



IsarFlow Whitepaper

Benutzerdefiniertes Billing & Accounting mit IsarFlow

Einführung

In modernen Unternehmensstrukturen wird eine verursachergerechte Abrechnung der Nettwerkkosten immer wichtiger, um finanzielle Transparenz gewährleisten zu können. IsarFlow bietet hierzu die Erfassung der Datenströme geordnet nach einzelnen IP Adressen oder ganzen Adressbereichen sowie eine flexible und bedienerfreundliche Konfiguration und Administration:

- Die Definitionen der Accounting-Regeln können individuell über ein spezielles Konfigurationsfile im IsarFlow-Analyzer erstellt und gepflegt werden.
- Die Abrechnung der gesammelten Daten erfolgt täglich
- Am Monatsende erfolgt eine automatische Zusammenfassung sämtlicher ausgewerteten Daten
- Die Daten können bequem im CSV-Format exportiert oder live über das command line interface (CLI) abgefragt

Das vorliegende Whitepaper betrachtet beispielhafte Kundenanforderungen und die entsprechende Umsetzung mit IsarFlow. Sämtliche Accounting-Daten werden in Intervallen von einem Tag erfasst, zusätzlich erfolgt eine Zusammenfassung am Monatsende.



Case Study #1 – Per Subnet Accounting

Im vorliegenden Fall soll ermittelt werden, wie sich der gesamte Netzwerk-Verkehr auf die Subnets verteilt. Zu diesem Zweck werden die zu erfassenden Subnets im Konfigurationsfile 'server_networks.cfg' aufgeführt. Alle Netze, die dort nicht erfasst sind werden bei der Auswertung der NetFlow-Daten nicht berücksichtigt

Pro Subnet wird die zugehörige Netzmaske mit angegeben, so dass sowohl IP Netzadressen als auch IP Hostadressen gemischt angegeben und ausgewertet werden können.

Dieses Accounting-Modell wird eingesetzt, wenn bestimmte Subnets verrechnet werden sollen und es irrelevant ist, von oder zu welchen Netzen der Verkehr gesendet wird.

Subnet Accounting – Konfiguration

Konfiguration der Abrechnungsbasis, hier der Benutzerdaten bzw. Subnetze in `billing.conf`:

```
[server_networks]
Definition          = SUBNET(IP, src, dst), sum(bytes), sum(packets)
IMPORT              = IP, ../etc/server_networks.cfg
```

Anlegen der oben angegebenen Importdatei `server_networks.cfg`:

```
bash$ cat ../etc/server_networks.cfg
#
# subnet's to be accounted
#
10.0.0.0/24
10.0.1.0/24
172.20.0.0/16
```

Subnet Accounting – Ergebnis

Subnet	Bytes	Packets
10.0.0.0/24	1042211808	2039896
172.20.0.0/16	997546518	1460779
10.0.1.0/24	23359578	332136

Die Summe über alle Zeilen im Ergebnis kann das tatsächlich transportierte Volumen überschreiten. Dies liegt daran, dass das Volumen für jedes Subnet aufaddiert wird, unabhängig davon, ob das Subnet als Source oder als Destination erfasst ist.



Case Study #2 – Server Accounting

Als Abweichung zum Subnet-Accounting können anstelle der Paare IP Subnet / Mask auch einzelne IP Hostadressen in der Konfigurationsdatei 'ip_only.cfg' aufgelistet werden. Dies wird beispielsweise verwendet, um ein monatlich angefallenes Datenvolumen pro Server zu verrechnen.

Auch hier gilt analog zum vorherigen Beispiel: alles was von und zu dieser Adresse gesendet wird, wird im Accounting berücksichtigt.

Server Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[ip_addr]
DEFINITION          = GROUP(IP_ONLY, inet_ntoa(src,dst)), sum(bytes)
IMPORT              = IP_ONLY, ../etc/billing/ip_only.cfg
```

Anlegen der oben angegebenen Importdatei:

```
bash$ cat ip_only.cfg
#
# single ip addresses to account
#
10.0.0.200
10.0.0.205
10.0.0.254
10.0.1.254
```

Server Accounting – Ergebnis

Server	Bytes
10.0.0.200	1004824231
10.0.0.254	17200889
10.0.0.205	4135528
10.0.1.254	64435



Case Study #3 – Net2Net Accounting

Im Gegensatz zum Subnet-Accounting wird beim Net2Net Accounting der Kommunikationspartner ebenfalls berücksichtigt, dadurch können Subnet-Paare doppelt auftreten.

Für die Kommunikation zwischen zwei Netzen A und B zwei verschiedene Einträge erstellt (pro Richtung einer). Dies gilt auch dann, wenn Source- und Destination IP Adresse im gleichen Subnet liegen.

Net2Net Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[subnet_to_subnet]
Definition          = matrix(subnet(ID1, src),subnet(ID1, dst)),
                    sum(bytes), sum(packets)
Import              = ID1, ../etc/billing/subnet_to_subnet.cfg
```

Anlegen der oben angegebenen Importdatei:

```
bash# cat subnet_to_subnet.cfg
#
# specify subnets here
#
10.0.0.0/24
10.0.1.0/24
217.233.0.0/16
217.172.0.0/16
```

Net2Net Accounting – Ergebnis

Subnet 1	Subnet 2	Bytes	Packets
10.0.0.0/24	10.0.0.0/24	19831610	229827
217.233.0.0/16	217.233.0.0/16	30121694	106599
10.0.1.0/24	10.0.1.0/24	3058186	28325
10.0.0.0/24	10.0.1.0/24	2004413	25993
217.172.0.0/16	217.233.0.0/16	67038	58
10.0.0.0/24	217.172.0.0/16	3506	43



Case Study #4 – Protokoll Accounting

Während in der graphischen Oberfläche von IsarFlow automatisch alle Protokolle, die im Netz erkannt wurden, ausgewertet werden, kann man sich beim Accounting durch die Konfiguration auf wenige Protokolle konzentrieren. Nur für diese relevanten und vordefinierten Protokolle werden die Summen erfasst.

Protokoll Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[protocol]
DEFINITION          = GROUP(PROTOCOL, protocol), sum(bytes)
IMPORT              = PROTOCOL, ../etc/billing/protocol.cfg
```

Anlegen der oben angegebenen Importdatei:

```
bash# cat protocol.cfg

#
# accounting for the following protocols:
#
ftp-data
Ncp
www-http
Iso-tsap
sna_ee
smtp
telnet
ICMP
ssh
snmp
```

Protokoll Accounting – Ergebnis

Application	Bytes	Packets
ftp-data	113.615.940	140.854
Ncp	58.603.791	218.765
www-http	30.333.451	60.326
Iso-tsap	26.749.179	193.518
sna_ee	11.257.585	24.803
smtp	8.312.526	12.319
telnet	4.308.584	70.219
ICMP	4.086.333	9.516
ssh	3.875.795	44.655
snmp	3.192.100	42.377



Case Study #5 – Server/Department Accounting

Die folgende Accounting Konfiguration kombiniert einzelne IP-Adressen mit Subnets. Dieser Ansatz ist sehr nützlich, um beispielsweise den Zugriff von Abteilungen (Subnets) auf zentrale Ressourcen (IP-Adressen) erfassen zu können.

Das Ergebnis ist hierbei eindeutig, d.h. es gibt keine "doppelten Paare" von Einträgen wie unter Net2Net.

Server/Department Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[departement_server]
DEFINITION                = hash(DEPARTMENT,
                                subnet(IP_INFO, src, dst)),
                                hash(SERVER_INFO,
                                subnet(SERVER_IP, src, dst)),
                                protocol, sum(bytes),
                                sum(packets)
IMPORT                     = IP_INFO, ../etc/billing/ip_info.cfg
IMPORT2                    = DEPARTMENT, ../etc/billing/ip_info.cfg
IMPORT3                    = SERVER_IP, ../etc/billing/server.cfg
IMPORT4                    = SERVER_INFO, ../etc/billing/server.cfg
```

Anlegen der oben angegebenen Importdateien:

```
bash# cat etc/ip_info.cfg
#
# Department networks + mapping to names
#
10.0.1.0/24;   LAB-Netz
10.0.0.200/32; OFFICE
172.20.0.0/16; VPN
```

```
bash# cat etc/server.cfg
#
# server ip addresses + mapping to names
#
10.0.0.254/32; CENTRAL_SERVER
10.0.1.14/32;   GSX
10.0.0.205/32; GSX-2
10.0.0.200/32; File-Server
```

Server/Department Accounting – Ergebnis

Department	Server	Bytes	Packets
OFFICE	File-Server	1004824231	1613270
VPN	File-Server	990160195	1421661
OFFICE	CENTRAL_SERVER	11683486	163493
LAB-Netz	File-Server	1444655	14326
LAB-Netz	GSX	1091862	10400
VPN	CENTRAL_SERVER	72492	1340
LAB-Netz	CENTRAL_SERVER	41039	302



Case Study #6 – Department/Port Accounting

In diesem Fall werden Protokoll und Subnet-Informationen kombiniert. Somit können z.B. bestimmte Services (Protokolle) für einzelne Subnetze abgerechnet werden.

Department/Port Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[Department_Port]
DEFINITION          = GROUP (PROT_MAP, hash (DEPARTMENT,
                                subnet (IP_INFO, src, dst)), protocol), sum (bytes)
IMPORT               = IP_INFO, ../etc/billing/ip_info.cfg
IMPORT2              = DEPARTMENT, ../etc/billing/ip_info.cfg
IMPORT3              = PROT_MAP, ../etc/billing/dep_prot.cfg
```

Anlegen der oben angegebenen Importdateien:

```
bash# cat etc/ip_info.cfg
#
# Departments
#
10.0.1.0/24; LAB-Netz
10.0.0.200/32; OFFICE
172.20.0.0/16; VPN
```

```
bash# cat etc/dep_prot.cfg
#
# department/protocol to bill
#
OFFICE; snmp
OFFICE; tacacs
OFFICE; radius
OFFICE; ICMP
LAB-Netz; ypbind
LAB-Netz; nntp
```

Department/Port Accounting – Ergebnis

Department	Protocol	Bytes
OFFICE	ICMP	14371156
OFFICE	snmp	10343245
LAB-Netz	nntp	992472
OFFICE	Tacacs	152673



Case Study #7 – einfaches MPLS/VPN Accounting

Dieses Beispiel gilt für ein "normales" VPN (ohne shared central services). NetFlow ist eingeschaltet auf allen PE-Interfaces ("ingress"). Somit kann kein doppeltes Accounting stattfinden.

Die VPN-Zugehörigkeit wird anhand von 2 Parametern identifiziert : die IP-Adresse des PE-Routers und der SNMP Interface Index des VPN-Interfaces. Über eine hash()-Funktion werden diese Parameter auf VPN-Namen gemappt.

Einfache MPLS/VPN – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[simple_vpn]
#
# Netflow is only exported at PE ingress direction (no duplicate flows in DB)
#
DEFINITION      = hash(VPN_INFO, inet_ntoa(ed_ip), input), sum(bytes), sum
                 (packets)
CONSOLE_HEADER  = VPN, Bytes, Packets
IMPORT          = VPN_INFO, ../etc/SIMPLE_VPN.txt
ACTIVE         = 0
```

Import File:

```
#
# VPN_INFO.txt
#
# ed_ip, ifIndex -> Customer-VPN
#
10.0.254.1; 1; VPN-A
10.0.254.1; 38; VPN-B
10.0.254.1; 39; VPN-B
```

Einfaches MPLS/VPN Accounting – Ergebnis

VPN	Bytes	Packets
VPN-A	964,181,523	7,180,576
VPN-B	284,611,924	1,720,016



Case Study #8 – MPLS/VPN Accounting pro Site

In diesem Fall ist NetFlow sowohl Ingress also auch Egress auf den PE's aktiviert. Dies bedeutet doppelte Flow-Informationen in der Datenbank, erlaubt jedoch eindeutiges & vollständiges Accounting pro Lokation.

Die IP-Adresse des PE-Routers und die der SNMP Interface Index (input ODER output) werden auf VPN-Namen gemappt.

MPLS/VPN Accounting pro Site – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[vpn]
#
# all flows to/from customer-site are exported via PE (mpls egress accounting)
#
DEFINITION      = hash(VPN_INFO, inet_ntoa(ed_ip), input?, output?),
                 sum(bytes), sum(packets)
CONSOLE_HEADER  = VPN, Bytes, Packets
IMPORT          = VPN_INFO, ../etc/SIMPLE_VPN.txt
ACTIVE         = 0
```

Import File:

```
#
# VPN_INFO.txt
#
# ed_ip, ifIndex -> Customer-VPN / SITE
#
10.0.254.1; 1; VPN-A
10.0.254.1; 38; VPN-B
10.0.254.1; 39; VPN-B
```

MPLS/VPN Accounting pro Site – Ergebnis

VPN	Bytes	Packets
VPN-A	7,423,968,560	15,498,664
VPN-B	1,091,865,434	7,461,548



Case Study #9 – Subnet pro VPN Accounting

Die Aktivierung von NetFlow erfolgt hier wie in der Case Study #7 (ingress/egress). Zusätzlich zur IP-Adresse des PE-Routers und des Interface Indexes wird die Subnet-Information für die VPN-Namenszuordnung verwendet. Dies ermöglicht ein Accounting im MPLS, wenn verschiedene Kunden gleiche Subnets verwenden.

Subnet pro VPN Accounting – Konfiguration

Feldbezeichner mit angehängtem '?' (siehe Beispiel-Konfiguration) gehören optional zum Hash-Key.

Anpassung der Konfigurationsdatei `billing.conf`:

```
[subnet_per_vpn]
#
# per subnet/per vpn accounting
#
DEFINITION      = subnet(IP_INFO, src, dst), hash(VPN_INFO, subnet(IP_INFO,
src, dst), inet_ntoa(ed_ip), input?, output?), sum(bytes), sum(packets)
CONSOLE_HEADER  = Subnet, VPN, Bytes, Packets
IMPORT          = VPN_INFO, ../etc/VPN_INFO.txt
IMPORT2         = IP_INFO, ../etc/VPN_INFO.txt
ACTIVE          = 0

Import File:

#
# VPN_INFO.txt
#
# ed_ip, ifIndex -> Customer-VPN
#
10.0.0.0/24; 10.0.254.1; 1; VPN-A
10.0.1.0/24; 10.0.254.1; 1; VPN-A
10.0.0.0/24; 10.0.254.1; 38; VPN-B
10.0.1.0/24; 10.0.254.1; 38; VPN-B
10.0.0.0/24; 10.0.254.1; 39; VPN-C
10.0.1.0/24; 10.0.254.1; 39; VPN-C
```

Subnet pro VPN Accounting – Ergebnis

Subnet	VPN	Bytes	Packets
10.0.0.0/24	VPN-A	994,779,209	7,877,820
10.0.0.0/24	VPN-B	827,942,465	5,861,859
10.0.1.0/24	VPN-A	247,337,971	1,194,709
10.0.1.0/24	VPN-B	227,521,032	1,124,320
10.0.0.0/24	VPN-C	12,156,078	82,217



Case Study #10 – Subnet/TOS Accounting

Dieses Verfahren ermöglicht Subnets anhand ihrer Priorisierung zu erfassen.

Subnet/TOS Accounting – Konfiguration

Anpassung der Konfigurationsdatei `billing.conf`:

```
[subnet_tos]
DEFINITION      = subnet(IP_INFO, src, dst), hash(TOS, tos),
                  sum(bytes), sum(packets)
CONSOLE_HEADER  = SUBNET, TOS, BYTES, PACKETS
IMPORT          = IP_INFO, ../etc/SUBNET_TOS_IP.txt
IMPORT2        = TOS, ../etc/TOS.txt
ACTIVE         = 0
```

IMPORT-FILE 1: Subnets die erfasst werden sollen

```
#
# sample input file for subnet billing
#
10.0.0.0/24
10.0.1.0/24
172.20.0.0/16
```

IMPORT-FILE 2: Zuordnung von TOS-Wert zu Klassen-Namen

```
#
# TOS byte mapping for QoS billing
#
0; Best Effort
96; Class 1
112; Class 2
16; Class 1
8; Class 2
```

Subnet/TOS Accounting – Ergebnis

Subnet	TOS	Bytes	Packets
10.0.0.0/24	Class 1	248,516,917	4,080,459
10.0.0.0/24	Best Effort	676,835,942	3,560,342
172.20.0.0/16	Best Effort	603,613,403	2,399,300
10.0.1.0/24	Best Effort	290,322,449	1,384,079
172.20.0.0/16	Class 1	93,166,403	700,725
10.0.0.0/24	Class 2	113,686,488	477,787
172.20.0.0/16	Class 2	106,774,196	357,470
10.0.1.0/24	Class 1	16,859,674	287,261
10.0.1.0/24	Class 2	1,358,909	17,632



n3k Informatik GmbH
offizieller IsarNet Partner

Ferdinand-Braun-Str. 3
74074 Heilbronn, Germany
www.n3k.de

T: +49 (0) 71 31 59 49 5 - 0
F: +49 (0) 71 31 59 49 5 - 100
vertrieb@n3k.de