ITRAINONLINE MMTK
NETWORK MANAGEMENT AND MONITORING HANDOUT

Developed by: Alberto Escudero-Pascual, IT +46

Table of Contents

1. About this document

These materials are part of the ItrainOnline Multimedia Training Kit (MMTK). The MMTK provides an integrated set of multimedia training materials and resources to support community media, community multimedia centres, telecentres, and other initiatives using information and communications technologies (ICTs) to empower communities and support development work.

1.1 Copyright information

1.2 Degree of Difficulty

The degree of difficulty of this unit is Advanced.

2. Introduction

Monitoring different aspects of a telecommunication system is a basic requirement in order to ensure the provision of a certain *service.* Understanding what is *valuable information* and being able to *collect the data* from the system are necessary prerequisites to be able to make adequate *decisions*.

Unfortunately, the data itself will not be able to solve your problem; data is not necessarily information and having information does not necessarily give you knowledge.

A good monitor (network management) system should be able to:

- Acquire/collect necessary data from the system
- Process and present the data. Provide different levels of detail of the acquired data
- Take automatic decisions if desired

It is a common mistake in the ISP-world to have a **tool-centric** approach to decision making. For example, when a certain tool is implemented in a network management system, all the decisions are based on the technical possibilities of the tool itself rather than on what the goals and priorities are for that certain telecommunication provider.

This unit takes a **goal-centric** approach to network management. Opposite to the **tool-centric** approach, a methodology that starts by defining clear goals to find the right tools are presented.

3. Goals versus data monitoring

The first and most important thing that a telecommunication/Internet service provider needs to consider, before implementing any kind of monitoring tool, is what goal(s) they want to accomplish and what challenges they face.

Having a (1) **goal** is necessary to think about which (2) **technical principles** are required to acquire the necessary information from the system. Identifying the technical principles make it possible to select, design and deploy certain (3) **tool**. The information provided by the tool should give us some extra knowledge to make a (4) **decision**.

Goals ⇒ Technical Principles ⇒ Tools ⇒ Decisions

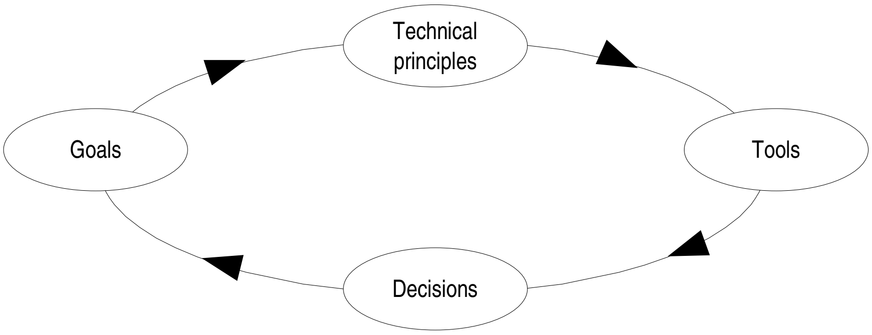


Image 1: The image shows the goal-centric methodology to monitoring (network management).

4. Monitoring service goals

In order to demonstrate this methodological approach to monitoring, three different goals will be presented that are very common in any wireless network deployment:

1. Save costs by reducing International bandwidth use
2. Provide better Quality of Service for VoIP
3. Manage Service and Network Growth

The following tables will show how the three different goals make use of common technical principles and how each of the goals require information-gathering from every layer of the OSI model of our communication system.

In a wireless network, like in any other telecommunication system, goals normally affect most of the layers of the OSI model. To ensure a good service it is necessary to understand not only the wireless-related aspects but the communication network as a whole.

4.1 Save costs in international bandwidth

Layer	***Technical principle***
Application	Caching, Detect/Block Spam/Viruses (**Bayesian Filters**)
Transport	**Traffic shaping (Queueing Principles)** **Traffic accounting (SNMP, Promisc)**
Network	Network Access Control (Firewalling) Traffic shaping Traffic accounting (SNMP, Promisc)
Media Access Control	**Wireless Access Control** Collect Wireless Layer-2 Data (**SNMP**)

Table 1: Technical principles (and corresponding OSI layers) that can be used to save international bandwidth.

4.2 Provide better QoS for VoIP

Layer	***Technical principle***
Application	
Transport	**Traffic shaping (Queueing Principles)** **Traffic accounting (SNMP, Promisc)**
Network	**Traffic shaping (Queueing Principles)** **Traffic accounting (SNMP, Promisc)**
Media Access Control	Collect Layer-2 Data (SNMP) Reduce wireless latency

Table 2: Technical principles (and corresponding OSI layers) that can be used to provide better QoS for VoIP systems.

4.3 Manage service and network growth

Layer	***Technical principle***
Application	Virus/Spam, SQL **(Service Balancing)**
Transport	Collect TCP/UDP Statistics, Firewall Balancing
Network	Collect IP Layer Statistics, **Routing principles**
Media Access Control	Collect Layer-2 Data (SNMP)

Table 3: Technical principles (and corresponding OSI layers) that can be used to manage service and network growth.

Failing to optimize any of the protocol layers will affect the overall service performance. For example: a high level of corrupted packets in the wireless network will have an impact in the overall TCP performance and high latency will be perceived by the user running a real time application.

The complexity not only comes from understanding each of the communication layers involved in a system, but also the interrelationships between them.

As with any other communication system, making a wireless network work is easy, but making a wireless network with good performance will take you time and experience. That is the principle that makes the Internet work today!

5. Technical principles

Before discussing available tools for network (management) monitoring, the technical principles behind the tools will be explained. Some technical principles can be implemented in many different forms to achieve different goals. Tools normally implement a subset of possibilities of a technical principle.

Understanding the technical principles will help you not only to choose the right tool but also to design a new tool if the ones available does not match your requirements.

5.1 SNMP

SNMP (Simple Network Management Protocol) is an operation and maintenance protocol specially designed for computer networks and individual network devices.

The first version of SNMP (SNMPv1) was developed by IETF back in 1993. SNMPv3 is the current standard but many (wireless) devices still only support old versions of the protocol.

Information gathering is based on a client/server architecture where a client program queries remote network devices for statistical information and private data. SNMP is the application layer protocol that is used for exchanging information.

Each of the SNMP-enable devices implements a database called MIB (Management Information Base). The database contains information gathered during the operation of the device. SNMP, in simple words, is the mechanism to Send Requests and Receive Responses about management information from active elements of the network.

The greatest strength of SNMP, which has given the protocol its widespread popularity, is its **support for interoperability** between different network devices. Network devices that supports SNMP agents spans from routers, computers and bridges to modems and printers.

Also, SNMP is flexible enough so it can be extended with device specific data. Many wireless vendors implement a **proprietary set** of wireless information data in their MIBs. Unfortunately, that means that although all the vendors implement SNMP, the mechanism to retrieve certain kind of wireless data can vary.

Vendors of wireless equipment normally provide their customer with their own “management tool” that uses SNMP to communicate with their wireless devices. Integration of different vendor management tools is normally very complicated as the code very seldom is open source. The best option can sometimes be to write your own wireless management system.

SNMP has also many other weaknesses. The protocol is not straightforward for programmers due to its complex encoding rules and it has also been criticized for being inefficient and wasting bandwidth. Every SNMP packet includes several useless data fields and SNMP variables are encoded in ways that makes that data packet unnecessary large.

When implementing any kind of monitor system based on SNMP you should consider:

1. SNMP is also “traffic” in your network, try to minimize overhead by making smart queries.
2. SNMPv1 does not provide an encrypted authentication. Mind your passwords.
3. SNMP consumes CPU cycles of your network devices.

5.2 Traffic accounting

Traffic accounting is a general technical principle for monitoring traffic statistics in computer networks. Information generated by traffic accounting is of great value for making network organization decisions, troubleshooting LANs and monitoring various hosts' activities.

Typical information that traffic accounting retrieves is

- packet and byte counts

- protocol distribution statistics (type, times, %)

- IP checksum errors
- discovery of active hosts
- data activity among hosts

There are many ways to gather traffic related information in a network. The most common approach is to enable SNMP in all routers and bridges of the network. It is interesting to see SNMP as an **active** way to obtain traffic-data related information. It is *active* because it is required to exchange SNMP traffic with the routers/bridges to obtain that information.

Another possibility to obtain traffic-related information without the need of sending any extra traffic over the network can be to **promiscuously** listen into the communication channel. Listening to the channel is a **passive** mechanism that does not involve the use of SNMP at all. However, there are two limitations to the second approach; you need to have direct access to data in the communication channel and the CPU/processing power to gather and digest the volume of information in the channel.

5.3 Traffic shaping

Traffic shaping is a method to *control the flow of traffic* in a network in order to optimize or guarantee a certain performance in the network. Traffic shaping is a result of enforcement of the packet queue disciplines in the routers. By managing those rules wisely, you can adjust the behaviour of your network concerning:

1. latency and congestion management
2. bandwidth management and fairness

Traffic shaping is normally done in the IP layer, by changing the ways that packets are queued and delivered in the routers. By shaping traffic in the IP layer, we also affect the distribution of resources in the radio channel.

It is important to mention here that some IEEE 802.11-based products have tried to implemented similar mechanisms in the wireless bridges by modifying the default behaviour of IEEE 802.11 MAC layer. Most of those mechanisms remain proprietary and interoperability between vendors is not ensured.

For example, Proxim implements a proprietary polling mechanism (WORP) that supports Asymmetric Bandwidth Throttling which allows adjustments to the data rate users receive and send. WORP allocates network capacity by assigning brief time slots to all the users that want to send and receive data and giving each of them a turn to use the bandwidth.

5.3.1 Queuing disciplines and latency

If you want to maintain interoperability between vendors and do not want to implement proprietary mechanism in the network, traffic shaping needs to be done in the IP level.

Traffic shaping is done by affecting the way we queue and deliver traffic in the so called active-elements of the network. Queuing disciplines are the rules that are applied on packets of data during their forwarding process. A queuing discipline implies different rules depending on the priority of the packet, the sender of the packet or the status of the queue in that moment.

Normally in a network, the queuing disciplines of the outgoing traffic are of much greater importance than the incoming ones. Since the bottleneck of a network normally is the uplink, (the link to the Internet connection) where the traffic of the local LAN is squeezed into one single pipe, the queuing disciplines of the outgoing traffic must be carefully managed.

A packet queue is a buffer that stores data packets while the amount of received packets in the router exceeds its ability to send. When the buffer is full, the router need to drop further incoming packets which results in retransmissions.

A packet that gets stuck in a queue too long, becomes timed-out and will force the sender to resend the packet, causing even more traffic to handle for the overloaded router.

The result of *overloaded buffers* is that the network will have a *very high latency* when the uplink becomes congested. Retransmissions due to time-outs can make a bad situation even worse.

One way to reduce the inconvenience of the latency, is to prioritize some packets higher than others. Packets that require user interaction (such as remote consoles, online gaming/chatting, VoIP etc.) get higher priority than protocols like HTTP, FTP and SMTP that rather require high bandwidth than low latency. This priority policy can be implemented by applying rules depending on service (which TCP port the packet is using).

In cases when several protocols are using the same TCP port as default, like SSH and SCP, another queuing policy must be applied. SSH and SCP are two protocols that both use TCP port 22 as default. SSH packets are normally very small (containing just a few key strokes) while a SCP packet can be much larger. SSH requires user interaction while SCP doesn't which might make you want to apply different policies on them. The mechanism to affect differently services that use the same port number is by setting different priorities based on packet sizes.

5.3.2 Bandwidth management by packet queueing

Bandwidth management by packet queuing can ensure Quality of Service (QoS) in your network. Bandwidth management implies limiting the bandwidth to a certain bit rate to a specific host or subnet or limiting the throughout for a specific type of service.

Bandwidth management can be implemented for several reasons. Maybe you don't want a large download to affect the QoS for other hosts in the network that are requesting lower bandwidth. Or, you might want to strive for a fair network where all hosts get throughput according to what they pay for the service.

To accomplish this, *classful and/or classless queuing disciplines* can be implemented in the routers.

A **classful queuing** discipline has a hierarchical structure with parent/child relationships (based on classes) and heritage of characteristics. Each host belongs to a class where characteristics as maximum bandwidth, queuing algorithm, ceiling limit (to borrow bandwidth) and port numbers are specified.

With bandwidth management and classful queuing, a specific bit rate can be allocated to a certain protocol depending on its type (TCP port). Also, different hosts (belonging to a certain class) can be allocated a certain bit rate to create fairness in the network.

One of the most popular classful queuing discipline is HTB (Hierarchical token bucket) that is used to distribute the data packets to different branches of the hierarchical tree model depending on a certain priority and bandwidth. Subclasses can be created to allow unused bandwidth to be shared between class members (children of the same subgroup). HTB is used to control the outbound bandwidth on a given link. By applying queuing rules on each incoming packet, it uses one physical link to simulate several slower links and to send different kinds of traffic on different simulated links.

Opposite to a classful discipline where traffic is divided in classes, a classless queueing disciplines are those that just reschedule, delay or drop data.

SFQ (Stochastic fairness queue) is a popular **classless discipline** that is used when the outgoing link is full. It does not provide any shaping of the traffic but only schedules the transmission or the packets so that *all connections get an equal share* of the bandwidth. This discipline has no effect on the traffic when the link is not full.

The SFQ queueing discipline attempts to distribute the opportunity to transmit data to the network among an arbitrary number of *traffic flows in a fair way*.

5.4 Bayesian filters[1]

Another technical principle that can be used to improve the service of your network is the use *Bayesian filters.* Bayesian Filters can be used to implement anti-spam systems that calculate the probability of a message to be spam based on its content.

Bayesian Filters is based on the idea that spam can be *filtered out* based on the probability that certain words or combination of words can identify a message as spam while other words will identify a message as *legitimate* content.

Bayesian Filters are adaptive which means that they learn from its experience to distinguish between good and bad content and they become more “intelligent” and more robust by time.

Bayesian filters are content based as many other spam filters. One of the differences is that the *manual list* of spam characteristics, that is the weakness of many content based filters, is eliminated and replaced by a list of words that the filter itself has created by analysing known content. The initial list is created by analysing a set of known spam messages and a set of known non-spam messages to build up a first classification to distinguish good content from bad.

The classification of good content is just as important as the classification of a bad content. The better the filter is adapted to the individual user the harder will it be for the spammer to get around the spam filter.

After creating the initial list of characteristics, the list will grow as the filter is being used. The filter will be personally adopted to the user since it learns from what the user reports as mistakes based on his/hers incoming mail.

Unlike other content based filters, Bayesian filters investigate the whole content of the message while others just examine the header and the subject field. Except from the message body, it also examines other parts of the message as:

- header (sender and message paths)
- embedded HTML code (colours etc.)
- word pairs and phrases
- meta information

An interesting option worth considering to place your anti-spam filters before the mails even transverse your wireless infrastructure. Placing mail rely agents with anti-spam features in the other side of your international link can bring 10-20% international bandwidth saving costs.

5.5 Viruses fingerprints

Anti-virus software has the ability to detect viruses and other forms of malicious software and remove them. The detection is done by examining the executable programs and documents and look for specific computer instructions that are used by known viruses.

Those computer instructions or a derivative of them are called the virus “fingerprint” or “signature”. The use of fingerprints is one of the technical principles that allows an anti-virus program to be able to search for known patterns in suspicious code.

For a anti-virus program to be successful, it requires that a database with virus fingerprints is constantly updated.

Unfortunately, detecting malicious code is not simple and virus code can mutate and modify themselves to change their fingerprints. Heuristic scanning algorithms, which test various permutations of known virus definitions, are used to predict and analyse how a virus might mutate and detect the new virus before it can spread further.

[1] The name “Bayesian filtering” comes from the English mathematician Thomas Bayes

Do not underestimate the impact of malicious code in the overall performance of your wireless network. Being able to identify and remove viruses in your customers e-mail or block traffic from infected host are activities as important as having a good signal/noise ratio in your wireless links.

6. Tools

This section takes a closer look at some of the free and open source tools that can be used to monitor your network. Some of the tools also include mechanisms to take concrete “actions” when a non-desire behaviour takes place.

The tools that will be presented are network monitoring tools (ntop, MRTG), spam filters (SpamAssassin) and anti-virus programs (CLAM AV).

6.1 Monitoring the “wireless”

Every vendor of wireless equipment provides a tool to monitor the activity in their devices. Configuration and monitoring is done via a specific software that runs in a *certain operating system.*

Unfortunately, when building and operating large wireless networks, you will find those monitoring tools very limited. The tools that come with the wireless equipment can not be integrated in your network management system and when different equipment is implemented in the network you will end up with n different tools running in your desktop in order to grasp the picture of what is going on in your wireless network.

One possibility to integrate different management tools into one single interface, is to try to obtain information by the use of SNMP MIB in each of the wireless products and put all the data together using auxiliary tools like MRTG.

6.2 MRTG

MRTG (Multi Router Traffic Grapher) is a Web-based network management tool that can monitor and display the evolution of network parameters during the time. MRTG uses SNMP to gather information from different SNMP-enabled routers and a set of graphical libraries to build information graphs.

MRTG was originally designed to present graphs of traffic load and bandwidth utilization but has been extended so it can represent almost any parameter that is changing during the time.

6.3 Monitoring wireless parameters using MRTG

In the following example, the basic principles to build your own wireless monitoring system using SNMP and MRTG will be explained. The example is based on the *Orinoco family* of products but the methodology can be applied to any other wireless devices.

Let's assume that we have a point to point (PtP) wireless link and we want to monitor the total bandwidth usage (Layer 3) and the status of the radio link (Layer 2).

Monitoring the wireless bridge to obtain the bandwidth use is as simple as monitoring the bandwidth use of any other SNMP-enable device (router, switch etc.)

The steps to configure MRTG can be summarized as follows:

1. Make sure that you have all the pre-requirements: a web server running, MRTG installed, the IP address and SNMP password of the device that you want to monitor.
2. Create a configuration file for MRTG. This step can be done manually or using auxiliary tools as *cfgmaker*
3. Create a "cron" process that runs MRTG using the configuration periodically

6.3.1 Bandwidth monitoring

After installing (1) MRTG and Apache Web Server, we will use the tool **cfgmaker** to create a (2) default configuration file for mrtg:

The default configuration file is obtained by running:

```
[aep@it46-d505 mrtg2]$ cfgmaker password@IP  > /etc/mrtg_b.cfg
```

where <password> and <IP> are the read-only SNMP password and IP address of the wireless bridge respectively.

The only change needed is the default configuration file *mrtg_b.cfg* to reflect the place we want the MRTG web pages to be available from:

For example:

```
WorkDir: /var/www/mrtg
```

indicates that all MRTG web pages and graphs will be placed in the /var/www/mrtg directory

Finally, we need to create a periodic task by adding the line /etc/crontab as follows:

```
*/5 * * * * root  /usr/bin/mrtg /etc/mrtg_b.cfg
```

MRTG will poll data from the wireless bridge every five minutes.

6.3.2 Signal/noise ratio monitoring

To monitor the signal/noise ratio of the wireless device we need to have access to the MIB of the wireless product. The MIB of the product will tell us which entries in the database that contain the information we are looking for.

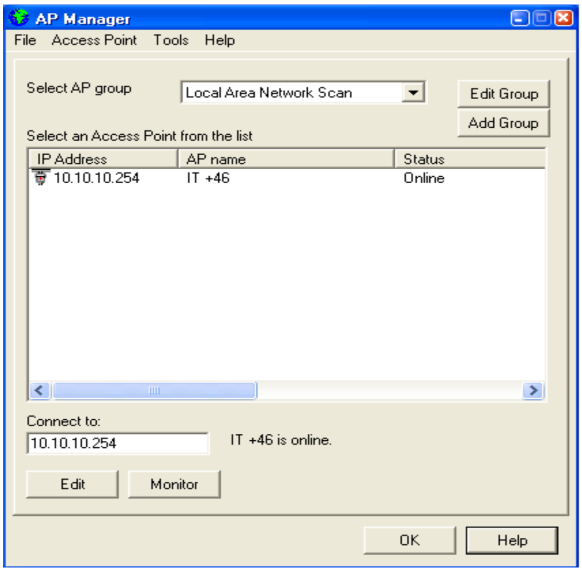
Having access to the MIB of a wireless device is not always easy. Most of the wireless parameters that you might want to monitor are part of the "proprietary" MIB, an extension included by the wireless product vendor and very seldom publicly documented.

If you do not have access to the MIB you might need to find it out by yourself (i.e. reverse engineering).

We need to find out which OIDs are used in the MIB for the information that we need to monitor. An OID (Object Identifier) is a number that identifies an object's position in the MIB. An OID is the way that we can refer to certain positions of the management "database".
When a monitor tool requests information to the wireless device, it performs a set of SNMP operations using certain OIDs. The OIDs indicate which part of the MIB we want to read or write data.

The best way to obtain the OIDs that we are looking for is to take the networking tool that comes with the product and monitor all the traffic that is exchange between the monitor tool and the wireless device.

In the example, the **Orinoco AP Manager** is installed:



Select AP group: Local Area Network Scan

Select an Access Point from the list

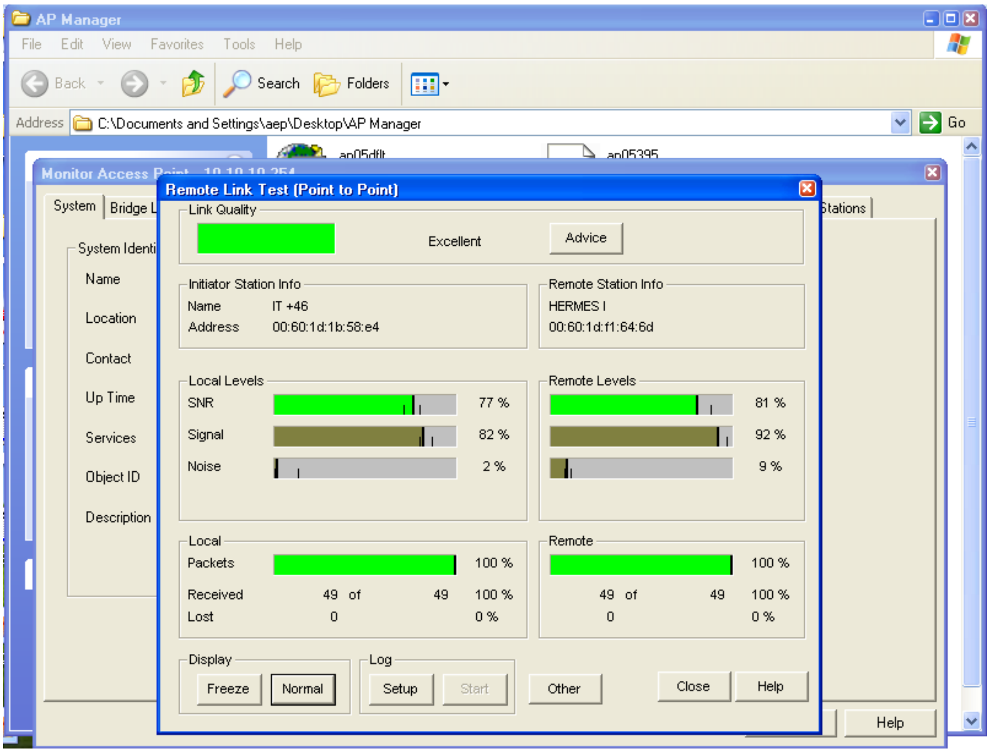
IP Address	AP name	Status
10.10.10.254	IT +46	Online

Connect to: 10.10.10.254

IT +46 is online.

Image 2: Using a Windows tools for monitoring an access point.

After connecting to the wireless access point, we choose the option “perform a **link-test**” and record all the traffic between the AP and the monitoring tool.



Address: C:\Documents and Settings\aep\Desktop\AP Manager

Remote Link Test (Point to Point)

Link Quality: Excellent

Initiator Station Info		Remote Station Info
Name	IT +46	HERMES I
Address	00:60:1d:1b:58:e4	00:60:1d:f1:64:6d

	Local Levels	Remote Levels
SNR	77 %	81 %
Signal	82 %	92 %
Noise	2 %	9 %

	Local				Remote			
Packets				100 %				100 %
Received	49	of	49	100 %	49	of	49	100 %
Lost	0			0 %	0			0 %

Image 3: Link-test of a Ptp measuring SNR and packet loss.

Using any of the traffic analysis tools (tcpdump, ethereal etc), we can obtain traffic exchange between the AP and the AP manager that looks like this:

```
19:41:21.448323 10.10.10.12.1260 > 10.10.10.254.snmp:  GetRequest(29)  .1.3.6.1.4.1.762.2.1.7.0
0x0000   4500 0048 77b2 0000 8011 99d5 0a0a 0a0c        E..Hw...........
0x0010   0a0a 0afe 04ec 00a1 0034 64bb 302a 0201        .........4d.0*..
```

```
0x0020   0004 0670 7562 6c69 63a0 1d02 0201 0302        ...public.......
0x0030   0100 0201 0030 1130 0f06 0b2b 0601 0401        .....0.0...+....
0x0040   857a 0201 0700 0500                            .z......
19:41:21.448854 10.10.10.254.snmp > 10.10.10.12.1260:  GetResponse(30)  .1.3.6.1.4.1.762.2.1.7.0=2 (DF)
0x0000   4500 0049 0037 4000 4011 1150 0a0a 0afe        E..I.7@.@..P....
0x0010   0a0a 0a0c 00a1 04ec 0035 62b5 302b 0201        .........5b.0+..
0x0020   0004 0670 7562 6c69 63a2 1e02 0201 0302        ...public.......
0x0030   0100 0201 0030 1230 1006 0b2b 0601 0401        .....0.0...+....
0x0040   857a 0201 0700 0201 02                         .z.......
```

The Lucent proprietary MIB that collects all the wireless related information uses the 1.3.6.1.4.1.762.2.5.* MIB variables. This MIB is common to many wireless access points like the Apple Airport and Lucent RG1000.

The **number of users connected to AP** can be retrieved by performing the following SNMP operations:

Write Integer 50 in OIDs:
1.3.6.1.4.1.762.2.5.5.1, 1.3.6.1.4.1.762.2.5.5.1, 1.3.6.1.4.1.762.2.5.5.3
Write Integer 3 in OIDs:
1.3.6.1.4.1.762.2.5.4.1, 1.3.6.1.4.1.762.2.5.4.2, 1.3.6.1.4.1.762.2.5.4.3
Retrive the OID:
1.3.6.1.4.1.762.2.5.1.0

Signal Noise Parameters of a wireless device <n> can be obtained by:

Write Integer 1500 in OID
1.3.6.1.4.1.762.2.5.2.1.27.n
Write Integer 25 in OID
1.3.6.1.4.1.762.2.5.2.1.26.n
Write Integer 80 in OID
1.3.6.1.4.1.762.2.5.2.1.25.n
The signal can be retrieved by reading the OID
1.3.6.1.4.1.762.2.5.2.1.32.n
The noise can be retrieved by reading the OID
1.3.6.1.4.1.762.2.5.2.1.33.n

Once we have obtained the necessary OIDs/MIB related information we need to write a script that performs similar SNMP operations to the Windows AP Manager (Do not forget that the whole idea is to get rid of AP manager and have the possibility to integrate the data in one single network management system!!).

A script that retrieves the link test information written using Linux **snmp-tools** is included in the Appendix of the unit.

The script collects the signal/noise values by SNMP and returns those values in a format that MRTG can use:

```
[aep@it46-d505 etc]$ /usr/local/bin/monitoring_PtP.sh
79
12
2:596
aep
```

The four values are the signal (79), the noise (12), a timestamp (2:596) and the name of the host (aep).

Now we are ready to put together in the same interface the bandwidth data (IP information) and the signal/noise data (wireless information)

By representing both the “bandwidth” and the “signal-noise ratio” in the same network management system interface, we get a wider picture of what is happening in network.

The picture shows the **integration** of wireless-monitoring with IP traffic monitoring with MRTG

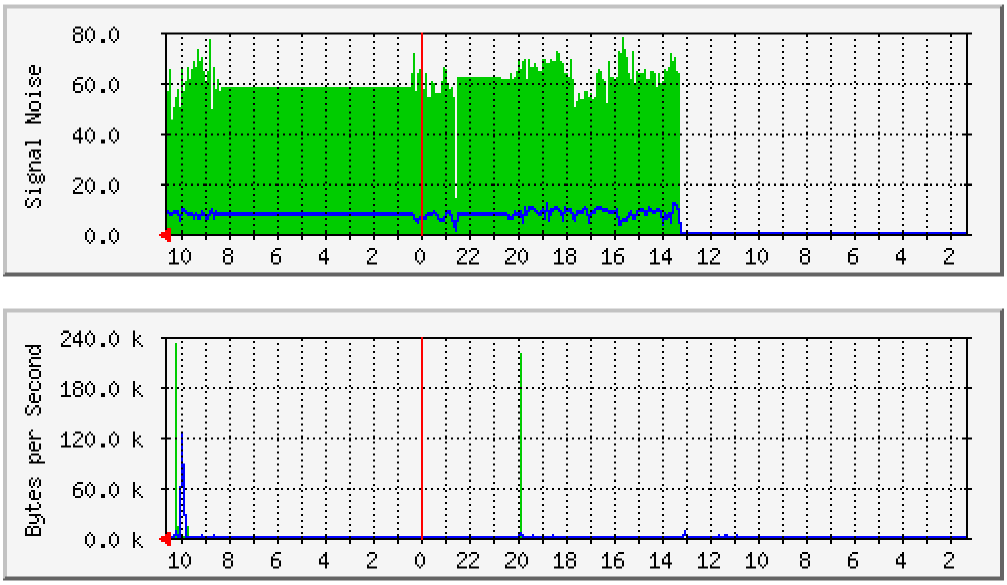


Image 4: Using MRTG to display SN in a wireless link and bandwidth usage simultaneously.

(Note: Discuss what information we extract from the diagrams)

6.4 Ntop

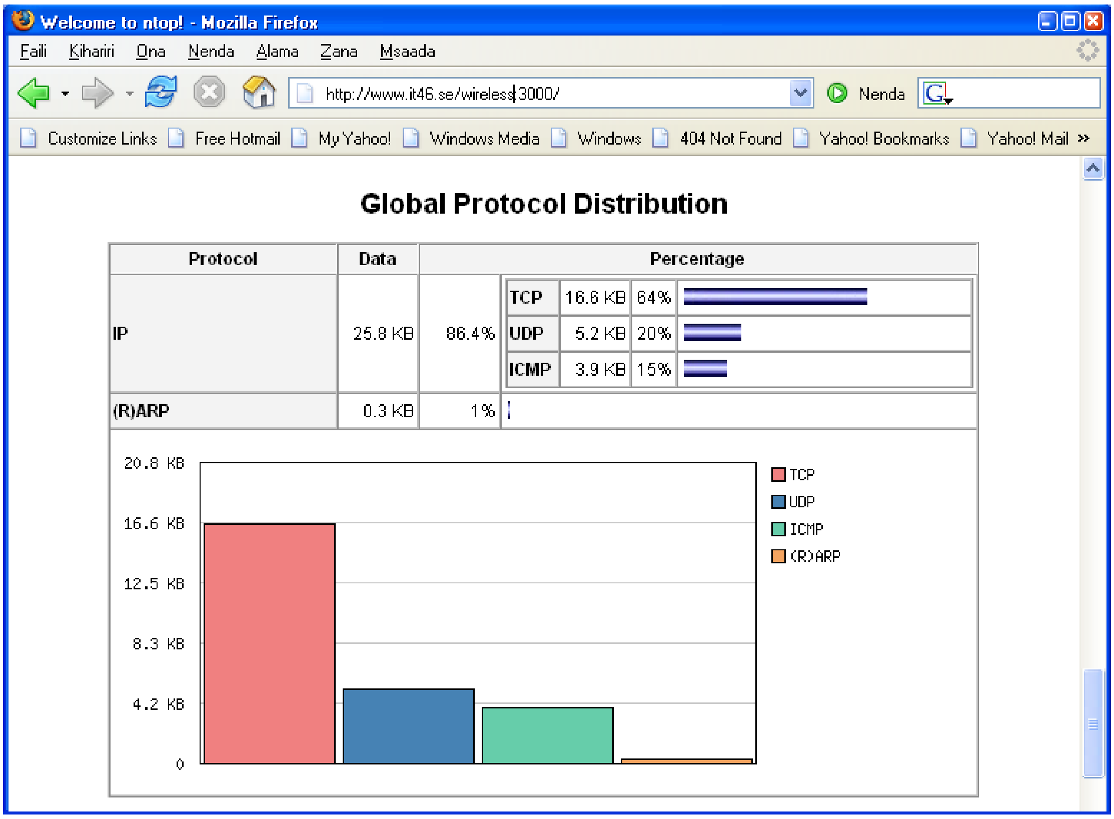
Ntop is a multi-platform free and open source ***IP traffic*** measurement and monitoring application. The whole functionality of Ntop (configuration and monitoring) is available via a Web Interface.

By IP traffic measurement tool we mean that no wireless-specific information is recorded by Ntop. Ntop does not record any wireless related information as Signal/Noise, number of associated stations etc. Ntop should be combined with a Layer-2 specific monitoring tool as the one described before.

Ntop functionality focus on:

- Traffic measurement
- Traffic characterization and monitoring
- Detection of network security violations
- Network optimization and Planning

A detailed description of Ntop's rich features is included in the appendix of the unit.



Global Protocol Distribution

Protocol	Data	Percentage				
IP	25.8 KB	86.4%	TCP	16.6 KB	64%	
			UDP	5.2 KB	20%	
			ICMP	3.9 KB	15%	
(R)ARP	0.3 KB	1%				

Image 5: Using Ntop to monitor the protocol distribution for a link.

6.5 Spam-assassin

This unit includes a special section about fighting spam. Spam has become one of the biggest nightmares in network management. Keeping your users away from SPAM is a mandatory goal for any service provider.

One of the most famous and used anti-spam programs is SpamAssassin. SpamAssasin is an intelligent spam filter that uses a number of different methods to differentiate *spam* from *desired* mail. SpamAssassin can be used by both email clients and servers to filter both incoming and outgoing mail.

SpamAssassin does not normally *block* suspicious mail but *tag* them and assign them a score depending on its content. The higher score that a message gets the higher probability that the message is classified as SPAM. In any case, it is up to the client to delete the messages tagged as SPAM.

One of its many strengths is its modular architecture that allows integration of new technologies into the filter and that it can be implemented in almost any email system.

Its primary methods for analysing email and determining its probability to be spam are the following:

- Header tests (sender, subject string)
- Body phrase tests with third-party rule sets (keywords, html code, IP addresses, URL)
- Bayesian filter
- Automatic address whitelist/blacklist
- Manual address whitelist/blacklist
- Collaborative spam identification databases
- DNS blocklists (RBL=RealTime Blackhole Lists)
- Character sets and locales

6.5.1 Collaborative spam identification database

One mechanism to detect spam is to use a *collaborative spam identification database.* A collaborative spam identification database is an online database of spam checksums that anyone can query for identifying a spam.

By calculating a checksum of a suspected spam message that arrives to the mail server, the online database can verify if that checksum has been reported as spam by anyone before. If that is the case, the score of the message rises.

SpamAssassin supports three different identification databases:

- Razor, http://razor.sourceforge.net
- Pyzor, http://pyzor.sourceforge.net
- DCC (Distributed Checksum Clearinghouse), http://www.rhyolite.com/anti-spam/dcc/

6.5.2 DNS blocklists

DNS Blocklists are another form of online databases to detect spam. They are also known as DNS Blacklists (DNSBLs) that lists the IP addresses of mail servers known (or believed) to be used by spammers.

Some settings of a mail server that DSNBLs reacts on to identify spam are:

- open SMTP relay
- open proxy
- open form to mail HTTP gateways
- dynamic IP pools

An open mail relay is an email server (mis)configured to allow anyone to relay (send) mail through it.

Nowadays, most ISPs are no longer using open relays to allow remote access to their customers. Instead solutions as SMTP AUTH and POP before SMTP are used. This means that the spammers have adopted other techniques to to send spam.

A interesting resource about DNSBL is available here:
http://www.scconsult.com/bill/dnsblhelp.html

6.6 Clam Antivirus (Clam AV)[2]

Computer Viruses can shut-down a wireless network in question of minutes. Many computer viruses trigger/perform network scanning or denial of service. The first consequence is that the wireless links get flooded. Viruses can make a VSAT link useless if access-lists are not placed quickly.

Clam AntiVirus is a GPL anti-virus toolkit for UNIX designed for email scanning on mail gateways. Clam AV supports automatic virus fingerprint updating. The fingerprint database is updated via the Internet.

Some of the features Clam Antivirus includes are:

- Fast and powerful scanning of directories and files
- Detection of over 30000 viruses, worms, and Trojans (including Microsoft Office and MacOffice macro viruses)
- Scans archives and compressed files with built in support for compression algorithms as Zip, Rar (2.0), Tar, Gzip, Bzip2, MS OLE2, MS Cabinet Files, MS CHM (Compiled HTML), MS SZDD compression format

[2] http://www.clamav.net/, Last accessed: 20050223

- Supports Portable Executable files compressed with UPX, FSG and Petite
- Advanced database updater with support for virus digital signatures and DNS based database version queries

Clam Antivirus **does not** delete, rename or clean the infected file. It simply detects and warns the user by tagging the message or similar.

7. Conclusions

Management of a network (wireless or not) requires that we start by defining our **goals** as a service provider. If we do not know what we want to achieve it is very difficult to decide what we want to monitor. If we do not know what to monitor is very difficult to find the tools that will help us to take decisions.

The five main issues you should remember from this unit can be summarized as:

1. *Collecting raw data* or *installing tools* is not enough to have a good operational network. Finding the adequate tools should not be a problem if you have a clear picture of what you need.
2. Monitoring the status of the physical radio links will not be enough if you network is flooded by virus activity. Stopping viruses will not be enough if your wireless links are not stable.
3. When implementing monitoring and/or network management systems you will need to learn how to integrate different available tools so all information you need in order to take decisions is easily available.
4. Think that if the tool does not satisfy your needs, it might be easier to write a simple tool that matches your needs rather than fighting with a tool that fulfils the needs of someone else.
5. Set up your goals ⇒ Identify technical principles ⇒ Find/Design tools ⇒ Make decisions.

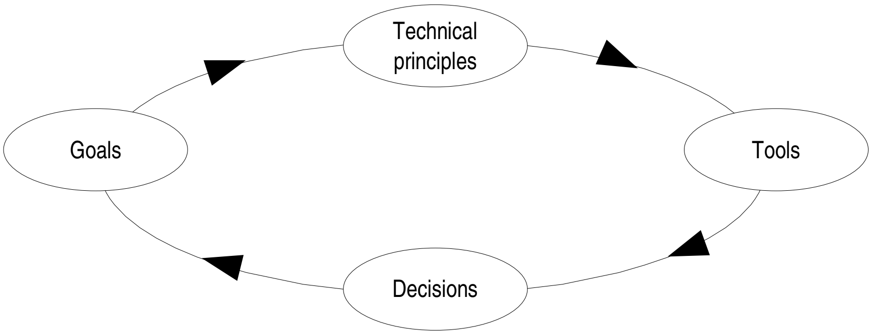


Image 6: The image shows the goal-centric methodology approach to monitoring (network management).

8. Appendix 1

NTOP Traffic Measurement

Ntop associates each captured packet with its sender/receiver. In this way, all activities that are connected to a single host can be extracted by knowing the host's name, NIC or IP address.

For each host, the following information can be retrieved by ntop:

- Sent and received data: Total amount of traffic generated or received (volume and packets) classified according to network protocol (IP, IPX, AppleTalk etc.) and IP protocol (FTP, HTTP, NFT etc.)
- IP multicast: Total amount of multicast traffic generated and received by the host.
- TCP session history: List currently active TCP sessions initiated or accepted by the hosts and associated traffic statistics.
- UDP traffic: Total amount of UDP traffic sorted by port.
- TCP/UDP services used: List IP based services provided by the host and the last five hosts that used them
- Operating System used by host
- Used bandwidth percentage (actual, average and peak)
- Traffic distribution (among subnets)
- IP traffic distribution (UDP vs TCP, relative distribution of IP protocols)

Ntop also gathers global information (not host oriented) regarding traffic distribution, packet distribution and bandwidth utilization.

NTOP traffic characterization and monitoring

Traffic monitoring implies identifying situations where network traffic does not following the “rules” or thresholds set by the administrator of the network. Ntop can discover the following situations:

- Duplicate use of an IP address
- Identification of all subnet routers
- Identification of all hosts that have their NIC set to promiscuous mode
- Detect misconfiguration of software applications
- Service misuse detection (such as hosts that don't use specified proxies)
- Protocol misuse (identification of hosts that use unnecessary protocols such as NetBEUI and IXP)
- Detection of hosts with excessive network bandwidth utilization

NTOP detection of network security violations

Most security attacks in a network comes from the network itself and not from external users. Ntop provides its users with support for tracking ongoing attacks and identifying security holes in their computer by offering the following features:

- “Portscan” and “slow portscan” detection: ntop reports name of the last three hosts that sent a packet to each post less than 1024. Portscan is detected in all hosts that ntop is monitoring.

- Spoofing detection (for packets belonging to the same subnet where ntop is running): Spoofing means that a host claims to be another host in the purpose of intercept packets. Ntop warns the user when two distinct IP addresses maps to the same hardware in the subnet.
- Spy detection: A spy is a host with its network card in promiscuous mode which enables capturing of packets independent of its destination.
- Trojan horse: Trojan horse appear to be friendly code but contains hidden malicious code that can destroy your computer. Trojan horses normally make use of well know ports. Ntop can therefore detect their presence by monitoring the traffic for those ports.
- Denial of Service (DoS): DoS is the behaviour of a host that sends packets with the SYN flag set (to open TCP connections) to the ports of the "victims" without proceeding the connection procedure. Eventually, the "victims" IP stack connections slots are all occupied and the host can not accept any further connection.

NTOP network optimization and planning

Suboptimal configurations of hosts and non-efficient utilization of available bandwidth lead to decreasing network performance. Ntop provides the following support to enhance the network performance:

- Identify unnecessary protocols (hosts that uses protocols that is not in use in the network)
- Identify suboptimal routing by keeping track of ICMP Redirect messages and analyse list of subnet routers
- Traffic characterization and distribution by studying traffic patterns
- Wiser bandwidth use: Studying the distribution of traffic among protocols can help the administrator to identify applications that need web proxies.

NTOP supports SQL databases if the user wish to save data gathered from a monitoring session.

/etc/mrtg_b.cfg

```
######################################################################
# Multi Router Traffic Grapher – Orinoco PtP signal/noise monitoring
######################################################################

# Global configuration
WorkDir: /var/www/mrtg
WriteExpires: Yes

Interval: 5
Target[load]: `/usr/local/bin/read_signal_noise.sh <password> <IP>`
Title[load]: SIGNAL NOISE
PageTop[load]: <H1>Signal Noise PtP</H1>
Options[load]: gauge,nopercent,integer
YLegend[load]: Signal Noise
ShortLegend[load]: -dbm
MaxBytes[load]: 100
LegendI[load]: Signal
LegendO[load]: Noise
```

read_signal_noise.sh <password> <IP>

```
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.5.1 i 50 >/dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.5.2 i 50 >/dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.5.3 i 50 >/dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.4.1 i 3  >/dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.4.2 i 3  >/dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.4.3 i 3 >/dev/null
users=`snmpget -c public -v 1 10.10.10.254 1.3.6.1.4.1.762.2.5.1.0 | awk '{print $4}'`
#echo The number of users is $users
#echo "TESTING LINK...."
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.2.1.27.1 i 1500 > /dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.2.1.26.1 i 25 > /dev/null
snmpset -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.2.1.25.1 i 8000 > /dev/null
signal=`snmpget -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.2.1.32.1 | awk '{print $4}'`
noise=`snmpget -c $2 -v 1 $1 1.3.6.1.4.1.762.2.5.2.1.33.1 | awk '{print $4}'`

#Return values for MRTG
echo $signal
echo $noise
UPTIME=`uptime | awk '{print $3$4}' | sed -e "s/,//g"`
echo $UPTIME
```