“pmacct, a new player in the network management arena”

http://www.pmacct.net

Paolo LUCENTE, CNR-Italy

Istanbul, 25 April 2006
What is pmacct?
pmacct: why, when and how .. (I)

• The project came out of operational needs, 3 years ago (beginnings of 2003)
• At the time it was easy to get data either:
  – “static”, ie. fixed view of your network traffic data. Full stop.
  – logged on the disk in a range of proprietary format; then APIs to get in touch with them.
  – nicely arranged on the console screen or web browser of choice.
pmacct: why, when and how .. (II)

Though, we were still missing:

• A way to get data from our network, being also able to choose how to report them and supporting multiple collection methods.

• A straight way to feed network data to external applications in order to build figures, graphs, plots, sums, etc.

• A straight way to powerfulness and flexibility offered by the SQL data language.
pmacct is a PASSIVE network monitoring tool

Passive network monitoring is basically an observation point; it enables us to understand:

✓ who is using the network.
✓ which applications/services are most used.
✓ how much bandwidth is in use over the time.
✓ are we generating DoS / target of a worm?
✓ how our BGP peerings behave.
✓ what is that sudden hill in the last traffic graph?
ACTIVE network monitoring tools

.. by contrast, they are probes injected in the network; and they enable us to understand different things:

✓ How many packets get lost?
✓ Do all the probes have the same trip?
✓ How much it takes to deliver the probe?
✓ Hey, let’s check that our premium IP offering works as expected under heavy traffic loads
pmacct, the modular architecture: one collector, multiple views
**pmacct**: reporting traffic data from broadband networks (I)
pmacct: reporting traffic data from broadband networks (II)
pmacct: an outlook of the distributed architecture
pmacct: about classification
pmacct: classification, RE

An example of Regular Expressions applied to classification (this is from the L7-filter project repository):

http/(0\.|1\.|0\.|1\.|1) [1-5][0-9][0-9] \[\x09-\x0d -~\]* (connection:|content-type:|content-length:|date:)|post \[\x09-\x0d -~\]* http/[01]\.[019]
pmacct: classification, SO

u_int32_t classifier(struct pkt_classifier_data *data, int caplen, void **context, void **rev_context, void **extra)
{
    struct rtp_context *ctx = NULL;
    rtp_hdr_t *hdr = (rtp_hdr_t *) data->payload_ptr;
    u_int16_t init;
    u_int8_t version, pt;

    init = ntohs(hdr->init);

    version = init >> 14;
    pt = init & 0x7f;

    if ( version == 2 && (pt < 35 || pt >= 96) ) {/* Possibly, we are facing a RTP stream */
        if (!(*context)) {/* We don't have enough data about the stream */
            ctx = malloc(sizeof(struct rtp_context));
            if (ctx) {
                ctx->seq = ntohs(hdr->seq);
                *context = ctx;

            }
            return 0;
        }
        else {
            ctx = (struct rtp_context *) *context;
            if (ntohs(hdr->seq) == ctx->seq+1) return 1;
        }
    }
    return 0;
}
pmacct: classification, RE vs. SO

✓ Regular Expressions (RE) classifiers are proficient against the packet payload, easy to develop and suitable for text-based protocols.
✓ Shared Object (SO) classifiers are powerful (ie. because of contexts), not limited to just catch patterns (ie. Machine Learning techniques) and deal smoothly with binary-encoded protocols. BUT require extensive and careful development.
“pmacct, a new player in the network management arena”

http://www.pmacct.net

Part II

Examples and results
The newbie hat: In+Out (sum) traffic per host (I)

shell> cat pmacctd-imt.conf

! pmacctd configuration example
!
interface: eth0
plugins: memory
!
aggregate: sum_host, flows
networks_file: networks.lst
The newbie hat: In+Out (sum) traffic per host  (II)

shell> ./pmacct -s

<table>
<thead>
<tr>
<th>SRC IP</th>
<th>PACKETS</th>
<th>FLOWS</th>
<th>BYTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.145.84.4</td>
<td>2</td>
<td>2</td>
<td>152</td>
</tr>
<tr>
<td>150.145.82.19</td>
<td>7594</td>
<td>38</td>
<td>6584356</td>
</tr>
<tr>
<td>150.145.87.15</td>
<td>1</td>
<td>1</td>
<td>128</td>
</tr>
<tr>
<td>150.145.90.255</td>
<td>2</td>
<td>2</td>
<td>466</td>
</tr>
<tr>
<td>150.145.80.51</td>
<td>127224</td>
<td>8819</td>
<td>23678985</td>
</tr>
<tr>
<td>150.145.81.18</td>
<td>2</td>
<td>2</td>
<td>460</td>
</tr>
<tr>
<td>150.145.87.159</td>
<td>83</td>
<td>11</td>
<td>8758</td>
</tr>
<tr>
<td>150.145.80.0</td>
<td>22</td>
<td>1</td>
<td>1144</td>
</tr>
<tr>
<td>150.145.87.108</td>
<td>1</td>
<td>1</td>
<td>247</td>
</tr>
<tr>
<td>150.145.84.156</td>
<td>34</td>
<td>9</td>
<td>2856</td>
</tr>
<tr>
<td>150.145.81.255</td>
<td>33</td>
<td>7</td>
<td>6662</td>
</tr>
<tr>
<td>150.145.82.10</td>
<td>1423</td>
<td>30</td>
<td>1091800</td>
</tr>
<tr>
<td>150.145.87.6</td>
<td>16787</td>
<td>3361</td>
<td>929034</td>
</tr>
</tbody>
</table>

[ ... continues ... ]
The newbie hat: In+Out (sum) traffic per host (III)

a) The –M : getting a specific entry wrapped by a formatted output
shell> ./pmacct -c src_host -M 150.145.80.101

<table>
<thead>
<tr>
<th>SRC IP</th>
<th>PACKETS</th>
<th>FLOWS</th>
<th>BYTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.145.80.101</td>
<td>287522</td>
<td>2616</td>
<td>273081046</td>
</tr>
</tbody>
</table>

b) The –N : getting the counters. Introducing the -r reset flag. The quick way to glue pmacct to external tools
shell> ./pmacct -c src_host -N 150.145.80.101 -r
334701089

shell> ./pmacct -c src_host -N 150.145.80.101
2790707
Building network traffic graphs (I)

interface: eth0
plugins: memory[out], memory[in]

aggregate[out]: src_net
aggregate_filter[out]: vlan and src net 150.145.80.0/20
imt_path[out]: /tmp/pmacct_out.pipe

aggregate[in]: dst_net
aggregate_filter[in]: vlan and dst net 150.145.80.0/20
imt_path[in]: /tmp/pmacct_in.pipe
Building network traffic graphs (II)

```bash
shell> cat mrtg-example.sh
#!/bin/sh

unset OUT
unset IN

OUT=`pmacct -c src_host –p /tmp/pmacct_out.pipe -N 150.145.80.0 -r`
IN=`pmacct -c dst_host –p /tmp/pmacct_in.pipe -N 150.145.80.0 -r`

echo $OUT
echo $IN
echo 0
echo 0
```
Building network traffic graphs (III)

```
shell> cat mrtg.conf
[ ... ]
# Target specific definitions
Target[pp]: `/usr/local/pmacct/scripts/mrtg-example.sh`
SetEnv[pp]: MRTG_INT_IP="150.145.80.0" MRTG_INT_DESCR=“Server LAN"
MaxBytes[pp]: 1250000
LegendI[pp]:
Title[pp]: Server LAN
PageTop[pp]: <H1>Server LAN</H1>
<TABLE>
  <TR><TD>System:</TD>  <TD>Server LAN</TD></TR>
  <TR><TD>Maintainer:</TD> <TD>CNR-BA Staff</TD></TR>
  <TR><TD>Ip:</TD>  <TD>150.145.80.0</TD></TR>
</TABLE>
[ ... ]
```
Network traffic data, the SQL way

interface: eth0
plugins: pgs ql[out], pgs ql[in]

aggregate[out]: src_host
aggregate_filter[out]: vlan and src net 150.145.80.0/20
sql_table[out]: acct_out

aggregate[in]: dst_host
aggregate_filter[in]: vlan and dst net 150.145.80.0/20
sql_table[in]: acct_in

sql_refresh_time: 60
sql_history: 1h
sql_history_roundoff: h
sql_preprocess: minb=60000
Network traffic data, the SQL way (II)

shell> psql -U pmacct -c "SELECT * FROM acct_out 
   WHERE ip_src = '150.145.80.101' 
   ORDER BY stamp_inserted DESC 
   LIMIT 10;"

<table>
<thead>
<tr>
<th>ip_src</th>
<th>packets</th>
<th>bytes</th>
<th>stamp_inserted</th>
<th>stamp_updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>150.145.80.101</td>
<td>355394</td>
<td>29925806</td>
<td>2006-01-08 16:00:00</td>
<td>2006-01-08 16:48:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>556245</td>
<td>46096570</td>
<td>2006-01-08 15:00:00</td>
<td>2006-01-08 16:00:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>26364</td>
<td>12618610</td>
<td>2006-01-08 14:00:00</td>
<td>2006-01-08 15:00:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>196319</td>
<td>16508068</td>
<td>2006-01-08 13:00:00</td>
<td>2006-01-08 14:00:01</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>341143</td>
<td>40921593</td>
<td>2006-01-08 12:00:00</td>
<td>2006-01-08 13:00:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>208050</td>
<td>30011464</td>
<td>2006-01-08 11:00:00</td>
<td>2006-01-08 12:00:01</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>196337</td>
<td>15404272</td>
<td>2006-01-08 10:00:00</td>
<td>2006-01-08 11:01:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>205970</td>
<td>16656939</td>
<td>2006-01-08 09:00:00</td>
<td>2006-01-08 10:00:03</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>376094</td>
<td>22589504</td>
<td>2006-01-08 08:00:00</td>
<td>2006-01-08 09:00:02</td>
</tr>
<tr>
<td>150.145.80.101</td>
<td>14779</td>
<td>6913855</td>
<td>2006-01-08 07:00:00</td>
<td>2006-01-08 08:01:01</td>
</tr>
</tbody>
</table>
(10 rows)
Network traffic data, the SQL way: what about “top N”?

```
shell> psql -U pmacct -c "SELECT port_dst, ip_proto, packets, bytes 
    FROM dst_ports_db 
    WHERE dst_src = '150.145.80.101' AND 
    stamp_inserted = '2006-01-09 12:00:00' 
    ORDER BY bytes DESC 
    LIMIT 10;"
```

<table>
<thead>
<tr>
<th>port_dst</th>
<th>ip_proto</th>
<th>packets</th>
<th>bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>119</td>
<td>6</td>
<td>1084915</td>
<td>1594897858</td>
</tr>
<tr>
<td>25</td>
<td>6</td>
<td>385883</td>
<td>374188510</td>
</tr>
<tr>
<td>80</td>
<td>6</td>
<td>24632</td>
<td>26649410</td>
</tr>
<tr>
<td>110</td>
<td>6</td>
<td>14595</td>
<td>15556361</td>
</tr>
<tr>
<td>22</td>
<td>6</td>
<td>10775</td>
<td>13201890</td>
</tr>
<tr>
<td>443</td>
<td>6</td>
<td>2943</td>
<td>1929708</td>
</tr>
<tr>
<td>143</td>
<td>6</td>
<td>911</td>
<td>1111241</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>607</td>
<td>879218</td>
</tr>
<tr>
<td>995</td>
<td>6</td>
<td>9399</td>
<td>541329</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>140</td>
<td>188855</td>
</tr>
</tbody>
</table>

(10 rows)
Network traffic data, the SQL way: classification and “top N”!

```sql
shell> psql -U pmacct -c "SELECT class_id, packets, bytes, flows 
FROM acct_v5 
ORDER BY bytes DESC 
LIMIT 10;"
```

<table>
<thead>
<tr>
<th>class_id</th>
<th>packets</th>
<th>bytes</th>
<th>flows</th>
</tr>
</thead>
<tbody>
<tr>
<td>nntp</td>
<td>533424546</td>
<td>534913922183</td>
<td>13480</td>
</tr>
<tr>
<td>http</td>
<td>567179034</td>
<td>409970727835</td>
<td>22581928</td>
</tr>
<tr>
<td>smtp</td>
<td>336913736</td>
<td>116445824169</td>
<td>17286471</td>
</tr>
<tr>
<td>ssh</td>
<td>139908289</td>
<td>108291107166</td>
<td>1110903</td>
</tr>
<tr>
<td>edonkey</td>
<td>167213900</td>
<td>107343376842</td>
<td>4501937</td>
</tr>
<tr>
<td>ftp</td>
<td>197626712</td>
<td>97059417721</td>
<td>139749</td>
</tr>
<tr>
<td>pop3</td>
<td>86367951</td>
<td>60221933775</td>
<td>1462006</td>
</tr>
<tr>
<td>ssl</td>
<td>62489714</td>
<td>34784217799</td>
<td>2602435</td>
</tr>
<tr>
<td>bittorrent</td>
<td>52031296</td>
<td>31068910458</td>
<td>414216</td>
</tr>
<tr>
<td>rtsp</td>
<td>20099589</td>
<td>9595494054</td>
<td>3959</td>
</tr>
</tbody>
</table>

(10 rows)
pmacct: results (I)
by Martin Pot, from RRDtool gallery
pmacct: results (II)

pmacct-fe screenshot (A)
pmacct: results (II)

pmacct-fe screenshot (B)
pmacct: results  (III)

network weather maps with GWEN
A preview of FloX, the flow explorer by Sven Anderson
Thank you for your attention!

http://www.pmacct.net

Paolo LUCENTE, paolo@pmacct.net