurement (RTFM) Working Group Measurement Architecture [RFC 2722].

3.1 Die NeTraMet Architektur

Ein NeTraMet System besteht aus folgenden Komponenten :

Meter	ein kleiner Host, der an einem Netzwerk Segment angeschlossen ist und den Verkehr mißt, der auf diesem Segment läuft (NeTraMet als Bestandteil des NeTraMet -Systems)		
Collector	bestehend aus:		
	Meter Reader	- bezieht Informationen von Metern (NeMaC)	
	Manager	- teilt Metern mit, welche Flows sie messen sollen und liest Daten von mehreren Meter-Readern aus (NeMaC)	

Der Meter Reader und der Manager entsprechen auf einem NeTraMet-System dem **NeMaC** Tool bzw. dem **Ne**TraMet **Ma**nager and **C**ollector.

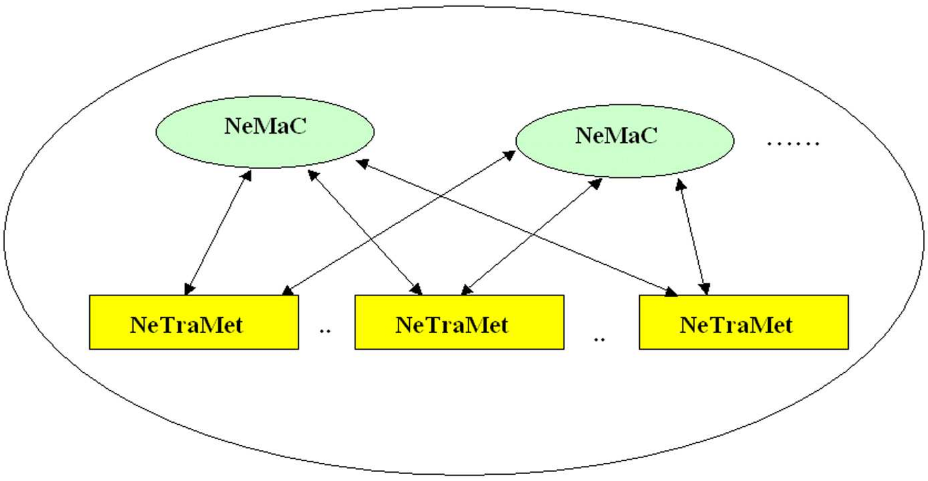

Abbildung 3: Ein Meter (NeTraMet) kann mehrere Manager und Collector (NeMaC) bedienen

3.2 Voraussetzungen für den Einsatz von NeTraMet

3.2.1 Hardware Ausstattung

Je mehr Traffic es zu messen gilt, umso höher sind die Anforderungen an die Hardware, die die Messungen durchführen soll. Je nach Taktfrequenz des Prozessors auf dem der Meter läuft, können mehr Pakete in der Sekunde (PPS) erfasst werden:

25 MHz - 386SX	1250-2250 PPS (packets per second) möglich
40 MHz - 486	max. 3000 PPS möglich
60 MHz - 586 PCI-Bus	max. 6000 PPS möglich

Da die Hardware immer günstiger wird, der Netzverkehr aber auch immer größer, würde der Einsatz eines Pentiums Prozessors (> 166 MHz) gerechtfertigt sein. Im Testbed wurde ein Pentium III Rechner mit einer 450 MHz Prozessor Taktung verwendet.

3.2.2 Software Voraussetzungen

Man kann NeTraMet auf unterschiedlichen Betriebssystemen laufen lassen: Solaris,SunOS, Irix und Linux werden unterstützt, aber auch Windows-Systeme. NeMaC läuft auf Unix-Systemen und ist im Einsatz auf Solaris-, SunOS -, Linux-, DEC Unix-, AIX- und HPUX-Systemen.

Wenn man NeTraMet auf einem Linux-System installiert, benötigt man noch zusätzlich die *libpcap* Bibliothek. Diese Bibliothek wird von Paket-Sniffer-Programmen verwendet, sie stellt eine Schnittstelle zur Verfügung, um Pakete von einem Netzanschluss abzufangen, zu analysieren und die in den Paket Header enthaltenen Informationen extrahieren.

Die im Testbed verwendete Linux Kernel-Version war 2.4 , aber auch auf einem 2.14 Kernel läuft NeTraMet, auf eigene Tests beruhend, einwandfrei.

3.3 Installation von NeTraMet

Um NeTraMet auf einen Linux Rechner zu installieren, benötigt man das *NeTraMet43.tar.gz* Archiv [NTM_SWD]. Das Archiv enthält die NeTraMet Dokumentation mit Beispiel Regeldateien, die Meter Service MIB und dem Source Code für CMU SNMP, zusätzlich die Tools NeTraMet, *nifty*, *srl* und NeMaC enthalten. Zusätzlich sollte man sich aber auch noch das NeTraMet und NeMaC Referenzhandbuch, das im PDF-Format zur Verfügung steht, herunterladen [NTM_SWD].

Um mit NeTraMet einen Zähler (Meter) aufzubauen, benötigt man aus dem Paket erstmal nur NeTraMet selber. Um Regeldateien für Messungen aufzustellen, gibt es den SRL-Compiler, mit dem man in SRL (SimpleRulesetLanguage) geschriebene Regeldateien in sogenannte rule-Dateien übersetzen kann. *nifty* ist ein zusätzliches Tool, das im Paket enthalten ist und dazu dient, um den Netzverkehr zu analysieren. Es bietet die Möglichkeit, den Traffic grafisch mit sogenanntem Plots darzustellen.

GNU autoconfig wird verwendet, um Makefiles für alle NeTraMet Programme zu erstellen, was es sehr leicht macht, NeTraMet Programme auf einem Unix System zu installieren.

Beim Kompilieren von NeTraMet kann man verschiedene Optionen mitgeben:

CLNS - erlaubt es NeTraMet auch CLNS Pakete zu messen/zählen

FULL_IPX - erlaubt es NeTraMet (10-Byte) IPX peer Adressen zu verwenden

V6 - ermöglicht es NeTraMet IPv6 Pakete zu erkennen und zu messen/zählen

Diese Optionen werden für alle NeTraMet Programme gesetzt, indem man die *configure* Datei editiert, bevor man die Programme kompiliert, oder indem man die *configure.in* Datei editiert und danach *autoconfig* verwendet, um eine neue *configure* Datei daraus zu bilden.

Im Testbed wurden auf dem Referenzrechner *projekt01* folgende Aufrufe getätigt:

1. LIBS Umgebungsvariable um das Library-Verzeichnis erweitern, in das auch *libcap* enthalten ist - je nach verwendeter Shell : *setenv LIBS -L/usr/local/lib*

2. Verzeichnis für NeTraMet erstellen: *mkdir /usr/local/NeTraMet43;cd /usr/local/NeTraMet43*

3. Kopieren des Archivs ins Verzeichnis: *cp NeTraMet43.tar.gz /usr/local/NeTraMet43/*

4. Archiv entpacken: *gunzip -d NeTraMet43.tar.gz*

5. *./configure* Skript aufrufen und NeTraMet's Makefiles zu erstellen

6. Make NetraMet: *make*

7. Programme installieren: *make install*

3.4 Starten des Meters

Sobald das Programm kompiliert ist, kann man es starten. NeTraMet öffnet einen UDP Port für SNMP, die Rechte dafür müssen vorhanden sein. Für den Aufruf von NeTraMet benötigt man root-Rechte. Jeden Meter, den man startet, muß eine gesetzte write-Community besitzen. Danach kann man NeTraMet starten, z.B.:

$> NeTraMet -w write

startet NeTraMet mit der write-Community „write". Die Option *-w* setzt die write-Community auf den übergebenen Wert, in diesem Beispiel *write*. NeMaC kann von jedem User aufgerufen werden und benötigt nur Zugriff zur *mib.txt* Datei im */usr/local/NeTraMet43/mib* Verzeichnis (die Beschreibungsdatei der von NeTraMet implementierten Meter MIB). Wenn man nicht immer den kompletten Pfad zur *mib.txt* eingeben will, kann man sich eine MIBTXT Umgebungsvariable definieren die den entsprechenden Pfad enthält.

3.4.1 Startoptionen des NeMaC's

Wenn man sich eigene Regel-Dateien für den/die Meter spezifiziert, dann kann man auch NeMaC starten. Man kann mit dem NeMaC gleichzeitig auf mehreren Metern zugreifen, hier empfiehlt sich dann NeMaC mit einer zusätzlichen Konfigurationsdatei zu starten, in der man die Meter einträgt, auf die er zugreifen soll.

Die für den NeMaC am häufigsten eingesetzten Aufrufoptionen sind:

-L	leitet die log-Datei für NeMaC ein, hier stehen nur die „Rückmeldungen" von NeMaC drin
-b	gibt den Namen der MIB-Datei an
-r	weist auf die Regel-Datei hin
-cSSS	gibt die Sammelzeit in Sekunden an. Wenn SSS=0 , dann wird NeMaC die Regel-Datei auf den Meter kopieren, aber keine Daten sammeln
-F	weist auf die Flow-Datei hin, in der NeMaC die gesammelten Daten reinschreibt

3.4.2 Startbeispiele für den NeMaC

Ein Aufruf von NeMaC mit einer Konfigurationsdatei (eingeleitet über die Option *-f* und hier namens *nm-config*) würde wie folgt lauten:

$> NeMaC -f nm-config

nm-config ist die Konfigurationsdatei und könnte z.B. folgendes enthalten:

```
-c900 -p -r rules.mynet default
      meter1 write-1
      meter2 write-2
-c300 meter3 write-3
```

Somit werden 3 Meter gestartet, alle mit der *rules.mynet* Regeldatei. *meter1*, *meter2* werden jede 15. Minute ausgelesen und *meter3* jede 5. Minute. (siehe Option *-c*)

Ein Beispielaufruf für einen Testlauf im Testbed lautete:

$> NeMaC -L heute.log -b mib.txt -r ../examples/rules.test -c120 -F flows.log linux-router write

Der Parameter *linux-router* bezieht sich auf den Meter, auf den NeMaC die Regeln lädt und die Daten einsammelt, *write* spezifiziert die Community.

NeMaC und NeTraMet kennen noch weitere Optionen. Diese werden im Anhang A (Seite 24) der Ausarbeitung erläutert.

3.5 SNMP in NeTraMet

Ein NeTraMet Meter kann auch über SNMP von einer Managementstation abgefragt werden. NeTraMet bringt einen eigenen SNMP-Agenten mit, der standardmäßig den UDP Port 161 öffnet. NeMaC arbeitet standardmäßig auf Port 1024. Beide Ports können durch bestimmte Aufrufoptionen auf andere Werte gesetzt werden.

In den ersten Versionen wurde für die Kommunikation zwischen NeTraMet und NeMaC CMU SNMP verwendet. 1995 erschien die erste Version von NeTraMet die über SNMPv2C [CHRW_96] mit NeMaC kommunizierte, aber nur mit einer Community-basierten Sicherheit arbeitet. Zur gleichen Zeit wurde NeTraMet auf OC3MON portiert, einem ATM Traffic Monitoring System [RFC 1252], das von NLANR (National Laboratory for Applied Network Research) und MCI (einem amerikanischen Telekommunikationsunternehmen) entwickelt worden war. Eine wichtige Eigenschaft eines Accounting Meters (Zählers) ist die Möglichkeit, Messdaten in einer effizienten Art und Weise zu sammeln. SNMP kann für diese Zwecke ineffizient sein, da jeder bezogene Wert von seinem Objekt-Identifikator begleitet wird. Um einen Long Wert auszulesen (4 Bytes) können noch 12 oder mehr Bytes für Objektidentifikatoren hinzukommen.

Die Versionen 2 und 3 von NeTraMet lösten dieses Problem, indem sie SNMP opaque objects benutzten, um mehrere Werte in einer Einheit an NeMaC weiterzureichen. Die MIB definiert ein Objekt namens Column Blob um dies zu ermöglichen. Ein Column Blob ist ein 3-dimensionelles SNMP Objekt , mit den Dimensionen Spaltennummer, LastTime (wann der Flow zum letzten mal als aktiv galt) und FlowIndex. Die maximale Column Blob Größe wurde so gewählt, daß sie in einem 500 Byte SNMP Paket passt und somit 50 bis 60 Attribut-Werte auf einmal transportieren kann.

Mit Version 4.1 hat die neue Meter MIB [RFC 2064] den Begriff „flow data package” eingeführt. Dies ist eine Liste von Attribut-Werten eines Flows , die von einem Meter durch einen einzigen SNMP GET request ausgelesen werden können. Mit den SNMPv2 GETBULK request kann eine Reihe von Paketen durch ein einziges 1500 Byte großes SNMP Paket ausgelesen werden. Diese Version bietet eine bessere Möglichkeit Daten zu beziehen, da mit dem fast gleichen Overhead wie bei den Column Blobs 1500 Byte Pakete bezogen werden können und somit die Anzahl der benötigten Pakete reduziert wird.

3.6 Flows in NeTraMet

Die Hauptaufgabe eines NeTraMet-Systems ist es, Flows zwischen zwei Host-Adressen zu messen. Host Adressen (adjacent,peer und transport) und ihre Masken werden in Stukturen gespeichert, die Schlüssel genannt werden. Ein Flow ist eine etwas größere Struktur, die zwei Schlüssel enthält, einen für die Quell- und einen für die Zieladresse. Allgemeine Attribute werden innerhalb von Variablen in einem Flow gespeichert.

Ein Flow ist ein Strom von Paketen, die zwischen zwei Hosts ausgetauscht werden, die wir als Flow-Quelle und Flow-Ziel bezeichnen. Flows sind bidirektional, so daß Pakete und Bytes in der „hin” und „von” Richtung gezählt werden können. Die „Identität” eines Flows wird durch die Adress-Attribute der beiden Hosts bestimmt und diese können dreierlei sein:

adjacent	(link layer)
peer	(network layer)
transport	(transport layer)

Bei Ethernet wäre die adjacent Adresse die Ethernet MAC Adresse, die peer Adresse kann eine IP Adresse, eine DECnet phase IV Adresse, eine Novell Netz Nummer, eine EtherTalk Adresse oder eine CLNS NSAP sein, das sind die 5 Protokolle, die momentan von NeTraMet unterstützt werden. Eine Transport Adresse ist bei TCP der Protokoll-Typ und die Quelle- und Ziel-Portnummer. Ähnliche Details sind in der Tranport-Adresse auch für andere Protokolle enthalten.

NeTraMet kann auch Pakete und Bytes für Protokolle zählen, die es nicht versteht. All diese Pakete werden in einem einzigen Flow zusammengefasst, der den peer Typ „other” hat.

3.6.1 Funktionsweise

Sobald ein Flow beobachtet wird, wird eine „count" Aktion angestossen. Der Meter (Zähler) stellt Speicherplatz zur Verfügung, vergibt einen FlowIndex und trägt es in eine „count" Tabelle ein, die als eine große Hash-Tabelle implementiert ist. Der Meter (Zähler) verwaltet auch eine Tabelle von Pointern (Zeigern) auf die „count" Tabelle, mit deren Hilfe auf die Flow Daten zugegriffen werden kann.

Sobald ein Flow erzeugt wurde, kann er unendlich lange bestehen. Mit der Zeit wird aber der Meter keinen freien Speicher für neue Flows haben. Um dieses Problem zu umgehen, benutzt NeTraMet einen inkrementellen Garbage Collector. In gleichen Abständen, die in der GarbageCollectionInterval Variablen festgehalten sind wird der GarbageCollector (GC) aufgerufen. Um zu entscheiden, welche Flows verworfen werden können, schaut der GC wie lange der Flow inaktiv war (d.h. dass keine Pakete in irgendeiner Richtung transportiert wurden) und wann seine Daten zum letzten Mal gesammelt wurden. Sollten Flows nicht oft genug gesammelt werden, kann der Meter bald keinen freien Speicher mehr haben. Dies wird verhindert durch einen niedrig-priorisierten Hintergrundprozess der das Verhältnis aktiver/inaktiver Flows mit der HighWaterMark Variable vergleicht. Ist die Variable kleiner als das Verhältnis, wird die Variable GarbageCollectTime erhöht. Durch das Setzen der HighWaterMark Variablen, wird verhindert, das gesammelte, aber noch nicht abgespeicherten Daten durch den GarbageCollector gelöscht werden. Die Variablenwerte können alle mit NeMaC gesetzt werden (siehe Anhang A).

3.6.2 Beispiele von Flow Dateien

Im folgenden ist eine Beispiel Flow-Datei abgedruckt:

```
##NeTraMet v4.3: -c120 -r ../examples/rules.sample linux-router eth0 10000 flows starting at 20:24:41 Sun 25 Feb 2001
#Format: flowruleset flowindex firsttime sourcepeertype sourcepeeraddress destpeeraddress topdus frompdus tooctets fromoctets
#Time: 20:24:41 Sun 25 Feb 2001 linux-router Flows from 0 to 2800
#Ruleset: 8 2 ../examples/rules.sample NeMaC
#Stats: aps=0 apb=0 mps=0 mpb=0 lsp=0 avi=99.9 mni=100.0 fiu=0 frc=0 gci=10 rpp=0.0 tpp=0.0 cpt=0.0 tts=8191 tsu=0
#EndData: linux-router
#Time: 20:26:00 Sun 25 Feb 2001 linux-router Flows from 2799 to 10607
#Stats: aps=7 apb=0 mps=43 mpb=0 lsp=0 avi=100.0 mni=99.0 fiu=5 frc=0 gci=10 rpp=4.5 tpp=1.0 cpt=1.0 tts=8191 tsu=5
8 1 4847 1 192.168.0.0 192.168.0.0 55 0 7220 0
8 2 4847 1 192.168.0.1 192.168.0.50 91 0 25299 0
8 3 4850 1 192.168.0.0 129.187.0.0 27 0 2387 0
8 4 4850 1 192.168.0.0 192.187.10.0 10 0 923 0
8 5 4924 1 192.168.0.0 213.68.34.0 3 0 186 0
#EndData: linux-router
#Time: 20:28:00 Sun 25 Feb 2001 linux-router Flows from 10606 to 22616
#Stats: aps=12 apb=0 mps=148 mpb=0 lsp=0 avi=100.0 mni=96.4 fiu=5 frc=0
gci=10 rpp=4.7 tpp=1.0 cpt=1.0 tts=8191 tsu=5
8 1 4847 1 192.168.0.0 192.168.0.0 200 0 24593 0
8 2 4847 1 192.168.0.1 192.168.0.50 283 0 235285 0
#EndData: linux-router
#Time: 20:30:00 Sun 25 Feb 2001 linux-router Flows from 22615 to 34625
#Stats: aps=2 apb=0 mps=4 mpb=0 lsp=0 avi=100.0 mni=100.0 fiu=5 frc=0 gci=10 rpp=4.5 tpp=1.0 cpt=1.0 tts=8191 tsu=5
8 1 4847 1 192.168.0.0 192.168.0.0 210 0 25193 0
8 2 4847 1 192.168.0.1 192.168.0.50 291 0 235717 0
#EndData: linux-router
#Time: 20:32:00 Sun 25 Feb 2001 linux-router Flows from 34624 to 46634
#Stats: aps=3 apb=0 mps=3 mpb=0 lsp=0 avi=100.0 mni=100.0 fiu=5 frc=0 gci=10 rpp=3.0 tpp=0.6 cpt=1.0 tts=8191 tsu=5
8 1 4847 1 192.168.0.0 192.168.0.0 211 0 25299 0
8 2 4847 1 192.168.0.1 192.168.0.50 292 0 235823 0
#EndData: linux-router
```

Die erste Zeile enthält den Aufruf von NeTraMet, so wie er auf der Kommandozeile abgesetzt wurde. Der nächste Zeileneintrag bestimmt das Format, nach dem die Daten gesammelt werden sollen z.B. Regel-Nummer, FlowIndex, Zeitpunkt als der Flow zum ersten Mal beobachtet wurde, Quelladresse, Zieladresse, Pakete an Ziel, Pakete an Quelle, Bytes an Ziel, Bytes an Quelle. Die nächste Zeile enthält die Startzeit und die darauf folgende die Regelmenge, die geladen wurde. Die fünfte Zeile enthält eine Liste von Statistikinformationen, die ebenfalls vom Meter gelesen werden sollen, um z.B. gewisse Auslastungsstatistiken daraus erstellen zu können. Folgende Werte können hier ausgelesen werden:

aps - Durchschnitt Pakete/Sekunde
apb - Durchschnitt Pakete Rückstand
mps - max. Pakete/Sekunde
mpb - max. Pakete Rückstand
lsp - Anzahl verlorener Pakete
avi - Durchschnitt Prozessorauslastung %
mni - min. Prozessorauslastung %
fiu - verwendete Flows
frc - gesammelte Flows
gci - Garbage Collector Intervall
rpp - zutreffende Regel/Paket
tpp - Zählungen/Paket
cpt - Vergleiche/Zählung
tts - gesamte zugeordnete Zähltabellen
tsu - verwendete Zähltabellen

3.7 Rules in NeTraMet

Um nur bestimmte Ströme messen zu können, kann man für den Meter verschiedene Regel-Dateien festlegen. Die Regel-Dateien (rule files) sind ASCII-Texte Dateien, die Informationen enthalten, die vom Meter und Collector für diese „maßgeschneiderten" Messungen benötigt werden.

Die Regeln werden in sogenannten Regelsatz zusammengefasst (rule sets). Die Regel-Mengen sind eine Tabelle von Regeln, die durch die Rule-Set Number identifiziert werden. Ein Accounting-Meter kann maximal 10 Regel-Mengen im Speicher haben und gewährleistet dem Manager, einfach zwischen den Mengen hin und her zu springen, indem er die CurrentRuleSet MIB Variable entsprechend setzt.

NeTraMet hat eine Default Regel-Menge „eingebaut", die die SetNumber 1 besitzt. Dadurch kann NeTraMet gleich beim Starten aktiv werden. Andere Regeln können durch den Manager auf den Meter geladen werden. Die Standard Regel-Menge kann durch den Manager nicht geändert werden.

Es gibt zwei Möglichkeiten Regel Dateien zu erstellen. Man kann die SimpleRulesetLanguage direkt benutzen und praktisch „per Hand" seine Regel-Dateien definieren, oder man benutzt eine C ähnliche Syntax und erstellt sich SRL-Programme, die vom SRL Compiler in Objekt Regel-Dateien übersetzt werden.

Eine Regel-Menge ist eigentlich ein Mustererkennungsprogramm für die Pattern Matching Engine (PME) des Meters [RFC 2064]. Mustererkennung kann manchmal ziemlich aufwendig sein, deswegen werden die Regeln zwecks Geschwindigkeitsoptimierung in Gruppen zusammengefasst, wenn sie die gleichen Attribute mit der gleichen Maske testen.

Eine Regel-Überprüfung erfolgt wie in den nächsten Schritten beschrieben:

1 Das Paket wird vom Meter analysiert
2.Zwei Key-Datenstrukturen werden gebildet, eine für die Quelle und eine für das Ziel
3.Ein Vergleich wird gestartet mit der aktuellen Regel-Menge mit den Schlüsseln in der Reihenfolge Quelle->Ziel, wenn eine Übereinstimmung stattfindet, wird in der Flow-Tabelle für diesen Flow der Zähler hochgezählt
4.Wenn die Überprüfung fehl schlägt, wird die Reihenfolge der Schlüsseln umgedreht Ziel->Quelle
5.Wenn diesmal auch keine Übereinstimmung stattfindet, wird das Paket verworfen

Es können symmetrische Vergleiche durchgeführt werden, d.h. man benutzt für beide Richtungen Quelle->Ziel und Ziel->Quelle die gleiche Maske, man kann aber auch Regeln aufstellen, in denen die Richtung auch eine Rolle spielt.

3.7.1 Welche Attribute können getestet werden ?

Die Attribute, die von NeTraMet überprüft werden können, werden in 5 Gruppen aufgeteilt:

adjacent Adresse
peer Adresse
transport Adresse
subscriber (in der aktuellen Version noch nicht implementiert)
general (allgemeine)

Adjacent Attribute: *Erklärung am Beispiel der IP Attribute:*

SourceAdjacentType - aktuelle NeTraMet Version unterstützt passive Schnittstellen, also sind SourceAdjacentType und DestAdjacentType immer gleich
DestAdjacentType

SourceAdjacentAddress - Ethernet MAC Adresse für Source und Dest
DestAdjacentAddress

SourceAdjacentMask - werden in Regeln benutzt um Felder innerhalb der adjacenten Adressen zu testen
DestAdjacentMask

Peer Attribute:

SourcePeerType(IPv4=1) - Netzwerkschicht Protokoll-Nummer
DestPeerType

SourcePeerAddress - IP-Adresse z.B. 192.168.0.1
DestPeerAddress

SourcePeerMask - werden in Regeln benutzt um Felder innerhalb der Peer Adresse zu testen
DestPeerMask

Transport Attribute:

SourceTransType - Nummer für das Protokoll der Transport-Schicht z.B: für TCP hat es den Wert 6
DestTransType
SourceTransAddress - enthalten die Port-Nummer von Quelle/Ziel

DestTransAddress
SourceTransMask - werden verwendet, um die Werte der Trans-Adressen zu überprüfen
DestTransMask

Allgemeine(general) Attribute:

SourceInterface, DestInterface - Schnittstellen-Nummer (1 für eth0)
SourceClass, DestClass, FlowClass - diese Werte werden nicht aus SourceKind, DestKind, FlowKind einem Paket erhalten, sondern können explizit abgespei chert werden
FlowIndex - Index des Flows

ToOctets/FromOctets	- Bytes hin/zurück
ToPDUs/FromPDUs	- Pakete hin/zurück
FirstTime/LastTime	- Zeitpunkt des ersten/letzten Pakets in einem Flow
FlowRuleSet	- Numer der Regelmenge die beim Beobachten des Flow verwendet wurde

3.7.2 Beispiel Rule-Dateien

Eine Regeldatei in SRL geschrieben, sieht wie folgt aus:

```
#
# braodcast.srl: Look for broadcast-packets
#
if SourcePeerType == dummy # PC meter time-filter packets
ignore;

if DestAdjacentAddress == FF-FF-FF-FF-FF-FF {
   save SourcePeerType;
   save SourcePeerAddress/32;
   save DestPeerAddress/32;
   save SourceTransType;
   save SourceTransAddress/16;
   save DestTransAddress/16;
   count;
}
set 9;
```

Kompiliert man die SRL Datei, erhält man folgenden Code:

```
#Source file: test-prog.srl
#Compiled by: SRL compiler, version 4.3
#Time:        11:09:36 Tue  5 Jun 2001
  sourcepeertype & 255.0 = 0.0: ignore, 0;
  destadjacentaddress & 255.255.255.255.255.255 = 255.255.255.255.255.255: gotoact, a1;
  null & 0 = 0: gotoact, n2;
n2:
g2:
  null & 0 = 0: nomatch, 0;
a1:
  sourcepeertype & 255.0 = 0.0: pushpkttoact, next;
  sourcepeeraddress & 255.255.255.255 = 0.0: pushpkttoact, next;
  destpeeraddress & 255.255.255.255 = 0.0: pushpkttoact, next;
  sourcetranstype & 255.0 = 0.0: pushpkttoact, next;
  sourcetransaddress & 255.255 = 0.0: pushpkttoact, next;
  desttransaddress & 255.255 = 0.0: pushpkttoact, next;
  null & 0 = 0: count, 0;
set 9;
```

3.8 Zugriffe und Rechte

Der Zugriff auf einen Meter erfolgt durchNeMaC über SNMP. Man kann auf den Meter auch über andere SNMP Tools zugreifen, wie z.B. mit dem HP Openview Network Node Manager. Der Benutzer kann über die Rule Dateien angeben, welche Daten gemessen werden sollen, in dem er sich eine entsprechende Regeldatei definiert, die mit Hilfe von NeMaC an den entsprechenden NeTraMet Meter weitergereicht wird.

Der Manager und Collector (NeMaC) kann gleichzeitig mit mehreren Metern kommunizieren. Es gibt die Möglichkeit, sich für den NeMaC eine Konfigurationsdatei (*.cfg) zu definieren, in der man all die Meter, mit denen der Manager in Verbindung steht, angeben kann (siehe Abschnitt I.1).

Bezüglich der Rechte, die man benötigt, um den Meter zu starten, so kann NeTraMet nur als root gestartet werden, wobei NeMaC als „normaler" User gestartet werden kann, mit der Voraussetzung, daß er auf das Verzeichnis, in denen die flow- und log-Dateien geschrieben werden sollen, Schreibrechte besitzt.

3.9 Beispielszenario

NeTraMet läuft auf einem Webserver und misst alle Anfragen, die von außerhalb an den Server gehen (simuliert werden). Normalerweise setzt man einen Meter auf einem Router, da dort der meiste Verkehr „abgefangen” werden kann. Um dem User eine Möglichkeit zu geben, sich leicht einfache Regel Dateien zusamme zu stellen, wird im nächsten Schritt ein Eingabe-Formular dafür implementiert.

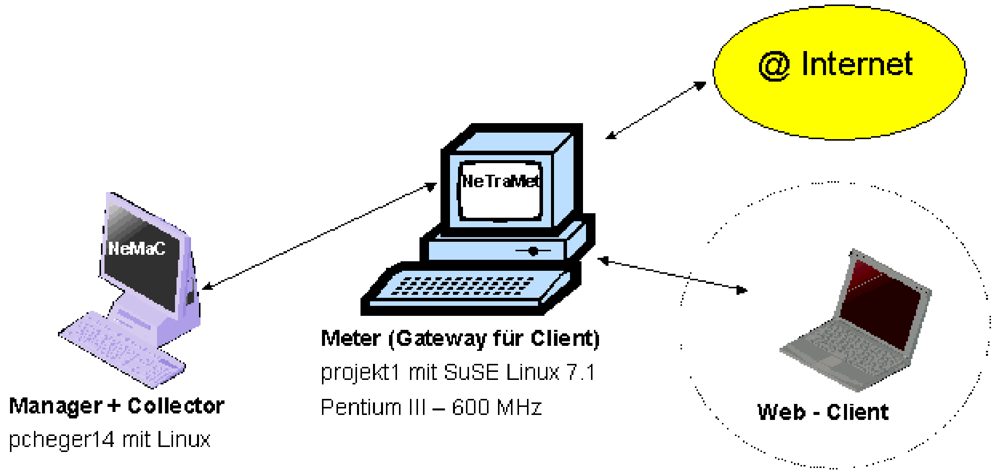

Abbildung 4: TrafficDataProvider Testbed

3.10 Tools um NeTraMet

Im Laufe der Zeit sind eine Menge Tools entwickelt worden, die zusammen mit NeTraMet verwendet werden können. Es gibt eine Fern-Konsole (*nm_rc* = NeTraMet remote Console), das NeMaC mit einem anderen Tool namens *fd_filter* verbindet, um es zu ermöglichen, daß man immer nach einer gewissen Anzahl von Sekunden, die „paketreichsten” Flows angezeigt bekommt. Ein anderes Tool, das noch zu erwähnen wäre, ist *nifty*, daß an Hand der Traffic Dateien, den Traffic durch sogenannte Plots darstellen kann.

4. Der MASA TrafficDataProvider Agent

Der TrafficDataProvider Agent wurde implementiert, um mit Hilfe von NeTraMet den Netzverkehr nach selbst definierten Regeln zu messen.

Der Agent ist aus dem DataProvider Agent [Lore 01] abgeleitet und wurde vollständig in das MASA Agentensystem integriert.

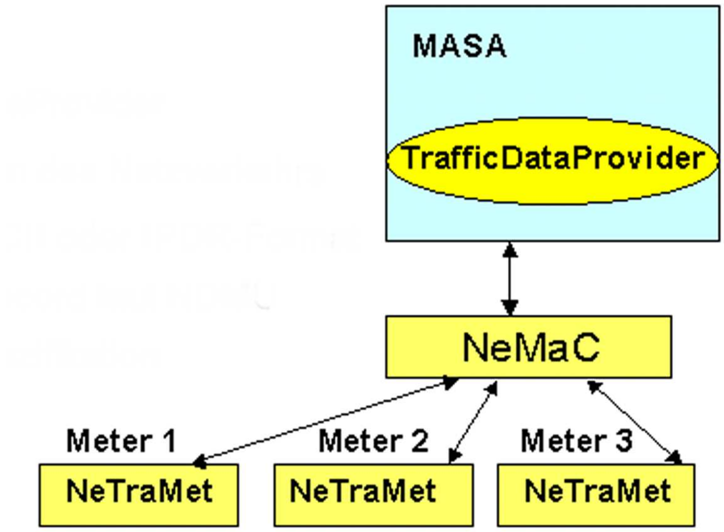

Abbildung 5: TrafficDataProvider Architektur

4.1 Aufbau des TrafficDataProvider Agenten

Der Agent umfasst drei Konfigurationsbereiche, die wiefolgt aufgeteilt sind:

RULE EDITOR
METER CONFIGURATION
GENERIC ACTIONS

4.2 Rule Editor

Der Rule Editor, stellt ein Formular zur Verfügung, um neue Regel-Dateien zu erstellen. Das ist der erste Schritt, den ein User machen muß, um Messungen durchführen zu können. Der Benutzer muss keine Kenntnisse über die SimpleRulesetLanguage (SRL) haben. Es können mit Hilfe des Formulars drei Arten von Regeln erstellt werden. IP-, MAC- und Transport-Adressen können überprüft werden. Nur eine Adresse muß angegeben werden, die dann von NeTraMet, einmal als Quell- und falls nicht vorhanden, als Zieladresse verwendet wird. Die Entscheidung, in welcher Richtung der Flow geht kann und muss nicht getroffen werden, diese Aufgabe übernimmt NeTraMet automatisch. Der Anwender kann entscheiden, welche Informationen er zu einem Flow erhalten möchte, in dem er die Formatierungen (Format Block im Formular) entsprechend auswählt, und ob der Flow tatsächlich registriert oder verworfen werden soll (die Option count oder ignore im Formular). Zusätzlich kann zu jeder Regel, noch ein Kommentar abgespeichert werden, der dazu dienen soll, die „Wirkung“ der Regel leichter zu identifizieren. Die erstellten Regeln können jederzeit editiert und abgeändert werden. Das Editieren einer Regel-Datei setzt aber SRL Kenntnisse voraus.

4.2.1 Aktionen

peer rule	-öffnet Formular zur Überprüfung einer IP-Adresse
adjacent rule	-öffnet Formular zur Überprüfung einer MAC-Adresse
transport rule	-öffnet Formular zur Überprüfung einer Transport-Adresse
edit rule	-editiert eine bereits erstellte Regel-Datei
save rule	-speichert die Änderungen in der Rule-Datei ab

Nach dem der Benutzer seine Regeln definiert hat, muß er nun im nächsten Schritt den gewünschten Meter konfigurieren.

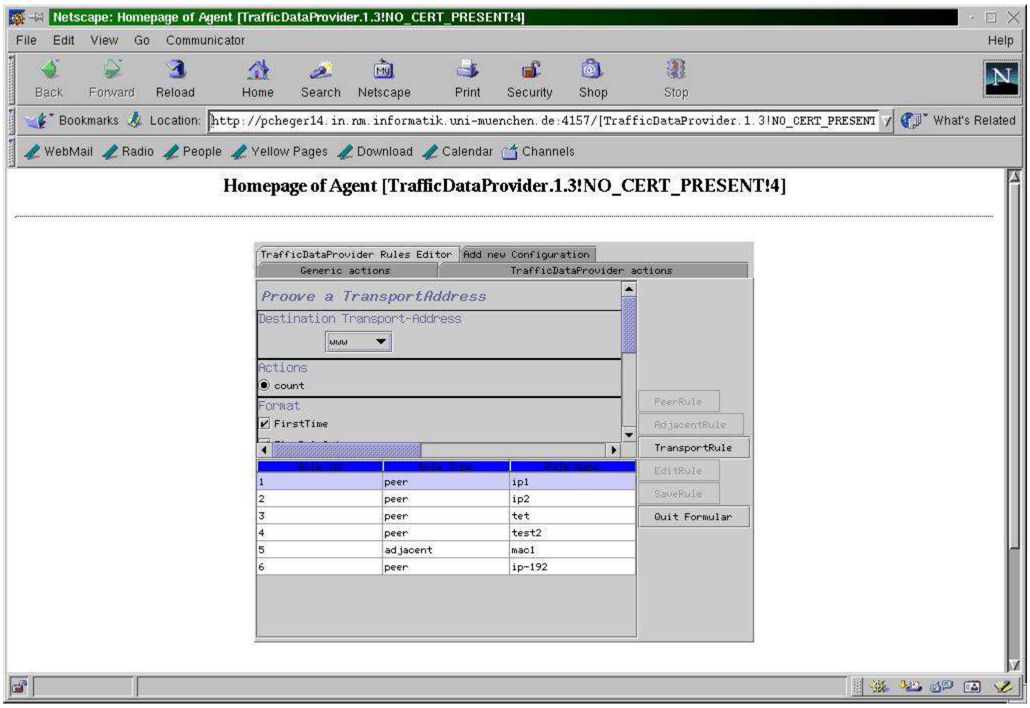

Abbildung 6: Der Rule Editor des TrafficDataProvider Agenten

4.3 Meter Configuration

In diesem Konfigurationsbereich wird der User entscheiden können, welche Regeln er auf einen Meter uploaden möchte. Er bekommt eine komplette Liste aller erstellten Regeln (available Rules) zur Verfügung und kann dann aus dieser auswählen, welche er für den Meter, dessen IP-Adresse ebenfalls in diesem Bereich anzugeben ist, hochladen möchte. In der Meter Configuration kann auch noch das Abfrageintervall in Sekunden angegeben werden und die SNMP write-community kann gesetzt werden. Über den „CONFIGURE" Button wird die Konfiguration für den Meter gespeichert, das Starten des Meters mit der erstellten Konfiguration erfolgt dann über den „START"- Button aus dem „Generic Actions" Bereich, in dem man in einer Tabelle alle Konfigurationsdateien angezeigt bekommt und bei Bedarf nochmal editieren kann.

4.3.1 Aktionen

add	-fügt die ausgewählte Rule-Datei der Konfigurationsdatei hinzu
rem	-entfernt die Rule-Datei aus der Konfiguration des Meters
configure	-erstellt und speichert die Konfigurationsdatei des Meters ab

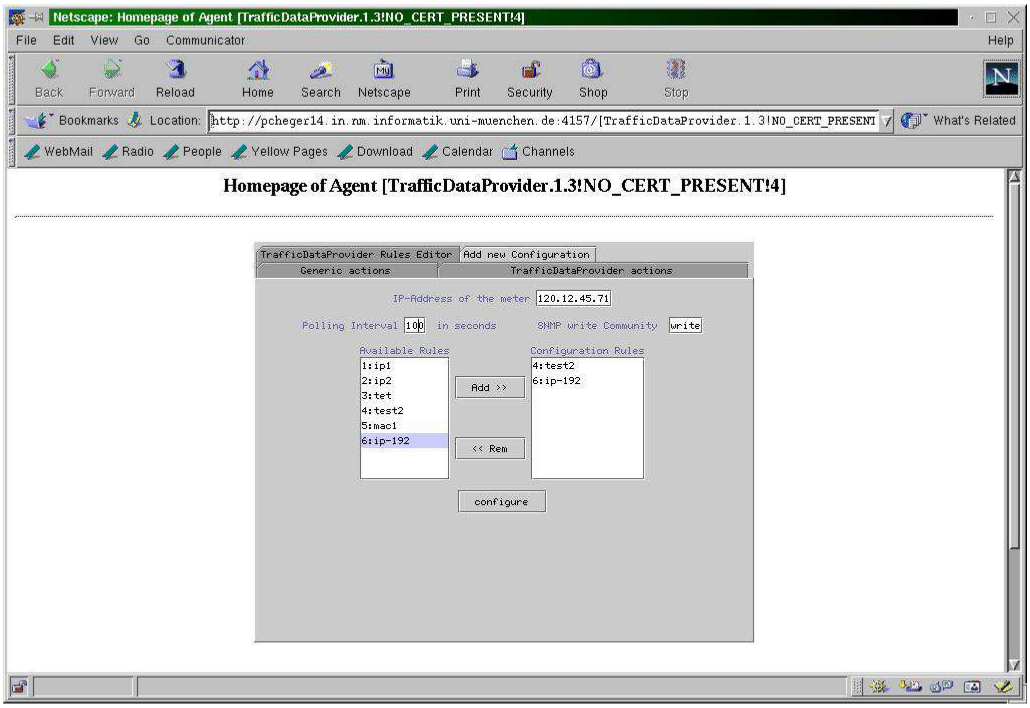

Abbildung 6: Configuration des Meters im TDPA

Wenn die Konfigurationsdatei für den Meter richtig erstellt wurde (Configure Button), bekommt der Anwender eine entsprechende Meldung angezeigt, im Gegenfall eine Fehlermeldung. Um nun die Messungen zu starten, muß über die "Generic Actions" der Meter gestartet werden.

4.4 Generic Actions

Dieser Konfigurationsbereich ist die „Zentrale" des Agenten. Hier werden alle Meter, die konfiguriert wurden, aufgelistet. Hier kann durch das Auswählen eines Meters die Messung dann gestartet werden. In den „Generic Actions" kann die Konfigurationsdatei eines jeden Meters jederzeit abgeändert werden und erneut auf den Meter geladen werden. Das Abfragen der einzelnen Meter erfolgt ebenfalls in diesem Bereich über den „Query" Button. Man hat die Möglichkeit, sich die Abfrage in zwei Formaten anzeigen zu lassen, entweder als flow-Anzeige (ASCII), wie sie von NeTraMet generiert wird, oder im IPDR-Format als XML Datei. Das IPDR (Internet Protocoll Data Record)-Format ist ein von der IPDR (www.ipdr.org) working group definiertes Format zur Darstellung von Daten entsprechend der Network Data Management Usage Spezifiktion. DieAusgabe erfolgt in einem XML-Dokument. Der TrafficDataProvider Agent ermöglicht es, die von den Meter gesammelten Daten im IPDR-Format als XML-Dateien abzuspeichern (CREATE XML).

4.4.1 Aktionen

start	- lädt auf den Meter die in der Konfigurationsdatei eingetragenen Regeln
edit configuration	- editiert die ausgewählte Konfigurationsdatei des Meters
save configuration	- speichert die Änderungen in der Konfigurationsdatei des Meters ab
quit edit/query	- beendet die Edit/Query-Anzeige
create xml	- erzeugt xml-Datei aus den gesammelten Daten im IPDR Format
query xml/flow	- zeigt die gesammelten Daten entweder als xml oder flow an

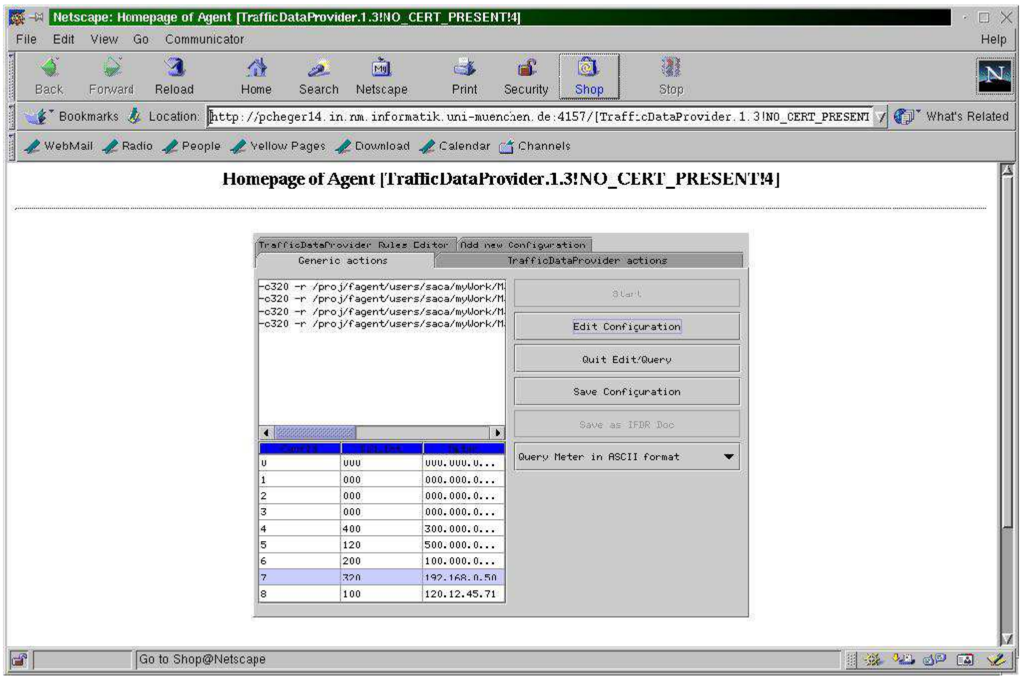

Abbildung 4: Frontpanel des TrafficDataProvider Agenten (Generic Actions)

4.5 Bemerkugen zur TrafficDataProvider API

Der TrafficDataProvider Schnittstelle implementiert die DataProvider Schnittstelle und erweitert diese um zwei Methoden. Die Methode „refresh“ dient dazu eine automatische Aktulisierung der Bildschirmanzeige zu erzielen. Die zweite Methode, mit der das DataProvider Interface erweitert wurde, ist die Methode „getPlainContent“, die dazu dient den Inhalt einer Rule- oder Konfigurationsdatei auszulesen und als einfachen Text auf dem Bildschirm anzuzeigen. Die Standardmethoden der DataProvider Schnittstelle sind von „außerhalb“, wie gewohnt aufzurufen. Bei den Methoden „addConfig“ und „editConfig“ sollten aber bei den Parameter der Methode auf folgendes geachtet werden:

addConfig(String id, String[] config)

- die „id“ muss angeben, ob es sich um die Definiton einer Regel- oder Konfigurationsdatei handelt
- ist id=RULE, dann soll eine Regeldatei erstellt werden und das config-Array muss dann folgende drei String-Werte enthalten: type(mac,ip,trans), name, file
- ist id=CONFIG, dann wird eine Konfigurationsdatei erstellt und das String-Array muß die folgenden drei Einträge führen: configfile(Name der Konfigurationsdatei), interval(Abfrageintervall), community(SNMP write-community)

editConfig(String id, String[] config)

- die „id“ entscheidet, ob eine bereits bestehende editierte Konfigurationsdatei gespeichert wird, oder
-ist id=“SAVE“ dann wird durch den Aufruf von saveFile( ), die Datei gespeichert
- ist id=“PUSH“, so wird durch den aufruf einer Hilfsmethode, die Liste der Konfigurationen aktualisiert
- das config-Array hat im Vergleich zu den anderen DataProvider Agenten, keinen Einfluss auf die Funktionalität des Agenten

Der TrafficDataProvider speichert all seine Nutzdaten in Dateien ab. Die Liste der verfügbaren Regeldateien, als auch die Tabelle der Konfigutaionen werden in getrennten Dateien verwaltet. Es sind eigentlich die zwei zentralen Dateien für den Agenten, die ihm als Datenbank dienen. Die Namen und Positionen der beiden Dateien im Dateisystem sind aus der property-Datei zu entnehmen.

5 Zusammenfassung und Ausblick

Der TrafficDataProvider Agent kommt dem Anwender sehr entgegen, da dieser keine NeTraMet Kenntnisse mitbringen muß, um Messungen durchführen zu können. Auch sind keine SRL Kenntnisse notwendig, um neue Regeln zu definieren. NeTraMet ist ein frei erhältliches Tool, das ohne großen Aufwand auf den verbreitesten System installiert werden kann. Es weist sicherlich auch ein paar Schwachstellen auf, wie z.B., die Tatsache, daß die Daten nicht agregiert werden können, sondern ein Flow auch mehrmals in den gesammelten Daten geführt wird. Womöglich kämen auch alternative Traffic Measurement Tools in Frage, wie z.B. ipaudit [ipa], das ebenfalls frei erhältlich ist, dafür aber nicht die Flexibilität, die NeTraMet mit der Simple Ruleset Language bietet, ermöglicht. Zum TrafficDataProvider könnte man sich noch Erweiterungen im Bereich der automatischen Installation von NeTraMet auf den gewünschten Meter überlegen, bzw. wäre die Abspeicherung der Konfiguration, der Regeln und der gesammlten Daten in einer Datenbank wie z.B. mySql [sql] (ebenfalls frei verfügbar) in Erwägung zu ziehen.

Anhang A : Optionen für NeTraMet und NeMaC

NeTraMet

-f nnn	setzt die maximale Flowanzahl auf nnn
-i ifn	teilt NeTraMet mit, welche Schnittstelle zu überwachen ist
-I ifn	teilt NeTraMet mit, daß diese Schnittstelle für die IP Kommunkation benutzt werden soll, aber nicht für die Überwachnung
-l	teilt dem Meter mit, daß die Längenangabe für die Anzahl der Bytes in einem Paket aus dem Feld im IP Header verwendet werden soll
-m ppp	spezifiziert den UDP Port, der für die Kommunikation mit den Meter Lesern wie z.B. NeMaC verwendet werden soll
-r rsc	spezifiziert eine Lese-Community für NeTraMet (nur eine darf spezifiziert werden)
-w wsc	spezifiziert die Schreib-Community (default is private) , Lese- und Schreib-Community müssen nicht identisch sein

NeMaC

-b mibfile	gibt die MIB-Datei an
-c nnn	setzt den Sammelinterval
-e file	spezifiziert die Rule Datei die auf den Meter geladen werden soll
-f cfgfile	spezifiziert die Konfigurationsdatei für den NeMaC
-g sss	spezifiziert den GarbageCollector Zeitintervall
-m ppp	spezifiziert den Port über den mit dem Meter kommuniziert wird (default 161)
-F name	spezifiziert den Namen der Flow-Datei

Literaturverzeichnis

[Flan 99] D. Flanagan, J. Farley, W. Crawford, K. Magnusson
„Java Enterprise in a nutshell“, O`Reilly, Erste Auflage, September 1999

[NTM_SWD] NeTraMet Software und Dokumentation
http://www2.auckland.ac.nz/net/Accounting/ntm.Release.note.html

[RFC 1272] C. Mills, D. Hrish, G. Ruth: RFC 1272: „Internet Accounting: Background“, RFC, Network Working Group, November 1991

[RFC 2722] N. Brownlee, C. Mills, G. Ruth: RFC 2722: „Traffic Flow Measurement: Architecture“, RFC,Network Working Group, Oktober 1999

[RFC 2123] N. Brownlee: RFC 2123: „Traffic Flow Measurement: Experiences with NeTraMet“, RFC,Network Working Group, März 1997

[RFC 2723] N.Brownlee: RFC 2723: „SRL: A Language for Describing Traffic Flows and Specifying Actions for Flow Groups“, RFC, Network Working Group, Oktober 1999

[RFC 2512] J. Case, K. McCloghrie, M. Rose, S. Waldbusser: RFC 2512 „Introduction to Community-based SNMPv2“, RFC, Network Working Group, Januar 1996

[RFC 1252] K. McCloghrie, J. Heinanen, W. Green, A. Prassad, RFC1252: „Accounting Information for ATM Networks“, RFC, Network Working Group, Februar 1999

[RFC 2064] N. Brownlee: RFC 2064: „Traffic Flow Measurement: Meter MIB“, RFC, Network Working Group, Januar 1997

[Lore 01] B. Lorenz: Fortgeschrittenen-Praktikum: „Erweiterung der MASA-Umgebung um Accounting Funktionalität“, August 2001

[Kemp 98] B. Kempter: Diplomarbeit Technische Universität München: „Entwurf eines Java/ CORBA-basierten Mobilen Agenten“, August 1998

[Ipa] Network Traffic Measurement with IPAUDIT
http://ipaudit.sourceforge.net

[Sql] MySql relationale Datenbank
http://www.mysql.com