

Network Interrogation Device (NID) Version 2.5

Developed by: P. MacLaren

Licensed under GPLv3 <https://gnu.org>

<https://github.com/NID-Developer/pcmac.git>

Revisions: June 5, 2021

The concept for the NID arose from the investigative requisite of performing network interrogation for collecting artifacts to be included for the case investigator. The NID is a dual-homed device based on the Raspberry Pi3 B+ platform. Dual-homed refers to a computer with two or more network interfaces. The dual-homed NID has an Ethernet interface and a Wi-Fi access point interface. A Wi-Fi access point is a device that creates a wireless local area network, or WLAN. An access point connects to a wired router, switch or hub via an Ethernet cable, and projects a Wi-Fi signal to a designated area.

The NID is configured as a routed Wi-Fi access point. This secure wireless network is entirely managed by the NID. A routed wireless access point can be created using the inbuilt wireless* features of the Raspberry Pi 4, Raspberry Pi 3 or Raspberry Pi Zero W, or by using a suitable USB wireless dongle that supports access point mode (recommended). The inbuilt wireless functionality of the Raspberry Pi does not allow for the range in order to have a solid connection. It is possible that some USB dongles may need slight changes to their settings. Documentation for configuring Raspberry Pi devices is located at URL: <https://www.raspberrypi.org/documentation/configuration/wireless/access-point-routed.md>

The remainder of this document will focus on the development, application, and deployment of utilizing the NID before, during and after an operation or event that requires additional collection of evidentiary value. The NID is very portable, light weight, cost effective and open source. Open source allows for modifications to the applications, and collaboration between developers.

The Physical requirements for the NID are the following suggested in this order:

1. An Ethernet cable not to exceed one hundred meters (328 feet) in length.
2. A Universal Serial Bus-A (USB-A) Wi-Fi dongle to extend wireless coverage for the NID. The Wi-Fi dongle can be connected utilizing a USB extension cable not to exceed five meters (16 feet).
3. A micro USB input of five volts and an electrical current input of two amps is required to power the NID.

The hardware can be purchased through various Raspberry Pi sources. For example, <https://www.canakit.com/>. Also, <https://www.raspberrypi.org/> is an excellent resource for Raspberry Pi hardware.

NID development involves the ability to semi-automate a collection process. The following steps are essential for network discovery to collect artifacts of evidentiary value:

1. Connecting a remote computer to the NID Wi-Fi network.
2. Utilize a secure shell (SSH) client to access the NID. PuTTY is a viable SSH client to utilize.
3. Understanding of basic UNIX\Linux commands.
4. Navigating the Raspberry Pi Graphical User Interface (GUI) utilizing a Virtual Network Computing (VNC) Viewer.

Connect to the NID Service Set Identifier (SSID), in this case the NID was created with fdle_nid as the SSID .

Login into the NID with an SSH client. SecureCRT was used on the remote client computer for access (Figure 2). The NID access point is configured on the 192.168.21.0/24 network to possibly avoid conflicting with the suspect network.

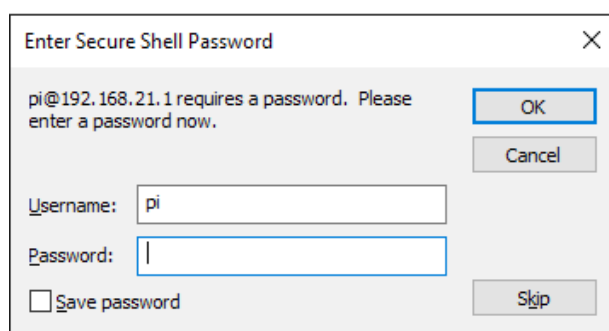
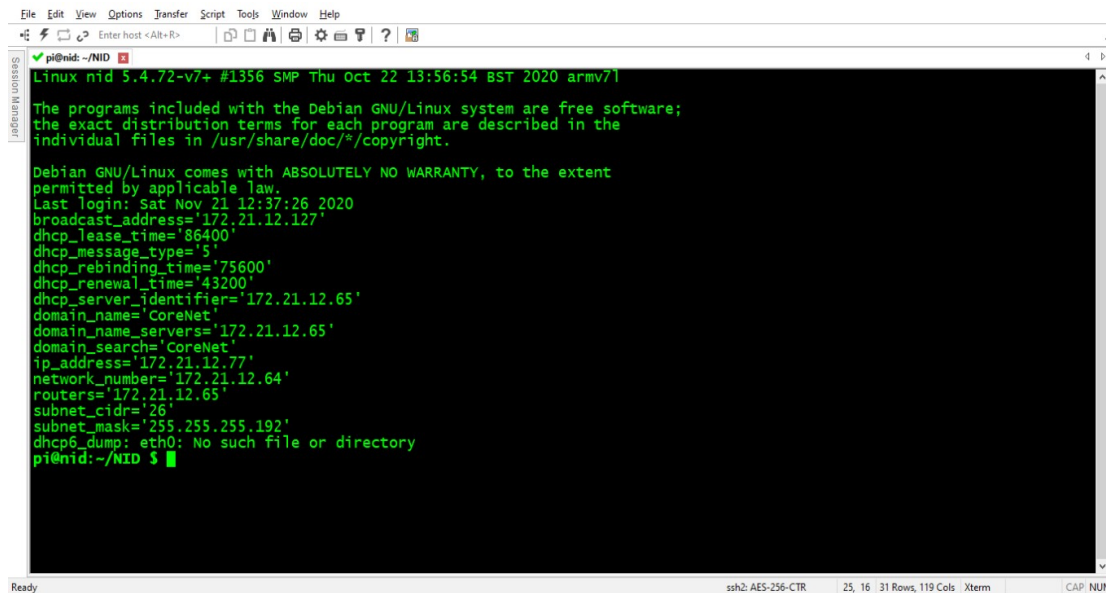


Figure 2

A login script was created, so that certain information can be utilized from the command that initiated. In particular, ***network_number='172.21.12.64'*** and ***subnet_cidr='26'***. This information is necessary to determine the Dynamic Host Configuration Protocol (DHCP) network address space that was provisioned from the a network. In this example ***ip_address='172.21.12.77'*** is applied to the NID interface eth0; which, is attached to the a network through the Ethernet interface (Figure 3).

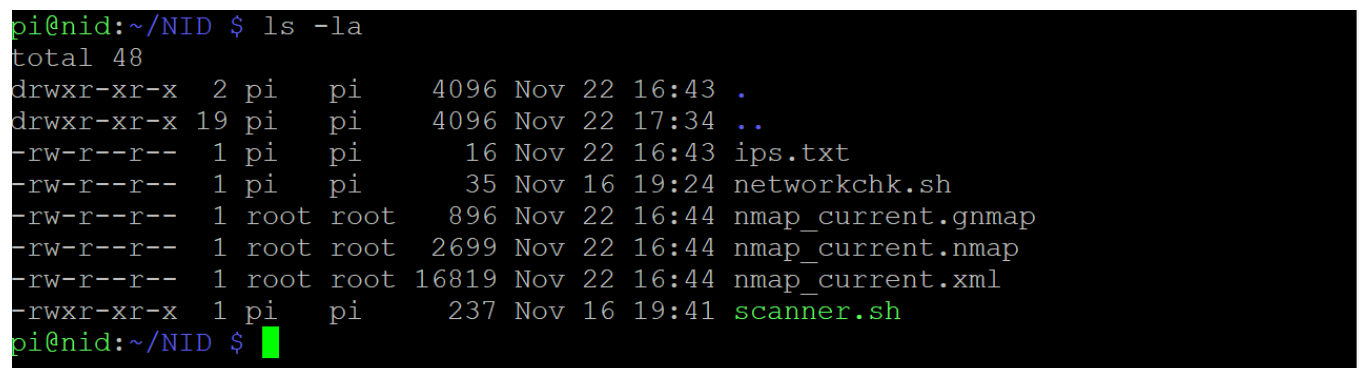


```
pi@nid:~/NID $  
Linux nid 5.4.72-v7+ #1356 SMP Thu Oct 22 13:56:54 BST 2020 armv7l  
The programs included with the Debian GNU/Linux system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sat Nov 21 12:37:26 2020  
broadcast_address='172.21.12.127'  
dhcp_lease_time='86400'  
dhcp_message_type='5'  
dhcp_rebinding_time='75600'  
dhcp_renewal_time='43200'  
dhcp_server_identifier='172.21.12.65'  
domain_name='CoreNet'  
domain_name_servers='172.21.12.65'  
domain_search='CoreNet'  
ip_address='172.21.12.77'  
network_number='172.21.12.64'  
routers='172.21.12.65'  
subnet_cidr='26'  
subnet_mask='255.255.255.192'  
dhcp6_dump: eth0: No such file or directory  
pi@nid:~/NID $
```

Figure 3

The login to the NID drops immediately into the NID directory. The path is the following: /home/pi/NID

The *scanner.sh* command runs and three files have now been created in the /home/pi/NID directory *nmap_current.gnmap*, *nmap_current.nmap* and *nmap_current.xml* (Figure 7):



```
pi@nid:~/NID $ ls -la  
total 48  
drwxr-xr-x  2 pi  pi   4096 Nov 22 16:43 .  
drwxr-xr-x 19 pi  pi   4096 Nov 22 17:34 ..  
-rw-r--r--  1 pi  pi    16 Nov 22 16:43 ips.txt  
-rw-r--r--  1 pi  pi    35 Nov 16 19:24 networkchk.sh  
-rw-r--r--  1 root root  896 Nov 22 16:44 nmap_current.gnmap  
-rw-r--r--  1 root root 2699 Nov 22 16:44 nmap_current.nmap  
-rw-r--r--  1 root root 16819 Nov 22 16:44 nmap_current.xml  
-rwxr-xr-x  1 pi  pi    237 Nov 16 19:41 scanner.sh  
pi@nid:~/NID $
```

Figure 7

The *nmap_current.xml* file can be opened in the GUI version of nmap known as ZenMap.

From the remote computer connected to the NID, a VNC session can be launched (Figure 8).

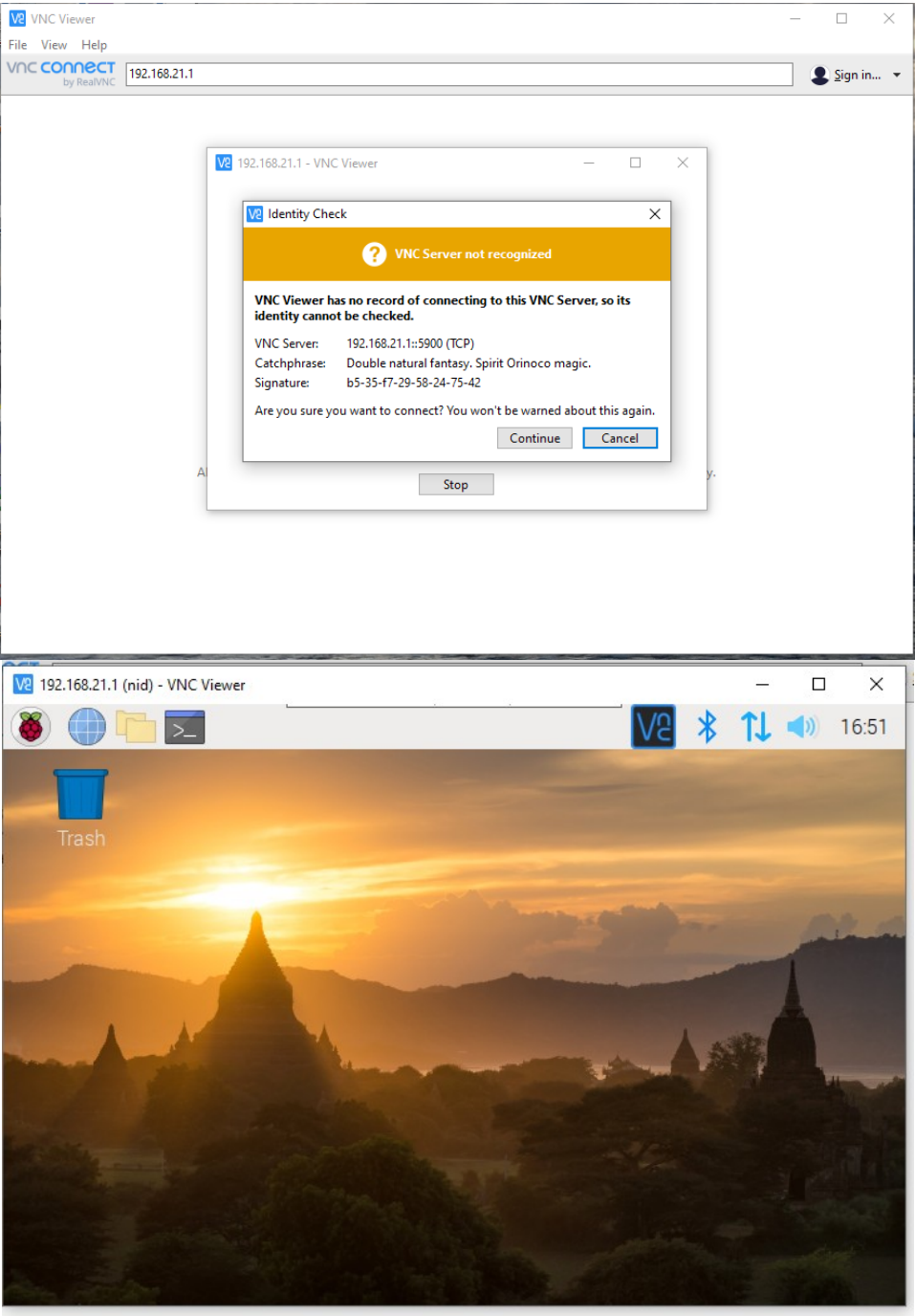


Figure 8

The VNC session initiates the NID desktop, so the Nmap results can be viewed in the GUI version of nmap, ZenMap.

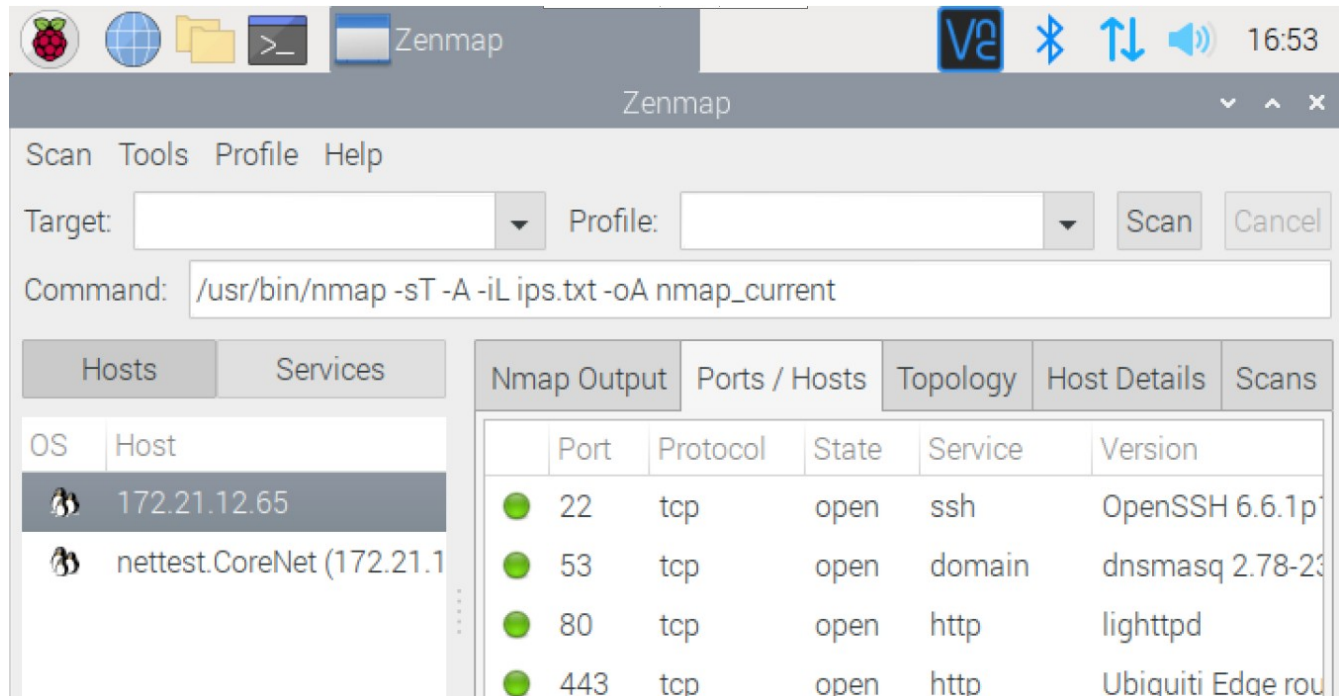


Figure 9

Figure 9 is the results from the command line Nmap scan that was initiated from the *scanner.sh* script. The actual Nmap command that executed can be viewed on this screen. The results display what ports and protocols the router is listening on, and additional detailed information is available for this host. Based on this report further analysis can be done on the router through a web browser to view additional configuration details on it.

NID version 2.0 has introduced TSHARK integration. TSHARK can be utilized during a network packet capture event. The requirement to capture network packets requires that the customer premise equipment (CPE), specifically a switch that has the ability to enable port mirroring. Port mirroring is used on a network switch to send a copy of packets seen on one switch port (or an entire VLAN) to a network monitoring connection on another port.

The TSHARK script would need to be modified to your environment for writing the .pcap file. The script is found in the following file path after login:

/NID/PacketCapture/1_PCapture.sh

The example script:

```
#!/usr/bin/env bash
sudo tshark -q --interface eth0 -F pcap -w /media/pi/SANS/TEST/capture.pcap &
```

A one and two terabyte external hard drive was tested with the script. By default the Raspberry PI platform will mount an external drive as ***/media/pi***

To stop the network packet capture process run the following script it will display the Process ID (PID) number:

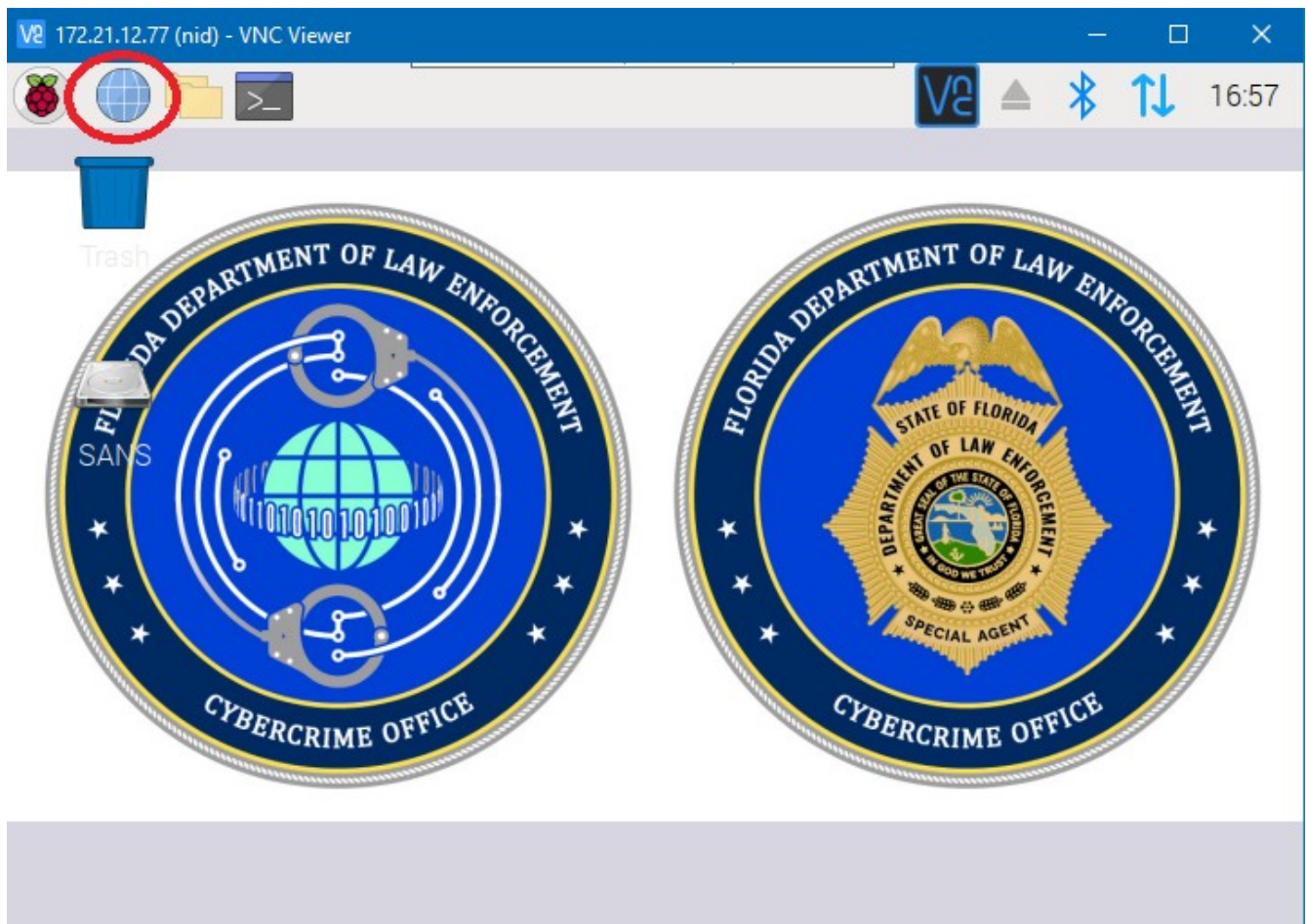
/NID/PacketCapture/2_processID.sh

PID numbers are dynamically assigned by the operating system (OS). The following is an example:

```
pi@nid:~/NID/PacketCapture $ ./2_processID.sh
root      10608  0.0  0.3  9948  3184 pts/0    S    15:03   0:00 sudo tshark -q --interface eth0 -F pcap -w /media/pi/
SANS/TEST/capture.pcap
root      10609  0.5  8.6 156296 81396 pts/0    S    15:03   0:01 tshark -q --interface eth0 -F pcap -w /media/pi/SANS/
TEST/capture.pcap
pi         10707  0.0  0.0   7348   540 pts/0    S+   15:07   0:00 grep tshark
pi@nid:~/NID/PacketCapture $ sudo kill 10608
pi@nid:~/NID/PacketCapture $ 1186 packets captured
```

Ideally, the Raspberry Pi 4 is the platform to use for long term network packet capture based on the architecture of the device. Currently development is in process for the Raspberry Pi 4, and the current deployment of this distribution.

NID documentation can be viewed after you VNC into the session by opening up the Web Browser on the Raspberry Pi, or by selecting the following icon circled in red:



NID Version 2.5

NID version 2.5 has included the following scripts to facilitate streamlining the process of network discovery.

These scripts are in the following path:

home/pi/NID

0_net_cidr.sh (New)

1_scanner.sh

2_verify.sh

3_collected.sh

4_clean.sh (Updated)

0_net_cidr.sh = Automates the network and Classless Inter-Domain Routing (CIDR) it is a compact representation of its network mask or subnet mask, and writes the results to the ips.txt file.

1_scanner.sh = This scripts launches the nmap command.

2_verify.sh = This script writes an MD5 hash value to *verify.txt* of the collected scanner results.

3_collected.sh = This script moves the collected data to an external device and zips the collected data. This script will need to be modified for your environment.

4_clean.sh = Only run this script after you are satisfied with your collected results, and you have saved them off to a location that can be reviewed later. Otherwise the collection process is deleted. Typically this should be run when you are at a new location before the *1_scanner.sh* process.

The NID road-map for development continues with the ability to create scripts for network interrogation utilizing Python 2.x and 3.x, Bash, Perl, PowerShell Core and other tools.

* The command that permanently turns off the internal Wi-Fi network card can be found at this URL: <https://raspberrypi.com/disable-wifi-raspberry-pi/>