Open Source VoIP Traffic Monitoring

Luca Deri <deri@ntop.org>

Why VoIP is a Hot Topic ?

- Thanks to open source projects (e.g. Asterisk, Gizmo), and custom Linux distributions (e.g. Asterisk@Home) setting up a VoIP server is becoming simpler.
- Many modern DSL routers (e.g. Linksys, Fritz!Box) now feature VoIP support via a telephony plug.
- Proprietaries VoIP systems like Skype, GoogleTalk or VoIPStunt! made VoIP very simple allowing virtually every PC-user to take advantage of VoIP.
- VoIP is currently integrated into many applications (e.g. Office 12) or online assistance/support (e.g. ether.com, estara.com)

Motivation for This Work

- Working groups (e.g http://voip.internet2.edu/, Terena TF-VVC) are mainly focusing on infrastructure.
- Traffic sniffers (e.g. ethereal) are suitable for analyzing specific calls and not for permanent VoIP traffic monitoring.
- VoIP servers (e.g. Asterisk) do not offer calls monitoring but just call info (CDR, call data record).
- Commercial VoIP traffic analyzers (e.g. Telchemy VQmon) are very expensive and are not easy to integrate with other tools.
- No specific VoIP open source traffic analyzer tools available to date.

Project Goals

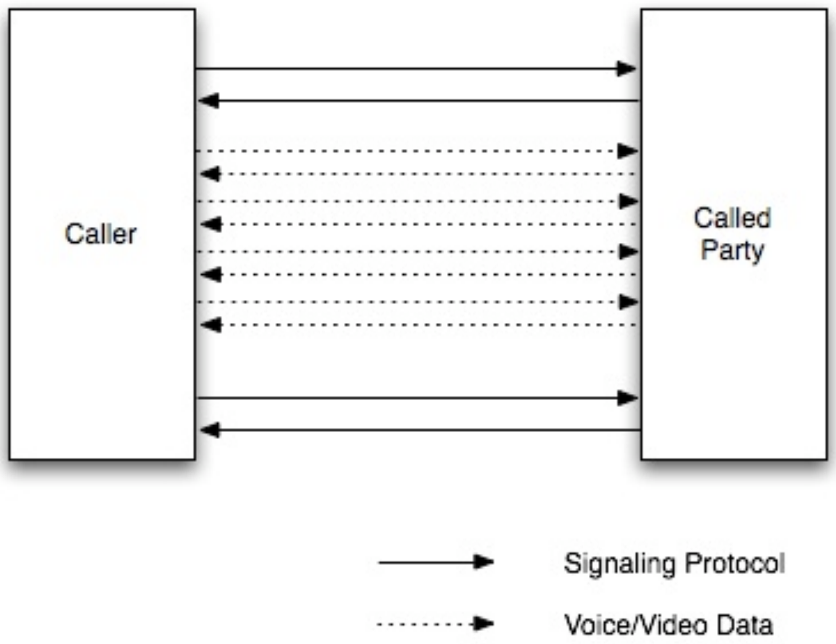
- Provide long-term monitoring, contrary to what available VoIP monitoring tools do.
- Handling standard VoIP protocols as well, as much as possible, proprietary protocols.
- Decode calls, hence identify peers (who's calling who) and client applications (useful for VoIP accounting, billing or fraud detection).
- Provide VoIP metrics such as packet loss and latency, as well as voice quality.
- Generate traffic trends in order to identify how VoIP traffic is changing over the time (e.g. VoIP client/provider usage statistics).

Approach Being Used

- Enrich ntop, a home-grown open-source passive traffic monitoring application, for making it VoIP traffic aware.
- Define some metrics suitable for monitoring key VoIP traffic characteristics.
- Export VoIP measurements via Netflow v9/IPFIX, by means of nProbe (home-grown GPL NetFlow probe).
- Motivation
  - ntop users can also monitor VoIP without having to use any specialized VoIP traffic analysis application (VoIP is not a first class citizen).
  - The use of NetFlow/IPFIX allows VoIP measurements to be made available to any netflow aware application (open design).

VoIP Basics

- Signaling
  - User location
  - Session
    - Setup
    - Negotiation
    - Modification
    - Closing
- Transport
  - Encoding, transport, etc.



Standard VoIP Protocols

- SIP
  - IETF - 5060/5061 (TLS) - “HTTP-like, all in one”
  - Proprietary extensions
  - Protocol becoming an architecture
- H.323
  - Protocol family
  - ASN.1 based
  - H.235 (security), Q.931+H.245 (management), CODECs etc.
- RTP (Real Time Protocol)
  - 5004/udp, RTCP: used to transport voice and video
  - No QoS/bandwidth management
  - Data is encoded using codecs

Proprietary VoIP Protocols

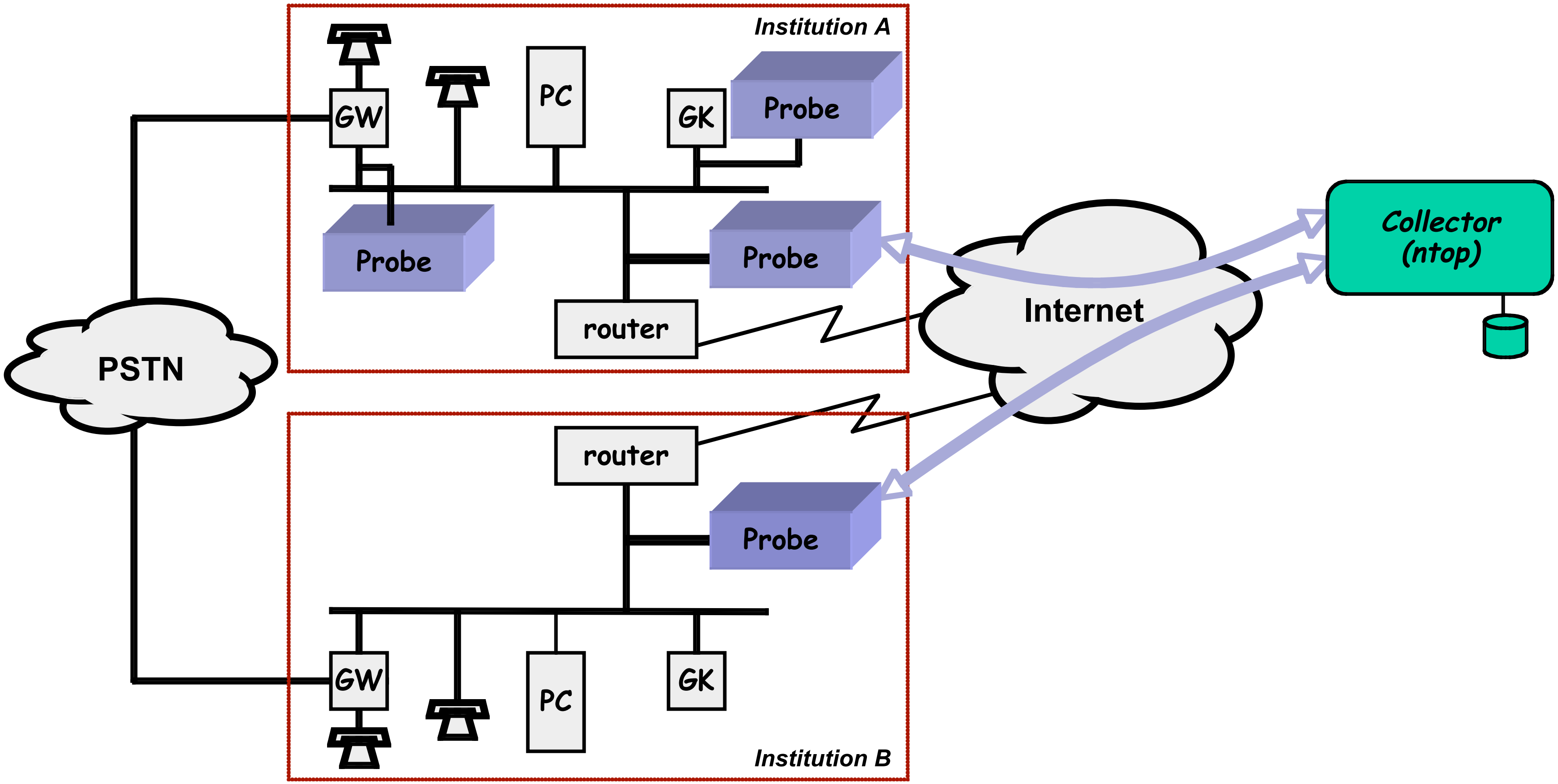
- Cisco Skinny
  - Signaling protocol, easy to decode and handle.

- Asterix
  - IXP (Inter Asterisk eXchange) Protocol: open protocol.

- Skype
  - Decentralized architecture (P2P).
  - Ability to call both users and plain phones.
  - Phone calls are both encrypted and obfuscated.
  - Totally closed source development model.

- What to do then?
  - Fully support standard VoIP protocols.
  - Support proprietary protocols as much as possible.

Flow-based Monitoring Architecture



Standard VoIP: Implemented Metrics [1/2]

- SIP
  - Unique call identifier used for accounting/billing and tracking problems as well as correlating protocol messages.
  - Call parties: caller and called party.
  - Codecs being used (useful for identifying voice quality issues due to the use of codecs with poor quality).
  - Time of important call events such as beginning of the call (e.g. used to identify performance issues on the SIP gateway).
  - RTP ports where the call will take place (used for associating a signaling flow with the phone call just negotiated).

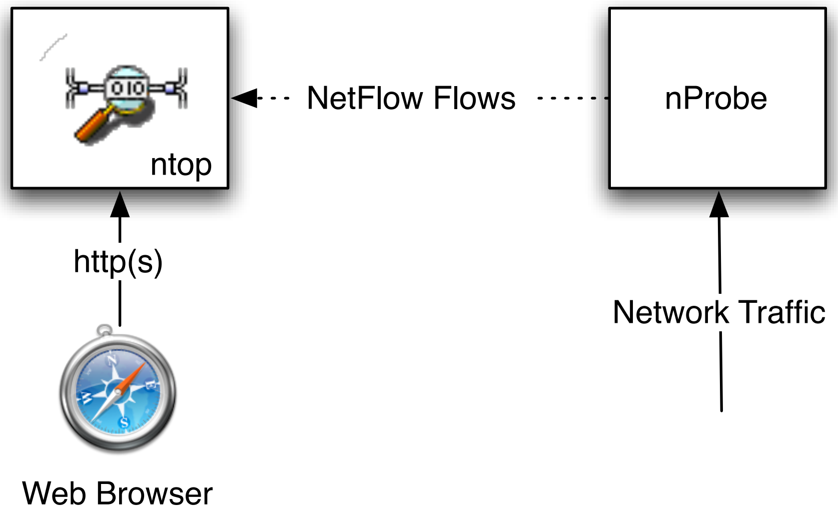
- RTP
  - Source identifiers and time-stamp for the first and last RTP flow packet.
  - Jitter calculated in both (in to out, and out to in) directions.
  - Number of packets lost as well as maximum packet time delta in both directions.
  - Identifier of RTP payload type as specified in [rfc2862].

Standard VoIP: Implemented Metrics [2/2]

SIP Metrics	RTP Metrics
SIP_CALL_ID	RTP_FIRST_SSRC
SIP_CALLING_PARTY	RTP_FIRST_TS
SIP_CALLED_PARTY	RTP_LAST_SSRC
SIP_RTP_CODECS	RTP_LAST_TS
SIP_INVITE_TIME	RTP_IN_JITTER
SIP_TRYING_TIME	RTP_OUT_JITTER
SIP_RINGING_TIME	RTP_IN_PKT_LOST
SIP_OK_TIME	RTP_OUT_PKT_LOST
SIP_ACK_TIME	RTP_OUT_PAYLOAD_TYPE
SIP_RTP_SRC_PORT	RTP_IN_MAX_DELTA
SIP_RTP_DST_PORT	RTP_OUT_MAX_DELTA

Note: no H.323 support (obsoleted by SIP).

NetFlow VoIP Architecture




```
nprobe -n 192.168.0.1:2055 -U 257 -T "%LAST_SWITCHED
%FIRST_SWITCHED %IN_BYTES %IN_PKTS %OUT_BYTES %OUT_PKTS
%SIP_CALL_ID%SIP_CALLING_PARTY %SIP_CALLED_PARTY
%SIP_RTP_CODECS %SIP_RTP_SRC_PORT %SIP_RTP_DST_PORT"
```

ntop VoIP Support: SIP/RTP

83.175.52.136 :49650 <VoIP>	83.175.54.75 :25000	27.0 KB	055487214 called 055470167
83.175.52.136 :49652 <VoIP>	83.175.54.75 :quake	39.5 KB	055487214 called 055470167

Host Type	VoIP Host
Known Users	055470167 [VoIP]

ntop VoIP Support: Skype [1/2]

Host	Domain	IP Address
PowerBook G4 Luca		10.96.6.166

Host Type	VoIP Host HTTP Server
Known Users	yuri's music [DAAP] luca deri's music [DAAP]
Host Healthness (Risk Flags)	1. Unexpected packets (e.g. traffic to closed port or connection reset): 2. Unexpected packets (e.g. traffic to closed port or connection reset): [Rcvd: rst] [Sent: closed-empty] [Rcvd: hostnet unreac]

ntop VoIP Support: Skype [2/2]

Client	Server	Data Sent	Data Rcvd	Duration	Inactive	L7 Proto
PowerBook G4 Luca :54045/skype.gif	modemcable223.209-131-66.mc.videotron.ca :22598	59	387	0 sec	8:27	skypetoskype
PowerBook G4 Luca :54045/skype.gif	bzq-88-153-37-147.red.bezeqint.net :accelenet	46	54	0 sec	6:01	skypetoskype
PowerBook G4 Luca :54045/skype.gif	cpe001111861b9e-cm00e06f179444.cpe.net.cable.rogers.com :tvbus	498	470	12 sec	8:15	skypetoskype
PowerBook G4 Luca :54045/skype.gif	warbler.csail.mit.edu :bridgecontrol	412	39	0 sec	8:15	skypetoskype
PowerBook G4 Luca :54045/skype.gif	c-69-138-253-151.hsd1.md.comcast.net :14285	59	46	0 sec	8:27	skypetoskype
PowerBook G4 Luca :54045/skype.gif	user-12lm8mn.cable.mindspring.com :4793	46	54	0 sec	6:01	skypetoskype
PowerBook G4 Luca :54045/skype.gif	f-ray-xps.econ.nyu.edu :54004	270	78	0 sec	6:00	skypetoskype

- Protocol Patterns: http://l7-filter.sourceforge.net
- Pattern Engine: http://www.pcre.org/

Open Issues and Future Work

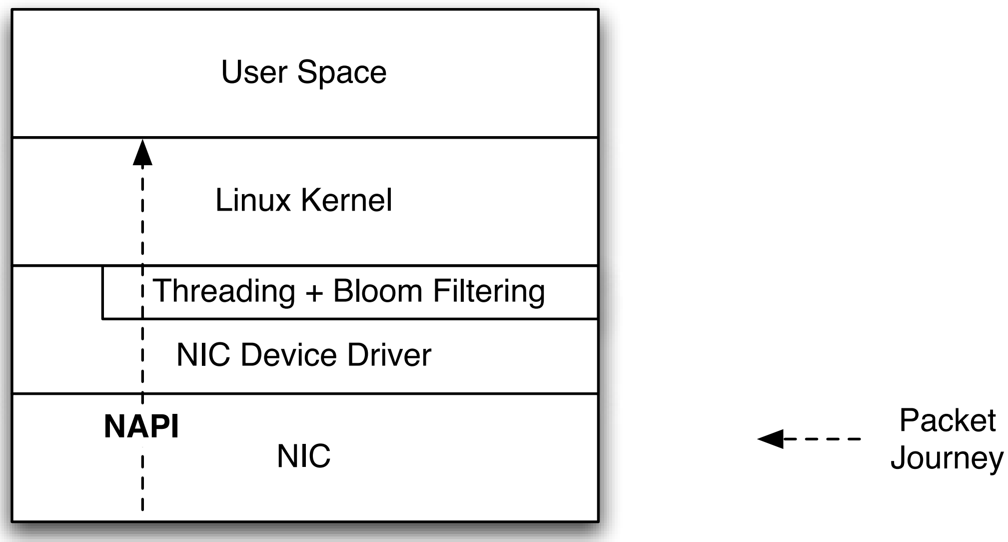
- Skype support is poor (general problem with proprietary protocols).
- Implement payload analysis (e.g. voice quality).
- Handle RTCP XS (RTP Control Extended) reports sent by telephony equipment (it contains calls information).
- Implement new metrics such as MOS (Mean Opinion Score) and R-Factor, used to score traffic calls quality. The drawback is that most information (e.g. ITU E.411 recommendation) is proprietary and not freely available in the internet.
- NetFlow/IPFIX are standard protocols but not suitable to carry CRD (Call Data Records) at least in terms of latency. Solution: threat CDRs as high-priority flows.

Challenges in VoIP Packet Capture

- VoIP traffic is usually very little compared to the rest of traffic.
- Capture starts from filtering signaling protocols and then intercepting voice payload.
- BPF-like filtering is not effective (one filter only)
- It is necessary to add/remove filters on the fly as calls start/end.
- We need to have hundred of active filters (a few per call).

- Solution
  - Filter packets directory on the device driver (not into the kernel layer).
  - Implement hash/bloom based filtering (limited false positives).
  - Memory effective (doesn't grow as filters are added).
  - Currently implemented on Linux on Intel GE cards.
  - Great performance (virtually no packet loss at 1 GBit): better than nCap/PF_RING

Dynamic Packet Filtering [1/2]




```
echo "+ip=192.168.0.10,port=80" > /proc/net/eth1/rules
echo "-proto=tcp" > /proc/net/eth1/rules
```

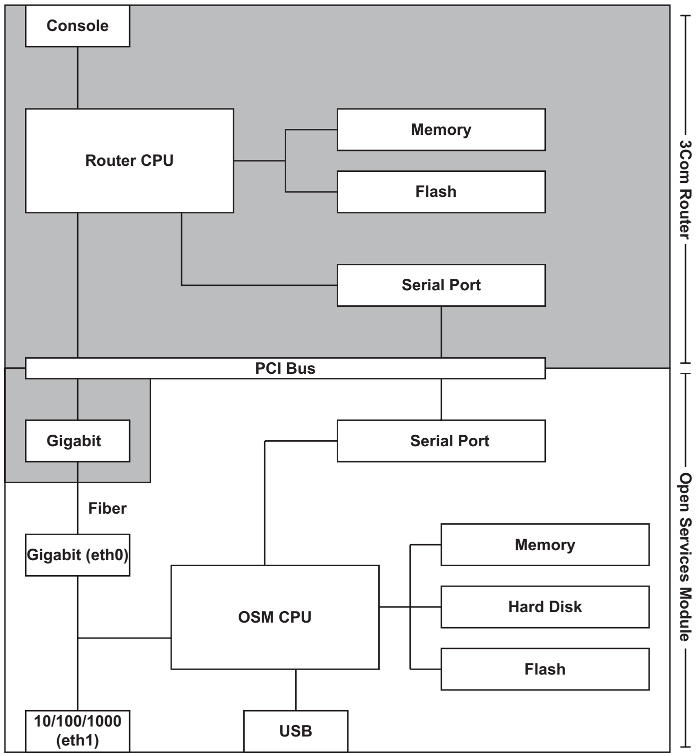
Dynamic Packet Filtering [2/2]

ID	Intel Driver	Bloom Filters	Packet Size	Total Input Rate (both adapters)	System Load	Packet Loss
1	Threaded	No filters	900 bytes	270 Kpps	1.14	No
3	Threaded	One IP match	900 bytes	270 Kpps	1.66	No
4	Threaded	No filters	Random 64-1518	890 Kpps	1.34	No
5	Vanilla	No filters	Random 64-1518	890 Kpps	2.45	Moderate (< 10%)
6	Threaded	One IP match	Random 64-1518	890 Kpps	1.40	No
7	Threaded	No filters	64 bytes	2.89 Mpps	3.68	Moderate (< 20%; interface counters do not keep up with traffic)
8	Threaded	One IP match	64 bytes	2.89 Mpps	> 4	Strong (> 20%)

Commercial Capture Cards

- Generic Capture Cards (Endace and Napatech/Xyratex)
  - Advantages
    - Wire speed packet capture (up to 10 Gbit).
    - Ability to filter traffic at wire speed.
  - Limitations
    - The number of filters that can be specified is too small to be usable with VoIP (e.g. Endace allows FPGA-based 7 filters to be specified in the latest 10 Gbit card).
    - Filter reconfiguration leads to packet loss as the card is blocked during reconfiguration (e.g Napatech/Xyratex card can take a few seconds or more).
    - Filters are simple packet-offset "data & mask" (i.e. no packet parsing or BPF-like facilities)
- Specialized Packet Capture Cards (Mutech)
  - Ability to operate as an accelerated packet capture card.
  - Advanced packet filtering facilities suitable for VoIP traffic analysis.
  - On-board FPGA-based voice quality analysis.

3Com OSM Router/Switch



- Ability to specify up to 1024 ASIC-based complex filters (L2 and port range support).
- Ability to mirror (passive) and redirect (passive+active) traffic.
- Use of dynamic bloom filtering on top of ASIC traffic filtering.
- OSM traffic analysis via nProbe.

Availability

- Paper and Documentation:
  - http://luca.ntop.org/VoIP.pdf
  - http://luca.ntop.org/Blooms.pdf
- Code and Applications
  - http://www.ntop.org/