NANOG 76

HACKATHON RECAP

Syed Ahmed

ORACLE

Hackathon Recap

- 8th hackathon organized by NANOG
- Roughly had 50 participants
- Working in 10 different teams
- A lot of college students
- Bunch of first timers

Hackathon Goals

- Theme of hackathon was deploying active monitoring solution
- Extract topology info using some kind of automation
- Build network map using topology info
- Calculate all possible best path between end points
- Probe all possible best paths and account for failure as it happens

Experience

- Day started with explaining hack, lab topology
- We gave a quick of overview of tools and protocols that might help to come up with solution
- It was interesting to notice a lot of participants have heard about exabgp but never got chance to use it.
- Some of the participants have not heard about BGP – LS and SCAPY.
- Teams worked on hack from 10:30 to 6pm
- At 6pm each team presented their work
- Winner was decided based on voting

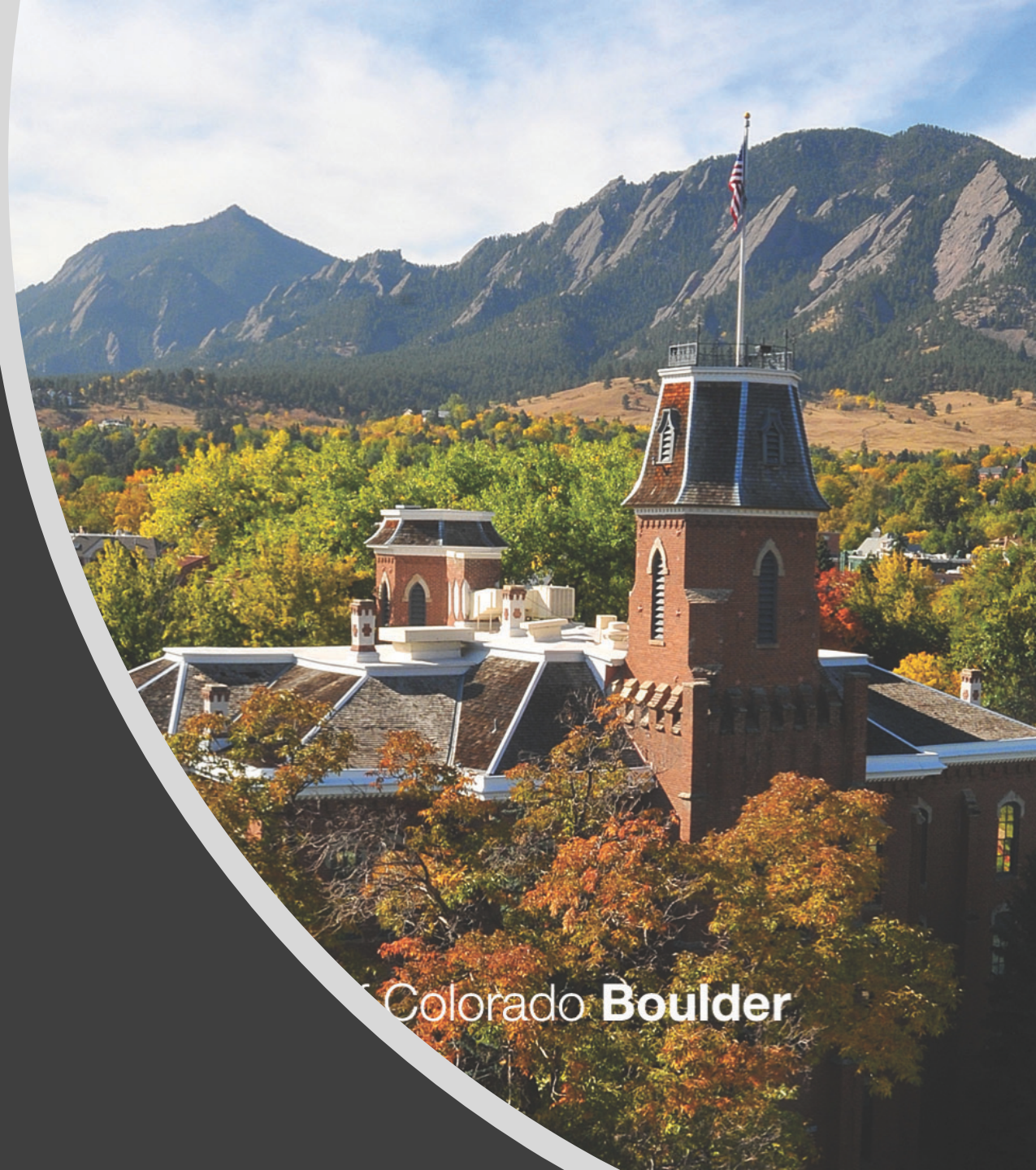
Experience (contd.)

- We also encourage participants to bring their own project
- Project that got resurrected from NANOG67. PiCon: Console servers from RPi’s
- Winner of this hackathon are
  - NetBuffs
  - picon: Console servers from RPis

NANOG 76 HACKATHON

Team: **NETBUFFS**

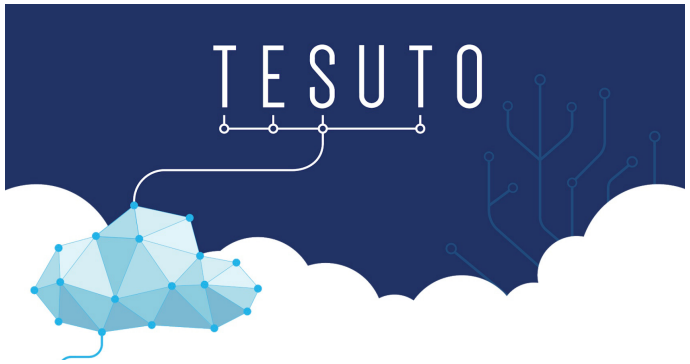
University of Colorado Boulder



Thank you!

ORACLE
Cloud Infrastructure

PLATINUM SPONSOR



Problem Statement

- Build a system to automate network monitoring and failure detection
- Reduce time to identify network and application problems
- Automate reporting of the issue based on the detected failure
- Make use of open source tools

Hackathon Goals

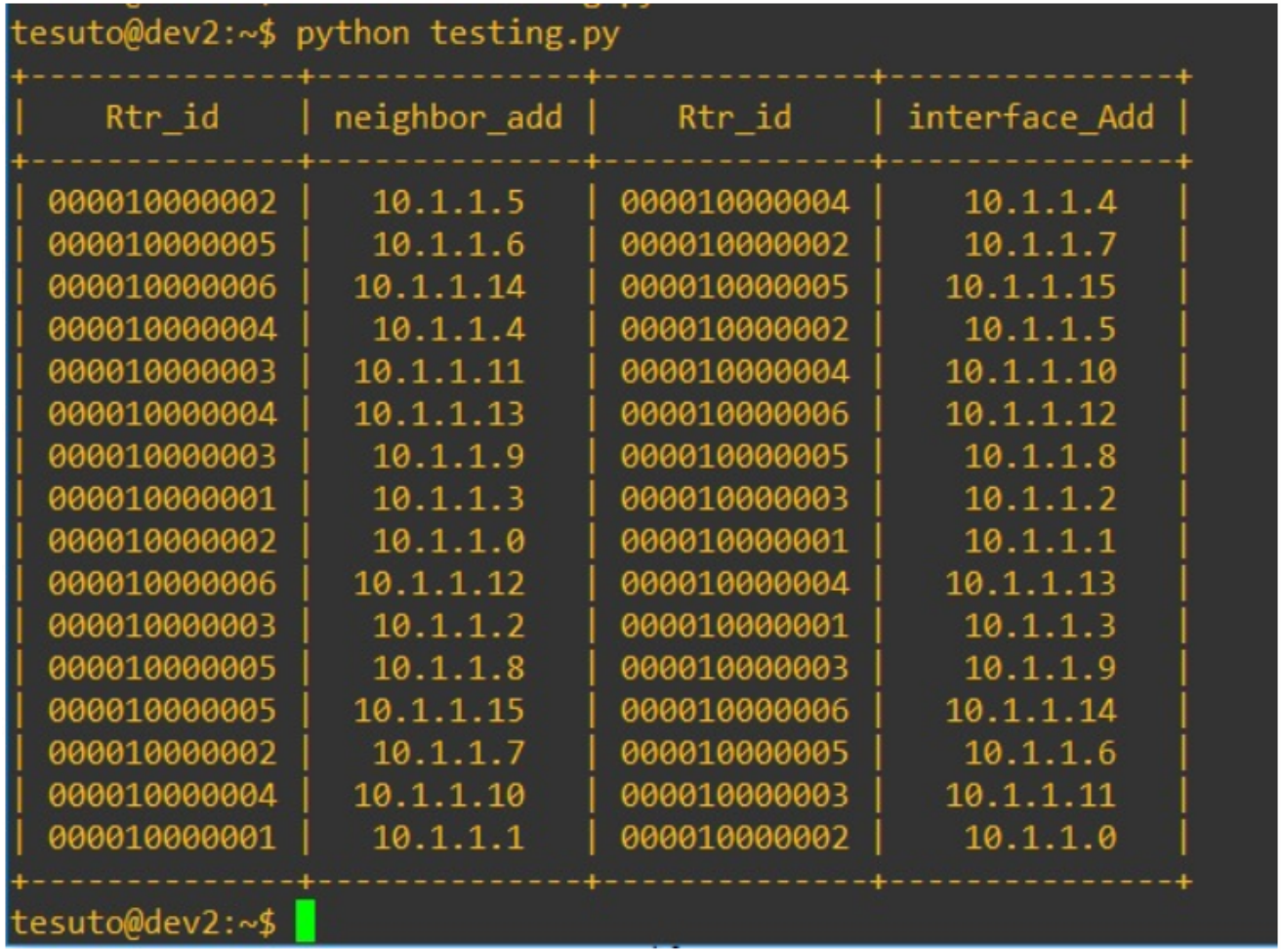
- Extract topology information (number of all possible paths and hops)
- Active monitoring of network paths for failures
- Automatically report failures

Parsing network configuration

- Below information of each link was parsed from routes.txt:
- Neighbor ID
- Neighbor address
- Interface ID
- Interface Address

Tools Used:

- Python
- Json
- ExaBGP

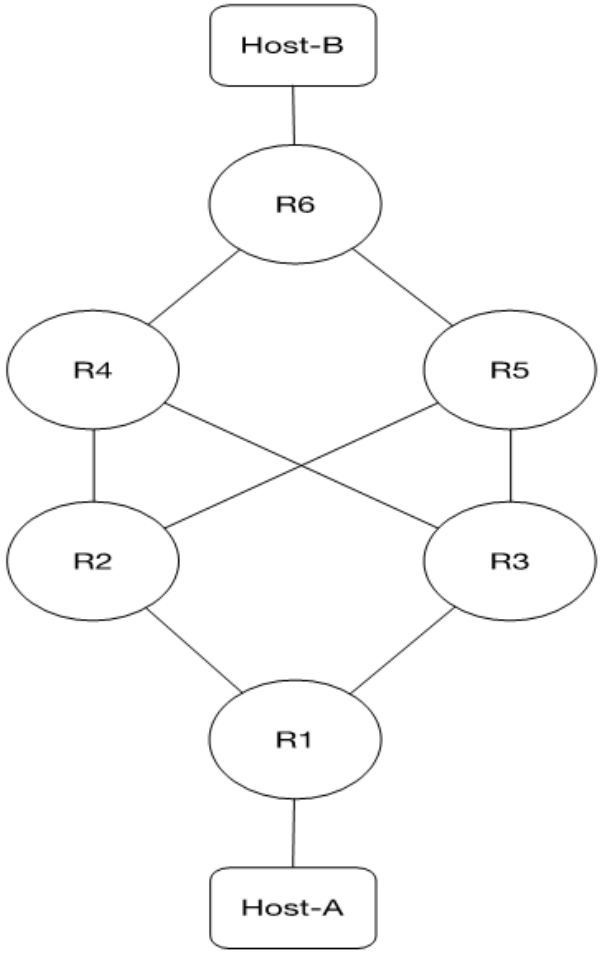


```
tesuto@dev2:~$ python testing.py
+--------------+--------------+--------------+--------------+
|    Rtr_id    | neighbor_add |    Rtr_id    | interface_Add |
+--------------+--------------+--------------+--------------+
| 000010000002 |   10.1.1.5   | 000010000004 |    10.1.1.4   |
| 000010000005 |   10.1.1.6   | 000010000002 |    10.1.1.7   |
| 000010000006 |  10.1.1.14   | 000010000005 |   10.1.1.15   |
| 000010000004 |   10.1.1.4   | 000010000002 |    10.1.1.5   |
| 000010000003 |  10.1.1.11   | 000010000004 |   10.1.1.10   |
| 000010000004 |  10.1.1.13   | 000010000006 |   10.1.1.12   |
| 000010000003 |   10.1.1.9   | 000010000005 |    10.1.1.8   |
| 000010000001 |   10.1.1.3   | 000010000003 |    10.1.1.2   |
| 000010000002 |   10.1.1.0   | 000010000001 |    10.1.1.1   |
| 000010000006 |  10.1.1.12   | 000010000004 |   10.1.1.13   |
| 000010000003 |   10.1.1.2   | 000010000001 |    10.1.1.3   |
| 000010000005 |   10.1.1.8   | 000010000003 |    10.1.1.9   |
| 000010000005 |  10.1.1.15   | 000010000006 |   10.1.1.14   |
| 000010000002 |   10.1.1.7   | 000010000005 |    10.1.1.6   |
| 000010000004 |  10.1.1.10   | 000010000003 |   10.1.1.11   |
| 000010000001 |   10.1.1.1   | 000010000002 |    10.1.1.0   |
+--------------+--------------+--------------+--------------+
tesuto@dev2:~$
```

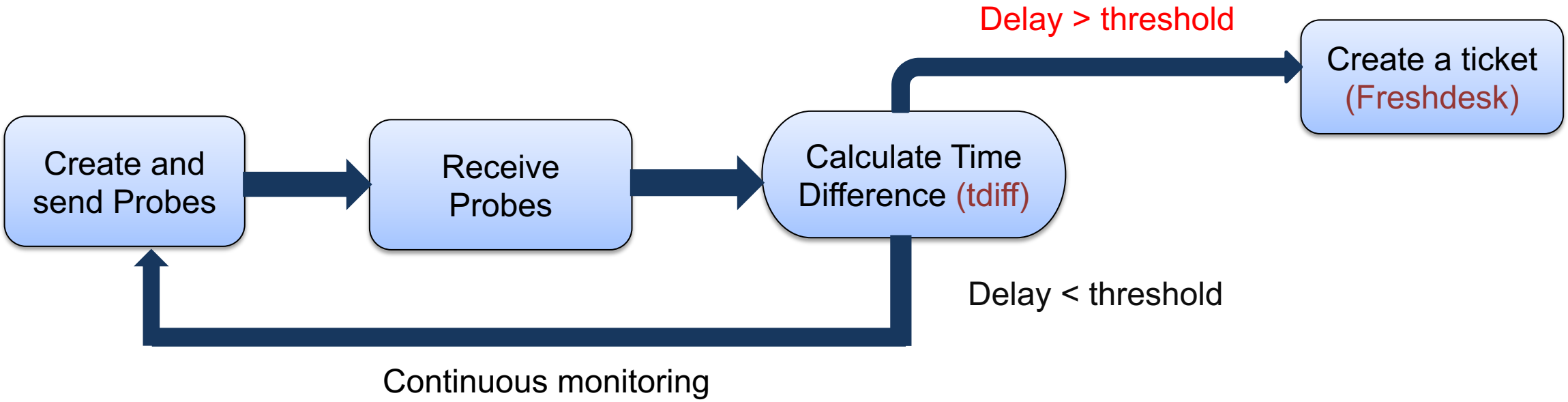
Topology

- Fetched all possible paths from end points
- Tool used: NetworkX

```
tesuto@agent:~$ python3 test.py
['dev1', 'R1', 'R3', 'R5', 'R2', 'R4', 'R6', 'dev2']
['dev1', 'R1', 'R3', 'R5', 'R6', 'dev2']
['dev1', 'R1', 'R3', 'R4', 'R2', 'R5', 'R6', 'dev2']
['dev1', 'R1', 'R3', 'R4', 'R6', 'dev2']
['dev1', 'R1', 'R2', 'R5', 'R3', 'R4', 'R6', 'dev2']
['dev1', 'R1', 'R2', 'R5', 'R6', 'dev2']
['dev1', 'R1', 'R2', 'R4', 'R3', 'R5', 'R6', 'dev2']
['dev1', 'R1', 'R2', 'R4', 'R6', 'dev2']
```



Workflow


Delay > threshold
Create a ticket
(Freshdesk)
Create and
send Probes
Receive
Probes
Calculate Time
Difference (tdiff)
Delay < threshold
Continuous monitoring

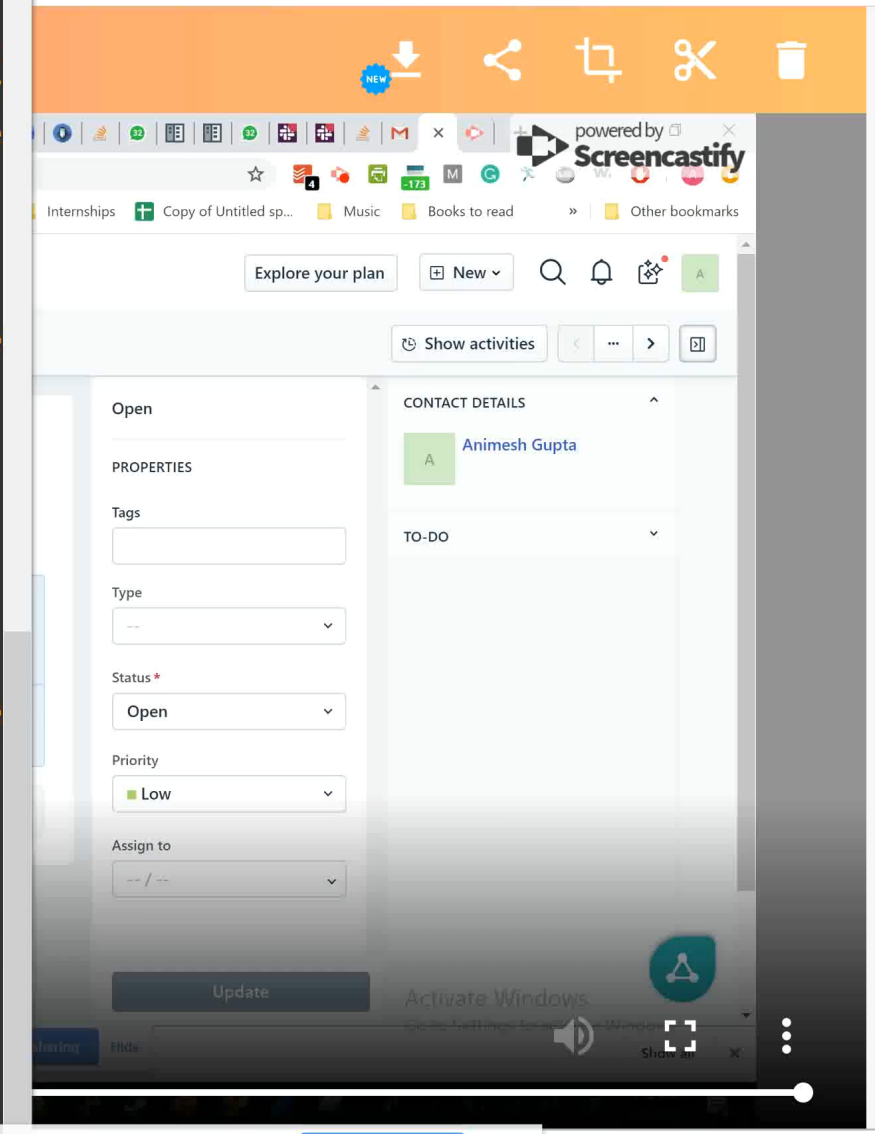
Probe creation

- Create IP GRE headers to trace all the paths.
- Used SCAPY to create the packets.
- Send the packet from source to destination.

```
from struct import pack,unpack
from scapy.all import *
#from time import *
import time
s = conf.L3socket(iface="eth1")
packetSentCount=1

for i in range (packetSentCount):
    p1 = IP(src='20.0.0.2', proto=47,tos=0, dst='20.0.0.1',version=4,ttl=64)
    p2=GRE()
    p3=IP(src='20.0.0.2', proto=47, dst='10.1.1.1')
    p4=GRE()
    p5=IP(src='20.0.0.2', proto=47, dst='10.1.1.5')
    p6=GRE()
    p7=IP(src='20.0.0.2', proto=47, dst='10.1.1.13')
    p8=GRE()
    p9=IP(src='20.0.0.2', proto=47, dst='20.0.0.6')
    p10=GRE()
    p11=IP(src='20.0.0.2', proto=17, dst='20.0.0.2')

    p12=UDP(dport=3000, sport=3000)
    #payload=pack('dlll',time.time(), i, 0 ,packetSentCount)
    #payload=pack('l',0)
    value = time.time()

    payload= str(value)
    packet=p1/p2/p3/p4/p5/p6/p7/p8/p9/p10/p11/p12/payload
    #packet=p1/p2/p11/p12/payload
    s.send(packet)
```

```
import smtplib

def print_return_packet_details(x):
    #sendTime,seq,probe,total=unpack('dlll',x.load)
    #print (x.Time, sendTime, seq, probe, total)

    print(x.load)
    print(x.time)
    threshold = 1
    difference = float(x.time)-float(x.load)
    if (float(x.time)-float(x.load)) < threshold:
        print("Delay in this route is higher than threshold. Creating a new ticket for debugging!")


        fromaddr = 'animeshgupta720@gmail.com'
        toaddrs  = 'support@animeshgupta720help.freshdesk.com'
        msg = 'There is a high latency on the R1-R2-R4-R6 route. Please take necessary action.'


        server = smtplib.SMTP("smtp.gmail.com:587")

        server.starttls()
        username='animeshgupta720@gmail.com'
        password=[redacted]

        server.login(username,password)

        server.sendmail(fromaddr, toaddrs, msg)
        #server.quit()
```

```
    server.login(username,password)
  File "/usr/lib/python2.7/smtplib.py", line 623, in login
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (534, '5.7.14 <https://acc
vGuqSrJrRnsJuY0I_JUsAHRhVQPF6D4FF7628Alc6n9Huzg6oSjj53wHGS7
87-yQVh> Please\n5.7.14 log in via your web browser and the
/mail/answer/78754 g41sm2326496uah.12 - gsmtp')
tesuto@dev1:~$ sudo python test4.py
1560115679.91
1560115679.98
Delay in this route is higher than threshold. Creating a ne
tesuto@dev1:~$ cat testing.py ^C
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$ sudo vi test4.py
tesuto@dev1:~$ sudo python test4.py
1560115968.87
1560115968.93
Delay in this route is higher than threshold. Creating a ne
w ticket for debugging!
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$ sudo python test4.py
```



Comments *Learn more*

Comments appear next to your videos when you play them on Google Drive.

Allow others to comment

NEW COMMENT

Uploaded To

Google Drive

COPY LINK VIEW ON DRIVE

Video Details

00:03

3840×2160

Jun 9, 20195:34 PM

965.0 KB

Screencastify - Screen Video Recorder is sharing your screen. Stop sharing Hide

Show all

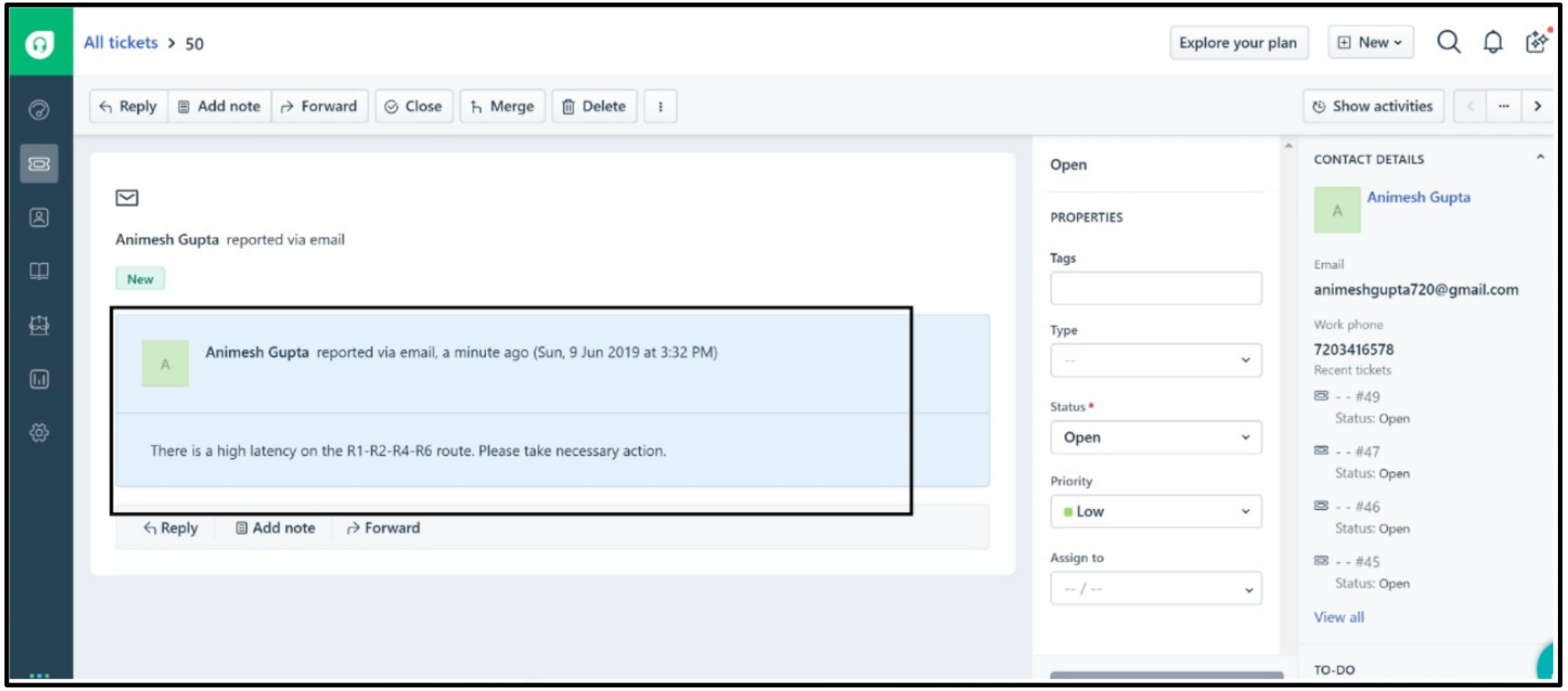
5:35 PM 6/9/2019

```
tesuto@dev1: ~
87-yQVh> Please\n5.7.14 log in via your web browser and the
/mail/answer/78754 g41sm2326496uah.12 - gsmtp')
tesuto@dev1:~$ sudo python test4.py
1560115679.91
1560115679.98
Delay in this route is higher than threshold. Creating a ne
tesuto@dev1:~$ cat testing.py ^C
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$ sudo vi test4.py
tesuto@dev1:~$ sudo python test4.py
1560115968.87
1560115968.93
Delay in this route is higher than threshold. Creating a ne
w ticket for debugging!
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$ sudo python test4.py
1560116123.1
1560116123.15
Delay in this route is higher than threshold. Creating a ne
w ticket for debugging!
tesuto@dev1:~$
```

```
tesuto@dev1: ~
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$
tesuto@dev1:~$ sudo python test3.py
tesuto@dev1:~$
tesuto@dev1:~$ sudo python test3.py
tesuto@dev1:~$
```

Report Failure

- Raised ticket and reported issue by integrating our script with a ticketing tool (Freshdesk)



Future Scope

- **Scalability**

  Scale this to a higher number of nodes

- **Traffic Engineering**

  Reroute traffic to different paths as per issue reported

- **Visualization**

  Use data visualization tools such as Grafana to create dashboards

Takeaway

- Tools – Scapy, NetworkX, Python, ExaBGP
- Power of Network Programmability
- Planning
- Team work
- Don’t give up!

Team Members

- Animesh Gupta (animesh.gupta@colorado.edu)
- Vibhum Chandorkar (vibhum.chandorkar@colorado.edu)
- Ameya Korgaonkar (ameya.korgaonkar@colorado.edu)
- Jose Dahlson Irenish Kumar (joir9977@colorado.edu)
- Apurva Bhiwapurkar (Apurva.bhiwapurkar@colorado.edu)

THANK YOU ☺

>_

PICON

Anand Oliver Ryan Taylor

NANOG76 Active Monitoring - Picon

Common issues:

- Islands of console capability
- Poor documentation
- Little or no proactive testing
- Pain when needed but not present

Open Hyperterm..... It's in the start menu.....
the button in the lower corner of your screen

The Solution

Open-source self-discovery framework

- Commodity hardware
- Operate over untrusted network
- Single view of all capabilities
- Pick and implement advanced features over time
    - Concurrency, logging of console output, point-and-click access

All you need

Less than $50 USD!

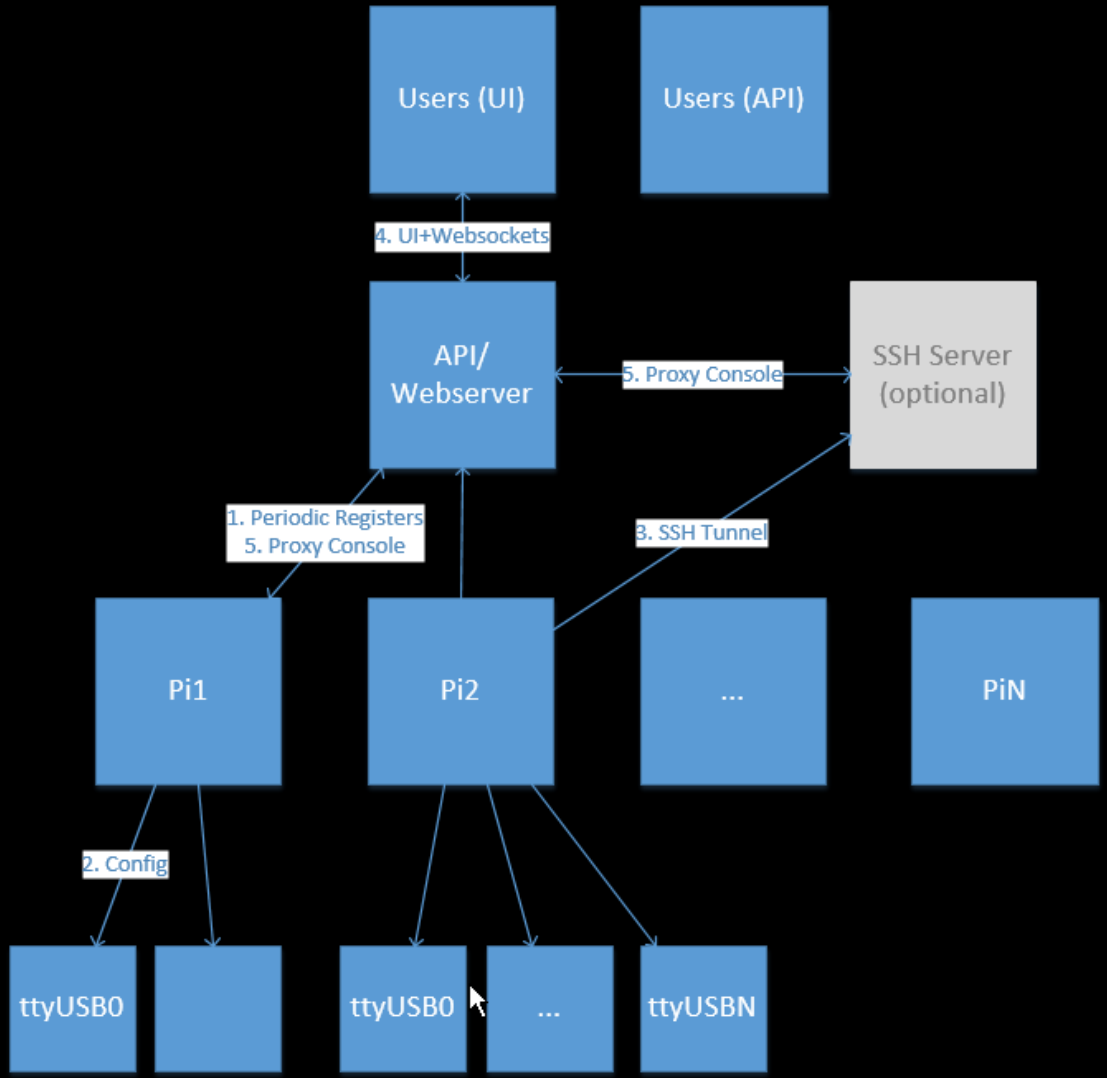
USB Serial Adapter

Raspberry Pi 3

```
{
    'addrs': ['2001:DB8'],
    'ports': ['ttyUSB0',
              'ttyUSB1']
}
```

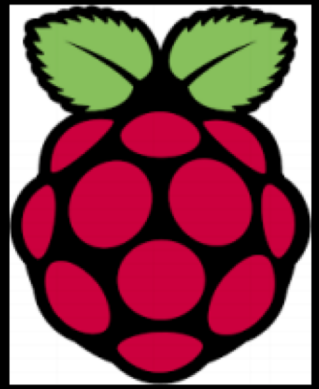
3G/4G, Wifi, Ethernet

PICON


Users (UI)
Users (API)
4. UI+Websockets
API/
Webserver
5. Proxy Console
SSH Server
(optional)
1. Periodic Registers
5. Proxy Console
3. SSH Tunnel
Pi1
Pi2
...
PiN
2. Config
ttyUSB0
ttyUSB0
...
ttyUSBN

Why Pi?

- Inexpensive
- Linux based
- Puppet, Chef, Anisible
- Wide range of hardware/hack support
- Linux Authentication (kerberos, ssh keys, etc)
- 1G Ram, SD Card Storage, decent CPU



Initial Work

- Python3, Flask, Bootstrap
- Webserver+API
    - API for registration, UI for device list
- Self-registering agent
    - Reported available TTY, interfaces, addresses
    - SSH to central server

Next Steps

- Additional Web and UI Capabilities (configurability)
- Programmatic console access
- In-browser console access
- Proxy agent for joining other consoles (Cisco, Digi, Avocent, etc)
- Physical configurations for deployment such as cases, custom FTDI daughterboards, etc

```
(picon-venv) picon@picon:~/picon/agent $ python3 agent-bootstrap.py -vvv http://172.28.10.2:5000/api
2019-06-09 22:56:11,855 INFO: Starting PiCon Agent...
2019-06-09 22:56:11,856 INFO: Using endpoint http://172.28.10.2:5000/api/s, holdtime 300s, reporting interval 60
2019-06-09 22:56:11,925 DEBUG: Starting new HTTP connection (1): 172.28.10.2:5000
2019-06-09 22:56:12,010 DEBUG: http://172.28.10.2:5000 "POST /api/register HTTP/1.1" 200 88
2019-06-09 22:56:12,013 INFO: Successfully registered with endpoint http://172.28.10.2:5000/api/register
2019-06-09 22:56:12,013 DEBUG: Sent JSON in POST body:
{
  "holdtime": 300,
  "hostname": "picon",
  "interfaces": {
    "eth0": {
      "addrs": [
        "172.28.10.5"
      ],
      "state": true
    },
    "lo": {
      "addrs": [],
      "state": false
    },
    "wlan0": {
      "addrs": [
        "199.187.221.226",
        "2620:0:ce0:101:ba27:ebff:fe43:b34c"
      ],
      "state": true
    }
  },
  "ports": [
    "ttyUSB0"
  ],
  "sn": "000000003316e619"
}
2019-06-09 22:56:12,014 DEBUG: Received JSON in POST response:
{
  "status": "ok",
  "tunnel": {
    "port": 2220,
    "server": "172.28.10.3"
  }
}

2019-06-09 22:56:12,015 INFO: Starting SSH tunnel connection to 172.28.10.3:2220
```

PICON Home Devices

Hostname	Serial	Addresses	Ports	First seen	Last seen	Status
picon	000000003316e619	**eth0** 172.28.10.5 **wlan0** 199.187.221.226 2620:0:ce0:101:ba27:ebff:fe43:b34c	ttyUSB0 *fw-1* ttyUSB1 *available/future use*	2019-06-09 19:52:42.452234	2019-06-09 21:46:44.543895	

picon

Serial Number
0000000003316e619
First Seen
2019-06-09 19:52:42.452234
Last Seen
2019-06-09 21:45:44.276528
Tunnel Details
Port: 2220

Network Interfaces

Interface Name	State	Addresses
eth0	1	172.28.10.5
wlan0	1	199.187.221.226 2620:0:ce0:101:ba27:ebff:fe43:b34c

Serial Ports

Port name	
ttyUSB0	fw-1 ✔ ✖
ttyUSB1	available/future use

PICON Home Devices

Hostname	Serial	Addr	Last seen	Status
picon	000000003316e619	eth0 172.2 wlan 199.1 2620	2019-06-09 21:47:44.775380	

Console for port ttyUSB0 on host picon

Refresh | Request Exclusive

User <websockets.server.WebSocketServerProtocol object at 0x7fa7a30f5710> requested Exclusivity

3 users connected

Close | Save changes

Accomplishments

- Team gained valuable experience with:
  - Websockets package and async workflows
  - Interacting with tty via python serial module
  - More features of Bootstrap and rusty jquery skills
- Various UI tweaks and progress
- Websockets POC is embedded in codebase

Take-aways

Flask and Bootstrap simplify non-developer progress

Websockets and asynchronous IO do not

Trying to implement the UI workflow will take much more planning

Less-technical takeaways from the challenge

Future Work/Next Steps

- Integrate Websockets with console access
- Basic authentication and privilege system
- Identify hardware for POC deployment
- Extend/abstract configuration
- Proxy agent

Thanks

https://github.com/piconsole/picon

Thanks Tesuto, Oracle and NANOG for this event