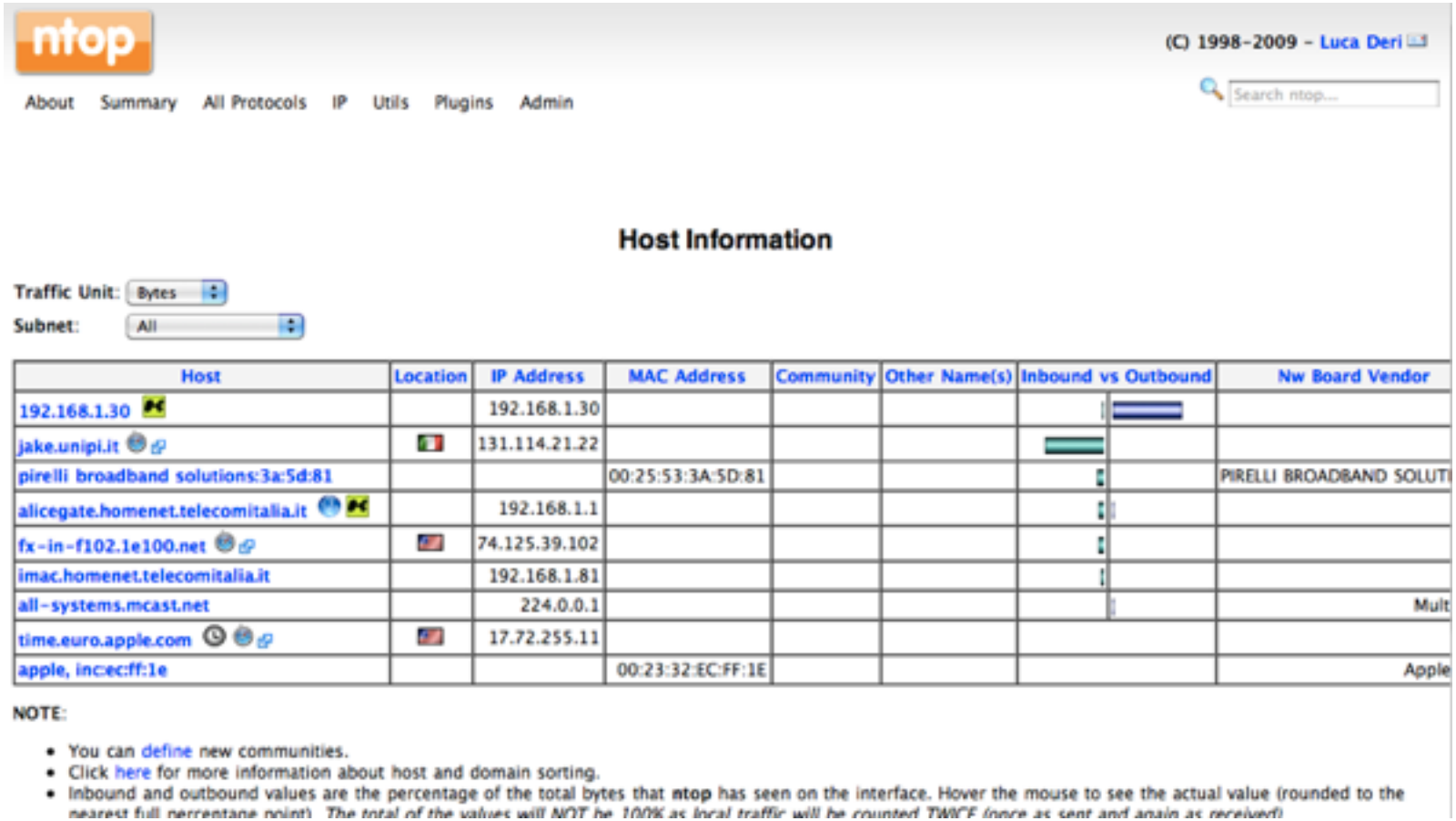
Integrating ntop with Python

What's ntop ?

ntop is a simple, open source (GPL), portable traffic measurement and monitoring tool, which supports various management activities, including network optimization and planning, and detection of network security violations.

Welcome to ntop



ntop

(C) 1998-2009 - Luca Deri

About Summary All Protocols IP Utils Plugins Admin

Search ntop...

Host Information

Traffic Unit: Bytes

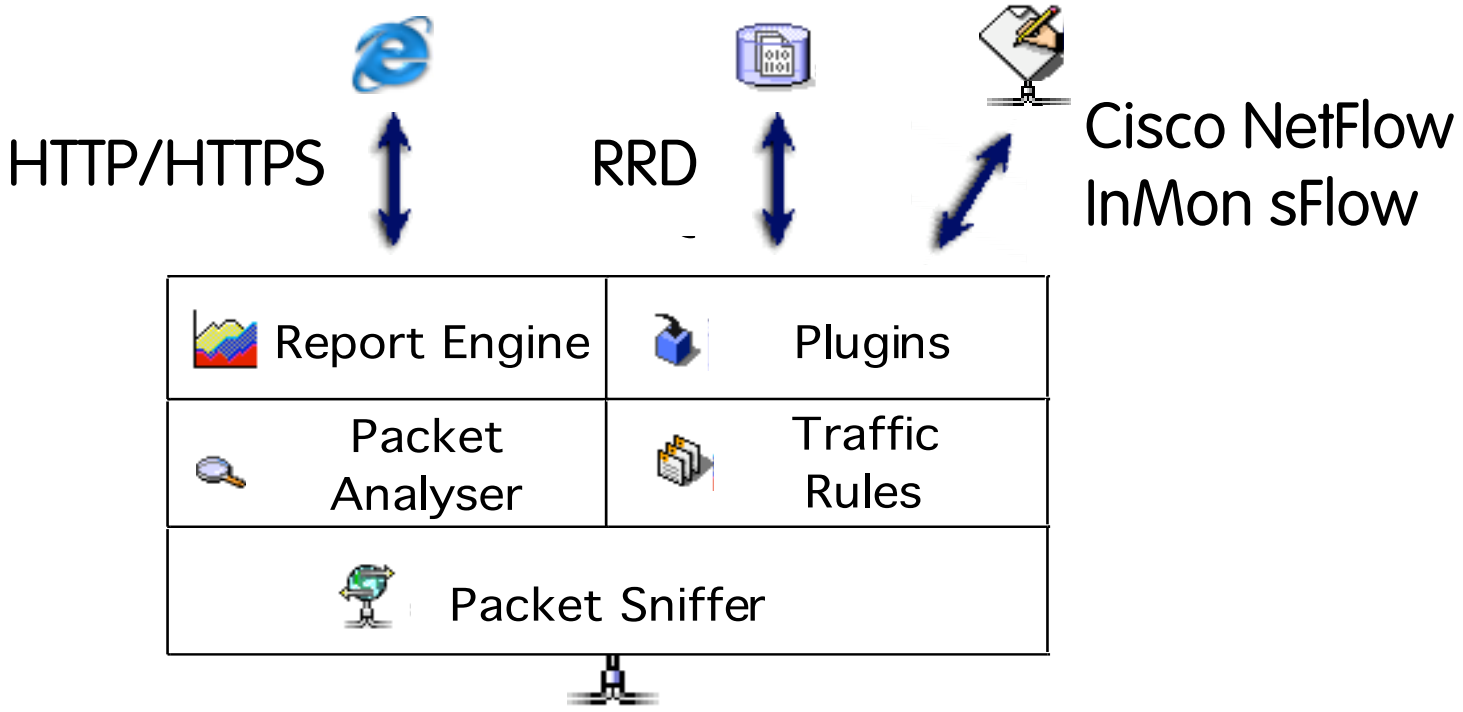
Subnet: All

Host	Location	IP Address	MAC Address	Community	Other Name(s)	Inbound vs Outbound	Nw Board Vendor
192.168.1.30		192.168.1.30					
jake.unipi.it		131.114.21.22					
pirelli broadband solutions:3a:5d:81			00:25:53:3A:5D:81				PIRELLI BROADBAND SOLUTI
alicegate.homenet.telecomitalia.it		192.168.1.1					
fx-in-f102.1e100.net		74.125.39.102					
imac.homenet.telecomitalia.it		192.168.1.81					
all-systems.mcast.net		224.0.0.1					Mult
time.euro.apple.com		17.72.255.11					
apple, inc:ec:ff:1e			00:23:32:EC:FF:1E				Apple

NOTE:

- You can define new communities.
- Click here for more information about host and domain sorting.
- Inbound and outbound values are the percentage of the total bytes that **ntop** has seen on the interface. Hover the mouse to see the actual value (rounded to the nearest full percentage point). *The total of the values will NOT be 100% as local traffic will be counted TWICE (once as sent and again as received)*

ntop Architecture



Towards ntop Scripting [1/2]

- ntop report engine is written in C
  - Pros:
    - Fast and efficient
    - Tight to the ntop architecture
  - Cons:
    - Changing anything in pages requires C/ntop coding skills
    - Inability to modify/change web pages on the fly without ntop restart.

- ntop engine is monolithic and it represents "the view of network" from ntop's point of view.
  - Pros:
    - Small in size and efficient while handling binary packets
  - Cons:
    - ntop was not designed to offer a simple API for extending its engine

Towards ntop Scripting [2/2]

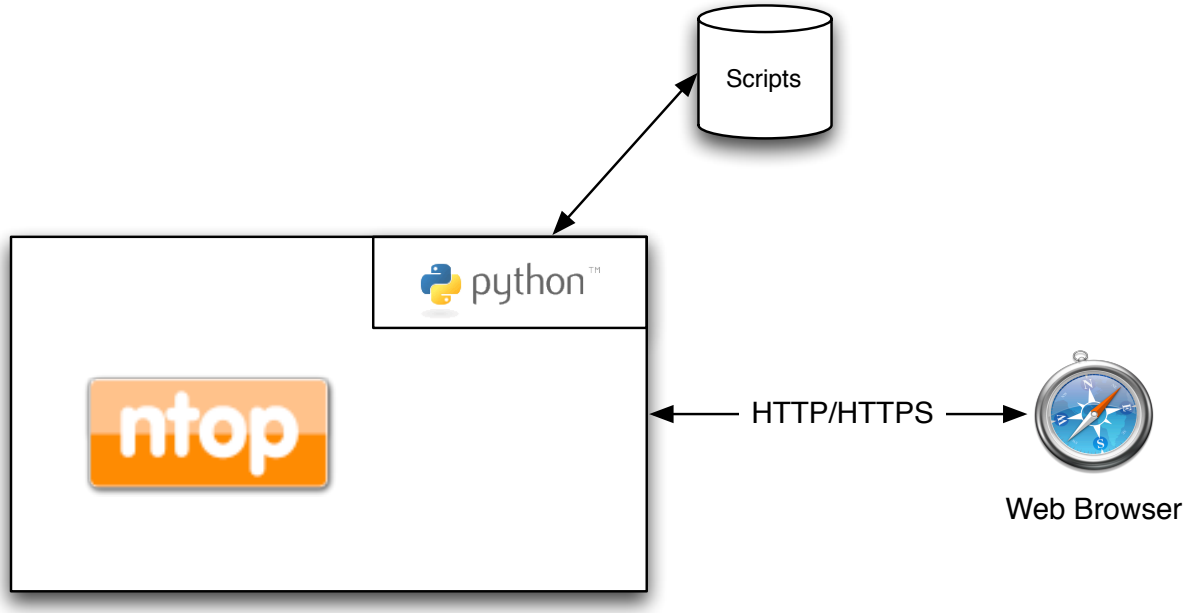
Why is ntop scripting necessary ?

- It allows ntop to be easily extended in non-performance critical sections.
- It can provide an uniform API for non ntop core-developers to add new functionalities:
  - Easily: scripting vs. C skills can be often found among system administrator
  - The API allows users to extend the application without breaking or adding extra-weight on the core that's still under control of core-developers.
  - Scripting languages offers many features (e.g. HTML page templates, or PDF support) not easily implementable using plain C.
  - Code can run on a sandbox without interfering with the engine.
  - Memory management, in particular for rendering HTML content, is handled automatically by the interpreter.

ntop Scripting Attempts

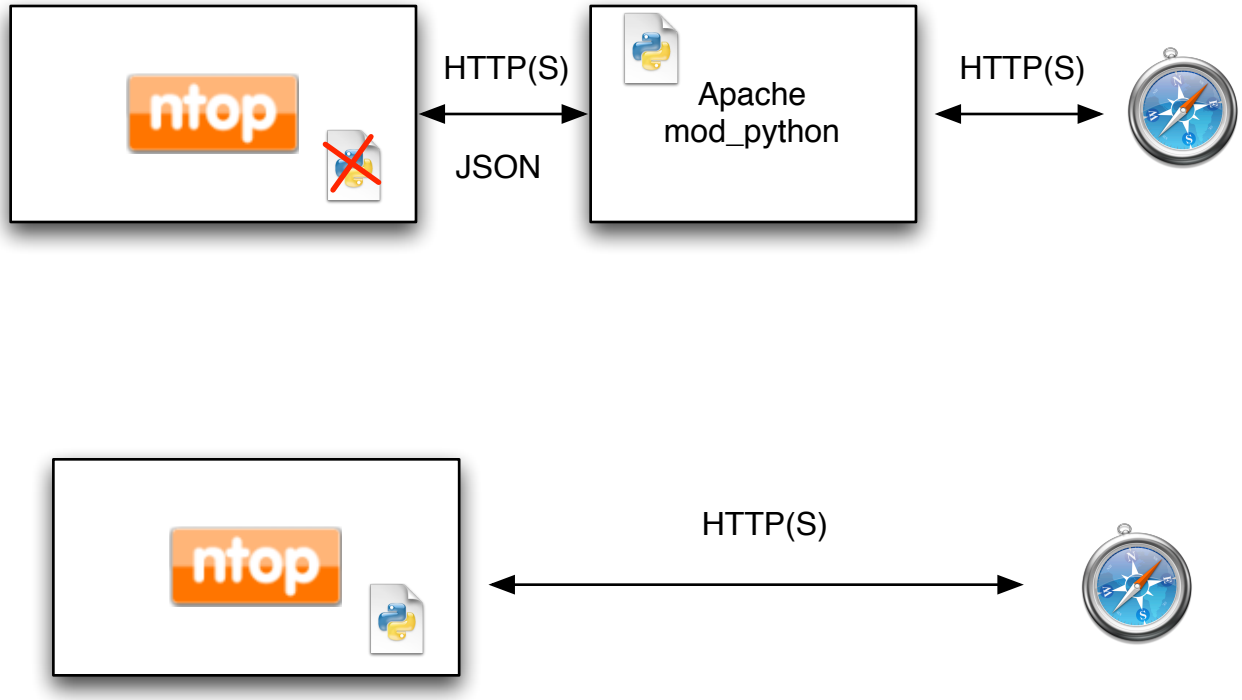
- In mid '2000 a Perl-plugin was added to ntop
  - Support of scriptability in ntop
  - Nightmare to compile across OS (Linux vs Win vs OSX) and Perl versions
  - Although Perl can be embedded, its design does not ease this task.
  - Very heavy interpreter: it can be used for web reporting not for the engine (too much memory used and persistent interpreter is complicated).
- Why not Lua ?
  - Easy to embed, very light, scripts can be compiled (perhaps you don't want to share the source code?)
  - Unfortunately Lua has a uncommon syntax (not too many developers like it), and it support too few functionalities with the result that it was just a better C.
- And Finally Python...
  - Love at first sight: easy to embed, feature rich, efficient.

ntop Python Scriptability

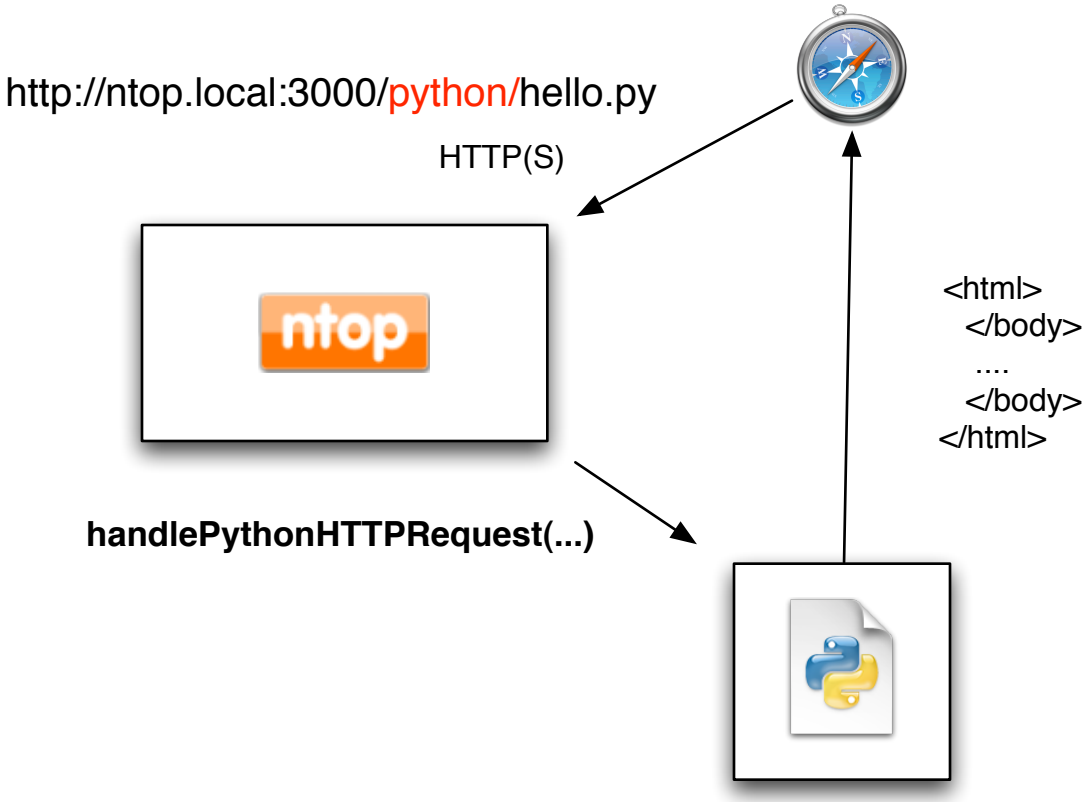


- Ntop web server can execute python scripts:
  - Methods to access the state of ntop
  - Python cgi module process forms and html url parameters
  - Mako templates generate dynamic html pages

External vs. Embedded Scripting



ntop Python Engine: Script Lifecycle



ntop Python Engine: Interpreter Lifecycle

....
ntop.c
ntop_darwin.c
ntop_win32.c
pbuf.c
plugin.c
pluginSkeleton.c
prefs.c
protocols.c
python.c
report.c
reportUtils.c
.....

```
static void init_python_ntop(void) {
  createMutex(&python_mutex);
  Py_InitModule("ntop", ntop_methods);
  Py_InitModule("interface", interface_methods);
  Py_InitModule("host", host_methods);
  Py_InitModule("fastbit", fastbit_methods);
}

int handlePythonHTTPRequest(char *url, uint postLen) {
/* 1 - Parse HTTP(S) request */
...

/* 2 - Setup Environment */
safe_snprintf(__FILE__, __LINE__, buf, sizeof(buf),
              "import os\nos.environ['DOCUMENT_ROOT']='%s'\n"
              "os.environ['REQUEST_METHOD']='POST'\n"
              "os.environ['CONTENT_TYPE']='application/x-www-form-urlencoded'\n"
              "os.environ['CONTENT_LENGTH']='%u'\n",
              document_root, postLen);
 PyRun_SimpleString(buf);

 PyRun_SimpleFile(fd, python_path); /* 3 - Run the script */
}

void term_python(void) {
  Py_Finalize();   /* Cleaning up the interpreter */
}
```

ntop Python Engine: Methods Implementation

```
static PyMethodDef ntop_methods[] = {
  { "sendHTTPHeader", python_sendHTTPHeader, METH_VARARGSl METH_KEYWORDS, "" },
  { "returnHTTPnotImplemented", python_returnHTTPnotImplemented, METH_VARARGS, "" },
  { "returnHTTPversionServerError", python_returnHTTPversionServerError, METH_VARARGS, "" },
  { "getFirstHost",    python_getFirstHost,    METH_VARARGS, "" },
  { "getNextHost",     python_getNextHost,     METH_VARARGS, "" },
  .....
  { NULL, NULL, 0, NULL }
}
```

```
static PyObject* python_getFirstHost(PyObject *self, PyObject *args) {
  int actualDeviceId;

  /* parse the incoming arguments */
  if(!PyArg_ParseTuple(args, "i", &actualDeviceId))
    return NULL;

  ntop_host = getFirstHost(actualDeviceId);

  return Py_BuildValue("i", ntop_host ? 1 : 0);
}
```

ntop/Win32 and Python

- In Unix there's the concept of stdout/stdin/stderr.
- Each python script can read from stdin and print on stdout/stderr.
- Prior to execute a script, file descriptors for std* are redirected to the interpreter.
- This means that a script that calls print(...) will actually not print on the ntop console but on the returned HTTP page.
- On Windows:
  - The std* concept is also supported.
  - Unfortunately std* can be redirected only when a new process (not thread) is spawn.
  - The consequence is that on ntop/Win32 calls to print(...) do print on console and not on the returned HTTP page.
  - Please use ntop.sendString(...) method instead.

ntop Python Engine: Native Types

```
static PyObject* python_getGeoIP(PyObject *self, PyObject *args) {
  PyObject *obj = PyDict_New();
  GeoIPRecord *geo = (ntop_host && ntop_host->geo_ip) ? ntop_host->geo_ip : NULL;

  if(geo != NULL) {
    PyDict_SetItem(obj, PyString_FromString("country_code"),
                                PyString_FromString(VAL(geo->country_code)));
    PyDict_SetItem(obj, PyString_FromString("country_name"
                                PyString_FromString(VAL(geo->country_name)));
    PyDict_SetItem(obj, PyString_FromString("region"), PyString_FromString(VAL(geo->region)));
    PyDict_SetItem(obj, PyString_FromString("city"), PyString_FromString(VAL(geo->city)));
    PyDict_SetItem(obj, PyString_FromString("latitude"), PyFloat_FromDouble((double)geo->latitude));
    PyDict_SetItem(obj, PyString_FromString("longitude"), PyFloat_FromDouble((double)geo->longitude));
  }

  return obj;
}
```

Mixing ntop with Python Modules

- Persistent interpreter: minimal startup time
- The python interpreter spawn by ntop has full modules visibility (i.e. no need to re-install modules as with other scripting languages such as Perl)
- Installed python modules are automatically detected by the ntop interpreter.
- The interpreter can handle both source (.py) and binary compiled (.pyc) scripts.
- ntop-interpreted scripts can be modified while ntop is running.

- Limitations
  - As the python interpreter is persistent, new modules installed after the interpreter has been started (i.e. after ntop startup) might not be detected.
  - Do NOT call exit functions (e.g. sys.exit()) otherwise the ntop interpreter will quit!

Changing ntop Behavior via Python

- In other embedded interpreters (e.g. Perl) the interpret is spawn on a new process and it gets a copy of the environment.
- This means that whatever a script changes in the environment, changes are blown up after the script is over.
- The consequence is that scripts cannot be used for implementing selected portions of the ntop engine but for reporting only.

- Python is different...
  - Scripts can modify the ntop behavior: methods can be implemented for both getting and setting a value.
  - Changes, by means of set(), are actually changing the value into the ntop engine and not a copy.
  - Beware: this does not apply on Unix when ntop is started without '-K' option as in this case each script is executed into a new process.

Simple ntop/Python Script

```
import ntop;

import host;

import cgi, cgitb

cgitb.enable();

form = cgi.FieldStorage();

ntop.printHTMLHeader("Welcome to ntop+Python ["+ntop.getPreference("ntop.devices")
+"]", 1, 0);

ntop.sendString("<center><table border>\n");

ntop.sendString("<tr><th>MAC Address</th><th>IP Address</th><th>Name</th><th>#
Sessions</th><th># Contacted Peers</th><th>Fingerprint</th><th>Serial</th></tr>\n");

while ntop.getNextHost(0):

    ntop.sendString("<tr><td align=right>"+host.ethAddress()+"</td>"

                +"<td align=right>"+host.ipAddress()+"</td>"+"<td
align=right>"+host.hostResolvedName()+"</td>"

                +"<td align=center>"+host.numHostSessions()+"</td>"+"<td
align=center>"+host.totContactedSentPeers()+"</td>"

                +"<td align=right>"+host.fingerprint()+"</td>"+"<td
align=center>"+host.serial()+"</td>"+"</tr>\n");

ntop.sendString("</table></center>\n");

ntop.printHTMLFooter();
```

Python Modules

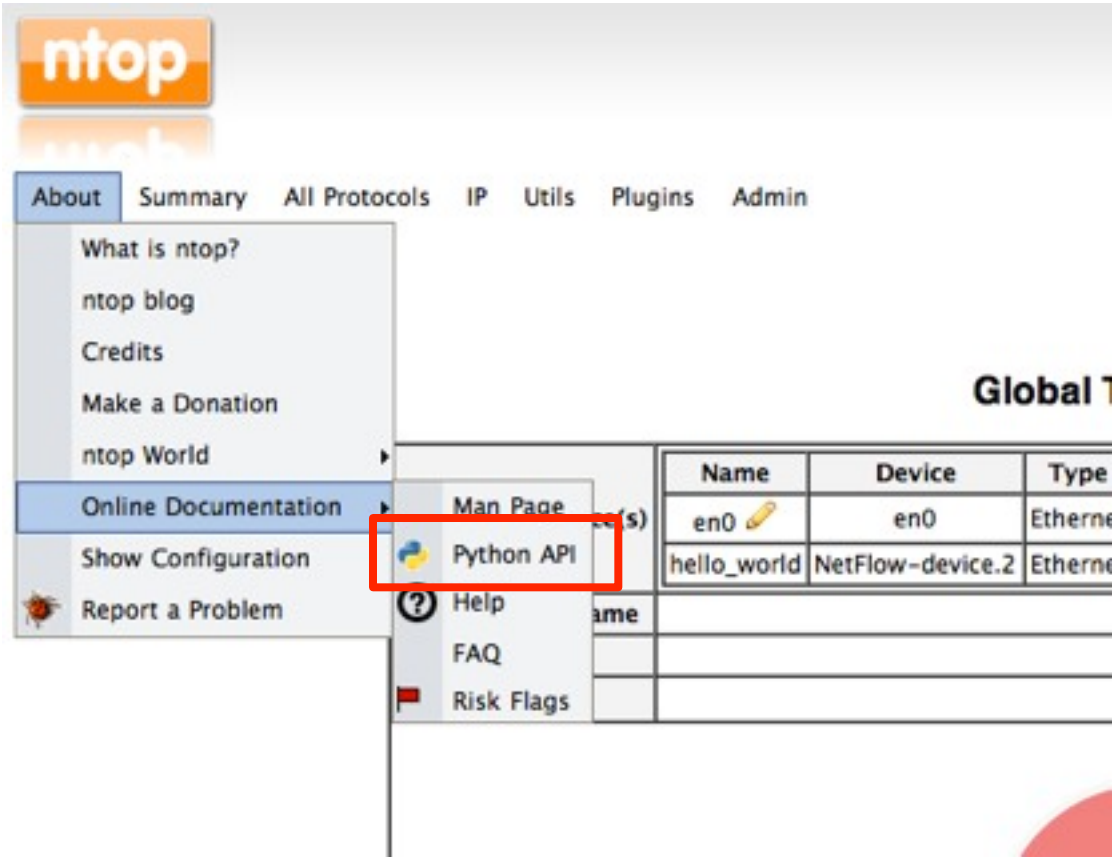
- ntop implements three python modules:
  - ntop (sendString, getNextHost, getPreference…)
    - Interact with ntop engine
  - host (serial, geoIp, ipAddress…)
    - Drill-down on a specific host instance selected via the ntop.*
  - interfaces (name, numInterfaces, numHosts…)
    - Report information about know ntop instances

- All scripts executed via ntop must be installed into the python/ directory

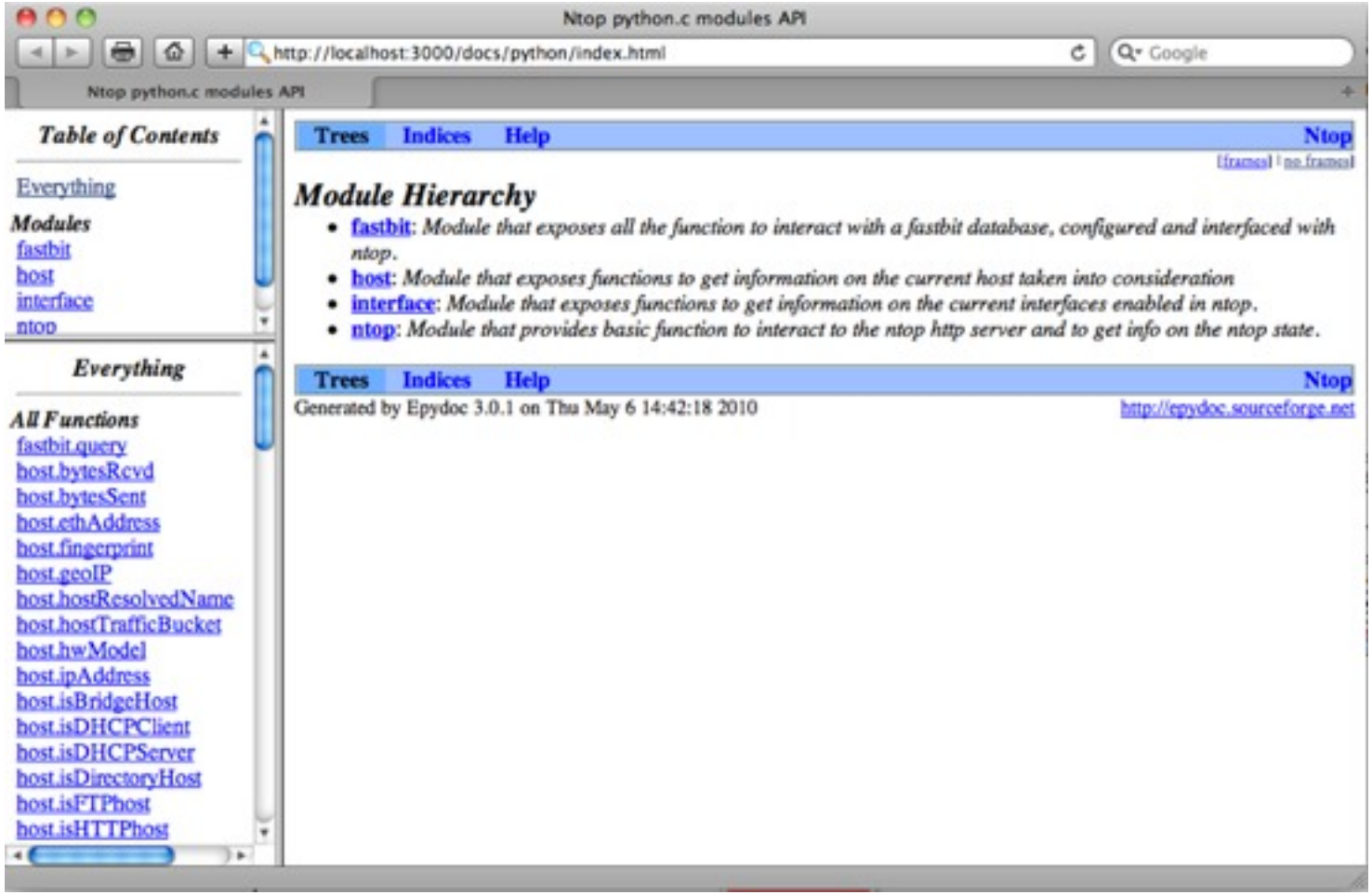
Some Python Advantages

- High level object oriented scripting language
- Easy to embed and to extend
- Fast and portable across platforms
- Supports template technology for building html pages
- Open source

Python Online Documentation [1/2]



Python Online Documentation [2/2]



ntop Python Modules: ntop

- Allow people to:
  - Return content to remote users via HTTP
  - Find hosts using various criteria such as IP address
  - Retrieve information about ntop (e.g. version, operating system etc.)
  - Read/write preferences stored on GDBM databases
  - Update RRD archives

```
rsp = {}

rsp['version'] = ntop.version();
rsp['os'] = ntop.os();
rsp['uptime'] = ntop.uptime();

ntop.sendHTTPHeader(1) # 1 = HTTP
ntop.sendString(json.dumps(rsp, sort_keys=False, indent=4))
```

```
ntop.printHTMLHeader("Welcome to ntop+Python ["+ntop.getPreference("ntop.devices")
+"]", 1, 0);

ntop.sendString("Hello World\n");

ntop.printHTMLFooter();
```

ntop Python Modules: interface

- Allow people to:
  - List known ntop interfaces
  - Retrieve interface attributes
  - Access interface traffic statistics

```
ifnames = []

try:
        for i in range(interface.numInterfaces()):
                ifnames.append(interface.name(i))

except Exception as inst:
    print type(inst)     # the exception instance
    print inst.args      # arguments stored in .args
    print inst           # __str__ allows args to printed directly

ntop.sendHTTPHeader(1) # 1 = HTML
ntop.sendString(json.dumps(ifnames, sort_keys=True, indent=4))
```

ntop Python Modules: host

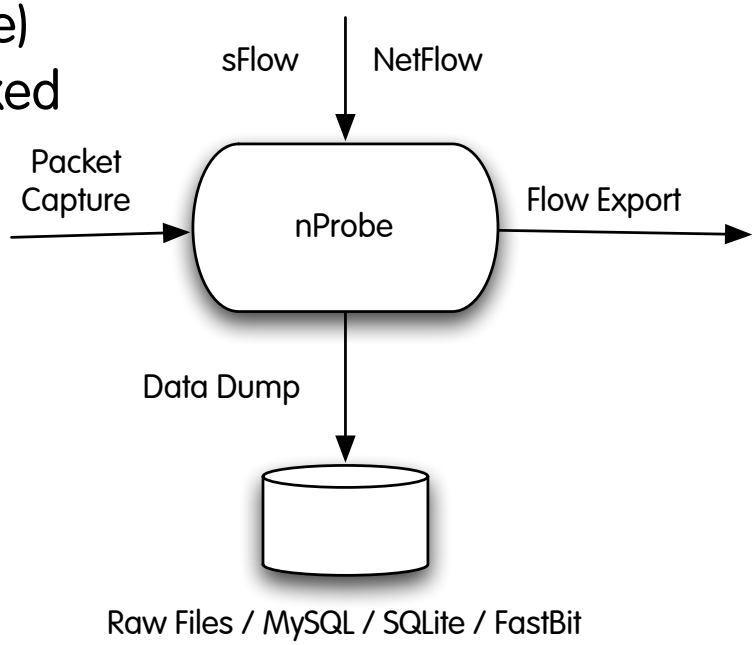
- For a given host it allows people to:
  - Retrieve attributes (e.g. check whether a given host is a HTTP server)
  - Access traffic statistics (e.g. traffic sent/received)
  - This is the core module for accessing host traffic information

```
ntop.printHTMLHeader("Welcome to ntop+Python", 1, 1);

while ntop.getNextHost(0):
    pprint.pprint(host.sendThpt())
    pprint.pprint(host.receiveThpt())
```

ntop Python Modules: fastbit

- Fastbit is a column-oriented database that features compressed bitmap indexes.
- nProbe (a Cisco NetFlow compliant probe) allows flows to be saved on fastbit-indexed databases.
- This ntop modules allow queries to be performed on fastbit databases.

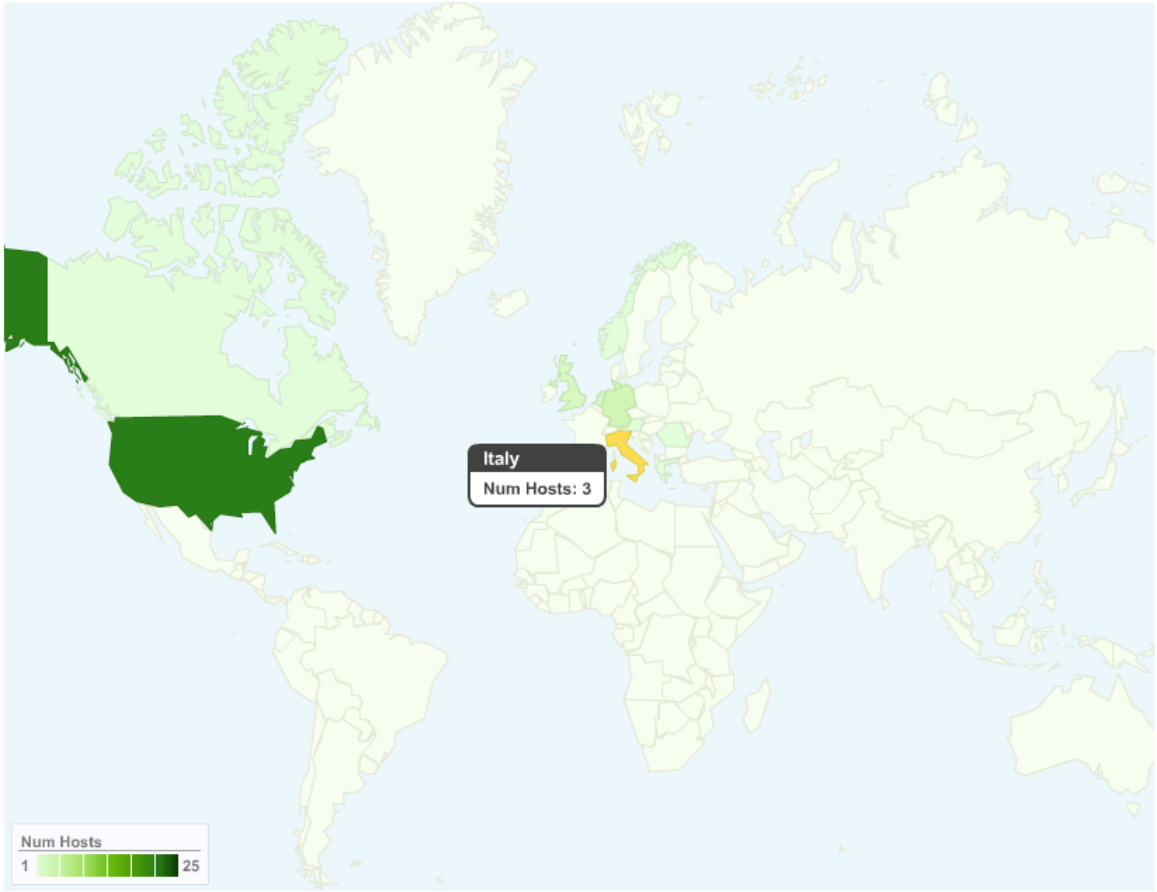



```
print "Query: SELECT %s FROM %s WHERE %s LIMIT %i" %(selectArg,os.path.join
(pathFastBit, fromArg),  whereArg, limit)
res = fastbit.query(os.path.join(pathFastBit, fromArg), selectArg, whereArg,
limit)
print 'Number of records: %i' % len(res['values'])
```

Host Region Map [1/3]

- Interactive Flash™ world map, that displays hosts distribution by country and by cities of a selected country
- Ntop + 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 ntop server for updated data

Host Region Map [2/3]



Host Region Map [3/3]

Country	Num Hosts	Code
United States	25	US
Germany	4	DE
Netherlands	3	NL
Italy	3	IT
United Kingdom	3	GB
Greece	1	GR
Norway	1	NO
Canada	1	CA
Romania	1	RO
Austria	1	AT

City	Num Hosts
London	2
Warwick	1

Geolocation Summary	
Hosts considered for analysis	48
Hosts located in unknown countries	5

RRDAlarm

- It allows network administrators to
  - Configure thresholds for RRD databases
  - Perform a periodical threshold check
  - Emit alarms when thresholds are crossed

- A threshold is defined as:

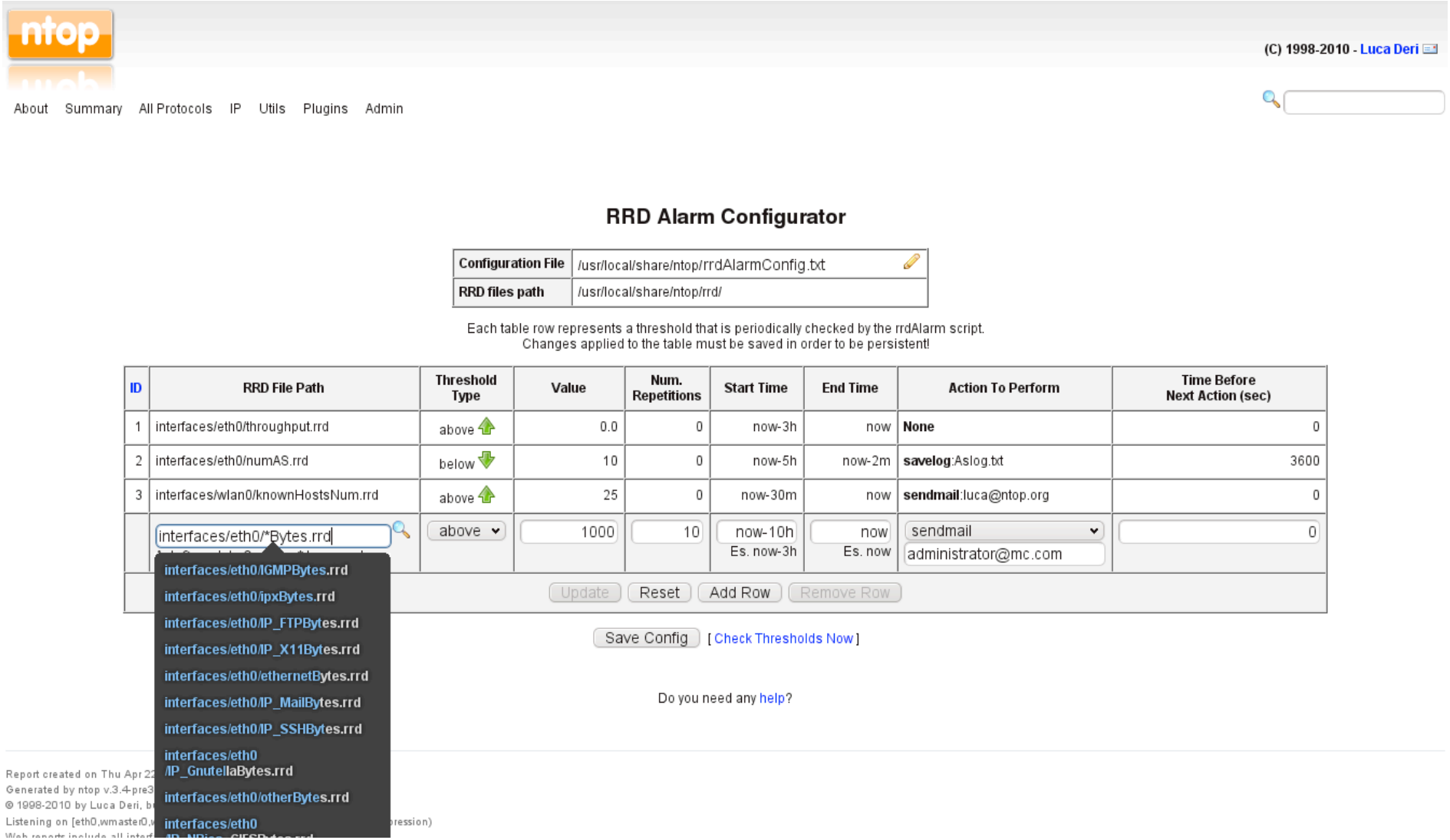
  RRDs Files, Type, Value, Number of repetitions, Time Start/End, Action to perform in case of match, Time before next action (rearm)

- Whenever a threshold is exceeded an alarm is triggered and the specific script associated to that threshold is run.
  - E.g. savelog: mylog.txt, or sendmail: deri@ntop.org

RRDAlarm Configuration [1/2]

- Create or load a configuration files for RRDAlarm
- View, set, modify existing thresholds
- Autocomplete feature for RRD File Path field
  - To see the actual file/s associated to the threshold
  - Browser Ajax request, json response (json module)
- Parameters validation (javascript and python regex)
- Start a check with html report

Using RRDAlarm Configuration [2/2]



RRDAlarm Check [1/2]

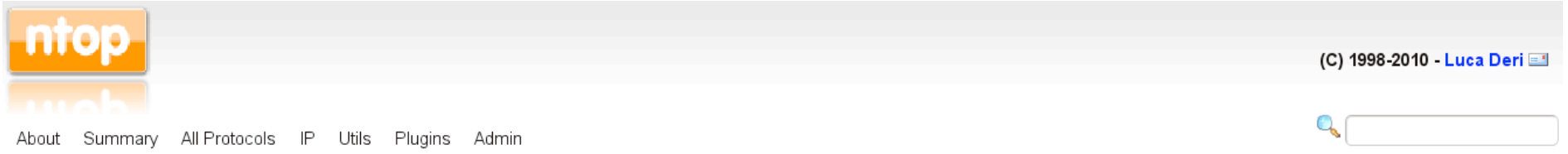
- Performs a check based on the configuration file passed
- Uses Python pickle to store information on the thresholds exceeded and the alarms triggered
- Stores persistently
  - the number of alarms triggered and the time of execution in two different RRD databases.
  - A history of the actions executed so far.
- RRD databases access is based on ntop/python rrdtool interface

RRDAlarm Check [2/2]

- Modus Operandi:
  - Html output, for interactive testing purpose
  - Batch (quiet) mode for continuous periodical check
    - CRON script to perform a GET every minute on URL
    - e.g. http://localhost:3000/python/rrdAlarm/start.py?noHTML=true

- Further actions (to perform in case of threshold cross) can be installed adding new scripts to the `ntopInstallPath/python/script` directory

RRDAlarm Example



RRD Alarm Report

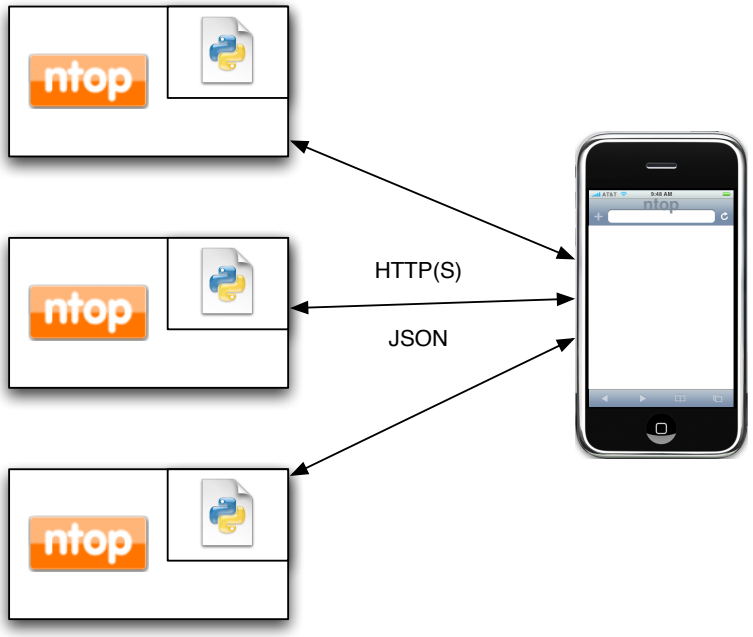
Script duration	0.08 sec.
Number of files checked	28
Number of alarms fired	1

Details

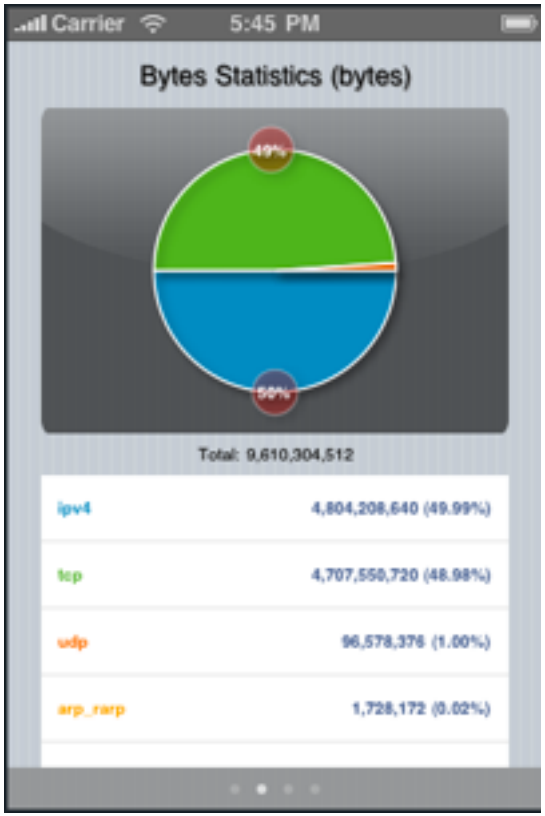
ID	File	Value	Type	Threshold	Time	Action	
1	/usr/local/share/ntop/rrd/interfaces/eth0/throughput.rrd	1697.6	above	0.0	Sat, 24 Apr 2010 01:08:20	None	ALARM FIRED
2	/usr/local/share/ntop/rrd/interfaces/eth0/numAS.rrd	-	below	10.0	Fri, 23 Apr 2010 22:10:00	None	OK
3	/usr/local/share/ntop/rrd/interfaces/wlan0/knownHostsNum.rrd	-	above	25.0	Sat, 24 Apr 2010 01:10:00	None	OK
4	/usr/local/share/ntop/rrd/interfaces/eth0/IGMPBytes.rrd	-	above	1000.0	Sat, 24 Apr 2010 01:10:00	None	OK
4	/usr/local/share/ntop/rrd/interfaces/eth0/ipxBytes.rrd	-	above	1000.0	Sat, 24 Apr 2010 01:10:00	None	OK
4	/usr/local/share/ntop/rrd/interfaces/eth0/IP_FTPBytes.rrd	-	above	1000.0	Sat, 24 Apr 2010 01:10:00	None	OK
4	/usr/local/share/ntop/rrd/interfaces/eth0/IP_X11Bytes.rrd	-	above	1000.0	Sat, 24 Apr 2010 01:10:00	None	OK

ntop on-the-go [1/2]

- Apple iPhone is commonly used as mobile web pad.
- Accessing ntop information in mobility is often required by network administrators.
- The ntop web GUI can be accessed via Apple Safari, however a tighten and more comprehensive interface was necessary.
- Ability to control several ntop instances via a single device.
- Access traffic information as well as configuration information.
- Available (soon) on the AppleStore.



ntop on-the-go [2/2]



References

- ntop Web Site: http://www.ntop.org/
- Author Papers: http://luca.ntop.org

All work is open-source and released under GPL.