# An SDN-Based NAC Implementation to Recognise Indicators of a Compromise via Malicious DNS Queries

Raymond C. Nunez
Paul Patrick C. Prantilla

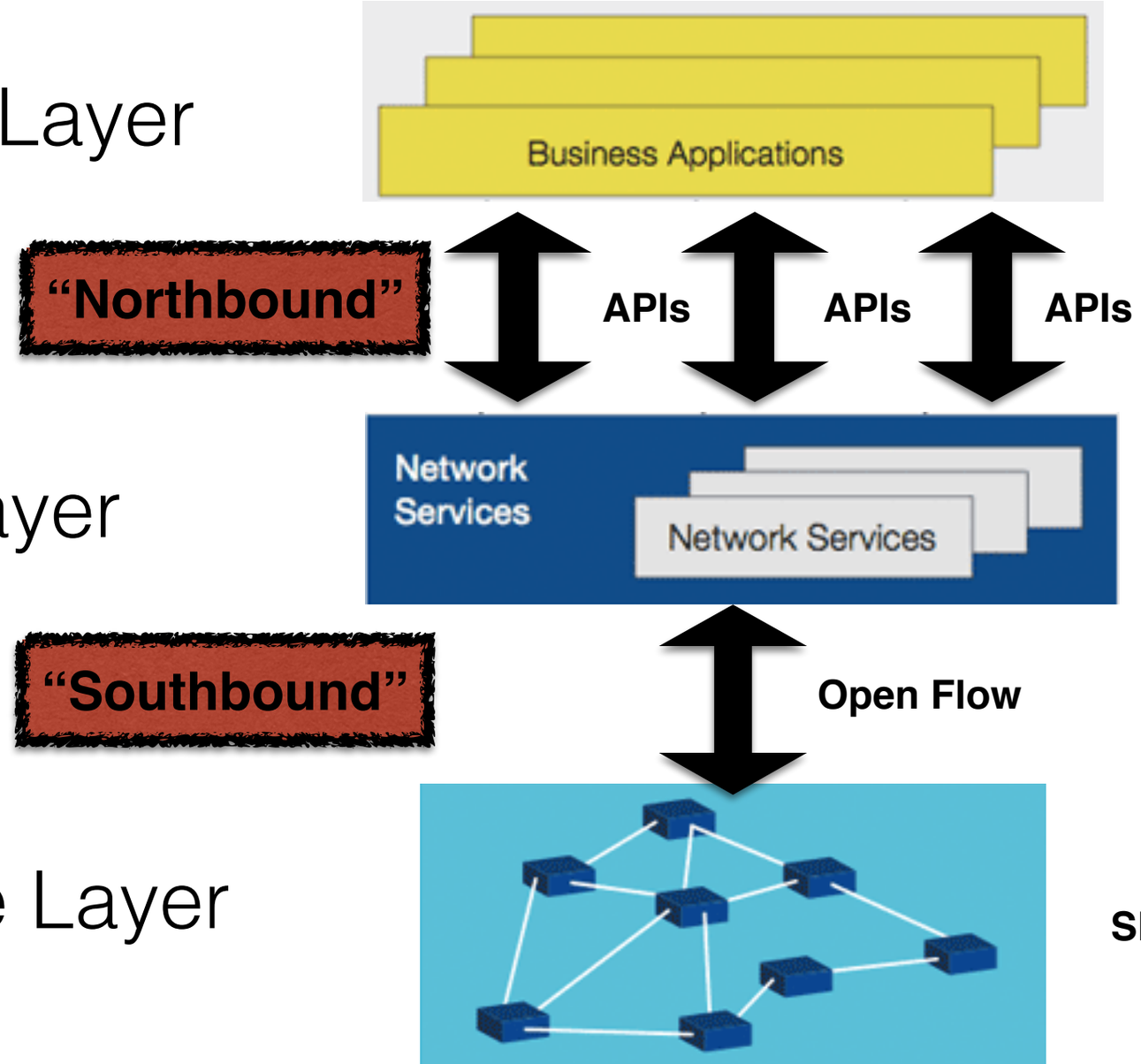
# What is this Project about?

- Use SDN to implement common Network Access Control (NAC) functionality that has can be easily extended.
- Implement using cheap machines (Raspberry Pis) and common SDN-compatible switches.
- Use optimizations to reduce operations and queries performed by the SDN controller when detecting compromises.

# What makes SDNs great?

Finally paves way for an official standard for decoupling the control layer from the forwarding layer in network devices.

Application Layer




**Full Applications**

Control Layer

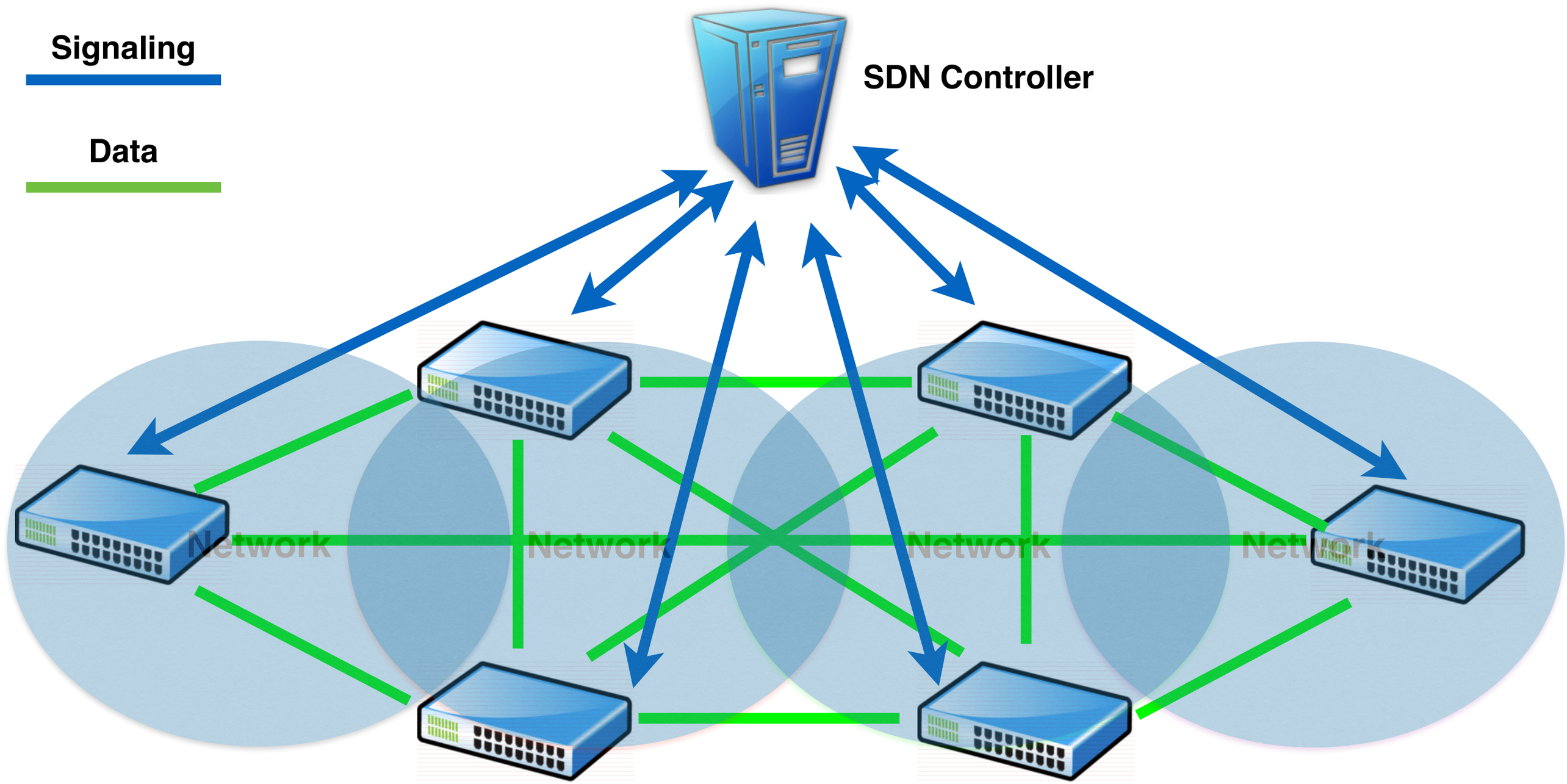
**DNS, VLANs, Firewalls Routers, etc**

Infrastructure Layer

**SDN Compatible Switches**

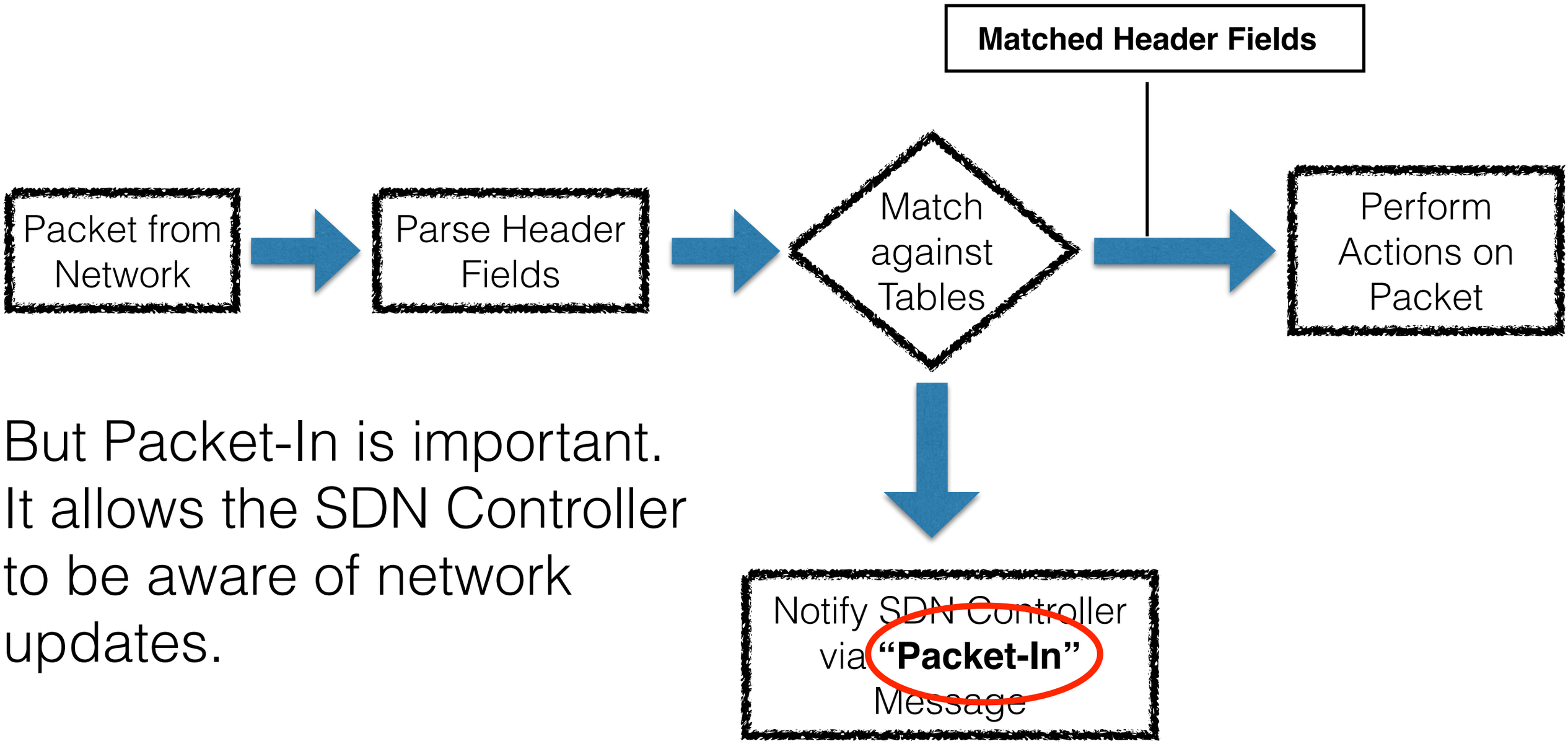
**Source : https://www.opennetworking.org**

# Basic SDN Topology



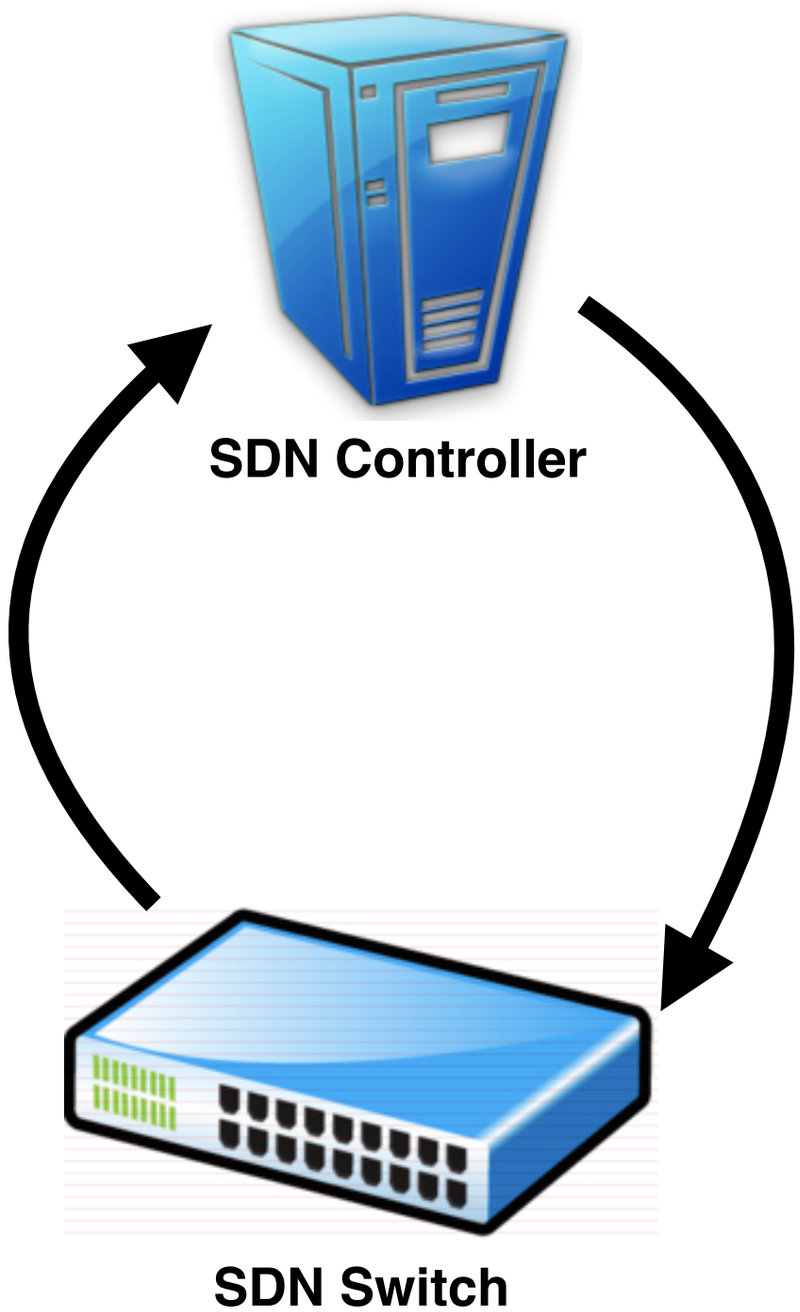
# The Problem of “Packet-In”

Vulnerable to DDOS using Packet-In or simple overuse. This leads to instability in the Control Plane.




But Packet-In is important. It allows the SDN Controller to be aware of network updates.

# Instability in the Control Plane



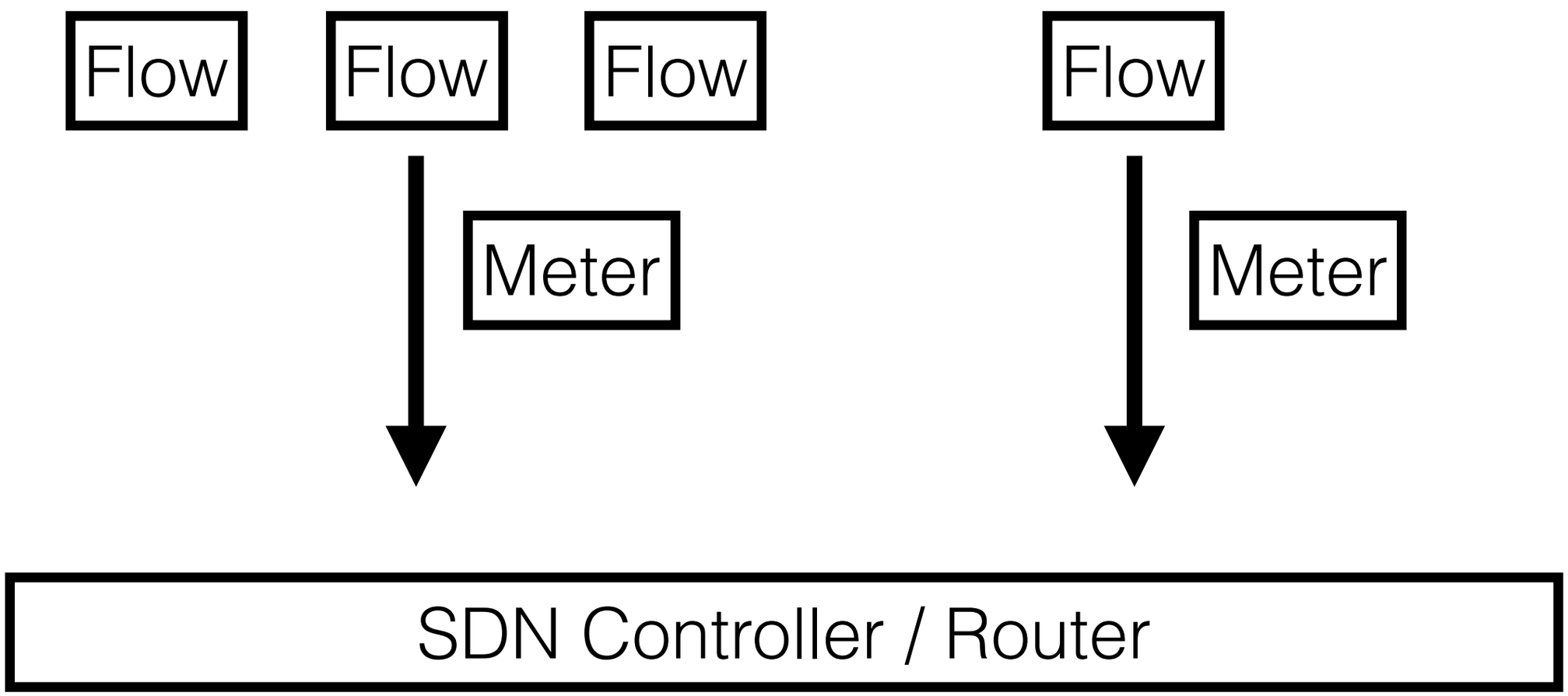

Packet-In

**x 9000!**

Insert, Delete or Modify Flow

Switch receive too many packets and send many Packet-In messages before the controller installs the new flow entries.

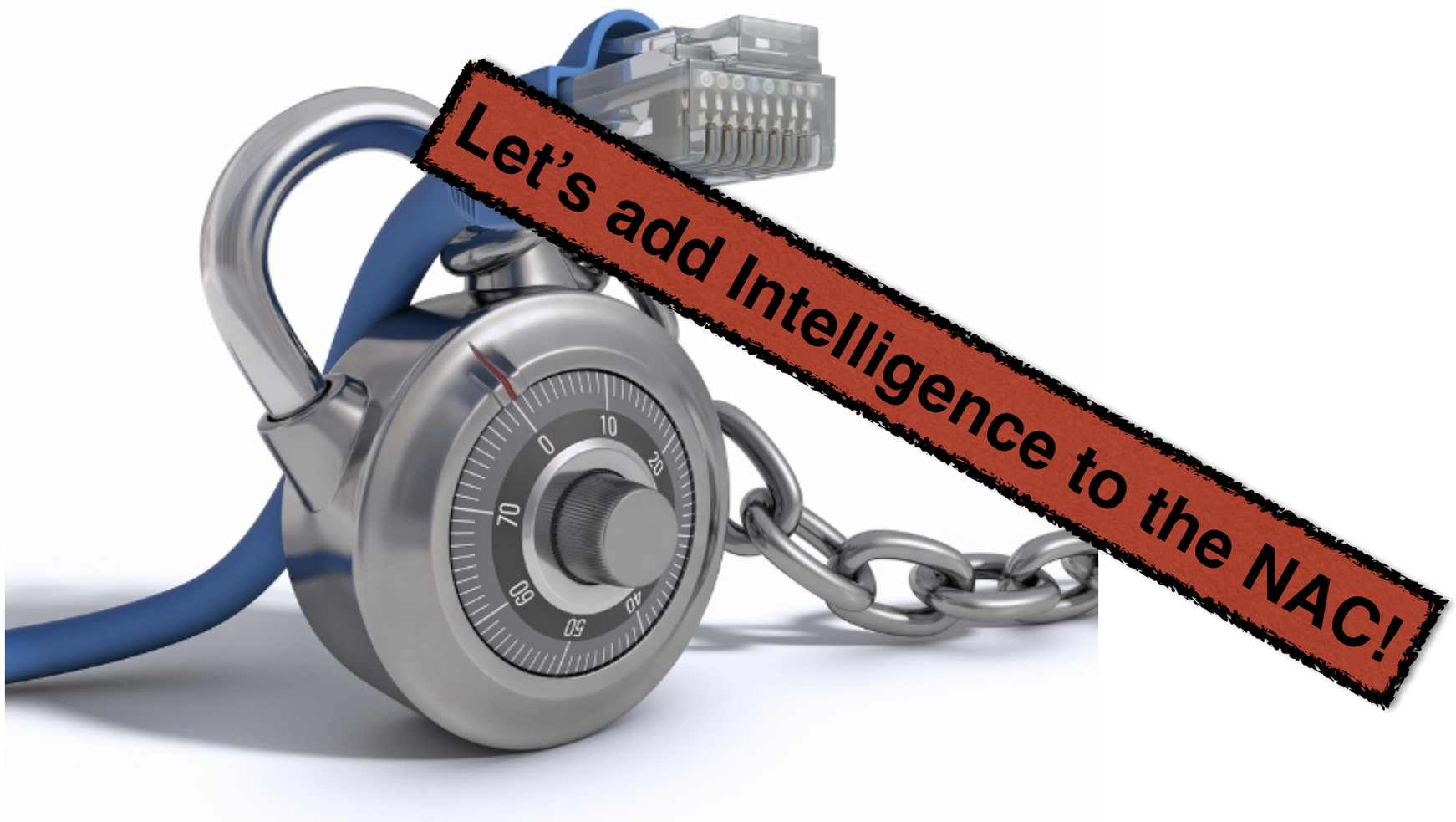
# Changes in OpenFlow 1.3 that Mitigates this Problem




The **“Meter”** limits the rate of packets in a group of flows by the number of packets or bytes. We can use **“Meter”** to limit the rate of a group of Packet-In messages.

# NAC in a Nutshell

- Controls entry into an access network
- Controls capability of nodes in an access network




http://www.truegen.be/

# What we wanted - The Short Story

Intercept DNS queries from network clients and forward the queries to the SDN Controller.

Look up the **A or CNAME** record in the DNS query using a local dictionary cache of blacklisted sites and/or online API.

If found…

Move the device that sent the DNS query to a remediation VLAN

# Implementation Approaches

Simulation

LAME!

Emulation

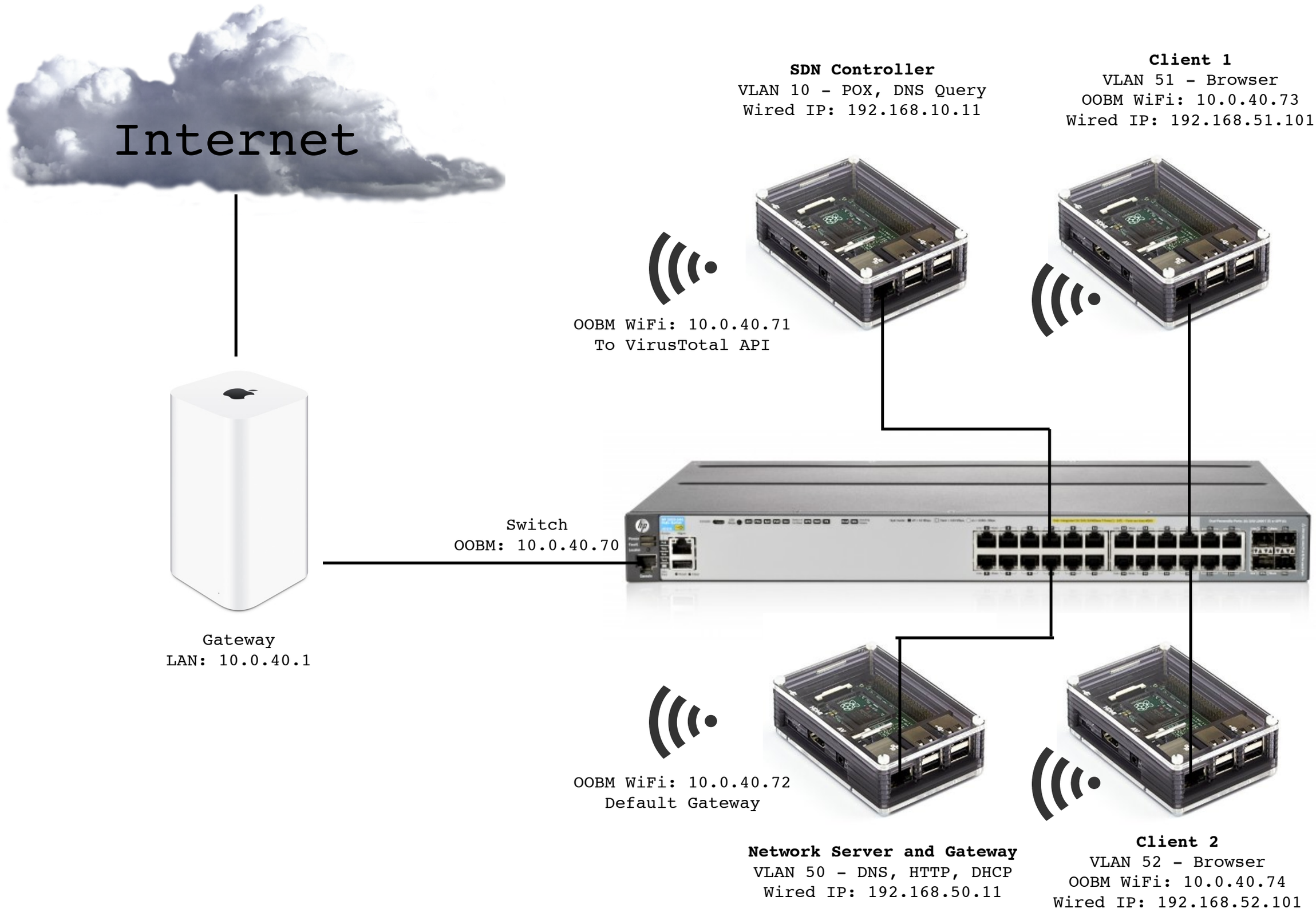
LAME!

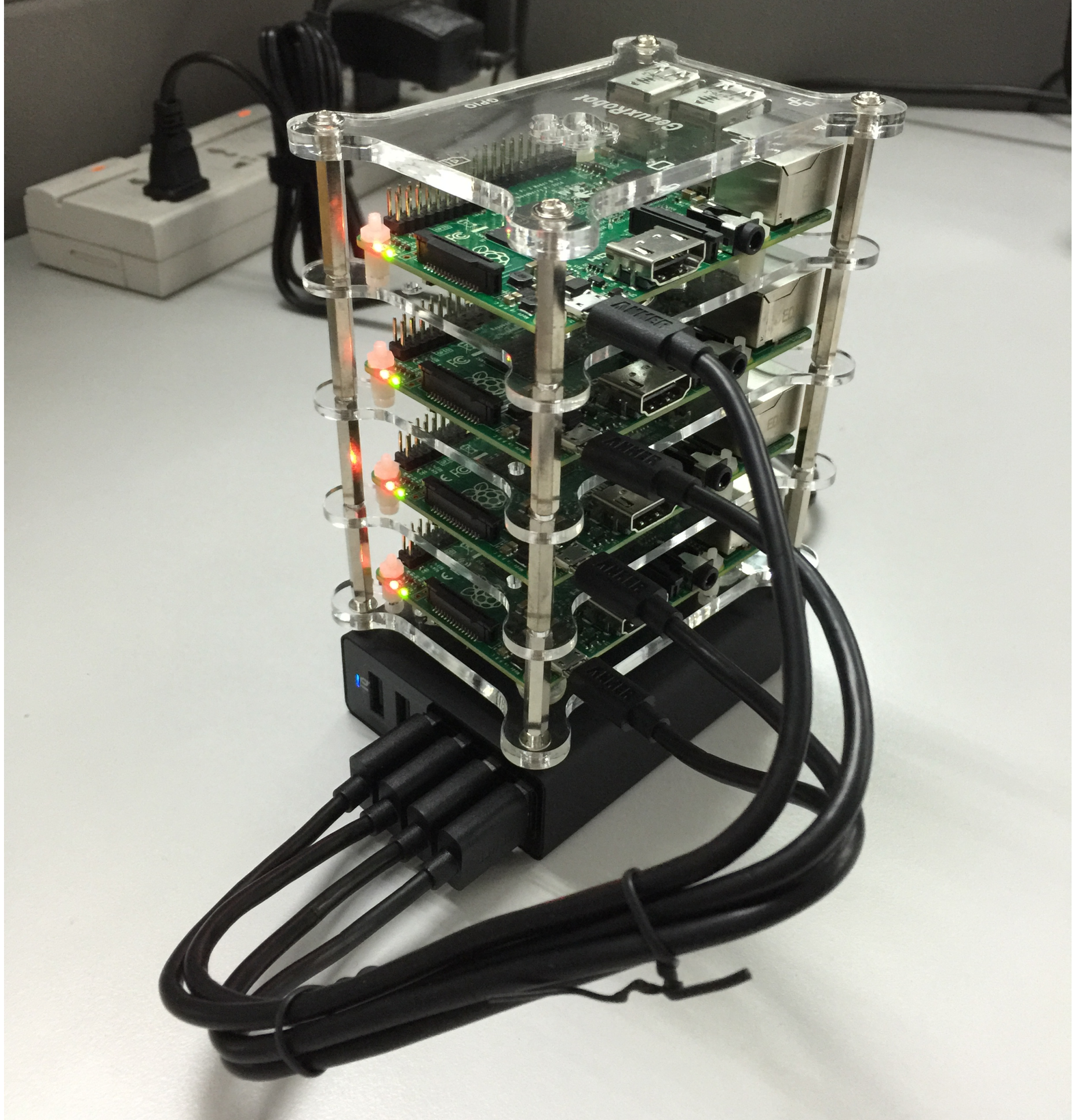
Real World

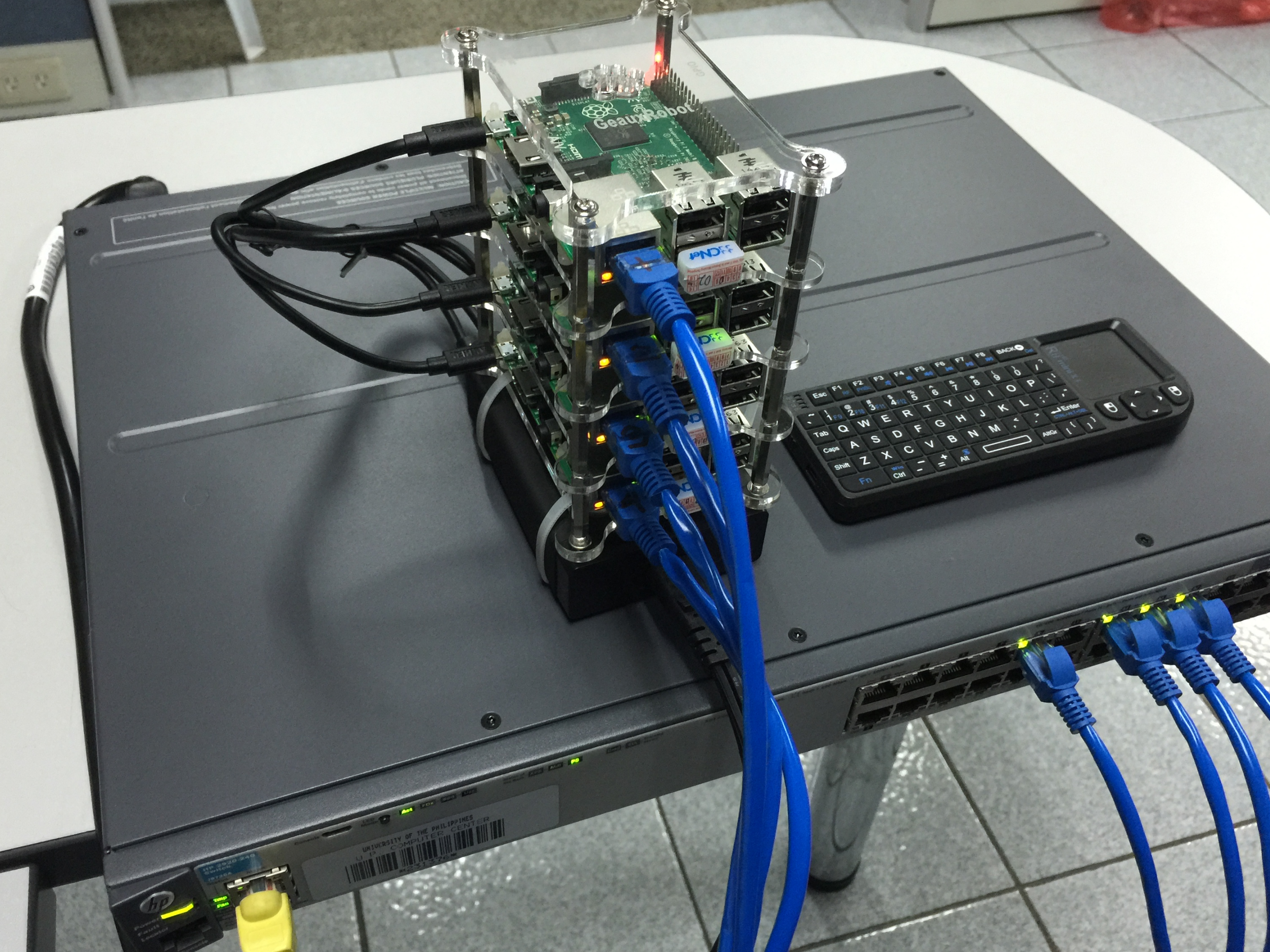
L33T!

# Our Setup

- HP 2960 OpenFlow Compatible Switch - Thank you UPCC
- 4x Raspberry Pi 2 (ARMv7 900MHz Quad Core - 1GB RAM)
  - SDN Controller
  - Server
  - Client 1
  - Client 2
- 6 Port 40Watt USB Charger
- Wireless Access point for OOBM


Internet
SDN Controller
VLAN 10 - POX, DNS Query
Wired IP: 192.168.10.11
Client 1
VLAN 51 - Browser
OOBM WiFi: 10.0.40.73
Wired IP: 192.168.51.101
OOBM WiFi: 10.0.40.71
To VirusTotal API
Switch
OOBM: 10.0.40.70
Gateway
LAN: 10.0.40.1
OOBM WiFi: 10.0.40.72
Default Gateway
Network Server and Gateway
VLAN 50 - DNS, HTTP, DHCP
Wired IP: 192.168.50.11
Client 2
VLAN 52 - Browser
OOBM WiFi: 10.0.40.74
Wired IP: 192.168.52.101


GeauxRobot
GPIO


GeauxRobot
UNIVERSITY OF THE PHILIPPINES
COMPUTER CENTER

# Demo

# Challenges

- HP Switch can support OpenFlow 1.3 but POX only supports OpenFlow 1.0
- There is no way to easily change VLAN IDs per port.
- There is no direct way to shutdown a port to re-trigger a DHCP lease.
- Virus Total is a constantly updated database of 63 AV solutions. But the Free VT API limits free users to 4 queries a minute

# Remediation VLAN Script

switchvlan.sh UNREGISTERED

switchvlan.sh

```bash
#!/bin/bash

PORT=$1
VLAN=$2
IPADDRESS="192.168.10.1"
PASSWORD="al*********"

echo "Changing $PORT to VLAN $VLAN"

(sleep 2; echo "$PASSWORD"; sleep 2; echo "x"; sleep; \
    echo "config terminal"; sleep 2; echo "vlan $VLAN"; sleep 1; \
    echo "untagged $PORT"; sleep 1; \
    echo "interface ethernet $PORT";sleep 1; \
    echo "disable"; sleep 15; echo "enable"; sleep 1; \
    echo "logout"; sleep 1; echo "y"; sleep 1; echo "n") | \
socat - EXEC:"ssh -l manager $IPADDRESS",pty,setsid,ctty

echo "Done"
```

Line 1, Column 1 | Tab Size: 4 | Shell Script (Bash)

# Experimental Setup

- State University Network
- Total Upstream bandwidth of 1.3 Gbps from 3 ISPs
- Dual Stack IPv4 and IPv6
- 15,000 - Users  (8,000 Peak Concurrent)

# Optimization - Whitelisting

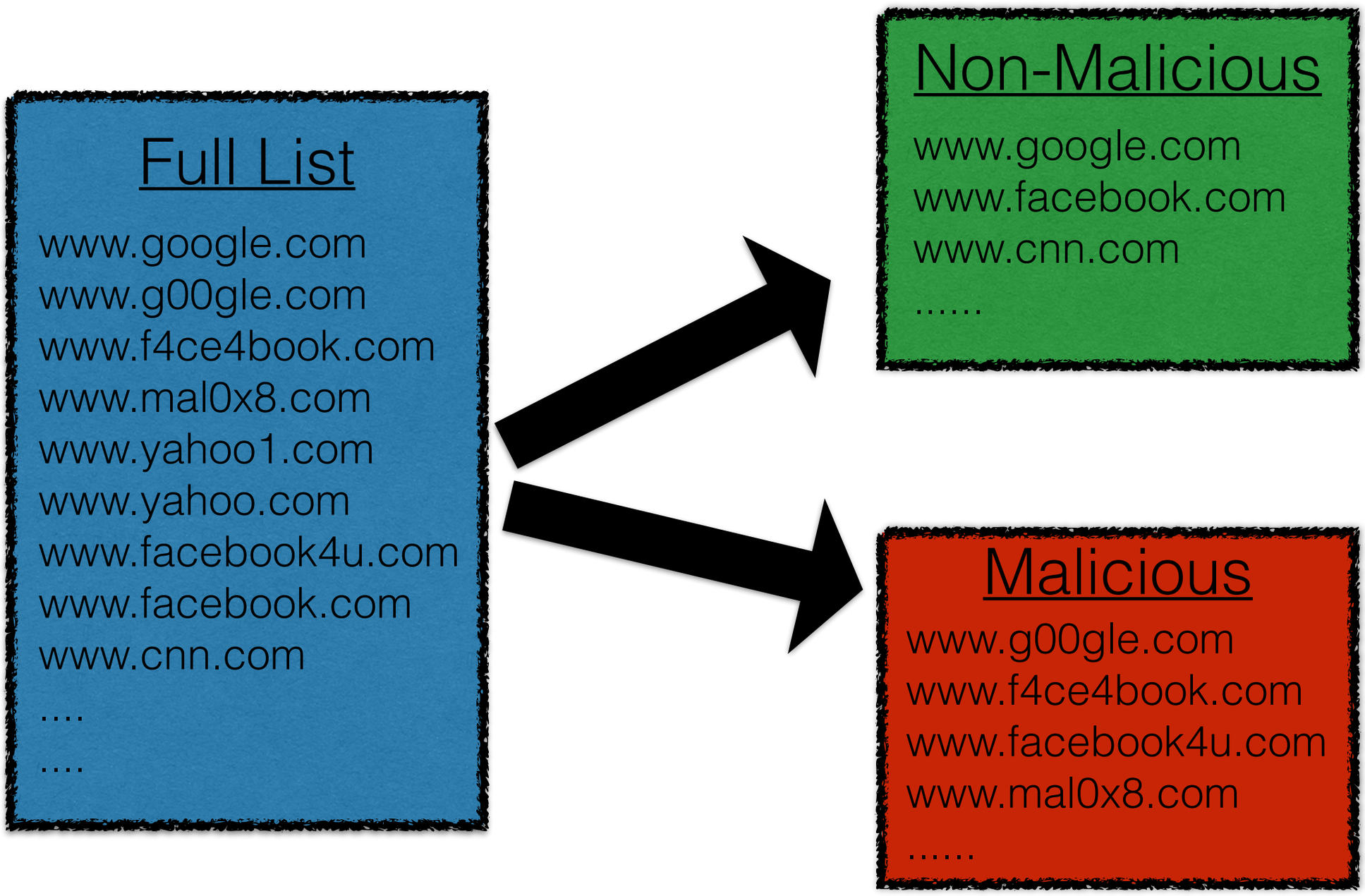
Whitelisting of the popular domains below would cut lookups by as much as 60%

| Domain | Page Views | Percentage |
|---|---|---|
| facebook.com | 70,000,000 | 43.7% |
| google.com | 10,000,000 | 6.4% |
| akamaihd.net | 8,000,000 | 5.2% |
| yahoo.com | 4,240,000 | |
| dropbox.com | 4,000,000 | 2.5% |

Over 60%!!!

# Optimization - Caching

Cache Domain Query results to a **malicious** and **non-malicious list.**



Thank you!