Large-Scale Flow Monitoring Through Open Source Software

Luca Deri <deri@ntop.org>
Monitoring Goals

- Analysis of LAN and WAN Traffic
- Unaggregated raw data storage for the near past (-3 days) and long-term data aggregation on selected network traffic metrics (limit: available disk space)
- Data navigation by means of a web 2.0 GUI
- Geolocation of network flows and their aggregation based on their geographical source.
- Integration with routing information in order to provide accurate traffic path analysis.
Traffic Collection Architecture [1/2]

• Available Options
  1. Exploit network equipment (routers and switches)
     – Advantages:
       • Maximize investment.
       • Avoid adding extra network equipment/complexity in the network.
       • No additional point of Failure
     – Disadvantages:
       • Often is necessary to buy costly netflow engines
       • Have to survive with bugs (e.g. Juniper have issues with AS information)
Traffic Collection Architecture [2/2]

2. Custom Network Probes

- **Advantages**
  - Ability to avoid limitations of commercial equipment
  - (Often) Faster and more flexible than hw probes

- **Disadvantages**
  - Add complexity to the net
  - Need to mirror/wiretap traffic
Introduction to Cisco NetFlow

• Flow: “Set of network packets with some properties in common”. Typically (IP src/dst, Port src/dst, Proto, TOS, VLAN).

• Network Flows contain:
  – Peers: flow source and destination.
  – Counters: packets, bytes, time.
  – Routing information: AS, network mask, interfaces.
Collection Architectures [1/2]

- Live feed
- flow collector
- flow-capture
- flow-rsync transfer
- NetFlow export
- Backbone
- Flow Archive
- flow enabled router
Collection Architectures [2/2]
Flow Journey: Creation
Flow Journey: Export

1. Flow Cache—The First Unique Packet Creates a Flow

<table>
<thead>
<tr>
<th>SrcIf</th>
<th>SrcIPadd</th>
<th>DstIf</th>
<th>DstIPadd</th>
<th>Protocol</th>
<th>TOS</th>
<th>Flgs</th>
<th>Pkts</th>
<th>Src Port</th>
<th>Src Msk</th>
<th>Src AS</th>
<th>Dst Port</th>
<th>Dst Msk</th>
<th>Dst AS</th>
<th>NextHop</th>
<th>Bytes/Pkt</th>
<th>Active</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fa1/0</td>
<td>173.100.212</td>
<td>Fa0/0</td>
<td>10.0.227.12</td>
<td>11</td>
<td>80</td>
<td>10</td>
<td>11000</td>
<td>162</td>
<td>/24</td>
<td>163</td>
<td>/24</td>
<td>15</td>
<td>10.0.232</td>
<td>1528</td>
<td>1745</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Fa1/0</td>
<td>173.100.321</td>
<td>Fa0/0</td>
<td>10.0.227.12</td>
<td>11</td>
<td>80</td>
<td>10</td>
<td>11000</td>
<td>162</td>
<td>/24</td>
<td>163</td>
<td>/24</td>
<td>15</td>
<td>10.0.232</td>
<td>740</td>
<td>415</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Fa1/0</td>
<td>173.100.202</td>
<td>Fa0/0</td>
<td>10.0.227.12</td>
<td>11</td>
<td>80</td>
<td>10</td>
<td>11000</td>
<td>162</td>
<td>/24</td>
<td>163</td>
<td>/24</td>
<td>15</td>
<td>10.0.232</td>
<td>1428</td>
<td>1145</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Fa1/0</td>
<td>173.100.621</td>
<td>Fa0/0</td>
<td>10.0.227.12</td>
<td>11</td>
<td>80</td>
<td>10</td>
<td>11000</td>
<td>162</td>
<td>/24</td>
<td>163</td>
<td>/24</td>
<td>15</td>
<td>10.0.232</td>
<td>1040</td>
<td>245</td>
<td>14</td>
<td></td>
</tr>
</tbody>
</table>

- Inactive Flow (15 sec is default)
- Long Flow (30 min (1800 sec) is default)
- Flow ends by RST or FIN TCP Flag

2. Flow Aging Timers

3. Flows Packaged in Export Packet
   Non-Aggregated Flows—Export Version 5 or 9

4. Transport Flows to Reporting Server
## Flow Format: NetFlow v5 vs v9

<table>
<thead>
<tr>
<th></th>
<th>v5</th>
<th>v9</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flow Format</strong></td>
<td>Fixed</td>
<td>User Defined</td>
</tr>
<tr>
<td><strong>Extensible</strong></td>
<td>No</td>
<td>Yes (Define new FlowSet Fields)</td>
</tr>
<tr>
<td><strong>Flow Type</strong></td>
<td>Unidirectional</td>
<td>Bidirectional</td>
</tr>
<tr>
<td><strong>Flow Size</strong></td>
<td>48 Bytes (fixed)</td>
<td>It depends on the format</td>
</tr>
<tr>
<td><strong>IPv6 Aware</strong></td>
<td>No</td>
<td>IP v4/v6</td>
</tr>
<tr>
<td><strong>MPLS/VLAN</strong></td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Flow Format: NetFlow v9/IPFIX
InMon sFlow

Switch/Router

sFlow agent

ASIC

HW Packet Sampling

Network Traffic

sFlow Datagram

- Packet header (e.g. MAC, IPv4, IPv6, IPX, AppleTalk, TCP, UDP, ICMP)
- Sample process parameters (rate, pool etc.)
- Input/output ports
- Priority (802.1p and TOS)
- VLAN (802.1Q)
- Source/destination prefix
- Next hop address
- Source AS, Source Peer AS
- Destination AS Path
- Communities, local preference
- User IDs (TACACS/RADIUS) for source/destination
- URL associated with source/destination
- Interface statistics (RFC 1573, RFC 2233, and RFC 2358)

% Sampling Error <= 196 * sqrt(1 / number of samples)

[http://www.sflow.org(packetSamplingBasics/)
Integrated Network Monitoring

- sFlow enabled switches
  - Core network switches
- RMON enabled switches
  - L2/L3 Switches
- NetFlow enabled routers

Traffic Analysis & Accounting Solutions

- Network-wide, continuous surveillance
  - 20K+ ports from a single point
- Timely data and alerts
  - Real-time top talkers
  - Site-wide thresholds and alarms
- Consolidated network-wide historical usage data
Traffic Collection: A Real Scenario

- Registro.it
- Juniper Switch
- Juniper Router
- NetFlow v9
- sFlow v5
- anifani.nic.it
- monitor.nic.it
- Level 3
- GARR
Heterogeneous Flow Collection

- sFlow v5
  - nProbe
  - Fastbit

- NetFlow v9
  - nProbe
  - Fastbit

- Web Console
- Web Server
nProbe: sFlow/NF/IPFIX Probe+Collector

Packet Capture → nProbe → Flow Export

sFlow → nProbe → Data Dump

NetFlow → nProbe → Data Dump

Data Dump → Raw Files / MySQL / SQLite / FastBit
Problem Statement [1/2]

- NetFlow and sFlow are the current state-of-the-art standard for network traffic monitoring.
- As the number of generated flows can be quite high, operators often use sampling in order to reduce their number.
- Sampling leads to inaccuracy so it cannot always be used in production networks.
- Thus network operators have to face the problem of collecting and analyzing a large number of flow records.
Problem Statement [2/2]

Where to store collected flows?

- Relational Databases
  - Pros: Expressiveness of SQL for data search.
  - Cons: Sacrifice flow collection speed and query response time.

- Raw Disk Archives
  - Pros: Efficient flow-to-disk collection speed (> 250K flow/s).
  - Cons: Limited query facilities as well search time proportional to the amount of collected data (i.e. no indexing is used).
Towards Column-Oriented Databases [1/3]

• Network flow records are read-only, shouldn’t be modified after collection, and several flow fields have very few unique values.

• B-tree/hash indexes used in relational DBs to accelerate queries, encounter performance issues with large tables as:
  – need to be updated whenever a new flow is stored.
  – require a large number of tree-branching operations as they use slow pointer chases in memory and random disk access (seek), thus taking a long time.

• Thus with relational DBs it is not possible to do live flow collection/import as index update will lead to flow loss.
Towards Column-Oriented Databases [2/3]

• A column-oriented database stores its content by column rather than by row. As each column is stored contiguously, compression ratios are generally better than row-stores because consecutive entries in a column are homogeneous to each other.

• Column-stores are more I/O efficient (than row stores) for read-only queries since they only have to read from disk (or from memory) those attributes accessed by a query.

• Indexes that use bit arrays (called bitmaps) answer queries by performing bitwise logical operations on these bitmaps.
Towards Column-Oriented Databases [3/3]

- Bitmap indexes perform extremely well because the intersection between the search results on each value is a simple AND operation over the resulting bitmaps.
- As column data can be individually sorted, bitmap indexes are also very efficient for range queries (e.g. subnet search) as data is contiguous hence disk seek is reduced.
- As column-oriented databases with bitmap indexes provide better performance compared to relational databases, the authors explored their use in the field of flow monitoring.
nProbe + FastBit

- FastBit is not a database but a C++ library that implements efficient bitmap indexing methods.
- Data is represented as tables with rows and columns.
- A large table may be partitioned into many data partitions and each of them is stored on a distinct directory, with each column stored as a separated file in raw binary form.
- nProbe natively integrates FastBit support and it automatically creates the DB schema according to the flow records template.
- Flows are saved in blocks of 4096 records.
- When a partition is fully dumped, columns to be indexed are first sorted then indexed.
# Performance Evaluation: Disk Space

<table>
<thead>
<tr>
<th>Tool</th>
<th>Indexes</th>
<th>Result (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MySQL</td>
<td>No/With Indexes</td>
<td>1.9 / 4.2</td>
</tr>
<tr>
<td>FastBit</td>
<td>Daily Partition (no/with Indexes)</td>
<td>1.9 / 3.4</td>
</tr>
<tr>
<td></td>
<td>Hourly Partition (no/with Indexes)</td>
<td>1.9 / 3.9</td>
</tr>
<tr>
<td>nfdump</td>
<td>No indexes</td>
<td>1.9</td>
</tr>
</tbody>
</table>

Results are in GB
### Performance Evaluation: Query Time [1/2]

**nProbe+FastBit vs MySQL**

<table>
<thead>
<tr>
<th>Query</th>
<th>MySQL</th>
<th>nProbe + FastBit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No Index</td>
<td>Daily Partitions</td>
</tr>
<tr>
<td></td>
<td>With Indexes</td>
<td>No Cache</td>
</tr>
<tr>
<td>Q1</td>
<td>20.8</td>
<td>22.6</td>
</tr>
<tr>
<td>Q2</td>
<td>23.4</td>
<td>69</td>
</tr>
<tr>
<td>Q3</td>
<td>796</td>
<td>971</td>
</tr>
<tr>
<td>Q4</td>
<td>1033</td>
<td>1341</td>
</tr>
<tr>
<td>Q5</td>
<td>1754</td>
<td>2257</td>
</tr>
</tbody>
</table>

Results are in seconds
Performance Evaluation: Query Time [2/2]

nProbe+FastBit vs nfdump

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>nProbe+FastBit</td>
<td>45 sec</td>
</tr>
<tr>
<td>nfdump</td>
<td>1500 sec</td>
</tr>
</tbody>
</table>

SELECT IPV4_SRC_ADDR, L4_SRC_PORT, IPV4_DST_ADDR, L4_DST_PORT, PROTOCOL FROM NETFLOW WHERE IPV4_SRC_ADDR=X OR IPV4_DST_ADDR=X

worth 19 GB of data (14 hours of collected flows)

nfdump query time = (time to sequentially read the raw data) + (record filtering time)
nProbe + Fastbit: Collection Architecture

deri@MacBook-2.local  239>  ls /tmp/2010/04/06/16/20
total 352
 8 -part.txt  24 IPV4_DST_ADDR.idx  16 LAST_SWITCHED
 8 DST_AS    16 IPV4_NEXT_HOP   8 OUTPUT_SNMP
 8 DST_MASK  16 IPV4_SRC_ADDR   8 PROTOCOL
16 FIRST_SWITCHED  24 IPV4_SRC_ADDR.idx  8 SRC_AS
 8 INPUT_SNMP  8 L4_DST_PORT   8 SRC_MASK
16 IN_BYTES  48 L4_DST_PORT.idx  8 SRC_TOS
16 IN_PKTS   8 L4_SRC_PORT    8 TCP_FLAGS
16 IPV4_DST_ADDR 48 L4_SRC_PORT.idx

--fastbit <dir> | Base directory where FastBit files will be created.
--fastbit-rotation <mins> | Every <mins> minutes a new FastBit sub-directory is created
                              | so that each directory contains at most <mins> minutes.
                              | Default 5 min(s).
--fastbit-template <flow template> | Fields that will be dumped on FastBit partition. Its syntax
                              | is the same as the -T flag. If this flag is not specified,
                              | all the specified flow elements (-T) will be dumped.
--fastbit-index <flow template> | Index each directory containing FastBit files as soon as
                              | the directory has been dumped. The flow template specifies
                              | which columns will be indexed. Its syntax is the same as
                              | the -T flag. This option requires that fbindex application
                              | is installed or built. If this flag is not specified, all
                              | columns will be indexed.
--fastbit-exec <cmd> | Execute the specified command after a directory has been
                              | dumped (and optionally indexed). The command must take an
                              | argument that is the path to the directory just dumped.
Host Geolocation [1/3]

- Host geolocation is a known problem (via http://en.wikipedia.org/wiki/Geoip)
- Need to handle thousand flows/sec (no inline internet query)
- Requirements: IP -> Location e IP -> ASN
Host Geolocation [2/3]

• Interactive Flash™ world map, that displays hosts distribution by country and by cities of a selected country
• nProbe + GeoIP + Python + Google Visualization. The script
  – Cycles through all the hosts seen by ntop
  – Gets their GeoIP info
  – Counts them based on their location.
• Google GeoMap and Visualization Table
• Ajax/JSON communications with web server for updated data
Host Geolocation [3/3]
How to Add Geolocation Data [1/3]

- Routers are unable to export any geolocation information.
- NetFlow/IPFIX flows do not contain any information about geolocation into standard flow formats.
- Solution:
  - Let the collector add geolocation information to flows received by routers
  - Let the softprobe export this information to collectors.
How to Add Geolocation Data [2/3]

- nProbe takes advantage of GeoIP library (GPL) to
  - Add geolocation information to flows
  - Map IP addresses to ASN (Autonomous System Numbers) for adding ASN awareness.

- GeoIPASNum.dat  (ASN)
- GeoLiteCity.dat   (GeoLocation)
if(host->ipVersion == 4)
    return(GeoIP_record_by_ipnum(readOnlyGlobals.geo_ip_city_db, host->ipType.ipv4));
#ifdef INET6
    else if(host->ipVersion == 6)
        return(GeoIP_record_by_ipnum_v6(readOnlyGlobals.geo_ip_city_db, host->ipType.ipv6));
#endif

char *rsp = NULL;
u_int32_t as;

if(ip.ipVersion == 4)
    rsp = GeoIP_name_by_ipnum(readOnlyGlobals.geo_ip_asn_db, ip.ipType.ipv4);
else {
#ifdef INET6
    rsp = GeoIP_name_by_ipnum_v6(readOnlyGlobals.geo_ip_asn_db, ip.ipType.ipv6);
#endif
}

as = rsp ? atoi(&rsp[2]) : 0;
free(rsp);
deri@anifani 205> pwd
/home/deri/fastbit/netflow/2010/05/24/23/25

deri@anifani 206> ls
total 115848
  4 -part.txt  1848 INPUT_SNMP  928 PROTOCOL
1848 DST_AS   3692 IN_BYTES   204 PROTOCOL.idx
3692 DST_AS_PATH_1  3692 IN_PKTS  1848 SRC_AS
3692 DST_AS_PATH_2  3692 IPV4_DST_ADDR  3692 SRC_AS_PATH_1
3692 DST_AS_PATH_3  3564 IPV4_DST_ADDR.idx  3692 SRC_AS_PATH_2
3692 DST_AS_PATH_4  3692 IPV4_NEXT_HOP  3692 SRC_AS_PATH_3
3692 DST_AS_PATH_5  3692 IPV4_SRC_ADDR  3692 SRC_AS_PATH_4
3692 DST_AS_PATH_6  3528 IPV4_SRC_ADDR.idx  3692 SRC_AS_PATH_5
3692 DST_AS_PATH_7  1848 L4_DST_PORT  3692 SRC_AS_PATH_6
3692 DST_AS_PATH_8  5144 L4_DST_PORT.idx  3692 SRC_AS_PATH_7
1848 DST_IP_COUNTRY  1848 L4_SRC_PORT  3692 SRC_AS_PATH_8
3692 FIRST_SWITCHED  3692 LAST_SWITCHED  1848 SRC_IP_COUNTRY
1848 FLOW_PROTO_PORT  1848 OUTPUT_SNMP  928 TCP_FLAGS
BGP Data Integration [1/2]

- nProbe
- Patricia Tree
- TCP
- BGP4
- Juniper Router
- BGP Client (Net-BGP)
- Initial BGP Table Dump
- Live BGP Update
# Constructor
$update = Net::BGP::Update->new(
    NLRI => [ qw( 10/8 172.168/16 ) ],
    Withdraw => [ qw( 192.168.1/24 172.10/16 192.168.2.1/32 ) ],
    # For Net::BGP::NLRI
    Aggregator => [ 64512, '10.0.0.1' ],
    AsPath => [ 64512, 64513, 64514 ],
    AtomicAggregate => 1,
    Communities => [ qw( 64512:10000 64512:10001 ) ],
    LocalPref => 100,
    MED => 200,
    NextHop => '10.0.0.1',
    Origin => INCOMPLETE,
);
What if you have no BGP Router? [1/3]
What if you have no BGP Router? [2/3]

Index of /rrc10/2010.06

<table>
<thead>
<tr>
<th>Name</th>
<th>Last modified</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parent Directory</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bview.20100601.0759.gz</td>
<td>01-Jun-2010 08:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100601.1559.gz</td>
<td>01-Jun-2010 16:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100601.2359.gz</td>
<td>02-Jun-2010 00:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100602.0759.gz</td>
<td>02-Jun-2010 08:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100602.1559.gz</td>
<td>02-Jun-2010 16:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100602.2359.gz</td>
<td>03-Jun-2010 00:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100603.0759.gz</td>
<td>03-Jun-2010 08:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100603.1559.gz</td>
<td>03-Jun-2010 16:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100603.2359.gz</td>
<td>04-Jun-2010 00:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100604.0759.gz</td>
<td>04-Jun-2010 08:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100604.1559.gz</td>
<td>04-Jun-2010 16:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100604.2359.gz</td>
<td>05-Jun-2010 00:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100613.2359.gz</td>
<td>14-Jun-2010 00:00</td>
<td>8.6M</td>
<td></td>
</tr>
<tr>
<td>bview.20100614.0759.gz</td>
<td>14-Jun-2010 08:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>bview.20100614.1559.gz</td>
<td>14-Jun-2010 16:00</td>
<td>8.7M</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0000.gz</td>
<td>01-Jun-2010 00:04</td>
<td>15K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0005.gz</td>
<td>01-Jun-2010 00:10</td>
<td>14K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0010.gz</td>
<td>01-Jun-2010 00:15</td>
<td>19K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0015.gz</td>
<td>01-Jun-2010 00:20</td>
<td>17K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0020.gz</td>
<td>01-Jun-2010 00:25</td>
<td>12K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0025.gz</td>
<td>01-Jun-2010 00:29</td>
<td>13K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0030.gz</td>
<td>01-Jun-2010 00:35</td>
<td>17K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0035.gz</td>
<td>01-Jun-2010 00:40</td>
<td>19K</td>
<td></td>
</tr>
<tr>
<td>updates.20100601.0040.gz</td>
<td>01-Jun-2010 00:45</td>
<td>11K</td>
<td></td>
</tr>
</tbody>
</table>
What if you have no BGP Router? [3/3]

- **libbpgdump** can be used to read BGP dump and updates.
- Periodically poll the RIPE RIS directory searching for full dumps or updates.
- Connect to the probe and refresh the routes according to the values being read.
- **NOTE:** always use the BGP dumps for a location near to you in order to have your view of the Internet.
Implementing a Web 2.0 GUI

• Web server: Lighttpd (easy and fast), avoid Apache.

• Ajax: use established frameworks such as jQuery or Prototype.

• Implement class libraries used to read your monitoring data. Python is used for speed, ease of use and script compilation.

• Use templates (e.g. Mako) for generating (XML-free) pages.

• Web frameworks are perhaps easier to use, but you will be bound to them forever (pros and cons).
Storing Historical Data [1/2]

- RRD is the de-facto standard for permanently storing numerical data.

```perl
$rrd = "$dataDir/$agent-$ifIndex.rrd";
if(! -e $rrd) {
    RRDs::create ($rrd, "--start", $now-1, "--step", 20,
    "DS:bytesIn:COUNTER:120:0:10000000",
    "DS:bytesOut:COUNTER:120:0:10000000",
    "RRA:AVERAGE:0.5:3:288");
    $ERROR = RRDs::error;
    die "$0: unable to create `$rrd': $ERROR\n" if $ERROR;
}
RRDs::update $rrd, "$now:$ifInOctets:$ifOutOctets";
if ($ERROR = RRDs::error) {
    die "$0: unable to update `$rrd': $ERROR\n";
}
```
Storing Historical Data [2/2]

- RRD has several limitations:
  - Only one (quantity one) numerical data can be stored at each time interval (e.g. # of bytes received).
  - You must know ‘in advance’ what you want to store. For instance you can’t store anything like ‘the name and amount of traffic sent by the top host’: the top host changes overtime, so you need an rrd per top host and this is not what you want.
  - Sets or lists of data (e.g. top protocols with bytes on interval X) cannot be stored in RRD.
Beyond RRD

• Requirements:
  – Store network values are tuples (list of `<name>:<value>`, where `<value>` can also be a list).
  – Ability to aggregate tuples using a user-defined function (i.e. not just max/min/average).
  – Manipulate values as RRD does: create, update, last, export, fetch and graph.
  – Graph: images are not enough as we have tuples (not just one value) and also because the user must be able to interact with data, not just look at it.
pSWTDB [1/4]

• pSWTDB (Sliding Window Tuple DB).
• python class used to store tuples on disk using data serialization (called pickle on python).
  – Pros:
    • native in python
    • portable across datatypes (i.e. no need to define the type)
  – Cons:
    • Slow as RRD (deserialize/update/serialize at each update)
• Same principle of RRD with the exception that here we use tuples and not numerical values.
pSWTDB [2/4]

- It comes with aggregation functions such as:
  - Each time interval has a list of (key, value).
  - Sum values with same key.
  - Sort values
  - Discard values ranking after position X (e.g. take the top/bottom X values).

- Examples
  - Top X protocols (list of <proto>:<value>)
  - Top X hosts (list of <host>:(<proto>:<value>,...))
Data are plotted using SVG/JavaScript.

Users can interact with data (pan, zoom, move).

Multiple criteria can be plotted at the same time (e.g. top X hosts and Y protocols).

Clicking on data can be used to trigger GUI updates.
deri@MacLuca.local 233> cat pcreate.py
#!/usr/bin/python
import pSWTDB

t = pSWTDB.pSWTDB('ptest.pkl')
# Hearbeat is 5 min
  t.create(300)

# Keep 60 samples, one per minute
  t.add_base_aggregation('1min', 60, 60)

# Keep 50 samples, each aggregating 5 samples
# of the base aggregation
  t.add_aggregation('5min', 5, 50, pSWTDB.sum, '')

# Keep 60 samples, each aggregating 24 samples
# of the 5min aggregation
  t.add_aggregation('hour', 24, 60, pSWTDB.sum, '5min')

# Keep 30 samples, each aggregating 12 samples
# of the hour aggregation
  t.add_aggregation('day', 12, 30, pSWTDB.sum, 'hour')

deri@MacLuca.local 238> cat pfetch.py
#!/usr/bin/python
import pSWTDB
import pprint

t = pSWTDB.pSWTDB('IT.pkl')
ret = t.fetch('', 'now-1h', 'now')
print t.plot(ret)
Traffic Data Analysis [1/4]

Flow collection and storage in FastBit Archive Format (5 min timeframe partition)

Column data sort and data indexing

Partition data analysis

Metrics persistent storage

```
deri@anifani 203> ls -lL
 total 24
 4 -rwxr-xr-x 1 deri deri 1377 Mar 27 12:06 cities.py*
 4 -rwxr-xr-x 1 deri deri  950 Mar 23 23:21 flows.py*
 4 -rwxr-xr-x 1 deri deri 2162 May 22 13:49 top_n_flows_countries.py*
 4 -rwxr-xr-x 1 deri deri 2106 Mar 25 15:48 top_n_l7_protocols.py*
 4 -rwxr-xr-x 1 deri deri 4565 May 22 14:32 top_n_proto_countries.py*

deri@anifani 204> pwd
/home/deri/nProbe/fastbit/python/partition_scripts
```
Traffic Data Analysis [2/4]

deri@anifani 208> ls -l
total 24
16 drwxr-xr-x 3 root root 16384 May 25 08:21 aggregations/
 4 drwxr-xr-x 4 deri deri 4096 Mar 27 12:07 queries/
 4 drwxr-xr-x 6 deri deri 4096 Mar 18 19:37 rrd/
deri@anifani 209> ls -l *
aggregations:
total 34000
   20  -rw-r--r--  1 root root 18768 May 25 16:12 A1.pkl
 164  -rw-r--r--  1 root root 167641 May 25 16:12 A2.pkl
 152  -rw-r--r--  1 root root 154778 May 25 16:12 AD.pkl
 216  -rw-r--r--  1 root root 219872 May 25 16:13 AE.pkl
 148  -rw-r--r--  1 root root 148012 May 25 16:13 AF.pkl
 152  -rw-r--r--  1 root root 152841 May 25 16:13 AG.pkl
 100  -rw-r--r--  1 root root 100615 May 25 16:12 AI.pkl
...
 152  -rw-r--r--  1 root root 154259 May 25 16:13 YE.pkl
 12  -rw-r--r--  1 root root 10101 May 25 15:13 YT.pkl
 200  -rw-r--r--  1 root root 201469 May 25 16:12 ZA.pkl
 148  -rw-r--r--  1 root root 151246 May 25 16:12 ZM.pkl
 156  -rw-r--r--  1 root root 156071 May 25 16:12 ZW.pkl
 308  -rw-r--r--  1 root root 315311 May 25 16:13 all_countries.pkl
   4  -rw-r--r--  1 root root    789 May 25 15:10 ne.pkl
 24 drwxr-xr-x 2 root root 20480 May 22 13:57 top_hosts/

queries:
total 8
 4 drwxr-xr-x 7 deri deri 4096 May  1 00:05 2010/

rrd:
total 144
 48  -rw-r--r--  1 root root 47128 May 25 16:13 bits.rrd
 12 drwxr-xr-x  2 root root 12288 May  6 02:06 bytes/
 12 drwxr-xr-x  475 root root 12288 May 16 19:26 country/
 12 drwxr-xr-x  2 root root 12288 May 24 23:36 flows/
 48  -rw-r--r--  1 root root 47128 May 25 16:13 flows.rrd
 12 drwxr-xr-x  2 root root 12288 May 12 20:42 pkts/
Traffic Data Analysis [3/4]

```
rrd/country/CH/mandelspawn.rrd
rrd/country/CH/gds_db.rrd
rrd/country/CH/dircproxy.rrd
rrd/country/CH/rmtcfg.rrd
rrd/country/CH/ssh.rrd
rrd/country/CH/isisd.rrd
rrd/country/CH/cfinger.rrd
rrd/country/CH/gris.rrd
rrd/country/CH/daap.rrd
rrd/country/CH/x11.rrd
rrd/country/CH/postgresql.rrd
rrd/country/CH/amanda.rrd
rrd/country/CH/zephyr-hm.rrd
rrd/country/CH/gsigatekeeper.rrd
rrd/country/CH/fax.rrd
rrd/country/CH/netbios-ssn.rrd
rrd/country/CH/afs3-fileserver.rrd
rrd/country/CH/cvpsserver.rrd
rrd/country/CH/ospf6d.rrd
rrd/country/CH/bpcd.rrd
rrd/country/CH/proofd.rrd
rrd/country/CH/afs3-errors.rrd
rrd/country/CH/ggz.rrd
rrd/country/CH/tproxy.rrd
rrd/country/CH/cfengine.rrd
rrd/country/CH/x11-6.rrd
rrd/country/CH/msp.rrd
rrd/country/CH/rje.rrd
rrd/country/CH/sane-port.rrd
rrd/country/CH/smtp.rrd
```

deri@anifani 213> ls queries/2010/05/25/16/00/
total 1172
1164 cities.pkl 8 top_n_l7_protocols.pkl
Traffic Data Analysis [4/4]

deri@anifani 215> ~/nProbe/fastbit/python/dump.py cities.pkl |m
Remote Probe Deployment [1/2]

• In order to monitor a distributed network it is often necessary to deploy remote probes.
• Exporting flows towards a central location is not always possible:
  – Limited bandwidth available.
  – Need to have a separate/secure network/tunnel as flows contain sensitive data.
  – Interference with other network activities.
  – Export of raw flows is much more costly than exporting the metrics we’re interested in.
Remote Probe Deployment [2/2]

- Exporting data on off-peak times is not an option:
  - We would introduce latency in data consumption.
  - The amount of data to transfer is not significantly reduced (zip flows) with respect to live data export.
  - Unable to use the system for near-realtime analysis and alarm generation.

- Better solution
  - Create a web service for querying data remotely in realtime
  - Export aggregated metrics (e.g. .pkl files)
Web Interface: Internals [1/3]

Observation Period (5 min)

Components Communication via Ajax/jQuery

Google Maps

Python Pickle (Historical)
Web Interface: Internals [2/3]

RRD Charts
(Data Context host/time via jQuery)
### Live FastBit Query+Aggregation

**Python Glue Software**

**Distance: 1**

<table>
<thead>
<tr>
<th>ASN</th>
<th>AS Name</th>
<th>Traffic</th>
<th>Flows</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>2597</td>
<td>REGISTRO CCTLD IT</td>
<td>1.0 GB</td>
<td>100.0%</td>
<td>1367295</td>
</tr>
</tbody>
</table>

**Distance: 2**

<table>
<thead>
<tr>
<th>ASN</th>
<th>AS Name</th>
<th>Traffic</th>
<th>Flows</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>3356</td>
<td>Level 3 Communications, LLC</td>
<td>358.3 MB</td>
<td>633432</td>
<td></td>
</tr>
<tr>
<td>137</td>
<td>GARR Italian academic and research network</td>
<td>12.0 MB</td>
<td>6951</td>
<td></td>
</tr>
<tr>
<td>12637</td>
<td>Seeweb Srl</td>
<td>3.1 MB</td>
<td>2949</td>
<td></td>
</tr>
<tr>
<td>21056</td>
<td>Welcome Italia S.p.A.</td>
<td>468.0 KB</td>
<td>1183</td>
<td></td>
</tr>
<tr>
<td>21309</td>
<td>ACANTHO SPA</td>
<td>434.8 KB</td>
<td>284</td>
<td></td>
</tr>
<tr>
<td>64862</td>
<td>??</td>
<td>216.0 KB</td>
<td>270</td>
<td></td>
</tr>
<tr>
<td>16004</td>
<td>MIX S.r.L.</td>
<td>9.3 KB</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>15469</td>
<td>Warinet NOC AS</td>
<td>365.0 bytes</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Distance: 3**

<table>
<thead>
<tr>
<th>ASN</th>
<th>AS Name</th>
<th>Traffic</th>
<th>Flows</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>9035</td>
<td>Wind Telecomunicazioni spa</td>
<td>35.5 MB</td>
<td>34395</td>
<td></td>
</tr>
<tr>
<td>702</td>
<td>UUNET - Commercial IP service provider in Europe</td>
<td>25.9 MB</td>
<td>31974</td>
<td></td>
</tr>
<tr>
<td>6762</td>
<td>Telecom Italia international high speed,</td>
<td>21.6 MB</td>
<td>36706</td>
<td></td>
</tr>
<tr>
<td>3549</td>
<td>Global Crossing</td>
<td>17.7 MB</td>
<td>22630</td>
<td></td>
</tr>
<tr>
<td>24940</td>
<td>Hetzner Online AG RZ-Nuernberg</td>
<td>16.0 MB</td>
<td>8078</td>
<td></td>
</tr>
<tr>
<td>6453</td>
<td>Teleglobe Inc.</td>
<td>15.6 MB</td>
<td>22171</td>
<td></td>
</tr>
<tr>
<td>12956</td>
<td>Telefonica Data Autonomous System</td>
<td>12.4 MB</td>
<td>28594</td>
<td></td>
</tr>
<tr>
<td>1239</td>
<td>Sprint</td>
<td>11.2 MB</td>
<td>17075</td>
<td></td>
</tr>
</tbody>
</table>
Using Geolocation Data [1/2]
Using Geolocation Data [2/2]
Disk and Memory Usage

• Collection of ~5k flows netflow/sec
• Each 5 min partition takes ~150 MB in FastBit format (32 GB/day)
• Partitions with raw data stay 3 days on disk (limited by available disk space)
• Each tuple archive in pickle format takes up to 400 KB (112 MB in total, almost constant).
• BGP patricia tree (inside the probe) of all routing tables takes about ~100 MB