FINFISHER: FinFly ISP 2.0

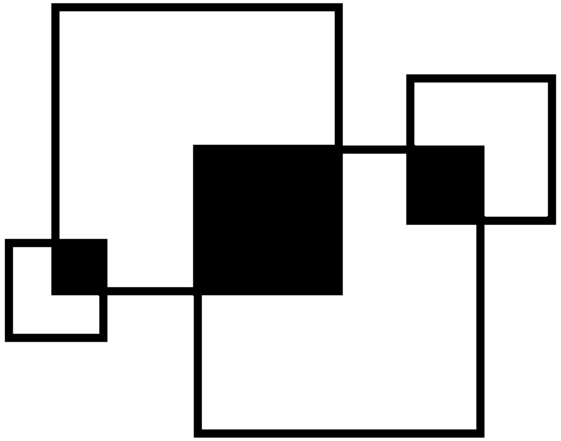
Infrastructure Product Training

WWW.GAMMAGROUP.COM

FINFISHER

IT INTRUSION

Table of content



1. Introduction

2. The infrastructure

   - ADMF Client and Infection GUI
   - Administration: ADMF
   - iProxy: NDP01/02
   - Radius Probe: RP01/02
   - Communication

3. Use Case Infection

4. System handling

5. Technical details

6. Incident handling

1. Introduction

Who we are

WWW.GAMMAGROUP.COM

FINFISHER

IT INTRUSION

Introduction

Delegates:

Nicolas Mayencourt

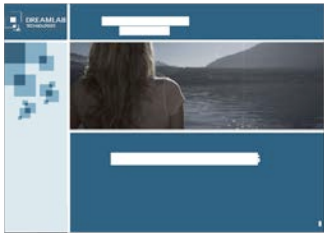
Head of Dreamlab Technologies AG

Member of the Board of Directors, ISECOM

Member OWASP

Richard Sademach

Head of Operations Dreamlab Technologies AG



2. The infrastructure

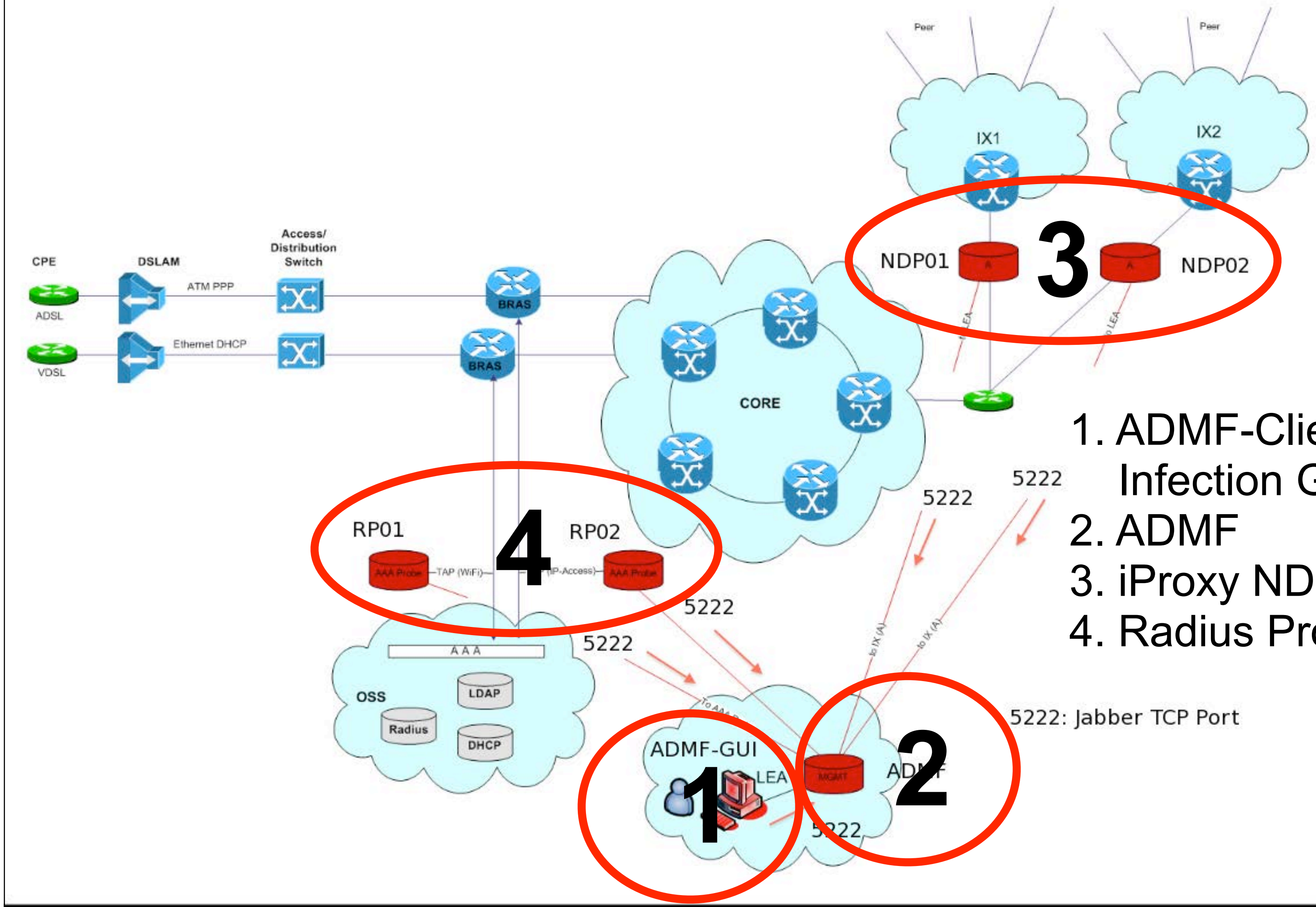
Overview & components

FINFISHER

IT INTRUSION

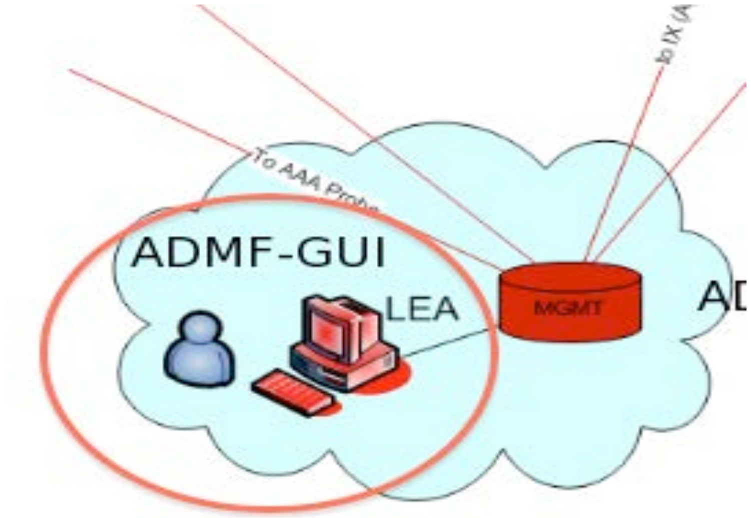
WWW.GAMMAGROUP.COM

Infrastructure overview: components



1. ADMF-Client & Infection GUI
2. ADMF
3. iProxy NDP01/02
4. Radius Probe RP01/02

1. ADMF Client and Infection GUI

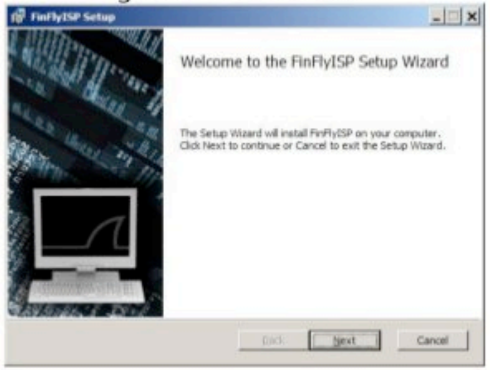


- ADMF Client
- Graphical User Interface for managing Infections
- Configuring Infections
- Selection of Infection method
- Realtime status information
- Management of all components

1. ADMF Client → Infection GUI

Separate Training

Figure 1: Welcome Screen

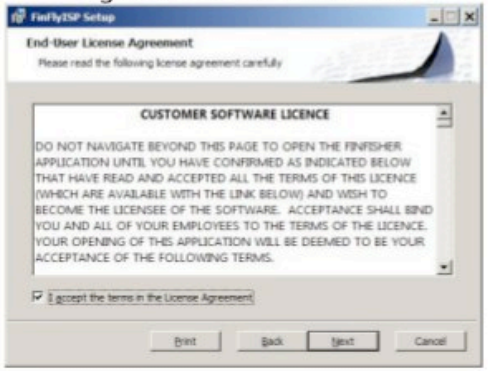


FinFlyISP Setup

Welcome to the FinFlyISP Setup Wizard

The Setup Wizard will install FinFlyISP on your computer.
Click Next to continue or Cancel to exit the Setup Wizard.

Figure 2: License Information



FinFlyISP Setup

End-User License Agreement

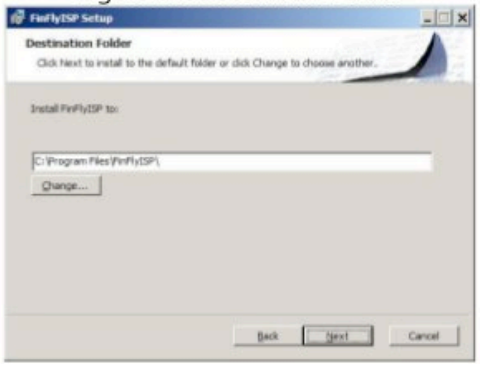
Please read the following license agreement carefully

CUSTOMER SOFTWARE LICENCE

DO NOT NAVIGATE BEYOND THIS PAGE TO OPEN THE FINFISHER APPLICATION UNTIL YOU HAVE CONFIRMED AS INDICATED BELOW THAT HAVE READ AND ACCEPTED ALL THE TERMS OF THIS LICENCE (WHICH ARE AVAILABLE WITH THE LINK BELOW) AND WISH TO BECOME THE LICENSEE OF THE SOFTWARE. ACCEPTANCE SHALL BIND YOU AND ALL OF YOUR EMPLOYEES TO THE TERMS OF THE LICENCE. YOUR OPENING OF THIS APPLICATION WILL BE DEEMED TO BE YOUR ACCEPTANCE OF THE FOLLOWING TERMS.

☑ I accept the terms in the License Agreement

Figure 3: Select Install Folder



FinFlyISP Setup

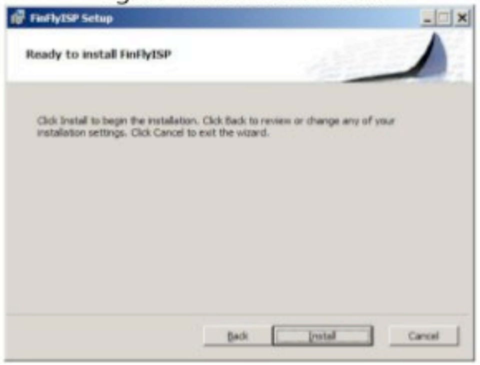
Destination Folder

Click Next to install to the default folder or click Change to choose another.

Install FinFlyISP to:

C:\Program Files\FinFlyISP\

Figure 4: Start Installation



FinFlyISP Setup

Ready to install FinFlyISP

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

1. ADMF Client and Infection GUI

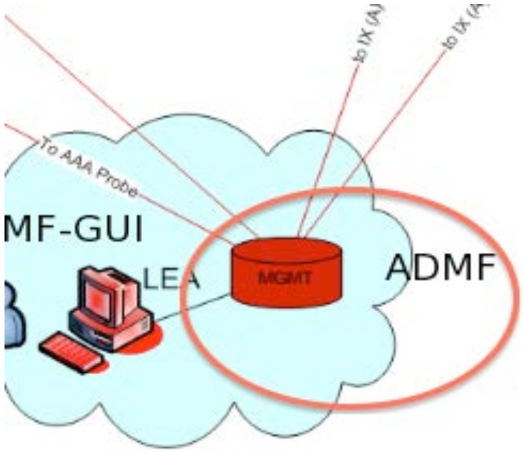
Hardware:

- HP Compaq 8000 Elite Business PC
- 1 x Copper 10/100/1000

Software:

- FinFly ISP GUI
- XMPP Client
- Windows 7 Ultimate

2. ADMF - Central Administration Function



- Core component of the FinFly ISP infrastructure
- Realtime communication with all components
→ NDP, RP, FinFly Gui
- Configuration and initiation of infections
on the ADMF
- Provisioning of the ADMF Client , iProxy and RP
- Realtime exchange of information and states
→ Targets coming online, being infected, etc
- RFC XMPP protocol used for secure and
encrypted communication (TLS based)

2. ADMF - Central Administration Function

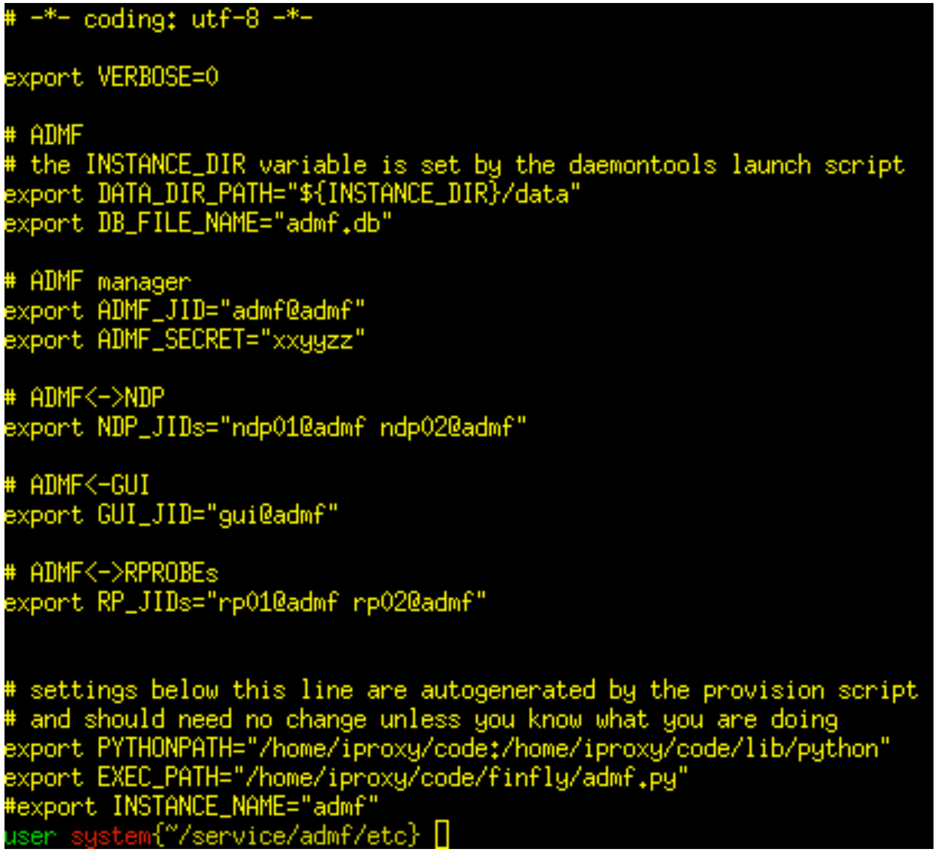
Hardware:

- HP DL380 G6
- 2x Intel(R) Xeon(R) CPU X5550 @ 2.67GHz
- Memory: 12 GB
- 3 x 146 GB SAS 2,5" (Raid 5)
- 4 x Copper 10/100/1000
- 1 x ILO (Integrated Lights Out)
- OS:Linux GNU (Debian 5.0), hardened by Dreamlab best practices

Software:

- ADMF → Adminstration function
- Ejabberd (XMPP server)

ADMF Configuration



```
# -*- coding: utf-8 -*-

export VERBOSE=0

# ADMF
# the INSTANCE_DIR variable is set by the daemontools launch script
export DATA_DIR_PATH="${INSTANCE_DIR}/data"
export DB_FILE_NAME="admf.db"

# ADMF manager
export ADMF_JID="admf@admf"
export ADMF_SECRET="xxyyzz"

# ADMF<->NDP
export NDP_JIDs="ndp01@admf ndp02@admf"

# ADMF<-GUI
export GUI_JID="gui@admf"

# ADMF<->RPROBEs
export RP_JIDs="rp01@admf rp02@admf"


# settings below this line are autogenerated by the provision script
# and should need no change unless you know what you are doing
export PYTHONPATH="/home/iproxy/code:/home/iproxy/code/lib/python"
export EXEC_PATH="/home/iproxy/code/finfly/admf.py"
#export INSTANCE_NAME="admf"
user system{~/service/admf/etc}
```

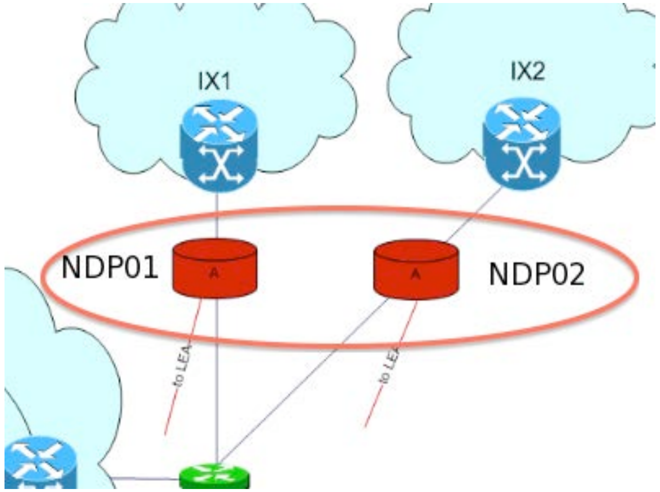
ADMF Configuration

Name: instance.conf

Path:

/home/iproxy/service/admf/etc/

3. NDP01 / NDP02 → iProxy



- Network data processing component

- Infections remotely activated/deactivated via the ADMF/ADMF GUI

- Provisioning of the actual target IP-Address from the RP via the ADMF

- Each NDP bridge is equipped with a carrier grade 10GB/s fiber bypass module

- In case of hardware or logical failures this module switches automatically to bypass-mode. Thus traffic will never be interrupted.

- Attention this is a highly dynamic bridge / fw environment: **DO NOT change any configuration manually**

⚡ The NDP has been specifically configured for this network. Any configuration change of the network i.e. protocolstacks, media, failover features etc must be tightly coordinated with Dreamlab. Not doing so most probably will lead to an unusable system.

3. NDP01 / NDP02 → iProxy

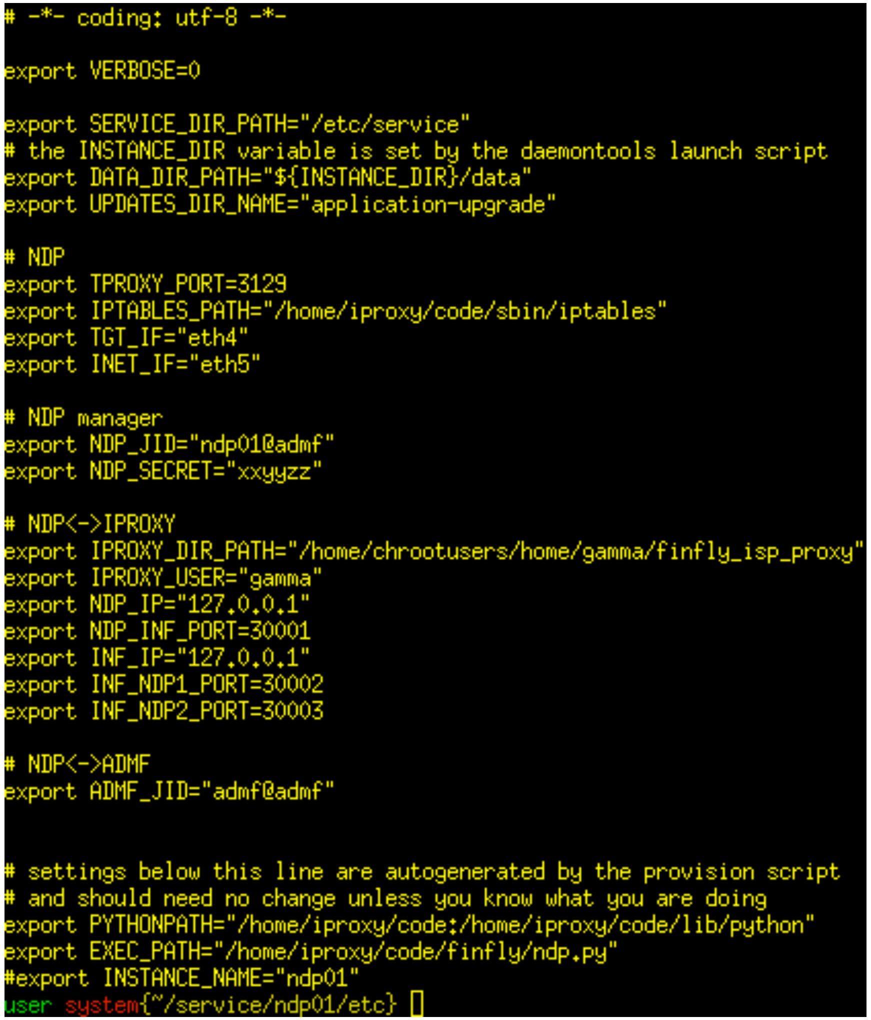
Hardware:

- HP DL380 G7
  2x Intel(R) Xeon(R) CPU X5650 @ 2.67GHz
- Memory: 12 GB
- 3 x 146 GB SAS 2,5" (Raid 5)
- 4 x Copper 10/100/1000
- 1 x Fiber Multimode Bypass NIC
- 1 x ILO (Integrated Lights Out)
- OS:Linux GNU (Debian 5.0), hardened
  by Dreamlab best practices

Software:

- NDP → Network Data Processor
- IProxy → infection Proxy
- ADMF Client

NDP Configuration



```
# -*- coding: utf-8 -*-

export VERBOSE=0

export SERVICE_DIR_PATH="/etc/service"
# the INSTANCE_DIR variable is set by the daemontools launch script
export DATA_DIR_PATH="${INSTANCE_DIR}/data"
export UPDATES_DIR_NAME="application-upgrade"

# NDP
export TPROXY_PORT=3129
export IPTABLES_PATH="/home/iproxy/code/sbin/iptables"
export TGT_IF="eth4"
export INET_IF="eth5"

# NDP manager
export NDP_JID="ndp01@admf"
export NDP_SECRET="xxyyzz"

# NDP<->IPROXY
export IPROXY_DIR_PATH="/home/chrootusers/home/gamma/finfly_isp_proxy"
export IPROXY_USER="gamma"
export NDP_IP="127.0.0.1"
export NDP_INF_PORT=30001
export INF_IP="127.0.0.1"
export INF_NDP1_PORT=30002
export INF_NDP2_PORT=30003

# NDP<->ADMF
export ADMF_JID="admf@admf"


# settings below this line are autogenerated by the provision script
# and should need no change unless you know what you are doing
export PYTHONPATH="/home/iproxy/code:/home/iproxy/code/lib/python"
export EXEC_PATH="/home/iproxy/code/finfly/ndp.py"
#export INSTANCE_NAME="ndp01"
user system{~/service/ndp01/etc} 
```

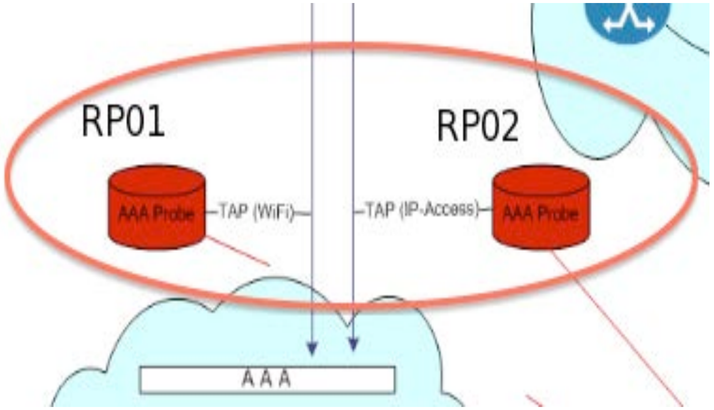
<u>NDP Configuration</u>

Name: instance.conf

Path:

/home/iproxy/service/ndp0[12]/etc/

4. RP01 / RP02 → Radius probe



- Realtime monitoring of the AAA processes: Targets coming online, receiving IP addresses, changing IP addresses, going offline
- Recording of the RADIUS authentications and accounting dialogues
- Being always up-to-date of the target IP address
- RP sends information to the ADMF
- The ADMF provisions the NDP's
- For statically configured IP addresses this is not needed

The target identification has been specifically configured for the local setup. Any configuration changes of the AAA / Radius setup must be tightly coordinated with Dreamlab. Failure to do so will most probably lead to an unusable system.

4. RP01 / RP02 → Radius probe

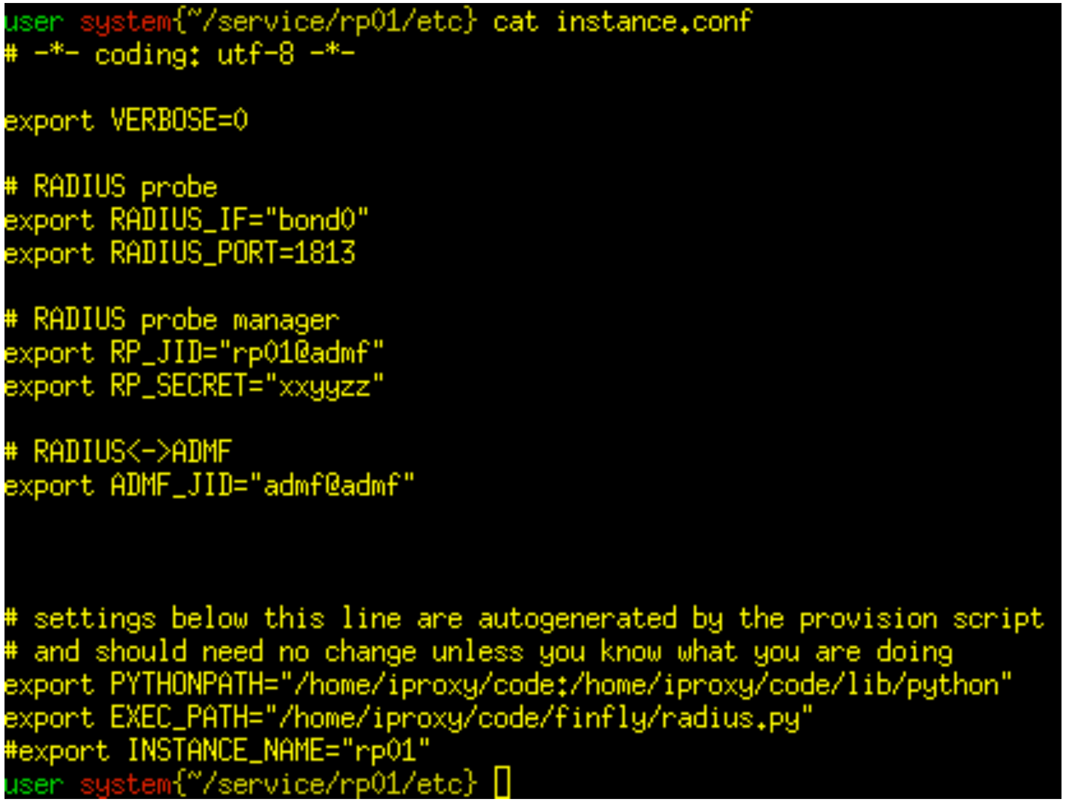
Hardware:

- HP DL380 G6
- 2x Intel(R) Xeon(R) CPU X5550 @ 2.67GHz
- Memory: 12 GB
- 3 x 146 GB SAS 2,5'' (Raid 5)
- 4 x Copper 10/100/1000
- 1 x Intel quad port 1G copper
- 1 x ILO (Integrated Lights Out)
- OS:Linux GNU (Debian 5.0), hardened by Dreamlab best practices

Software:

- RP → Radius Probe
- ADMF Client

RP Configuration



```
user system{~/service/rp01/etc} cat instance.conf
# -*- coding: utf-8 -*-

export VERBOSE=0

# RADIUS probe
export RADIUS_IF="bond0"
export RADIUS_PORT=1813

# RADIUS probe manager
export RP_JID="rp01@admf"
export RP_SECRET="xxyyzz"

# RADIUS<->ADMF
export ADMF_JID="admf@admf"



# settings below this line are autogenerated by the provision script
# and should need no change unless you know what you are doing
export PYTHONPATH="/home/iproxy/code:/home/iproxy/code/lib/python"
export EXEC_PATH="/home/iproxy/code/finfly/radius.py"
#export INSTANCE_NAME="rp01"
user system{~/service/rp01/etc} 
```

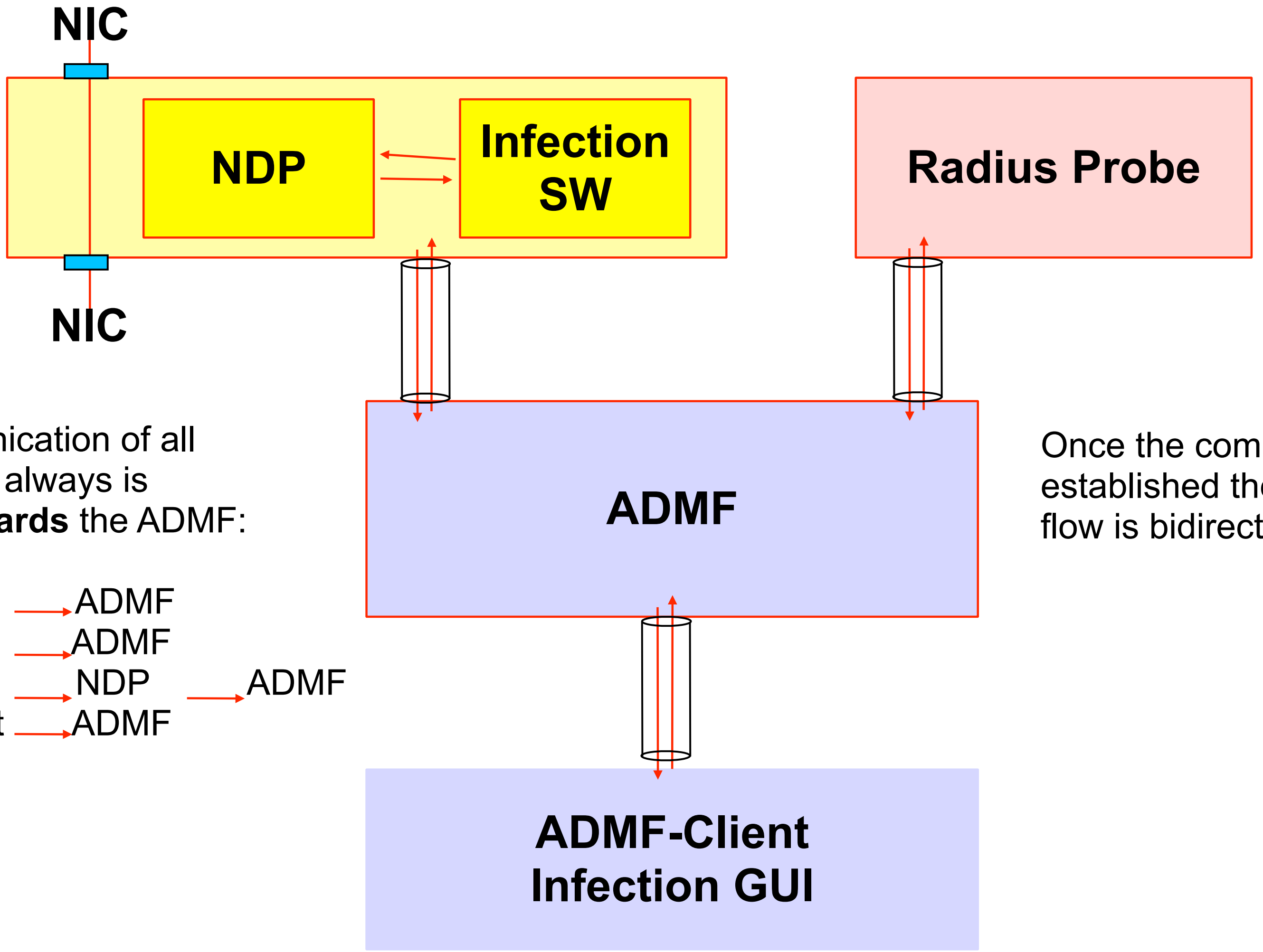
RP Configuration

Name: instance.conf

Path:

/home/iproxy/service/rp0[12]/etc/

Communication visualized



The communication of all components always is initiated **towards** the ADMF:

RP → ADMF
NDP → ADMF
Inf.SW → NDP → ADMF
ADMF-Client → ADMF

Once the communication is established the information flow is bidirectional (red arrows).

Communication: Traffic matrix

from / to	ADMF	ADMF-GUI	NDP	RP
ADMF	none	none	TCP 62200	TCP 62200
ADMF-GUI	TCP 62200 / TCP 17990 / TCP 443 / TCP 5222 TCP 23	none	TCP 62200 / TCP 17990 / TCP 443 TCP 23	TCP 62200 / TCP 17990 / TCP 443 TCP 23
NDP	TCP 62200 / TCP 5222	none	none	TCP 62200
RP	TCP 62200 / TCP 5222	none	TCP 62200	none

3. Use Case

Infection

FINFISHER
IT INTRUSION

Use Case → Infection

Step	Direction	Action content	Details
1	GUI -> ADMF	Infect a target	Send infection information Target information / infection mode
2	ADMF -> Radius probe	Start monitoring and set a trap on this target	Actual IP address of target is known
3	Radius -> ADMF -> NDP / iProxy	Handover actual IP address	IP address
4	iProxy -> NDP	Iproxy requests NDP to analyse the datastream on IP address and „interesting“ traffic	Target IP address
5	NDP -> iProxy	Handover traffic matching the request	Stream is redirected to iProxy
6	iProxy	changes the traffic and modifies the data by adding the infection parts	

Use Case → Infection

Step	Direction	Action content	Details
6	iProxy	changes the traffic and modifies the data by adding the infection parts	
7	iProxy -> NDP	iProxy sends the modifed traffic back to NDP	
8	NDP Reinject	NDP recalculates checksums, resequences TCP/IP packets and reinjects the traffic into the stream	
9	Target infection done	Data successfully sent to target	

Use Case → Infection

10. Infection succeeded → Start operating the target

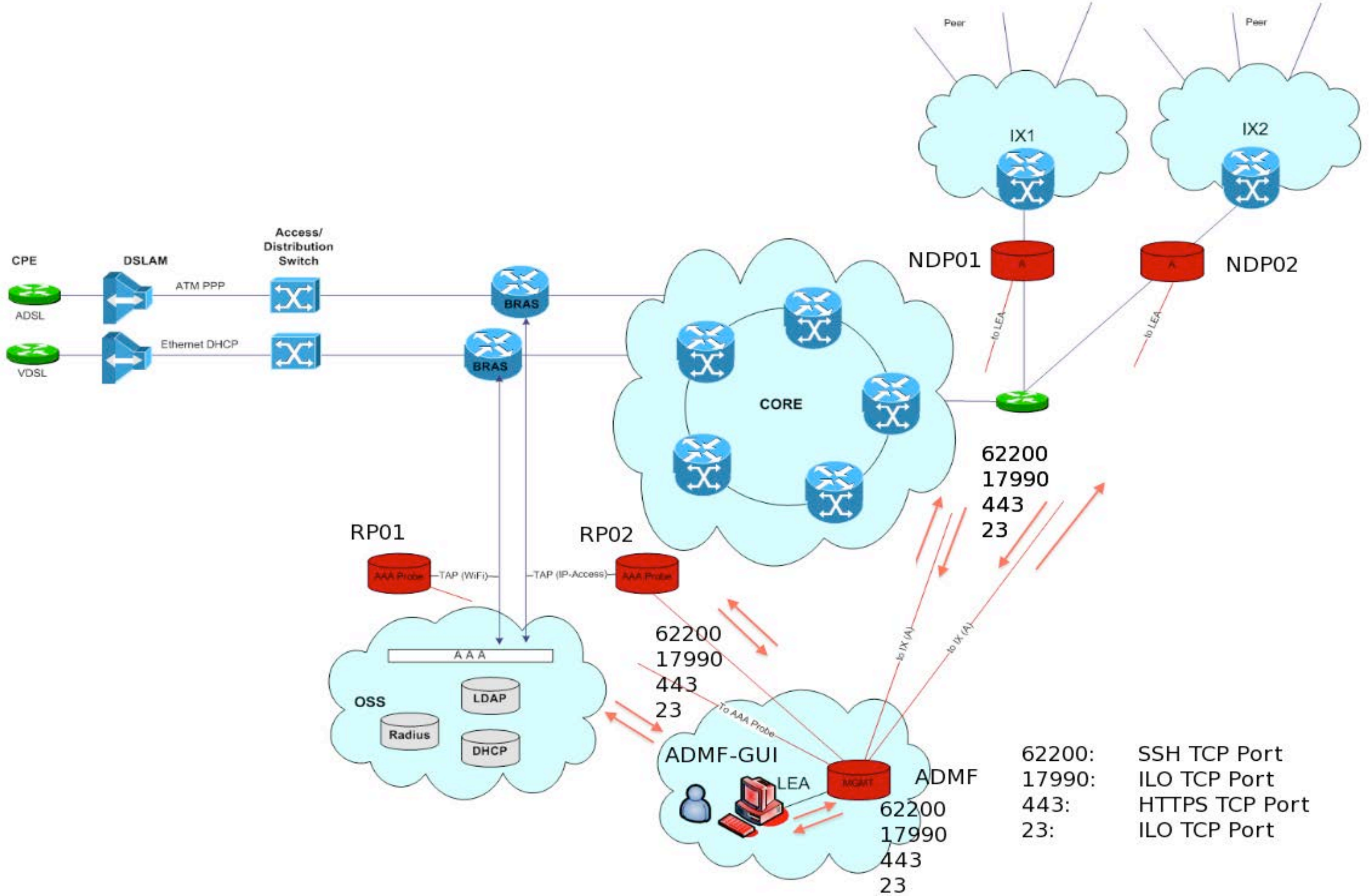
Seperate training

3. System handling

Management network
ILO access

FINFISHER
IT INTRUSION

Management network



62200:	SSH TCP Port
17990:	ILO TCP Port
443:	HTTPS TCP Port
23:	ILO TCP Port

Management network access

The iProxy components can either be accessed via SSH or ILO.
These interfaces are solely made available on the management network.

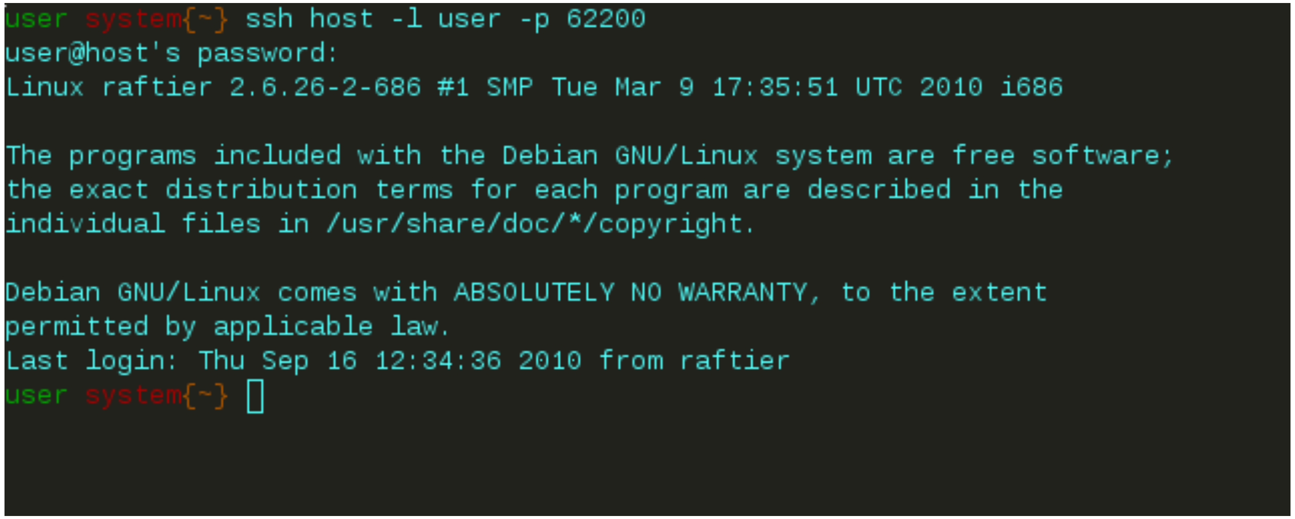
- SSH :

Secure shell is being used to directly access the iProxy components for all configuration changes, operation and debugging on system-level

- ILO :

Integrated lights out management is the dedicated access being used to manage system HW-components. i.e.: stop/start of the system hardware, hardware-monitoring, remote system console, etc

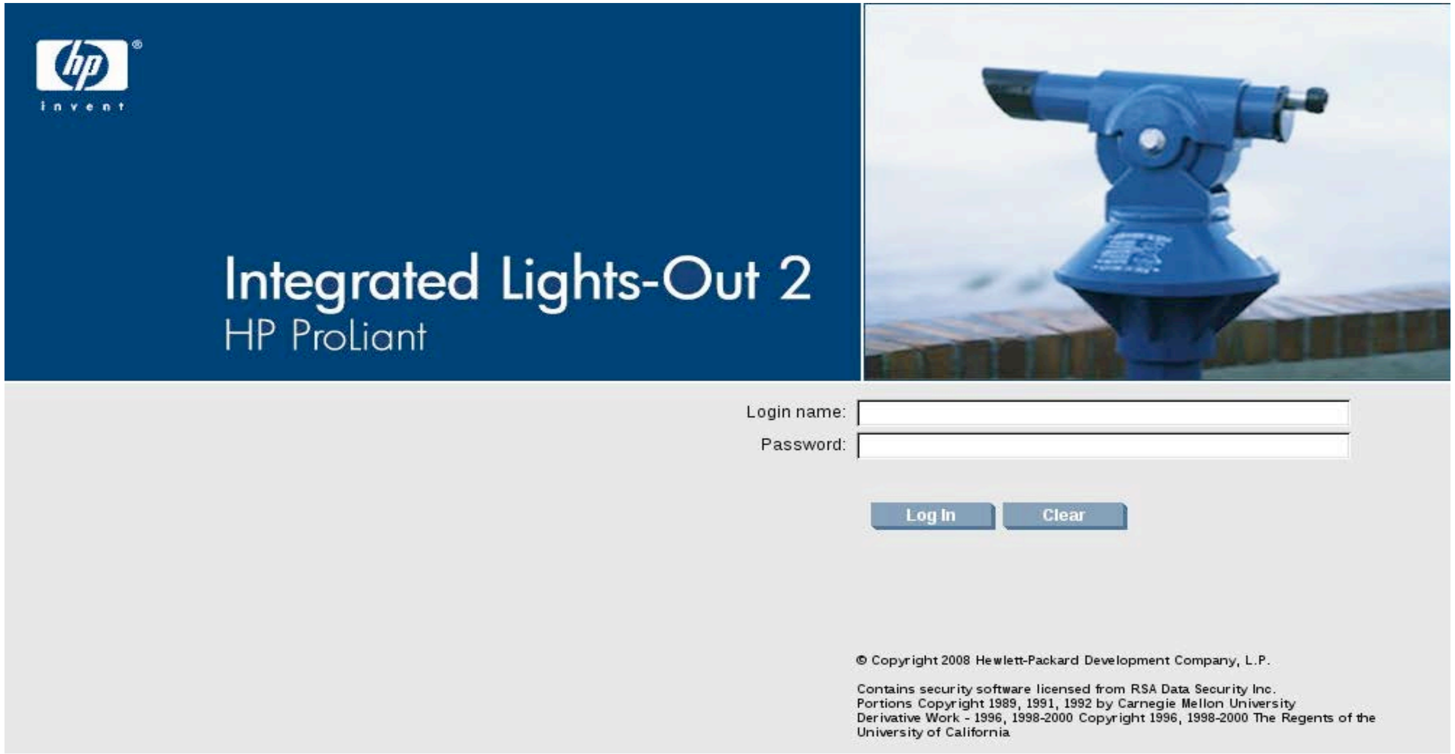
SSH access



```
user system{~} ssh host -l user -p 62200
user@host's password:
Linux raftier 2.6.26-2-686 #1 SMP Tue Mar 9 17:35:51 UTC 2010 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Sep 16 12:34:36 2010 from raftier
user system{~} 
```

SSH : secure shell maintenance access on system level

ILO access



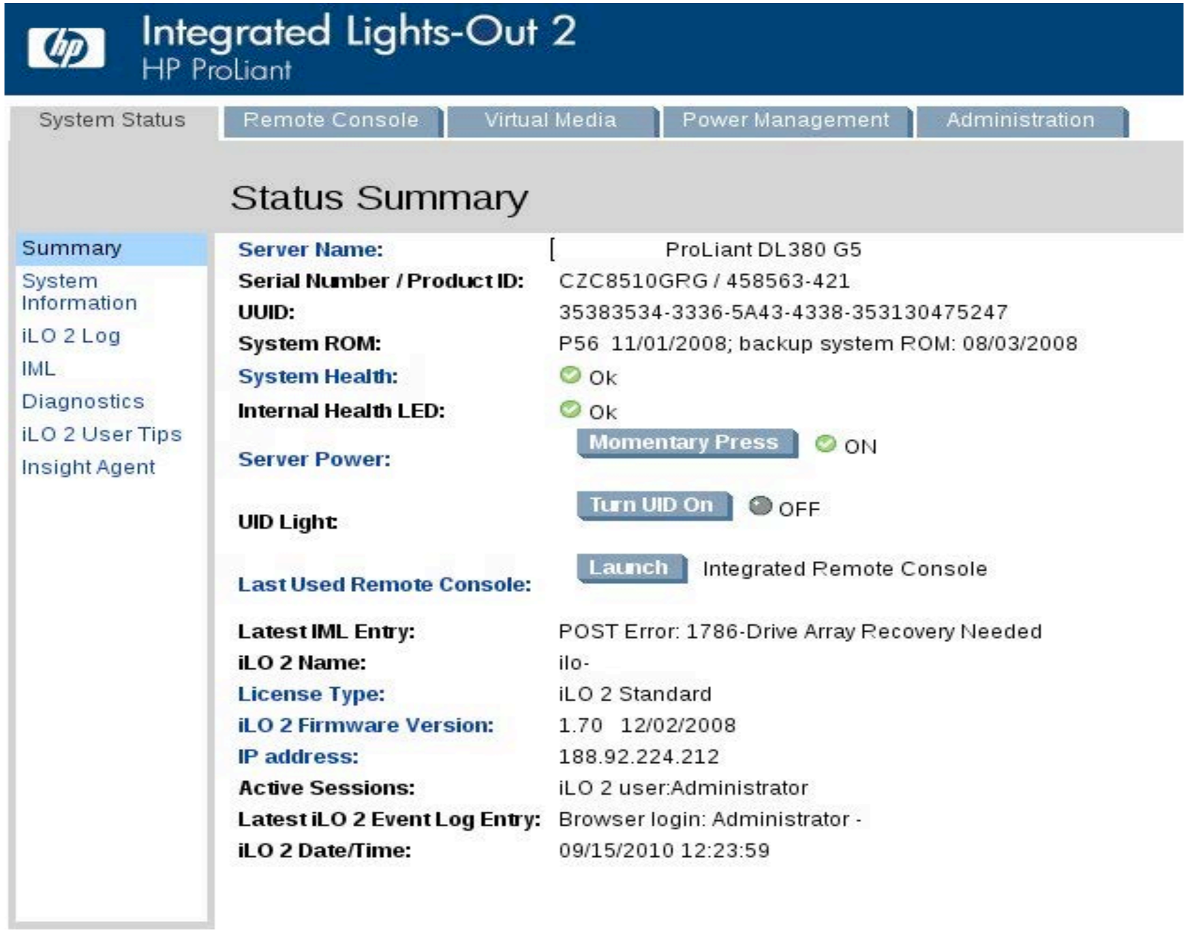
Login name:

Password:

© Copyright 2008 Hewlett-Packard Development Company, L.P.

Contains security software licensed from RSA Data Security Inc.
Portions Copyright 1989, 1991, 1992 by Carnegie Mellon University
Derivative Work - 1996, 1998-2000 Copyright 1996, 1998-2000 The Regents of the University of California

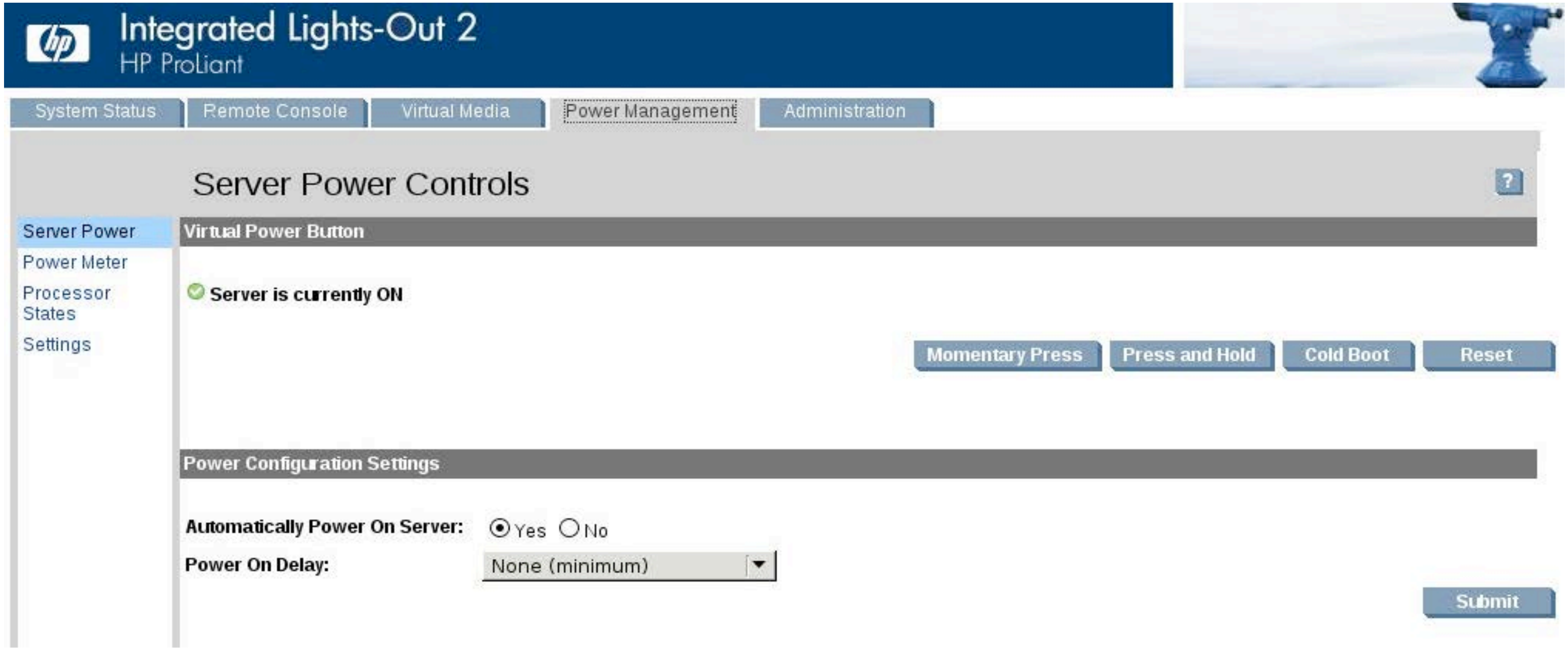
ILO access



Status Summary

Server Name:	[ProLiant DL380 G5
Serial Number / Product ID:	CZC8510GRG / 458563-421
UUID:	35383534-3336-5A43-4338-353130475247
System ROM:	P56 11/01/2008; backup system ROM: 08/03/2008
System Health:	✓ Ok
Internal Health LED:	✓ Ok
Server Power:	Momentary Press ✓ ON
UID Light:	Turn UID On ● OFF
Last Used Remote Console:	Launch Integrated Remote Console
Latest IML Entry:	POST Error: 1786-Drive Array Recovery Needed
iLO 2 Name:	ilo-
License Type:	iLO 2 Standard
iLO 2 Firmware Version:	1.70 12/02/2008
IP address:	188.92.224.212
Active Sessions:	iLO 2 user:Administrator
Latest iLO 2 Event Log Entry:	Browser login: Administrator -
iLO 2 Date/Time:	09/15/2010 12:23:59

ILO access



Server Power Controls

Virtual Power Button

✓ Server is currently ON

Power Configuration Settings

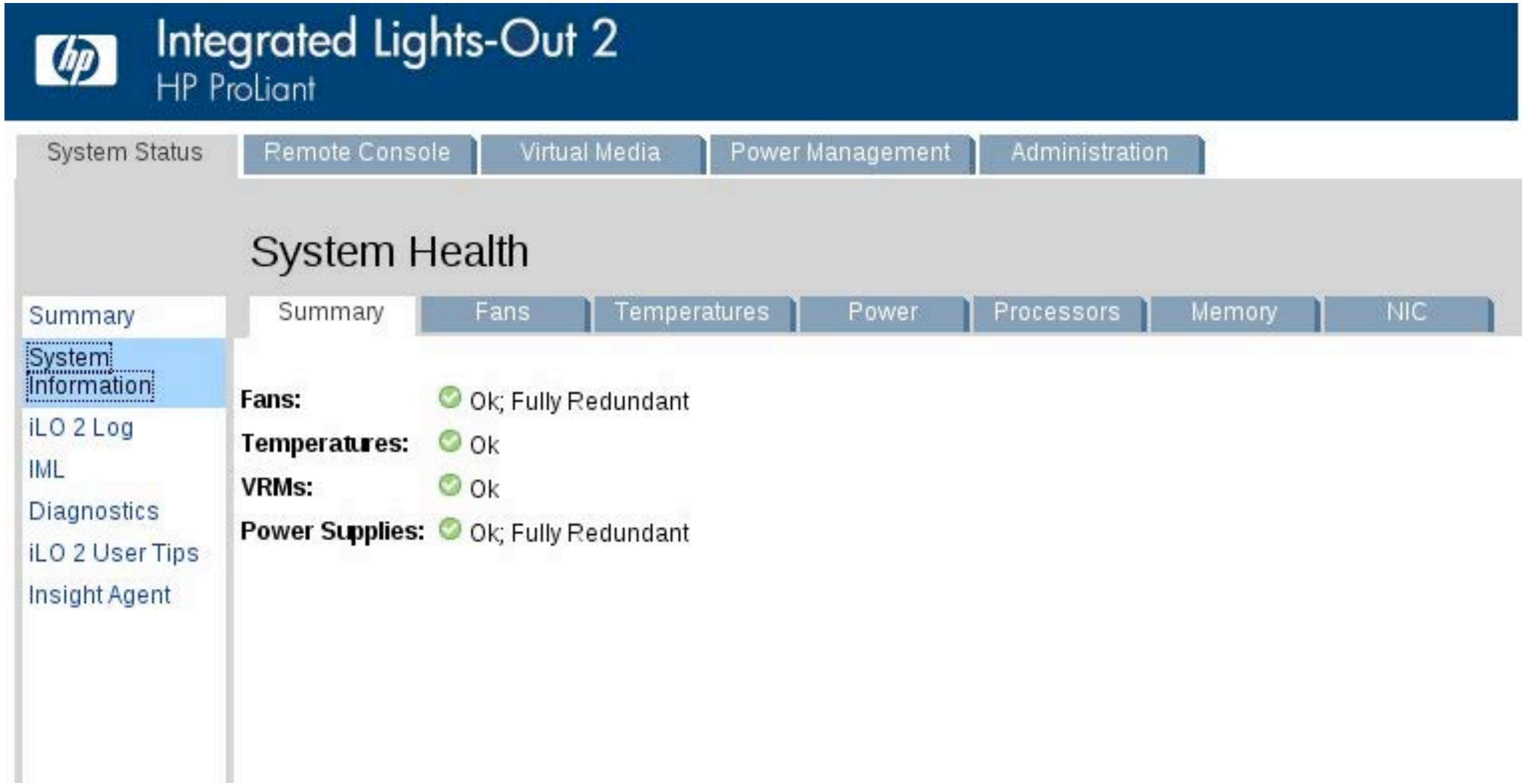
Automatically Power On Server: ● Yes ○ No

Power On Delay: None (minimum)

ILO Power: button press for “power on/power off”

Attention: It really works !

ILO access



System Health

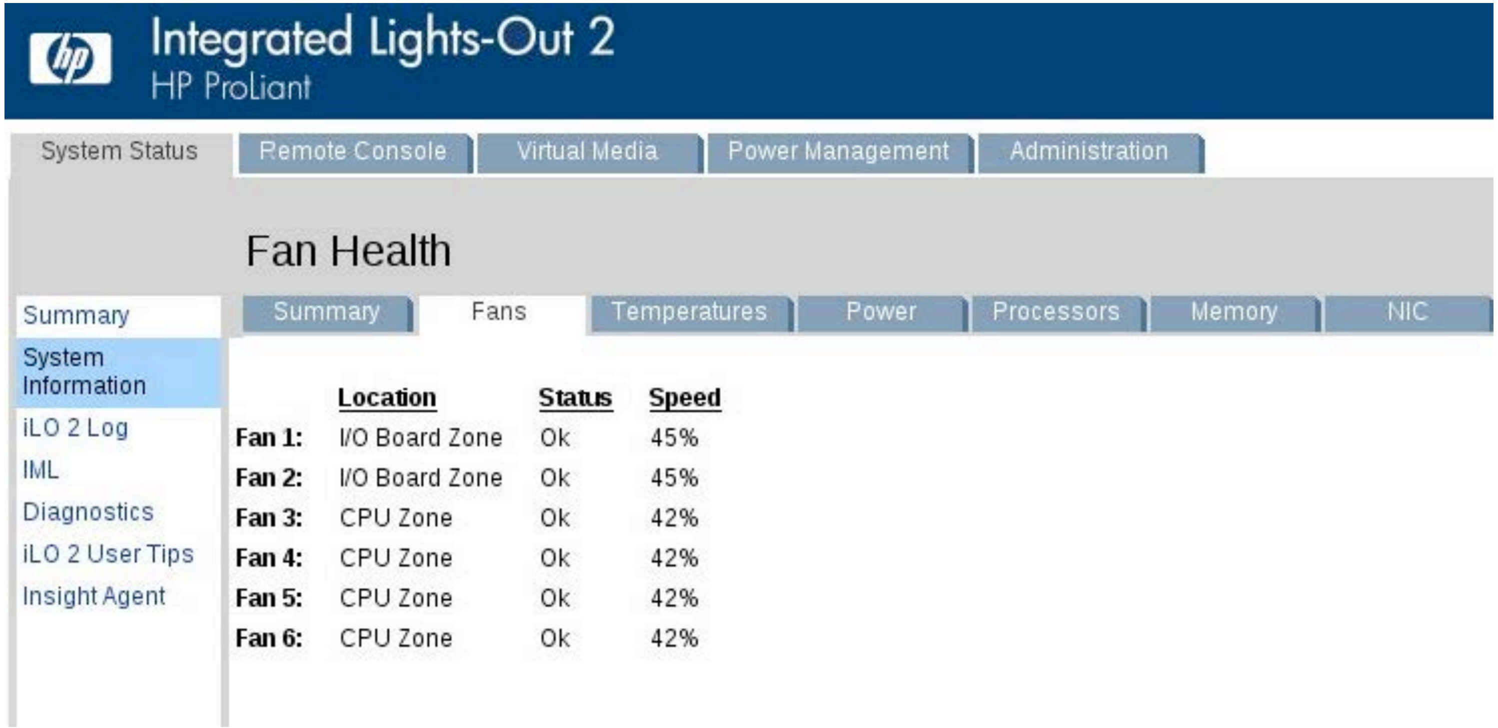
Fans: ✓ Ok; Fully Redundant

Temperatures: ✓ Ok

VRMs: ✓ Ok

Power Supplies: ✓ Ok; Fully Redundant

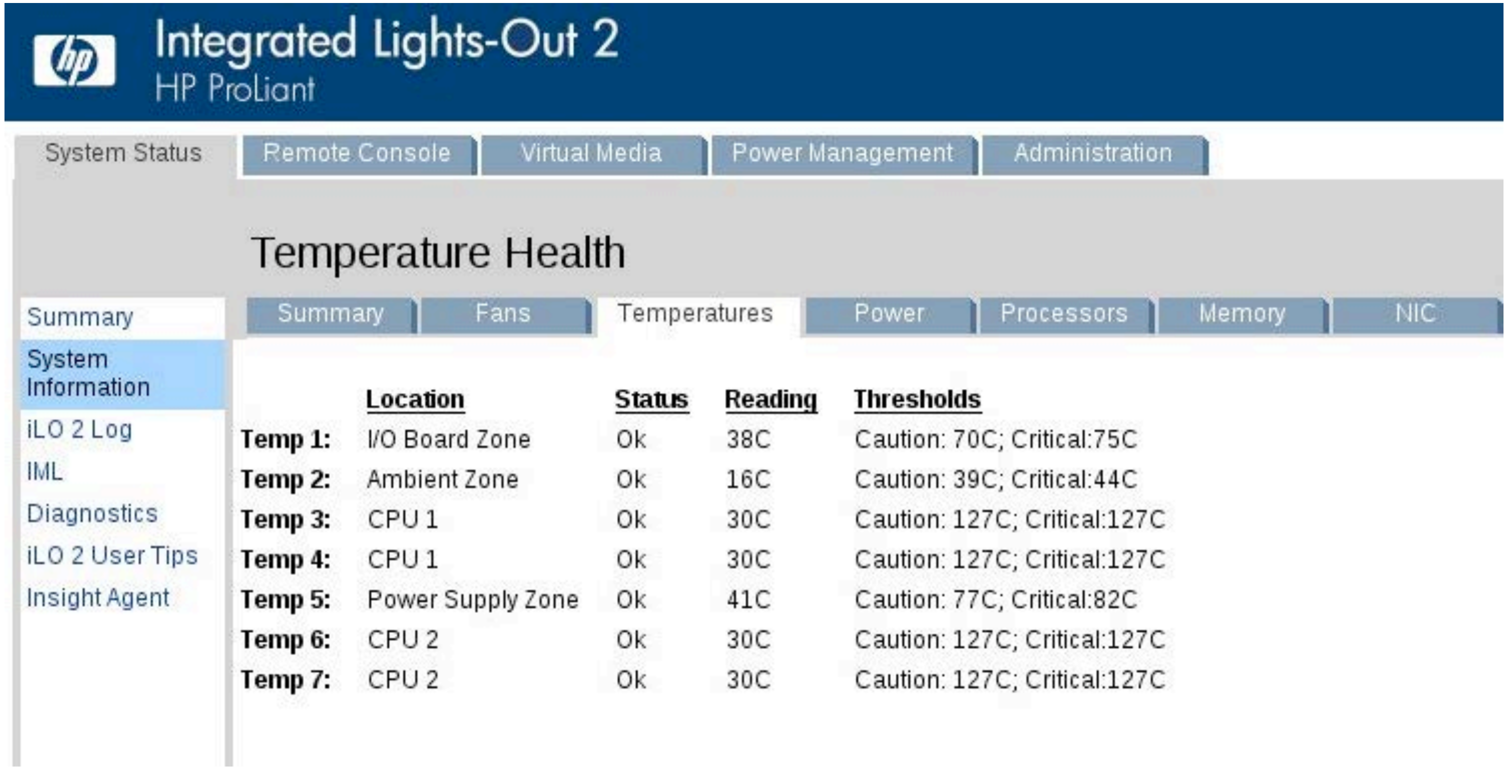
ILO access



Fan Health

	Location	Status	Speed
Fan 1:	I/O Board Zone	Ok	45%
Fan 2:	I/O Board Zone	Ok	45%
Fan 3:	CPU Zone	Ok	42%
Fan 4:	CPU Zone	Ok	42%
Fan 5:	CPU Zone	Ok	42%
Fan 6:	CPU Zone	Ok	42%

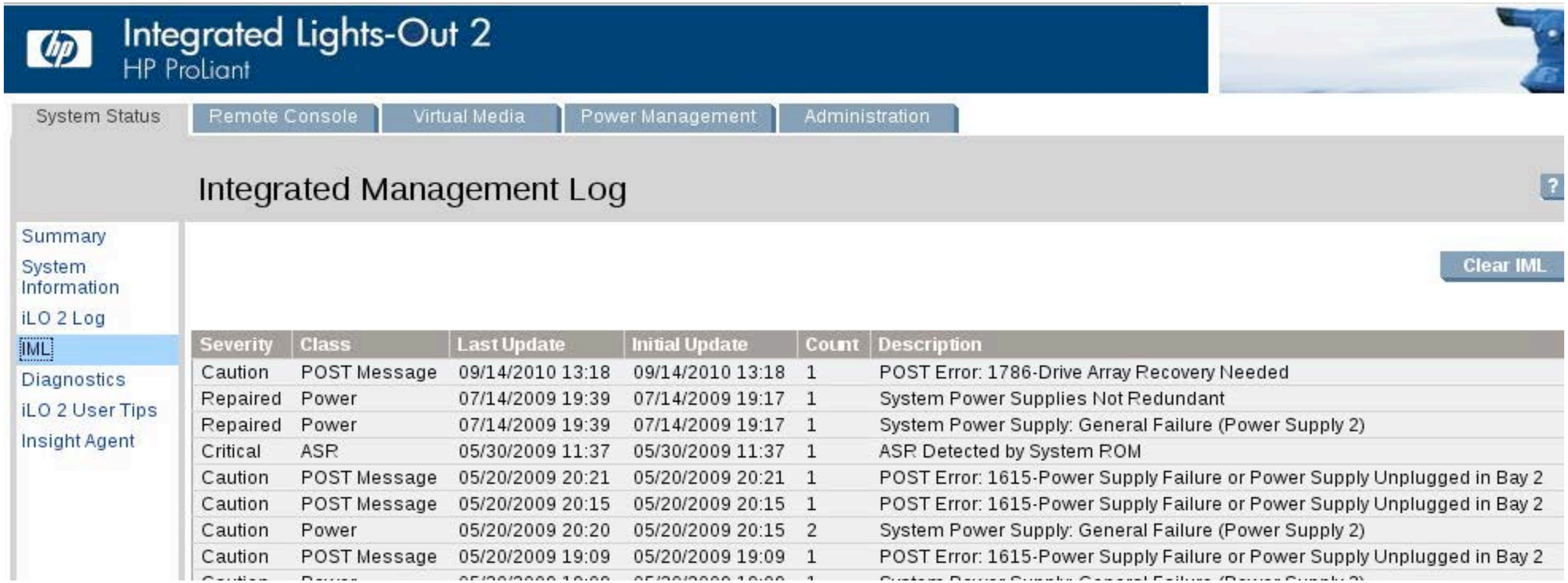
ILO access



Temperature Health

	Location	Status	Reading	Thresholds
Temp 1:	I/O Board Zone	Ok	38C	Caution: 70C; Critical:75C
Temp 2:	Ambient Zone	Ok	16C	Caution: 39C; Critical:44C
Temp 3:	CPU 1	Ok	30C	Caution: 127C; Critical:127C
Temp 4:	CPU 1	Ok	30C	Caution: 127C; Critical:127C
Temp 5:	Power Supply Zone	Ok	41C	Caution: 77C; Critical:82C
Temp 6:	CPU 2	Ok	30C	Caution: 127C; Critical:127C
Temp 7:	CPU 2	Ok	30C	Caution: 127C; Critical:127C

ILO access

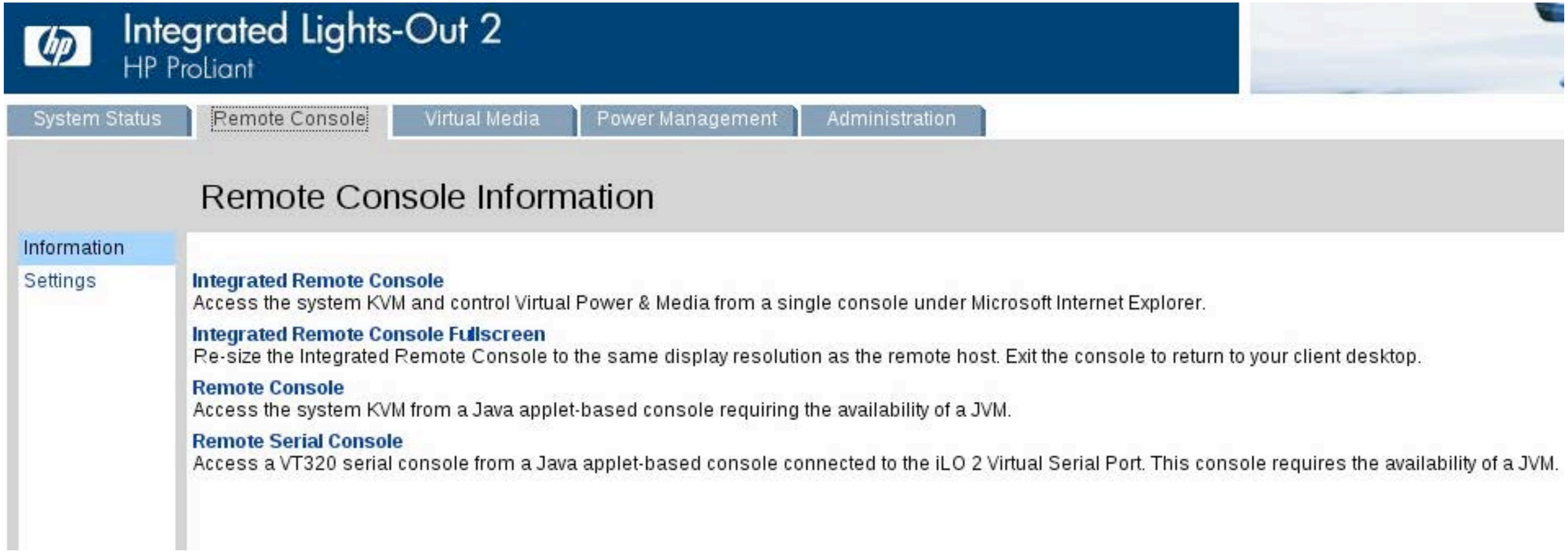


Integrated Management Log

Severity	Class	Last Update	Initial Update	Count	Description
Caution	POST Message	09/14/2010 13:18	09/14/2010 13:18	1	POST Error: 1786-Drive Array Recovery Needed
Repaired	Power	07/14/2009 19:39	07/14/2009 19:17	1	System Power Supplies Not Redundant
Repaired	Power	07/14/2009 19:39	07/14/2009 19:17	1	System Power Supply: General Failure (Power Supply 2)
Critical	ASR	05/30/2009 11:37	05/30/2009 11:37	1	ASR Detected by System ROM
Caution	POST Message	05/20/2009 20:21	05/20/2009 20:21	1	POST Error: 1615-Power Supply Failure or Power Supply Unplugged in Bay 2
Caution	POST Message	05/20/2009 20:15	05/20/2009 20:15	1	POST Error: 1615-Power Supply Failure or Power Supply Unplugged in Bay 2
Caution	Power	05/20/2009 20:20	05/20/2009 20:15	2	System Power Supply: General Failure (Power Supply 2)
Caution	POST Message	05/20/2009 19:09	05/20/2009 19:09	1	POST Error: 1615-Power Supply Failure or Power Supply Unplugged in Bay 2

Log information from low level hardware components

ILO access



Remote Console Information

Integrated Remote Console
Access the system KVM and control Virtual Power & Media from a single console under Microsoft Internet Explorer.

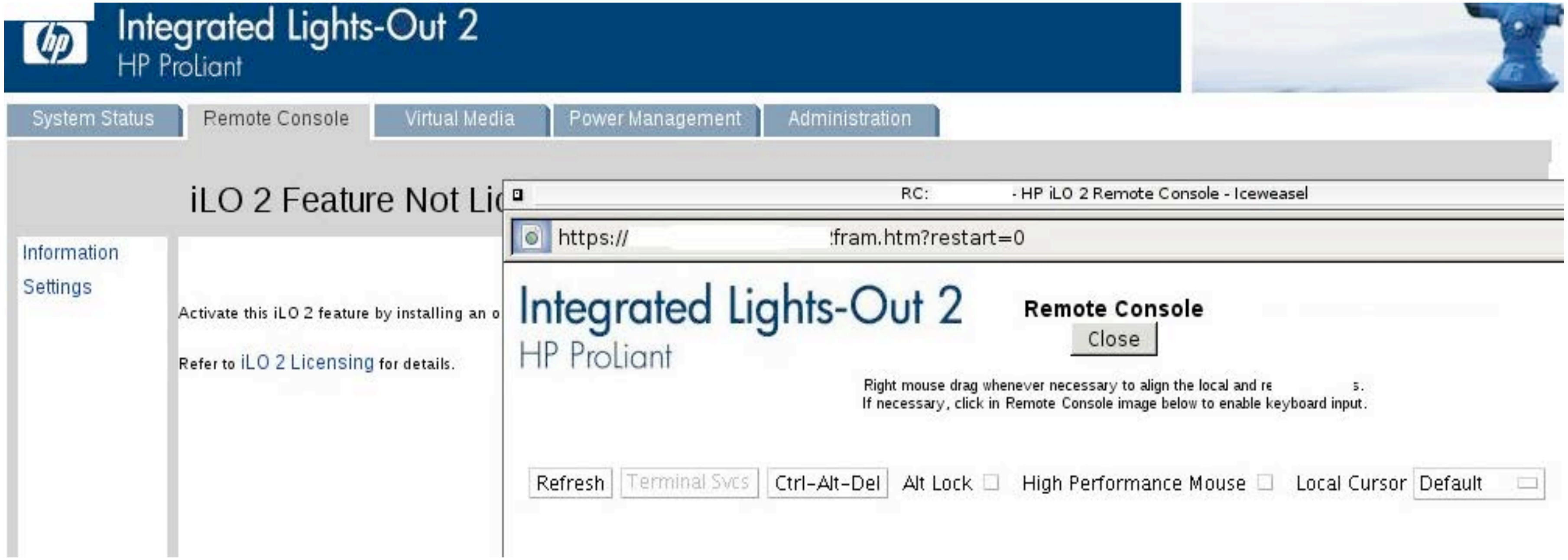
Integrated Remote Console Fullscreen
Re-size the Integrated Remote Console to the same display resolution as the remote host. Exit the console to return to your client desktop.

Remote Console
Access the system KVM from a Java applet-based console requiring the availability of a JVM.

Remote Serial Console
Access a VT320 serial console from a Java applet-based console connected to the iLO 2 Virtual Serial Port. This console requires the availability of a JVM.

ILO System remote console information: choose the remote console

ILO access



iLO 2 Feature Not Lic

Activate this iLO 2 feature by installing an o

Refer to iLO 2 Licensing for details.

https:// !fram.htm?restart=0

Right mouse drag whenever necessary to align the local and re s.
If necessary, click in Remote Console image below to enable keyboard input.

ILO: access the OS via the ILO remote console

6. Technical Details

Commonly used SW components
System and Bios Hardening

FINFISHER
IT INTRUSION

Commonly used SW components

- Daemontools:
  - Used to provide a high level of availability for the installed core SW components
- Ssh:
  - Remote secure command-line access to the iProxy components for management purposes
- Ntp:
  - Being used for synchronizing the time on the iProxy components
- Syslog-ng:
  - Used for collecting all system and application events
  - Possibility to send a copy of the events to a defined e-mail address
- Shorewall (Except the NDP-Component):
  - High level configuration user-land frontend for the onboard firewalls

System and Bios Hardening

- System:
  - Firewall configured deny all, allow specifically
  - Removed unnecessary services
  - Disabled Ipv6
  - No direct root login allowed
  - Minimal software stack
  - Security optimized configuration for all services
- Bios:
  - Boot order and media
  - Bios password
  - In case of power failure: Auto power on

7. Incident Handling

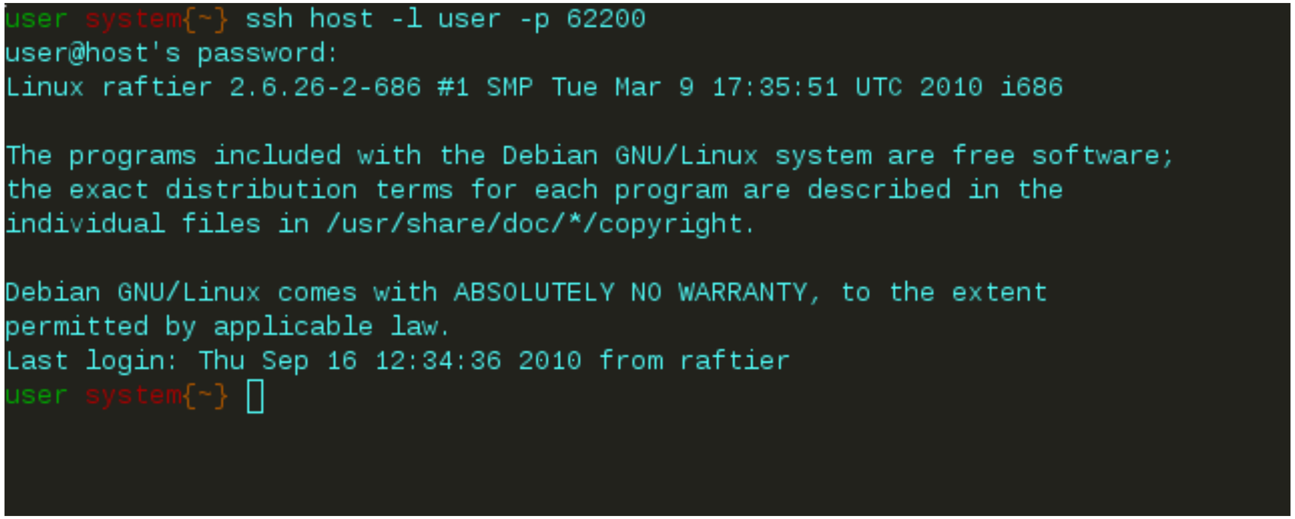
Hands on / System Training

WWW.GAMMAGROUP.COM

FINFISHER

IT INTRUSION

SSH access



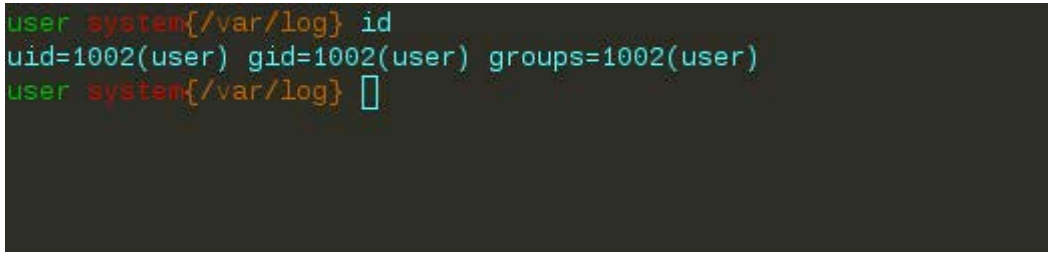
```
user system{~} ssh host -l user -p 62200
user@host's password:
Linux raftier 2.6.26-2-686 #1 SMP Tue Mar 9 17:35:51 UTC 2010 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Sep 16 12:34:36 2010 from raftier
user system{~} ▯
```

Secure shell / SSH is used for accessing the iProxy-components:

Command: ssh host –l user –p 62200

Parameters: host: hostname
-l username
-p portnumber

User Identification

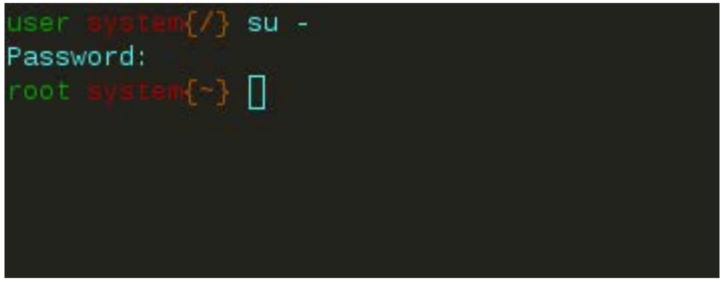


```
user system{/var/log} id
uid=1002(user) gid=1002(user) groups=1002(user)
user system{/var/log}
```

The command `id` is used for identifying the active user:

```
Command:     id
Parameters:  n.a.
Output:      uid (user-id), gid (group-id), groups (groups the
              user belongs to)
```

Using root-privileges



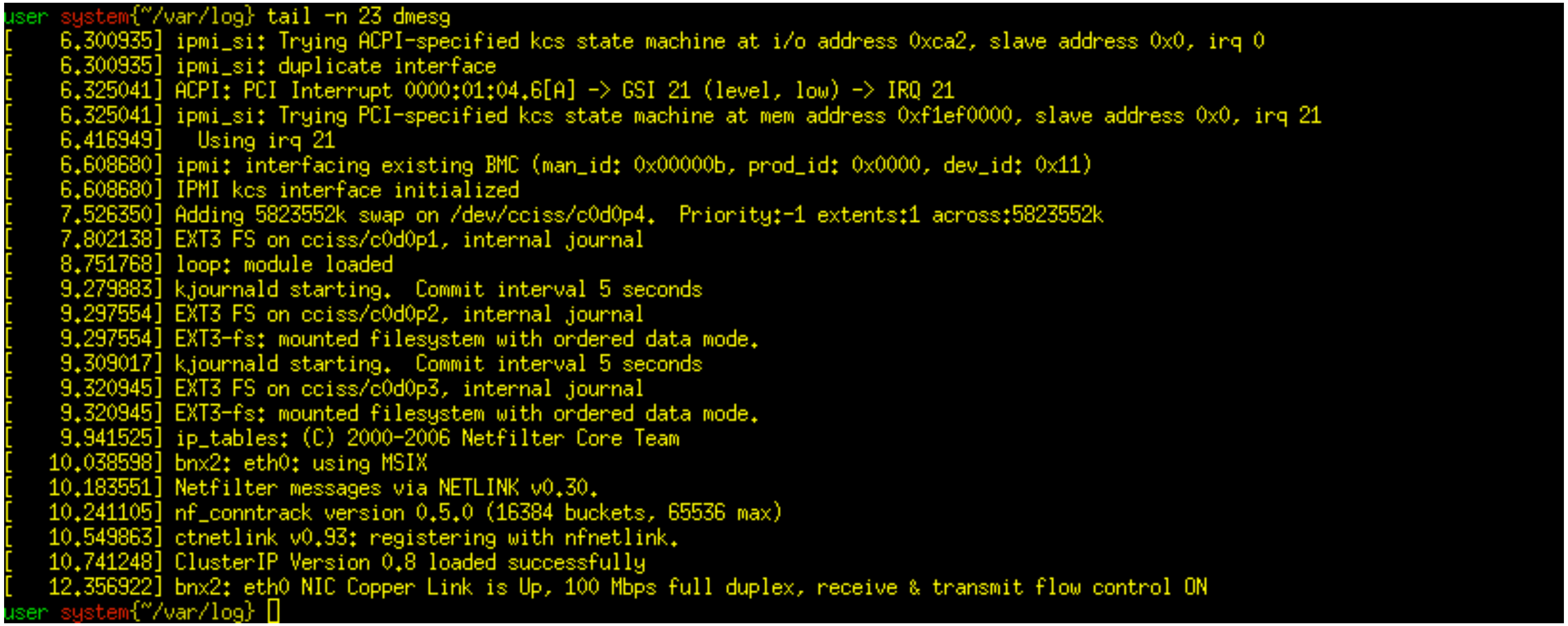
```
user system{/} su -
Password:
root system{~} []
```

The command `su` is used to gain root-privileges:

Command: su -
Parameters: - (to start the root-shell from home-path)
Output: n.a.

Attention: You are working on live systems, you may break things!

Kernel debug messages

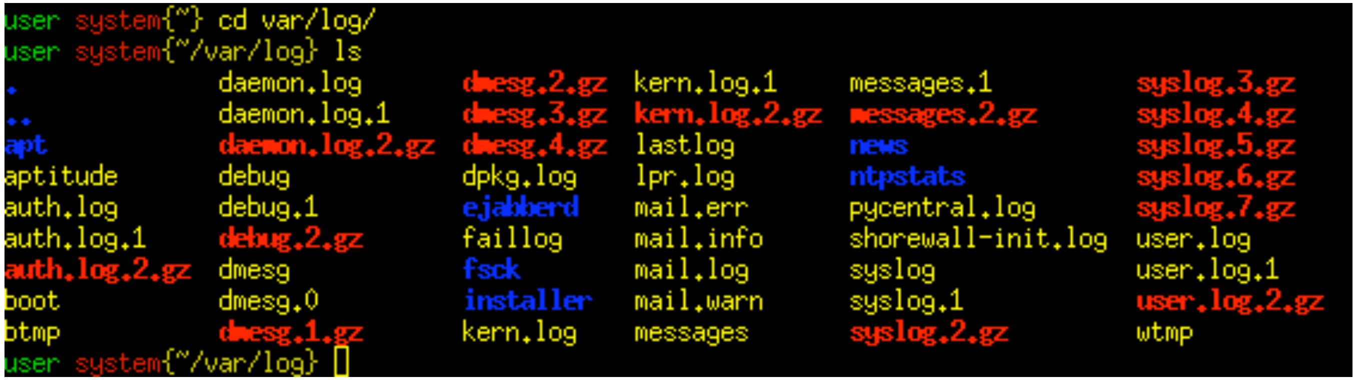


```
user system{~/var/log} tail -n 23 dmesg
[    6.300935] ipmi_si: Trying ACPI-specified kcs state machine at i/o address 0xca2, slave address 0x0, irq 0
[    6.300935] ipmi_si: duplicate interface
[    6.325041] ACPI: PCI Interrupt 0000:01:04.6[A] -> GSI 21 (level, low) -> IRQ 21
[    6.325041] ipmi_si: Trying PCI-specified kcs state machine at mem address 0xf1ef0000, slave address 0x0, irq 21
[    6.416949]   Using irq 21
[    6.608680] ipmi: interfacing existing BMC (man_id: 0x00000b, prod_id: 0x0000, dev_id: 0x11)
[    6.608680] IPMI kcs interface initialized
[    7.526350] Adding 5823552k swap on /dev/cciss/c0d0p4.  Priority:-1 extents:1 across:5823552k
[    7.802138] EXT3 FS on cciss/c0d0p1, internal journal
[    8.751768] loop: module loaded
[    9.279883] kjournald starting.  Commit interval 5 seconds
[    9.297554] EXT3 FS on cciss/c0d0p2, internal journal
[    9.297554] EXT3-fs: mounted filesystem with ordered data mode.
[    9.309017] kjournald starting.  Commit interval 5 seconds
[    9.320945] EXT3 FS on cciss/c0d0p3, internal journal
[    9.320945] EXT3-fs: mounted filesystem with ordered data mode.
[    9.941525] ip_tables: (C) 2000-2006 Netfilter Core Team
[   10.038598] bnx2: eth0: using MSIX
[   10.183551] Netfilter messages via NETLINK v0.30.
[   10.241105] nf_conntrack version 0.5.0 (16384 buckets, 65536 max)
[   10.549863] ctnetlink v0.93: registering with nfnetlink.
[   10.741248] ClusterIP Version 0.8 loaded successfully
[   12.356922] bnx2: eth0 NIC Copper Link is Up, 100 Mbps full duplex, receive & transmit flow control ON
user system{~/var/log}
```

The command `dmesg` is used for displaying kernel debug messages:

Command:	dmesg
Parameters:	n.a.
Output:	see above

Dir containing all system logs

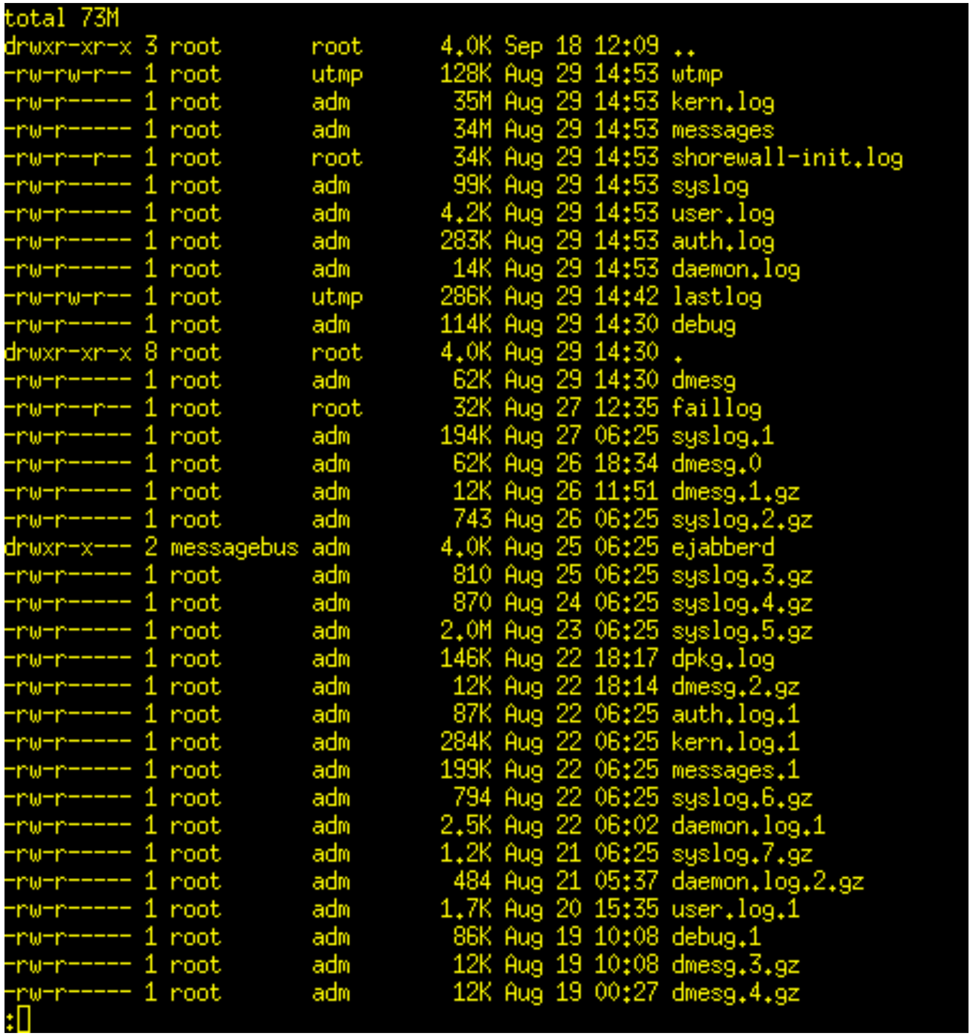


```
user system{~} cd var/log/
user system{~/var/log} ls
.              daemon.log       dmesg.2.gz  kern.log.1     messages.1          syslog.3.gz
..             daemon.log.1     dmesg.3.gz  kern.log.2.gz  messages.2.gz       syslog.4.gz
apt            daemon.log.2.gz  dmesg.4.gz  lastlog        news                syslog.5.gz
aptitude       debug            dpkg.log    lpr.log        ntpstats            syslog.6.gz
auth.log       debug.1          ejabberd    mail.err       pycentral.log       syslog.7.gz
auth.log.1     debug.2.gz       faillog     mail.info      shorewall-init.log  user.log
auth.log.2.gz  dmesg            fsck        mail.log       syslog              user.log.1
boot           dmesg.0          installer   mail.warn      syslog.1            user.log.2.gz
btmp           dmesg.1.gz       kern.log    messages       syslog.2.gz         wtmp
user system{~/var/log} 
```

The command `ls` lists the directory containing all system log files:

Command:	ls
Parameters:	i.e: -lah
Path:	/var/log
Important Log Files:	daemon.log, messages, kern.log, auth.log, dmesg, syslog

List log directory by date



```
total 73M
drwxr-xr-x 3 root       root      4.0K Sep 18 12:09 ..
-rw-rw-r-- 1 root       utmp      128K Aug 29 14:53 wtmp
-rw-r----- 1 root       adm        35M Aug 29 14:53 kern.log
-rw-r----- 1 root       adm        34M Aug 29 14:53 messages
-rw-r--r-- 1 root       root       34K Aug 29 14:53 shorewall-init.log
-rw-r----- 1 root       adm        99K Aug 29 14:53 syslog
-rw-r----- 1 root       adm       4.2K Aug 29 14:53 user.log
-rw-r----- 1 root       adm       283K Aug 29 14:53 auth.log
-rw-r----- 1 root       adm        14K Aug 29 14:53 daemon.log
-rw-rw-r-- 1 root       utmp      286K Aug 29 14:42 lastlog
-rw-r----- 1 root       adm       114K Aug 29 14:30 debug
drwxr-xr-x 8 root       root      4.0K Aug 29 14:30 .
-rw-r----- 1 root       adm        62K Aug 29 14:30 dmesg
-rw-r--r-- 1 root       root       32K Aug 27 12:35 faillog
-rw-r----- 1 root       adm       194K Aug 27 06:25 syslog.1
-rw-r----- 1 root       adm        62K Aug 26 18:34 dmesg.0
-rw-r----- 1 root       adm        12K Aug 26 11:51 dmesg.1.gz
-rw-r----- 1 root       adm        743 Aug 26 06:25 syslog.2.gz
drwxr-x--- 2 messagebus adm       4.0K Aug 25 06:25 ejabberd
-rw-r----- 1 root       adm        810 Aug 25 06:25 syslog.3.gz
-rw-r----- 1 root       adm        870 Aug 24 06:25 syslog.4.gz
-rw-r----- 1 root       adm       2.0M Aug 23 06:25 syslog.5.gz
-rw-r----- 1 root       adm       146K Aug 22 18:17 dpkg.log
-rw-r----- 1 root       adm        12K Aug 22 18:14 dmesg.2.gz
-rw-r----- 1 root       adm        87K Aug 22 06:25 auth.log.1
-rw-r----- 1 root       adm       284K Aug 22 06:25 kern.log.1
-rw-r----- 1 root       adm       199K Aug 22 06:25 messages.1
-rw-r----- 1 root       adm        794 Aug 22 06:25 syslog.6.gz
-rw-r----- 1 root       adm       2.5K Aug 22 06:02 daemon.log.1
-rw-r----- 1 root       adm       1.2K Aug 21 06:25 syslog.7.gz
-rw-r----- 1 root       adm        484 Aug 21 05:37 daemon.log.2.gz
-rw-r----- 1 root       adm       1.7K Aug 20 15:35 user.log.1
-rw-r----- 1 root       adm        86K Aug 19 10:08 debug.1
-rw-r----- 1 root       adm        12K Aug 19 10:08 dmesg.3.gz
-rw-r----- 1 root       adm        12K Aug 19 00:27 dmesg.4.gz
:
```

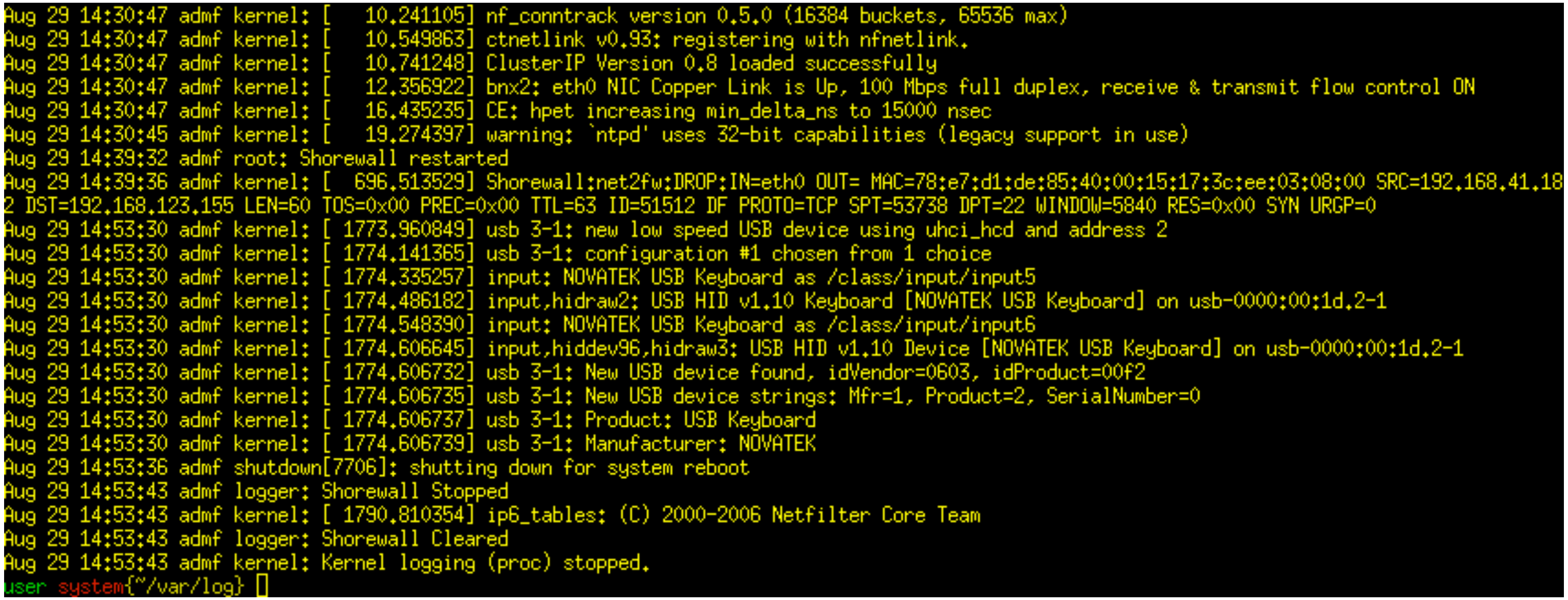
List the log directory by date:

Command: ls -laht

Parameters: -l = list
-a= all
-h= human readable
-t = sort by date

Output: all files sorted by date

Messages log

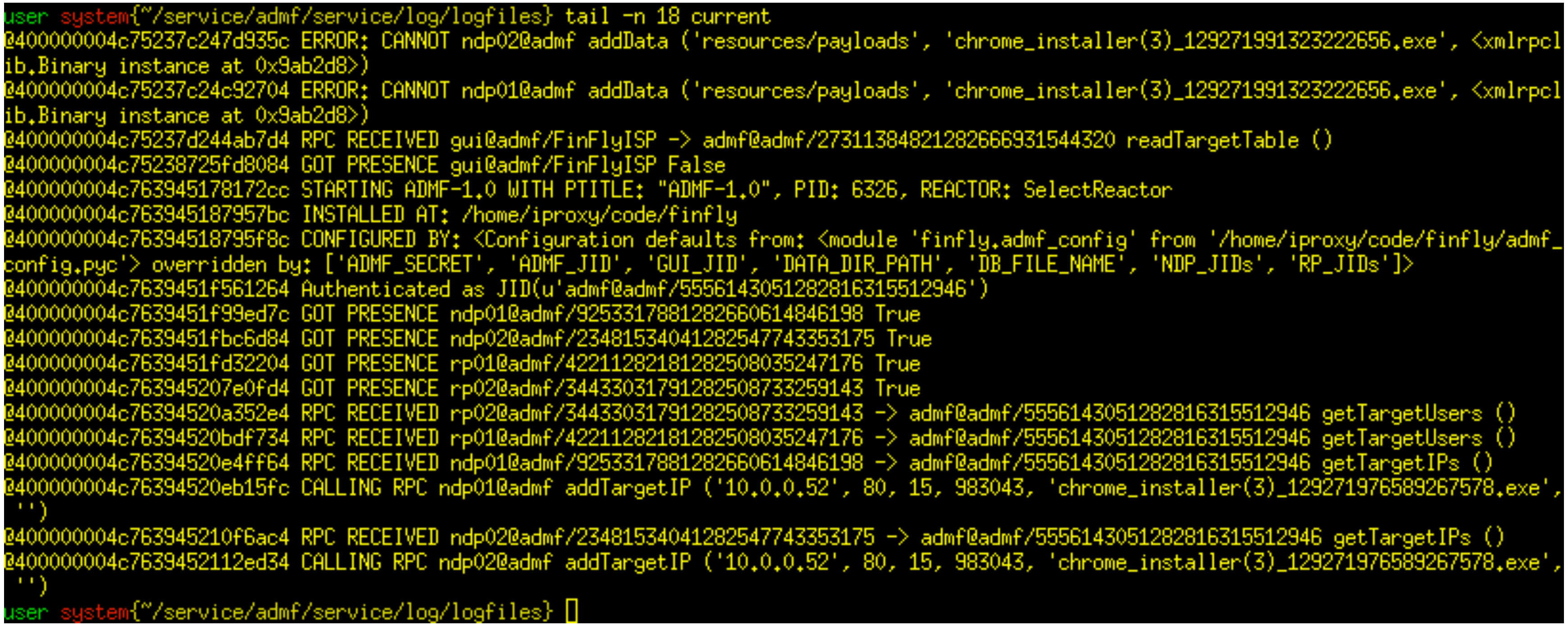


```
Aug 29 14:30:47 admf kernel: [   10.241105] nf_conntrack version 0.5.0 (16384 buckets, 65536 max)
Aug 29 14:30:47 admf kernel: [   10.549863] ctnetlink v0.93: registering with nfnetlink.
Aug 29 14:30:47 admf kernel: [   10.741248] ClusterIP Version 0.8 loaded successfully
Aug 29 14:30:47 admf kernel: [   12.356922] bnx2: eth0 NIC Copper Link is Up, 100 Mbps full duplex, receive & transmit flow control ON
Aug 29 14:30:47 admf kernel: [   16.435235] CE: hpet increasing min_delta_ns to 15000 nsec
Aug 29 14:30:45 admf kernel: [   19.274397] warning: `ntpd' uses 32-bit capabilities (legacy support in use)
Aug 29 14:39:32 admf root: Shorewall restarted
Aug 29 14:39:36 admf kernel: [  696.513529] Shorewall:net2fw:DROP:IN=eth0 OUT= MAC=78:e7:d1:de:85:40:00:15:17:3c:ee:03:08:00 SRC=192.168.41.18
2 DST=192.168.123.155 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=51512 DF PROTO=TCP SPT=53738 DPT=22 WINDOW=5840 RES=0x00 SYN URGP=0
Aug 29 14:53:30 admf kernel: [ 1773.960849] usb 3-1: new low speed USB device using uhci_hcd and address 2
Aug 29 14:53:30 admf kernel: [ 1774.141365] usb 3-1: configuration #1 chosen from 1 choice
Aug 29 14:53:30 admf kernel: [ 1774.335257] input: NOVATEK USB Keyboard as /class/input/input5
Aug 29 14:53:30 admf kernel: [ 1774.486182] input,hidraw2: USB HID v1.10 Keyboard [NOVATEK USB Keyboard] on usb-0000:00:1d.2-1
Aug 29 14:53:30 admf kernel: [ 1774.548390] input: NOVATEK USB Keyboard as /class/input/input6
Aug 29 14:53:30 admf kernel: [ 1774.606645] input,hiddev96,hidraw3: USB HID v1.10 Device [NOVATEK USB Keyboard] on usb-0000:00:1d.2-1
Aug 29 14:53:30 admf kernel: [ 1774.606732] usb 3-1: New USB device found, idVendor=0603, idProduct=00f2
Aug 29 14:53:30 admf kernel: [ 1774.606735] usb 3-1: New USB device strings: Mfr=1, Product=2, SerialNumber=0
Aug 29 14:53:30 admf kernel: [ 1774.606737] usb 3-1: Product: USB Keyboard
Aug 29 14:53:30 admf kernel: [ 1774.606739] usb 3-1: Manufacturer: NOVATEK
Aug 29 14:53:36 admf shutdown[7706]: shutting down for system reboot
Aug 29 14:53:43 admf logger: Shorewall Stopped
Aug 29 14:53:43 admf kernel: [ 1790.810354] ip6_tables: (C) 2000-2006 Netfilter Core Team
Aug 29 14:53:43 admf logger: Shorewall Cleared
Aug 29 14:53:43 admf kernel: Kernel logging (proc) stopped.
user system{~/var/log} []
```

The messages file contains all important system logs:

Command: cat
Parameters: /var/log/messages
Output: see above

ADMF Log

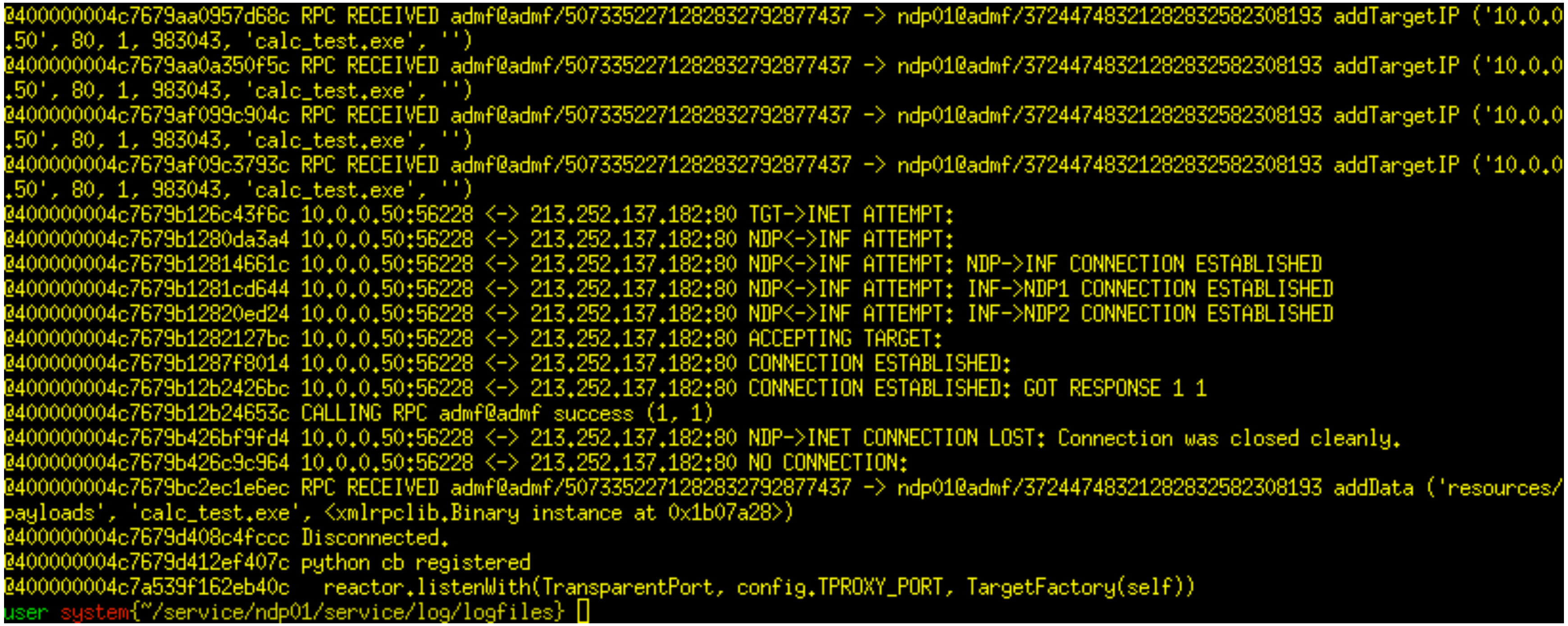


```
user system{~/service/admf/service/log/logfiles} tail -n 18 current
@400000004c75237c247d935c ERROR: CANNOT ndp02@admf addData ('resources/payloads', 'chrome_installer(3)_129271991323222656.exe', <xmlrpcl
ib.Binary instance at 0x9ab2d8>)
@400000004c75237c24c92704 ERROR: CANNOT ndp01@admf addData ('resources/payloads', 'chrome_installer(3)_129271991323222656.exe', <xmlrpcl
ib.Binary instance at 0x9ab2d8>)
@400000004c75237d244ab7d4 RPC RECEIVED gui@admf/FinFlyISP -> admf@admf/27311384821282666931544320 readTargetTable ()
@400000004c75238725fd8084 GOT PRESENCE gui@admf/FinFlyISP False
@400000004c763945178172cc STARTING ADMF-1.0 WITH PTITLE: "ADMF-1.0", PID: 6326, REACTOR: SelectReactor
@400000004c763945187957bc INSTALLED AT: /home/iproxy/code/finfly
@400000004c76394518795f8c CONFIGURED BY: <Configuration defaults from: <module 'finfly.admf_config' from '/home/iproxy/code/finfly/admf_
config.pyc'> overridden by: ['ADMF_SECRET', 'ADMF_JID', 'GUI_JID', 'DATA_DIR_PATH', 'DB_FILE_NAME', 'NDP_JIDs', 'RP_JIDs']>
@400000004c7639451f561264 Authenticated as JID(u'admf@admf/5556143051282816315512946')
@400000004c7639451f99ed7c GOT PRESENCE ndp01@admf/9253317881282660614846198 True
@400000004c7639451fbc6d84 GOT PRESENCE ndp02@admf/23481534041282547743353175 True
@400000004c7639451fd32204 GOT PRESENCE rp01@admf/42211282181282508035247176 True
@400000004c763945207e0fd4 GOT PRESENCE rp02@admf/34433031791282508733259143 True
@400000004c76394520a352e4 RPC RECEIVED rp02@admf/34433031791282508733259143 -> admf@admf/5556143051282816315512946 getTargetUsers ()
@400000004c76394520bdf734 RPC RECEIVED rp01@admf/42211282181282508035247176 -> admf@admf/5556143051282816315512946 getTargetUsers ()
@400000004c76394520e4ff64 RPC RECEIVED ndp01@admf/9253317881282660614846198 -> admf@admf/5556143051282816315512946 getTargetIPs ()
@400000004c76394520eb15fc CALLING RPC ndp01@admf addTargetIP ('10.0.0.52', 80, 15, 983043, 'chrome_installer(3)_129271976589267578.exe',
 '')
@400000004c763945210f6ac4 RPC RECEIVED ndp02@admf/23481534041282547743353175 -> admf@admf/5556143051282816315512946 getTargetIPs ()
@400000004c7639452112ed34 CALLING RPC ndp02@admf addTargetIP ('10.0.0.52', 80, 15, 983043, 'chrome_installer(3)_129271976589267578.exe',
 '')
user system{~/service/admf/service/log/logfiles} ▯
```

The ADMF log file contains all messages from the admf service:

Log File Path:	/home/iproxy/service/admf/service/log/logfiles/current
Command:	less
Parameter:	/home/iproxy/service/admf/service/log/logfiles/current
Output:	see above

NDP Log

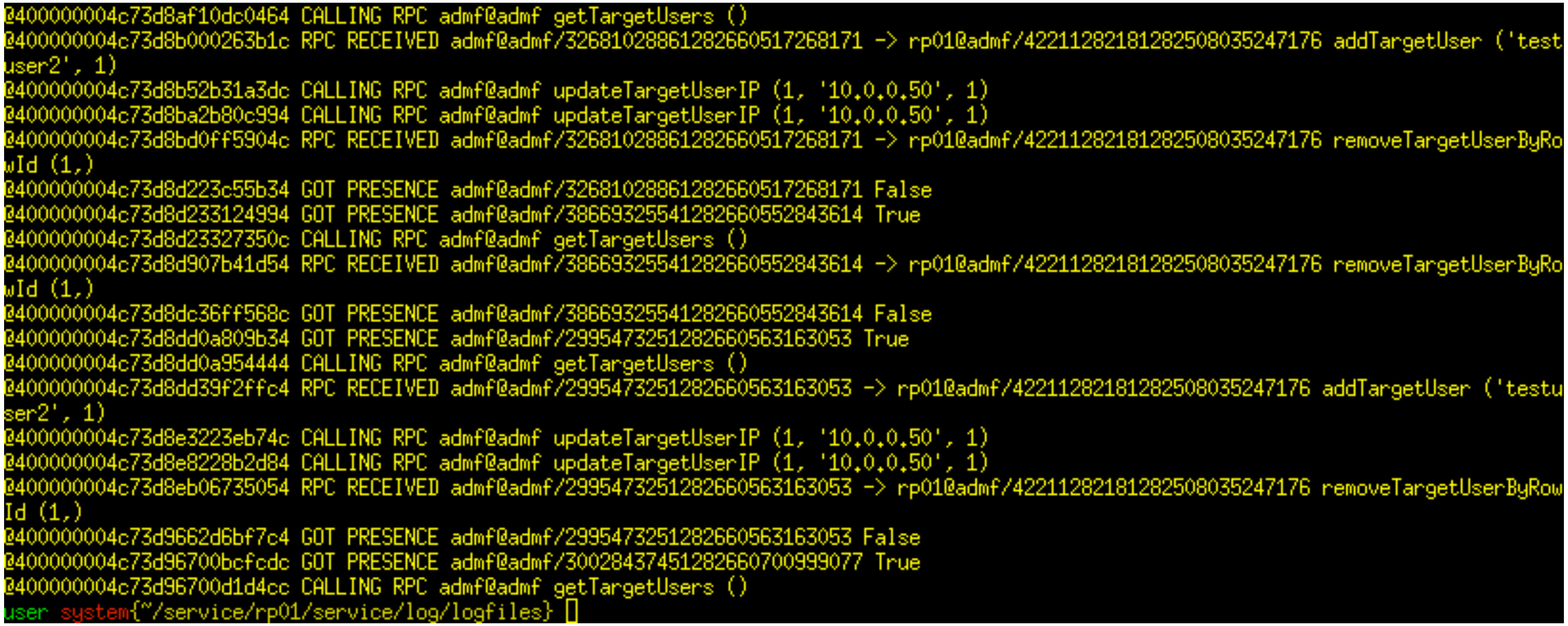


```
@400000004c7679aa0957d68c RPC RECEIVED admf@admf/5073352271282832792877437 -> ndp01@admf/3724474832128283258230819З addTargetIP ('10.0.0
.50', 80, 1, 983043, 'calc_test.exe', '')
@400000004c7679aa0a350f5c RPC RECEIVED admf@admf/5073352271282832792877437 -> ndp01@admf/37244748321282832582308193 addTargetIP ('10.0.0
.50', 80, 1, 983043, 'calc_test.exe', '')
@400000004c7679af099c904c RPC RECEIVED admf@admf/5073352271282832792877437 -> ndp01@admf/37244748321282832582308193 addTargetIP ('10.0.0
.50', 80, 1, 983043, 'calc_test.exe', '')
@400000004c7679af09c3793c RPC RECEIVED admf@admf/5073352271282832792877437 -> ndp01@admf/37244748321282832582308193 addTargetIP ('10.0.0
.50', 80, 1, 983043, 'calc_test.exe', '')
@400000004c7679b126c43f6c 10.0.0.50:56228 <-> 213.252.137.182:80 TGT->INET ATTEMPT:
@400000004c7679b1280da3a4 10.0.0.50:56228 <-> 213.252.137.182:80 NDP<->INF ATTEMPT:
@400000004c7679b12814661c 10.0.0.50:56228 <-> 213.252.137.182:80 NDP<->INF ATTEMPT: NDP->INF CONNECTION ESTABLISHED
@400000004c7679b1281cd644 10.0.0.50:56228 <-> 213.252.137.182:80 NDP<->INF ATTEMPT: INF->NDP1 CONNECTION ESTABLISHED
@400000004c7679b12820ed24 10.0.0.50:56228 <-> 213.252.137.182:80 NDP<->INF ATTEMPT: INF->NDP2 CONNECTION ESTABLISHED
@400000004c7679b1282127bc 10.0.0.50:56228 <-> 213.252.137.182:80 ACCEPTING TARGET:
@400000004c7679b1287f8014 10.0.0.50:56228 <-> 213.252.137.182:80 CONNECTION ESTABLISHED:
@400000004c7679b12b2426bc 10.0.0.50:56228 <-> 213.252.137.182:80 CONNECTION ESTABLISHED: GOT RESPONSE 1 1
@400000004c7679b12b24653c CALLING RPC admf@admf success (1, 1)
@400000004c7679b426bf9fd4 10.0.0.50:56228 <-> 213.252.137.182:80 NDP->INET CONNECTION LOST: Connection was closed cleanly.
@400000004c7679b426c9c964 10.0.0.50:56228 <-> 213.252.137.182:80 NO CONNECTION:
@400000004c7679bc2ec1e6ec RPC RECEIVED admf@admf/5073352271282832792877437 -> ndp01@admf/37244748321282832582308193 addData ('resources/
payloads', 'calc_test.exe', <xmlrpclib.Binary instance at 0x1b07a28>)
@400000004c7679d408c4fccc Disconnected.
@400000004c7679d412ef407c python cb registered
@400000004c7a539f162eb40c   reactor.listenWith(TransparentPort, config.TPROXY_PORT, TargetFactory(self))
user system{~/service/ndp01/service/log/logfiles} 
```

The NDP log file contains all messages from the ndp service:

Log File Path: /home/iproxy/service/ndp/service/log/logfiles/current
Command: less
Parameter: /home/iproxy/service/ndp/service/log/logfiles/current
Output: see above

RP Log

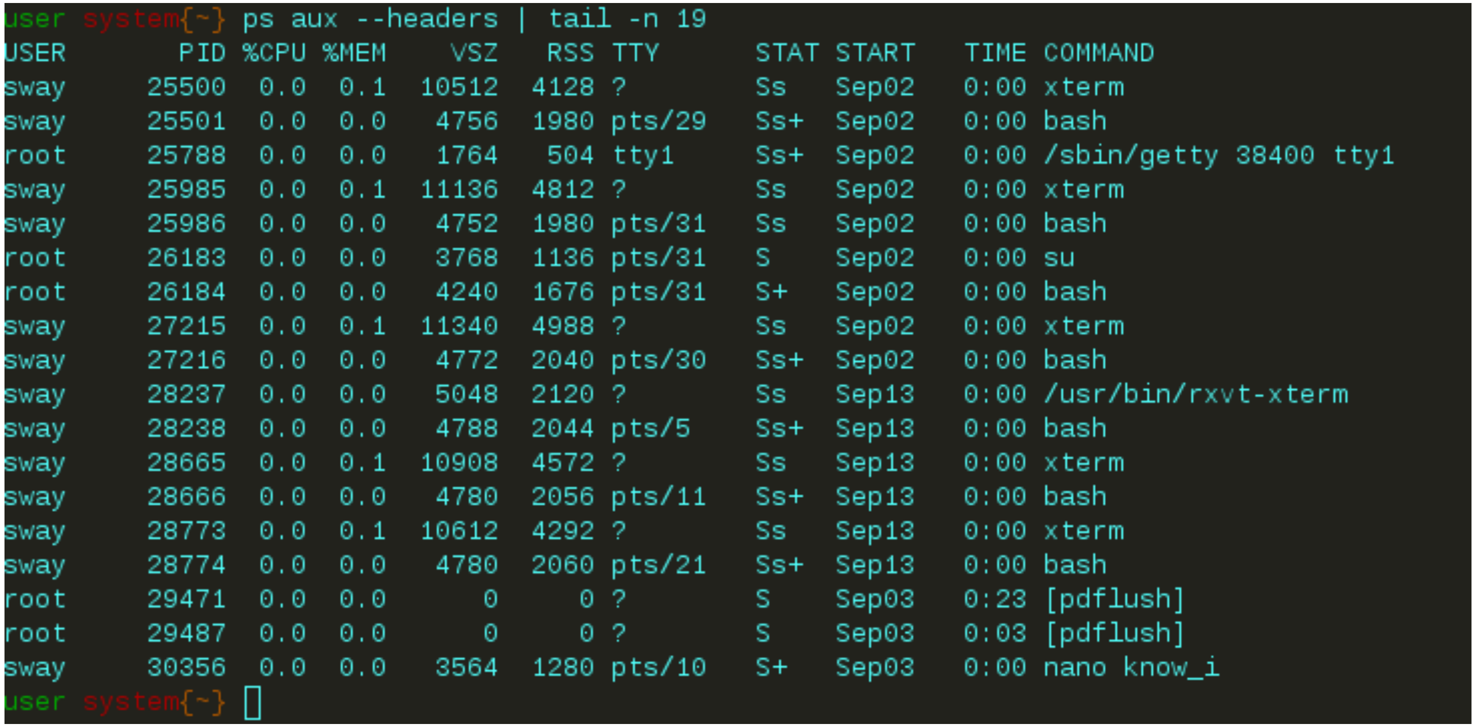


```
@400000004c73d8af10dc0464 CALLING RPC admf@admf getTargetUsers ()
@400000004c73d8b000263b1c RPC RECEIVED admf@admf/32681028861282660517268171 -> rp01@admf/42211282181282508035247176 addTargetUser ('test
user2', 1)
@400000004c73d8b52b31a3dc CALLING RPC admf@admf updateTargetUserIP (1, '10.0.0.50', 1)
@400000004c73d8ba2b80c994 CALLING RPC admf@admf updateTargetUserIP (1, '10.0.0.50', 1)
@400000004c73d8bd0ff5904c RPC RECEIVED admf@admf/32681028861282660517268171 -> rp01@admf/42211282181282508035247176 removeTargetUserByRo
wId (1,)
@400000004c73d8d223c55b34 GOT PRESENCE admf@admf/32681028861282660517268171 False
@400000004c73d8d233124994 GOT PRESENCE admf@admf/38669325541282660552843614 True
@400000004c73d8d23327350c CALLING RPC admf@admf getTargetUsers ()
@400000004c73d8d907b41d54 RPC RECEIVED admf@admf/38669325541282660552843614 -> rp01@admf/42211282181282508035247176 removeTargetUserByRo
wId (1,)
@400000004c73d8dc36ff568c GOT PRESENCE admf@admf/38669325541282660552843614 False
@400000004c73d8dd0a809b34 GOT PRESENCE admf@admf/2995473251282660563163053 True
@400000004c73d8dd0a954444 CALLING RPC admf@admf getTargetUsers ()
@400000004c73d8dd39f2ffc4 RPC RECEIVED admf@admf/2995473251282660563163053 -> rp01@admf/42211282181282508035247176 addTargetUser ('testu
ser2', 1)
@400000004c73d8e3223eb74c CALLING RPC admf@admf updateTargetUserIP (1, '10.0.0.50', 1)
@400000004c73d8e8228b2d84 CALLING RPC admf@admf updateTargetUserIP (1, '10.0.0.50', 1)
@400000004c73d8eb06735054 RPC RECEIVED admf@admf/2995473251282660563163053 -> rp01@admf/42211282181282508035247176 removeTargetUserByRow
Id (1,)
@400000004c73d9662d6bf7c4 GOT PRESENCE admf@admf/2995473251282660563163053 False
@400000004c73d96700bcfcdc GOT PRESENCE admf@admf/30028437451282660700999077 True
@400000004c73d96700d1d4cc CALLING RPC admf@admf getTargetUsers ()
user system{~/service/rp01/service/log/logfiles} 
```

The RP log file contains all messages from the rp service:

Log File Path:	/home/iproxy/service/rp/service/log/logfiles/current
Command:	less
Parameter:	/home/iproxy/service/rp/service/log/logfiles/current
Output:	see above

List all running processes

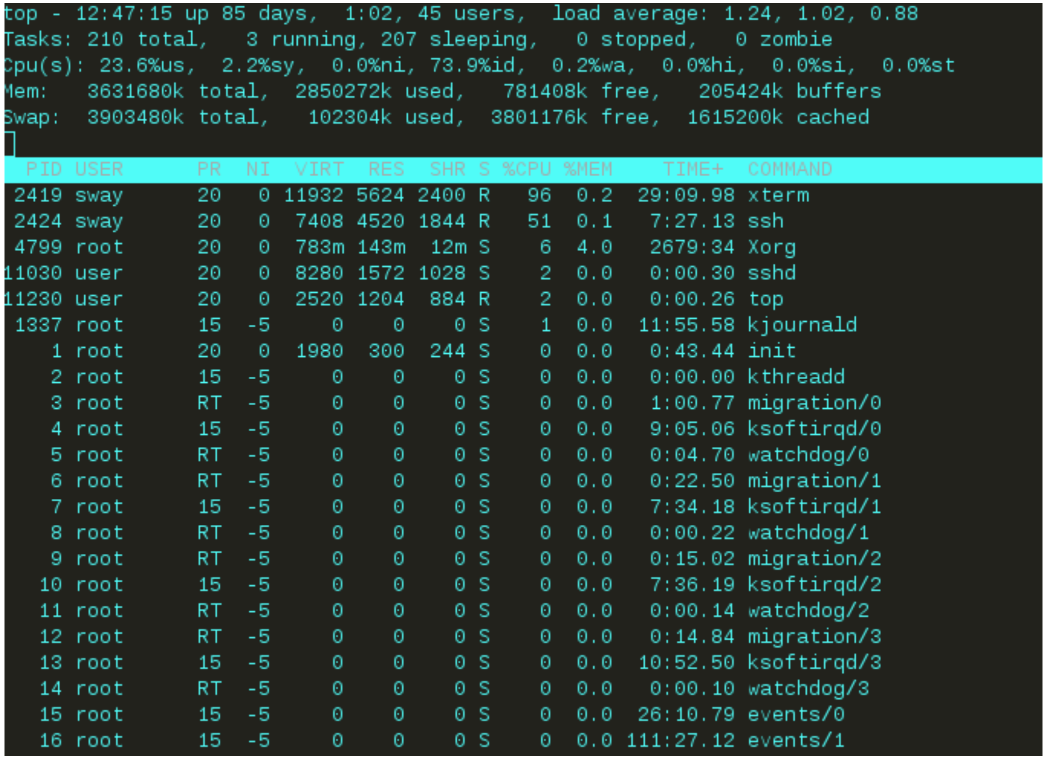


```
user system{~} ps aux --headers | tail -n 19
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
sway     25500  0.0  0.1 10512 4128 ?        Ss   Sep02   0:00 xterm
sway     25501  0.0  0.0  4756 1980 pts/29   Ss+  Sep02   0:00 bash
root     25788  0.0  0.0  1764  504 tty1     Ss+  Sep02   0:00 /sbin/getty 38400 tty1
sway     25985  0.0  0.1 11136 4812 ?        Ss   Sep02   0:00 xterm
sway     25986  0.0  0.0  4752 1980 pts/31   Ss   Sep02   0:00 bash
root     26183  0.0  0.0  3768 1136 pts/31   S    Sep02   0:00 su
root     26184  0.0  0.0  4240 1676 pts/31   S+   Sep02   0:00 bash
sway     27215  0.0  0.1 11340 4988 ?        Ss   Sep02   0:00 xterm
sway     27216  0.0  0.0  4772 2040 pts/30   Ss+  Sep02   0:00 bash
sway     28237  0.0  0.0  5048 2120 ?        Ss   Sep13   0:00 /usr/bin/rxvt-xterm
sway     28238  0.0  0.0  4788 2044 pts/5    Ss+  Sep13   0:00 bash
sway     28665  0.0  0.1 10908 4572 ?        Ss   Sep13   0:00 xterm
sway     28666  0.0  0.0  4780 2056 pts/11   Ss+  Sep13   0:00 bash
sway     28773  0.0  0.1 10612 4292 ?        Ss   Sep13   0:00 xterm
sway     28774  0.0  0.0  4780 2060 pts/21   Ss+  Sep13   0:00 bash
root     29471  0.0  0.0     0    0 ?        S    Sep03   0:23 [pdflush]
root     29487  0.0  0.0     0    0 ?        S    Sep03   0:03 [pdflush]
sway     30356  0.0  0.0  3564 1280 pts/10   S+   Sep03   0:00 nano know_i
user system{~} ▯
```

The command `ps` lists processes running on the system:

Command: ps -aux
Parameters: -a = all processes, -u = list by user-id, -x = list by tty
Output: all running processes, see above

Realtime system performance statistics

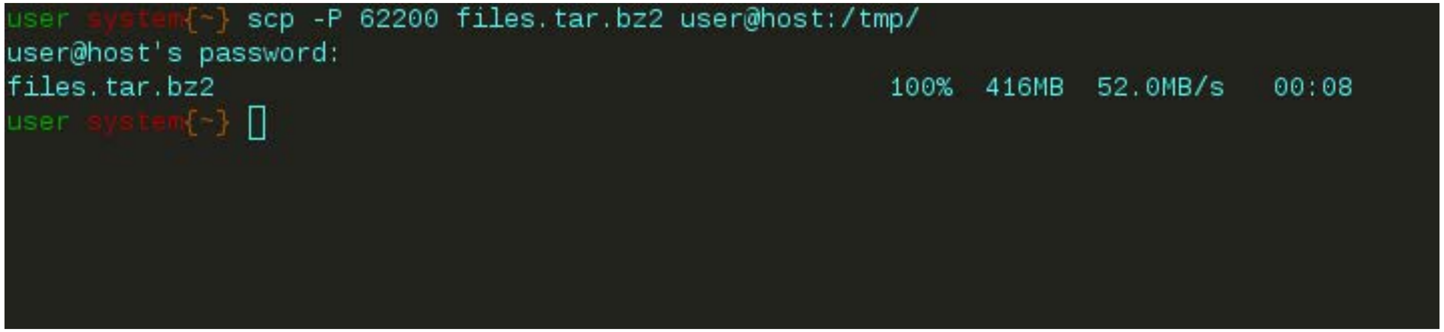


```
top - 12:47:15 up 85 days,  1:02, 45 users,  load average: 1.24, 1.02, 0.88
Tasks: 210 total,   3 running, 207 sleeping,   0 stopped,   0 zombie
Cpu(s): 23.6%us,  2.2%sy,  0.0%ni, 73.9%id,  0.2%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   3631680k total,  2850272k used,   781408k free,   205424k buffers
Swap:  3903480k total,   102304k used,  3801176k free,  1615200k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2419 sway      20   0 11932 5624 2400 R   96  0.2  29:09.98 xterm
 2424 sway      20   0  7408 4520 1844 R   51  0.1   7:27.13 ssh
 4799 root      20   0  783m 143m  12m S    6  4.0   2679:34 Xorg
11030 user      20   0  8280 1572 1028 S    2  0.0   0:00.30 sshd
11230 user      20   0  2520 1204  884 R    2  0.0   0:00.26 top
 1337 root      15  -5     0    0    0 S    1  0.0  11:55.58 kjournald
    1 root      20   0  1980  300  244 S    0  0.0   0:43.44 init
    2 root      15  -5     0    0    0 S    0  0.0   0:00.00 kthreadd
    3 root      RT  -5     0    0    0 S    0  0.0   1:00.77 migration/0
    4 root      15  -5     0    0    0 S    0  0.0   9:05.06 ksoftirqd/0
    5 root      RT  -5     0    0    0 S    0  0.0   0:04.70 watchdog/0
    6 root      RT  -5     0    0    0 S    0  0.0   0:22.50 migration/1
    7 root      15  -5     0    0    0 S    0  0.0   7:34.18 ksoftirqd/1
    8 root      RT  -5     0    0    0 S    0  0.0   0:00.22 watchdog/1
    9 root      RT  -5     0    0    0 S    0  0.0   0:15.02 migration/2
   10 root      15  -5     0    0    0 S    0  0.0   7:36.19 ksoftirqd/2
   11 root      RT  -5     0    0    0 S    0  0.0   0:00.14 watchdog/2
   12 root      RT  -5     0    0    0 S    0  0.0   0:14.84 migration/3
   13 root      15  -5     0    0    0 S    0  0.0  10:52.50 ksoftirqd/3
   14 root      RT  -5     0    0    0 S    0  0.0   0:00.10 watchdog/3
   15 root      15  -5     0    0    0 S    0  0.0  26:10.79 events/0
   16 root      15  -5     0    0    0 S    0  0.0 111:27.12 events/1
```

The command `top` lists in realtime all processes running on the system:

Command: top –d1
Parameters: -d = delay in seconds (here = 1 second)
Output: see above

Secure filecopy over SSH

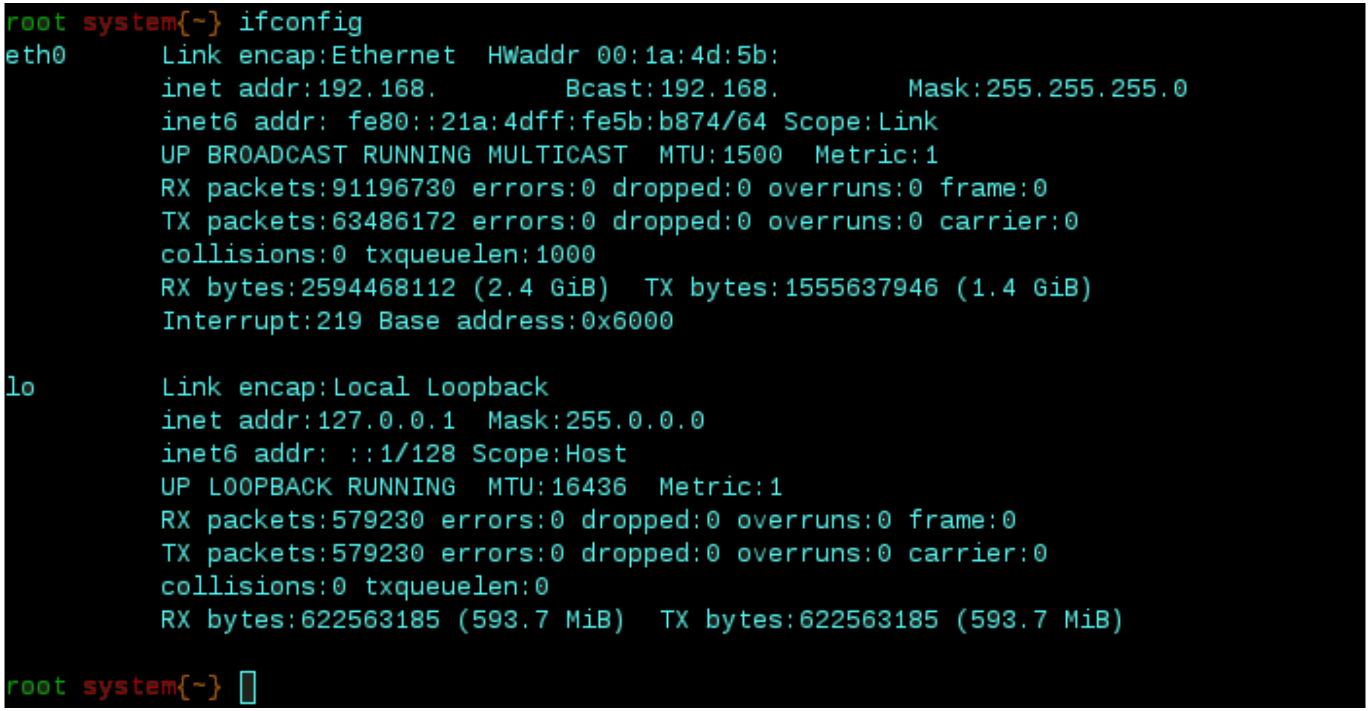


```
user system{~} scp -P 62200 files.tar.bz2 user@host:/tmp/
user@host's password:
files.tar.bz2                                     100%  416MB  52.0MB/s   00:08
user system{~} 
```

The command `scp` is used for copying files from one server to another via ssh:

Command: scp –P 62200 files user@host:/directory
Parameters: -P 62200 (Portnumber to be used),
files = the filename to be copied,
user@host = user who logs into the target system,
/directory: where to copy the file
Output: see above

List active network interface configurations

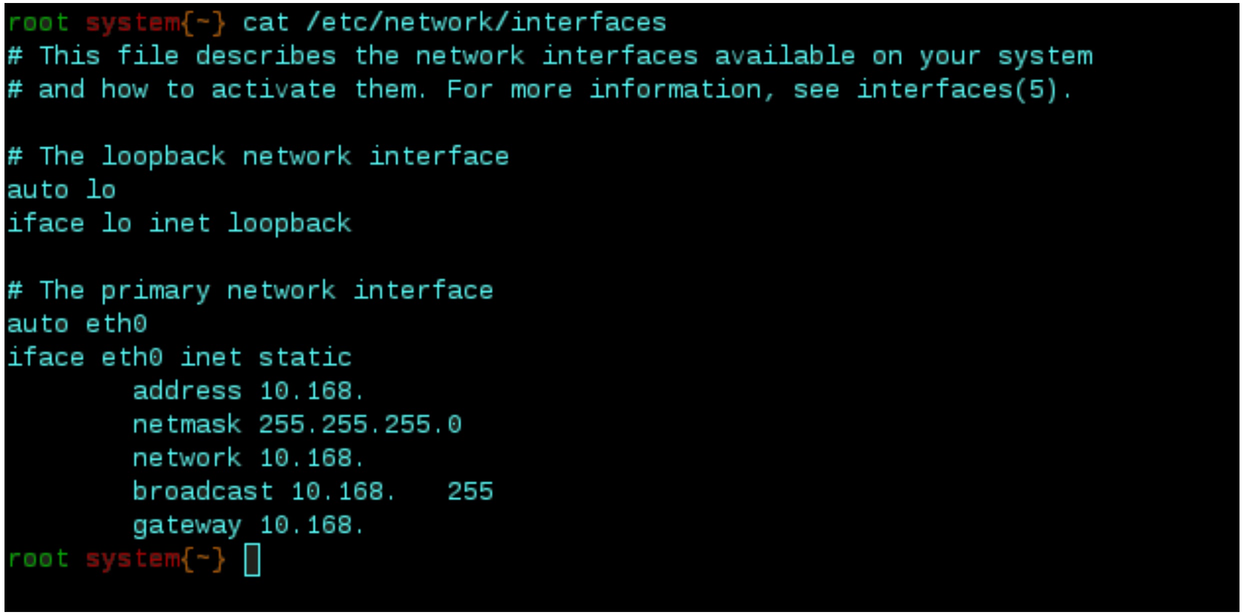


```
root system{~} ifconfig
eth0      Link encap:Ethernet  HWaddr 00:1a:4d:5b:
          inet addr:192.168.        Bcast:192.168.        Mask:255.255.255.0
          inet6 addr: fe80::21a:4dff:fe5b:b874/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:91196730 errors:0 dropped:0 overruns:0 frame:0
          TX packets:63486172 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2594468112 (2.4 GiB)  TX bytes:1555637946 (1.4 GiB)
          Interrupt:219 Base address:0x6000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:579230 errors:0 dropped:0 overruns:0 frame:0
          TX packets:579230 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:622563185 (593.7 MiB)  TX bytes:622563185 (593.7 MiB)

root system{~} 
```

The command \`ifconfig\` is used for listing active nic configurations:

Command: ifconfig
Parameters: n.a.
Output: see above

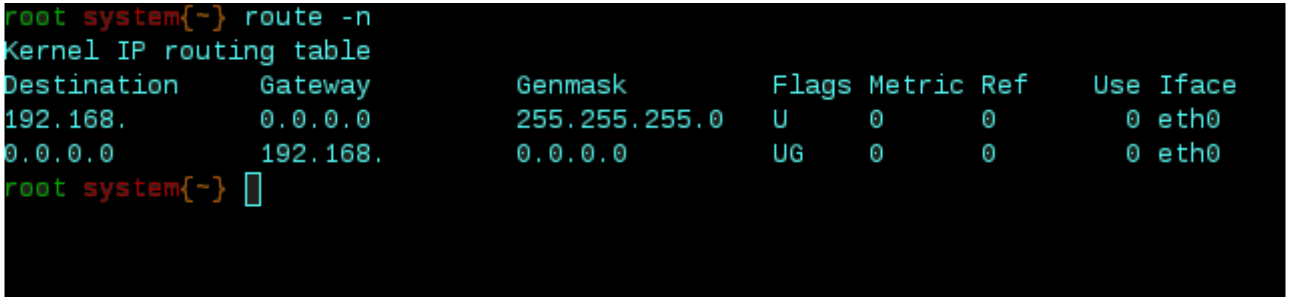
Network interface configuration



```
root system{~} cat /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
        address 10.168.
        netmask 255.255.255.0
        network 10.168.
        broadcast 10.168.    255
        gateway 10.168.
root system{~} 
```

The network configuration is stored in configuratin files on the systems. The file is on /etc/network/interfaces

List active routing configuration

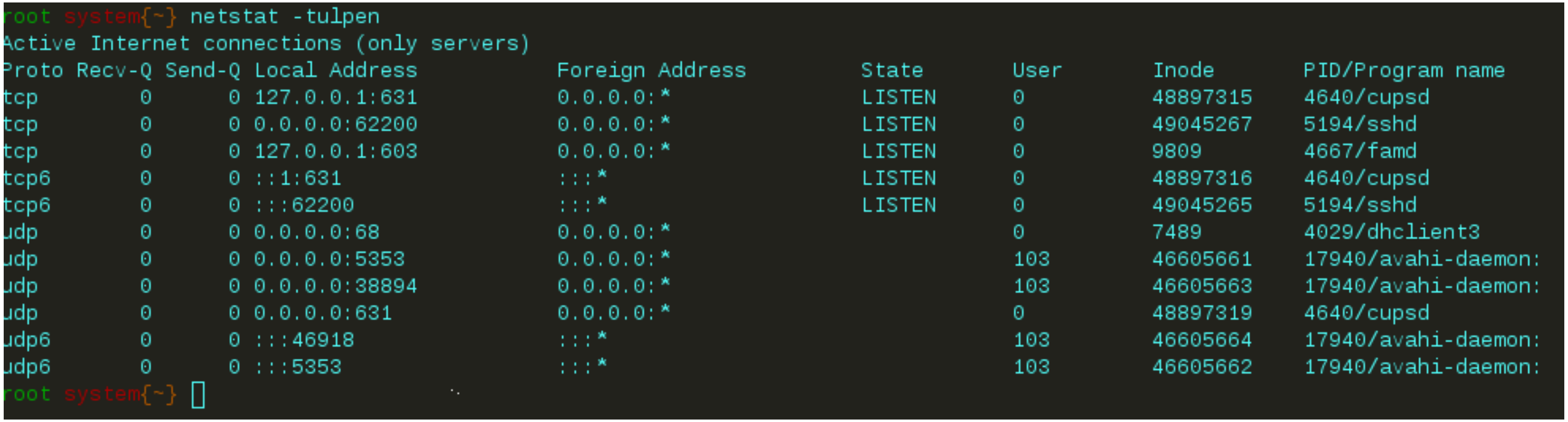


```
root system{~} route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.        0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.        0.0.0.0         UG    0      0        0 eth0
root system{~}
```

The command `route` is used for listing the active routes:

Command:	route
Parameters:	-n = do not resolve IP addresses
Output:	routing table

Show network statistics

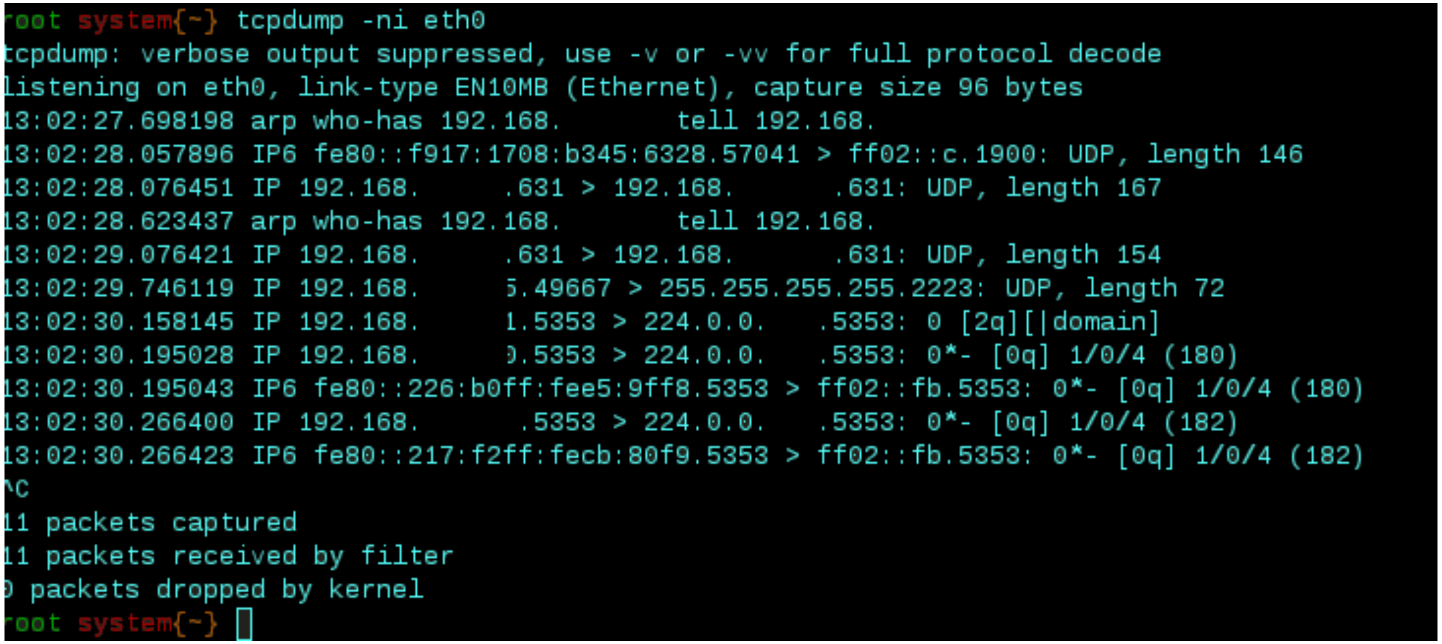


```
root system{~} netstat -tulpen
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User       Inode      PID/Program name
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      0          48897315   4640/cupsd
tcp        0      0 0.0.0.0:62200           0.0.0.0:*               LISTEN      0          49045267   5194/sshd
tcp        0      0 127.0.0.1:603           0.0.0.0:*               LISTEN      0          9809       4667/famd
tcp6       0      0 ::1:631                 :::*                    LISTEN      0          48897316   4640/cupsd
tcp6       0      0 :::62200                :::*                    LISTEN      0          49045265   5194/sshd
udp        0      0 0.0.0.0:68              0.0.0.0:*                           0          7489       4029/dhclient3
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           103        46605661   17940/avahi-daemon:
udp        0      0 0.0.0.0:38894           0.0.0.0:*                           103        46605663   17940/avahi-daemon:
udp        0      0 0.0.0.0:631             0.0.0.0:*                           0          48897319   4640/cupsd
udp6       0      0 :::46918                :::*                                103        46605664   17940/avahi-daemon:
udp6       0      0 :::5353                 :::*                                103        46605662   17940/avahi-daemon:
root system{~} 
```

The command `netstat` is used for listing network statistics:

Command: netstat
Parameters: -t = tcp-connection, -u = udp, -l = list, -p = program,
e= extended output, -n = do not resolve IP address
Output: Network statistics

Analyze network packets

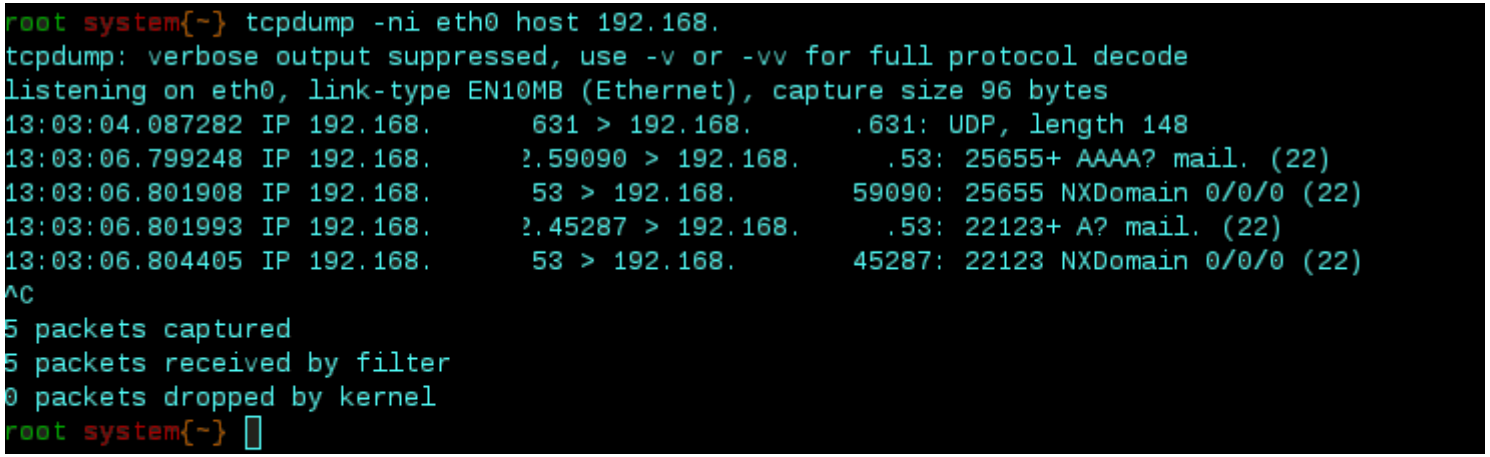


```
root system{~} tcpdump -ni eth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:02:27.698198 arp who-has 192.168.[redacted] tell 192.168.[redacted]
13:02:28.057896 IP6 fe80::f917:1708:b345:6328.57041 > ff02::c.1900: UDP, length 146
13:02:28.076451 IP 192.168.[redacted].631 > 192.168.[redacted].631: UDP, length 167
13:02:28.623437 arp who-has 192.168.[redacted] tell 192.168.[redacted]
13:02:29.076421 IP 192.168.[redacted].631 > 192.168.[redacted].631: UDP, length 154
13:02:29.746119 IP 192.168.[redacted]5.49667 > 255.255.255.255.2223: UDP, length 72
13:02:30.158145 IP 192.168.[redacted]1.5353 > 224.0.0.[redacted].5353: 0 [2q][|domain]
13:02:30.195028 IP 192.168.[redacted]9.5353 > 224.0.0.[redacted].5353: 0*- [0q] 1/0/4 (180)
13:02:30.195043 IP6 fe80::226:b0ff:fee5:9ff8.5353 > ff02::fb.5353: 0*- [0q] 1/0/4 (180)
13:02:30.266400 IP 192.168.[redacted].5353 > 224.0.0.[redacted].5353: 0*- [0q] 1/0/4 (182)
13:02:30.266423 IP6 fe80::217:f2ff:fecb:80f9.5353 > ff02::fb.5353: 0*- [0q] 1/0/4 (182)
^C
11 packets captured
11 packets received by filter
0 packets dropped by kernel
root system{~}
```

The command `tcpdump` is used to analyze network packets:

Command: tcpdump
Parameters: -n= do not resolve IP address, -i = interface name to dump
Output: see above

Analyze contents of packets on a network



```
root system{~} tcpdump -ni eth0 host 192.168.[redacted]
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:03:04.087282 IP 192.168.[redacted] 631 > 192.168.[redacted].631: UDP, length 148
13:03:06.799248 IP 192.168.[redacted]2.59090 > 192.168.[redacted].53: 25655+ AAAA? mail. (22)
13:03:06.801908 IP 192.168.[redacted] 53 > 192.168.[redacted]59090: 25655 NXDomain 0/0/0 (22)
13:03:06.801993 IP 192.168.[redacted]2.45287 > 192.168.[redacted].53: 22123+ A? mail. (22)
13:03:06.804405 IP 192.168.[redacted] 53 > 192.168.[redacted]45287: 22123 NXDomain 0/0/0 (22)
^C
5 packets captured
5 packets received by filter
0 packets dropped by kernel
root system{~} █
```

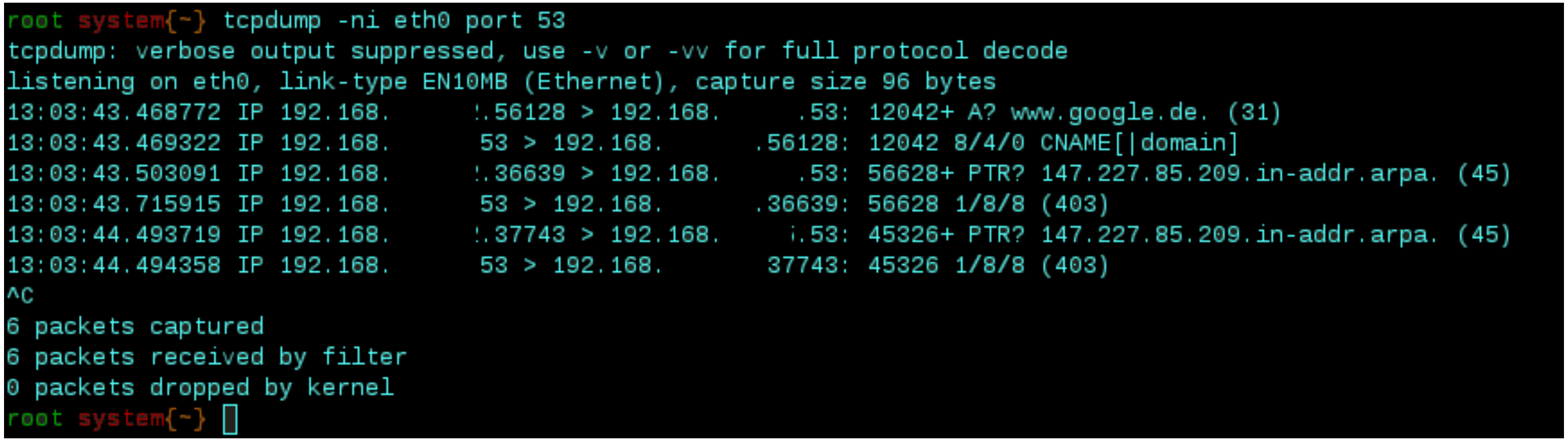
The command `tcpdump` is used to analyze network packets:

Command: tcpdump

Parameters: -n= do not resolve IP address, -i = interface name to dump,
host = hostaddress to filter on

Output: see above

Analyze contents of packets on a network

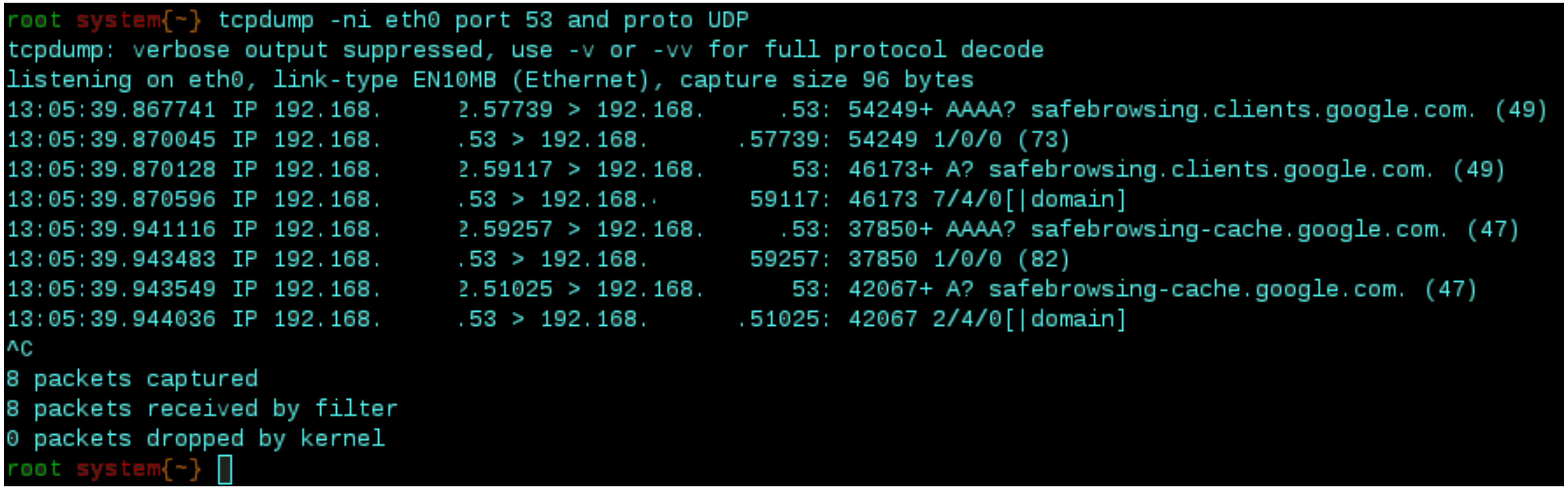


```
root system{~} tcpdump -ni eth0 port 53
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:03:43.468772 IP 192.168.[redacted].56128 > 192.168.[redacted].53: 12042+ A? www.google.de. (31)
13:03:43.469322 IP 192.168.[redacted] 53 > 192.168.[redacted].56128: 12042 8/4/0 CNAME[|domain]
13:03:43.503091 IP 192.168.[redacted].36639 > 192.168.[redacted].53: 56628+ PTR? 147.227.85.209.in-addr.arpa. (45)
13:03:43.715915 IP 192.168.[redacted] 53 > 192.168.[redacted].36639: 56628 1/8/8 (403)
13:03:44.493719 IP 192.168.[redacted].37743 > 192.168.[redacted].53: 45326+ PTR? 147.227.85.209.in-addr.arpa. (45)
13:03:44.494358 IP 192.168.[redacted] 53 > 192.168.[redacted] 37743: 45326 1/8/8 (403)
^C
6 packets captured
6 packets received by filter
0 packets dropped by kernel
root system{~} ▯
```

The command `tcpdump` is used to analyze network packets:

Command: tcpdump
Parameters: -n= do not resolve IP address, -i = interface name to dump,
port = port to filter on
Output: see above

Analyze contents of packets on a network



```
root system{~} tcpdump -ni eth0 port 53 and proto UDP
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
13:05:39.867741 IP 192.168.     2.57739 > 192.168.     .53: 54249+ AAAA? safebrowsing.clients.google.com. (49)
13:05:39.870045 IP 192.168.     .53 > 192.168.      .57739: 54249 1/0/0 (73)
13:05:39.870128 IP 192.168.     2.59117 > 192.168.      53: 46173+ A? safebrowsing.clients.google.com. (49)
13:05:39.870596 IP 192.168.     .53 > 192.168.      59117: 46173 7/4/0[|domain]
13:05:39.941116 IP 192.168.     2.59257 > 192.168.     .53: 37850+ AAAA? safebrowsing-cache.google.com. (47)
13:05:39.943483 IP 192.168.     .53 > 192.168.      59257: 37850 1/0/0 (82)
13:05:39.943549 IP 192.168.     2.51025 > 192.168.      53: 42067+ A? safebrowsing-cache.google.com. (47)
13:05:39.944036 IP 192.168.     .53 > 192.168.      .51025: 42067 2/4/0[|domain]
^C
8 packets captured
8 packets received by filter
0 packets dropped by kernel
root system{~}
```

The command `tcpdump` is used to analyze network packets:

Command: tcpdump –ni eth0 port 53 and proto UDP
Parameters: -n= do not resolve IP address, -i = interface name to dump, port = Port to filter on, proto = Protocol to filter on,
Output: see above

Daemon Tools Usage

Daemon Tools is used for starting / stopping the iProxy services

a Daemon Tools File structure is needed:

```
/home/iproxy/service/admf
                     /data/
                     /etc/instance.conf
                     /service
                             /log/
                             /run
                             /supervise/
```

→ To activate the service admf, the /home/iproxy/service/admf/service directory has to be linked in to the /etc/service folder

Daemon Tools Usage

Daemon Tools is used for starting / stopping the iproxy services

Once the service is linked and activated it constantly restarts itself when having problems

The activated service can be controlled via the “svc” command:

- svc -t /etc/service/admf: sends a TERM Signal, and automatically restarts the daemon after it dies
- svc -d /etc/service/admf: sends a TERM Signal, and leaves the service down
- svc -u /etc/service/admf: brings the service back up
- svc -o /etc/service/admf: runs the service once

Hands on experience on demand

What would you like to explore in greater detail ?

- Collecting network traces
- Collecting logs
- Collecting evidence
- More system training
- Tell us

Incident handling

Basically the systems just work. In case something does not work or you are not sure:

1) Collect data, evidences, log files

2) Contact our helpdesk

3) More details (including contact) in the system manual

4) We fix things together

Questions ?

Thank you for your attention !

WWW.GAMMAGROUP.COM

FINFISHER

IT INTRUSION