

# **The Unofficial VoIP Debug Cheat Sheet**

## **Leonid Consulting**

Leonid Consulting

July 2007

V1.2

## Contents

---

Contents.....	2
Introduction .....	2
Debugging Tools on the OSI Stack .....	4
Debugging Techniques by Platform.....	5
At the Customer Premise (General).....	5
XTen.....	5
Adtran IAD .....	6
Acme SBC .....	6
BroadWorks.....	8
Cisco IOS.....	8

## Introduction

---

### Scope

The purpose of this document is to provide a quick reference for end to end VoIP system troubleshooting, with a focus on call processing/SIP.

These recommendations are provided as is at the current time. Third parties may change their implementations and you should have the proper training before using any piece of equipment. Leonid offers no warranty or other guarantee on these techniques.

### The Process of Troubleshooting

Great troubleshooters are made not born! Four generic steps to be sure you should be able to cover as a troubleshooter:

1. Know what you expect to happen

If you don't have a strong fundamental understanding of the relevant protocols, you will always struggle to be a consistently successful troubleshooter. You may have a few "formula fixes" that work a lot of the time, but when the tricky issues come up you'll struggle to progress to steps 2, 3. Set aside time to learn the protocols; you'll almost definitely find it was time well spent. One of the best ways to do this once you have the basics down is to go back and dig in on any issue that stumped you. You won't have time to do this in the heat of the moment, but once you have things up and running, find a way to go and learn more about whatever stumped you.

2. Be able to form a hypothesis about what may be wrong

A substantial part of this comes with experience. But at a minimum know how to work your way up the stack (see below). The important part is to be able to be very specific about what is not working vs. what you expect to have happen and be able to put forward possible reasons for it.

3. Be able to test that hypothesis to isolate the issue

Know the tracing and debugging facilities on your equipment. If you don't know how to go in and look at traffic at the relevant levels in the stack for all your elements, your network is at risk. Talk to your colleagues, talk to your vendors, and make sure you know how to isolate signaling on every element for debugging purposes. Beyond that, with experience you'll learn shortcuts and tricks to abbreviate the debugging process. Always be on the lookout for this since debugging since efficiency is a key aspect of any troubleshooting process.

4. Remember that you're always a student

The moment that you decide it's more important to look like you know something than to really know it, your ability to improve as a troubleshooter plummets. There's always someone that knows more than you do about a topic, and there are always new things to learn. Aggressively seek out information about new topics and watch how efficient troubleshooters do things.

## Debugging Tools on the OSI Stack

---

The table below provides a quick review of common issues and debugging techniques on the OSI stack. The notation with the brackets “[ ]” means that an argument/parameter needs to be entered with the command.

**Table 1 Debugging Techniques by OSI Layer**

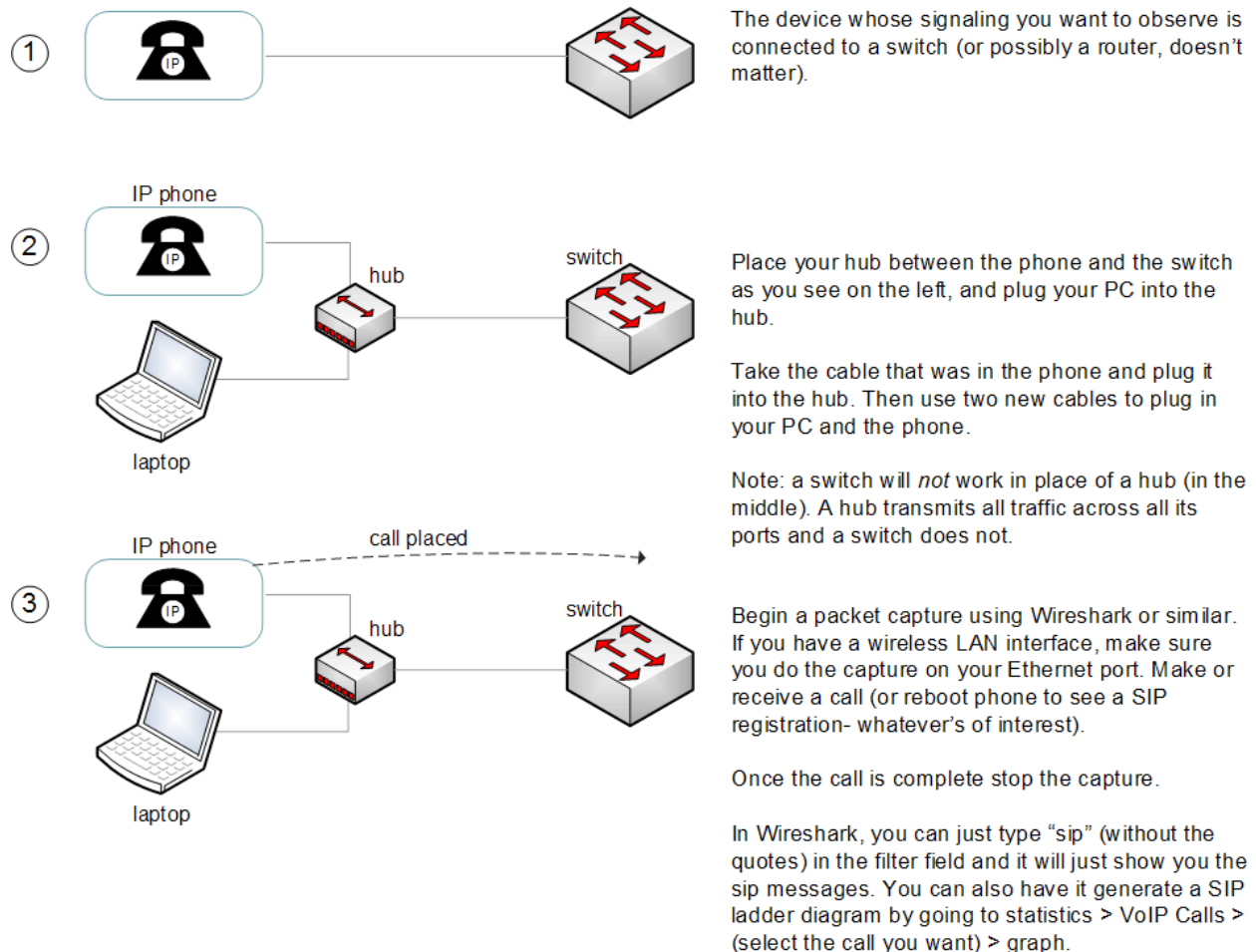
Layer	Common Issue	Debugging Tools
Application Layer	here the space of issues widen based on what the particular application does and how it's implemented	SIP: see section on logs below  DNS: nslookup  Wireshark is a good general purpose tool for analysing packet captures.
Presentation Layer	(not generally an issue in VoIP-related communications)	
Session Layer	(not generally an issue in VoIP-related communications)	
Transport Layer (TCP, UDP)	firewalls blocking services/ports between endpoints	TCP: telnet [host] [port] (opens connection to target host on specified port)  UDP: the best way is to snoop the traffic between the hosts on Solaris use the snoop command and on Linux use the tcpdump
Network Layer (IP)	routes not provisioned	ping [destination IP]
Link Layer (Ethernet, etc.)	duplex mismatches, interfaces not configured/properly	Unix: ifconfig -a (general state of interfaces) ndd -get /dev/bge0 link_speed (to see duplex settings)  Linux: mii-tool will give you what you need for most cases
Physical Layer (basic electrical signalling- wavelengths, etc.)	cable unplugged, network interface down or partially down	indicator lights on physical ports, vendor-specific utilities

## Debugging Techniques by Platform

### At the Customer Premise (General)

Let's say you're at the customer premise and you don't have access to your core system to observe logs. You may not even want it: you can get a view on what's happening with just a hub and a PC.

If you don't have ready access to SIP logs on the customer, you can snoop the traffic of interest with a simple hub and a PC. See below:



### XTen

The older versions of the XTen Lite soft client are very useful for quick UA setup and debugging. Version X\_lite-Xten-Win32-1103m-14262 will work and a recent search on Google (7/07) showed several mirror sites offering the file. The Lite version is available free from XTen (there's an enhanced version available for sale).

On the XTen client click the settings button

Then go to: System Settings > SIP Proxy > [profile] to set up the user account.

Now you're ready to make a call, but first see if the phone has come up and registered properly by going to: Settings > Advanced System Settings > Diagnostic > Diagnostic Log. This is a real time log of the UA's activity and a good way to quickly check if an account is working properly.



## Adtran IAD

The Adtran IAD (900 series) offers excellent logging and monitoring capabilities.

To view current registration:

```
Mystery_908#show sip trunk-registration T01
```

Trk	Identity	Reg'd	Grant	Expires	Success	Failed	Requests	Challenges	Rollovers
T01	4158782250	Yes	3599	589	2058	165	2307	127	39
T01	4158782251	Yes	3599	589	2048	192	2396	199	39
T01	4158782252	Yes	3599	589	2048	189	2396	202	39
T01	4158782253	Yes	3599	589	2046	190	2398	205	39
T01	4158782929	Yes	3599	529	2056	163	2321	145	39

This assumes your sip trunk is called T01. If it's not just look at the configuration ('show run').

To view the SIP signaling in a terminal session, issue the command 'debug sip stack messages'. When you're done issue the command 'undebug all' to turn it off (to turn off all logging).

If you're having an issue with registration and you want to have all the lines attempt registration issue the command 'Mystery\_908#sip trunk-registration force-register'.

See AdTran 900 series manual (available on CD) for more info.

## Acme SBC

The Acme SBC logs can be tailed or FTP'ed off the SD.

The basic procedure for using the tail facility on the Acme CLI (described in more detail in Acme's Maintenance Guide) is:

1. enable logging:

```
notify sipd siplog
```

```
log-level sipd [info, debug]
```

- use debug level with extreme caution; even on a lightly loaded system it can push CPU into the danger zone

2. check CPU to make sure you're OK

```
show processes CPU
```

3. open tail

for example, to tail the sip messages:

```
tail-logfile-open sipd sipmsg.log
```

note: you'll probably want to log your session somewhere using the relevant facilities in your SSH or telnet client since in a production environment you'll quickly roll through your client's buffer

4. close tail

for example, to close tail the sip messages:

```
tail-logfile-close sipd sipmsg.log
```

5. disable logging:

```
notify sipd nosiplog
```

```
log-level sipd notice
```

The basic procedure for retrieving via FTP (described in more detail in Acme's Maintenance Guide) is:

1. enable logging:

```
notify sipd siplog
```

```
log-level sipd [info, debug]
```

- use debug level with extreme caution; even on a lightly loaded system it can push CPU into the danger zone

2. check CPU to make sure you're OK

```
show processes CPU
```

3. disable logging:

```
notify sipd nosiplog
```

```
log-level sipd notice
```

4. retrieve logs

```
ftp to SD
```

```
cd /ramdrv/logs
```

look at timestamps on rolling on sipmsg.log and sipd.log- pull off the last few that were created around the current time

you'll need to search through them sequentially for the target phone numbers to find your messages

See the Acme "Maintenance and Troubleshooting" guide for more info.

## BroadWorks

BroadWorks runs on commercial OS's Solaris and RedHat Linux and you can view the logs in real time just by using the tail -f command.

### 1. get into the logs

```
cd /var/broadworks/logs/[appserver, routingserver]
```

(applicationserver for the AS; routingserver for the NS)

```
ls -ltr
```

### 2. tail the logs

tail -f on latest version of the XS... log if you want to look at SIP/CallP or PS.... logs if you want to look at the database processes

See the BroadWorks "Maintenance Guide" for more information.

## Cisco IOS

### 1. enable logging

```
debug voice ccapi inout
```

- general callP

```
debug ccsip messages
```

- SIP messages

```
debug isdn q931
```

- ISDN messages

### 2. watch CPU to make sure you are OK

```
show processes CPU
```

### 3. monitor via terminal

```
term mon
```

```
no term mon (to turn off)
```

- note: if there's a lot of traffic it may be preferable to log the output to a buffer

### 4. log to buffer (optional)

```
mystery_GW#
```

```
mystery_GW#conf t
```

Enter configuration commands, one per line. End with CNTL/Z.

```
mystery_GW(config)#logging buff 100000 deb
```

```
mystery_GW(config)#no logging console
```

```
mystery_GW#clear log
```

### 5. view buffer contents

```
ter len 0  
sh log
```

See the Cisco Command Reference for more detail.