
This graphic is dope, Thanks and Credit to Demolabs/whomever designed it! Thank you!. :-)


FORENSICS, NETWORKS, DEFENSE

---

# ❄ PCAPXRAY ❄

Srinivas Piskala Ganesh Babu

# Agenda 

PLS FEEL FREE TO REFER THE FLOW DIGRAM PLACED

- Motivation? 🤩
- Why? 🧐
- What? 😮
- Wait Whaat? 😐
- Ok, How? 🤔
- Demo 🤓

Motiv?

# Motivation

- Chance to be a detective 🕵️‍♀️ 🕵️ (play sherlock)
- Uncover hidden events/info 🔍
- What happened? Who did what? What did they take? 👀
- Personally, solving Ctfs/ Puzzles - Image/File, Network, Memory
- Seek more puzzles at ctftime, https://www.netresec.com/?page=PcapFiles, https://wiki.wireshark.org/SampleCaptures, packet total


# The Forensics Problem

- Huge Data points to look at ? Sequence of events ? 🏋️
- Convergence or Pattern recognition ? 🔎
- Privilege of “incident" recon ( no revisiting/ live info ) ? 🎬
- Learn as much from the given asset 📚
- Takes Time !!! ⏱️


# Network Forensics Problem

## Asset: Packet Capture (Pcap)

- Create a way to understand the network 👩‍💻
- Identify Events/Pattern
  - Possible covert 👽
  - Possible malicious 👾
  - Tor 🦹🦹
- Different Protocols and payloads 💼

Hmm?

# Ok, What do we need?

## Asset: Packet Capture (Pcap)

*Given an asset, we need*

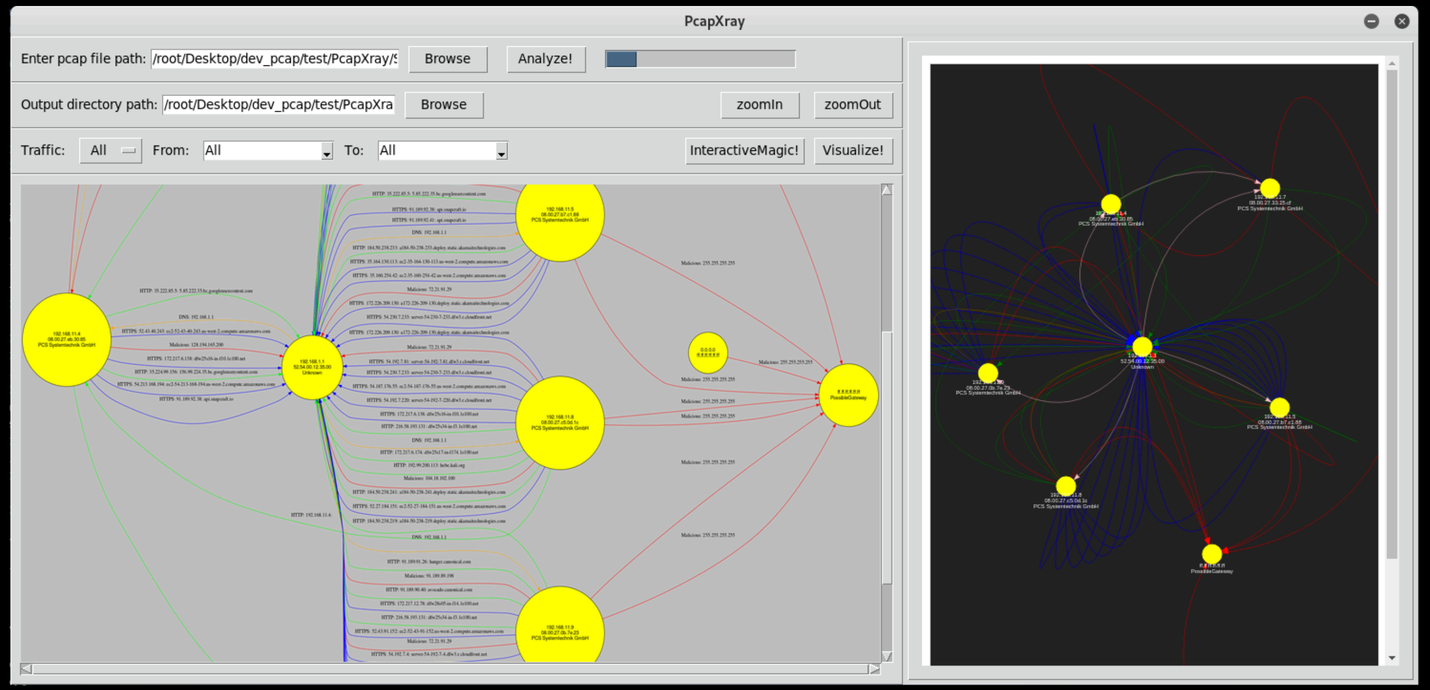
- Navigation / Map 🗺️
- UI/ Understand/Drawing 👁️
- Detection / Pattern / Highlight 🚥
- Speed 💨
- Summary/ Results 📜
- Interaction 🖐️
- Could not find something that does this 🤷

# What?

# PcapXray

## Asset: Packet Capture (Pcap)

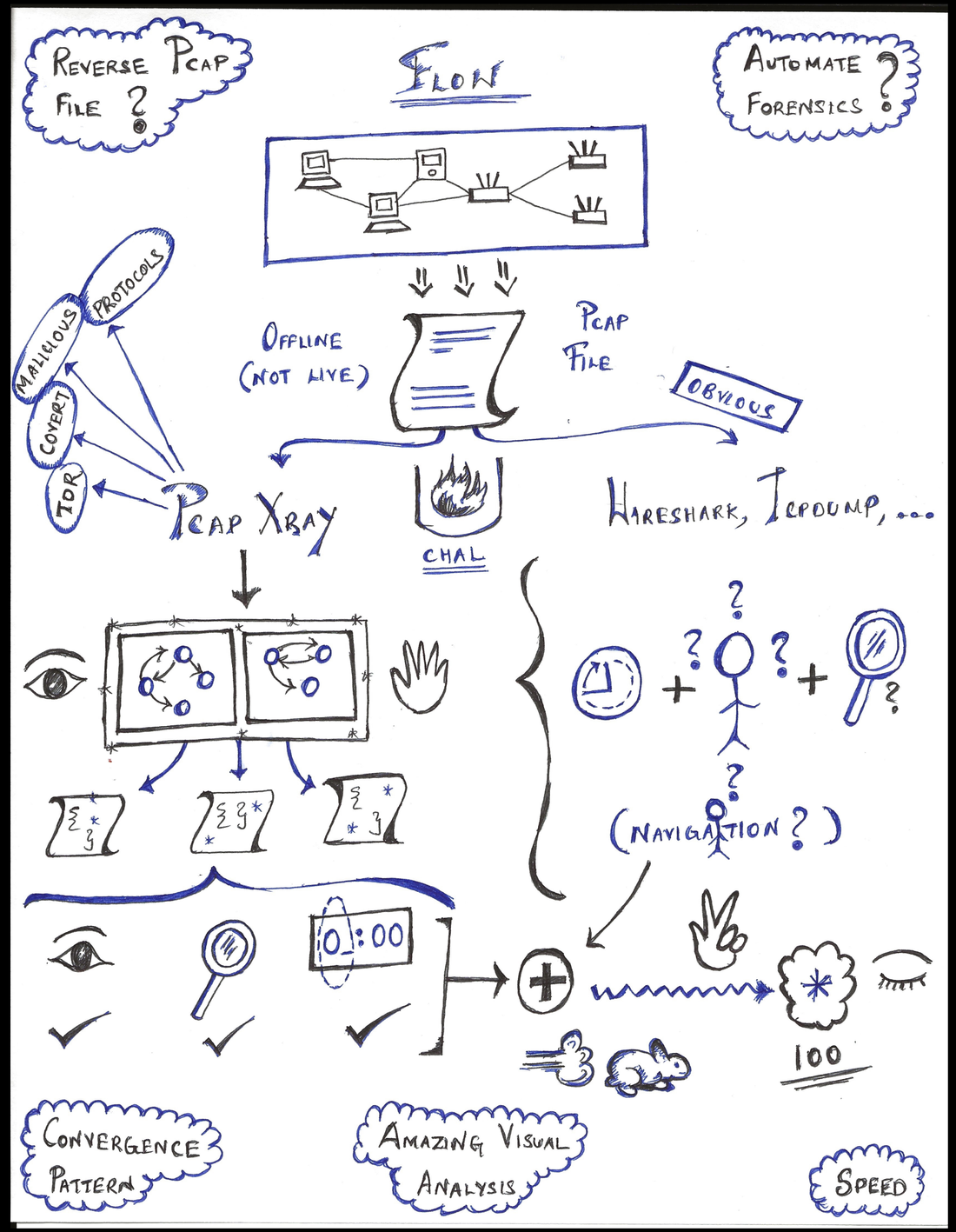
- Transforms the asset to a 🚀
  - drawing (reverse a pcap) 🎨
  - Interactive data 🖐️
- Highlight / Segregate / Detect patterns or traffic 🚦
- Generate full duplex flow map with payloads ( identify hosts ) - 🧳
- Still improvising from being a prototype. 🔥, built based on forensics challenges



Wait Whaat?

# Flow: Go Hybrid

- Not any replacement but making it easier/ faster !, taking a less obvious path
- Wireshark, TcpDump remain the best goto for analysis.



Ok, How?

# Bolt and Nuts

PcapFile

Trigger (main)

TK Widget

{ Frontend }

Json DB

PcapReader (engine) module

Plot map

Communication Details

Device Details

Malicious

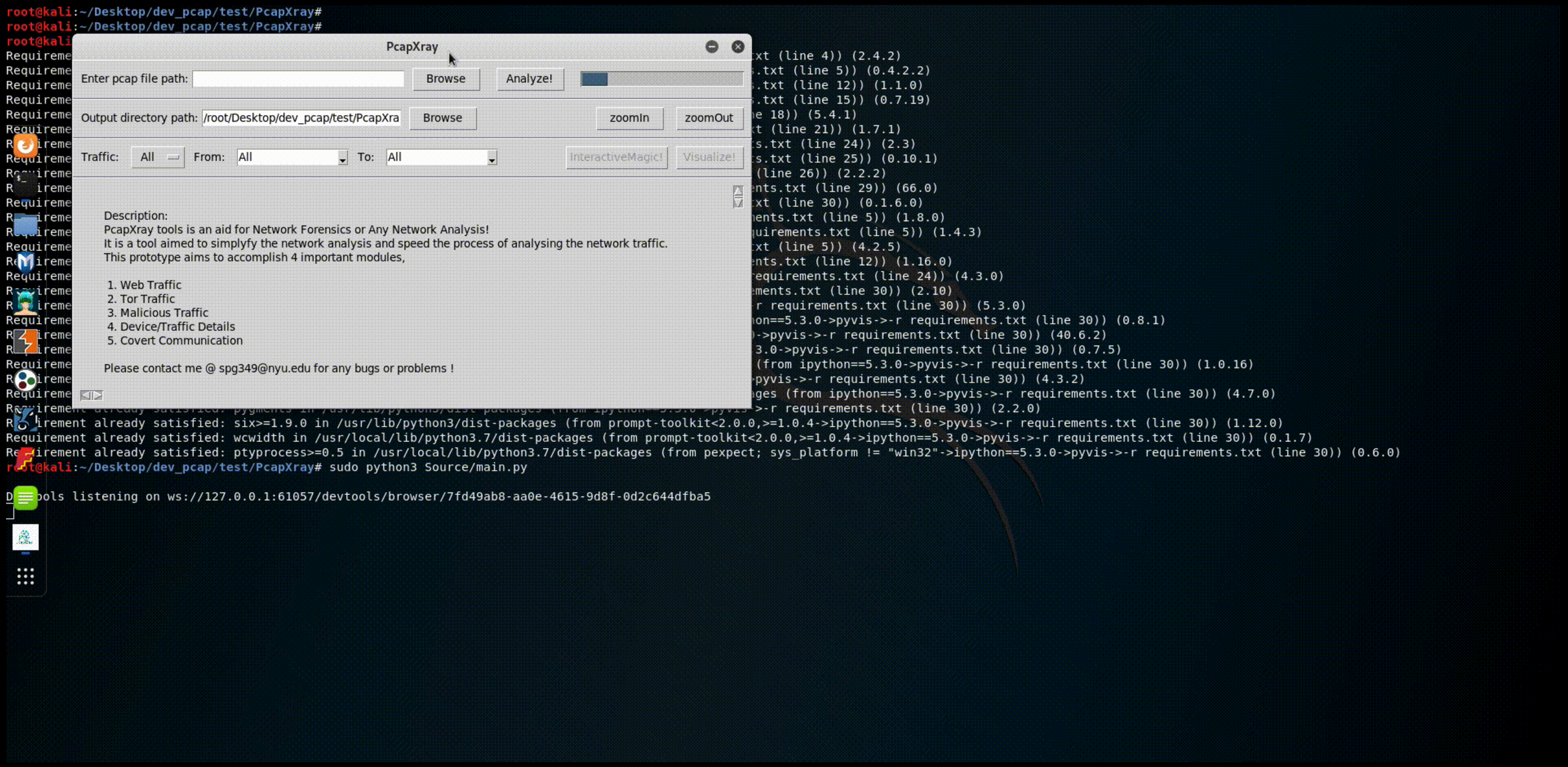
Tor

Covert

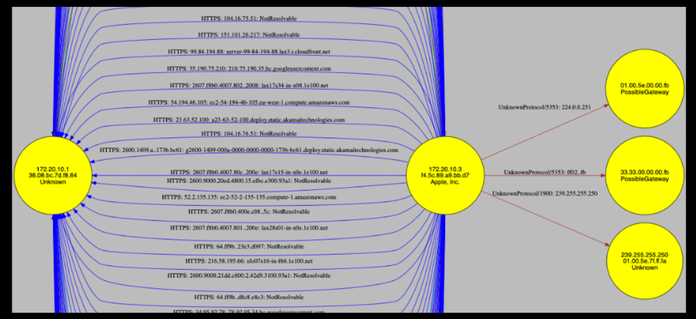
Interactive Graph Render

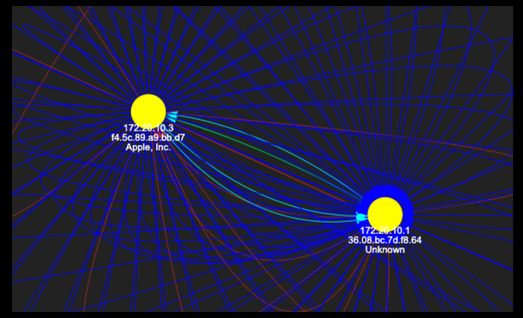
{ Backend }

# Demo


root@kali:~/Desktop/dev_pcap/test/PcapXray#
root@kali:~/Desktop/dev_pcap/test/PcapXray#
root@kali:~/Desktop/dev_pcap/test/PcapXray#
PcapXray
Enter pcap file path:
Browse
Analyze!
Output directory path: /root/Desktop/dev_pcap/test/PcapXra
Browse
zoomIn
zoomOut
Traffic: All
From: All
To: All
InteractiveMagic!
Visualize!
Description:
PcapXray tools is an aid for Network Forensics or Any Network Analysis!
It is a tool aimed to simplyfy the network analysis and speed the process of analysing the network traffic.
This prototype aims to accomplish 4 important modules,
1. Web Traffic
2. Tor Traffic
3. Malicious Traffic
4. Device/Traffic Details
5. Covert Communication
Please contact me @ spg349@nyu.edu for any bugs or problems !
Requirement already satisfied: six>=1.9.0 in /usr/lib/python3/dist-packages (from prompt-toolkit<2.0.0,>=1.0.4->ipython==5.3.0->pyvis->-r requirements.txt (line 30)) (1.12.0)
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages (from prompt-toolkit<2.0.0,>=1.0.4->ipython==5.3.0->pyvis->-r requirements.txt (line 30)) (0.1.7)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.7/dist-packages (from pexpect; sys_platform != "win32"->ipython==5.3.0->pyvis->-r requirements.txt (line 30)) (0.6.0)
root@kali:~/Desktop/dev_pcap/test/PcapXray# sudo python3 Source/main.py
ools listening on ws://127.0.0.1:61057/devtools/browser/7fd49ab8-aa0e-4615-9d8f-0d2c644dfba5

# Output




- PNG of the drawing / HTML of the interactive map
- **Report**
  - Packet Details
  - Device Details
  - Communication Details

```
src/dst/port : {
   "Ethernet": {
      "dst": "",
      "src": ""
   },
   "Payload": {
      "forward": [ “” ],
      "reverse": [ “” ]
   },
   "covert": false,
   “file signature”: []
}
```

```
deviceDetails: {
  "<Mac>": {
    "device_vendor": "",
    "ip": "",
    "node": "",
    "vendor_address": [
      "",
    ]
  }
}
```

```
Tor Traffic: []

Malicious Traffic: []

Destination DNS: {
  "<IP>": {
    "domain_name": "",
    "mac": ""
  },
}

Lan Hosts: {
  "<MAC>": {
    "device_vendor": "",
    "ip": "",
    "node": "",
    "vendor_address": [ “” ]
  }
}

Tor Nodes: []
```

# Known Limitations 

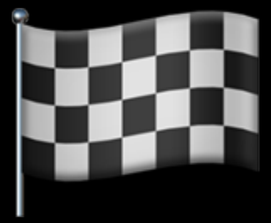
- Large Pcap Files 
- Increase speed 
- Threading and Tk UI 
- Graph becomes clumsy with large nodes 
- Code cleanup / restructure ( but it is extensible and modularized ) 

# 🙏 Credits 🙌

- Thanks for making it better
  - Professor Marc Budofsky
  - All the Library developers
  - Preethi ( Testing )

Fin! 🏁

# The End



- Thank 🙏 you for stopping by PcapXray
- Willing to use/help/contribute
  - Check out GitHub - https://github.com/srinivas11789/PcapXray

Questions/Feedback ?