influxdata®

AN INFLUXDATA CASE STUDY

How ntop Built a Web-Based Traffic Analysis and Flow Collection with InfluxDB

Luca Deri

Founder, ntop

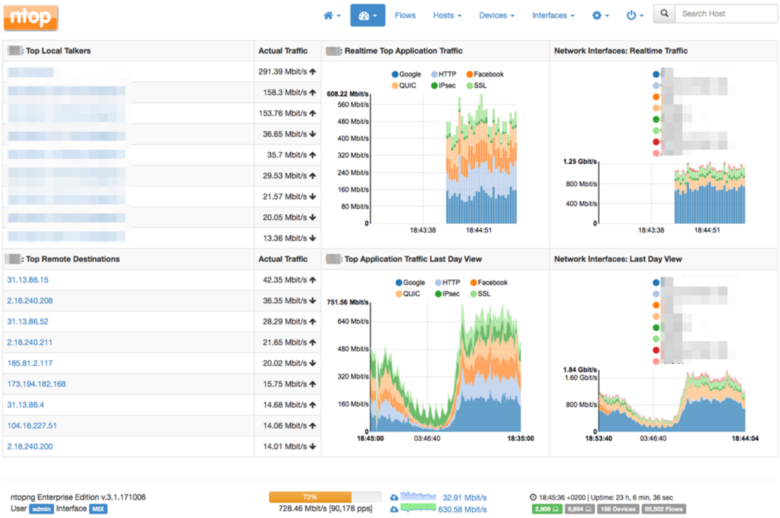
MARCH 2019

Company in brief

ntop develops open source network traffic monitoring applications. It first app, called ntop (circa 1998), is a web- based network monitoring application. ntop's products range from traffic monitoring, high speed packet processing to deep-packet inspection and IDS/IPS acceleration (Bro and Suricata). Its software powers many commercial products. ntop aims to provide better yet cost-effective traffic monitoring; go beyond standard metrics and increase traffic visibility by analyzing key protocols in detail; and promote open-source software while protecting selected IPRs. All commercial ntop tools are available at no cost for research and education. ntopng is the next generation version of the original ntop, a network traffic probe that monitors network usage. ntopng is based on libpcap (Linux) and has been written in a portable way to virtually run on every Unix platform, MacOSX and on Windows. ntopng provides an intuitive, encrypted web user interface for exploring real-time and historical traffic information.

Case overview

ntop wanted to collect real-time network traffic monitoring metrics and events to support ntopng, the new generation version of its network traffic probe "ntop" that monitors network usage. ntop users needed visibility into their networks to understand and control network traffic and health. So in ntopng, the company replaced an RRD database with the open source InfluxDB time series database to build web-based traffic analysis & flow collection and achieve network traffic and security monitoring in real-time.



: Top Local Talkers	Actual Traffic
[redacted]	291.39 Mbit/s ↑
[redacted]	158.3 Mbit/s ↑
[redacted]	153.76 Mbit/s ↑
[redacted]	36.65 Mbit/s ↓
[redacted]	35.7 Mbit/s ↑
[redacted]	29.53 Mbit/s ↑
[redacted]	21.57 Mbit/s ↓
[redacted]	20.05 Mbit/s ↓
[redacted]	13.36 Mbit/s ↓

: Top Remote Destinations	Actual Traffic
31.13.86.15	42.35 Mbit/s ↑
2.18.240.208	36.35 Mbit/s ↓
31.13.86.52	28.29 Mbit/s ↑
2.18.240.211	21.65 Mbit/s ↑
185.81.2.117	20.02 Mbit/s ↓
173.194.182.168	15.75 Mbit/s ↑
31.13.86.4	14.68 Mbit/s ↑
104.16.227.51	14.06 Mbit/s ↑
2.18.240.200	14.01 Mbit/s ↓

ntopng Enterprise Edition v.3.1.171006
User admin Interface MIX
73%
728.46 Mbit/s [90,178 pps]
32.91 Mbit/s
630.58 Mbit/s
18:45:36 +0200 | Uptime: 23 h, 6 min, 36 sec
2,809 6,894 186 Devices 65,502 Flows

ntopng: the real-time network traffic probe using InfluxDB

> *"It's very important today to deliver data with high granularity, because people want to correlate host monitoring with traffic monitoring. We believe we made a good choice jumping to InfluxDB."*
>
> ***Luca Deri,*** *founder*

The business problem

Control over a network begins with visibility. It requires a clear understanding of traffic flowing on a network, and such knowledge is the first step towards evaluating potential network capacity, performance, and even security issues. Further, event correlation can provide timely information about network health.

ntop wanted to give its customers the ability to analyze network traffic in a simple way and to be their source of network monitoring data. Because ntop's users don't need low-level information and often just want to know high-level facts (knowing if the network is fast enough for example), ntop had to translate networking information, such as packet loss, into something meaningful. In particular, ntop needed to understand if something unexpected is happening on the network, who is causing it and how to fix it. ntop's original solution already provided a visualization platform, but they needed flexible and scalable time series functionality to achieve monitoring granularity. ntop wanted to build a solution in line with its approach to network traffic monitoring:

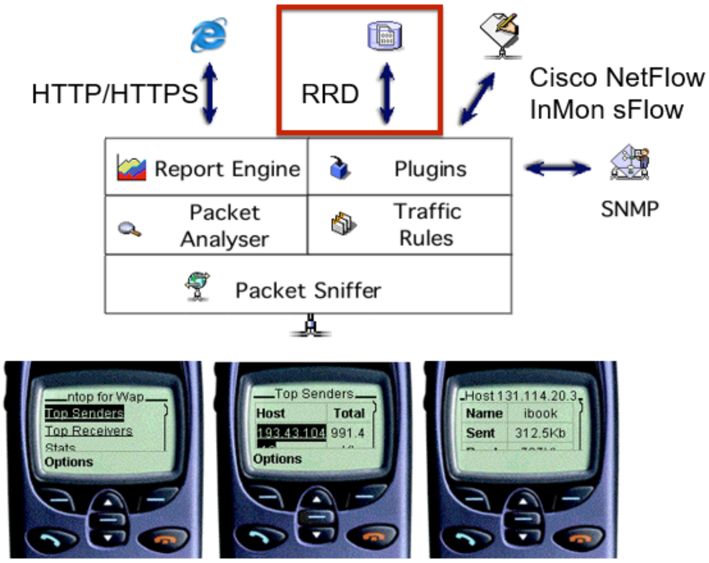
- Ability to handle high volume monitoring data from host and network metrics granularity
- Leverage modern multicore/NUMA architectures in order to promote scalability
- Use commodity hardware for producing affordable, long-living (no vendor lock), scalable (use new hardware by the time it becomes available) monitoring solutions
- Use open source to spread the software and let the community test it on uncharted places

Most monitoring protocols (SNMP and NetFlow) have been designed to produce average data.
While it was once important to speak a certain language, such as SNMP or NetFlow, in the networking world, what matters today isn't really where the data is coming from but how to manipulate it. The format today is generally JSON, and when the data is translated from a specific format to JSON, it can easily be manipulated. Further, with ntopng, ntop wanted to complement existing information with information from network traffic.

The technical problem

To achieve monitoring granularity and stay relevant, ntop had to rethink its application, in which time series data was being stored in an RRD database. The original ntop, created in 1998, was a C-based app embedding a web server able to capture traffic and analyze it. Contrary to many tools available at that time, ntop used a web GUI to report traffic activities.

ntop Architecture



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 and limiting counter polling.

While Packet Analysis provides basic information — for understanding network traffic issues, usage, performance, and security flaws — packets are too fine-grained. So ntop needed to aggregate them into flows (5 tuple IP/port src/dst, protocol). The first step was to move from packets to flows, and to create and commute metrics on these flows.

The company's design goals for ntopng were:

- Clean separation between the monitoring engine and reporting facilities
- Robust, crash-free engine
- Platform scriptability for enabling extensions or changes at runtime without restart

- Real-time (most monitoring tools aggregate data every 5 minutes and present it when it's too late)
- Many new features including HTML 5-based dynamic GUI, categorization, Deep Packet Inspection (DPI)

To store time series data, ntopng had used an RRD-based system for historical reasons (used in the original ntop platform) and because RRD is file-based (pro) which allows it to run on the same host as ntopng with no external service dependency. Yet the RRD design had disadvantages:

- Unable to cope with large number of hosts (5 min aggregation would still generate too much data, forcing one to take larger aggregation windows — not effective for monitoring — or keep number of hosts small), so ntop was unable to save all data to disk
- High load on the filesystem in particular on low-end/embedded devices that feature slow disks
- Old programming library (designed as a tool rather than a library)
- Thread support that works but that is mostly a hack
- Too many library dependencies even if ntopng uses it as a time series database with no graph generation
- A library that has not developed for 5+ years and that has become a project in maintenance mode

Beside the above RRD design limitations, the main driving forces for replacing RRD in ntopng were its inability to easily compare time series and handle long-term measurements without aggregation and "native" loss of precision.

The solution

> *"We decided to adopt InfluxDB for a few reasons. First of all because the ecosystem is very good. So we don't have just the database but we have an ecosystem in the vibrant community."*

Why InfluxDB?

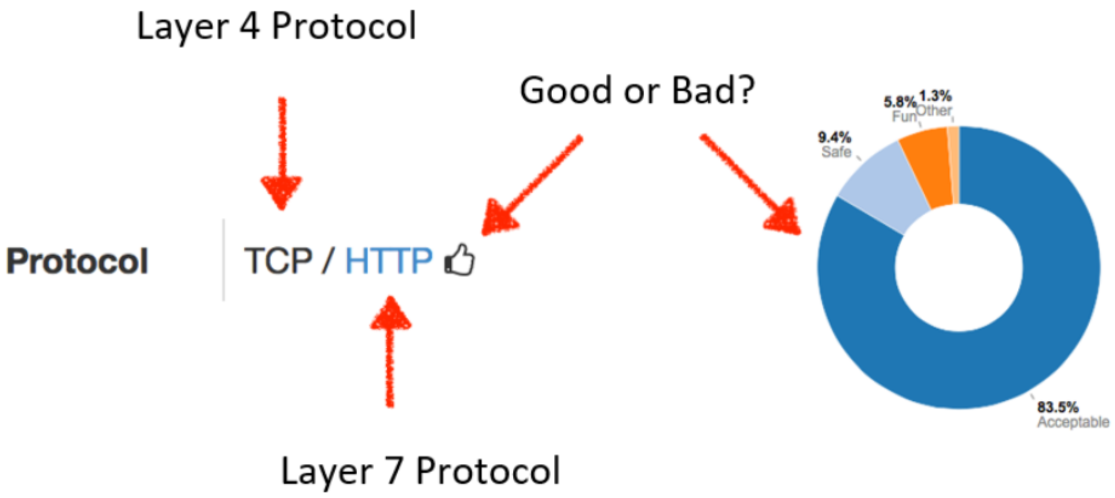
ntop started to evaluate InfluxDB as early as 2014 but decided to wait a bit longer before integrating it into the project. In 2017, ntop decided that RRD was becoming a real bottleneck to ntopng evolution, so after assessing InfluxDB and Prometheus as viable options, they chose InfluxDB for key reasons:

- Ecosystem (the TICK Stack), vibrant community and company behind the project

- Flexibility (ntop preferred to push data to InfluxDB, rather than ntopng to be pulled as Prometheus does)
- Popularity among other ntop users using it in various projects and happy with its performance and stability
- Generic DB, not targeting a specific domain such as Prometheus

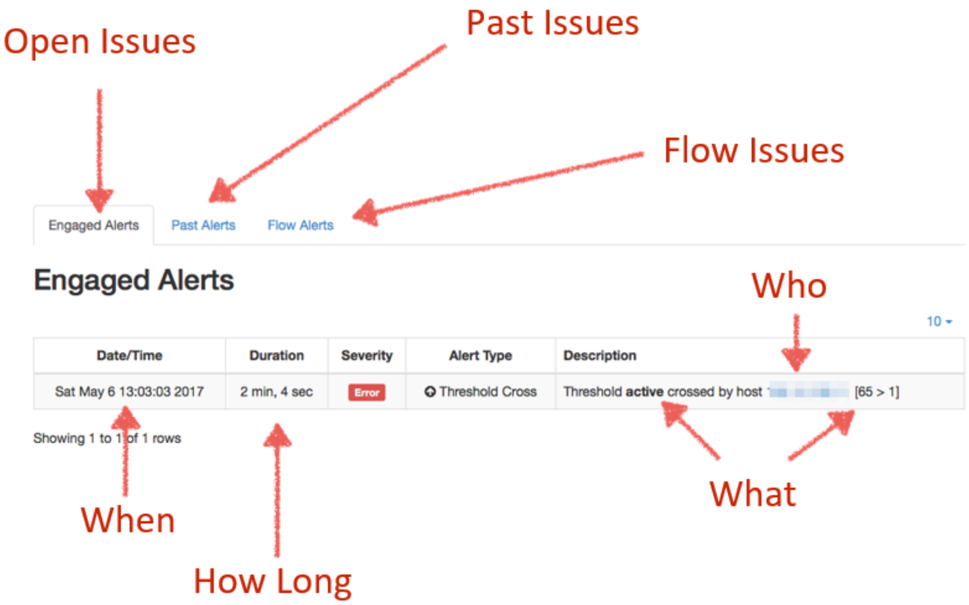
The ntopng engine has been designed for real-time traffic measurement. It can report packet-based information in real-time, and the engine can be polled continuously while processing traffic. ntopng analyzes network traffic, has a web service, and shows network traffic according to values criteria, flows, cost, interfaces and so on. It requires almost no configuration and reports information in-memory. Even users with little experience can view network health with a simple pie chart.

ntopng: Interpreting Data



ntopng also has alert functionality, to alert users when something goes wrong, and provides alert information in context to show why the problem originated. ntopng complements traditional passive discovery (learning from traffic flowing in the network) as well as active discovery (probing the network).

Passive Alerts

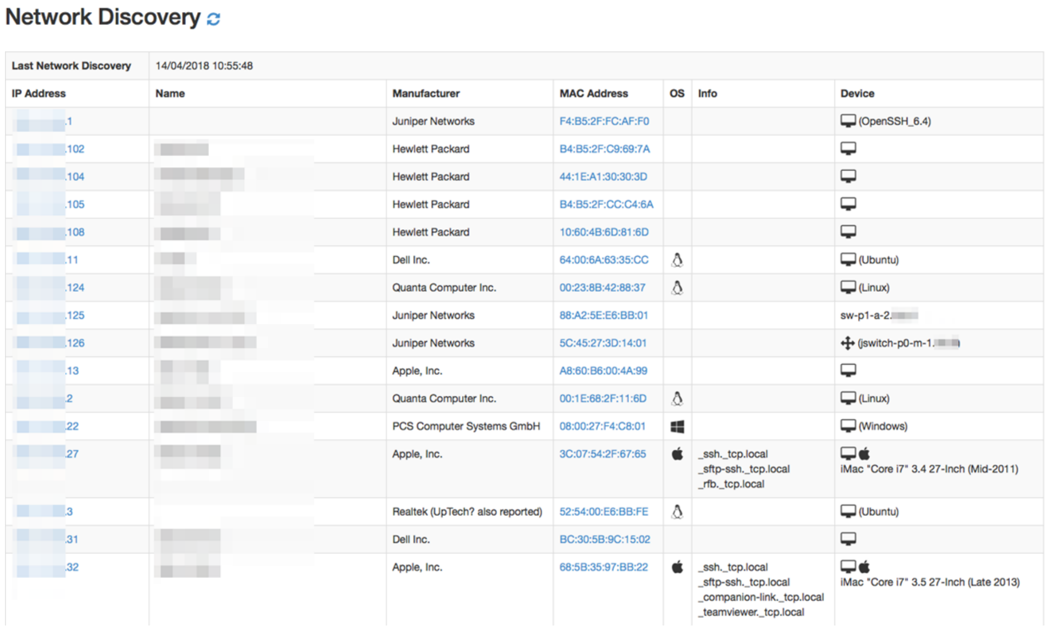


Engaged Alerts

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

Showing 1 to 1 of 1 rows

Active Discovery

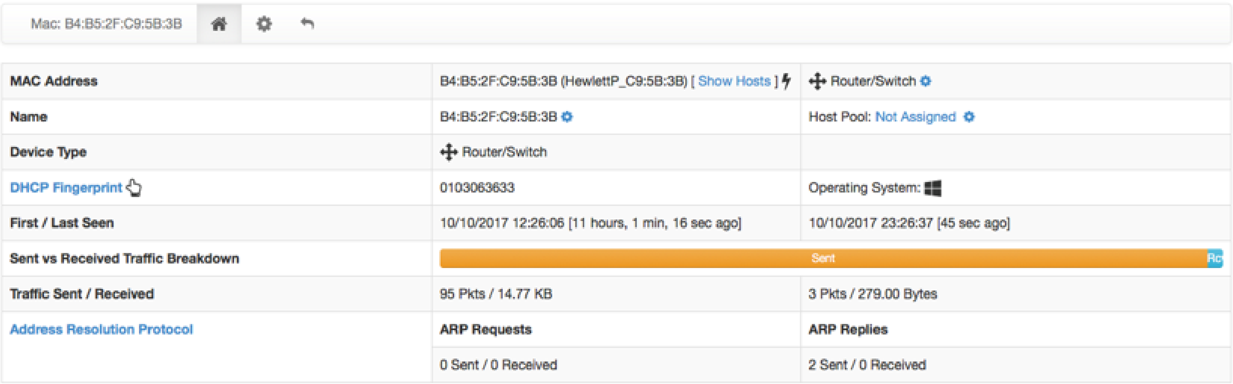


Network Discovery

Last Network Discovery	14/04/2018 10:55:48					
IP Address	**Name**	**Manufacturer**	**MAC Address**	**OS**	**Info**	**Device**
[redacted].1		Juniper Networks	F4:B5:2F:FC:AF:F0			🖥 (OpenSSH_6.4)
[redacted].102	[redacted]	Hewlett Packard	B4:B5:2F:C9:69:7A			🖥
[redacted].104	[redacted]	Hewlett Packard	44:1E:A1:30:30:3D			🖥
[redacted].105	[redacted]	Hewlett Packard	B4:B5:2F:CC:C4:6A			🖥
[redacted].108	[redacted]	Hewlett Packard	10:60:4B:6D:81:6D			🖥
[redacted].11	[redacted]	Dell Inc.	64:00:6A:63:35:CC	🐧		🖥 (Ubuntu)
[redacted].124	[redacted]	Quanta Computer Inc.	00:23:8B:42:88:37	🐧		🖥 (Linux)
[redacted].125	[redacted]	Juniper Networks	88:A2:5E:E6:BB:01			sw-p1-a-2.[redacted]
[redacted].126	[redacted]	Juniper Networks	5C:45:27:3D:14:01			✥ (jswitch-p0-m-1.[redacted])
[redacted].13	[redacted]	Apple, Inc.	A8:60:B6:00:4A:99			🖥
[redacted].2	[redacted]	Quanta Computer Inc.	00:1E:68:2F:11:6D	🐧		🖥 (Linux)
[redacted].22	[redacted]	PCS Computer Systems GmbH	08:00:27:F4:C8:01	⊞		🖥 (Windows)
[redacted].27	[redacted]	Apple, Inc.	3C:07:54:2F:67:65		_ssh._tcp.local _sftp-ssh._tcp.local _rfb._tcp.local	🖥 iMac "Core i7" 3.4 27-Inch (Mid-2011)
[redacted].3		Realtek (UpTech? also reported)	52:54:00:E6:BB:FE	🐧		🖥 (Ubuntu)
[redacted].31	[redacted]	Dell Inc.	BC:30:5B:9C:15:02			🖥
[redacted].32	[redacted]	Apple, Inc.	68:5B:35:97:BB:22		_ssh._tcp.local _sftp-ssh._tcp.local _companion-link._tcp.local _teamviewer._tcp.local	🖥 iMac "Core i7" 3.5 27-Inch (Late 2013)

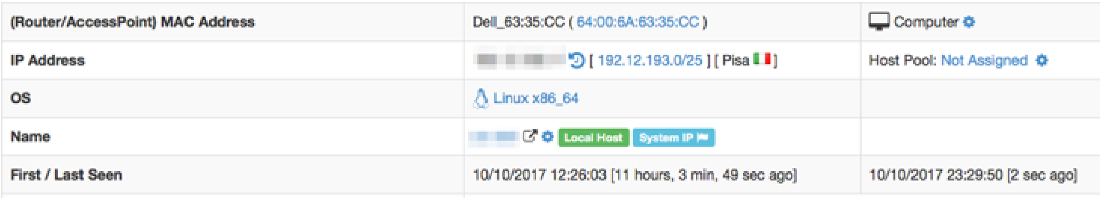
ntopng also offers the ability to interpret network traffic in a more low-level fashion, to move from high-level overview of a certain device traffic, to a low-level analysis of the flow when needed.

ntopng Drill Down



Mac: B4:B5:2F:C9:5B:3B

MAC Address	B4:B5:2F:C9:5B:3B (HewlettP_C9:5B:3B) [Show Hosts] ⚡	✣ Router/Switch ⚙
Name	B4:B5:2F:C9:5B:3B ⚙	Host Pool: Not Assigned ⚙
Device Type	✣ Router/Switch	
DHCP Fingerprint	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	Sent Rc	
Traffic Sent / Received	95 Pkts / 14.77 KB	3 Pkts / 279.00 Bytes
Address Resolution Protocol	ARP Requests	ARP Replies
	0 Sent / 0 Received	2 Sent / 0 Received

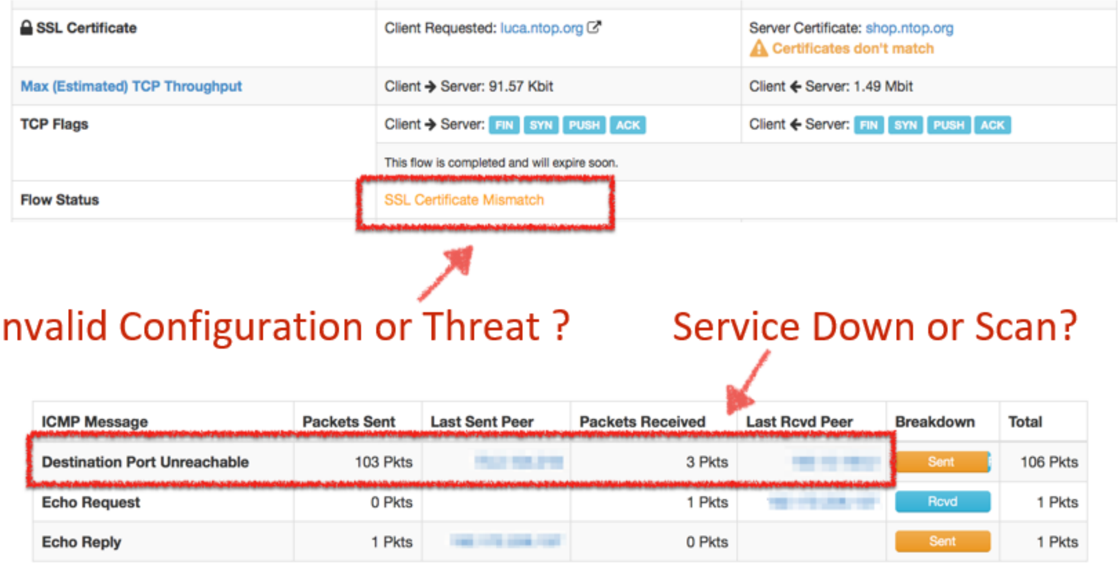


(Router/AccessPoint) MAC Address	Dell_63:35:CC (64:00:6A:63:35:CC)	Computer ⚙
IP Address	[redacted] [192.12.193.0/25] [Pisa 🇮🇹]	Host Pool: Not Assigned ⚙
OS	Linux x86_64	
Name	[redacted] ⚙ Local Host System IP	
First / Last Seen	10/10/2017 12:26:03 [11 hours, 3 min, 49 sec ago]	10/10/2017 23:29:50 [2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

By producing data, ntopng provides an idea of the problem and performs lightweight interpretation through alerts. Yet a network administrator has to decide whether the data presents a problem or not.

ntopng can be used to export data to other people, to other applications, and to do that externally.

ntopng Monitoring



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

Invalid Configuration or Threat ?

Service Down or Scan?

ICMP Message	Packets Sent	Last Sent Peer	Packets Received	Last Rcvd Peer	Breakdown	Total
Destination Port Unreachable	103 Pkts	[redacted]	3 Pkts	[redacted]	Sent	106 Pkts
Echo Request	0 Pkts		1 Pkts	[redacted]	Rcvd	1 Pkts
Echo Reply	1 Pkts	[redacted]	0 Pkts		Sent	1 Pkts

In summary, ntopng features include:

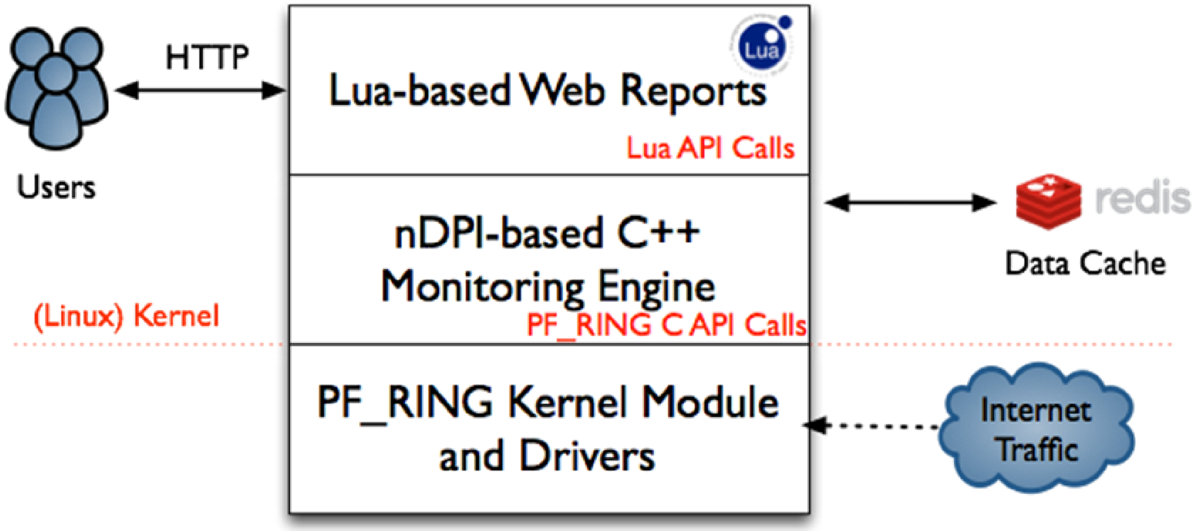
- Embedded alerting system pluggable with Nagios and messaging systems
- Usage as Grafana datasource
- Ready for OpenStack, Wireshark, Vagrant, Docker, Elastic
- nDPI: passive mode = monitoring
- Support for NetFlow/sFlow/SNMP
- Passive/Active Network Device Discovery
- Traffic Behavior Analysis

Technical architecture

> *“InfluxDB is therefore the step forward with respect to RRD. We had to do many changes in our code, not because of the InfluxDB design, but because InfluxDB offers us a lot of new opportunities that we want to exploit.”*

ntopng Architecture:

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



The ntopng Monitoring Engine is coded in C++ and based on 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, and Host.

nDPI library

DPI is a technique for inspecting the packet payload for the purpose of extracting metadata (e.g. protocol). Since ntop found available DPI toolkits on the market to be proprietary, expensive, and to cause vendor lock-ins, and since company sought a DPI toolkit compatible with the open source concept, they created their own GPL DPI toolkit (nDPI) in order to build an open DPI layer for ntop and third-party applications:

- nDPI supports over 240 protocols.
- All flows are analyzed through nDPI to associate an application protocol to them.
- Layer 7 (L7) statistics are available per flow, host, and interface (from which monitoring data is received). L7 refers to the topmost layer of the Open Systems Interconnect Model known as the application layer (the layer that supports end-user processes and applications).
- For network interfaces and local hosts, nDPI statistics are saved persistently to disk.

InfluxDB migration implications

To allow people to use both RRD and InfluxDB until the migration is complete, ntop created a time series Lua layer inside ntopng. RRD/InfluxDB differences had to be masked during data extraction:

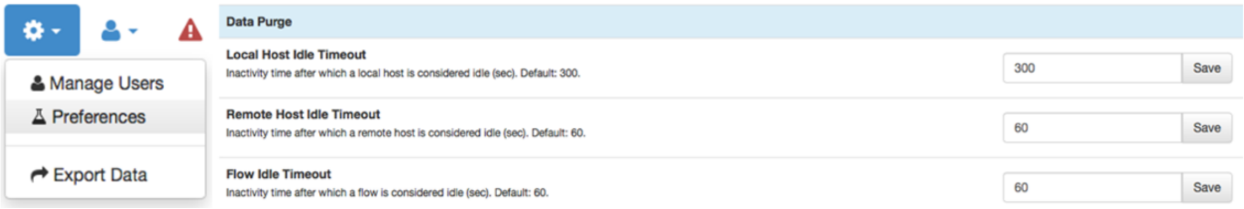
- RRD has automatic data consolidation/rollup that ntop configured in InfluxDB with continuous queries.
- RRD handles natively counters/gauges whereas on InfluxDB, ntop needed to do that in queries.
- In RRD, ntop set the time policy and data value boundaries at archive creation. In InfluxDB, they needed to do that during data extraction to make applications aware of that.
- RRD normalizes automatically data at a specific resolution based on the query time-range. With InfluxDB, they had more precise measurements that had to be properly handled.

ntop has completed data export to InfluxDB via HTTP for all counters. ntopng moved from 5-min host counters to 1 minute or less, and built a time series Lua library for hiding RRD/InfluxDB differences so that existing ntopng users can migrate to Influx and RRD/InfluxDB can be handled seamlessly in the ntopng GUI.

Information lifecycle

Information takes the following path in ntopng architecture:

- 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.



Data Purge

Local Host Idle Timeout
Inactivity time after which a local host is considered idle (sec). Default: 300.
300 Save

Remote Host Idle Timeout
Inactivity time after which a remote host is considered idle (sec). Default: 60.
60 Save

Flow Idle Timeout
Inactivity time after which a flow is considered idle (sec). Default: 60.
60 Save

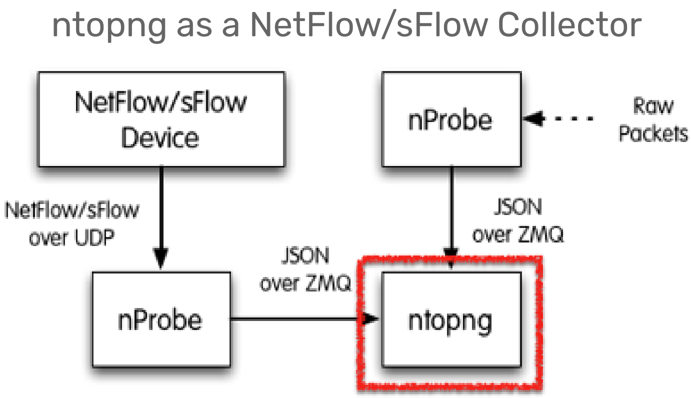
Packet processing journey

The packet journey is very simple and consists of 4 steps:

1. Packet capture through PF_RING - ntop's infrastructure for capturing packets (Linux) - or libpcap (part of every Unix, Windows, or MacOS cell distribution)
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. Moving to the next packet

ntopng as a NetFlow/sFlow collector

ntopng can collect traffic information sent by routers (NetFlow) and switches (sFlow). Flow information needs binary-to-JSON translation since ntopng was designed to be agnostic with respect to flow low-level details. So ntop's other component, nProbe, does the conversion from the flow format to JSON. This information is delivered over ZMQ (ntop's messaging queue) to ntopng. ntopng can do packet collection, workflow collection, and can correlate this information seamlessly.



Results

> *“Removing the inability to monitor large networks with many counters and with low granularity is compulsory, and InfluxDB is definitively adequate for this task.”*

ntopng allows comprehensive traffic views from very high-level to very low-level. This enables users to identify which host, physical port, or switch is causing issues and to troubleshoot the problem since such visibility provides a complete report of the network’s state. Using InfluxDB, ntopng is open to “big data” systems that can scale with data in volume and speed. It is able to export monitoring information in JSON format towards various systems including Elasticsearch / Logstash and ZMQ. ntopng is also able to collect, self-produce (from packets), and export monitoring information by normalizing it in JSON format.

ntopng enables:

- High-speed web-based traffic analysis and flow collection
- Persistent traffic statistics in RRD format
- Layer 7 analysis by leveraging nDPI

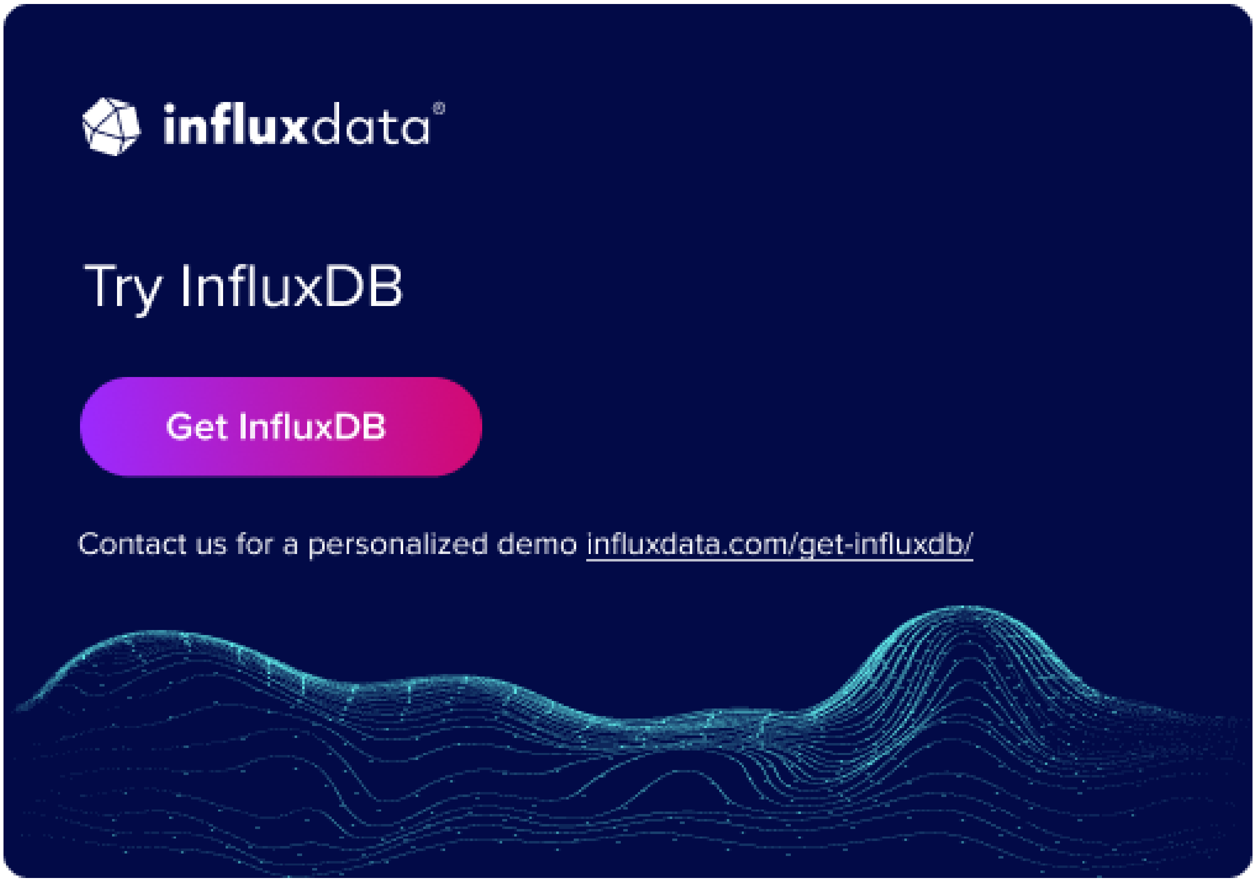
ntopng can be used in a variety of use cases such as:

- Monitoring of a Physical Interface - An example is a physical NIC card.
- Flow Collection - This requires ntopng to be used in conjunction with nProbe which can act as probe/proxy.
- Correlation of host, interface and network flow monitoring data to monitor traffic per application/user

ntopng is available in three versions: Community, Professional (Small Business Edition) and Enterprise. The Community version is free to use and open source. The Professional and Enterprise editions offer some extra features that are particularly useful for SMEs or larger organizations.
Using InfluxDB for ntopng, ntop is fulfilling its mission of real-time network traffic monitoring while maintaining its commitment to being, as its community saying goes, “open-source from the source”.

About InfluxData

InfluxData is the creator of InfluxDB, the leading time series platform. We empower developers and organizations, such as Cisco, IBM, Lego, Siemens, and Tesla, to build transformative IoT, analytics and monitoring applications. Our technology is purpose-built to handle the massive volumes of time-stamped data produced by sensors, applications and computer infrastructure. Easy to start and scale, InfluxDB gives developers time to focus on the features and functionalities that give their apps a competitive edge. InfluxData is headquartered in San Francisco, with a workforce distributed throughout the U.S. and across Europe. For more information, visit influxdata.com and follow us @InfluxDB.



influxdata®

Try InfluxDB

Get InfluxDB

Contact us for a personalized demo influxdata.com/get-influxdb/

548 Market St. PMB 77953, San Francisco, CA 94104