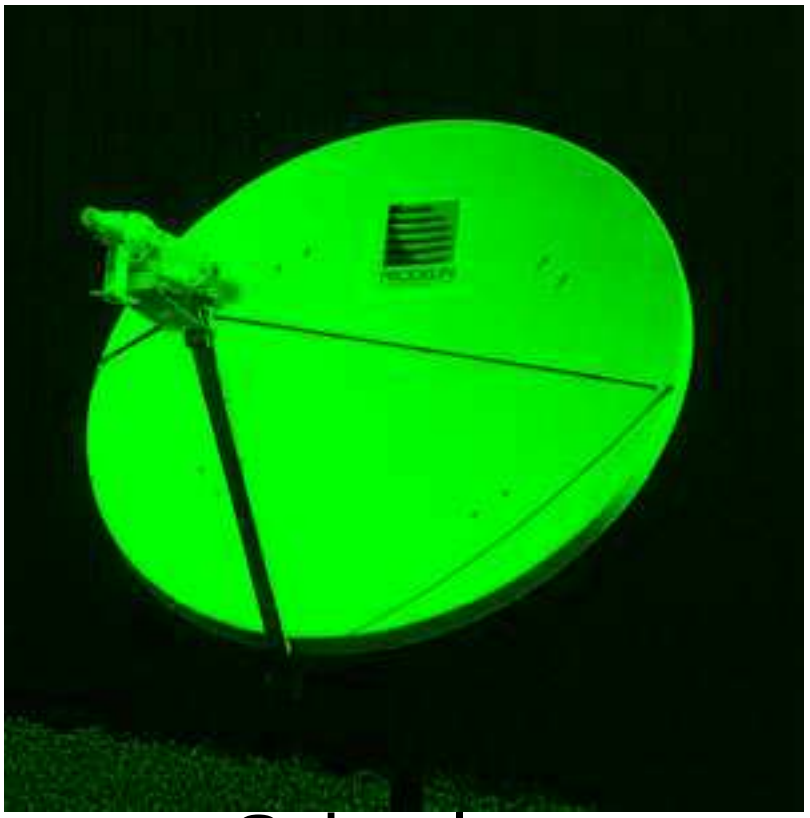
Bandwith Optimisation: Case Study of SHESTCO-NET



School on
Radio Use for
Information and Communication Technology
Abdus Salam ICTP, Trieste, Italy
2-21 February 2003.

M.G. Zebaze Kana <zebaze@shestco.org>

Content

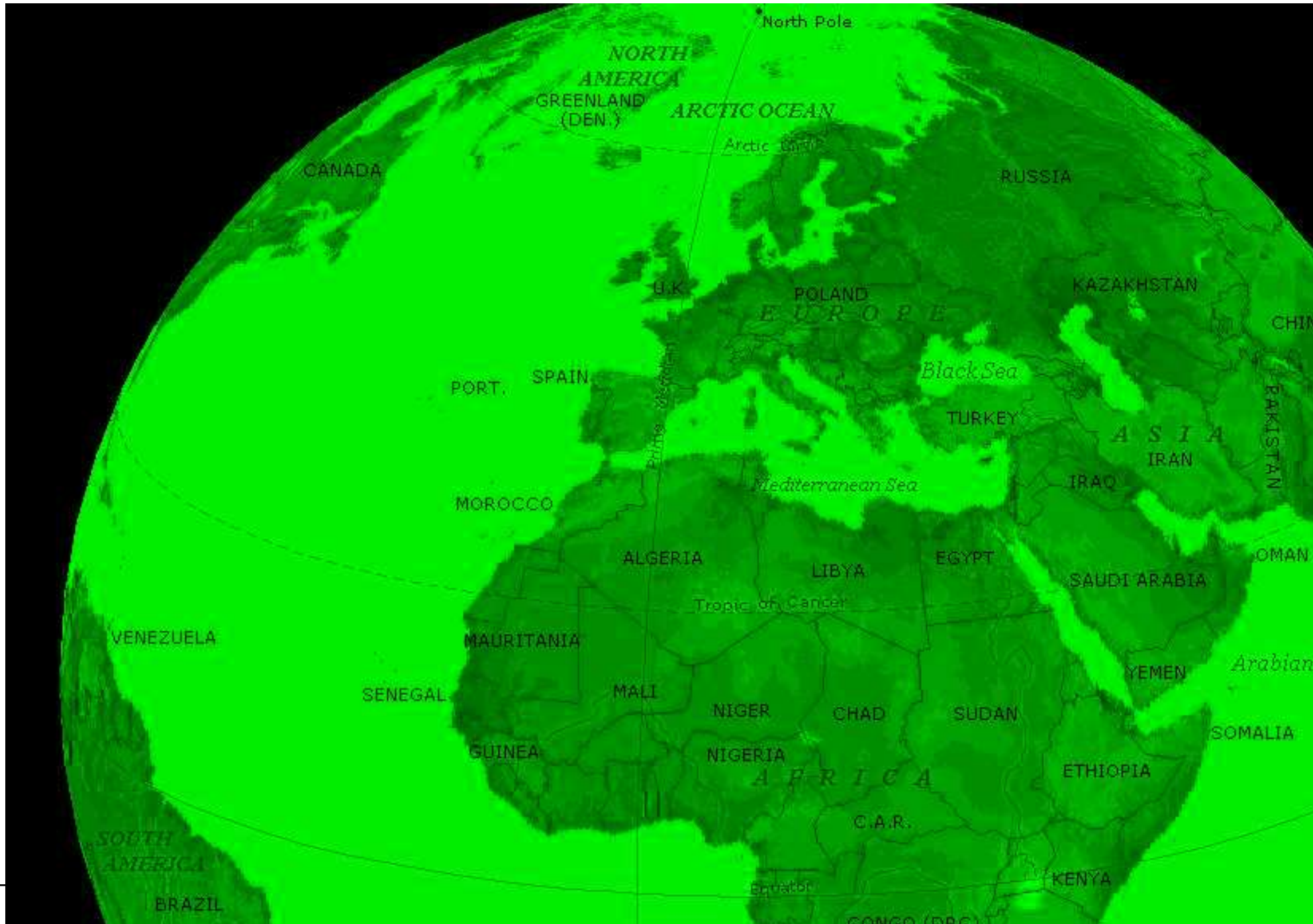
- Introduction
- Design Requirements
- Topology
- Hardware
- Operating Systems
- Why Linux?
- Other Operating Systems
- Getting our hands dirty
- Configuration
  - NAT
  - Traffic Shaping
  - Cache
  - Routing
  - DNS
  - MTA

Introduction

- SHESTCO at a Glance
  SHEda Science & Technology COmplex (SHESTCO), Abuja, Nigeria
  - Three National Laboratories
    - Physics Advanced Laboratory (PAL)
    - Biotechnology Advanced Laboratory (BAL)
    - Chemistry Advanced Laboratory (CAL)
  - One Nuclear Technology Center (NTC)
    - Gamma Iradiation Facilities (GIF)
  - Remote Administrative Unit

- Objectives
  - Campus Network interconnecting of the various Laboratories
  - Wireless Access for Mobile Computing
  - VLAN linking the Laboratories and the Remote Administrative Unit about 60 miles apart
  - Internet Access

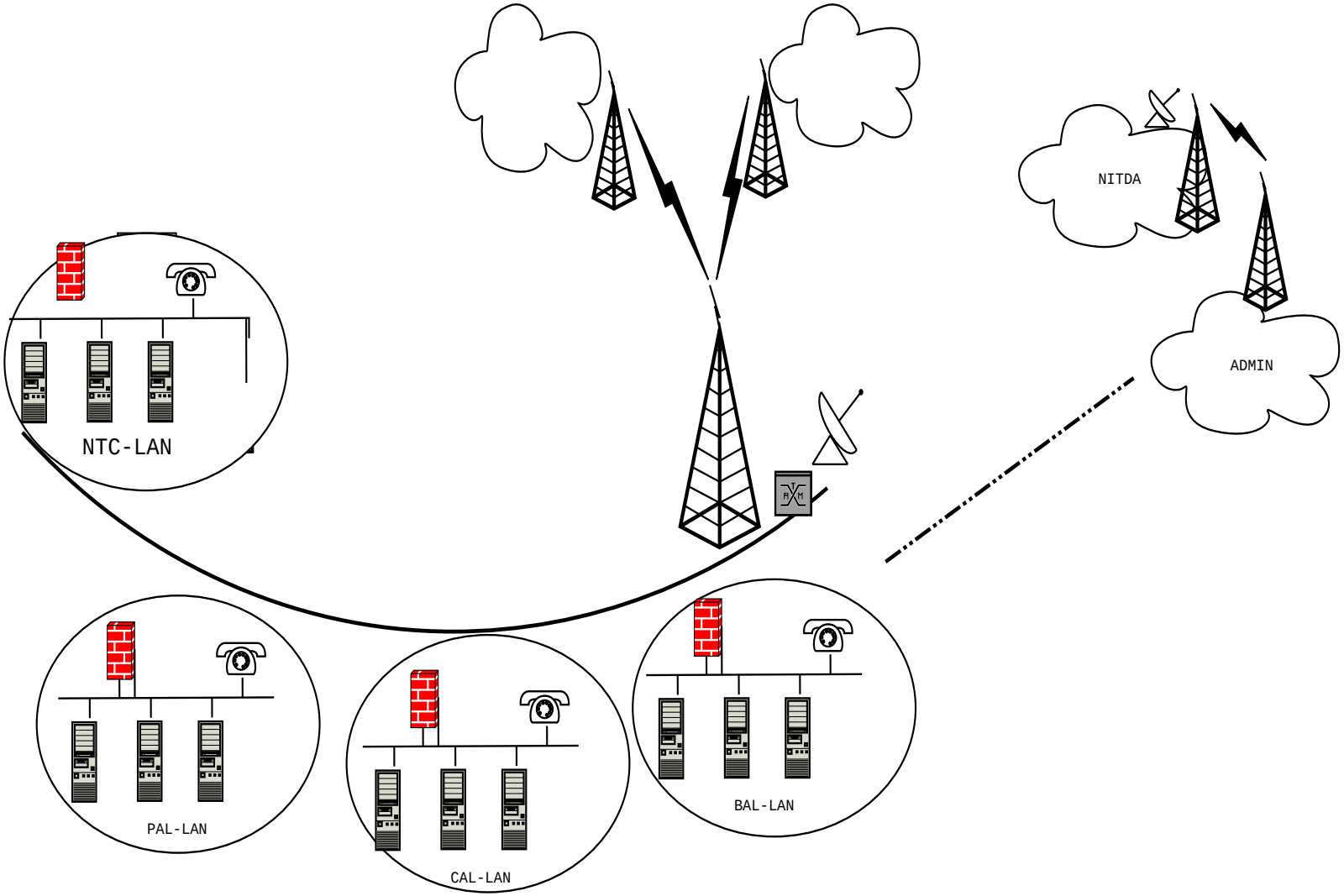
Our Location: Africa ==> Nigeria



Our Location: Nigeria ==> Abuja

Design Requirements

- Keep the cost low
- scalable Network
- VoIP
- Bandwidth: at least 100 MBPS Intranet Linking the Labs
- Remote Access for Instruments Technical Support
- Distance Leaning
- Video Conferencing


NITDA
ADMIN
NTC-LAN
PAL-LAN
CAL-LAN
BAL-LAN

Design Constraints

- Problems
  - Low Bandwith
  - Limited IP Addresses (IPv4)
  - Budget
  - Landscape (LOS)

- Some Solutions
  - Cahe engines
  - Traffic Shapping & Filtering
  - ACL (Access Control List)
  - Traffic Prioritising
  - NAT and Eventual Migration to IPng
  - not resources too demanding OS

Topology

- Hybrid
- Backbone
  - FDDI ring
- Distribution
- Access
  - Logical Bus
- Radio link to an ISP (back-up route)
- VSAT link

Network Evolution: Phase 1

1998: Quasi- Dailup Access

- Single User account with the ISP
- Laptop + MODEM
- Telephone Access 10 Km away from the Laboratories

Problems:

- E-mail only
- Service not reliable
- Delay
- Land Lines not available

Network Evolution: Phase 2

1999: Dialup Access over Microwave link

- Acquistion of DX 200 Microwave Link
  - Link from the Labs to the PTSN
- Several Single user accounts with the ISP
- Individual dialup accesses from the Laboratories
- Desktop PCs (not interconnected) + MODEMs

Problems

- Huge phone bills for individual accesses

Network Evolution: Phase 3

1999: LAN + Masquerading + Proxy

mail queue at the ISP end

- Two dialup links to 2 different ISPs

- LAN
  - One Linux Server
    - Sendmail
    - fetchmail
    - squid proxy
    - masquerading
    - cache only DNS

  Many interconnected Workstation

Problems

Low Bandwidth

- Phone bills relatibely hurge

Network Evolution: Phase 4

2000: Microwave Link + RAS

- BreezeAccess
  - Bandwidth: 64
- 1 Proxy Server
  - local DNS
  - Squid
  - filtering
- 1 Mail Sever
  - Sendmail
  - fetchmail
  - RAS
- 2 MODEMs for dial-in

Problems

- LOS

Hardware

- Servers
  - Compaq Proliant ML Servers with Hadware RAID and 2 hot plugable PCI slots
- Workstations
  - Any
- Router (Compaq Proliant + Linux
- RAS (PR 4000)
- 3COM 100/10 stackable switches
- Breezenet AccessPoint & sectorial antenna
- BreezeNet Bridge & directional antenna
- APC Smart UPS 3000 and Standby Generator

Operating System: Linux

Why Linux?

- Availability
  - Open source
  - Price (free!)
- Highly Configurable
- Networking ready
- Many tools available
- Ported to many architechtures
- Relatively easy to write new drivers
- Modular
- Free and comercial support available(Should in case you run into problems)
- Highly Secure (if properly configured)

Other Operating Systems

- MS Windows & Linux on WSs
- PC - Unix integration using SAMBA

Getting our Hands Dirty

Configurations

- NAT
- Transparent proxy
- Traffic shapping and prioritising
- DNS
- Any other needed service:
  - MTA
  - http
  - ftp

NAT

- IPtables
- Masquerade
- Redirection

IPtables Configuration

```
#!/bin/sh
#
# rc.firewall-2.4-strong
#
FWVER=0.80s-1
# Strong IPTABLES firewall with IP Masquerade (NAT) for SHESTCO-nig LAN
# support for 2.4.x kernels.
#
# Log:
# 0.80s -Modified by M.G. Zebaze Kana for SHESTCO Nigeria LAN, Dec. 2002.
# 0.62s - Initial version based upon the basic 2.4.x rc.firewall

echo -e "\nLoading STRONGER rc.firewall - version $FWVER..\n"
# Some definitions for our convenience
IPTABLES=/sbin/iptables
LSMOD=/sbin/lsmod
DEPMOD=/sbin/depmod
INSMOD=/sbin/insmod
GREP=/bin/grep
AWK=/bin/awk
SED=/bin/sed
IFCONFIG=/sbin/ifconfig
#Setting the EXTERNAL and INTERNAL interfaces for the network

# Each IP Masquerade network needs to have at least one
# external and one internal network. The external network
# is where the natting will occur and the internal network
# should preferably be addressed with a RFC1918 private address
# scheme.

EXTIF="eth0"
INTIF="eth1"
echo " External Interface: $EXTIF"
echo " Internal Interface: $INTIF"
```

IPtables Configuration

```
echo " ---"

# Determine the external IP automatically:
# ----------------------------------------

EXTIP="`$IFCONFIG $EXTIF | $AWK /$EXTIF/'{next}//{split($0,a,":");split(a[2],a," ");print a[1];exit}'`"

# For users who wish to use STATIC IP addresses:
#EXTIP="your.static.PPP.address"
echo " External IP: $EXTIP"
echo " ---"

# Assign the internal TCP/IP network and IP address
INTNET="192.168.1.0/24"
INTIP="192.168.1.1/24"
echo " Internal Network: $INTNET"
echo " Internal IP: $INTIP"
echo " ---"
# Setting a few other local variables
#
UNIVERSE="0.0.0.0/0"

#=============================================================================
# Need to verify that all modules have all required dependencies
#
echo " - Verifying that all kernel modules are ok"
$DEPMOD -a
echo -en " Loading kernel modules: "
# With the new IPTABLES code, the core MASQ functionality is now either
# modular or compiled into the kernel. This HOWTO shows ALL IPTABLES
# options as MODULES. If your kernel is compiled correctly, there is
# NO need to load the kernel modules manually.

# NOTE: The following items are listed ONLY for informational reasons.
# There is no reason to manual load these modules unless your
```

IPtables Configuration

```
# kernel is either mis-configured or you intentionally disabled
# the kernel module autoloader.

# Upon the commands of starting up IP Masq on the server, the
# following kernel modules will be automatically loaded:

# NOTE: Only load the IP MASQ modules you need. All current IP MASQ
# modules are shown below but are commented out from loading.
# ============================================================

#Load the main body of the IPTABLES module - "ip_tables"
# - Loaded automatically when the "iptables" command is invoked
#
# - Loaded manually to clean up kernel auto-loading timing issues
#
echo -en "ip_tables, "

#Verify the module isn't loaded. If it is, skip it

if [ -z "` $LSMOD | $GREP ip_tables | $AWK {'print $1'} `" ]; then
$INSMOD ip_tables
fi

#Load the IPTABLES filtering module - "iptable_filter"
#
# - Loaded automatically when filter policies are activated
#Load the stateful connection tracking framework - "ip_conntrack"
#
# The conntrack module in itself does nothing without other specific
#conntrack modules being loaded afterwards such as the "ip_conntrack_ftp"

# module
#
# - This module is loaded automatically when MASQ functionality is
# enabled
#
```

IPtables Configuration

```
# - Loaded manually to clean up kernel auto-loading timing issues
#
echo -en "ip_conntrack, "

#Verify the module isn't loaded. If it is, skip it

if [ -z "‘ $LSMOD | $GREP ip_conntrack | $AWK {'print $1'} ‘" ]; then
$INSMOD ip_conntrack
fi

#Load the FTP tracking mechanism for full FTP tracking
#
# Enabled by default -- insert a "#" on the next line to deactivate
#
echo -e "ip_conntrack_ftp, "

#Verify the module isn't loaded. If it is, skip it

if [ -z "‘ $LSMOD | $GREP ip_conntrack_ftp | $AWK {'print $1'} ‘" ]; then
$INSMOD ip_conntrack_ftp
fi

#Load the IRC tracking mechanism for full IRC tracking
#
# Enabled by default -- insert a "#" on the next line to deactivate
#
echo -en " ip_conntrack_irc, "

#Verify the module isn't loaded. If it is, skip it

if [ -z "‘ $LSMOD | $GREP ip_conntrack_irc | $AWK {'print $1'} ‘" ]; then
$INSMOD ip_conntrack_irc
fi

#Load the general IPTABLES NAT code - "iptable_nat"
# - Loaded automatically when MASQ functionality is turned on
```

IPtables Configuration

```
#
# - Loaded manually to clean up kernel auto-loading timing issues
#
echo -en "iptable_nat, "

#Verify the module isn't loaded. If it is, skip it

if [ -z "` $LSMOD | $GREP iptable_nat | $AWK {'print $1'} `" ]; then
$INSMOD iptable_nat
fi
#Loads the FTP NAT functionality into the core IPTABLES code
# Required to support non-PASV FTP.
#
# Enabled by default -- insert a "#" on the next line to deactivate
#
echo -e "ip_nat_ftp"
#
#Verify the module isn't loaded. If it is, skip it
#
if [ -z "` $LSMOD | $GREP ip_nat_ftp | $AWK {'print $1'} `" ]; then
$INSMOD ip_nat_ftp
fi
echo " ---"

# Just to be complete, here is a list of the remaining kernel modules
# and their function. Please note that several modules should be only
# loaded by the correct master kernel module for proper operation.
# ----------------------------------------------------------------------
#
# ipt_mark - this target marks a given packet for future action.
# This automatically loads the ipt_MARK module
#
# ipt_tcpmss - this target allows to manipulate the TCP MSS
# option for braindead remote firewalls.
# This automatically loads the ipt_TCPMSS module
#
```

IPtables Configuration

```
#
# ipt_limit - this target allows for packets to be limited to
# to many hits per sec/min/hr
#
# ipt_multiport - this match allows for targets within a range
# of port numbers vs. listing each port individually
#
# ipt_state - this match allows to catch packets with various
# IP and TCP flags set/unset
#
# ipt_unclean - this match allows to catch packets that have invalid
# IP/TCP flags set
#
# iptable_filter - this module allows for packets to be DROPped,
# REJECTed, or LOGged. This module automatically
# loads the following modules:
#
# ipt_LOG - this target allows for packets to be
# logged
#
# ipt_REJECT - this target DROPs the packet and returns
# a configurable ICMP packet back to the
# sender.
#
# iptable_mangle - this target allows for packets to be manipulated
# for things like the TCPMSS option, etc.
#CRITICAL: Enable IP forwarding since it is disabled by default since
#
# Redhat Users: you may try changing the options in
# /etc/sysconfig/network from:
#
# FORWARD_IPV4=false
# to
# FORWARD_IPV4=true
#
```

IPtables Configuration

```
echo " Enabling forwarding.."
echo "1" > /proc/sys/net/ipv4/ip_forward
# Dynamic IP users:
#
# If you get your IP address dynamically from SLIP, PPP, or DHCP,
# enable the following option. This enables dynamic-address hacking
# which makes the life with Diald and similar programs much easier.
#
## echo " Enabling DynamicAddr.."
## echo "1" > /proc/sys/net/ipv4/ip_dynaddr
## echo " ---"

##############################################################################
####
#
# Enable Stronger IP forwarding and Masquerading
#
# NOTE: In IPTABLES speak, IP Masquerading is a form of SourceNAT or SNAT


#
# NOTE #2: The following is for  anglican-nig with internal LAN address in
# 192.168.1.x network with a  "24" bit subnet (netmask 255.255.255.0)
# mask connecting to the Internet on external interface "eth0".
# This  will MASQ internal traffic out to the Internet

#Clearing any previous configuration


# Unless specified, the defaults for INPUT, OUTPUT, and FORWARD to DROP.
#
# You CANNOT change this to REJECT as it isn't a vaild setting for a
# policy. If you want REJECT, you must explictly REJECT at the end
# of a giving INPUT, OUTPUT, or FORWARD chain

echo " Clearing any existing rules and setting default policy to DROP.."
$IPTABLES -P INPUT DROP
```

IPtables Configuration

```
$IPTABLES -F INPUT
$IPTABLES -P OUTPUT DROP
$IPTABLES -F OUTPUT
$IPTABLES -P FORWARD DROP
$IPTABLES -F FORWARD
$IPTABLES -F -t nat
#Not needed and it will only load the unneeded kernel module
#$IPTABLES -F -t mangle

# Flush the user chain.. if it exists
if [ -n "`$IPTABLES -L | $GREP drop-and-log-it`" ]; then
$IPTABLES -F drop-and-log-it
fi

# Delete all User-specified chains
$IPTABLES -X

# Reset all IPTABLES counters
$IPTABLES -Z

#Configuring specific CHAINS for later use in the ruleset
#
# NOTE: Without the --log-level set to "info", every single
# firewall hit will goto ALL vtys. This is a very big
# pain.

echo " Creating a DROP chain.."
$IPTABLES -N drop-and-log-it
$IPTABLES -A drop-and-log-it -j LOG --log-level info
$IPTABLES -A drop-and-log-it -j DROP
echo -e "\n - Loading INPUT rulesets"

##############################################################################
# INPUT: Incoming traffic from various interfaces. All rulesets are
# already flushed and set to a default policy of DROP.
```

IPtables Configuration

```
# loopback interfaces are valid, very important.
$IPTABLES -A INPUT -i lo -s $UNIVERSE -d $UNIVERSE -j ACCEPT

# local interface, local machines, going anywhere is valid
$IPTABLES -A INPUT -i $INTIF -s $INTNET -d $UNIVERSE -j ACCEPT

# remote interface, claiming to be local machines, IP spoofing, bad boys!
$IPTABLES -A INPUT -i $EXTIF -s $INTNET -d $UNIVERSE -j drop-and-log-it

# external interface, from any source, for ICMP traffic is valid
#
# If you would like your machine to "ping" from the Internet,
# enable this next line
$IPTABLES -A INPUT -i $EXTIF -p ICMP -s $UNIVERSE -d $EXTIP -j ACCEPT

# remote interface, any source, going to permanent PPP address is valid
#
#$IPTABLES -A INPUT -i $EXTIF -s $UNIVERSE -d $EXTIP -j ACCEPT

# Allow any related traffic coming back to the MASQ server in
$IPTABLES -A INPUT -i $EXTIF -s $UNIVERSE -d $EXTIP -m state --state ESTABLISHED,RELATED -j ACCEPT

# ----- Begin OPTIONAL Section -----
#
# DHCPd - We run an INTERNAL DHCPd server for some workstations
$IPTABLES -A INPUT -i $INTIF -p tcp --sport 68 --dport 67 -j ACCEPT
$IPTABLES -A INPUT -i $INTIF -p udp --sport 68 --dport 67 -j ACCEPT

# HTTPd - Enable the following lines if you run an EXTERNAL WWW server
#
#echo -e " - Allowing EXTERNAL access to the WWW server"
#$IPTABLES -A INPUT -i $EXTIF -m state --state NEW,ESTABLISHED,RELATED # -p tcp -s $UNIVERSE -d $EXTIP --dport 80 -j
ACCEPT
#
# ----- End OPTIONAL Section -----
```

IPtables Configuration

```
# Catch all rule, all other incoming is denied and logged.
#
$IPTABLES -A INPUT -s $UNIVERSE -d $UNIVERSE -j drop-and-log-it

echo -e " - Loading OUTPUT rulesets"

##############################################################################
# OUTPUT: Outgoing traffic from various interfaces. All rulesets are
# already flushed and set to a default policy of DROP.

# loopback interface is valid.
$IPTABLES -A OUTPUT -o lo -s $UNIVERSE -d $UNIVERSE -j ACCEPT

# local interfaces, any source going to local net is valid
$IPTABLES -A OUTPUT -o $INTIF -s $EXTIP -d $INTNET -j ACCEPT

# local interface, any source going to local net is valid
$IPTABLES -A OUTPUT -o $INTIF -s $INTIP -d $INTNET -j ACCEPT

# outgoing to local net on remote interface, stuffed routing, deny
$IPTABLES -A OUTPUT -o $EXTIF -s $UNIVERSE -d $INTNET -j drop-and-log-it

# anything else outgoing on remote interface is valid
$IPTABLES -A OUTPUT -o $EXTIF -s $EXTIP -d $UNIVERSE -j ACCEPT

# ----- Begin OPTIONAL Section -----

# DHCPd - We run an INTERNAL DHCPd server
$IPTABLES -A OUTPUT -o $INTIF -p tcp -s $INTIP --sport 67 -d 255.255.255.255 --dport 68 -j ACCEPT
$IPTABLES -A OUTPUT -o $INTIF -p udp -s $INTIP --sport 67 -d 255.255.255.255 --dport 68 -j ACCEPT

# ----- End OPTIONAL Section -----
```

IPtables Configuration

```
# Catch all rule, all other outgoing is denied and logged.
$IPTABLES -A OUTPUT -s $UNIVERSE -d $UNIVERSE -j drop-and-log-it

echo -e " - Loading FORWARD rulesets"

###############################################################################
# FORWARD: Enable Forwarding and thus IPMASQ
#
echo " - FWD: Allow all connections OUT and only existing/related IN"
$IPTABLES -A FORWARD -i $EXTIF -o $INTIF -m state --state ESTABLISHED,REL
ATED -j ACCEPT
$IPTABLES -A FORWARD -i $INTIF -o $EXTIF -j ACCEPT

## We shall allow in the future connection some services (ssh, ftp, www)IN
## rule to be inserted here

# Catch all rule, all other forwarding is denied and logged.
$IPTABLES -A FORWARD -j drop-and-log-it

echo "Redirecting www requests to SQUID proxy ..."
$IPTABLES -t nat -A PREROUTING -s $INTNET -i $INTIF -p tcp --dport www -j REDIRECT --to 8080

echo " - NAT: Enabling SNAT (MASQUERADE) functionality on $EXTIF"
#
#More liberal form
#$IPTABLES -t nat -A POSTROUTING -o $EXTIF -j MASQUERADE
#
#Stricter form
$IPTABLES -t nat -A POSTROUTING -o $EXTIF -j SNAT --to $EXTIP

###############################################################################

echo -e "\nStrong and secured rc.firewall-2.4 $FWVER done.\n"
echo -e "(C) M.G. Zebaze Kana <zebaze@shestco.org>, December 2002.\n"
```

Traffic Shaping and Prioritising

Cache

- squid

Routing

- iproute2

MTA

- Sendmail

Current Projet

- VSAT Link
  - 2.5 m dish
  - Paradise P300 Satellite MODEM
  - CODAN Transceiver
- Intial Bandwidth
  - Uplink: 128 Kbps
  - Downlink: 256 Kbps
- FDDI ring Distribution
- Wireless Access
- ATM
- VLAN
- P4000 RAS (over E1 lines)

Need more Information?

- HowTos
  for more information.

Thank you