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.
• Proprietary 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).

ntop.org
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.
• 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

ntop.org
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

Institution A

GW
PC
GK
Probe
router

Institution B

GW
PC
GK
Probe
router

PSTN

Internet

Collector (ntop)
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].
<table>
<thead>
<tr>
<th>SIP Metrics</th>
<th>RTP Metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIP_CALL_ID</td>
<td>RTP_FIRST_SSRC</td>
</tr>
<tr>
<td>SIP_CALLING_PARTY</td>
<td>RTP_FIRST_TS</td>
</tr>
<tr>
<td>SIP_CALLED_PARTY</td>
<td>RTP_LAST_SSRC</td>
</tr>
<tr>
<td>SIP_RTP_CODECS</td>
<td>RTP_LAST_TS</td>
</tr>
<tr>
<td>SIP_INVITE_TIME</td>
<td>RTP_IN_JITTER</td>
</tr>
<tr>
<td>SIP_TRYING_TIME</td>
<td>RTP_OUT_JITTER</td>
</tr>
<tr>
<td>SIP_RINGING_TIME</td>
<td>RTP_IN_PKT_LOST</td>
</tr>
<tr>
<td>SIP_OK_TIME</td>
<td>RTP_OUT_PKT_LOST</td>
</tr>
<tr>
<td>SIP_ACK_TIME</td>
<td>RTP_OUT_PAYLOAD_TYPE</td>
</tr>
<tr>
<td>SIP_RTP_SRC_PORT</td>
<td>RTP_IN_MAX_DELTA</td>
</tr>
<tr>
<td>SIP_RTP_DST_PORT</td>
<td>RTP_OUT_MAX_DELTA</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Port (VolP)</th>
<th>Port (Called)</th>
<th>Bandwidth</th>
<th>Caller ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>83.175.52.136</td>
<td>49650</td>
<td></td>
<td></td>
<td>055487214 called 055470167</td>
</tr>
<tr>
<td>83.175.54.75</td>
<td>25000</td>
<td></td>
<td>27.0 KB</td>
<td></td>
</tr>
<tr>
<td>83.175.52.136</td>
<td>49652</td>
<td>83.175.54.75</td>
<td>39.5 KB</td>
<td>055487214 called 055470167</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Host Type</th>
<th>VolP Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known Users</td>
<td>055470167 [ VolP ]</td>
</tr>
</tbody>
</table>
### ntop VoIP Support: Skype [1/2]

<table>
<thead>
<tr>
<th>Host</th>
<th>Domain</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerBook G4 Luca</td>
<td></td>
<td>10.96.6.166</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Host Type</th>
<th>VoIP Host</th>
<th>HTTP Server</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Known Users</th>
<th>yuri's music [DAAP]</th>
<th>luca der's music [DAAP]</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host Healthiness (Risk Flags)</th>
<th>1.</th>
<th>2.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unexpected packets (e.g. traffic to closed port or connection reset):</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[Rcvd: rst] [Sent: closed-empty] [Rcvd: hostnet unreac]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### ntop VoIP Support: Skype [2/2]

#### Protocol Patterns:
http://l7-filter.sourceforge.net

#### Pattern Engine:
http://www.pcre.org/

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
<th>Data Sent</th>
<th>Data Rcvd</th>
<th>Duration</th>
<th>Inactive</th>
<th>L7 Proto</th>
</tr>
</thead>
<tbody>
<tr>
<td>PowerBook G4 Luca</td>
<td>modern cable 223.209-131-66.mc.videotron.ca</td>
<td>59</td>
<td>387</td>
<td>0 sec</td>
<td>8:27</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>bzq-88-153-37-147.red.bezeqint.net</td>
<td>49</td>
<td>54</td>
<td>0 sec</td>
<td>6:01</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>cpe001111861b9e-cm00e06f79444.cpe.net.cable.rogers.com</td>
<td>498</td>
<td>470</td>
<td>12 sec</td>
<td>8:15</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>warbler.csail.mit.edu</td>
<td>412</td>
<td>39</td>
<td>0 sec</td>
<td>8:15</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>c-69-138-253-151.hsd1.md.comcast.net</td>
<td>59</td>
<td>46</td>
<td>0 sec</td>
<td>8:27</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>user-12m8mn.cable.mindspring.com</td>
<td>46</td>
<td>54</td>
<td>0 sec</td>
<td>6:01</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>f-ray-xps.econ.nyu.edu</td>
<td>270</td>
<td>78</td>
<td>0 sec</td>
<td>6:00</td>
<td>skypoetoskype</td>
</tr>
</tbody>
</table>
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

ntop.org
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]

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

ntop.org
Availability

- Paper and Documentation:
  - http://luca.ntop.org/VoIP.pdf

- Code and Applications
  - http://www.ntop.org/