

Gestione di Reti

Alessio Delgadillo

Anno accademico 2020-2021

Indice

I Teoria	4
1 Introduzione	5
1.1 Motivazioni	6
1.2 Network Management Dimensions	7
1.3 Terminologia e concetti fondamentali	8
1.3.1 Managed Objects	8
1.3.2 Management Information Base (MIB)	9
1.3.3 Manager/Agent Paradigm	9
1.3.4 Functional Areas (FCAPS)	10
1.3.5 Management Architectures Overview	11
1.3.6 Servizi e protocolli: alcune definizioni	12
1.3.7 Rappresentazione e stratificazione dei servizi	12
1.3.8 Time Diagrams	12
1.3.9 ISO/OSI-Reference Model	14
1.3.10 Internet Layer Model	15
1.3.11 Standardization	16
1.4 Overview: Abstract Syntax Notation One	17
1.4.1 Primitive ASN.1 Datatypes	18
1.4.2 Assembled ASN.1 Datatypes	19
1.4.3 Reduced Datatypes	19
1.4.4 Basic Encoding Rules (BER)	20
2 Internet Management	23
2.1 Overview	23
2.2 Structure of Management Information (SMIV2)	25
2.2.1 Object Identifier and Instance Identifier	27
2.2.2 MIB Module	30
2.2.3 MIB-Compiler	35
2.3 Fundamental MIBs	36
2.4 SNMP Version 1	37
2.4.1 Formato messaggi SNMPv1	39

2.5	SNMP Version 2c	42
2.6	SNMPv3	45
2.6.1	Modello architetturale di SNMPv3	46
2.6.2	Problemi di sicurezza	46
2.6.3	Agents SNMP	48
II	Pratica	51
3	Ethernet	52
3.1	Ethernet Framing	56
3.2	Operation of Ethernet Switches	57
4	SNMP Monitoring	59
4.1	“system” Group	59
4.2	“interface” Group	60
4.3	Uso del gruppo “arp”	63
4.4	Bridge MIB	63
4.5	SNMP vs. CLI Counters	64
4.6	RMON: Remote Monitoring	65
5	Monitoraggio del flusso	67
5.1	Real-Time Flow Measurement (RTFM)	67
5.1.1	Cosa è un flusso di rete?	68
5.1.2	NetFlow Architecture	69
5.1.3	Nozioni di base su Cisco NetFlow	70
5.1.4	Principi di NetFlow v9	72
5.1.5	IPFIX	72
5.1.6	Aggregazione e filtraggio	73
5.1.7	Flussi e sicurezza	74
6	Metriche e rilevamento delle anomalie	75
6.1	Metriche	75
6.1.1	Benchmarking	75
6.2	Rilevamento delle anomalie	78
6.2.1	Medie e previsioni	79
6.2.2	Misurare una deviazione	81

III	Altro	82
7	RRDtool as a Communication	83
7.1	Introduzione	83
7.2	RRD	83
8	nDPI	86
8.1	Introduzione	86
8.2	Classificare il traffico di rete	87
8.2.1	Welcome to nDPI	88

Parte I

Teoria

Capitolo 1

Introduzione

“Without data you’re just another person with an opinion.”
- W. Edward Deming, Data Scientist

Senza misure non è possibile produrre i risultati attesi, misurare i miglioramenti e quantificare il successo. Il traffico di rete non fa eccezione a questa regola.

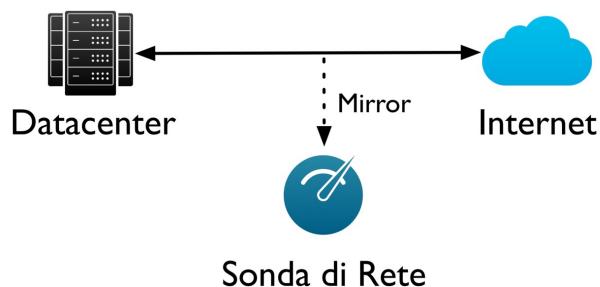
“If you can’t measure it, you can’t improve it.”
(Lord Kelvin, 1824 – 1907)

“If you can’t measure it, you can’t manage it.”
(Peter Drucker, 1909 – 2005)

Come si misura il traffico di rete

Individuare un punto della rete dove passa il traffico da analizzare (tipicamente vicino al router) e installarvi una **sonda di rete** capace di analizzare tale traffico:

- **Passiva:** solo analisi, no modifica/blocco traffico.
- **Attiva:** il traffico attraversa la sonda che può bloccarlo se necessario.



Che misure di rete possiamo fare?

Analisi **quantitativa** di traffico

- Top Talkers (Senders and Receivers).
- Destinazioni verso/da le quali viene scambiato traffico.
- Protocolli applicativi utilizzati (Skype, HTTP, Email).
- Rendicontazione traffico per host.

Analisi **qualitativa** di traffico

- Utilizzo di traffico non permesso (es. Tor o VPN anonime).
- Identificazione errori ed anomalie di rete che possono causare malfunzionamenti nell'utilizzo dei servizi di rete.

Gestione di Rete

Dal punto di vista del monitoraggio una rete è divisa in due grandi categorie:

- **Element management:** gestione dei device e delle periferiche.
- **Analisi dei dati:** analisi del traffico di rete.

1.1 Motivazioni

Situazione attuale:

- le “informazioni” acquisiscono un crescente significato come risorse strategiche.
- una rete di computer non è più solo un elemento di supporto in un’impresa, ma assume sempre più frequentemente una posizione chiave.
- il numero di computer interconnessi è aumentato notevolmente negli ultimi anni. Questo processo probabilmente continuerà a persistere.
- la complessità e la funzionalità dei componenti cresce in corrispondenza delle prestazioni dell’hardware disponibile.

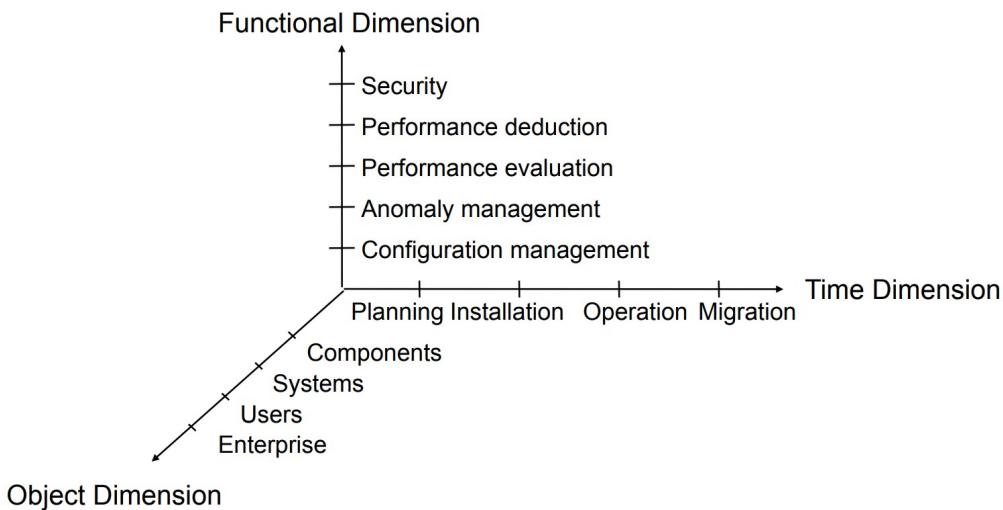
Richiesta:

- Disponibilità permanente dei servizi di rete con qualità ottimale.
- Riduzione dei costi per l’infrastruttura di rete dell’azienda.

Necessità: gestione assistita da computer di reti eterogenee.

1.2 Network Management Dimensions

L'element management si divide in tre parti: la **dimensione funzionale**, la **dimensione del tempo** e la **dimensione dell'oggetto**.



La *dimensione funzionale* fornisce informazioni sulle funzionalità svolte da un determinato componente. Gli aspetti di un componente di rete che devono essere considerati sono:

- **Gestione della configurazione:** capacità di impostare delle opzioni di funzionamento di un oggetto. Deve essere resa disponibile a chi vi interagisce: è necessaria ogni volta che c'è un problema.
- **Gestione di un'anomalia:** definire una base line, ossia capire il comportamento medio della rete, e controllare se il comportamento è stabile nel tempo. Ci sono due strade (di solito perseguitate contemporaneamente) per individuare le anomalie: andare a *definire dei paletti* oltre i quali le attività sono considerate errate o andare a *definire delle metriche* per osservare quanto il comportamento atteso si discosta dal comportamento attuale. L'anomalia deve essere messa in relazione anche a fattori esterni come il numero di utenti che usano la rete, il periodo/stagione.
- **Valutazione della performance:** è una misura utile all'operatore per riuscire a capire se la rete funziona correttamente.
- **Deduzione della performance:** riuscire a creare un'aspettativa rispetto al futuro. Conoscendo la rete e conoscendo il comportamento degli utenti è possibile prevedere le performance future.

- **Sicurezza:** far sì che la rete funzioni in maniera sicura. Se siamo un operatore non ci possiamo chiedere se tutti gli utenti utilizzino protocolli non sicuri, ma ci dobbiamo occupare di quei casi che creano problemi a tutti.

La *dimensione del tempo* è necessaria a chi costruisce la rete per prevedere le operazioni future che vi verranno fatte.

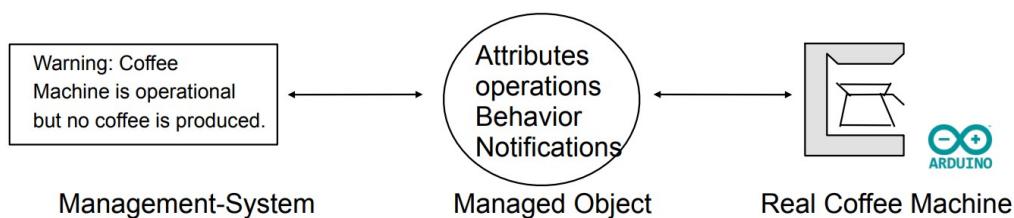
La *dimensione dell'oggetto* ha come obiettivo il bilanciamento della rete per garantire il funzionamento ottimale.

1.3 Terminologia e concetti fondamentali

Il controllo, il coordinamento e il monitoraggio delle risorse avviene tramite la manipolazione dai cosiddetti **managed objects**:

“Un managed object è la vista astratta di una risorsa che presenta le sue proprietà come viste dalla (e ai fini della) gestione.” (ISO 7498-4)

I managed objects sono una rappresentazione astratta di una risorsa reale. Il confine di un managed object specifica quali dettagli sono accessibili a un sistema di gestione e quali sono schermati (black box).



I managed objects non corrispondono necessariamente agli oggetti, come si sa dalla programmazione orientata agli oggetti. Le variabili semplici corrispondono ai MO nella gestione di Internet.

1.3.1 Managed Objects

Gli **attributi** descrivono lo stato, la condizione, degli oggetti gestiti: possono cambiare quando cambia la condizione dell'oggetto reale e possono essere manipolati mediante **operazioni** di gestione. Quest'ultime hanno lo scopo di rendere possibile l'accesso a un oggetto gestito e tipicamente sono operazioni di **get**, **set**, **create** e **delete**. Si noti che il numero e il tipo di operazioni influenzano le prestazioni e la complessità dell'oggetto.

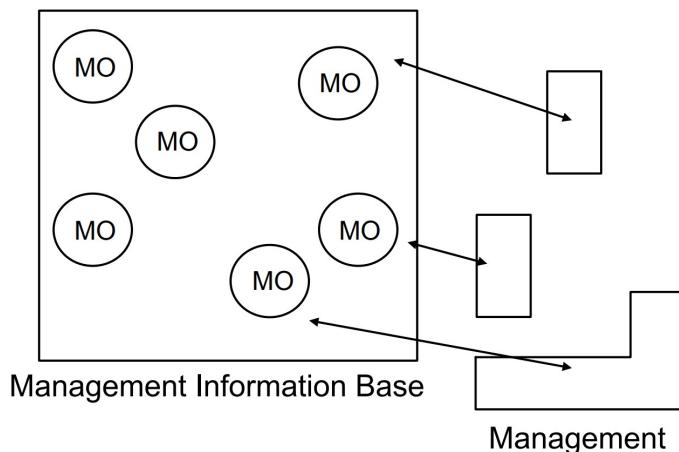
Il **comportamento** dei managed objects è normalmente definito in un inglese semplice e determina la semantica e l'interazione con la risorsa reale.

Inoltre vengono definite delle **notifiche**, messaggi che possono essere generati da un managed object quando si verificano situazioni specifiche.

1.3.2 Management Information Base (MIB)

L'unione di tutti i managed objects contenuti in un sistema costituisce il **Management Information Base (MIB)** del sistema:

“L'insieme di managed objects all'interno di un sistema, insieme ai loro attributi, costituisce il management information base di quel sistema.”
(ISO 7498-4)



Un MIB dovrebbe essere noto sia all'implementatore che al manager.

MIB Modularity

I managed objects di un sistema sono generalmente definiti in più definizioni MIB utilizzando un linguaggio di specifica. I moduli sono stati introdotti nei MIB per consentire la modularità del design, infatti diversi moduli possono essere definiti da diversi team e la funzionalità di gestione può essere gradualmente estesa e modificata (anche mediante MIB proprietari). Sistemi diversi possono supportare diversi moduli/versioni MIB.

1.3.3 Manager/Agent Paradigm

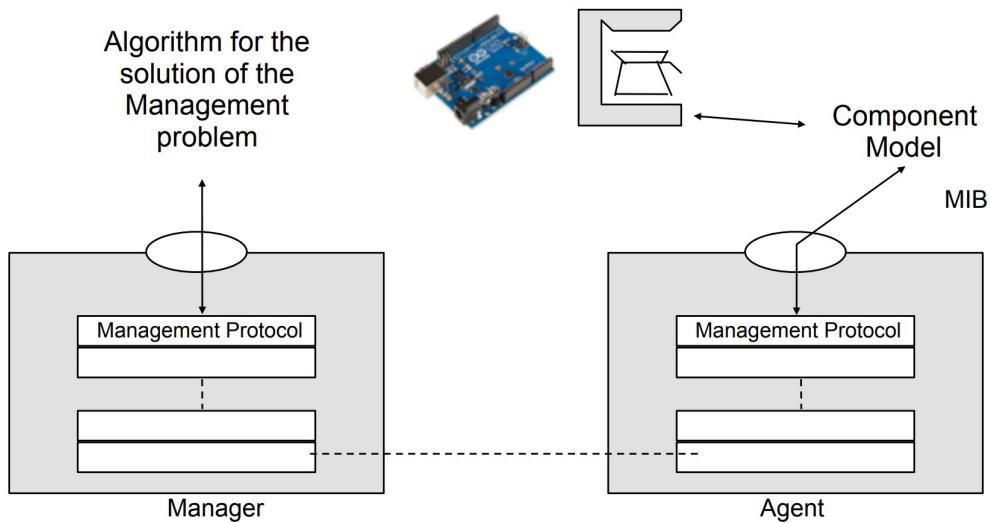
L'agent implementa i MOs MIB accedendo alle risorse reali ed elabora le richieste di un manager al quale trasmette le risposte appropriate. Quindi

protegge i MOs da accessi non autorizzati utilizzando le regole di controllo dell'accesso e l'autenticazione della comunicazione con il partner ed invia notifiche su importanti cambiamenti di stato nel MIB.

Il manager, invece, è colui che esercita il controllo (controlla le funzioni) ed è quindi l'istanza cruciale: può avviare operazioni di gestione mediante appropriate operazioni di protocollo per la manipolazione di MO. Il suo compito è ricevere messaggi dagli agenti e trasmetterli (per la gestione) alle applicazioni appropriate.

Management Protocol

Le applicazioni di gestione e i MO spesso non sono sullo stesso nodo: l'agent gira all'interno della risorsa, parla con la parte fisica di quest'ultima e rimane in ascolto in attesa di richieste; il manager invece sta fuori dalla risorsa (sta all'interno del manager di sistema) e invia i comandi all'agent. I manager identificheranno gli agenti e la risorsa in base all'indirizzo IP e la porta. Un **management protocol** implementa l'accesso agli oggetti gestiti a distanza codificando i dati di gestione che vengono poi protetti durante il trasferimento.



1.3.4 Functional Areas (FCAPS)

Le applicazioni gestionali possono essere suddivise in cinque aree funzionali:

- **Fault manager:** rilevamento, isolamento e riparazione degli errori.

- **Configuration manager:** produzione e amministrazione delle informazioni di configurazione, amministrazione dei nomi, avvio, verifica e cessazione dei servizi.
- **Account management:** inserimento dei dati di consumo (utilizzo), distribuzione e monitoraggio dei contingenti, fatturazione cliente per consumo di risorse.
- **Performance management:** raccolta dati statistici, determinazione delle prestazioni del sistema, modifiche ai sistemi per aumentare l'efficienza.
- **Security management:** produzione e verifica delle policy di sicurezza, generazione e distribuzione di password e account, report e analisi di eventi rilevanti per la sicurezza.

Queste cinque aree funzionali sono identificate tramite l'acronimo FCAPS. Le aree non sono reciprocamente indipendenti (la misurazione dei dati ha spesso un impatto sulla configurazione del sistema). Le funzioni di base (ad esempio il monitoraggio di un contatore per i valori di soglia) risiedono spesso in diverse aree funzionali.

1.3.5 Management Architectures Overview

La struttura delle informazioni di gestione definisce le regole della descrizione dei Managed Objects:

- Identificazione e designazione di MOs.
- Composizione dei MO.
- Comportamento dei MO.
- Rapporti con altri MO.
- Possibili operazioni e messaggi interni dei MO.

Quindi, contiene la definizione dei tipi di dati, della struttura e della sintassi per la descrizione dei MO. La quantità delle descrizioni dei MO in conformità a queste regole definisce il *Management Information Base* (MIB).

I protocolli e servizi di gestione definiscono i servizi e abilitano l'accesso ai MO remoti. Possono essere utilizzati diversi protocolli per l'implementare i servizi definiti: la primitiva del servizio e le operazioni del protocollo appropriato influenzano notevolmente l'efficienza e la complessità del sistema di gestione.

1.3.6 Servizi e protocolli: alcune definizioni

Servizio: definito come una funzione astratta fornita da una rete.

Primitiva di servizio: le singole funzioni elementari sono chiamate primitive di servizio. I servizi ISO/OSI tipici sono:

request	richiesta di servizio
indication	indicazione che è stato richiesto un servizio
response	reazione del servizio a una richiesta di servizio
confirm	conferma che è stato fornito un servizio richiesto

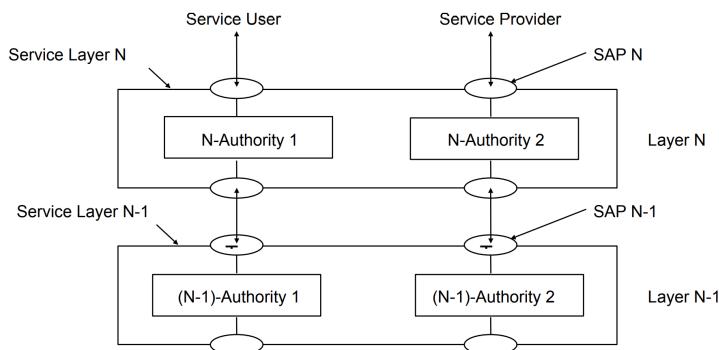
Service Access Point (SAP): l'interfaccia su cui la primitiva del servizio può accedere come punto di accesso al servizio.

Entità: i servizi forniti dalle cosiddette istanze.

Protocollo: le regole e le restrizioni in base alle quali le istanze interagiscono con altre istanze.

1.3.7 Rappresentazione e stratificazione dei servizi

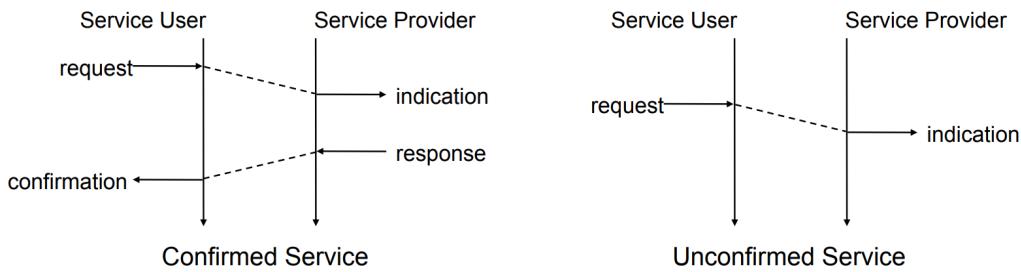
La definizione degli strati è un principio fondamentale per la strutturazione dei sistemi di comunicazione. I servizi di uno strato possono accettare solo primitive di servizi di strati adiacenti.



1.3.8 Time Diagrams

I diagrammi temporali chiariscono le connessioni temporali e spaziali tra le primitive di servizio.

I servizi si suddividono in due gruppi, *confermati* e *non confermati*. Un **servizio confermato** è tale se a fronte di una richiesta manda una risposta contenente l'esito dell'esecuzione della primitiva. Un servizio è **non confermato** se a fronte di una richiesta non viene mandata nessuna risposta.

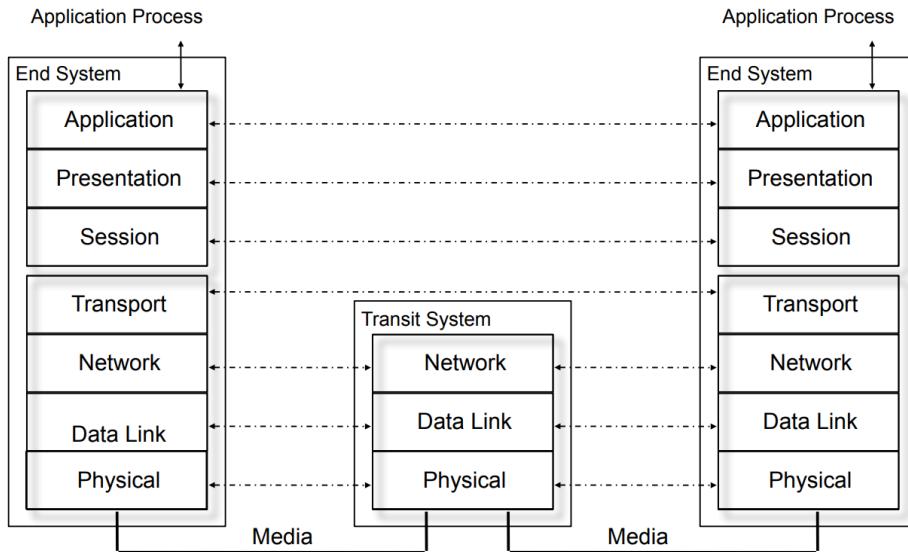


L’asse verticale è l’asse del tempo, l’asse orizzontale fornisce la distanza spaziale tra utenti e fornitori di servizi. Le richieste di servizio di un servizio confermato possono comportare una conferma positiva o negativa. Le richieste di servizio di un servizio non confermato non vengono riconosciute.

In entrambi i casi ciò che il mittente invia, spesso, non è uguale a ciò che il destinatario riceve. Si consideri il caso di messaggistica: quando il mittente scrive il messaggio specifica il numero del destinatario; quest’ultimo quando riceve il messaggio non visualizza il proprio numero, bensì quello del mittente (oltre ad una serie di informazione aggiuntive come ad esempio la data). Alcuni protocolli come `http` possono infatti modificare il payload del messaggio, soprattutto con l’utilizzo dei proxy `http`; altri protocolli come `smtp` invece estendono gli header dei messaggi. In linea generale tutti i protocolli di tipo “store-and-forward” eseguono delle alterazioni al messaggio, per questa ragione nei time diagrams “`request`” e “`indication`” assumono connotati diversi.

Si noti che quando due entità devono comunicare devono usare un meccanismo di serializzazione/codifica in modo tale che le richieste e le risposte siano riconosciute da entrambe le parti. Nel mondo delle reti è necessario mantenere il minor numero di informazioni aggiuntive al dato, così da sprecare meno spazio possibile per la codifica/decodifica dei dati (per questa ragione spesso vengono mandate informazioni in formato compresso).

1.3.9 ISO/OSI-Reference Model



La pila ISO/OSI presenta sette livelli, i primi tre vengono definiti “*livelli di presentation*” e sono utili per lo scambio corretto dei messaggi e per la gestione del loro formato tra applicazioni diverse. I dispositivi intermedi di rete presentano solo gli ultimi tre livelli della pila ISO/OSI. La rete in questo caso deve essere a conoscenza di tutto (anche dei formati diversi inviati sulla rete stessa).

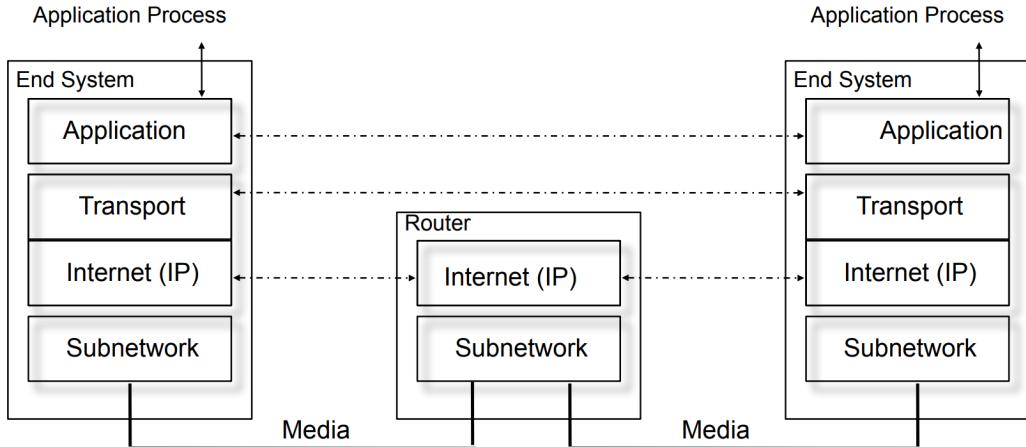
ISO/OSI Higher Layers	
Session Layer	Sincronizzazione e coordinamento dei processi comunicativi. Controllo della sessione (checkpoint per il ripristino).
Presentation Layer	Trasformazione e adattamento delle presentazioni dei dati (es ASCII EBCDIC). Serializzazione delle strutture dati ai fini del trasferimento. Compressione dati.
Application Layer	Fornitura di servizi fondamentali, utilizzabili direttamente da qualsiasi applicazione inclusi (ma non limitati a): trasferimento di file, terminali virtuali, amministrazione dello spazio dei nomi, accesso al database, gestione della rete, reti di comunicazione elettronica, controllo del processo e della stampa...

<i>ISO/OSI Transport System</i>	
Physical Layer	Trasporto di un flusso di bit su un supporto. Trasporto a seconda delle caratteristiche del supporto utilizzato. Rappresentazione dei valori 0 e 1 (es. livelli di tensione). Sincronizzazione tra mittenti e destinatari. Definizione di spine standard per l'interconnessione dei media.
Data Link Layer	Trasporto di un frame di bit. Comunicazione dati tra sistemi che condividono un media comune. Rilevamento e ripristino degli errori di trasferimento. Controllo del flusso per gestire i picchi di traffico (congestione). Implementazione solitamente in hardware su schede adattatrici (es. Scheda Ethernet).
Network Layer	Determinazione di un percorso attraverso la rete (routing). Multiplex di connessioni di rete su una connessione condivisa. Rilevamento e ripristino degli errori tra i sistemi finali. Controllo del flusso tra sistemi finali. Divisione di un pacchetto in più frame.
Transport Layer	Comunicazione end-to-end tra le applicazioni. Connessioni virtuali su servizi di datagramma senza connessione. spesso Rilevamento e ripristino degli errori tra le applicazioni. Controllo del flusso tra le applicazioni. Utilizzo simultaneo di più servizi.

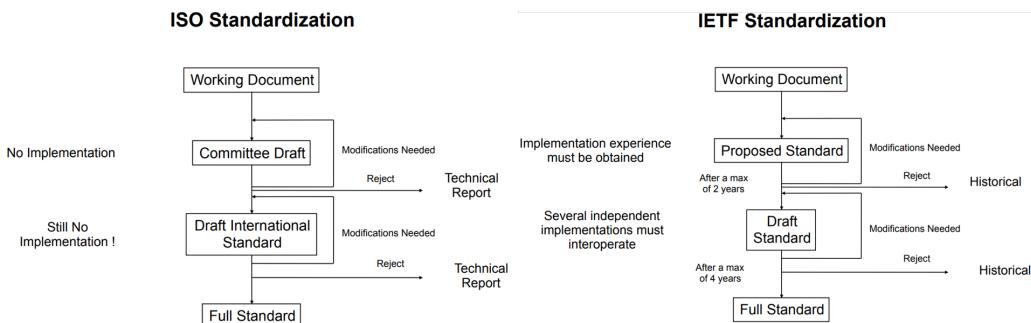
1.3.10 Internet Layer Model

Nel mondo Internet sono presenti molti meno livelli nella pila dei protocolli, questo perché molti problemi non vengono più demandati alla rete. Se si considera il caso di scambio di file in diversi formati, tali informazioni non necessitano di essere presenti sulla rete come nel caso del modello ISO/OSI, ma sono gestite a livello applicativo: solo le applicazioni necessitano di essere

compatibili tra di loro, non l'intera parte sottostante. Il modello di Internet a livello di gestione è molto più semplice del modello ISO/OSI, questo è possibile notarlo anche nell'ambito della standardizzazione.



1.3.11 Standardization

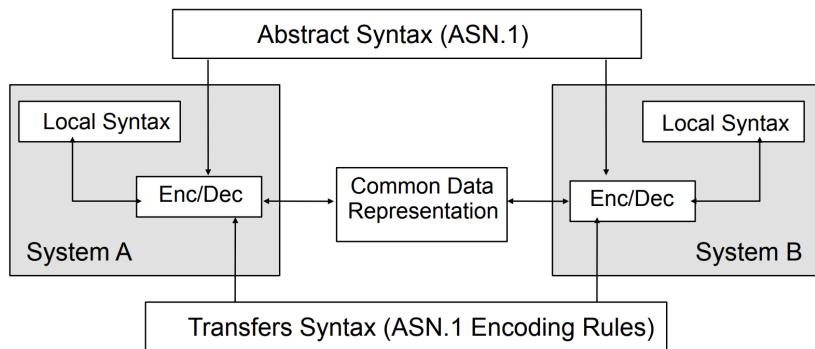


Nella **ISO standardization** si organizzano un insieme di gruppi di lavoro e si instaura un comitato di persone che discutono del problema da risolvere e dove sporadicamente si eseguono modifiche e si scrivono rapporti. Se ciò che si vuole fare non è una buona idea, è possibile eseguire un *technical report*: un documento tecnico dove si fa un riassunto delle cose fatte che però sono andate male. Quando il comitato si è messo d'accordo, si eseguono i vari draft dello standard all'interno dei quali di passa all'approccio pratico del problema. In tutte le fasi della standardizzazione non c'è implementazione ed è quindi possibile che si possano presentare ancora dei problemi una volta arrivati alle fasi finali dello standard. Nel corso del tempo verranno poi rilasciate delle patch/fix per andare a risolvere tutti i problemi che ancora non erano stati presi in considerazione.

Nell'**IETF standardization** l'organizzazione dei processi di standardizzazione è molto diversa: innanzitutto si fornisce un timeout (historical) entro il quale l'idea deve essere proseguire, inoltre se il proposed standard va avanti, l'implementazione presentata inizia ad essere testata su più piattaforme (vengono realizzate più implementazioni). Nella IETF standardization quindi, quando si arriva a parlare di full standard quest'ultimo è già stato testato più volte (al contrario di come avviene nella standardizzazione OSI).

1.4 Overview: Abstract Syntax Notation One

Abstract Syntax Notation One (ASN.1) è una sintassi per la definizione di strutture dati e formati di messaggi. Gli obiettivi dell'ASN.1 sono lo scambio di informazioni tra macchine con architetture hardware differenti (8/16/32/64 bit, little/big-endian) e l'indipendenza dai linguaggi di programmazione esistenti (linguaggio neutro). Inoltre, la codifica dei dati durante il trasferimento dovrebbe essere selezionabile tra mittenti e destinatari (negoziazione). In sostanza, separa la presentazione dei dati dalla rappresentazione della struttura dei dati specifica dell'applicazione: la sintassi astratta definisce le strutture dati durante il trasferimento e determina in quale forma queste strutture dati verranno trasferite in serie su una rete.



ASN.1 definisce una sintassi astratta standardizzata; consente diverse regole di codifica che trasformano la sintassi astratta in un flusso di byte adatto al trasferimento. **BER (Basic Encoding Rules)** definisce la mappatura tra sintassi astratta e di trasferimento.

Le applicazioni normalmente utilizzano una sintassi locale a seconda del linguaggio di programmazione utilizzato.

1.4.1 Primitive ASN.1 Datatypes

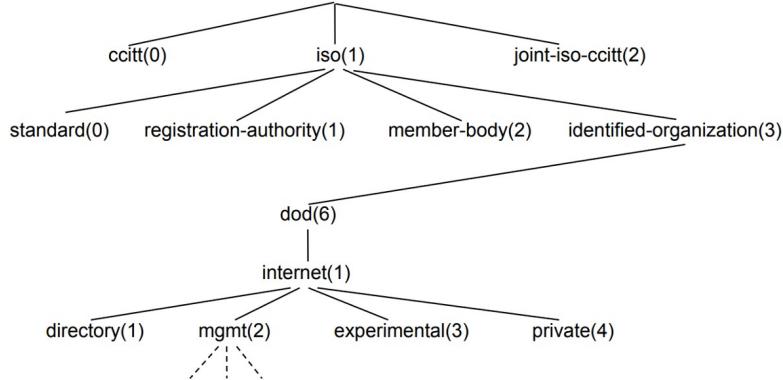
I nomi dei tipi di dati ASN.1 iniziano con una lettera maiuscola, mentre i nomi dei valori ASN.1 (costanti) iniziano con una lettera minuscola; le parole chiave ASN.1 e i nomi delle macro sono composti solo da lettere maiuscole. I commenti sono racchiusi tra “-”.

- **BOOLEAN**: può assumere solo i valori predefiniti TRUE e FALSE.
- **INTEGER**: copre tutti i possibili numeri interi. Nessuna delimitazione dell’intervallo di numeri.
- **BIT STRING**: una sequenza di bit. La lunghezza non deve essere divisibile per 8.
- **OCTET STRING**: una sequenza di ottetti (byte). È il tipo di base per diversi set di caratteri e altri tipi derivati (**GeneralizedTime**, **UTCTime**).
- **ENUMERATED**: tipo di enumerazione. I valori possibili devono essere determinati dalla definizione dei tipi di dati derivati.
- **OBJECT IDENTIFIER**: identificazione univoca di un nodo nell’albero di registrazione ISO. Percorso della radice dell’albero al nodo di destinazione.
- **ObjectDescriptor**: una stringa di caratteri per l’identificazione di un nodo nell’albero di registrazione. Non necessariamente unico.
- **ANY**: qualsiasi tipo di dati ASN.1 (unione di tutti i tipi di dati ASN.1, come “void” in C).
- **EXTERNAL**: dati non descritti utilizzando una definizione ASN.1.
- **NULL**: un simbolo sostitutivo, per indicare in un tipo di dati assemblato l’assenza di un valore.

ISO Registration Tree

Quando si trasferiscono dei dati da un client ad un server, chi fa la richiesta e chi la riceve deve poter comprendere in modo univoco e non ambiguo ciò che gli viene comunicato; ciò è possibile grazie all’ISO Registration Tree: viene utilizzato per identificare in modo univoco definizioni, documenti, oggetti, ... Possiede una struttura gerarchica simile ai file system gerarchici, tutti i nodi di un livello identificati da un numero univoco. Il percorso dalla radice

del Registration Tree a un nodo è una sequenza numerica divisa da punti chiamata **Object Identifier** (ad esempio 1.3.6.1).



1.4.2 Assembled ASN.1 Datatypes

A partire dai tipi di base, si possono creare delle composizioni (l'equivalente delle struct in C).

- **SEQUENCE**: corrisponde alle strutture in C o ai record in Pascal. La sequenza degli elementi in una **SEQUENCE** è fissa.
- **SET**: simile a **SEQUENCE**, con la differenza che la sequenza degli elementi non è specificata.
- **SEQUENCE OF**: quantità ordinata (elenco) di dati omogenei.
- **SET OF**: quantità non ordinata di dati omogenei.
- **CHOICE**: tipo di selezione, simile all'union in C.
- **REAL**: consiste nel tipo di dati **INTEGER** esteso con mantissa ed esponente.

1.4.3 Reduced Datatypes

Definizione di ulteriori tipi di dati restringendo l'ambito dei tipi di dati esistenti. La sintassi esatta dipende dal tipo di dati primitivo sottostante.

```

LottoNumber ::= INTEGER (1...90)
MD5Key      ::= OCTET STRING (SIZE (16))
IPAddress   ::= OCTET STRING (SIZE (4|16))
Counter32    ::= INTEGER (0...4294967295)
Integer32    ::= INTEGER (-2147483648...2147483647)
Unsigned64   ::= INTEGER (0...18446744073709551615)

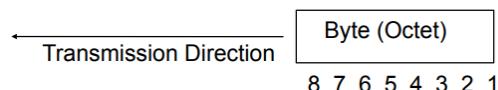
```

Le limitazioni dell'ambito vengono applicate ai tipi di dati derivati (ad esempio `SEQUENCE OF MD5Key`). La restrizione del tipo di dati `INTEGER` ha senso in quanto i computer odierni normalmente operano internamente con numeri a 32 o 64 bit.

1.4.4 Basic Encoding Rules (BER)

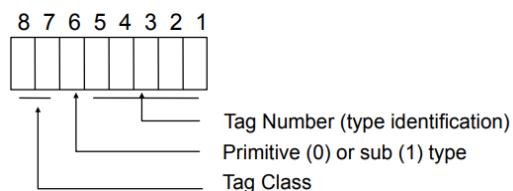
Le regole di codifica di base determinano come un tipo di dati ASN.1 può essere rappresentato come una stringa di byte e dove ciascuna variabile è identificata da un tag, la lunghezza del valore in byte e il valore di quei byte. Tale codifica consente a un destinatario di ricostruire il tipo di un messaggio dal flusso di byte ricevuto.

La codifica BER è un po' inefficiente poiché spesso ci sono informazioni non necessarie da trasferire; l'uso dei campi `OPTIONAL` ne ha ulteriormente complicato la definizione. Definisce anche la direzione di trasmissione del flusso di bit diversa dalla codifica dei tipi di dati ASN.1:



Coding Tags Classes

Ogni tag è codificato in un byte:

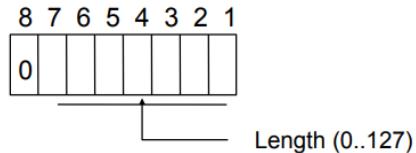


Tag classes	
	Bit 8 Bit 7
UNIVERSAL	0 0
APPLICATION	0 1
CONTEXT-SPECIFIC	1 0
PRIVATE	1 1

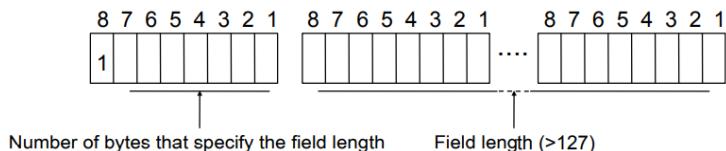
Coding Field Length

Il campo della lunghezza indica la lunghezza del valore immediatamente successivo.

- Lunghezza compresa tra 0...127:



- Lunghezza > 127:



Value Coding

Per ogni primitivo tipo ASN.1 esiste una regola che consente di tradurre i valori in un flusso di byte e viceversa.

Le regole per INTEGER e OCTET STRING sono semplici, mentre le regole per OBJECT IDENTIFIER sono relativamente complesse. I valori assemblati (SEQUENCE, SEQUENCE OF) sono facilmente rappresentabili codificando ogni singolo elemento. Con i costrutti CHOICE viene trasferito solo il valore disponibile, quindi il tag associato deve essere univoco.

La lunghezza della codifica BER deve essere ben nota (nessun valore fittizio) quando viene codificato un valore. Con alcune limitazioni è anche possibile specificare la lunghezza dopo il valore.

La decodifica è più difficile quando la lunghezza è specificata dopo il valore.

30 1B	SEQUENCE, Length 27
02 01 00	INTEGER, Length 1, "0"
04 06 70 75 62 6C 69 63	OCTET STRING, Length 6, "public"
A1 0E	GetNextRequest-PDU, Length 14
02 04 36 A2 8F 07	INTEGER, Length 4, "916623111"
02 01 00	INTEGER, Length 1, "0"
02 01 00	INTEGER, Length 1, "0"
30 00	SEQUENCE OF, Length 0

La codifica dei valori primitivi non è sempre così semplice come nell'esempio (alcuni tipi di dati possono essere codificati sia in forma breve che lunga).

Capitolo 2

Internet Management

2.1 Overview

Per gestire una rete, il suo traffico o la sua sicurezza è necessario prima di tutto che la rete funzioni. È quindi necessario un **metodo di comunicazione fra componenti di rete** per sapere se va tutto bene oppure no.

1987	Simple Gateway Monitoring Protocol (SGMP)	
1987	High-level Entity Management System (HEMS)	
1988	Simple Network Management Protocol	proposed
1990	Simple Network Management Protocol	standard 15, 16
1991	Management Information Base II	standard 17
1993	SNMP Version 2 (Party/Party/Context)	historical
1996	SNMP Version 2 (Communities)	draft
1998	SNMP Version 3 (User-based)	draft

Quando furono creati i primi protocolli per la **gestione di Internet** vennero prese le idee base di ICMP: creare un **set d'istruzioni minimo per capire se la rete stesse funzionando ed in caso negativo cosa stesse andando male**. La caratteristica fondamentale di SNMP è l'essere **semplice**. Tale semplicità ha permesso il supporto del protocollo su apparecchi di rete con un costo ragionevole: “*gli switch di fascia medio-bassa possono essere gestiti tutti con SNMP*”.

SNMPv1 ha una grande diffusione soprattutto nella comunicazione dei dati. I tentativi di standardizzazione di SNMPv2 sono falliti. SNMPv3 con SNMPv1 è stato accettato da un'ampia comunità di produttori di reti. La comunità degli utenti ha accettato molto bene SNMPv3 in termini di supporto e sviluppo.

SNMP Development Goals

- Minimizzazione del numero e della complessità delle funzioni di gestione che sono implementate da un agente:
 - Riduzione dei costi di sviluppo per agenti gestionali (applicazioni semplici).
 - Ubiquity: utilizza la stessa tecnologia di gestione per tutti i dispositivi (stampanti o Cray).
 - Estendibilità dell'applicazione: sviluppo di nuove funzioni di gestione senza la necessità di modificare gli agenti.
- Estensibilità definendo nuovi MIB.
- Indipendenza da architetture di computer o di rete esistenti.
- Robustezza grazie a un semplice servizio di trasporto senza connessione (UDP).
- Nessuna dipendenza da altri servizi di rete.
- L'aggiunta della gestione a dispositivi/applicazioni nuovi/esistenti dovrebbe essere poco costosa, semplice da sviluppare e con funzionalità limitate.

Purtroppo alcuni di questi obiettivi originari sono andati persi: il termine “semplice” si riferisce al protocollo e non alle specifiche o all’implementazione di applicazioni gestionali.

Trap Directed Polling

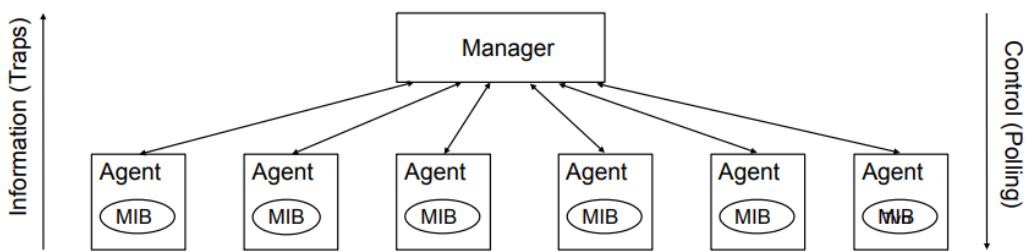
Il manager **esercita il controllo** su vari Agent eseguendo richieste continue (**polling**): poiché viene usato UDP per mandare messaggi, non è possibile conoscere gli stati degli Agent in ogni momento e quindi è necessario richiederli periodicamente; usando TCP ciò non sarebbe necessario perché la connessione persistente sarebbe sufficiente per conoscere lo stato dell’agente.

Si ipotizzi che l’intervallo di tempo che passa fra un poll ed un altro sia di cinque minuti: **il Manager è cieco nei confronti dell’Agent a intervalli di cinque minuti**, ovvero assume che l’agent rimanga attivo/nello stesso stato per i cinque minuti successivi alla risposta. Questo non è sempre vero: l’Agent potrebbe “rompersi” subito dopo aver risposto al poll.

Per risolvere questo problema gli Agent effettuano **trap** al Manager informandolo quando avviene un **cambio di stato importante**. Tuttavia l’Agent

è in grado di **effettuare una trap se l'evento non è catastrofico** perciò il polling periodico è comunque necessario. Questo periodo di tempo varia a seconda dell'affidabilità dell'Agent con cui il Manager ha a che fare: **un Manager scritto bene riporta un concetto di fiducia** al suo interno, ovvero tenderà ad effettuare polling più spesso verso quegli Agent che si sono dimostrati meno affidabili.

In sintesi, i managers SNMP interrogano a intervalli regolari gli agenti SNMP: gli agents possono segnalare casi eccezionali a un manager inviando una trap; il manager SNMP può adattare la strategia di polling alla ricezione di trap (polling diretto da trap).



SNMP è un modello rigorosamente centralizzato, in cui il manager implementa l'intera funzionalità e responsabilità.

SNMP Application Areas

SNMP non è utilizzato solo per la gestione della rete, ma anche per il controllo e monitoraggio dei processi produttivi e dei sistemi informatici complessi e per il monitoraggio di programmi applicativi complessi (database relazionali, componenti SAP R/3, ...).

Sul mercato sono disponibili molti toolkit SNMP buoni, ma sono disponibili pochissime applicazioni per risolvere problemi di gestione complessi: l'implementazione di applicazioni speciali, o la conversione delle linee guida della procedura locale, in genere è relativamente complessa e costosa.

2.2 Structure of Management Information (SMIv2)

L'attuale modello di informazioni noto come “*Structure of Management Information version 2*” (SMIv2) è definito e si basa su semplici variabili tipizzate, in particolar modo si basa sul sottoinsieme esteso di ASN.1 (1998). Ogni variabile ha un tipo di dati ASN.1 primitivo, non assemblato:

- INTEGER, OCTET STRING, OBJECT IDENTIFIER, NULL
- Integer32, Unsigned32, Gauge32, Counter32, Counter64, IpAddress, TimeTicks, Opaque

Non implementa strutture dati complesse e operazioni sulle variabili.

Le variabili sono **scalari** (esattamente un'istanza) o colonne in una **tavola** bidimensionale “concettuale” (zero o più variabili) e su di esse possono essere applicate solo le operazioni di “read” e “write”. Tuttavia il protocollo SNMP consente la manipolazione di elenchi di variabili.

I MIB (Management Information Bases) SMIv2 vengono definiti utilizzando speciali macro ASN.1. Sfrutta la complessità delle nuove definizioni di MIB: definizione di funzionalità di base e di tipi primitivi da utilizzare nei nuovi MIB.

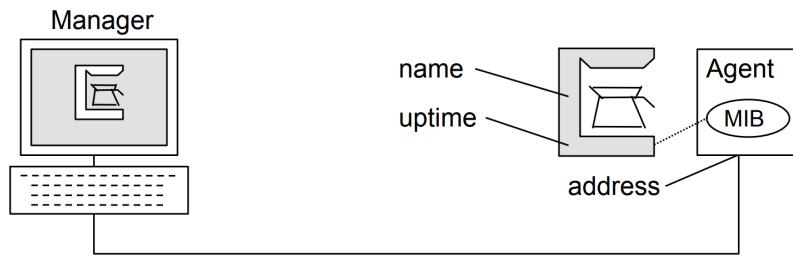
SMIv2	SMIv1	Description
INTEGER	INTEGER	Integer Numbers (-2147483648..2147483647)
OCTET STRING	OCTET STRING	Sequence of bytes (octets).
OBJECT IDENTIFIER	OBJECT IDENTIFIER	Unique identifier.
Integer32	INTEGER	32 bit Integers (-2147483648..2147483647)
Unsigned32	-	32 bit Positive Integers (0..4294967295)
Gauge32	Gauge	“Thermometer” Integer (0..4294967295)
Counter32	Counter	32 bit non decreasing counter (0..4294967295)
Counter64	-	64 bit non decreasing counter (0..18446744073709551615)
TimeTicks	TimeTicks	Time in 1/100th of seconds
IpAddress	IpAddress	4 Byte IPv4 Address
Opaque	Opaque	Unspecified ASN.1 Type (not recommended)
BITS	-	Bits in a OCTET STRING
-	NetworkAddress	Network Address (not recommended)

SMIv2 Basic Datatypes (RFC 2578)

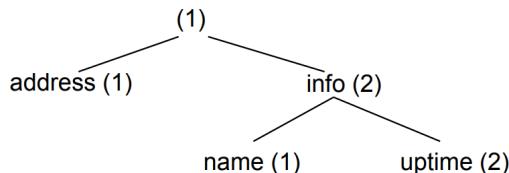
Un'altra caratteristica di SMI è che il tipo intero non si usa moltissimo, si preferisce utilizzare **gauge** e **counter**: sono entrambi interi ma possiedono una semantica ben precisa. Un **gauge** è un numero che rappresenta una quantità che va da un minimo ad un massimo. La temperatura, ad esempio, è un gauge (non bisogna farci calcoli sopra e può aumentare rimanendo tra un minimo ed un massimo, quando lo si legge quindi si estrapola il valore definitivo). Un **counter** è un intero a 32 bit come **gauge**, ma ha delle proprietà diverse: il contatore, infatti, non decresce (o rimane del valore corrente o aumenta)

e il valore estrappolato da una variabile con questo tipo non è utilizzabile nell'immediato, ma lo si può usare per differenza (tachimetro dell'automobile e voglio sapere in un giorno quanti chilometri ho fatto).

A MIB Use Case



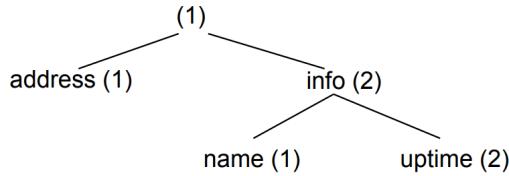
Definizione delle variabili nell'ISO Registration tree. I nodi vengono definiti a scopo di denominazione. Le foglie dell'albero rappresentano gli oggetti gestiti (cioè “la carne”). I nodi secondari possono essere utilizzati per organizzare logicamente i tipi di oggetto.



2.2.1 Object Identifier and Instance Identifier

Nel Registration tree ogni oggetto può essere identificato mediante un object identifier univoco. Gli sviluppi concreti (istanza) di un tipo di oggetto sono designati univocamente da un cosiddetto *identificatore di istanza*. Un identificatore di istanza univoco si ottiene allegando identificatori di istanza all'identificatore di oggetto.

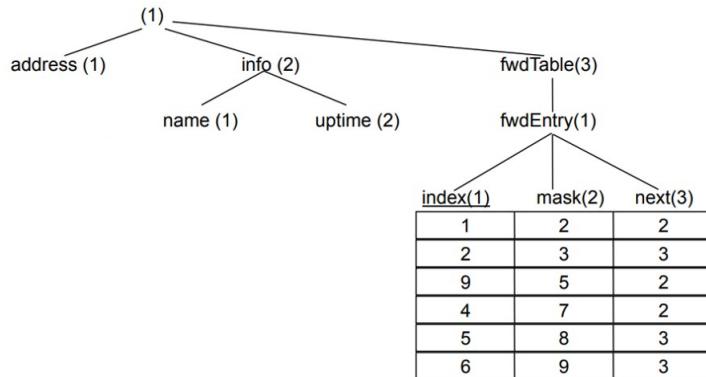
Gli oggetti scalari hanno fondamentalmente **una sola istanza** il cui identificatore ha valore 0 (e.g. `sysName.0`). Gli identificatori di istanza per le variabili non scalari derivano dalla denominazione univoca di una tabella concettuale. Poiché l'identificatore di oggetto può contenere fino a 128 elementi, i nomi delle istanze non possono essere infinitamente complessi.



Object Identifier	Instance Identifier	Type	Value
1.1	0	IpAddress	10.1.2.1
1.2.1	0	OCTET STRING	"FilterFresh"
1.2.2	0	TimeTicks	54321

I nomi dei nodi MIB sono rilevanti solo per gli utenti umani.

I descrittori devono essere univoci all'interno di un modulo MIB, sebbene possano essere utilizzati più volte in diversi moduli MIB (si ottengono descrittori univoci dalla combinazione dei nomi dei moduli e dei descrittori).



Le tabelle sono definite fondamentalmente con due "nodi ausiliari": il primo nodo definisce la tabella ed è di tipo **SEQUENCE OF**, mentre il secondo nodo definisce una voce (una riga) nella tabella ed è di tipo **SEQUENCE**; questo è l'unico uso consentito di **SEQUENCE** e **SEQUENCE OF** in SNMP SMIv2.

Il risultato della colonna e dell'identificatore di istanza (codice della tabella) è un identificatore di oggetto univoco per ciascuna voce di tabella.

Esempio di tabella (convenzione OID \Rightarrow valore):

1.3.1.1.1 \Rightarrow 1	1.3.1.3.1 \Rightarrow 2	1.3.1.2.4 \Rightarrow 7
1.3.1.2.1 \Rightarrow 2	1.3.1.1.4 \Rightarrow 4	1.3.1.2.7 \Rightarrow inesistente

Tables Naming

La denominazione delle tabelle è molto importante poiché influisce sul modo in cui si accede alle tabelle stesse. Ne esistono due tipi: uno usa numeri di riga (non utilizzati da SNMP), l'altro usa una colonna indice (il metodo usato da SNMP).

Un indice di tabella non è necessariamente un INTEGER, ad esempio la routing table utilizza un indirizzo IP come indice della tabella.

Routing Table		
destination(1)	policy(2)	next(3)
130.89.16.23	1	130.89.16.23
130.89.16.23	2	130.89.16.127
192.168.10.12	1	172.16.1.18
192.168.10.12	2	172.16.1.12

Un indice di tabella può essere composto da più componenti:

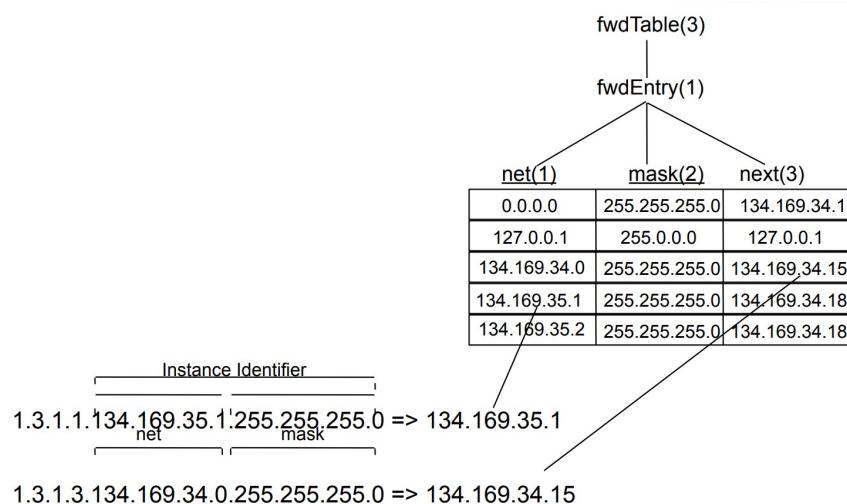
$$X . C . I_1 . I_2 \dots . I_n$$

X OID of the table: identifica la tabella

C Column number: identifica la colonna

$I_1 \dots I_n$ Index value (indice nella colonna)

Una tabella di routing IP è la combinazione di indirizzo IP e maschera di rete IP necessaria per soddisfare le regole di instradamento. I singoli byte dell'indirizzo IP vengono specificati come identificatori secondari individuali.



Rules for the Specification of Instance Identifier values

Valori per tipi fondamentali:

- Valori per `INTEGER`: un singolo valore intero.
- Valori per `OCTET STRING` di lunghezza fissa: ogni singolo byte viene trattato come un valore individuale.
- Valori per `OCTET STRING` di lunghezza variabile: il primo valore è la lunghezza, seguita da ogni singolo byte.
- Valori per `OBJECT IDENTIFIER`: il primo valore è la lunghezza, seguita da ogni singolo byte.

La parola chiave `IMPLIED` può essere utilizzata senza il byte di lunghezza se non porta ad ambiguità.

La lunghezza massima dei valori `OBJECT IDENTIFIER` è limitata a 128 elementi, quindi gli identificatori di istanza non saranno complessi arbitrari.

2.2.2 MIB Module

Tipi di oggetti simili vengono combinati in moduli MIB, ogni modulo MIB deve avere un nome univoco (lettere maiuscole). I moduli MIB sono (quasi) normali moduli ASN.1 e obbediscono alle regole lessicali ASN.1. Le definizioni possono essere importate da altri moduli MIB con l'aiuto dell'istruzione `IMPORT`; tutte le macro ASN.1 SMI utilizzate devono essere importate esplicitamente.

```
COFFEE-MIB DEFINITIONS ::= BEGIN

IMPORT      MODULE-IDENTITY, OBJECT-TYPE, enterprises,
            IpAddress, TimeTicks    FROM SNMPv2-SMI;
...
END
```

Module-Identities (RFC 2578)

```
<descriptor> MODULE-IDENTITY
    LAST-UPDATED <ExtUTCTime>
    ORGANIZATION <Text>
    CONTACT-INFO <Text>
    DESCRIPTION <Text>
```

```
[REVISION <ExtUTCTime>
DESCRIPTION <Text>]*
 ::= <ObjectIdentifier>
```

Definisce le informazioni amministrative, ad es. informazioni di contatto e numero di versione. Le clausole REVISION e DESCRIPTION non sono obbligatorie e possono ripetersi più volte.

```
IF-MIB DEFINITIONS ::= BEGIN
IMPORTS ...
ifMIB MODULE-IDENTITY
    LAST-UPDATED "9611031355Z"
    ORGANIZATION "IETF Interface MIB Working Group"
    CONTACT-INFO " Keith McCloghrie      408-526-5260
                    Cisco Systems, Inc.   kzm@cisco.com
                    170 West Tasman Drive
                    San Jose, CA 95134-1706, US"
    DESCRIPTION "The MIB module to of describe generic objects for network interface
                 sub-layers. This MIB is an updated version of MIB II's ifTable,
                 and incorporates the extensions defined in RFC 1229."
    REVISION "9602282155Z"
    DESCRIPTION "Revisions made by the Interfaces MIB WG"
    REVISION "9311082155Z"
    DESCRIPTION "Initial revision, published as part of RFC 1573."
 ::= { mib-2 31 }
...
END
```

Esempio di Module Identities

Object Identities (RFC 2578)

```
<descriptor> OBJECT-IDENTITY
    STATUS <Status>
    DESCRIPTION <Text>
    [REFERENCE <Text>]
 ::= <ObjectIdentifier>
```

Definisce e registra un valore dell'identificatore di oggetto: consente l'allocazione di qualsiasi nodo all'interno dell'albero di registrazione.

La clausola STATUS definisce se il nodo allocato è “obsoleto”, “corrente” o “deprecato”. Il REFERENCE opzionale viene utilizzato per fare riferimento a ulteriori informazioni (simile ai collegamenti ipertestuali HTML).

```

zeroDotZero      OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
        "A value used for null Identifiers."
    ::= { 0 0 }

snmpUDPDomain   OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
        "The SNMPv2 over UDP transport domain. The corresponding
         transport address is of type SnmpUDPAddress."
    ::= { snmpDomains 1 }

snmpIPXDomain   OBJECT-IDENTITY
    STATUS      current
    DESCRIPTION
        "The SNMPv2 over IPX transport domain. The corresponding
         transport address is of type SnmpIPXAddress."
    ::= { snmpDomains 5 }

```

Esempio di Object Identities

Object Types (RFC 2578)

```

<descriptor> OBJECT-TYPE
    SYNTAX <Syntax>
    [UNITS <Text>]
    MAX-ACCESS <Access>
    STATUS <Status>
    DESCRIPTION <Text>
    [REFERENCE <Text>]
    [INDEX <Index>]
    [AUGMENTS <Index>]
    [DEFVAL <Value>]
    ::= <ObjectIdentifier>

```

Macro per la definizione di tipologie di oggetti e tabelle concettuali.

Le clausole INDEX e AUGMENTS sono consentite solo per la definizione mediante tabelle. Esattamente una delle clausole precedenti deve essere specificata durante la definizione della tabella.

```

tcpRtoMin OBJECT-TYPE
    SYNTAX      Integer32
    UNITS      "milliseconds"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The minimum value permitted by a TCP implementation for the
         retransmission timeout, measured in milliseconds. More
         refined semantics for objects of this type depend upon the
         algorithm used to determine the retransmission timeout. In
         particular, when the timeout algorithm is rsre(3), an object
         of this type has the semantics of the LBO and quantity
         of described in RFC 793."
    ::= { tcp 2 }

sysORTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF SysOREntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "The (conceptual) table listing the capabilities of the
         local SNMPv2 entity acting in an agent role with respect to
         various MIB modules. SNMPv2 entities having dynamically-
         configurable support of MIB modules will have a
         dynamically-varying number of conceptual rows."
    ::= { system 9 }

sysOREntry OBJECT-TYPE
    SYNTAX      SysOREntry
    MAX-ACCESS not-accessible
    STATUS      current
    DESCRIPTION
        "An entry (conceptual row) in the sysORTable."
    INDEX      { sysORIndex }
    ::= { sysORTable 1 }

```

Notification-Types (RFC 2578)

```
<descriptor> NOTIFICATION-TYPE
    [OBJECTS <Objects>]
    STATUS <Status>
    DESCRIPTION <Text>
    [REFERENCE <Text>]
    ::= <ObjectIdentifier>
```

Macro per la registrazione di un evento; in caso di evento, un manager o un agente può inviare una notifica appropriata a un altro manager.

Le clausole **OBJECTS** definiscono quali oggetti MIB devono essere contenuti nella descrizione dell'evento. La clausola **DESCRIPTION** deve descrivere quali istanze si intendono caso per caso.

New Types from Textual Conventions

In SNMP non si possono definire nuovi tipi di dato, per questa ragione si utilizza la **textual convention** ridefinendo tipi di dato già esistenti per definire un tipo di dato dal punto di vista della rappresentazione. Tuttavia, altri tipi potrebbero non essere derivati da una *textual convention*. Una clausola **DISPLAY-HINT** definisce una semplice figura della rappresentazione ASN.1 di un valore in un formato leggibile per gli esseri umani. La clausola **DISPLAY-HINT** può essere utilizzata solo insieme al tipo di dati **INTEGER** e **OCTET STRING** e da cui deriva.

Si noti che una *textual convention* può determinare restrizioni sull'ambito e non può definire un tipo assemblato.

Le convenzioni testuali sono definite nella RFC 2579.

```
<descriptor> ::= TEXTUAL-CONVENTION
    [DISPLAY-HINT <Text>]
    STATUS <Status>
    DESCRIPTION <Text>
    [REFERENCE <Text>]
    SYNTAX <Syntax>
```

La clausola **DISPLAY-HINT** definisce una figura bidirezionale della rappresentazione utilizzata internamente su una rappresentazione leggibile per gli esseri umani. Nella clausola **SYNTAX** possono essere utilizzati solo tipi di dati di base (si possono quindi limitare ulteriormente le convenzioni testuali non esistenti).

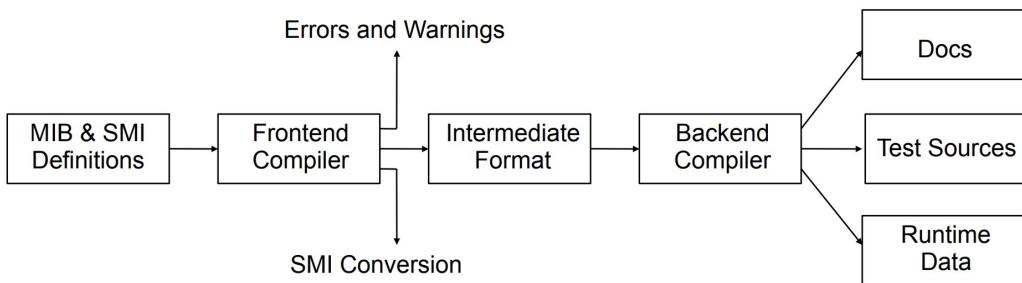
Tutte le altre semantiche devono essere definite nella clausola **DESCRIPTION**.

Further SMIv2 Macros

SMI definisce delle macro su ASN.1 che permettono di raggruppare gli oggetti tra di loro. Quando si creano dei MIB, è possibile che i vari oggetti implementino solo una parte dell'intera funzionalità. Queste macro hanno la possibilità di dividere gli oggetti in gruppi, così facendo sarà presente un **gruppo minimo**, ovvero il modulo del dispositivo che permetterà a quest'ultimo di essere conforme alla specifica.

- **OBJECT-GROUPS**: consente la definizione di gruppi di tipi di oggetti correlati; questa macro può essere utilizzata nella macro **MODULE-COMPLIANCE**.
- **NOTIFICATION-GROUPS**: consente la definizione di gruppi di tipi di notifica correlati; questa macro può essere utilizzata nella macro **MODULE-COMPLIANCE**.
- **MODULE-COMPLIANCE**: definisce uno o più vincoli che le implementazioni MIB devono soddisfare.
- **AGENT-CAPABILITIES**: descrive le capacità di un'implementazione MIB reale.

2.2.3 MIB-Compiler



Gli oggetti una volta definiti vengono scritti in formato testuale sul file system e possono essere utilizzati per più fini. Prima di eseguire una conversione di formato avviene una *verifica formale* allo scopo di verificare l'assenza di problemi; questa operazione risulta particolarmente utile nelle fasi di **IMPORT** da un altro oggetto. Se la verifica formale va a buon fine si ottiene una rappresentazione intermedia per il backend-compiler, il quale può produrre la documentazione (utilizzando le descrizioni presenti nei notification typer,

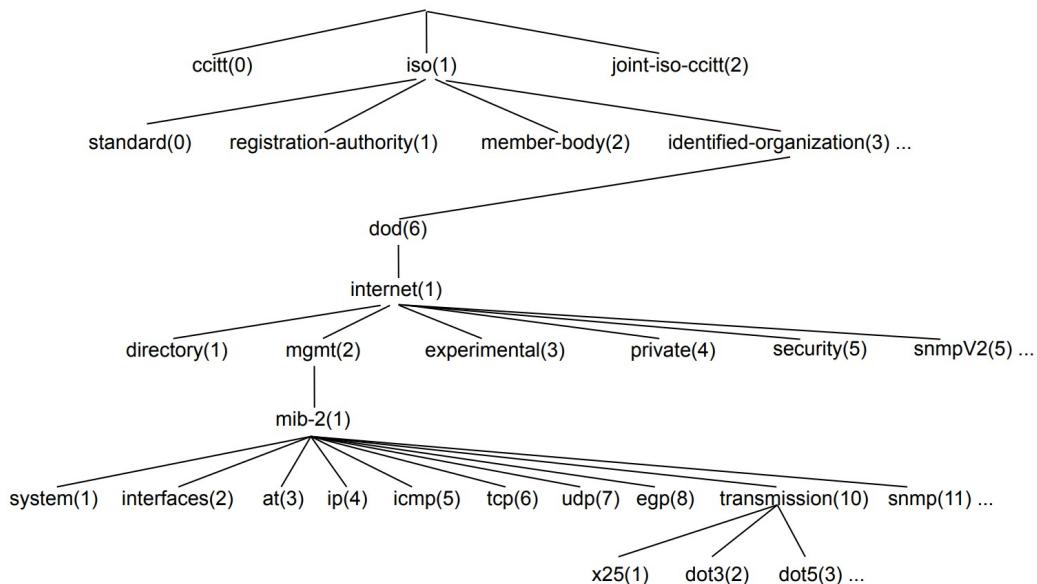
object identities, ...), dei test case per verificare se l'implementazione è corretta (se ho una variabile definita come read-only, il test deve fallire in caso si provi a fare una scrittura) e i run time data (generati dopo la lettura dei dati).

Osservazione: non esiste un formato intermedio standardizzato o generalmente accettato.

2.3 Fundamental MIBs

MIB-II (RFC 1213) definisce i tipi di oggetto per i protocolli Internet IP, ICMP, UDP, TCP, SNMP (e altre definizioni non rilevanti qui); fondamentalmente modella la gestione dello stack del protocollo TCP/IP. La definizione MIB-II ha come obiettivi definire gli errori di base e la gestione della configurazione per i protocolli Internet, evitare informazioni ridondanti nel MIB e avere pochissimi e deboli oggetti di controllo. Inoltre, l'implementazione del MIB non dovrebbe interferire con le normali attività di rete e nessun tipo di oggetto deve dipendere da essa.

Complessivamente 170 tipi di oggetti: alcune definizioni MIB si sono rivelate troppo semplici e minime (tabella di instradamento, tabella di interfaccia), mentre altre presuppongono un formato di indirizzo a 4 byte, quindi queste tabelle devono essere ridefinite per IP versione 6 (IPv6).



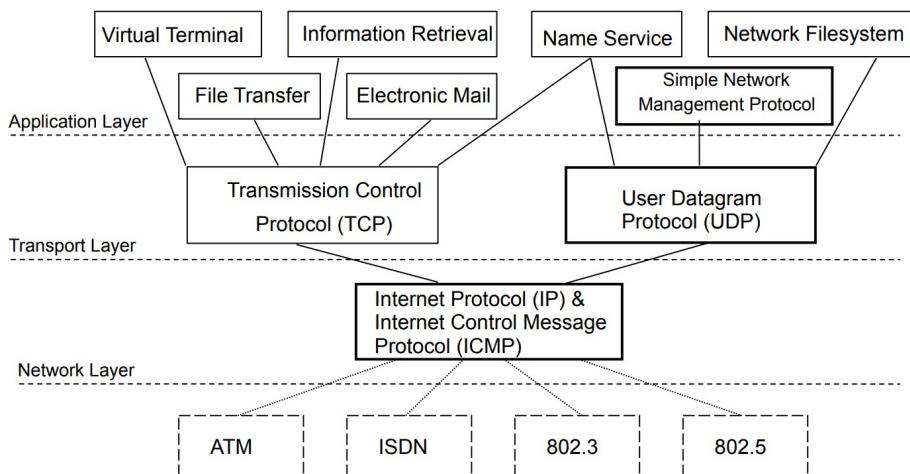
Il MIB-II è uno dei più importanti perché è nato con SNMP ed ha portato alla diffusione di quest'ultimo: il protocollo http è utile fin tanto che ci sono

siti web da leggere, MIB-II funge da sito web, infatti all'interno di quest'ultimo sono presenti le informazioni di monitoraggio da analizzare con SNMP. Infine, MIB-II è importante perché permette di gestire tutte quelle che sono le informazioni di base di un sistema basato su SNMP (informazioni sulle interfacce di rete, sull'ARP table, sul protocollo IP, sull'ICMP, ...).

Osservazioni su MIB-II

- Il ramo “**transmission**” ospita tutte le definizioni MIB che si occupano della trasmissione delle informazioni (X.25, PPP, RS232, SONET, ISDN, IEEE 802.3, IEEE 802.5, FDDI, ...)
- Il ramo “**at**” (ARP Table) è stato sostituito da un'estensione del gruppo di IP.
- Il ramo **egp** (External Gateway Protocol) non è più utilizzato in quanto il protocollo EGP al giorno d'oggi non ha alcuna importanza.
- Molti altri moduli MIB sono stati registrati nel nodo “**mib-2**”. L'assegnazione dei numeri di registrazione è delegata allo IANA (Internet Assigned Numbers Authority).
- In questi giorni sarebbe bene introdurre una certa modularità nel MIB in modo che i diversi rami possano essere aggiornati in modo indipendente.

2.4 SNMP Version 1



SNMP (*Simple Network Management Protocol*) si trova nella parte alta dello stack TCP/IP, nell’Application Layer. Nel caso del mondo ISO/OSI corrisponde al livello 7, dove si trovano le applicazioni usate dagli utenti.

È un protocollo che sfrutta **UDP** come servizio del livello di trasporto, perciò si tratta di un protocollo *connectionless*. In realtà esiste anche una versione che supporta il protocollo TCP, tuttavia si privilegia quella che fa uso di UDP.

Ordinamento lessicografico

Le istanze MIB sono disposte nel MIB lessicograficamente in base al valore dell’identificatore di oggetto che identifica l’istanza. Il registro SNMP utilizza questa caratteristica per leggere (*walk*) tabelle concettuali o MIB sconosciuti.

Object Identifier	Value	Object Identifier	Value
1.1.0	10.1.2.3	1.3.1.2.3	5
1.2.1.0	“FilterFresh”	1.3.1.2.4	7
1.2.2.0	54321	1.3.1.2.5	8
1.3.1.1.1	1	1.3.1.2.6	9
1.3.1.1.2	2	1.3.1.3.1	2
1.3.1.1.3	3	1.3.1.3.2	3
1.3.1.1.4	4	1.3.1.3.3	2
1.3.1.1.5	5	1.3.1.3.4	2
1.3.1.1.6	6	1.3.1.3.5	3
1.3.1.2.1	2	1.3.1.3.6	3
1.3.1.2.2	3		

Esempio di ordinamento lessicografico

Con questo ordinamento la struttura concettuale della tabella viene persa poiché l’output del percorso è un elenco e non più una tabella.

Il protocollo SNMP opera solo su questo elenco organizzato.

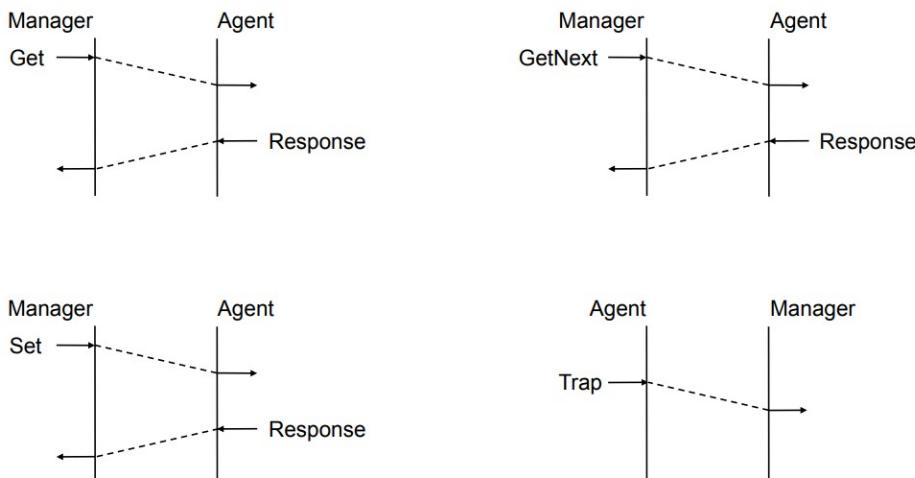
SNMPv1 operations

Le primitive principali di SNMPv1 sono

- get** legge il valore di un Object Identifier
- getNext** legge il valore dell’Object Identifier successivo a quello attuale
- set** imposta il valore relativo a un Object Identifier
- trap** manda una notifica al manager

Le prime tre operazioni sono sempre “iniziate” dal manager, la **trap** è l’unica che permette all’agent di contattare il manager: è necessaria per mandare una notifica relativa a un cambio di stato al manager poiché usare il *polling* sarebbe poco efficiente (poco scalabile) e poco sicuro. Se si utilizzassero solo le trap senza il ciclo di polling, nessuno controllerebbe che gli agent abbiano svolto correttamente il loro lavoro e il loro funzionamento (operazione che in origine era affidata al manager mediante il polling). In SNMP c’è il problema di **agent unreachable**: il manager pensa che stia andando tutto bene, mentre l’agent sta cercando di mandargli messaggi informativi che però non arrivano a causa di un guasto alla rete.

L’approccio SNMP è sensato se il numero di dispositivi (agent) da controllare è piccolo; se la quantità di dispositivi da monitorare è molto grande, c’è la necessità di arricchire le trap da parte degli agent rendendole più potenti: per esempio, nel caso di uno switch a 48 porte, la trap deve coprire lo spazio di gestione di tutte le porte.



Nota: il protocollo SNMP può scambiare solo (un elenco di) scalari.

2.4.1 Formato messaggi SNMPv1

Il primo campo dei messaggi SNMP contiene la *versione*. Il secondo campo contiene una stringa, *community*, usata per l’autenticazione degli utenti (controllo sui diritti, come una password).

SNMP message:

version	community	SNMP PDU		
---------	-----------	----------	--	--

GetRequest, GetNextRequest, SetRequest:

PDU type	request-id	0	0	variable-bindings
----------	------------	---	---	-------------------

GetResponse:

PDU type	request-id	error-status	error-index	variable-bindings
----------	------------	--------------	-------------	-------------------

Trap:

PDU type	enterprise	address	generic	specific	timestamp	vbs
----------	------------	---------	---------	----------	-----------	-----

variable-bindings:

name ₁	value ₁	name ₂	value ₂	...	name _n	value _n
-------------------	--------------------	-------------------	--------------------	-----	-------------------	--------------------

Infine vi è il **PDU** (*Protocol Data Unit*) che varia a seconda del tipo di messaggio. Nel caso della request la **get**, la **getNext** e la **set** contengono un'identificativo unico della richiesta usato per “legare” la successiva risposta e un campo contenente una serie di <Object Identifier, valore>: \simeq 1400 byte a disposizione, nel caso in cui si sfiori tale dimensione l'invio del messaggio fallisce e viene generato un codice di errore specifico. La **trap** ha un formato diverso.

All'inizio di ogni PDU vi è un campo **type** per distinguere il tipo di messaggio.

Osservazione: l'agent gestisce le richieste in modo atomico (una richiesta per volta), quelle che vengono ricevute e che non possono essere elaborate vengono accodate. Se le primitive vengono eseguite su un singolo object-ID piuttosto che su un insieme si perde la proprietà di atomicità (le richieste in questo caso vengono gestite come nei programmi multithreaded).

Get Operation

L'operazione **get** può essere utilizzata per leggere una o più variabili. Possibili errori durante l'elaborazione di un'operazione **get**:

noSuchName	l'istanza richiesta non esiste o non è una foglia
tooBig	il risultato della richiesta non rientra nella risposta (UDP)
genErr	si è verificato qualsiasi altro errore

In caso di più errori, solo un errore viene segnalato come indice di errore e stato di errore sono univoci nella PDU.

GetNext Operation

Recupera il nome dell'oggetto e il valore dell'istanza successiva, viene utilizzata per rilevare le strutture MIB e leggere le tabelle.

`getNext` consente di leggere le istanze MIB secondo l'ordine lessicografico: utilizzando più/successive operazioni `getNext` è possibile leggere l'intero MIB senza conoscerne la struttura. Possibili errori durante l'elaborazione di un'operazione `getNext`:

<code>noSuchName</code>	l'istanza richiesta non esiste (= fine del MIB)
<code>tooBig</code>	il risultato della richiesta non rientra nella risposta (UDP)
<code>genErr</code>	si è verificato qualsiasi altro errore

Set Operation

L'operazione di `set` scrive i valori in una o più istanze MIB atomicamente. Con l'aiuto dell'operazione `set` è possibile creare anche nuove istanze MIB, se la definizione MIB lo consente (non esiste una procedura standard definita in SNMPv1 per la creazione dell'istanza). Possibili errori durante l'elaborazione di un'operazione `set`:

<code>noSuchName</code>	l'istanza richiesta non esiste e non può essere creata
<code>badValue</code>	il valore specificato è di tipo sbagliato
<code>tooBig</code>	il risultato della richiesta non rientra nella risposta (UDP)
<code>genErr</code>	si è verificato qualsiasi altro errore

Viene definito anche il codice di errore `readOnly`, ma generalmente non viene usato.

Trap Operation

Con l'operazione `trap` gli agents possono informare un manager riguardo un evento; poiché non si vogliono mischiare i traffici di `get` e `set` con quelli di `trap`, in questo caso si usa la porta 162.

Per ricevere queste trap, il Manager deve avere un demone sulla porta 162 che resta in ascolto e le gestisce. Nel caso più comune, se non vi è nessuno ad ascoltare il traffico in arrivo o la macchina è spenta, i pacchetti vengono persi. Nel caso meno comune vengono comunque persi ma viene anche inviato un messaggio ICMP `destination unreachable` alla sorgente dei messaggi.
Nota: un manager può essere configurato per ignorare le traps!

La ricezione di un'operazione `trap` non è riscontrata, quindi non è affidabile in quanto può essere persa durante il trasferimento. La generazione di

traps può portare alle cosiddette “*trap storm*”, ad esempio se dopo un’interruzione di corrente tutti i dispositivi vogliono visualizzare contemporaneamente il riavvio. Gli agenti possono essere normalmente configurati con gli indirizzi IP degli host in cui è possibile inviare trap, tuttavia non esiste una tecnica standard in SNMPv1 per tale configurazione dell’agente: di solito viene utilizzato un file di configurazione (non il MIB). Sebbene si utilizzino le traps, il polling è ancora necessario (ad esempio, l’agente potrebbe essere inattivo).

2.5 SNMP Version 2c

Il termine SNMPv2 è ambiguo. SNMPv2c doveva risolvere i problemi della v1 tra cui l’autenticazione; la sola presenza della *comunità* come misura di sicurezza (che passava in chiaro) non era abbastanza considerando che al tempo venivano usati gli hub: il traffico veniva mandato ovunque nelle reti locali ed era quindi molto semplice reperire la “password” da usare per eseguire primitive SNMP. Esistono alcune varianti di SNMP versione 2:

- SNMPv2p: Versione con modello di sicurezza basato sulle parti, storico.
- SNMPv2c: SNMPv2 con banale modello di sicurezza basato sulla comunità, sperimentale.
- SNMPv2u: SNMPv2 con un modello di sicurezza basato sull’utente, storico.
- SNMPv2*: SNMPv2 con modello di sicurezza e amministrazione, storico.

Ci furono varie proposte ma il loro sviluppo andava per le lunghe quindi la community venne lasciata e SNMPv2 diventò SNMPv2c. Quest’ultimo ha riscontrato una certa diffusione, sebbene IETF non lo abbia mai standardizzato.

Sono state aggiunte le **eccezioni**, le quali consentono di segnalare gli errori di accesso alle istanze alle autorità MIB, senza causare il fallimento dell’intera operazione (come accadeva in SNMPv1), e due nuove primitive: **getBulk**, per velocizzare lo scorrimento di tabelle, e **inform** che si comporta da trap confermata.

SNMPv2 Exception	SNMPv1 Status	Used by
noSuchObject	noSuchName	Get
noSuchInstance	noSuchName	Get
endOfMibView	noSuchName	GetNext, GetBulk

SNMPv2 Exceptions (RFC 1905)

Get and GetNext Operations

Le istanze MIB non esistenti producono un'eccezione e non un errore. Simile alle operazioni SNMPv1 equivalenti.

Set Operation

Ci sono 14 possibili codici di errore durante l'elaborazione delle operazioni di set:

wrongValue	wrongEncoding	wrongType
wrongLength	inconsistentValue	noAccess
notWritable	noCreation	inconsistentName
resourceUnavailable	commitFailed	undoFailed

Ci sono altri due codici di errore che sono stati definiti ma non utilizzati realmente: `readOnly` e `permissionError`. Nessun supporto di codici di errore il tipo di oggetto.

GetBulk Operation

`getBulk` è stata introdotta per supplire ai problemi di `getNext` **velocizzando lo scorrimento** di tabelle. Scorrere una tabella con `getNext` implica dover fare una richiesta, attendere la risposta, inserire l'OID della risposta in una nuova richiesta... quindi per n righe di tabelle, si eseguono n `getNext` e, nel caso di host remoto, ci sarà un po' di latenza che renderà questo lungo; quindi, nella `getBulk` è stata inserita la possibilità di ricevere più righe di risposta per una singola richiesta e non solo: è stato pensato anche di unire `get` e `getNext`. La `getBulk` ha il seguente formato:

```
getBulk(non-repeating=n, max-repetitions=m, OID)
```

Il parametro `non-repeating` indica quanti OID di quelli specificati richiedono una `get` (normale), al resto dei parametri *verrà applicata una `getNext` in automatico*. Il parametro `max-repetitions` indica quante `getNext` eseguire sugli OID che non hanno richiesto una `get`.

Nota: qualora la tabella su cui si esegue la `getBulk` sia più piccola di `max-repetitions`, verranno ritornati tutti e soli i valori della tabella. Quindi, con una sola richiesta-risposta è possibile fare fino a n `getNext` e ciò non è male poiché il numero di messaggi in rete è diminuito (più efficiente).

Senza la conoscenza della lunghezza di una tabella è difficile per il manager selezionare un numero adatto per il numero massimo di ripetizioni:

- se il numero massimo di ripetizioni è troppo piccolo, non vi è alcun aumento di efficienza di `getBulk` rispetto all'operazione `getNext`;
- se `max-repetitions` è troppo grande, viene letto un gran numero di istanze non necessarie.

L'agente può eventualmente produrre una risposta, che può perdere in reti grandi/occupate o non essere elaborata affatto dal gestore (questo fa sì che il gestore ritrasmetta la richiesta).

Se il numero massimo di ripetizioni è elevato e la lettura delle istanze MIB richiede molto tempo, gli agenti possono ricevere più volte la richiesta del manager (ad esempio a causa della ritrasmissione) bloccando così l'agente per un po' di tempo.

Inform Operation

La struttura della PDU corrisponde a quella di una `trap` SNMPv2. Consente ai (nuovi) gestori di parlare tra loro (interazione limitata da SNMPv1 con agente-gestore o viceversa). La ricezione di un messaggio `inform` viene confermata con un messaggio di risposta.

Nella pratica non ha avuto molto successo perché i problemi non sono così semplici; se un mittente manda una `inform` e non riceve nessuna risposta, non può saperne il motivo: il destinatario potrebbe non averla mai ricevuta, il destinatario potrebbe aver risposto ma tale risposta si è persa oppure il destinatario potrebbe non rispondere e basta. Tuttavia, il problema risiede nel fatto che la `inform` complica SNMP: i mittenti devono ricordarsi selettivamente le `inform` alle quali non hanno ricevuto risposta e devono avere anche un timeout per ognuna di esse (i device che usano SNMP ci mettono tanto per rispondere alle `inform` in quanto il processo di SNMP non viene schedulato spesso); memorizzarsi le `inform` senza risposta è necessario per evitare che il mittente rinvii le `inform` alle quali ha ricevuto risposta.

Nota bene: va considerato che un agent può avere più manager, quindi il numero di `inform` da memorizzare può (ipoteticamente) aumentare. Inoltre, anche i manager vogliono sapere se gli agent/manager hanno ricevuto la loro risposta alla `inform` quindi anche loro hanno questo problema. Questo

problema ha decretato “il fallimento” della `inform` nella maggior parte dei casi.

SNMPv2 vs SNMPv1

- **Gestione contatori anche a 64 bit** (meno rischio di wrap dei contatori in connessioni ad alta velocità).
- `getBulk` consente di **fare *walk* più veloci** e diminuire il numero di richieste-risposte permettendo di fare polling di più Agent contemporaneamente.
- **Messaggi di errore più precisi sulle primitive.**
- Nuovi datatype.
- Introdotti nuovi protocolli di trasporto (in SNMPv3 si userà addirittura TCP).
- Le `trap` possono essere mandate a più manager con un solo messaggio.

2.6 SNMPv3

Obiettivi di progettazione di SNMPv3:

- Emissione di operazioni `set` protette.
- Definizione (si spera) di un modello di architettura longeva.
- Supporto sia di implementazioni semplici ed economiche che complesse e più costose (scalabilità).
- Indipendenza dagli standard.
- Utilizzo di materiale esistente (principalmente MIB) quando possibile (riutilizzo del progetto).
- SNMP deve rimanere il più semplice possibile.

Diverse implementazioni (commerciali e open source) disponibili. La diffusione in reti reali è ancora relativamente piccola (la maggior parte dei dispositivi di rete utilizza ancora SNMPv1).

2.6.1 Modello architetturale di SNMPv3

Il motore SNMP di un'entità SNMP è costituito da diversi sottosistemi e un dispatcher. Il modello manager/agente viene sostituito da una serie di “applicazioni” più piccole: la modularità consente l'avanzamento incrementale di SNMP tramite SNMP Context (RFC 2571).

SNMP Context

Un *context* è una quantità di informazioni di gestione a cui può avere accesso un'entità SNMP. Per ogni sottosistema un'entità SNMP ha potenzialmente accesso a diversi contesti e le stesse informazioni possono essere presenti in più contesti. In un dominio di gestione un'istanza di un Managed Objects è identificata in modo univoco dai seguenti elementi:

- l'identificazione dei motori SNMP in un'entità SNMP;
- il nome del contesto in un'entità SNMP;
- l'identificazione del tipo di Managed Object;
- l'identificazione dell'istanza.

Nota: l'identificazione di un motore SNMP non ha nulla a che fare con il loro indirizzamento.

Per rendere l'architettura a lungo termine hanno cambiato il trasporto e aggiunto nuovi protocolli ma soprattutto hanno deciso che ogni volta che si fosse implementata una nuova versione, sarebbe stato scelto il nuovo eventuale componente da linea di comando al momento della richiesta. Per la sicurezza invece è stato aggiunto il modello `username-psw` e hanno lasciato aperta la possibilità di un nuovo modello (sempre per il discorso di “implementare un'eventuale nuova versione”).

2.6.2 Problemi di sicurezza

I problemi di sicurezza di SNMP prima di SNMPv3 erano:

- Il messaggio è autentico? La query SNMP ricevuta è stata inviata in tal modo dal mittente? È sicuro che questo messaggio non sia stato modificato o non sia stato creato ad hoc?
- Chi ha fatto la richiesta *x*? La comunità non dà alcuna informazione sull'utente che ha fatto la richiesta poiché è una sola ed è uguale per tutti.

- Non erano presenti restrizioni sugli oggetti acceduti.
- Non erano presenti restrizioni per determinati utenti.

Integrità dei dati e autenticazione

Come è possibile sapere che i dati rimangano integri durante il loro viaggio? Come si fa a essere certi che nessuno ne cambi il contenuto? Tipicamente si usa una **funzione hash** (funzione che dato un input variabile restituisce un numero solitamente corto e di lunghezza fissa, per esempio somma dei byte nel pacchetto) che prende in input i dati del pacchetto e restituisce un **MAC** (un *digest*) che viene inviato insieme ai dati. Quando il pacchetto giunge a destinazione, la funzione hash viene ricalcolata con i dati appena arrivati e se il MAC corrisponde allora il pacchetto viene considerato integro.

Per l'integrità dei dati sarebbe sufficiente questo, mentre per l'autenticazione si necessita anche di una chiave privata: Manager e Agent, nella loro configurazione, concordano questa chiave privata che non transiterà mai sul filo. Al momento del calcolo della funzione hash, oltre ai dati, viene aggiunta in input anche la chiave privata, perciò, il MAC calcolato comprende anch'essa. Questa cosa è importante perché un ipotetico MITM potrebbe prendere il pacchetto inviato, modificarlo e ricalcolare la hash e il destinatario non si accorgerebbe di niente perché il MAC sarebbe uguale. Con la chiave questa cosa non si può fare perché essa è **nota solo a mittente e destinatario legali e reali**; il ricalcolo della funzione hash fatto dall'"impostore" non corrisponderebbe con quello che il destinatario rifarebbe non appena riceve il pacchetto SNMP perché la hash "falsa" non è stata calcolata con la chiave (in quanto sconosciuta all'"impostore"). Il destinatario, quindi, scarta il pacchetto. Così si ottiene anche l'autenticazione.

Protezione da ripetizione di messaggi vecchi

Si supponga che, finita una giornata, si esegua una **set** che spegne tutti gli apparati di rete che non servono più: questo pacchetto potrebbe essere catturato, salvato e reiniettato in rete in un secondo momento per creare un disservizio. Per evitare questo problema l'IETF ha imposto che Agent e Manager abbiano un clock sincronizzato, ciò può essere fatto sincronizzando l'ora in tutte le device di rete: all'interno del pacchetto SNMP è stato aggiunto un orario e una validità (in *epoch*) che rappresentano il momento in cui il pacchetto è stato forgiato e la sua durata; quindi, quando un mittente invia un pacchetto scrive l'orario di invio e la durata di questo pacchetto. A questo punto il ricevente controllerà se il pacchetto ricevuto contiene un orario per lui valido, se non è così scarta il pacchetto.

Nota bene: per aggirare ciò, si potrebbe alterare l'orario scritto nel pacchetto? No, perché il messaggio contiene l'*hash* e di conseguenza il destinatario scarterebbe il pacchetto per diversità di MAC.

Protezione contro lo sniffing

Per evitare lo *sniffing*, i pacchetti SNMP vengono **criptati con un cifrario a chiave simmetrica** (DES) usando la stessa chiave che viene usata per creare il MAC con la funzione hash. Di conseguenza, nasce spontaneo pensare che la funzione hash sia inutile se si sceglie la crittografia: no, non è inutile poiché in SNMPv3 si può scegliere di mandare un messaggio criptato senza hash, non criptato con hash o in qualsiasi altro modo si voglia. Principalmente questa cosa è stata fatta per la diversificazione degli apparati e soprattutto perché un tempo criptare e decriptare costava molto ai processori, perciò, era buona cosa poter scegliere se farlo o no. La protezione contro lo sniffing è quindi semplicemente la crittografia.

MIB Views

È possibile scrivere all'interno del file di configurazione di SNMP quali rami del MIB (e quindi quali OID) possono vedere determinati utenti. In questo modo si può negare la vista di determinati OID a utenti non autorizzati. Supponiamo il caso di una rete di media grandezza nella quale si ha uno switch condiviso fra le postazioni degli utenti: gli utenti possono interrogare solo le porte dello switch che hanno a che fare coi loro dispositivi.

2.6.3 Agents SNMP

Agenti monolitici

In un data center si vorrebbe evitare di fare polling su indirizzi diversi e porte diverse e si vorrebbe fare un *merge* così da fare polling allo stesso destinatario. Un modo per fare merge è usare un **agent monolitico** e un **method dispatcher**: si “mettono insieme” vari agent i quali sono preceduti dal method dispatcher. Quando il Manager crea la query SNMP inserendo gli OID di interesse, essa arriva al method dispatcher il quale, a seconda di quali sono gli OID definiti dai *variable bindings*, dispatcha le richieste agli Agent che hanno i MIB appropriati per rispondere.

L'unico problema è che non si possono inserire *variable bindings* che fanno parte di MIB diversi nella stessa richiesta. È necessario che chi fa le query le faccia di modo che contengano *variable bindings* che fanno parte dello stesso MIB e quindi dirette allo stesso Agent. L'agent monolitico è da intendersi

come un insieme di più librerie linkate insieme e ciascuna implementa una parte di esso.

Proxy Agents

Un proxy agent è simile ad un agent monolitico soltanto che non è monolitico: gli SNMP Agents a cui il proxy dispatcher inoltra le query SNMP possono essere anche su macchine distanti fra loro (non sono più librerie da linkare insieme di cui ognuna rappresenta una parte dell'Agent monolitico); tuttavia è sempre necessario che la query SNMP contenga *variable bindings* dello stesso MIB. Inoltre è necessario che nessun Agent collegato al proxy dispatcher implementi lo stesso MIB.

AgentX-Protocol Version 1

È la maniera più pulita per estendere un Agent SNMP: fa sì che la configurazione venga imparata a runtime. In questo caso è possibile mischiare *variable bindings* facenti parte di MIB diversi nella stessa query SNMP.

Si noti che prima di raggiungere l'agent che implementa AgentX, il messaggio è SNMP ma dopo non lo è più perché gli agent AgentX non implementano SNMP: si limitano a inoltrare messaggi in formato SNMP agli Agent Sub ma non implementano SNMP in maniera diretta. Il formato del messaggio in sé è di tipo AgentX ma al suo interno ha dei “pezzi” di messaggio in formato SNMP. In altre parole, non possiamo interrogare il Master AgentX con una query SNMP poiché non ha OID al suo interno!

Il Master AgentX si deve accendere per primo. Apre la porta 705 TCP e poi vengono avviati i Sub AgentX.

- Il Sub fa la **Open** verso il Master.
- Il Sub fa la **IndexAllocate** specificando gli OID a cui può rispondere (se il Master non ha già qualcuno che ha allocato quegli OID risponderà positivamente, negativamente altrimenti).
- Se il Master ha risposto in maniera positiva, il Sub procede a registrare gli indici.
- Il Sub procede a registrare le capabilities per tutti gli OID che ha registrato (**get**, **set**, ...).

Quando il sistema viene interrotto, si spengono per prima i Sub effettuando le operazioni precedenti ma tutte al contrario: rimuove capabilities, deregistra gli indici, dealloca gli indici e chiude.

Con questa metodologia il master AgentX non ha alcun interesse a sapere in maniera cablata dove e come sono fatti gli Agent (il master non ha bisogno che venga inserita in lui una conoscenza verso i Sub che si connetteranno) proprio perché grazie al protocollo AgentX, tramite le connessioni TCP sulla porta 705, tutte le informazioni necessarie vengono mandate dai Sub quando si connettono (`Open`, `IndexAllocate`, `Register`, `AddCaps`).

Parte II

Pratica

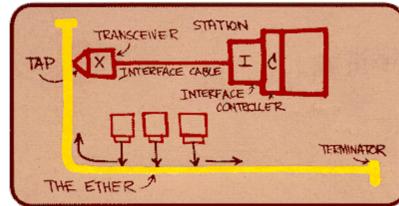
Capitolo 3

Ethernet

L'Ethernet è il sistema di comunicazione per computer più diffuso. Ha avuto un gran successo perché è un sistema molto semplice (ciò non vuol dire che fosse il migliore).

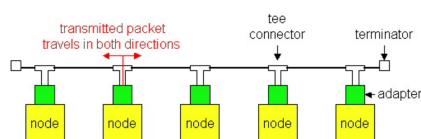
Nella versione iniziale Ethernet è semplicemente un filo, un cavo coassiale (simile a quello della TV) alle cui estremità ci sono due resistenze, chiamate *terminatori*, che impediscono effetti di disturbo (“rimbalzi di segnale”).

Le stazioni che vogliono connettersi alla rete hanno un “vampire” tap per collegarsi al cavo Ethernet. A questo tap era attaccato un cavo che terminava nell'Ethernet Controller (scheda di rete) della stazione.



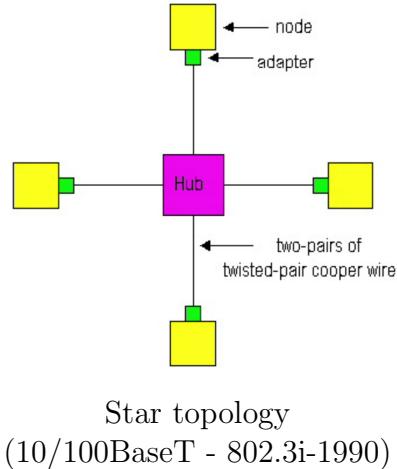
Bob Metcalfe, mid 1970s

Nel tempo questa tipologia è cambiata, da prima andando a utilizzare un connettore a T al posto del “vampire” tap.



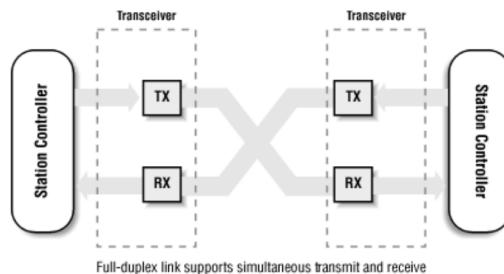
Bus Topology
(10Base2 - 802.3a-1988)

Negli anni '90 si è poi diffusa la tipologia a stella, in cui abbiamo uno hub/switch al quale i computer si connettono con un cavo di rete.



La differenza principale tra le due implementazioni è che nella prima soluzione il collegamento è condiviso da tutti i nodi connessi, ciò comporta la possibilità di avere delle “collisioni”, mentre con la struttura a stella si hanno dei collegamenti punto-punto.

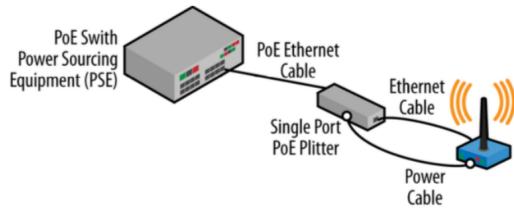
Ethernet può funzionare in **half-duplex** (storico) e **full-duplex** come specificato in 802.3x (ethernet moderno). In modalità full duplex, le stazioni possono trasmettere e ricevere simultaneamente poiché non vi è alcun conflitto per l'utilizzo del supporto condiviso.



PoE (Power over Ethernet)

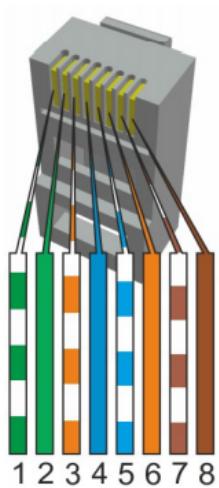
L'Ethernet oggi non è usato solo per trasmettere dati, ma anche per fornire alimentazione.

Standard **PoE** (Power over Ethernet): soluzione elegante per unire dati e alimentazione utilizzata principalmente per dispositivi a bassa potenza come antenne e punti di accesso WiFi. PoE Type 2: 25.5 W, 600 mA.

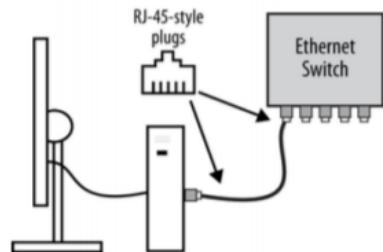


Ethernet Pinout

La Ethernet è composta da otto cavi accoppiati come mostrato in figura.



Pin	Description	10base-T	100Base-T	1000Base-T
1	Transmit Data+ or BiDirectional	TX+	TX+	BI_DA+
2	Transmit Data- or BiDirectional	TX-	TX-	BI_DA-
3	Receive Data+ or BiDirectional	RX+	RX+	BI_DB+
4	Not connected or BiDirectional	n/c	n/c	BI_DC+
5	Not connected or BiDirectional	n/c	n/c	BI_DC-
6	Receive Data- or BiDirectional	RX-	RX-	BI_DB-
7	Not connected or BiDirectional	n/c	n/c	BI_DD+
8	Not connected or BiDirectional	n/c	n/c	BI_DD-

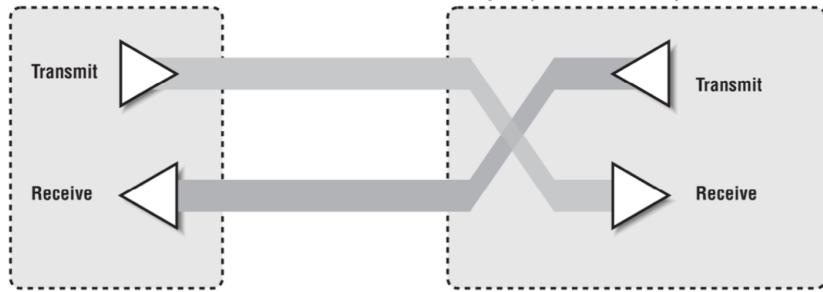


Esiste un ulteriore metodo di accoppiamento in cui i cavi sono accoppiati per colore.

Alternativamente esistono i cablaggi in fibra ottica. Tipicamente sono composti da due fibre fisicamente separate, ma unite da un “cappuccio” che serve per tenerle insieme.

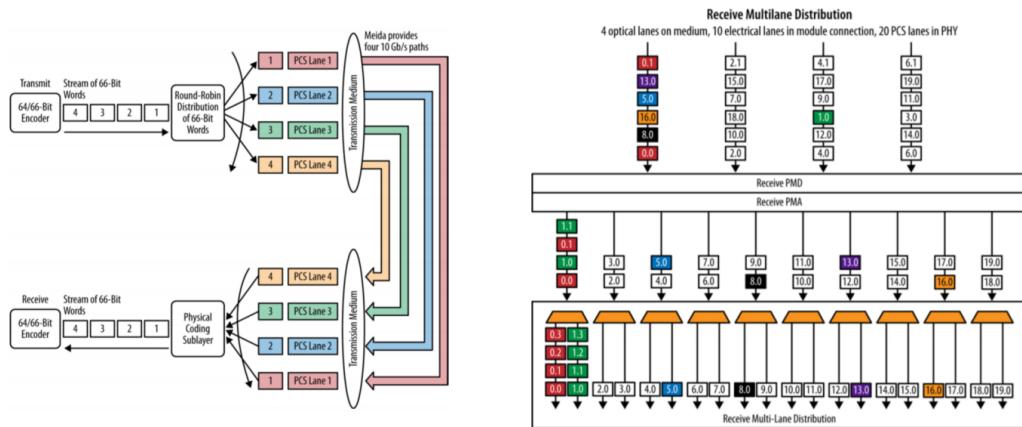


Cavo della fibra ottica



Fibra ottica

Nella fibra a 40 Gbit si usano quattro cavi da 10 Gbit collegati a un “distributore” *round-robin*. Nella 100 Gbit si usano dieci cavi 10 Gbit accoppiati tra loro. Questo perché spesso, invece di partire da zero, si parte da qualcosa di esistente e lo si adatta alle esigenze.



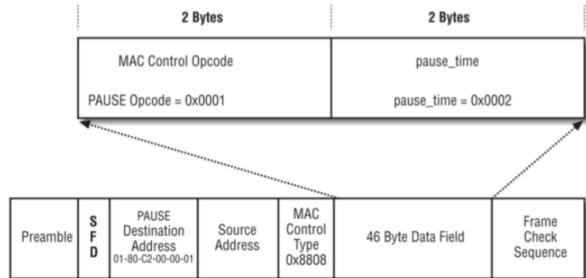
40 Gbit Ethernet

100 Gbit Ethernet

Primi standard

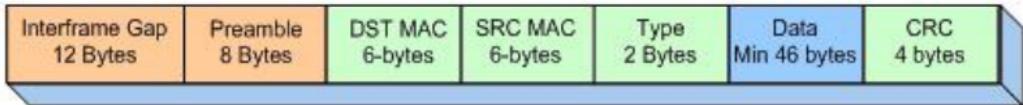
Ethernet Flow Control

Il controllo del flusso è un meccanismo utilizzato dal ricevitore (il ricevitore è il re!) per richiedere al mittente una breve pausa nella trasmissione del frame inviando un comando PAUSE all’indirizzo multicast di destinazione 01:80:C2:00:00:01.



In alcuni casi l'uso dei PAUSE frame non risulta molto comodo, come nel caso in cui si vuole testare la capacità di una scheda di rete. In questo caso (su Linux) si può usare il comando `ethtool` per specificare di ignorare i PAUSE frame.

3.1 Ethernet Framing



Interframe Gap: pausa (tempo) minima tra i pacchetti che consente al ricevitore di prepararsi per il frame successivo (96 nsec su Gbit Ethernet).

Preamble: una sequenza di 56 bit 0/1 alternati seguita da 8 bit a 1, consentendo al mittente di sincronizzarsi con il ricevitore.

CRC: sequenza di controllo del frame basata sull'utilizzo del CRC (Cyclic Redundancy Check) per rilevare (ma non correggere) gli errori di trasmissione.

Questi meccanismi permettono l'assenza del campo che indica la lunghezza dei dati trasmessi.

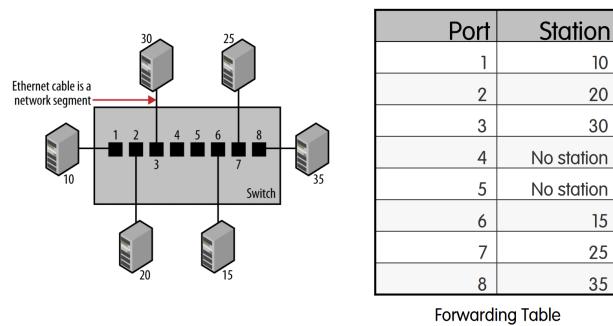
La dimensione minima del pacchetto Ethernet è di 60 byte: i frame più corti sono riempiti.

IFG	12
Preamble	8
Min Ethernet Frame	60
CRC	4
Total (bytes)	84

Speed	bits/sec	bytes/sec	PPS (bps/84)	Packet Rate (l/PPS) nsec
1 Gbps	1.000.000.000	125.000.000	1.488.095	672,00
10 Gbps	10.000.000.000	1.250.000.000	14.880.952	67,20
100 Gbps	100.000.000.000	12.500.000.000	148.809.524	6,72

3.2 Operation of Ethernet Switches

Gli switch Ethernet sono dispositivi “invisibili” alla rete, in quanto non modificano i frame Ethernet collegati a ponte tra le porte, e non richiedono alcuna configurazione poiché apprendono la topologia e la configurazione della rete analizzando il traffico di rete.



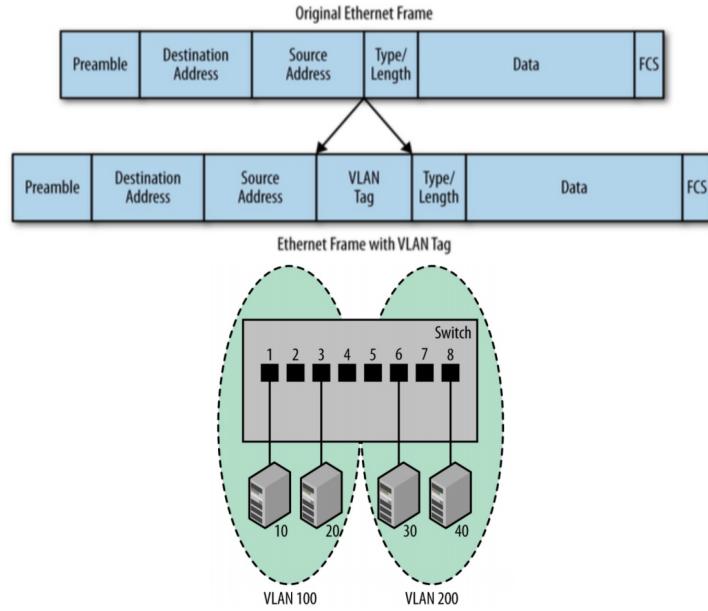
Le stazioni pubblicizzano la loro presenza inviando frame ethernet; la tabella di inoltro viene riempita dinamicamente e automaticamente (nessuna configurazione) man mano che vengono ricevuti i pacchetti in entrata. Poiché lo switch può essere annidato in un’architettura ad albero/mesh, è possibile collegare più stazioni alla stessa porta.

Il traffico viene inoltrato in base all’indirizzo ethernet di destinazione:

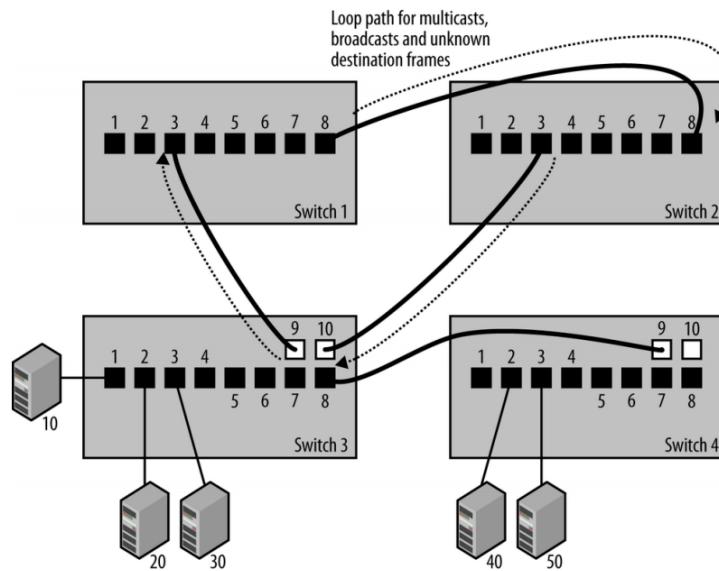
- Quando viene ricevuto un frame, il MAC di destinazione viene cercato nella tabella di inoltro.
- Se trovato, i frame vengono inviati alla stazione di destinazione (layer 2 anycast).
- Altrimenti il traffico viene inviato a tutte le porte tranne quella da cui è stato ricevuto il frame (broadcast layer 3).

Nota bene: le **VLAN** influiscono sull’inoltro del traffico creando più domini di broadcast. Al campo Type dell’Ethernet frame viene aggiunto un tag che

identifica un dominio: i pacchetti mandati in broadcast su una VLAN non vengono trasmessi all'esterno di essa.



Poiché Ethernet richiede che esista un solo percorso tra due stazioni qualsiasi, esistono dei protocolli che permettono a degli switch di disabilitare alcuni link (sempre attivi dal punto di vista elettrico) nel caso in cui “capiscano” che esiste un loop.



Capitolo 4

SNMP Monitoring

MIB-II (RFC 1213) definisce i tipi di oggetto per i protocolli Internet IP, ICMP, UDP, TCP, SNMP (e altre definizioni non rilevanti qui). Fondamentalmente modella la gestione dello stack del protocollo TCP/IP.

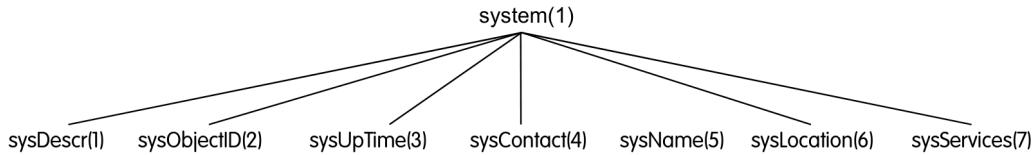
In totale 170 tipi di oggetti: alcune definizioni MIB si sono rivelate troppo semplici e minime (tabella di instradamento, tabella di interfaccia), alcune definizioni MIB presuppongono un formato di indirizzo a 4 byte, quindi queste tabelle devono essere ridefinite per IPv6.

Obiettivi della definizione MIB-II:

- Gestione di base degli errori e della configurazione per i protocolli Internet.
- Pochi deboli controlli
- Evitare informazioni ridondanti nel MIB.
- L'implementazione di MIB non dovrebbe interferire con le normali attività di rete.
- Nessun tipo di oggetto dipendente dall'implementazione.

4.1 “system” Group

Per leggere il **gruppo system** è possibile eseguire delle `get` individuali, oppure usare una `snmpwalk`: permette di leggere tutti gli object identifier che iniziano con un certo prefisso. Tutti gli oggetti facenti parte di tale gruppo sono **scalari**.

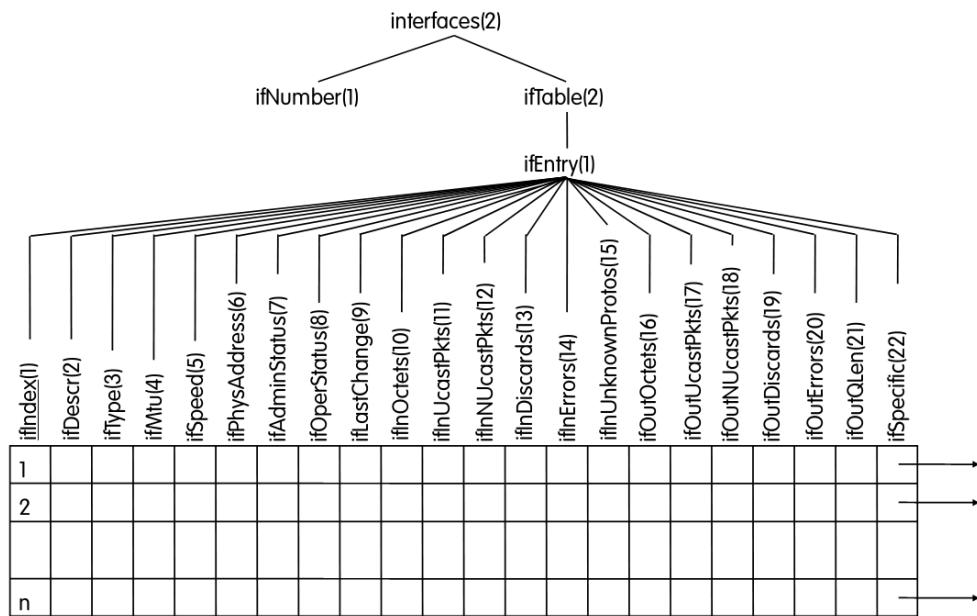


- **sysDescr**: contiene una descrizione approssimata di quello che fa una certa device sottoforma di una stringa libera.
- **sysObjectID**: è un object identifier che contiene informazioni riguardo la device (tipicamente contiene informazioni sul costruttore).
- **sysUpTime**: tempo di attività dell’agent SNMP, viene usato per determinare le discontinuità del servizio. Se $\text{sysUpTime}_{t_1} > \text{sysUpTime}_{t_2}$, dove $t_2 > t_1$, l’agent è stato reinizializzato e l’applicazione di gestione si basa sui valori precedenti.
- **sysContact**: contiene le informazioni per raggiungere il gestore della device.
- **sysName**: nome simbolico della device.
- **sysLocation**: indica dov’è fisicamente installata la device (la locazione potrebbe non essere aggiornata).
- **sysServices**: bitmap che riporta approssimativamente i servizi forniti dal sistema (utile quando non si capisce di che device si tratti usando **sysObjectID**).

In poche parole, il gruppo **system** è importante per mappare il dispositivo (tramite **sysObjectID.0**, **sysDescr.0** e **sysLocation.0**), controllare il wrapping check (**sysUpTime.0**) e segnalare problemi relativi al dispositivo all’amministratore (**sysContact.0**).

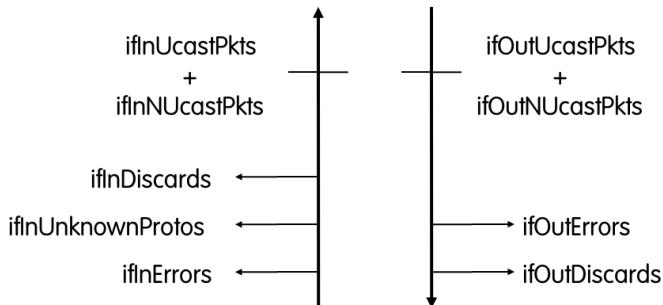
4.2 “interface” Group

Quando si interroga il gruppo delle interfacce si va a leggere e enumerare lo stato delle interfacce della periferica.



- **ifNumber**: numero delle interfacce.
- **ifIndex**: chiave della tabella.
- **ifDescr**: descrizione dell’interfaccia.
- **ifType**: tipo dell’interfaccia.
- **ifMtu**: MTU massima dell’interfaccia.
- **ifSpeed**: velocità dell’interfaccia di rete (in teoria quella reale).
- **ifPhysAddress**: MAC address dell’interfaccia.
- **ifAdminStatus**: lo stato amministrativo corrente dell’interfaccia. Valori: up (1), down (2), testing (3). Un valore diverso da up significa che l’interfaccia non è fisicamente presente sul sistema o che è presente ma non disponibile per il sistema operativo (per esempio il driver non è stato caricato). Caveat: SNMP MIB index holes.
- **ifOperStatus**: lo stato operativo corrente dell’interfaccia. Valori: up (1), down (2), testing (3). È simile a `ifconfig <device> up/down`.
- **ifOutQLen**: la lunghezza della coda dei pacchetti di output (in pacchetti). È utile per saperne di più sulle velocità di trasmissione e sul throughput (buffer pieno significa che il destinatario non è veloce come il mittente).

- **ifLastChange**: contiene il valore di **sysUpTime** nel momento in cui l’interfaccia è entrata nel suo stato operativo corrente. Utile per rilevare quando un’interfaccia cambia stato (per esempio se viene scollegato un cavo).



I diagrammi dei casi illustrano le dipendenze tra le variabili: il numero di pacchetti forniti da un’interfaccia di rete al successivo livello di protocollo superiore:

$$\text{ifInUcastPkts} + \text{ifInNUcastPkts}$$

il numero di pacchetti ricevuti dalla rete:

$$(\text{ifInUcastPkts} + \text{ifInNUcastPkts}) + \text{ifInDiscards} + \\ \text{ifInUnknownProtos} + \text{ifInErrors}$$

il numero di pacchetti effettivamente trasmessi:

$$(\text{ifOutUcastPkts} + \text{ifOutNUcastPkts}) - \text{ifOutErrors} - \text{ifOutDiscards}$$

Usare il gruppo “interface”

È la base del monitoraggio basato su SNMP: molti strumenti eseguono periodicamente il polling dei valori dalle interfacce (principalmente **ifInOctets** e **ifOutOctets**).

I valori sono aggregati e non divisi per protocollo, destinazione o AS; questa è una limitazione importante se è richiesto un monitoraggio a grana fine. Il motivo è che i contatori SNMP sono fondamentalmente i contatori del kernel “esposti” tramite SNMP. Gli errori di interfaccia possono essere utilizzati per rilevare problemi di comunicazione, in particolare sui collegamenti WAN.

Le statistiche sulla dimensione dei pacchetti non vengono riportate, tuttavia è possibile calcolare semplici statistiche sugli ottetti/pacchetti. Molti

produttori (ad esempio Cisco, Juniper) segnalano informazioni sulle interfacce fisiche e logiche (note anche come interfacce secondarie), altri (es. Extreme) hanno la voce nella tabella ma i contatori sono sempre zero. Utilizzando i contatori dell’interfaccia è possibile produrre report su:

- VLAN (LAN virtuale),
- PVC (circuito virtuale privato) sui collegamenti Frame Relay.

4.3 Uso del gruppo “arp”

Utile per accedere alla tabella arp (Address Resolution Protocol) di un dispositivo remoto. Può essere utilizzato per identificare attacchi di arp-poisoning o host configurati in modo errato (ad esempio indirizzi IP duplicati).

4.4 Bridge MIB

Utile per controllare lo stato degli interruptori L2/L3, non viene usato solo sui bridge. È in qualche modo complementare al MIB II in quanto fornisce informazioni agli host collegati alle porte dello switch. Usi comuni del bridge MIB:

- Per conoscere l’indirizzo MAC di un host collegato alla porta X/unità Y dello switch `dot1dTpFdbTable.dot1dTpFdbAddress` (**Nota:** il MIB II ha l’indirizzo MAC della porta dello switch).
- L’associazione MAC/porta è la base per rilevare la posizione fisica di un host. In effetti, poiché le porte dello switch sono solitamente collegate alle prese a muro, questo è un buon metodo per sapere chi è dove (utente → computer → porta dello switch → stanza/scrivania).
- Mantiene traccia dell’indirizzo MAC “precedente” (e dell’ora) connesso a una porta in modo che sia possibile tenere traccia degli utenti mentre si spostano da una stanza all’altra.
- Può essere utilizzato per rilevare le porte con più indirizzi MAC associati (trunk), quindi per rilevare utenti con più MAC (ad esempio un utente esegue VMware sul proprio PC o un utente è stato infettato da un virus/worm) o porte con uno switch collegato ad esso che la politica di rete potrebbe essere violata.

Con il Bridge MIB è possibile sapere dove sono attestati i MAC address (a patto che un nodo supporti SNMP), ma non permette di conoscere la **topologia** della rete; per fare ciò è possibile utilizzare il *Link Layer Discover Protocol (LLDP)*: permette di sapere a quale porta è fisicamente un'altra porta. LLDP si basa su messaggi Ethernet.

4.5 SNMP vs. CLI Counters

È opinione comune tra la comunità degli amministratori di rete che i contatori SNMP e CLI siano fondamentalmente una visione diversa della stessa cosa. A molti amministratori piacciono di più i contatori CLI, in quanto sono formattati per essere più comprensibili agli umani e molte implementazioni forniscono il comando per cancellare/azzerare il contatore. Si noti che la definizione di ciò che conta un dato contatore dipende dalla documentazione del fornitore.

I contatori CLI rimangono lo stile di vita di base nella gestione degli elementi. Il formato/aspetto dei contatori cambia da fornitore in fornitore, spesso anche all'interno dello stesso produttore come Cisco IOS, CatOS e PIX (sono rispettivamente il router, lo switch e il firewall OS utilizzati dai dispositivi Cisco).

I contatori SNMP invece permettono di “confrontare le mele con le mele”: i contatori hanno definizioni standard (definite da IETF, IEEE, alcuni fornitori...) indipendentemente dal tipo di elemento di rete o dal fornitore e dai nomi univoci a livello globale e difficili da pronunciare. Hanno una dimensione ben specificata, 32 o 64 bit (64 bit disponibile in SNMP v2c o v3), non iniziano necessariamente da zero e non sono progettati per essere usati direttamente dagli umani: richiedono una funzione delta per calcolare la velocità.

Nota: i buoni contatori sono generalmente derivati dalla specifica del protocollo sottostante.

Calcolare l'utilizzo della larghezza di banda con SNMP

Bandwidth Utilisation

$$\frac{(\Delta \text{ifInOctets} + \Delta \text{ifOutOctets}) \times 8 \times 100}{\Delta \text{time} \times \text{speed}}$$

Input Utilisation: $\Delta \text{ifInOctets} \times 8 \times 100 / (\Delta \text{time} \times \text{speed})$

Output Utilisation: $\Delta \text{ifOutOctets} \times 8 \times 100 / (\Delta \text{time} \times \text{speed})$

Cos'altro può fare SNMP?

- Rilevare e cancellare le connessioni TCP bloccate.
- Manipolare le voci ARP.
- Ottienere la temperatura ambientale.
- Controllare l'utilizzo della CPU.
- Monitorare gli alimentatori ridondanti/uninterruptible.
- Trovare utenti P2P (tabella NAT).
- Layout di topologia di rete (ad es. tramite CDP).

4.6 RMON: Remote Monitoring

Presente nella maggior parte delle apparecchiature di rete di fascia media-alta, spesso si tratta di implementazioni scarse/limitate. Alcuni fornitori vendono sonde autonome: caso preferito in quanto sono implementazioni complete del protocollo e non aggiungono ulteriore carico al router. Non tutte le implementazioni (in particolare quelle incorporate in router/switch) supportano l'intero standard ma solo gruppi SNMP selezionati.

Insieme a Cisco NetFlow è lo standard di monitoraggio industriale “affidabile”. Due versioni: RMON-1 (L2) e RMON-2 (L2/L3).

Cosa può fare RMON?

- Raccogliere i dati e segnalarli periodicamente a una stazione di gestione più centrale, che potenzialmente riduce il traffico sui collegamenti WAN e il sovraccarico di polling sulla stazione di gestione.
- Indicare quali host sono collegati alla LAN, quanto parlano e con chi.
- “Visualizzare” tutto il traffico LAN, utilizzo completo della LAN e non solo il traffico verso o attraverso il router.
- Filtrare e catturare i pacchetti (in modo da non dover visitare una LAN remota e collegare un analizzatore di LAN): è fondamentalmente uno sniffer remoto in grado di catturare il traffico in tempo reale (fino a quando il buffer di memoria integrato non è pieno).
- Raccogliere automaticamente i dati, confrontarli con le soglie e inviare trap alla stazione di gestione: ciò scarica gran parte del lavoro che potrebbe impantanare la stazione di gestione.

RMON vs. SNMP

Il protocollo SNMP viene utilizzato per controllare e configurare una sonda. Di solito i gestori GUI mascherano la complessità della configurazione basata su SNMP. Le statistiche e il traffico salvato vengono recuperati utilizzando SNMP dalle applicazioni di gestione per registrare statistiche su una rete e, eventualmente, porzioni selezionate del traffico di rete.

SNMP e RMON differiscono nel modo in cui raccolgono le statistiche sul traffico: SNMP esegue periodicamente delle richieste (polling) per ottenere le statistiche di rete (lo stato della rete è mantenuto dal gestore); RMON, d'altra parte, riduce lo stress del manager raccogliendo e archiviando le statistiche in contatori o *bucket* per il recupero da parte di una stazione di gestione.

Utilizzo della rete con RMON

La maggior parte dei gestori RMON utilizza i contatori RMON per calcolare l'utilizzo della rete. L'utilizzo della rete può essere calcolato per tutte le porte di un determinato switch a intervalli regolari.

Queste informazioni possono essere raccolte nel corso della giornata e utilizzate per generare un profilo di utilizzo della rete di uno switch o hub.

$$\% \text{ Network Utilisation} = \frac{100 \times ((\# \text{packets} \times 160) + (\# \text{octets} \times 8))}{\text{port speed} \times \text{time(sec)}}$$

RMON Alarm Group

Viene prodotto un evento ogni volta che viene superata una soglia, oppure quando un valore che prima era al di sopra di tale soglia ritorna all'interno dell'intervallo specificato. Considerazioni simili possono essere applicate a un valore di soglia inferiore.

Le soglie possono essere sul valore misurato (assoluto) o sulla differenza tra il valore corrente e l'ultimo valore misurato (valore delta).

Capitolo 5

Monitoraggio del flusso

5.1 Real-Time Flow Measurement (RTFM)

Misuratore molto flessibile e potente: contiene una serie di regole programmabili e può servire diversi lettori. È possibile anche programmare un comportamento da seguire in caso di sovraccarico.

Il lettore interroga il contatore, il quale è implementato da SNMP Meter MIB. Implementazione software gratuita NeTraMet, tuttavia non è stato accettato da parte dei produttori perché era complicato da usare (troppo potente). Specificato da RFC 2720-2724.

SNMP vs. Network Flows

SNMP si basa sul paradigma manager-agent: l'agente monitora la rete e informa il gestore (tramite **trap**) quando è accaduto qualcosa di importante (ad esempio l'interfaccia ha cambiato stato), mentre il manager mantiene lo stato dell'intero sistema leggendo periodicamente (polling) variabili (ad es. tramite SNMP **get**) dall'agent. Le variabili SNMP possono essere utilizzate sia per la gestione di elementi/dispositivi/sistema (ad es. informazioni su spazio su disco e partizioni) sia per il monitoraggio del traffico.

I flussi di rete vengono emessi da una sonda verso uno o più collettori a seconda delle condizioni di traffico.

- I flussi contengono informazioni sul traffico analizzato (non contengono informazioni sul dispositivo/sonda come le variabili MIB II).
- I flussi immessi hanno un formato ben definito (ad esempio Cisco Net-Flow v5) e spesso utilizzano UDP come trasporto (nessun protocollo specializzato come SNMP).

- Nessun concetto di flussi di “allarme”, né capacità della sonda di eseguire azioni basate sui flussi: tutta l’intelligenza è nel collettore.
- La strumentazione della sonda viene eseguita offline.
- Le sonde vengono attivate nel punto in cui scorre il traffico di rete (ad es. all’interno di router e switch).

Cosa bisogna aspettarsi con la misurazione dei flussi?

È possibile conoscere quali sono gli indirizzi IP/ASN più contattati, il tipo e la quantità del traffico (SMTP, Web, condivisione file, ecc.), i servizi che sono in esecuzione, riepiloghi sul traffico a livello di reparto, rintracciare i virus basati sulla rete sull’host, trovare (per esempio) i 100 server più contattati, trovare gli host più occupati, …

Cosa non è possibile misurare con i flussi?

Traffico non IP (ad esempio NetBIOS, AppleTalk), informazioni di livello 2 (ad es. modifiche dello stato dell’interfaccia), traffico filtrato (ad es. contatori dei criteri del firewall), statistiche per collegamento (ad es. utilizzo del collegamento, congestione, ritardo, perdita di pacchetti), statistiche dell’applicazione (ad es. latenza delle transazioni, numero di risposte positive/negative, errori di protocollo), …

5.1.1 Cosa è un flusso di rete?

Un flusso è un insieme di pacchetti che hanno un insieme di proprietà comuni, ad esempio indirizzo IP/porta comune. Un flusso viene (messo in coda per essere) emesso solo quando è scaduto.

Contenuto dei flussi di rete

Un flusso contiene:

- Peers: origine e destinazione del flusso.
- Counters: pacchetti, byte, tempo.
- Informazioni sul percorso: AS, maschera di rete, interfacce.

I flussi possono essere unidirezionali (impostazione predefinita) o bidirezionali (solo v9/IPFIX); due flussi unidirezionali opposti equivalgono a un flusso bidirezionale. I flussi bidirezionali possono contenere altre informazioni come il tempo di andata e ritorno, il comportamento del TCP, ...

Overhead vs. Accuracy

Più risultati di misurazione in più dati raccolti.
Maggior aggregazione del flusso, minore granularità.
Overhead (ad es. carico della CPU) su router, switch, host finali.

Sicurezza vs. Condivisione dei dati

I flussi emessi devono raggiungere i collettori su percorsi protetti
(ad esempio utilizzando una rete/VLAN diversa).
La privacy dell'utente deve essere rispettata.
Le misurazioni del traffico devono essere protette per non divulgare
importanti informazioni sulla rete a terzi.

5.1.2 NetFlow Architecture

I flussi vengono esportati (push) dalla sonda quando scaduti, contrariamente a SNMP in cui il manager interroga periodicamente l'agente. Il protocollo di trasporto del flusso è NetFlow (no SNMP) e il protocollo di configurazione sonda/collector non è specificato dal protocollo NetFlow.

Il raccoglitore NetFlow ha il compito di assemblare e comprendere i flussi esportati e combinarli o aggregarli per produrre i preziosi report utilizzati per l'analisi del traffico e della sicurezza.

Collection Architecture

È buona norma avere più router di frontiera (ad esempio per disaster recovery) e tipicamente i collettori vengono installati vicino a tali router. Inoltre si cerca di collezionare i dati in modo tale da non influenzare la rete, per esempio sincronizzando i dati verso una struttura centrale di sera (quando c'è meno traffico), o da non essere influenzati: se si usa UDP per la sincronizzazione, in caso di congestione di rete, i pacchetti trasmessi vengono persi.

In caso di reti complesse, è probabile che all'interno vi siano diversi router; per evitare di conteggiare più volte lo stesso traffico (*deduplicazione*) nei router interni si conta il traffico solo in una direzione mentre in quelli di frontiera si conta in entrambe.

Limiti dello spazio di raccolta: dipende dal traffico.

Using Flows

Nel flusso troviamo i contatori dei pacchetti e dei byte, fondamentali per calcolare l'uso di una rete. Sono presenti l'ora di inizio e l'ora di fine; ci sono gli indici delle interfacce (informazione che si poteva ottenere anche tramite SNMP, ma che prima non poteva essere caratterizzata). Inoltre sono presenti informazioni relative alla qualità del servizio (QoS), agli indirizzi IP (sorgente e destinazione), alle porte (sorgente/destinazione e TCP/UDP) e di routing (next hop, AS sorgente e destinazione, ...).

5.1.3 Nozioni di base su Cisco NetFlow

Diverse versioni v 1,5,6,7,8,9. La più comune è la v5, l'ultima è la v9. Ogni versione ha il proprio formato di pacchetto: v1,5,6,7,8 hanno un formato fisso/chiuso, specificato, invece il formato della v9 è dinamico e aperto alle estensioni. La v1 non ha numeri di sequenza (nessun modo per rilevare i flussi persi), v5,6,7,8 hanno numeri di sequenza di flusso (cioè tengono traccia del numero di flussi emessi), mentre v9 ha un numero di sequenza di pacchetti (non di flusso), quindi è facile conoscere il numero di pacchetti persi, ma non dei flussi persi. La “versione” definisce il tipo di dati nel flusso.

Analisi del traffico solo sul traffico in entrata (ovvero il traffico che entra nel router) solo IP (non su tutte le piattaforme). Flussi unidirezionali (fino a v8), bidirezionali su v9. IPv4 unicast e multicast: tutte le versioni di NetFlow; IPv6 è supportato solo dalla v9.

Protocollo aperto definito da Cisco e supportato su piattaforme IOS e CatIOS (nessun supporto NetFlow sui firewall PIX) e su piattaforme Cisco (ad esempio Juniper, Extreme).

Nascita e morte di un flusso

Ogni pacchetto che viene *inoltrato* all'interno di un router, o di uno switch di terzo livello, viene esaminato per un insieme di attributi del pacchetto IP. Tutti i pacchetti con lo *stesso* indirizzo IP di origine/destinazione, porte di origine/destinazione e interfaccia di protocollo vengono raggruppati in un flusso e quindi vengono conteggiati i pacchetti e i byte. I flussi attivi vengono archiviati in memoria nella cosiddetta **cache NetFlow**.

I flussi vengono terminati quando viene soddisfatta una di queste condizioni:

- La comunicazione di rete è terminata (ad es. un pacchetto contiene il flag TCP FIN).

- Il flusso è durato troppo a lungo (impostazione predefinita 30 min).
- Il flusso non è stato attivo (ovvero non sono stati ricevuti nuovi pacchetti) per troppo tempo (impostazione predefinita 15 sec).
- La cache del flusso era piena e il gestore della cache ha dovuto eliminare i dati.

Nota: la cache dei flussi ha una dimensione limitata, quindi spesso non è possibile memorizzare tutti i flussi.

La cache di NetFlow si riempie costantemente di flussi e il software nel router, o nello switch, cerca nella cache i flussi che sono terminati o scaduti e questi flussi vengono esportati sul server di raccolta NetFlow; di conseguenza i flussi di rete possono essere suddivisi in più flussi netflow che, se necessario, vengono riassemblati dal collettore di flusso.

Formato dei pacchetti

Come detto in precedenza, esistono principalmente due versioni di NetFlow (le altre non sono molto diffuse); la v5 è quella che storicamente è stata la più diffusa, tuttavia oggi sta diventando più comune la versione 9 a causa del suo supporto a IPv6 (la v5 manca anche del supporto alle VLAN).

In un pacchetto NetFlow v5 è possibile inviare al massimo 30 flussi per mantenere la sua dimensione sotto ~ 1480 bytes e di conseguenza evitare la frammentazione. Quando un flusso è terminato, la sonda lo tiene in memoria per un certo periodo di tempo per aspettare altri potenziali flussi; dopo tale lasso di tempo, il flusso viene mandato lo stesso.

I formati fissi (v1-v8) per l'esportazione sono:

- Facile da implementare.
- Consuma poca larghezza di banda.
- Facile da decifrare al collettore.
- Non flessibile (mancano le informazioni sulle porte di TCP, UDP e ICMP).
- Non estensibile (nessun modo per estendere il flusso a meno che non sia definita una nuova versione).
- Mancano alcune funzionalità: L2, VLAN, IPv6, MPLS.

5.1.4 Principi di NetFlow v9

Protocollo aperto (non proprietario) definito da Cisco nell'RFC 3954. I dati sono stati suddivisi in template e record: un template è composto da tipo e lunghezza, mentre un record composto da ID modello e valore. I template vengono inviati periodicamente e sono un prerequisito per la decodifica dei record di flusso; i record di flusso, invece, contengono il flusso vero e proprio.

Esistono anche gli option template e gli option record, i quali contengono la configurazione della sonda (ad es. velocità di campionamento del pacchetto/flusso, contatori di pacchetti dell'interfaccia).

Come nelle versioni precedenti il modello di esportazione è di tipo push: quando la sonda decide che ci sono dati da inviare, li invia. Inoltre la sonda manda regolarmente template (per evitare i casi in cui il collezionatore sia stato avviato dopo la sonda).

La v9 è indipendente dal protocollo sottostante e quindi è pronta per qualsiasi protocollo affidabile; può inviare sia il modello che il record di flusso in un'unica esportazione e può intercalare diversi record di flusso in un pacchetto di esportazione.

Anche NetFlow v9 è unidirezionale, però tramite i template è possibile configuralo in modo che sia bidirezionale.

NetFlow v5 vs. 9

	v5	v9
Flow	Format Fixed	User Defined
Extensible	No	Yes (Define new FlowSet Fields)
Flow Type	Unidirectional	Bidirectional
Flow Size	48 Bytes (fixed)	It depends on the format
IPv6 Aware	No	IP v4/v6
MPLS/VLAN	No	Yes

5.1.5 IPFIX

Obiettivo: trovare o sviluppare una tecnologia di misurazione del flusso di traffico IP di base e comune da rendere disponibile su (quasi) tutti i router futuri.

IPFIX è fortemente basato su NetFlow v9, ma ha la capacità di definire nuovi campi di flusso utilizzando un formato standard (OID). I messaggi sfruttano un trasporto basato su SCTP (Stream Control Transport Protocol), ma opzionalmente supportano anche UDP/TCP.

Attualmente è una bozza di specifica del protocollo e in conclusione non è altro che che NetFlow v9 su SCTP con qualche differenza.

5.1.6 Aggregazione e filtraggio

Aggregazione dei flussi

NetFlow/IPFIX offre la possibilità di aggregare flussi: quando vengono esportati, possono essere inviati a quintupletta

$$\langle \text{srcIP}, \text{dstIP}, \text{Proto}, \text{srcPort}, \text{dstPort} \rangle$$

oppure possono essere mandati aggregati. Lo stesso flusso può essere aggregato più volte utilizzando criteri diversi, ad esempio da flussi grezzi è possibile generare: un elenco dei protocolli, una matrice di conversazione (chi sta parlando con chi) e le porte TCP/UDP più utilizzate. Il flusso di aggregazione anticipato può far risparmiare tempo/memoria rispetto all'aggregazione tardiva, ad esempio, la matrice di conversazione (chi parla con chi) è molto più facile da implementare aggregando i dati sulla sonda invece di utilizzare flussi grezzi/non aggregati.

I flussi possono essere aggregati utilizzando criteri “esterni” e non solo in base a campi di flusso non elaborati. Di solito questi criteri esterni vengono applicati ai campi “chiave” (non “valore”) come porta, indirizzo IP, protocollo, ... e sono utilizzati per raggruppare i valori. I criteri vengono aggiunti (non sostituiti) ai campi esistenti.

L’aggregazione può essere eseguita dalla sonda, dal collettore o da entrambi: l’aggregazione della sonda è molto efficace in termini di utilizzo delle risorse e traffico del flusso di rete, mentre l’aggregazione dei collettori è più potente (ad es. flussi aggregati prodotti da diverse sonde), ma piuttosto costosa (riceve tutto l’aggregato).

Filtraggio dei flussi

Filtrare i flussi significa scartare i flussi in base ad alcuni criteri come durata del flusso (si scartano i flussi che sono durati meno di x secondi), sorgente e/o destinazione del flusso (ignora i flussi contenenti indirizzi di broadcast) o porta del flusso (ignora i flussi originati dalla porta x). Si noti che:

- filtraggio \neq aggregazione \rightarrow fanno due lavori diversi;
- il filtraggio e l’aggregazione possono coesistere;
- il filtraggio viene solitamente applicato prima dell’aggregazione dei flussi e non dopo.

5.1.7 Flussi e sicurezza

NetFlow/IPFIX può essere utilizzato per la sicurezza e non solo per la contabilità del traffico: rilevare portscan/portmap, rilevare attività su porte sospette o identificare le fonti di spam, come server non autorizzati.

Semplice sistema per identificare intrusioni basato sul flusso:

- flussi con un numero eccessivo di ottetti o pacchetti (inondazioni);
- sorgenti IP che contattano più di N destinazioni, scansione host;
- sorgenti IP che contattano più di M porte di destinazione su un singolo host (per porte 0-1023), scansione delle porte.

Si noti che nonostante IPFIX abbia il supporto ai MAC address, NetFlow resta un strumento di livello tre: non ha visibilità degli switch, per questo è possibile utilizzare sFlow.

Capitolo 6

Metriche e rilevamento delle anomalie

6.1 Metriche

Con sistemi come NetFlow è possibile andare a monitorare il traffico di rete, tuttavia è importante anche riuscire a capire che cosa monitorare, cioè quali sono le metriche importanti da monitorare. Dal punto di vista di un ISP è importante andare a monitorare il Service Level Agreements: il provider deve capire se la rete sta fuzionando, se le linee internet a sua disposizione sono adeguate per il traffico che sta passando e deve fare delle previsioni per riuscire a capire se ha abbastanza banda per crescere nel futuro. Dal punto di vista dell'utente non si parla di linea, di connettività, di Autonomous Systems... ma di servizi: riuscire a connettersi alla propria mailbox, connettersi a Google tramite il proprio browser, ...

Il traffico può essere monitorato:

- sugli end-system (poiché richiede di installarvi una sonda di solito viene evitato);
- sfruttando sistemi esistenti, per esempio usando NetFlow su un router;
- usando delle macchine appositamente configurate per lo scopo.

6.1.1 Benchmarking

Al contrario delle metriche della vita quotidiana (kg, litri, ...), le metriche usate per il traffico di rete spesso non sono standardizzate e le misure dei fornitori sono eseguite in modi leggermente diversi, rendendo i risultati difficili

da confrontare. Per questo gli operatori e il mondo delle reti hanno definito degli standard di *benchmarking* che permettono di creare dei termini di paragone.

- **Disponibilità:** può essere espressa come la percentuale di tempo che un sistema, un componente o un'applicazione di rete è disponibile per l'utente. È basata sull'affidabilità dei singoli componenti di una rete.

$$\% \text{ Availability} = \frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}$$

MTBF è il tempo medio tra i fallimenti e MTTR è il tempo medio per riparare il fallimento.

- **Tempo di risposta:** tempo che intercorre in un sistema tra un input e la sua reazione. È desiderabile che sia il più breve possibile.
 - **Network delay:** quanto una rete ritarda l'esecuzione di un'azione.
 - **Application delay:** quanto un'applicazione ritarda l'esecuzione di un'azione.
- **Throughput:** misura la quantità di dati che può essere inviata su un link in un certo momento. Spesso viene utilizzato per stimare la larghezza di banda disponibile sul link. Si noti che throughput e banda sono misure diverse: il primo è orientato all'applicazione.
- **Goodput:** come il throughput, ma considera solo il payload del pacchetto, cioè i "dati reali" che sono trasportati dalla rete. È utile per identificare le cosiddette *stealth connections*.
- **Utilizzo:** è una misura più precisa rispetto al throughput, si riferisce alla determinazione della percentuale di tempo in cui una risorsa è in uso in un determinato periodo di tempo. Spesso un utilizzo molto scarso significa che qualcosa non funziona come previsto.

Per interpretare tali metriche sono vengono utilizzate la media, la varianza e la deviazione standard. Viene anche definito un intervallo di confidenza (media \pm margine di errore), i dati al di fuori di tale intervallo vengono scartati.

Percentile: il valore al di sotto del quale cade una determinata percentuale di osservazioni in un gruppo di osservazioni.

Nel networking un percentile popolare è il 95esimo.

Quartile: è una divisione di serie in quarti.

- Q_1 è definito come il 25esimo percentile.
- Q_2 è definito come il 50esimo percentile.
- Q_3 è definito come il 75esimo percentile.

Outlier

Gli outliers (valori anomali) sono i valori che “si trovano al di fuori” degli altri. In statistica un valore è outlier quando cade al di fuori del

- limite inferiore: $Q_1 - 1,5 \cdot IQR$
- limite superiore: $Q_3 + 1,5 \cdot IQR$

dove IQR (interquartile range): $Q_3 - Q_1$. Gli outliers vengono utilizzati per individuare valori anomali, spesso denominati “aberranti”, ovvero che si discostano da uno standard accettato, tra cui:

- errori sperimentali/di misurazione;
- distribuzioni a coda pesante (cioè che vanno a zero molto lentamente) in cui uno o più valori molto grandi influenzano le statistiche.

Latenza e jitter

La **latenza** è la quantità di tempo che impiega un pacchetto per andare da sorgente a destinazione ed è molto importante per le applicazioni interattive. $\frac{RTT}{2}$ non è la latenza di una rete, ma la somma della latenza client-to-server e della latenza server-to-client è l'RTT.

Il **jitter** è la varianza dell'intra-packet delay su un link monodirezionale ed è molto importante per le applicazioni multimediali.

$$\text{jitter} = \frac{\sum_i (|x_i - x_{i-1}|)}{n - 1}$$

È un modo per misurare quanto irregolare è un segnale. **Nota:** il jitter tiene conto dell'ordine degli eventi mentre deviazione standard, media... non lo fanno.

Bandwidth

Per misurare la banda va definito inanzitutto l'intervallo di misura (TC), solitamente viene fissato a un secondo. Si tratta dell'intervallo di tempo, o “intervallo di larghezza di banda”, utilizzato per controllare i picchi di traffico. Il Burst Committed (BC) è il numero massimo di bit che la rete accetta di trasferire durante qualsiasi TC, mentre il CIR (Committed Information Rate) è la velocità con cui una rete accetta di trasferire informazioni in condizioni normali, mediata su un incremento minimo di tempo: si tratta di una delle principali metriche tariffarie negoziate ed è misurato in bit al secondo.

$$\text{CIR} = \frac{\text{BC}}{\text{TC}}$$

Inoltre viene definito il Burst Excess (BE), ossia il numero di bit che si tenta di trasmettere dopo aver raggiunto il valore BC. La velocità massima che può essere raggiunta (Maximum Data Rate) viene calcolata come

$$\frac{\text{BC} + \text{BE}}{\text{BC}} \cdot \text{CIR} = \frac{\text{BC} + \text{BE}}{\text{TC}}$$

Si noti che la banda è una misura per link, mentre il throughput è una misura per comunicazione.

Approcci di monitoraggio

- Misure attive: si inietta del traffico e si studia come la rete reagisce a tale traffico.
- Misure passive: non si inietta nessun traffico, ma si osserva ciò che passa tramite la rete.

Le misure attive sono spesso end-to-end, mentre quelle passive sono limitate al link dove viene osservato il traffico.

6.2 Rilevamento delle anomalie

Un'**anomalia** è un’osservazione che si discosta tanto dalle altre osservazioni da far sospettare che sia stata generata da un meccanismo diverso. Data una misura e trovati gli outliers è necessario capire se quest’ultimi rappresentino un evento interessante o dei dati non voluti (sbagliati): nel primo caso l’obiettivo è quello di analizzare l’outlier, nel secondo è eseguire un *data cleaning*.

<i>Definizioni</i>	
Serie	sequenza ordinata di numeri
Ordine	indice di un numero nella serie
Timeseries	serie di punti ordinati nel tempo
Osservazione	valore numerico osservato in un tempo specifico
Forecast	stima di un valore atteso in un momento specifico
Forecast Error	differenza dell'osservazione rispetto alla previsione
SSE	la somma degli errori di una serie al quadrato

Le serie temporali possono essere di due tipi, **univariate** e **multivariate**. Una *serie temporale univariata* è costituita da singole osservazioni (scalari) registrate in sequenza su incrementi di tempo uguali, per esempio la temperatura giornaliera, mentre una *serie temporale multivariata* ha più di una variabile dipendente dal tempo: ogni variabile dipende non solo dai suoi valori passati, ma ha anche una certa dipendenza da altre variabili; questa dipendenza viene utilizzata per prevedere i valori futuri, ad esempio le previsioni del tempo che dipendono da temperatura, vento, umidità, copertura nuvolosa.

Una serie temporale è *stazionaria* quando le sue proprietà statistiche, media e varianza, non cambiano nel tempo, cioè non hanno stagionalità o trend. Un contatore non è stazionario, ma può essere reso stazionario scrivendo la serie temporale come osservazione(t) – osservazione($t - 1$) [gauge]. La stazionarietà è importante poiché per la previsione abbiamo bisogno di una sorta di invarianza: intuitivamente una variabile della serie temporale è stazionaria rispetto a un qualche percorso di equilibrio se dopo uno shock tende a ritornare su quel percorso, invece, una serie non è stazionaria se si sposta su un nuovo percorso dopo uno shock. È molto difficile modellare il percorso di una variabile che cambia percorso se è soggetta a qualche shock.

6.2.1 Medie e previsioni

Data una serie ci sono diversi modi per prevedere il valore del punto $x + 1$.

- Media semplice: il prossimo punto è la media dei punti della serie.
- Media mobile: come la media semplice ma calcolata sugli ultimi n punti più rilevanti di quelli precedenti.
- Media mobile ponderata: uguale alla media mobile con un peso assegnato a ciascun punto in base alla loro età.

Single Exponential Smoothing

La seguente formula (Poisson, Holt o Roberts) afferma che il valore atteso/livellato (smoothed) è la somma di due prodotti (formula ricorsiva):

$$\hat{y}_x = \alpha \cdot y_x + (1 - \alpha) \cdot \hat{y}_{x-1}$$

α (*smoothing factor*) è un “decadimento della memoria”: indica l’influenza del passato. Il processo per trovare il migliore valore di α si chiama *fitting*.

Nota: questa è essenzialmente una funzione di livellamento piuttosto che una previsione.

In questo caso si fa una previsione sul futuro andando a considerare i valori passati: invece di dare un peso in base alla finestra temporale passata, si vuole dare un peso anche al trend (*double exponential smoothing*).

Chiamiamo \hat{y}_x *level* \mathcal{L}_x e definiamo il trend (o slope) di due punti consecutivi come la differenza tra tali valori: $b = y_x - y_{x-1}$. Otteniamo quindi

$$\mathcal{L}_x = \alpha \cdot y_x + (1 - \alpha) \cdot (\mathcal{L}_{x-1} + b_{x-1})$$

$$b_x = \beta \cdot (\mathcal{L}_x - \mathcal{L}_{x-1}) + (1 - \beta) \cdot b_{x-1}$$

$$\hat{y}_{x+1} = \mathcal{L}_x + b_x$$

dove β è detto *trend factor*.

Nel caso in cui si voglia tenere in considerazione anche la stagionalità dei valori, è necessario introdurre il *triple exponential smoothing* (o *Holt-Winters method*).

Triple Exponential Smoothing

Quando una serie si ripete a intervalli regolari è detta *stagionale*. La durata di una stagione è il numero di punti nella stagione. La componente stagionale è una deviazione rispetto a \hat{y}_{x+1} calcolato nel double exponential smoothing che si ripete per ogni stagione; quindi per ogni datapoint stagionale viene definita una componente stagionale s .

$$s_x = \gamma \cdot (y_x - \mathcal{L}_x) + (1 - \gamma) \cdot s_{x-L}$$

$$\hat{y}_{x+m} = \mathcal{L}_x + m \cdot b_x + s_{x-L+1+(m-1)\%L}$$

dove γ è detto *seasonal smoothing factor* e $S_{x-L+1+(m-1)\%L}$ indica che il seasonal factor deve essere ripetuto in loop per ogni durata della stagione.

Poiché la serie temporale ha una stagione, il numero di pronostici è arbitrario, consentendo di prevedere un numero arbitrario di punti (con errore crescente).

α , β e γ sono i parametri di adattamento dell'algoritmo e $0 < \alpha, \beta, \gamma < 1$. Valori maggiori indicano che l'algoritmo si adatta più velocemente e le previsioni riflettono osservazioni recenti nelle serie temporali; valori più piccoli significano che l'algoritmo si adatta più lentamente, dando più peso alla cronologia passata delle serie temporali. I valori di α , β e γ possono essere determinati cercando di determinare il più piccolo SSE (somma degli errori al quadrato) con un processo iterativo chiamato *fitting*.

6.2.2 Misurare una deviazione

Se Holt-Winters può prevedere un punto, possiamo rilevare anomalie quando l'osservazione si discosta troppo dalla previsione. La deviazione può essere rilevata quando l'osservazione non rientra nelle bande di confidenza minima e massima per un dato punto:

$$d_t = \gamma \cdot |(y_t - \hat{y}_t)| + (1 - \gamma) \cdot d_{t-m}$$

Bande di fiducia

- Banda superiore: $\hat{y}_t + \delta \cdot d_{t-m}$
- Banda inferiore: $\hat{y}_t - \delta \cdot d_{t-m}$

dove δ è detto *confidence scaling factor* e tipicamente è compreso tra 2 e 3.

Parte III

Altro

Capitolo 7

RRDtool as a Communication

7.1 Introduzione

Quello che abbiamo visto fino ad adesso è relativo ai cosiddetti “*elements*”, ovvero ai componenti di rete che in sostanza permettono che funzioni.

Quello che possiamo fare è di studiare se il traffico è più o meno aspettato nei volumi: non possiamo investigare nei contenuti quindi non possiamo sapere il tipo del traffico, ma possiamo conoscerne i volumi. Possiamo studiare lo stato delle porte per sapere se c’è una disconnessione, se c’è una riconnes-sione e sapere se c’è qualcuno che si è spostato o ha abbandonato la rete. Questo, soprattutto per la sicurezza, è un aspetto importante. C’è la necessità di sapere se avviene un cambiamento, di capire il tipo di periferica e di come quest’ultima sia collegata alla rete.

Tramite il Bridge MIB è possibile andare a scoprire dove le device siano effettivamente connesse: per monitorare una rete è importante non solo vedere il traffico ma assicurarci che venga da dove ci aspettiamo. Infine, tramite LLDP è possibile conoscere la topologia di una rete.

I dati raccolti tramite SNMP dove vengono memorizzati?

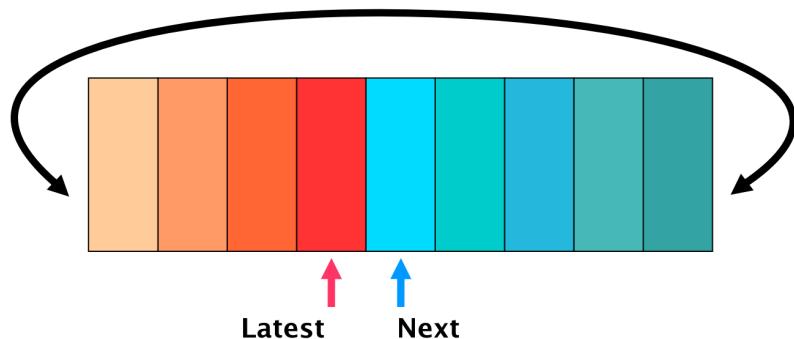
7.2 RRD

I database relazionali sono nati per supportare operazioni di inserimento, cancellazione e modifica dei dati. Nel mondo delle reti, e anche in altri mondi, queste operazioni non sono “normali”: le cancellazioni, come le modifiche, sono proibite; inoltre, quando la cardinalità del numero di righe per tabella aumenta sopra un certo livello, la dimensione dei file contenenti gli indici delle entries, i quali sono utilizzati proprio per velocizzare le operazioni di modifica,

può diventare più grande di quella del contenuto stesso. Se si monitorasse lo stato di una porta a intervalli di un minuto, che ci sia traffico o meno, si scriverebbero 60×24 righe all'interno di una tabella in un solo giorno. Se poi si considera il monitoraggio di un intero switch, è impensabile utilizzare un database SQL per memorizzare tutti i dati raccolti.

È necessario poter consolidare, raggruppare, i dati più vecchi per ottenere molteplici tipi di vista con granularità differente cosicché la dimensione su disco sia minore.

L'RRD (*Round Robin Database*) ha una dimensione di archiviazione prefissata: periodicamente si va a scrivere in uno slot e quando si terminano gli slot a disposizione, si va sovrascrivere il primo, ottenendo la cosiddetta “*sliding window*”.



Componenti di un file RRD

- Header statico: contiene il nome del file, un contatore di byte, un contatore di pacchetti, ...
- Live Header: è dove vengono contenuti gli indici; mantiene l'indice e il contenuto dell'ultimo slot scritto.
- Archivi RRA: spazi che contengono una metrica; quando scrivo sul database, scrivo su tutti gli archivi.

Che cos'è un Data Source? Quali sono le sorgenti dei dati a cui si attinge per scrivere in RRD? “Una qualsiasi cosa che sia numerica”: file di log, contatori SNMP, /proc entries, output da un programma, ...

Gestione di un dato sconosciuto

Se per qualche motivo non avviene una scrittura su un database che dovrebbe essere aggiornato periodicamente, non è possibile usare il valore ‘0’, ma è necessario indicare il valore come **sconosciuto**.

“*Sconosciuto*” è contagioso: $1 + \text{Sconosciuto} = \text{Sconosciuto}$

RRD permette di gestire situazioni in cui si è “persa” una scrittura a causa dello scostamento accumulato nel tempo.

Archivi multipli

I file RRD possono contenere più archivi: è possibile inserire più metriche diverse (byte, pacchetti), oppure la stessa metrica a risoluzione diversa. Al momento della creazione del database va indicato quanto è granulare il dato che sarà inserito e come fare le eventuali aggregazioni.

Capitolo 8

nDPI

8.1 Introduzione

La nDPI è una libreria che permette di fare la **Deep Packet Inspection**: un'attività che analizza i pacchetti di rete oltre agli header di protocollo; con altri tool di analisi dei pacchetti di rete come Wireshark è possibile analizzare principalmente gli header dei pacchetti (dove ci sono IP e porte). L'analisi di Wireshark è molto interessante però non è esaustiva (specialmente in questi ultimi anni): il traffico sta diventando sempre più complicato da gestire perché i protocolli esistenti sono continuamente in decrescita.

Un tempo veniva creato un protocollo per ogni cosa ma, con l'avvento del Web, HTTP è diventato uno dei protocolli predominanti; tuttavia, col tempo HTTP ha perso il senso per cui è stato inventato, ovvero per risolvere gli *hypertext*, ed è diventato una sorta di protocollo di trasferimento. Quindi, dato che i protocolli si stanno appiattendo, cioè vengono impiegati sempre gli stessi e talvolta vengono stravolti dal loro utilizzo originale, è necessario analizzare il traffico in maniera più profonda ovvero andare a guardare il payload. Il tool nDPI permette questo tipo d'analisi.

I motivi per cui si vuole monitorare il payload sono vari, ma il più importante è **classificare il traffico** (per capire se ci sono contenuti malevoli o non consentiti sulla nostra rete). Parlare di HTTP o UDP non ha più alcun senso poiché è troppo generico: è necessario capire cosa sia contenuto nei pacchetti per capire che tipo di traffico attraversi una rete e, di conseguenza, prendere delle decisioni. Ad esempio, potremmo rallentare, bloccare o velocizzare un determinato traffico per proteggere la rete o aumentarne le prestazioni.

8.2 Classificare il traffico di rete

Classificare il traffico non significa spiarlo, ma capirne la natura, sia per sicurezza che per migliorare l'esperienza dell'utente ottimizzando parametri di rete specifici. I principali metodi di classificazione includono

- **Classificazione della porta** TCP/UDP: agli albori di Internet, i protocolli di traffico di rete venivano identificati per protocollo e porta. Può classificare solo protocolli applicativi che operano su porte note (no rpc-bind o portmap): facile da imbrogliare e quindi inaffidabile (TCP/80 ≠ HTTP).
- **Classificazione basata su QoS** (DSCP): simile alla classificazione delle porte ma basata sui tag QoS. Di solito ignorato in quanto è facile da imbrogliare e falsificare.
- **Classificazione statistica**: classificazione dei pacchetti IP (dimensione, porta, flag, indirizzi IP) e dei flussi (durata, frequenza, ...) basato su regole scritte manualmente, o automaticamente utilizzando algoritmi di *machine learning* (ML). Il machine learning richiede un training set di ottima qualità e di solito è intensivo dal punto di vista computazionale. Il tasso di rilevamento può arrivare fino al 95% per i casi che erano coperti dal training set e scarsa precisione per tutti gli altri casi.
- **Deep Packet Inspection**: tecnica che ispeziona il carico utile del pacchetto; computazionalmente intensivo rispetto alla semplice analisi dell'intestazione del pacchetto. Preoccupazioni per la privacy e la riservatezza dei dati ispezionati. La crittografia sta diventando pervasiva, sfidando così le tecniche DPI. Nessun falso positivo a meno che gli strumenti DPI non utilizzino metodi statistici o analisi dell'intervallo/flusso IP.

Usare DPI nel monitoraggio del traffico

L'analisi dell'intestazione del pacchetto non è più sufficiente in quanto è *inaffidabile* e quindi *inutile*. Gli amministratori di sicurezza e di rete vogliono sapere quali sono i veri protocolli che scorrono su una rete, indipendentemente dalla porta utilizzata.

L'estrazione selettiva dei metadati (ad es. URL HTTP o User-Agent) è necessaria per eseguire un monitoraggio accurato e quindi questa attività dovrebbe essere eseguita dal toolkit DPI senza replicarlo sulle applicazioni di monitoraggio.

8.2.1 Welcome to nDPI

Nel 2012 è stato deciso di sviluppare un toolkit GNU LGPL DPI (basato su un progetto non mantenuto chiamato OpenDPI) al fine di costruire un livello DPI *aperto* per applicazioni `ntop` e di terze parti (Wireshark, netfilter, strumenti ML, ...). Ci sono più di 240 protocolli supportati divise in famiglie (è importante sapere la famiglia poiché il protocollo non è sempre conosciuto a livello globale): P2P (Skype, BitTorrent), messaging (Viber, Whatsapp, MSN, Facebook), multimedia (YouTube, Last.fm, iTunes), conferenza (Webex, CitrixOnLine), streaming (Zattoo, Icecast, Shoutcast, Netflix), business (VNC, RDP, Citrix, *SQL).

Cos’è un protocollo in nDPI?

Ogni protocollo è identificato come `<principale>.<minore>`, per esempio: `DNS.Facebook` o `QUIC.YouTube` e `QUIC.YouTubeUpload`. **Nota:** Skype o Facebook sono protocolli nel mondo nDPI ma non per IETF.

La prima domanda che le persone chiedono quando devono valutare un toolkit DPI è: quanti protocolli sono supportati? Ma questa non è la domanda giusta.

Oggi la maggior parte dei protocolli sono basati su HTTP/TLS. nDPI include il supporto per il rilevamento di protocolli basati su stringhe: nome della query DNS, campi di intestazione host/server HTTP, TLS/QUIC SNI (indicazione del nome del server).

Categorie nDPI

I protocolli sono troppi e aumentano ogni giorno; molte persone non hanno familiarità con i nomi di protocollo. Spesso le persone si fanno domande del tipo “Come posso impedire ai miei figli di utilizzare i social network?”. Soluzione: nDPI consente di raggruppare i protocolli in categorie definite dall’utente come VoIP, P2P, Cloud, ... le categorie possono includere migliaia di voci e possono essere (ri)caricate dinamicamente.

nDPI Internals

Le applicazioni che utilizzano nDPI sono responsabili della “cattura (inoltro in modalità inline) dei pacchetti” e del mantenimento dello stato del flusso. In base al protocollo/porta di flusso, tutti i dissettori che possono potenzialmente corrispondere al flusso vengono applicati in sequenza a partire da quello che più probabilmente corrisponde. Ogni dissettore è codificato in un

file “.c” diverso per motivi di modularità ed estensibilità. C’è un file .c aggiuntivo per la corrispondenza IP (ad esempio, identificare il traffico Spotify basato su Spotify AS).

Ciclo di vita della classificazione

In base al tipo di traffico i dissettori vengono applicati in sequenza a partire da quello che ha più probabilità di corrispondere al flusso; ad esempio, per TCP/80 viene provato per primo il dissettore HTTP. Ogni flusso mantiene lo stato per i dissettori non corrispondenti per saltarli nelle iterazioni future. L’analisi dura fino a quando non viene trovata una corrispondenza o dopo troppi tentativi (8 pacchetti è il limite superiore nella nostra esperienza).

Cattura del traffico di rete

Come detto in precedenza nDPI non cattura il traffico, ma ne esegue un’analisi: questo significa che il traffico gli deve essere passato in qualche modo. La libreria ha anche la necessità di dividere il traffico in connessioni, quindi quando gli viene passato un pacchetto gli deve essere specificato anche il suo flusso di appartenenza.

La cattura del traffico di rete che non è locale può essere eseguita tramite un *port mirror* (copia del traffico di rete) oppure si utilizza un *network-tap*.

- **Port mirror:** su una determinata porta dove viene attaccato l’analizzatore di rete, si duplica il traffico (SPAN) della porta che si vuole analizzare (ad esempio tramite nDPI). Se la porta di destinazione non ha una capacità di destinazione superiore (quantomeno doppia) a quella sorgente, alcuni pacchetti possono essere persi; per nDPI questo è un problema visto che analizza solamente la parte iniziale di una comunicazione (se mi perdo dei pacchetti diventa quindi difficile ricostruire il traffico successivo).
- **Network Tap:** il traffico entra ed esce dal tap per poi essere indirizzato sulle porte collegate al dispositivo di cattura/analisi del traffico. Il vantaggio dell’utilizzo del tap è che non si ha la perdita di pacchetti, questo perché si cattura una direzione per volta; il vero problema è che si necessita di due porte per l’analisi (una per ogni direzione del traffico, quindi una per il traffico in entrata e una per il traffico in uscita).