Capgemini

SAT1: 004: Introduction to Wireshark

Overview

This lab will teach basic Wireshark skills to include GUI identification and application navigation.

Time: 15 Minutes

Learning Objectives

Upon completion of this lab, you should be able to:

1. Familiarize yourself with the Wireshark interface.
2. Customize the Wireshark interface.

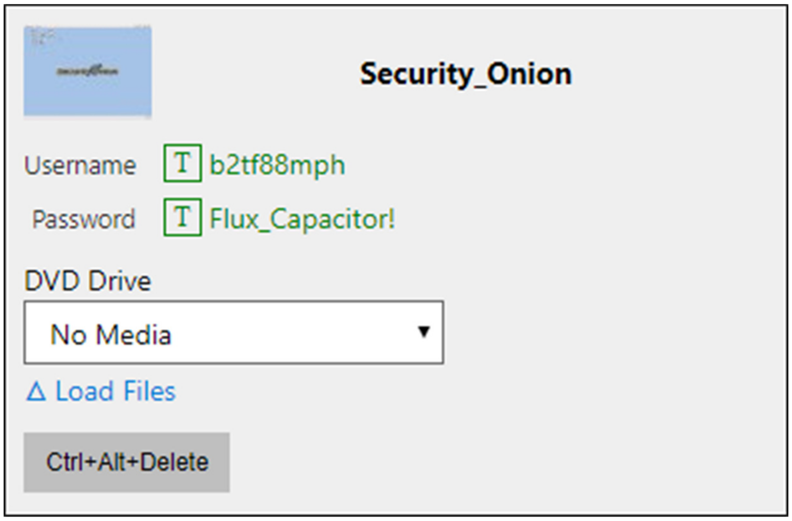
Resources

The following document has been provided to assist you in this lab.

Wireshark Cheat Sheet

Log in to the Lab Machine

Select the **Security_Onion** machine on the Machines Tab.



Security_Onion

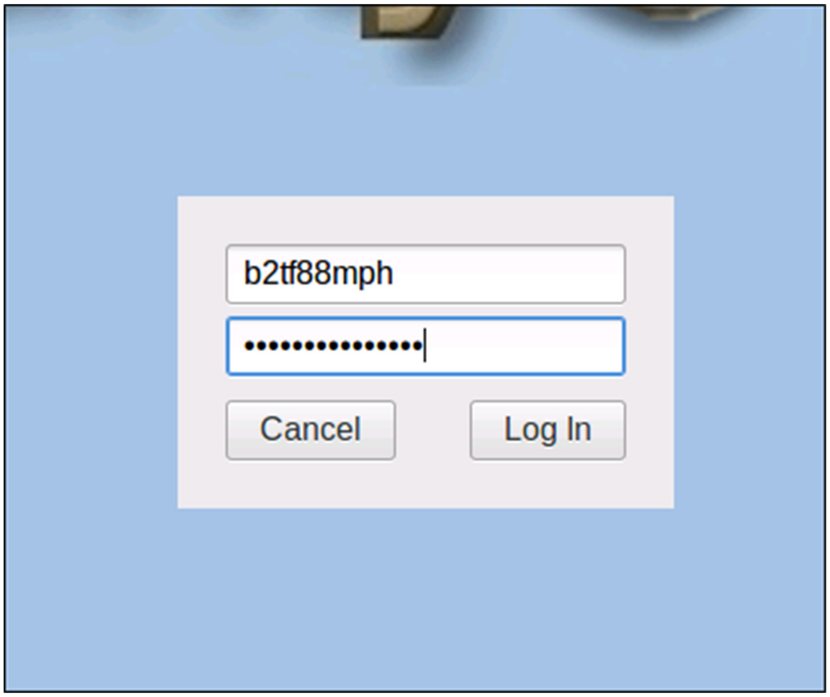
Username b2tf88mph

Password Flux_Capacitor!

DVD Drive

No Media

Enter the Username and Password on the **Security_Onion** machine, and click Log In.



b2tf88mph

•••••••••••••••

1.0 The User Interface

What is Wireshark?

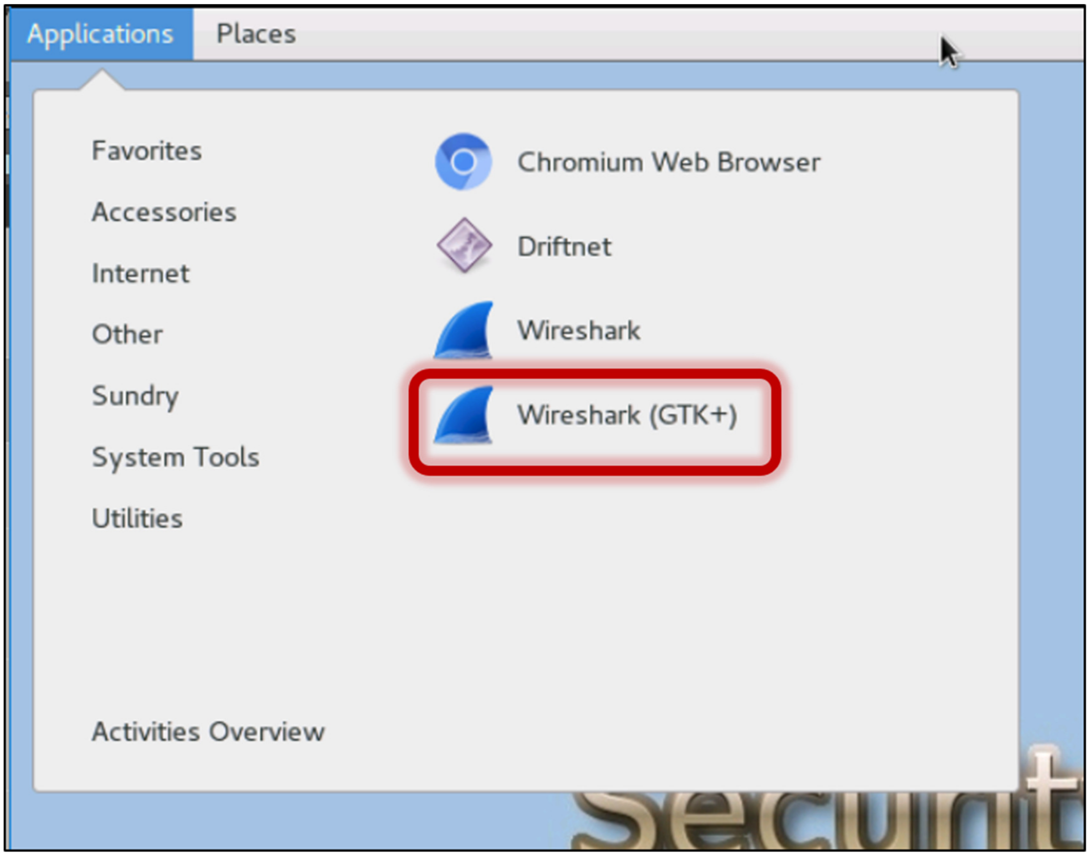
Wireshark is a free and open source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education.

Let’s get familiar with the interface, so we can learn to use it.

In this class we use Wireshark (GTK+) since it is the default program for PCAP files in **Security_Onion**. There are some graphical differences between the two versions, but no functional differences.

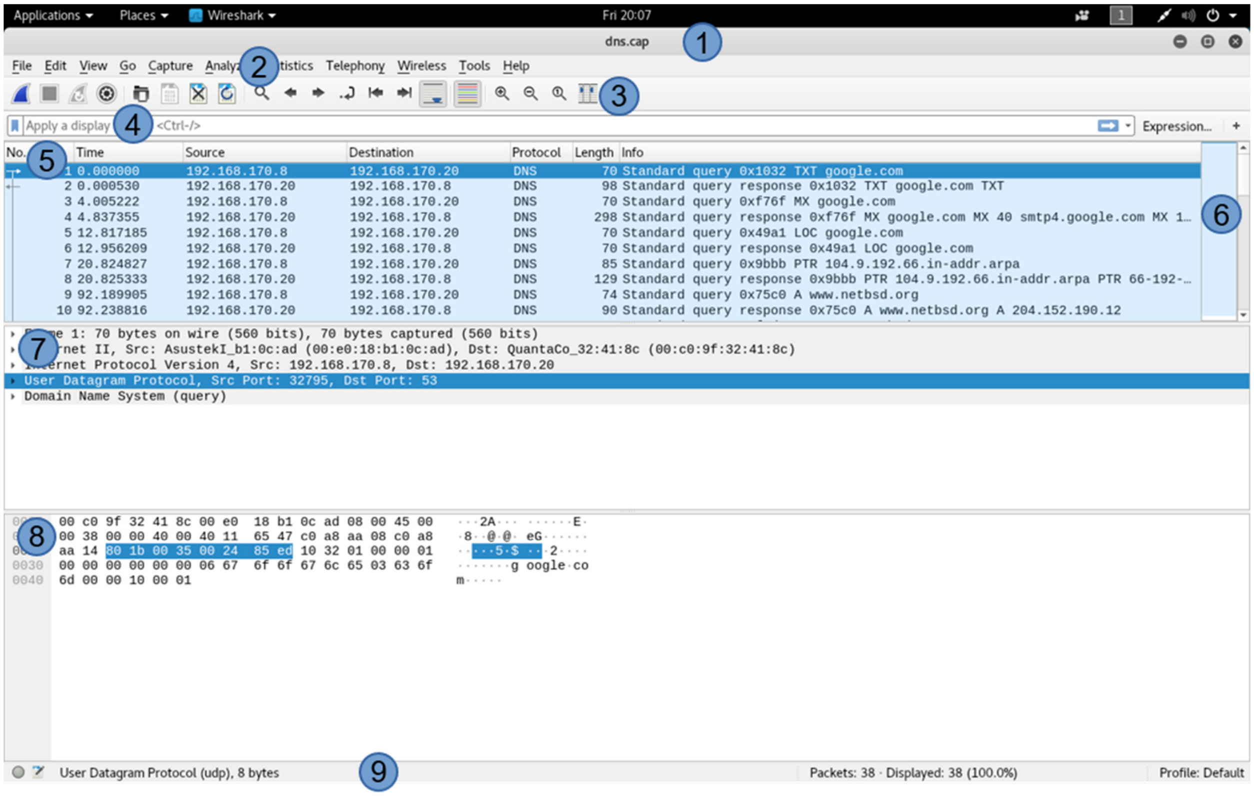
GTK+ is a a cross-platform widget toolkit for creating graphical user interfaces in LINUX builds.

First you will need to open the Wireshark application. In **Security_Onion** Wireshark can be found by clicking **Applications** and moving the mouse over **Internet** and select **Wireshark (GTK+)**.



Now that we have the application open, we want to explore the User Interface. Take a moment and explore the interface and find the following components.

```
   1. Title Bar
   2. Main Menu
   3. Main Toolbar
   4. Filter Toolbar
   5. Packet List
   6. Intelligent Scrollbar
   7. Packet Details
   8. Packet Bytes
9. Status Bar
```



```
No.   Time       Source          Destination     Protocol Length Info
    1 0.000000   192.168.170.8   192.168.170.20  DNS          70 Standard query 0x1032 TXT google.com
    2 0.000530   192.168.170.20  192.168.170.8   DNS          98 Standard query response 0x1032 TXT google.com TXT
    3 4.005222   192.168.170.8   192.168.170.20  DNS          70 Standard query 0xf76f MX google.com
    4 4.837355   192.168.170.20  192.168.170.8   DNS         298 Standard query response 0xf76f MX google.com MX 40 smtp4.google.com MX 1…
    5 12.817185  192.168.170.8   192.168.170.20  DNS          70 Standard query 0x49a1 LOC google.com
    6 12.956209  192.168.170.20  192.168.170.8   DNS          70 Standard query response 0x49a1 LOC google.com
    7 20.824827  192.168.170.8   192.168.170.20  DNS          85 Standard query 0x9bbb PTR 104.9.192.66.in-addr.arpa
    8 20.825333  192.168.170.20  192.168.170.8   DNS         129 Standard query response 0x9bbb PTR 104.9.192.66.in-addr.arpa PTR 66-192-…
    9 92.189905  192.168.170.8   192.168.170.20  DNS          74 Standard query 0x75c0 A www.netbsd.org
   10 92.238816  192.168.170.20  192.168.170.8   DNS          90 Standard query response 0x75c0 A www.netbsd.org A 204.152.190.12
```

```
▸ Frame 1: 70 bytes on wire (560 bits), 70 bytes captured (560 bits)
▸ rnet II, Src: AsustekI_b1:0c:ad (00:e0:18:b1:0c:ad), Dst: QuantaCo_32:41:8c (00:c0:9f:32:41:8c)
▸ Internet Protocol Version 4, Src: 192.168.170.8, Dst: 192.168.170.20
▸ User Datagram Protocol, Src Port: 32795, Dst Port: 53
▸ Domain Name System (query)
```

```
0000  00 c0 9f 32 41 8c 00 e0  18 b1 0c ad 08 00 45 00   ···2A··· ······E·
0010  00 38 00 00 40 00 40 11  65 47 c0 a8 aa 08 c0 a8   ·8··@·@· eG······
0020  aa 14 80 1b 00 35 00 24  85 ed 10 32 01 00 00 01   ·····5·$ ···2····
0030  00 00 00 00 00 00 06 67  6f 6f 67 6c 65 03 63 6f   ·······g oogle·co
0040  6d 00 00 10 00 01                                  m·····
```

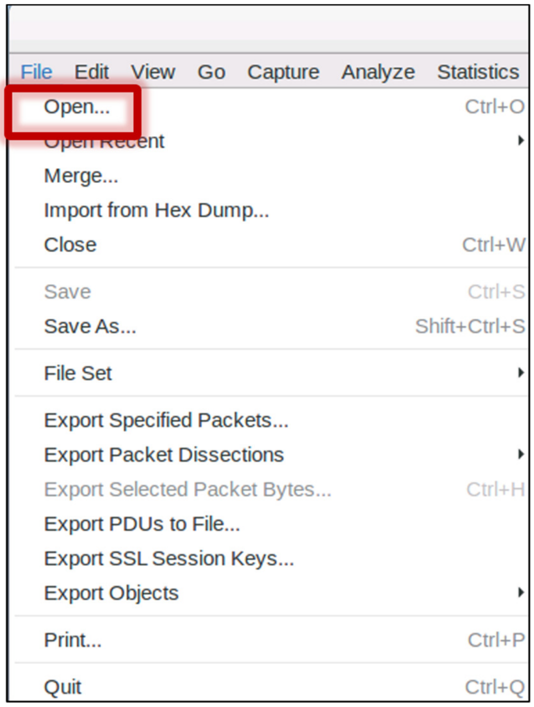
User Datagram Protocol (udp), 8 bytes

Packets: 38 · Displayed: 38 (100.0%)

Profile: Default

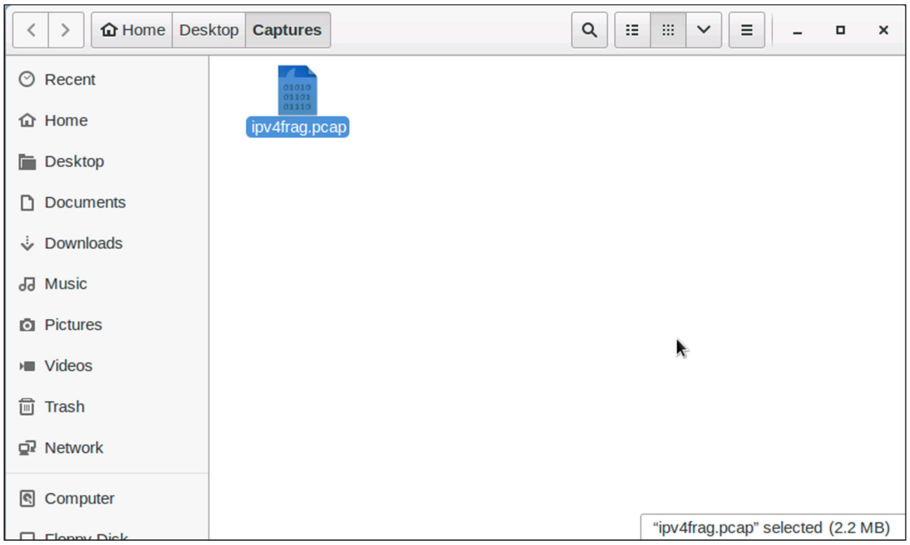
1.0 Opening a File in Wireshark

1.1 On the Main Menu, click on **File** and click on **Open**.



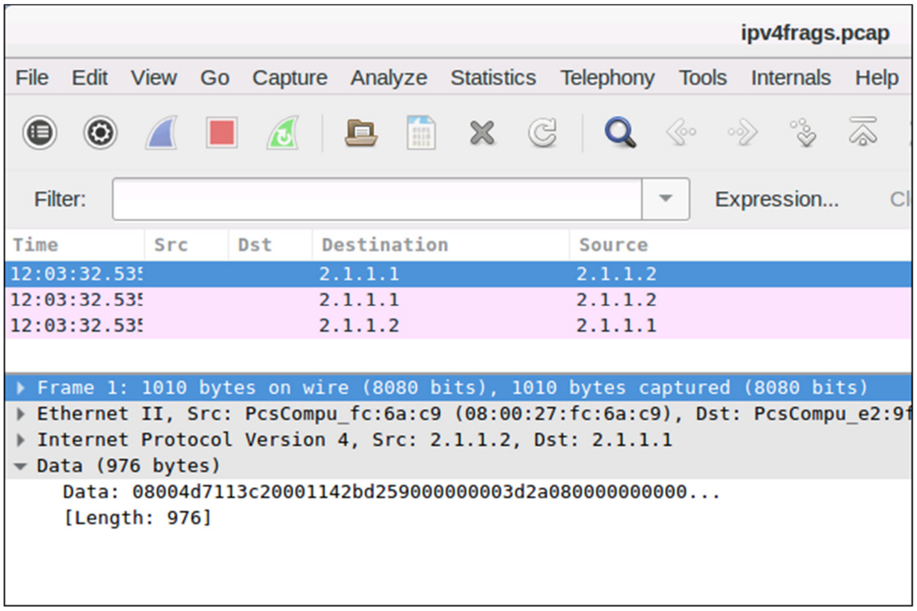
1.2 Click on **b2tf88mph**, then navigate to the **Desktop** folder followed by the **Captures** folder.

1.3 Open `ipv4frag.pcap`.



“ipv4frag.pcap” selected (2.2 MB)

1.4 In the Packet Details, click on the data line.

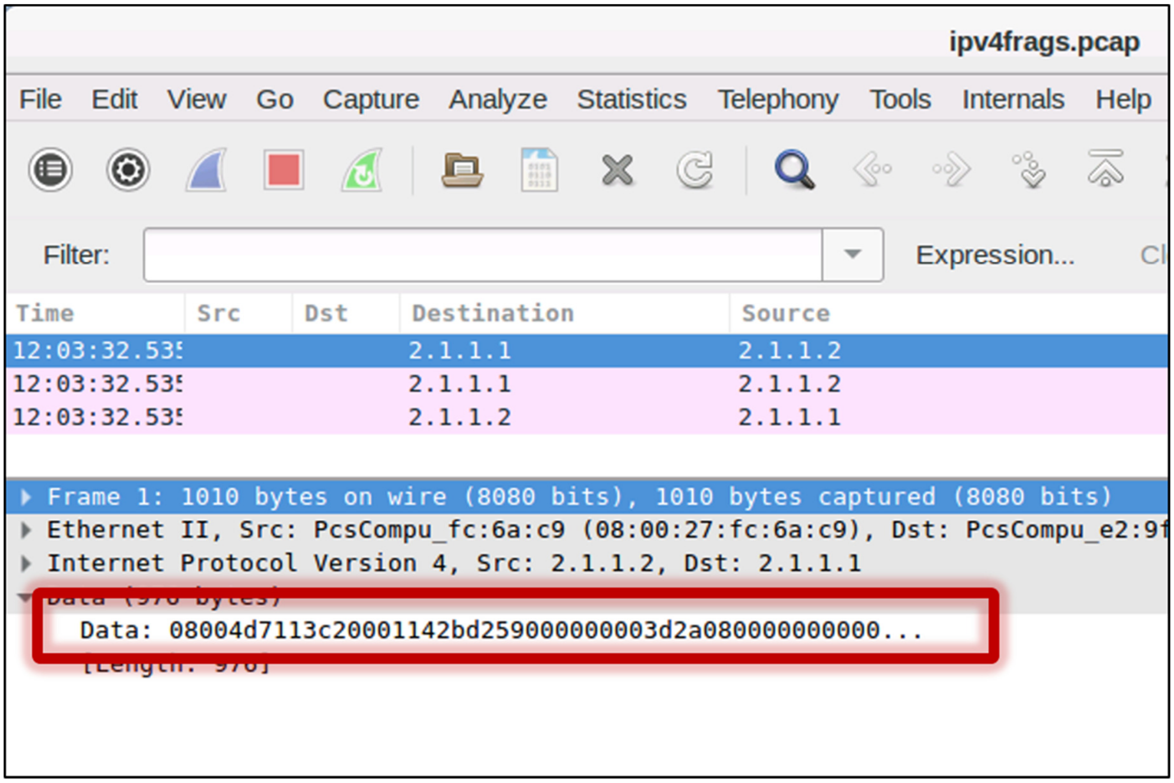


```
Time          Src   Dst   Destination          Source
12:03:32.53!              2.1.1.1              2.1.1.2
12:03:32.53!              2.1.1.1              2.1.1.2
12:03:32.53!              2.1.1.2              2.1.1.1
```

```
▸ Frame 1: 1010 bytes on wire (8080 bits), 1010 bytes captured (8080 bits)
▸ Ethernet II, Src: PcsCompu_fc:6a:c9 (08:00:27:fc:6a:c9), Dst: PcsCompu_e2:9f
▸ Internet Protocol Version 4, Src: 2.1.1.2, Dst: 2.1.1.1
▾ Data (976 bytes)
    Data: 08004d7113c20001142bd259000000003d2a080000000000...
    [Length: 976]
```

This will load the data from the packet selected in the Packet List.

As you can see, there is plaintext data of the alphabet. This may be helpful in particular packets to identify passwords and other plaintext data.


ipv4frags.pcap
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Cl

```
Time          Src    Dst    Destination        Source
12:03:32.53!                2.1.1.1            2.1.1.2
12:03:32.53!                2.1.1.1            2.1.1.2
12:03:32.53!                2.1.1.2            2.1.1.1
```

```
▸ Frame 1: 1010 bytes on wire (8080 bits), 1010 bytes captured (8080 bits)
▸ Ethernet II, Src: PcsCompu_fc:6a:c9 (08:00:27:fc:6a:c9), Dst: PcsCompu_e2:9f
▸ Internet Protocol Version 4, Src: 2.1.1.2, Dst: 2.1.1.1
▾ Data (976 bytes)
    Data: 08004d7113c20001142bd259000000003d2a080000000000...
    [Length: 976]
```

Great job, you have completed LAB004!

Thank You, you may now close this module.