

# Network Traffic and Security Monitoring Using ntopng and InfluxDB

Luca Deri <deri@ntop.org>  
@lucaderi

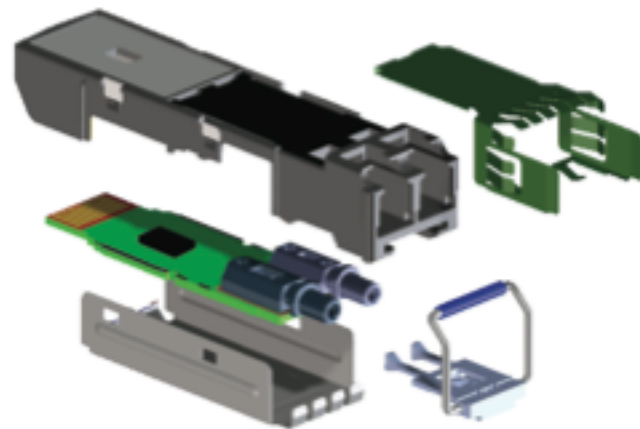
# Part I: Welcome to ntopng

# About ntop.org [1/2]

- ntop develops of open source network traffic monitoring applications.
- ntop (circa 1998) is the first app we released and it is a web-based network monitoring application.
- Today our products range from traffic monitoring, high-speed packet processing, deep-packet inspection, and IDS/IPS acceleration (bro and suricata).

# About ntop.org [2/2]

- Our software is powering many commercial products...



Integrated ASIC with JDSU technology



# ntop Goals

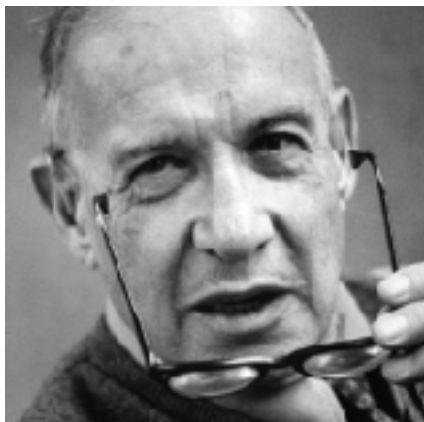
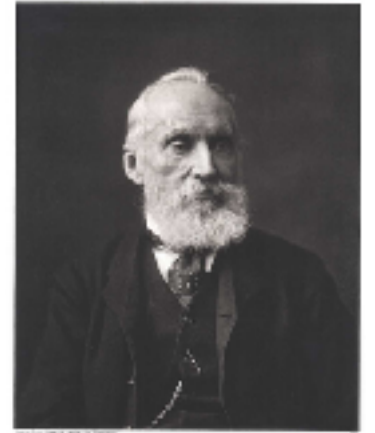
- Provide better, yet price effective, traffic monitoring solution by enabling users to have increased traffic visibility.
- Go beyond standard metrics and increase traffic visibility by analysing key protocols in detail.
- Promote open-source software, while protecting selected IPRs.
- All commercial ntop tools are available at no cost for research and education.

# ntop's Approach to Traffic Monitoring

- Ability to capture, process and (optionally) transmit traffic at line rate, any packet size.
- Leverage on modern multi-core/NUMA architectures in order to promote scalability.
- Use commodity hardware for producing affordable, long-living (no vendor lock), scalable (use new hardware by the time it is becoming available) monitoring solutions.
- Use open-source to spread the software, and let the community test it on unchartered places.

# Motivation For Traffic Monitoring

If you can't measure it, you can't improve it  
(Lord Kelvin, 1824 – 1907)



If you can't measure it, you can't manage it  
(Peter Drucker, 1909 – 2005)

# What Happens in Our Network?

- Do we have control over our network?
- It's not possible to imagine a healthy network without a clear understanding of traffic flowing on our network.
- Knowledge is the first step towards evaluation of potential network security issues.
- Event correlation can provide us timely information about our network health.



# Packets Never Lie

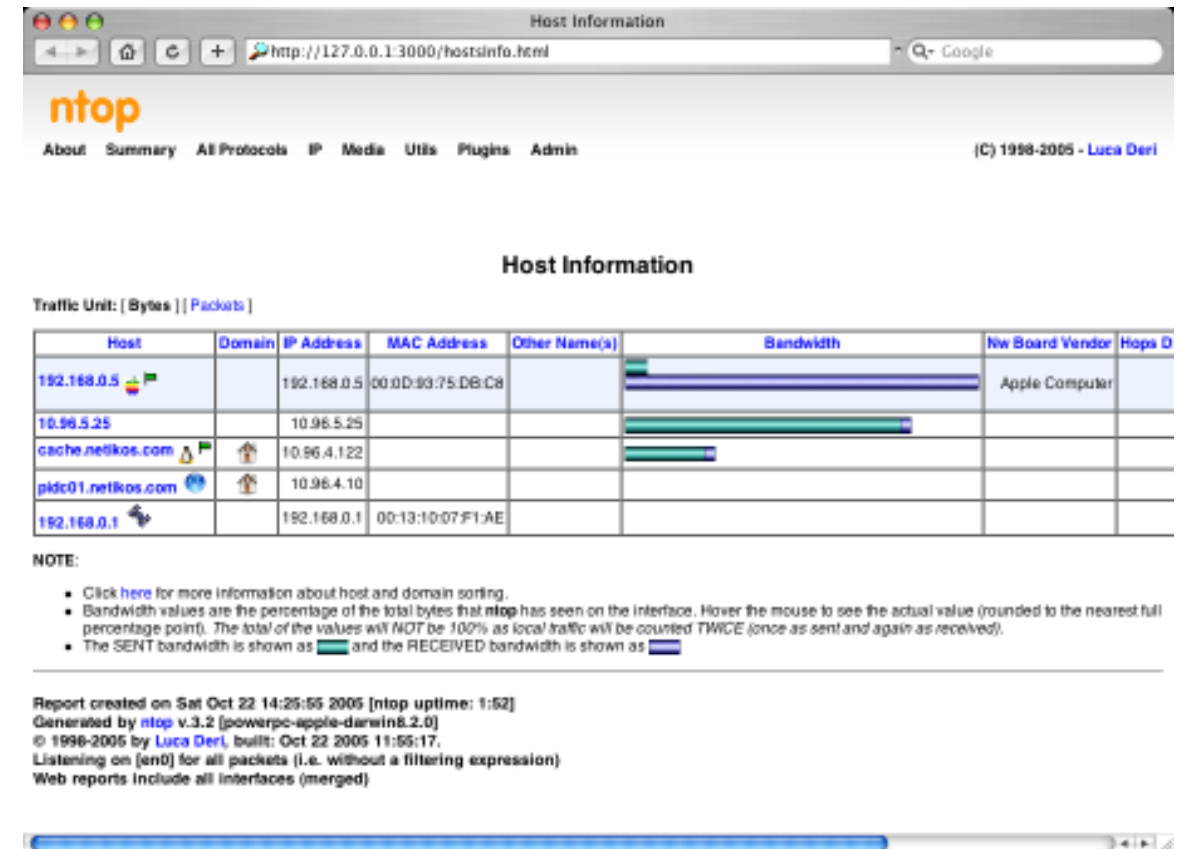
- Packet analysis provide useful information for understanding:
  - Network traffic issues.
  - Network usage not compliant with network policies (note: firewalls cannot help here).
  - Performances less than expected.
  - Potential security flaws.
  - Ongoing (latent) attacks.
  - Data breach.
- But... packets are too fine grained so we need to aggregate them into flows (5 tuple IP/port src/dst, protocol).

# How To Install ntopng

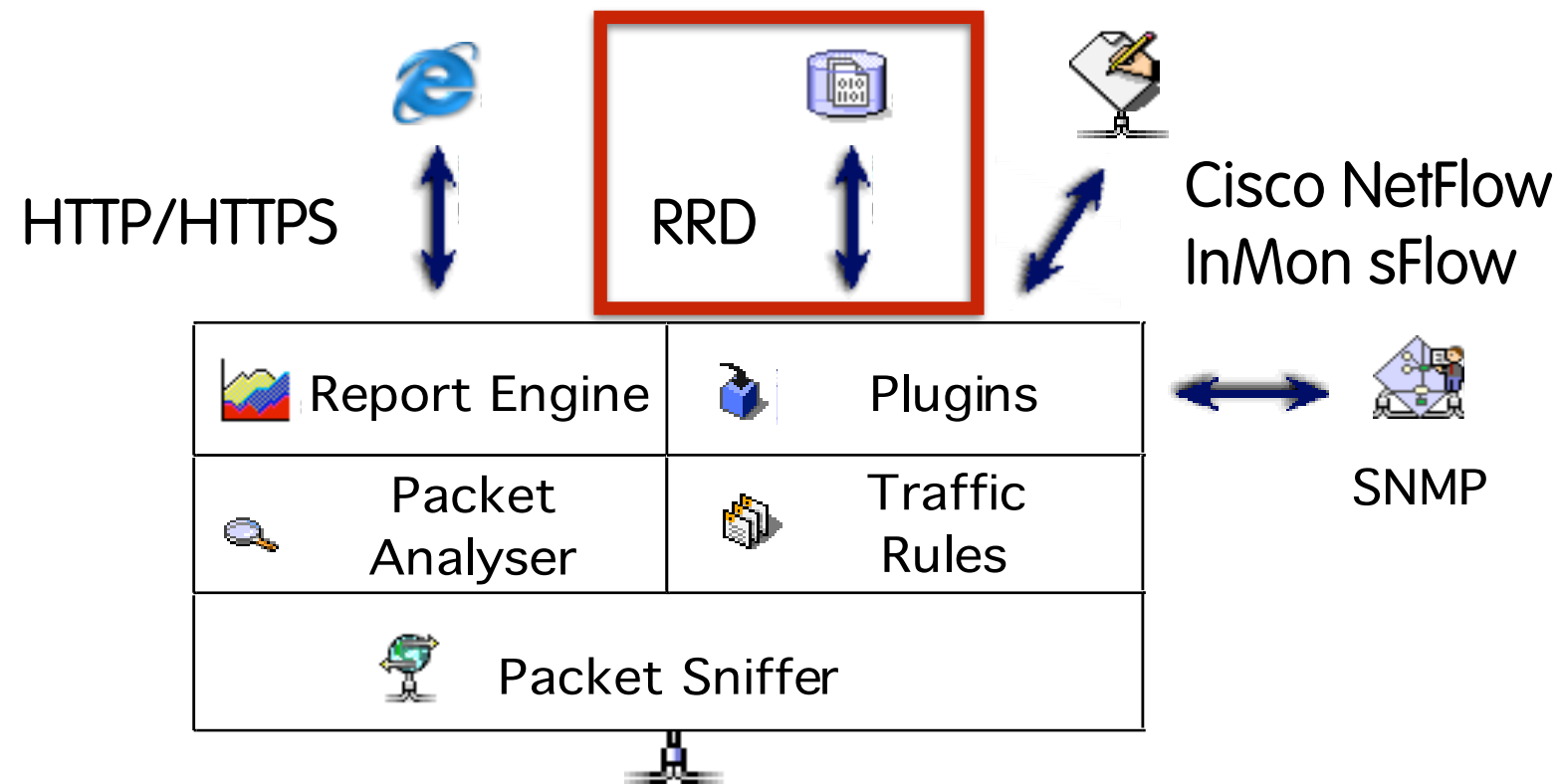
- Source code  
<https://github.com/ntop/ntopng>
- Distributions  
Ubuntu/Debian, FreeBSD.... (included in the distro)  
OSX (brew)
- Binary Packages (nightly + stable)  
<http://packages.ntop.org> (Debian/Ubuntu/CentOS,  
OSX, RaspberryPI/ARM)

# Some History

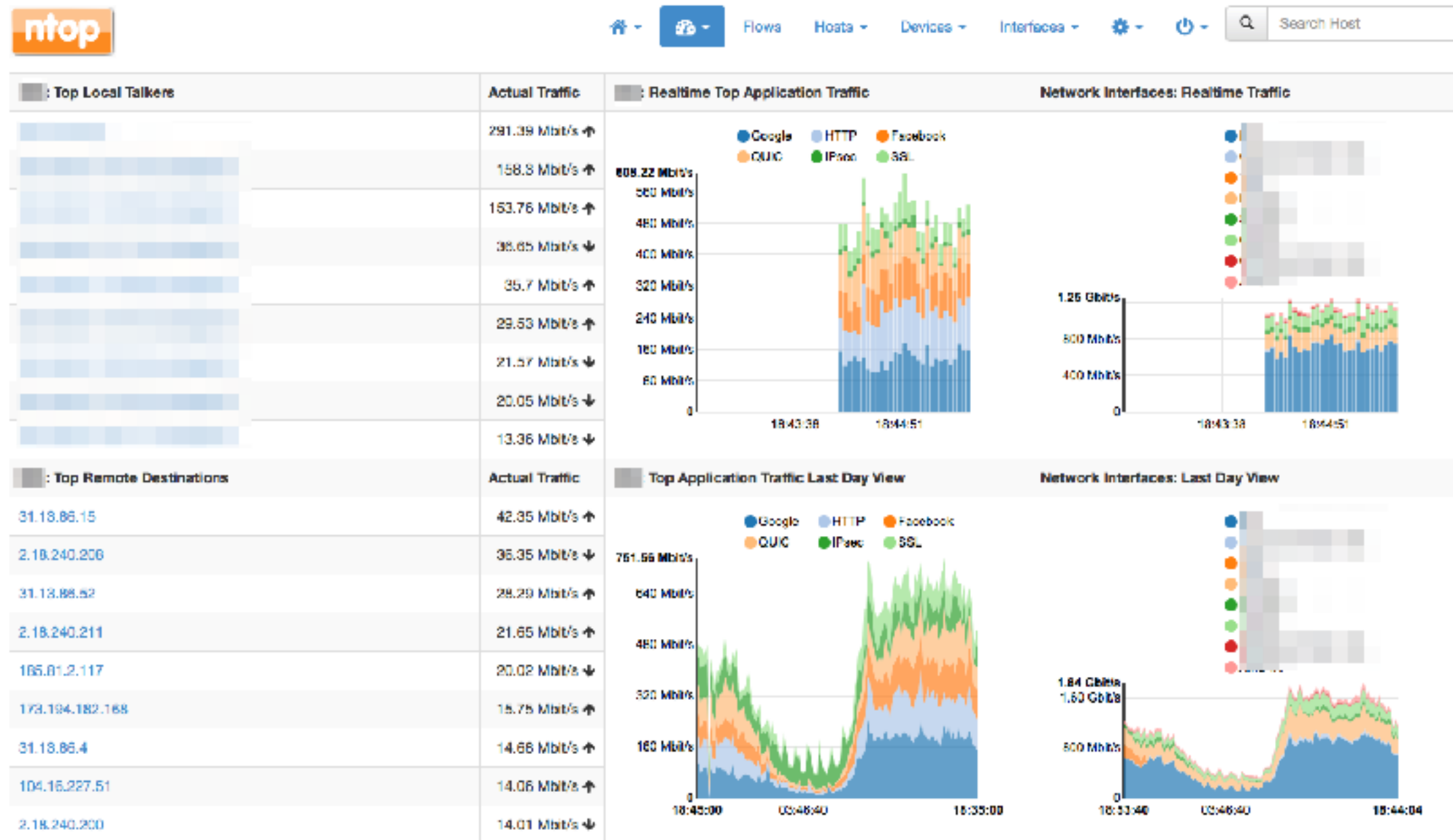
- In 1998, the original ntop has been created.
- It was a C-based app embedding a web server able to capture traffic and analyse it.
- Contrary to many tools available at that time, ntop used a web GUI to report traffic activities.
- It is available for Unix and Windows under GPL.



# ntop Architecture



# Welcome to ntopng



ntopng Enterprise Edition v3.1.171006  
User [admin](#) Interface [NIX](#)

73%  
726.46 Mbit/s [90.175 pps]

32.91 Mbit/s  
600.58 Mbit/s

© 18:45:36 +0200 | Uptime: 23 h, 6 min, 36 sec  
2,809 6,894 196 Devices 65,502 Flows

# ntopng Walk Through: Interpreting Data

Layer 4 Protocol

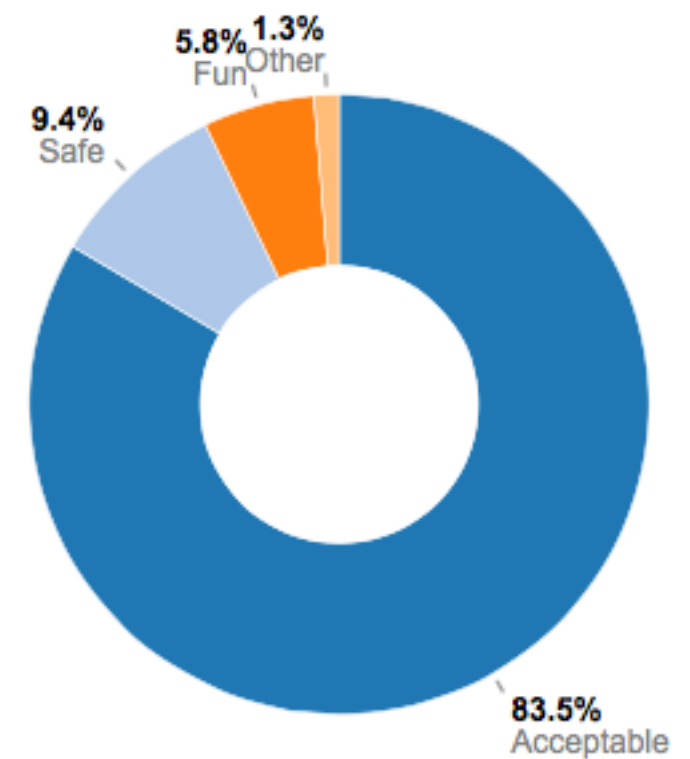
Good or Bad?

Protocol

TCP / HTTP



Layer 7 Protocol



# ntopng Walk Through: Alerts

Open Issues

Past Issues

Flow Issues

Engaged Alerts Past Alerts Flow Alerts

## Engaged Alerts

Who

10

Date/Time	Duration	Severity	Alert Type	Description
Sat May 6 13:03:03 2017	2 min, 4 sec	Error	Threshold Cross	Threshold <b>active</b> crossed by host [65 > 1]

Showing 1 to 1 of 1 rows

When








How Long

What














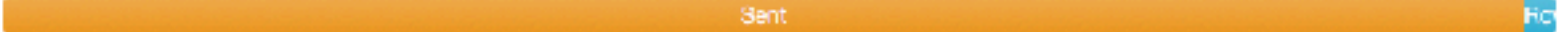
# ntopng Walk Through: Discovery











## Network Discovery

Last Network Discovery 14/04/2018 10:55:48						
IP Address	Name	Manufacturer	MAC Address	OS	Info	Device
.1		Juniper Networks	F4:B5:2F:FC:AF:F0			(OpenSSH_6.4)
.102		Hewlett Packard	B4:B5:2F:C9:69:7A			
.104		Hewlett Packard	44:1E:A1:30:30:3D			
.105		Hewlett Packard	B4:B5:2F:CC:C4:6A			
.108		Hewlett Packard	10:60:4B:6D:81:6D			
.11		Dell Inc.	84:00:6A:63:35:CC			(Ubuntu)
.124		Quanta Computer Inc.	00:23:6B:42:88:37			(Linux)
.125		Juniper Networks	88:A2:5E:E6:BB:01			sw-p1-a-2.
.126		Juniper Networks	5C:45:27:3D:14:01			(switch-p0-m-1)
.13		Apple, Inc.	A8:60:36:00:4A:99			
.2		Quanta Computer Inc.	00:1E:68:2F:11:6D			(Linux)
.22		PCS Computer Systems GmbH	08:00:27:F4:C8:01			(Windows)
.27		Apple, Inc.	3C:07:54:2F:67:65		_ssh._top.local _sftp-ssh._top.local _rfb._top.local	iMac 'Core i7' 3.4 27-Inch (Mid-2011)
.3		Realtek (UpTech? also reported)	52:54:00:E6:BB:FE			(Ubuntu)
.31		Dell Inc.	BC:30:6B:8C:15:02			
.32		Apple, Inc.	88:5B:35:97:BB:22		_ssh._top.local _sftp-ssh._top.local _companion-link._top.local _teamviewer._top.local	iMac 'Core i7' 3.5 27-Inch (Late 2013)



# ntopng Walk Through: Drill Down

Mac: B4:B5:2F:C9:5B:3B   		
MAC Address	B4:B5:2F:C9:5B:3B (HewlettP_C9:5B:3B) [ <a href="#">Show Hosts</a> ] 	 Router/Switch 
Name	B4:B5:2F:C9:5B:3B 	Host Pool: Not Assigned 
Device Type	 Router/Switch	
<a href="#">DHCP Fingerprint</a> 	0103063633	Operating System: 
First / Last Seen	10/10/2017 12:26:06 [11 hours, 1 min, 16 sec ago]	10/10/2017 23:26:37 [45 sec ago]
Sent vs Received Traffic Breakdown		
Traffic Sent / Received	95 Pkts / 14.77 KB	3 Pkts / 279.00 Bytes
<a href="#">Address Resolution Protocol</a>	ARP Requests	ARP Replies
	0 Sent / 0 Received	2 Sent / 0 Received

(Router/AccessPoint) MAC Address	Dell_63:35:CC ( 64:00:6A:63:35:CC )	 Computer 
IP Address	 [ 192.12.193.0/25 ] [ Pisa  ]	Host Pool: Not Assigned 
OS	 Linux x86_64	
Name	  Local Host  System IP 	
First / Last Seen	10/10/2017 12:26:03 [11 hours, 3 min, 49 sec ago]	10/10/2017 23:29:50 [2 sec ago]

# ntopng Walk Through: Monitoring

<b>SSL Certificate</b>	Client Requested: <a href="https://luca.ntop.org">luca.ntop.org</a>	Server Certificate: <a href="https://shop.ntop.org">shop.ntop.org</a> <b>Certificates don't match</b>
<b>Max (Estimated) TCP Throughput</b>	Client → Server: 91.57 Kbit	Client ← Server: 1.49 Mbit
<b>TCP Flags</b>	Client → Server: FIN SYN PUSH ACK	Client ← Server: FIN SYN PUSH ACK
<b>Flow Status</b>	This flow is completed and will expire soon. <b>SSL Certificate Mismatch</b>	

Invalid Configuration or Threat ?

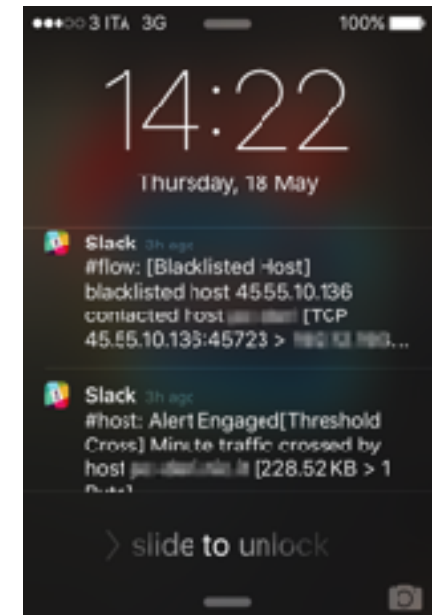
Service Down or Scan?

ICMP Message	Packets Sent	Last Sent Peer	Packets Received	Last Rcvd Peer	Breakdown	Total
<b>Destination Port Unreachable</b>	103 Pkts	[IP Address]	3 Pkts	[IP Address]	Sent	106 Pkts
Echo Request	0 Pkts		1 Pkts	[IP Address]	Rcvd	1 Pkts
Echo Reply	1 Pkts	[IP Address]	0 Pkts		Sent	1 Pkts

# Yes You Can

- Embedded alerting system pluggable with nagios and messaging systems.

- Use it as Grafana datasource



- Ready for     

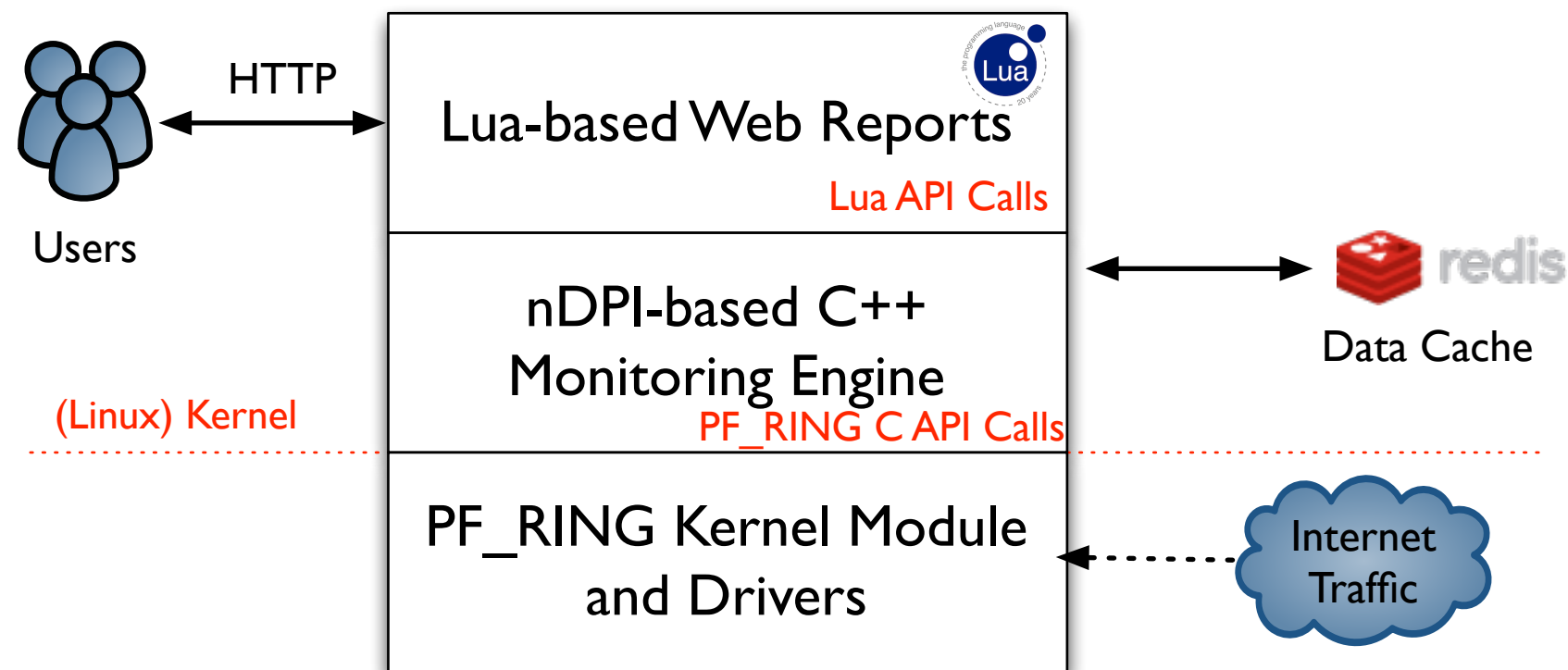
- nDPI: passive mode = monitoring, inline = IPS
- Support for NetFlow/sFlow/SNMP.
- Passive/Active Network Device Discovery.
- Traffic Behaviour Analysis.

# ntopng Design Goals

- Clean separation between the monitoring engine and the reporting facilities.
- Robust, crash-free engine.
- Platform scriptability for enabling extensions or changes at runtime without restart.
- Realtime: most monitoring tools aggregate data (5 mins usually) and present it when it's too late.
- Many new features including HTML 5-based dynamic GUI, categorisation, DPI.

# ntopng Architecture

- Three different and self-contained components, communicating with clean API calls.



# ntopng Monitoring Engine

- Coded in C++ and based the concept of flow (set of packets with the same 6-tuple).
- Flows are inspected with a home-grown DPI-library named nDPI aiming to discover the “real” application protocol (no ports are used).
- Information is clustered per:
  - (Capture) Network Device
  - Flow
  - Host

# Information Lifecycle

- ntopng keeps in memory live information such as flows and hosts statistics.
- As the memory cannot be infinite, periodically non-recent information is harvested.
- Users can specify preferences for data purge:



The screenshot shows the ntopng web interface for configuring data purge settings. On the left is a sidebar with navigation links: 'Manage Users' (with a user icon), 'Preferences' (with a flask icon), and 'Export Data' (with a circular arrow icon). The main content area is titled 'Data Purge' and contains three settings, each with a text input field and a 'Save' button:

Setting	Value	Action
<b>Local Host Idle Timeout</b> <small>Inactivity time after which a local host is considered idle (sec). Default: 300.</small>	300	Save
<b>Remote Host Idle Timeout</b> <small>Inactivity time after which a remote host is considered idle (sec). Default: 60.</small>	60	Save
<b>Flow Idle Timeout</b> <small>Inactivity time after which a flow is considered idle (sec). Default: 60.</small>	60	Save

# Packet Processing Journey

1. Packet capture: PF\_RING (Linux) or libpcap.
2. Packet decoding: no IP traffic is accounted.
3. IPv4/v6 Traffic only:
  - Map the packet to a 6-tuple flow and increment stats.
  - Identify source/destination hosts and increment stats.
  - Use nDPI to identify the flow application protocol
    - UDP flows are identified in no more than 2 packets.
    - TCP Flows can be identified in up to 15 packets in total, otherwise the flow is marked as “Unknown”.
4. Move to the next packet.



# The need for DPI in Monitoring [1/2]

- Limit traffic analysis at packet header level it is no longer enough (nor cool).
- Network administrators want to know the real protocol without relying on the port being used.
- Selected protocols can be “precisely dissected” (e.g. HTTP) in order to extract information, but on the rest of the traffic it is necessary to tell network administrators what is the protocol flowing in their network.

# The need for DPI in Monitoring [2/2]

- DPI (Deep Packet Inspection) is a technique for inspecting the packet payload for the purpose of extracting metadata (e.g. protocol).
- There are many DPI toolkits available but they are not what we looked for as:
  - They are proprietary (you need to sign an NDA to use them), and costly for both purchase and maintenance.
  - Adding a new protocol requires vendor support (i.e. it has a high cost and might need time until the vendor supports it) = you're locked-in.
- On a nutshell DPI is a requirement but the market does not offer an alternative for open-source.

# Say hello to nDPI



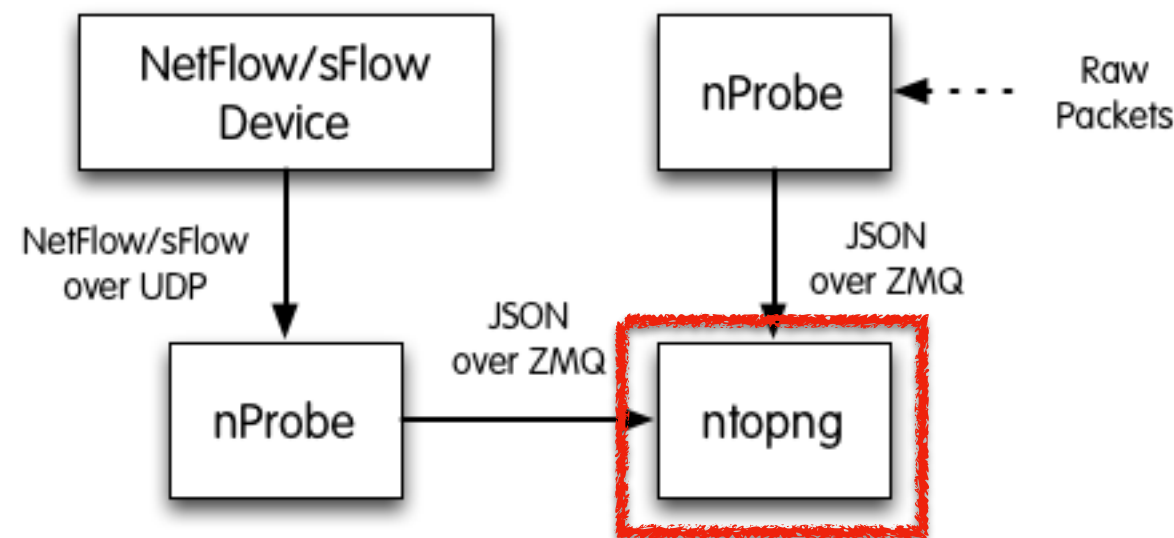
- ntop has decided to develop its own GPL DPI toolkit in order to build an open DPI layer for ntop and third party applications.
- Supported protocols (> 240) include:
  - P2P (Skype, BitTorrent)
  - Messaging (Viber, Whatsapp, MSN, The Facebook)
  - Multimedia (YouTube, Last.fm, iTunes)
  - Conferencing (Webex, CitrixOnline)
  - Streaming (Zattoo, Icecast, Shoutcast, Netflix)
  - Business (VNC, RDP, Citrix, \*SQL)

# nDPI on ntopng

- In ntopng all flows are analysed through nDPI to associate an application protocol to them.
- L7 statistics are available per flow, host, and interface (from which monitoring data is received).
- For network interfaces and local hosts, nDPI statistics are saved persistently to disk.






# ntopng as a NetFlow/sFlow Collector

- ntopng can collect traffic information sent by routers (NetFlow) and switches (sFlow).
- Flow information needs binary -> JSON translation as we designed ntopng agnostic with respect to flow low-level details.



# SNMP and Flow Collection [1/2]

- ntopng allows SNMP MIBs to be queried (MIB-II and Bridge MIB)







Description												
ObjectID		1.3.6.1.4.1.8072.3.2.10										
Uptime		1 day, 17 h, 58 min, 14 sec										
Contact		Me										
Name												
Location		Sitting on the Dock of the Bay										
Interface Index	Chart	Name	Type	MTU	Speed	Mac Address	Status	In Bytes	Out Bytes	In Discards	Port MACs	Last Change
1		lo	softwareLoopback		10 Mbit		Up	1.17 GB	1.17 GB			
2		Intel Corporation Ethem...	ethernetCsmacd	1500	1 Gbit	64:00:6A:83:35:CC	Up	269.12 MB	1.51 GB			
3	-	Intel Corporation 82540E...	ethernetCsmacd	1500		00:0E:0C:2C:0B:B4	Down					
4	-	docker0	ethernetCsmacd	1500		02:42:97:B3:2F:EA	Down					
5		as0t0	other	1500			Up	0 B	832 B			
6		virbr0	ethernetCsmacd	1500		FE:54:00:57:7C:58	Up	1.24 GB	1.35 GB			
7	-	virbr0-nic	ethernetCsmacd	1500		52:54:00:25:C9:8B	Down					
11		vnet0	ethernetCsmacd	1500	10 Mbit	FE:54:00:57:7C:58	Up	1.29 GB	1.37 GB			

# SNMP and Flow Collection [2/2]

- Both NetFlow and sFlow can be glued to SNMP through the device IP + Interface Id
- All necessary is to do, is configure in the SNMP menu the IP address and community of the SNMP-enabled devices

## SNMP Devices

10 ▾

Device IP	Chart	Device Name	Device Model	Description	Location	Actions
						Delete
						Delete
						Delete
						Delete
						Delete
						Delete

Showing 1 to 6 of 6 rows

[\[ Add New Device \]](#)

# ntopng and Big Data [1/2]

- Using a local SQL database to save flows persistently is good when flows are not too many and the system that runs ntopng has storage.
- For large deployments or disk-less systems (e.g. ARM-based PCs) it is desirable to upload flows on remote, cloud-based, systems able to scale with the number of flows.
- In essence ntopng has been opened to what is currently defined as “big data” systems that can scale with data in volume and speed.



# ntopng and Big Data [2/2]

- Currently ntopng is able to export monitoring information in JSON format towards various systems including
  - ElasticSearch / Logstash
  - ZMQ
- In essence ntopng is a sort of gateway that can collect, self-produce (from packets), and export monitoring information by normalising it in format (JSON).

# Part II: Network Monitoring and TimeSeries

# Monitoring Granularity

- Historically network devices such as routers and switches produce monitoring information based on the traffic that traverses such devices.
- Various techniques are used to avoid exhausting network device resources, in particular CPU and memory, including:
  - Packet sampling (i.e. consider one packet in  $N$  and not all traffic).
  - Limit counter polling. For instance SNMP device counters are updated every  $X$  seconds (e.g. 3 sec), so polling them too fast won't help to produce fine grained data.

# Average Values As Default Option [1/2]

- Most monitoring protocols have been designed to produce average data.
- If you wish limited SNMP polling frequency produces average data.
- NetFlow, the leading monitoring protocols for Internet traffic is even worse. Default flow duration is set in minutes, thus making measurements even less granular.

# Average Values As Default Option [2/2]

## Netflow RFC

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

SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	162	/24	5	163	/24	15	10.0.23.2	1528	1745	4
Fa1/0	173.100.3.2	Fa0/0	10.0.227.12	6	40	0	2491	15	/26	196	15	/24	15	10.0.23.2	740	41.5	1
Fa1/0	173.100.20.2	Fa0/0	10.0.227.12	11	80	10	10000	161	/24	180	16	/24	15	10.0.23.2	1428	1145.5	3
Fa1/0	173.100.6.2	Fa0/0	10.0.227.12	6	40	0	2210	19	/30	180	19	/24	15	10.0.23.2	1040	24.5	14

### 2. Flow Aging Timers

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

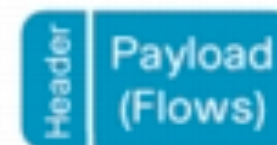
SrcIf	SrcIPadd	DstIf	DstIPadd	Protocol	TOS	Flgs	Pkts	Src Port	Src Msk	Src AS	Dst Port	Dst Msk	Dst AS	NextHop	Bytes/Pkt	Active	Idle
Fa1/0	173.100.21.2	Fa0/0	10.0.227.12	11	80	10	11000	00A2	/24	5	00A2	/24	15	10.0.23.2	1528	1800	4

### 3. Flows Packaged in Export Packet

Non-Aggregated Flows—Export Version 5 or 9

### 4. Transport Flows to Reporting Server

Export Packet



# ntopng Realtime Measurement

- The ntopng engine has been designed for realtime traffic measurement. It can report packet-based information in realtime and the engine can be polled continuously while processing traffic.
- While the user interface can be refreshed from 1-5 sec via Ajax/WebSocket, writing time series to disk can be difficult when the number of metrics is high in cardinality.

# Metrics Cardinality: SNMP

- Each interface has 5 counters (ifIn/outPackets, ifIn/outBytes, ifInErrors).
- Switches can have a few ports (e.g. 24) or 500+ ports on core switches.
- SMEs can have a few (e.g. 4) switches, while ISP can have hundred of them.
- In summary ntopng has been designed to poll thousand ports across multiple switches. For this reason, unless sFlow is used, it is not possible to implement sub-minute polling (small networks) or sub-5 minute on larger networks in order to avoid putting too much load on devices or producing too much network traffic due to SNMP.

# Metrics Cardinality: Internet Traffic [1/3]

- Usually traffic counters are kept one for local hosts (i.e. those that belong to the company being monitored).
- SMEs can have as low as 20 hosts (today this is the typical number of connected devices of a home, when considering IoT, tables, phones etc) or thousand of them.
- Even though nDPI supports ~240 protocols, most hosts do not use all of them, and usually they are limited to about 30 protocols/host.



# Metrics Cardinality: Internet Traffic [2/3]

- In total a host has:
  - 4 counters in/out packets + in/out bytes per nDPI protocol (so in average  $4 \times 30 = 120$ )
  - About 20 counters for other metrics such as retransmission, packets-out-of-order,...
  - In total in average  $120+20$  counters/host.
- In addition we need to add counters for additional elements such as visited autonomous systems, networks, countries etc.

# Metrics Cardinality: Internet Traffic [3/3]

- In summary on a network with 254 hosts (a /24 CIDR) we have ~35-50k counters to save every minute.
- Further challenges:
  - The number of these counters changes overtime (e.g. when a host is disconnected, no traffic is reported, so no counters are produced).
  - Counters granularity is not homogeneous: interface counters are per second, host per minute, SNMP every 5 mins.
  - It would be desirable to have more fine grained counters (a value every 5/10 sec) but this can significantly increase the complexity of our problem.

# ntopng and Timeseries [1/4]

- ntopng has an embedded “crontab” that executes tasks:
  - Per system (singleton), monitored interface (physical 1-3, logical up to 128).
  - Every second, minute, 5 minute, hour, day.
- Tasks
  - Second: Interface counters in/out packets/bytes, drops, packets out-of-order....
  - 5 Minutes: hosts and SNMP counters.

# ntopng and Timeseries [2/4]

- RRD-based system used for historical reasons (ntop used it) and because it is file based (pro) that allows us to run it on the same host where ntopng runs with no external service dependency.
- Cons:
  - Unable to cope with large number of hosts (5 mins are not enough) so we are unable to save all the data to disk.
  - High load on the filesystem in particular on low-end/embedded devices that feature slow disks.

# ntopng and Timeseries [3/4]

- RRD cons (cont).
  - ...old programming library designed as a tool rather than a library: every function call uses argc/argv.
  - Thread support works but it is mostly a hack.
  - Too many library dependencies even if ntopng uses it as time series database with no graph generation.
  - The library is not moving forward since a long time (5+ years) in terms of new developments and we do not expect this to change anytime soon. In essence this is at the moment a project in maintenance mode.

# ntopng and Timeseries [4/4]

- Beside all these limitation in RRD design, the main driving force for replacing it in ntopng are:
  - Inability to easily compare timeseries
    - Select all hosts that in the past hour has made 20% up of traffic with respect to the previous hour.
    - Trigger an alert when host X is sending/receiving double the traffic of the second top-sender in the last hour.
  - Inability to handle long-term measurements without aggregation and “native” loss of precision (RRD basically implements always an average and if you insert 1 you will extract 0.9999999999 that can be annoying sometimes).

# What Alternatives to RRD? [1/2]

- Since a few years, timeseries databases are becoming popular, and there are many interesting projects.
- In 2014 ntop started to evaluate InfluxDB 0.x but we decided to wait a bit longer before integrating it into the project as it was not mature enough (the engine was under heavy changes, we had data loss during developments).
- In 2017, we decided that RRD was becoming a real bottleneck to ntopng evolution so we started to find a solution to this problem.

# What Alternatives to RRD? [2/2]

- Out to the many timeseries databases we believe that the only two real options are InfluxDB and Prometheus, otherwise better to stay on RRD.
- ntopng currently supports both of them for data dump, but we have decided to bet in InfluxDB for a few reasons:
  - Ecosystem (Chronograf....), vibrant community and company behind the project.
  - Flexibility: we prefer to push data to Influx, rather than ntopng to be pulled as Prometheus does.
  - Other ntop users use it in various projects and they are happy of performance and stability.
  - Generic DB, not targeting a specific domain such as Prometheus.



# How to Further Improve Influx

- Change the application design in LibInflux + InfluxDB (that uses libInflux and adds an ingest layer).
- Let local applications push data through LibInflux using a C/C++ API:
  - No need of an intermediate text-based format.
  - Less security headaches as there is no need to perform DB authentication or data encryption when pushing data to the database.
  - Less overhead due to HTTP, no data loss when using UDP.

# Influx Migration Implications [1/2]

- Create a timeseries Lua layer inside ntopng to allow people to use both RRD and Influx until the migration is completed.
- Mask RRD/Influx differences during data extraction:
  - RRD has automatic data consolidation/rollup that instead we need to configure in Influx with continuous queries.
  - RRD handles natively counters/gauges whereas on Influx we need to do that in queries.

# Influx Migration Implications [2/2]

- Mask RRD/Influx differences (cont)
  - In RRD we set time the policy and data value boundaries at archive creation: in case our data is out of range, measurements are automatically discarded. In Influx we need to do that during data extraction so applications need to be aware of that.
  - RRD normalises automatically (and sometimes this is not a plus) data at a specific resolution based on the query time-range: with Influx we have more precise measurements that are a good thing, but that have to be properly handled otherwise we end up having too many data points in graphs that will result in slow Ajax-based rendering.

# Migration Progress

- Completed
  - Data export to Influx via HTTP for all counters
- In progress (due this summer)
  - ntopng rework to avoid periodic tasks and thus move from 5-min host counters to 1 minute or less. On small/medium networks we would like to go down to 5/10 seconds.
  - Timeseries Lua library for hiding RRD/InfluxDB differences so that existing ntopng users can migrate to Influx and we can handle seamlessly RRD/InfluxDB in the ntopng GUI (completed data writing, missing reading).

# Final Remarks

- InfluxDB is definitively a step forward with respect to RRD.
- The changes we had to do in our code to handle it were mostly due to the fact that ntopng was not sitting on top of a generic timeseries layer as it assumed RRD was living underneath.
- Removing the inability to monitor large networks with many counters and with low granularity is compulsory, and InfluxDB is definitively adequate for this task.