Open Source VoIP Traffic Monitoring

Luca Deri <deri@ntop.org>
What is VoIP?

• VoIP is the **routing** of **voice** conversations over the **Internet** or through any other **IP**-based network (Wikipedia).

• Advantages:
  – It allows people to talk over the Internet at low/no cost.
  – It allows users to travel anywhere in the world and still make and receive phone calls.
  – Seamless integration with traditional phones.

• Drawbacks
  – Calls quality depend on the network speed and reliability.
  – Most of existing telephony equipments are proprietary and hard to integrate with VoIP.
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 sport 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/](http://voip.internet2.edu/), Terena TF-VVC) are mainly focusing on infrastructure.
- Traffic sniffers (e.g. etheereal) 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.
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.

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), RTP, 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.

• Skype and VoIPStrunt
  – Decentralized architecture (P2P)
  – Ability to call both users and plain phones
  – Phone calls are both encrypted and obfuscated
  – Totally closed source development model
  – So far nobody had been able to decode the protocol

• What to do then?
  – Fully support standard VoIP protocols.
  – Offer as much as possible of visibility of proprietary protocols.
Monitoring Architecture

Institution A

Institution B

Collector (ntop)

PSTN

Internet

ntop.org
Standard VoIP: Implemented Metrics [1/2]

- **SIP**
  - Unique call identifier used for accounting/billing and tracking problems.
  - 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].

ntop.org
### Standard VolP: Implemented Metrics [2/2]

<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

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

<table>
<thead>
<tr>
<th>Host Type</th>
<th>VolIP Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known Users</td>
<td>055470167 [ VoIP ]</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Source Port</th>
<th>Destination Port</th>
<th>Size</th>
<th>Duration</th>
<th>Call ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>83.175.52.136</td>
<td>83.175.54.75</td>
<td>:49650 &lt;VolIP&gt;</td>
<td>:25000</td>
<td>27.0 KB</td>
<td>055487214 called 055470167</td>
<td></td>
</tr>
<tr>
<td>83.175.52.136</td>
<td>83.175.54.75</td>
<td>:49652 &lt;VolIP&gt;</td>
<td>:quake</td>
<td>39.5 KB</td>
<td>055487214 called 055470167</td>
<td></td>
</tr>
<tr>
<td>Host</td>
<td>Domain</td>
<td>IP Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------</td>
<td>------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td></td>
<td>10.96.6.166</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Host Type</th>
<th>VoIP Host</th>
<th>HTTP Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known Users</td>
<td>yuri's music [DAAP]</td>
<td>luca deri's music [DAAP]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Host Healthness (Risk Flags)</th>
<th>1.</th>
<th>2.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>▶️</td>
<td>▶️</td>
</tr>
<tr>
<td></td>
<td>▶️</td>
<td>▶️</td>
</tr>
</tbody>
</table>
### ntop VoIP Support: Skype [2/2]

<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.videoiron.ca</td>
<td>50</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>40</td>
<td>54</td>
<td>0 sec</td>
<td>6:01</td>
<td>skypoetoskype</td>
</tr>
<tr>
<td>PowerBook G4 Luca</td>
<td>cpe001111861b9e-cm00e06f179444.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-12l8mn.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>

- **Protocol Patterns:** http://l7-filter.sourceforge.net
- **Pattern Engine:** http://www.pcre.org/
Open Issues and Future Work

- Skype/VoipStunt! support is poor (general problem with proprietary protocols).
- Implement payload analysis (e.g. of popular H.264 codec).
- Handle RTPC XS 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.
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!
- Stay tuned!

ntop.org
Availability

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

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