

Data Visualization System

Introduction

The purpose of the Data Visualization System is to a tool that provides the user with an integrated view of the data generated by [the Evaluator Centric and Extensible Logger daemon \(ECELd\)](#) and the ability to tag and modify the data associated with a capture taken by ECELd.

Problem statement:

Data is a critical asset in cybersecurity; it helps researchers and practitioners develop novel technologies based on real past incidents. However, this asset is lacking, and even more, tools that specifically focus on the analysis of data are far and few between. The Evaluator Centric and Extensible Logger daemon (ECELd) is a tool that attempts to reduce this gap. The tool allows users to collect data for the purpose of analysis. Collected data include network data, keystrokes, system calls, and screenshots, among others. Additionally, the ECELd includes a wireshark component which allows users to annotate the network data using the popular packet sniffer application. Both the ECELd tool and the ECELd-wireshark component are integrated into the ECELd-netsys.

Team Members:

- Briana Sanchez
- Stephanie Medina
- Luisana Clarke
- Rocio Cardona
- Bianca Alvarez
- Dima Abdel Jaber

Client:

- Dr. Jaime Acosta

Guidance Team:

- Dr. Salamah Salamah

Description

The system consists of two primary components: the packet view shows ECELd-Wireshark and the timeline view shows the parsed ECELd-collected data using a horizontal layout. These two windows combined will allow for the user to have a better visualization of what is being analyzed. Many features are integrated in this system in order to bring better visualization and each feature will be specific to the type of data that is going to be displayed. In this system, each specific set of data will be considered a dataline. For example, a mouseclicks dataline will contain

different information than what a keypresses dataline contains, which will result in specific features for each. Moreover, with different datasets, there must be a way to be able to distinguish between each of them, so each dataline will have an assignable color in order to make it easier to tell them apart. This color assignment shall be displayed in both the timeline view and the packet view. Finally, the most important feature of this system will be synchronization. The synchronization feature shall be enabled and disabled depending on the user's preference. The main purpose for this feature is to allow the user to use both timeline view and packet view at the same time and be able to see the data that is correlated to where the user is currently analyzing.

Installing the Data Visualization System (DVS)

The Data Visualization System's source code can be found in the following link: <https://github.com/smedina7/DVS>. The system works in both Linux and Windows. Follow the instructions below for installation.

Windows Installation

The first step of installation is to clone the repository provided in the link above and navigate to the project folder.

Virtual Environment

Install eceld-wireshark using the installer provided. Type the following in the command prompt.

```
> nsis\Wireshark-win64-3.2.5.exe
```

You will get a prompt to install Wireshark. Install using default settings provided.

- Install Python's Virtual Environment Builder:
- Create and activate virtual environment:
- Install required dependencies:

```
> pip install virtualenv  
> virtualenv dvs-venv  
> dvs-venv\Scripts\activate
```

```
(dvs-venv)> pip install -r requirements.txt
```

Lastly, run main.py to start the DVS GUI.

```
(dvs-venv)> python main.py
```

Linux Installation

Requirements

In order for DVS to run properly, Eceld-Wireshark must be installed.

The following are ways to install it:

- Install Eceld-Netsys: Refer to the Git-Hub page

<https://github.com/ARL-UTEP-OC/eceld-netsys.git>

- Install Eceld-Wireshark: Refer to the Git-Hub page

<https://github.com/ARL-UTEP-OC/eceld-wireshark>

- Install with the DVS Installer

DVS installation steps

The first step of installation is to clone the repository provided in the link above and navigate to the project folder.

```
kali@kali:~/DVS$ sudo ./installDeb.sh
```

You will be prompted if you would like to install Eceld-Wireshark:

```
kali@kali:~/DVS$ sudo ./installDeb.sh Running apt-get updateHit:1
https://packages.microsoft.com/repos/vscode stable InReleaseHit:2
http://kali.download/kali kali-rolling InReleaseReading package
lists... DoneDVS depends on : eceld-wireshark would you like to
install it [Y/n]
```

Input "Y" to install Eceld-Wireshark, if already installed input "n" to skip.

Activate Environment:

```
kali@kali:~/DVS$ source venv/bin/activate
```

Run DVS:

```
(venv) kali@kali:~/DVS$ sudo python3 main.py --no-sandbox
```

Using the Data Visualization System (DVS)

The Data Visualization System has two main components: Packet View and Timeline View. This section will provide a tutorial on basic usage of each component. Furthermore, users will be able to use the DVS as a standalone program or integrated as part of ECEld.

Standalone Version

In the standalone version, users will be able to choose from what folder they'll be analyzing their data. In the event that the user already has the data that they need, this gives the flexibility to just do the data visualization and analyzation without having to open ECEld-NetSys.

Iteration 1 Components:

For the first iteration of the DVS system, the team focused on getting the Timeline View set up. This means that the packet view will not appear yet for this iteration. The reason for that is because the packet view is already implemented, and we will just need to combine the two windows once the timeline view is ready.

Iteration 2 Components:

For the second iteration of the DVS system, the team focused on getting the Packet View set up. The team updated front end and backend for this iteration.

Iteration 3 Components:

For the third iteration of the DVS system, the team focused on Synchronization features, color assignment, Suricata alerts, and more frontend features. The team also tested the system using large data and updated the installer.

Iteration 4 Components:

For the fourth iteration of the DVS system, the team focused on Synchronization features, color assignment, Suricata alerts, and more frontend features. The team also tested the system using large data and updated the installer. This iteration was expanding on features implemented in iteration 3.

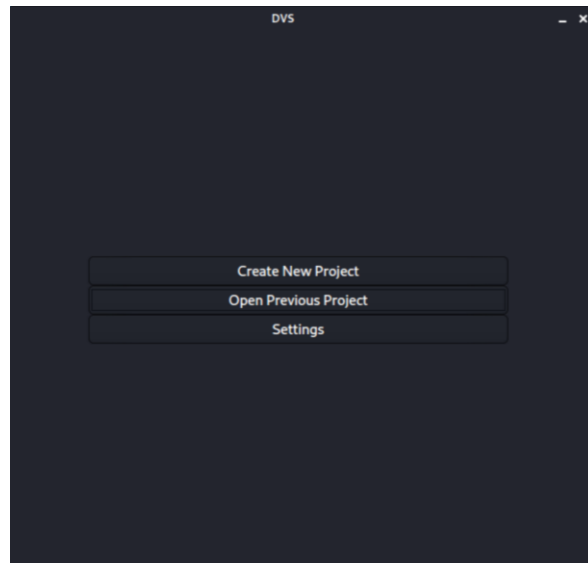
Iteration 5 Components:

For the fifth iteration of the DVS system, the team focused on Synchronization features, color assignment for Wireshark, Suricata alerts, packet comments, and more frontend features. The team also tested the system using large data and updated the installer. This iteration was expanding on features implemented in iteration 4.

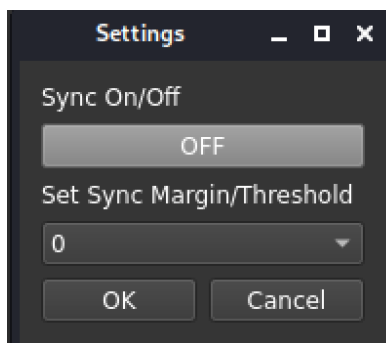
The following sections will review what are the current available features.

Start Window

Once the DVS is started, the following window will appear:

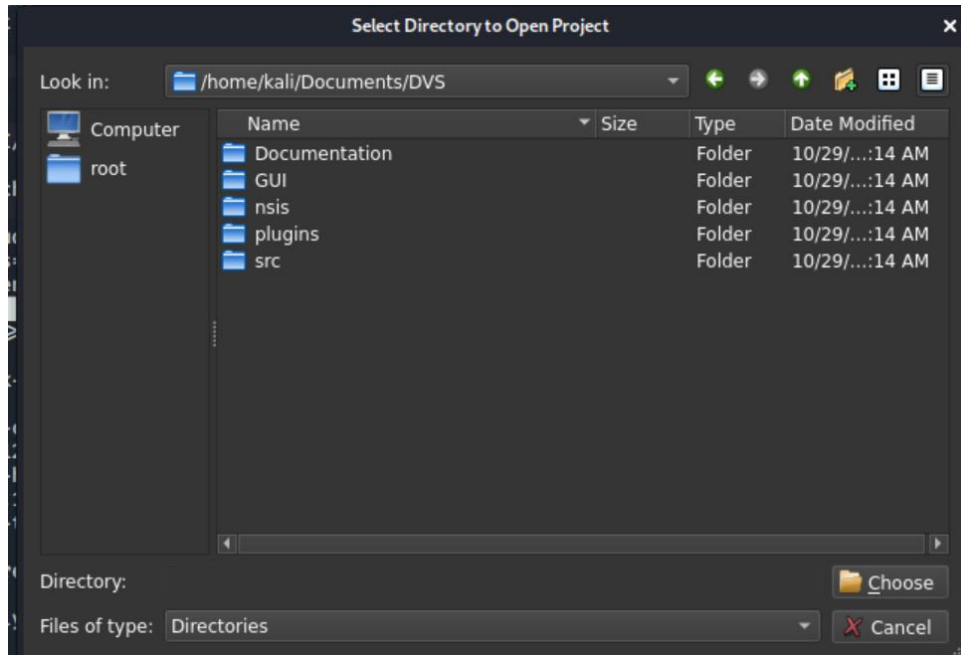


The settings button will display the following:



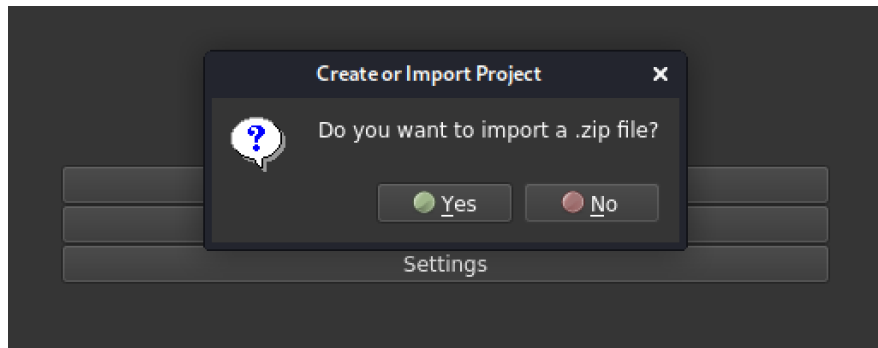
This is for sync configuration. For now, it only saves that values that were selected and it does not affect nor change the way the sync behaves. This shall be implemented until the end.

The "Open Previous Project" button shows a file browser to allow you to select a previous project. The following will appear when you select "Open Previous Project":

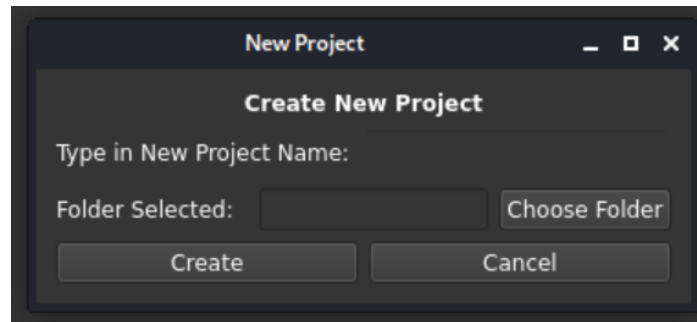


Once you select a project, the Timeline and the Packet view will appear.

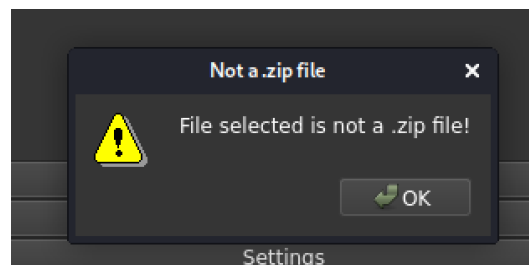
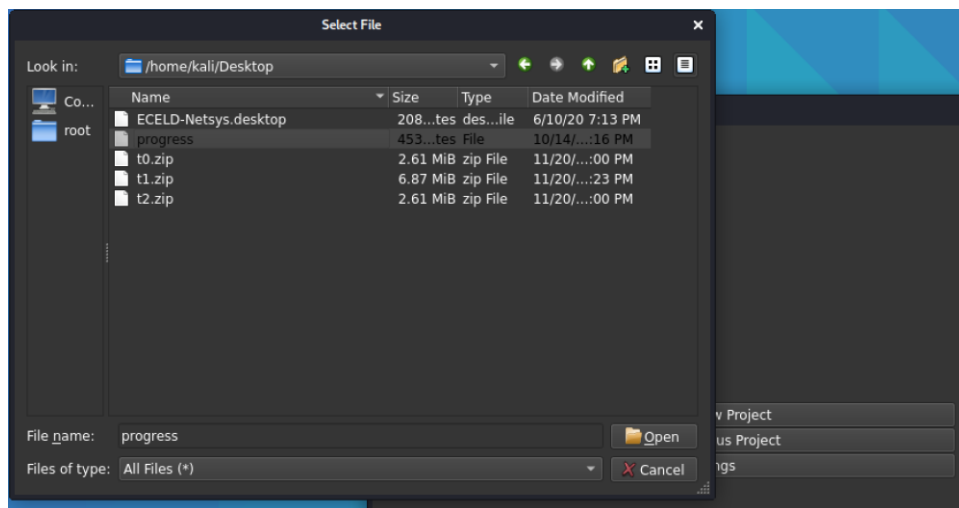
The third option "Create New Project" will ask if you would like to import a .zip file:



If you click on no, it will prompt you to name the project and select the folder where the data is stored. The create option will create a copy of the folder selected, which will be stored in DVS/Project Data.



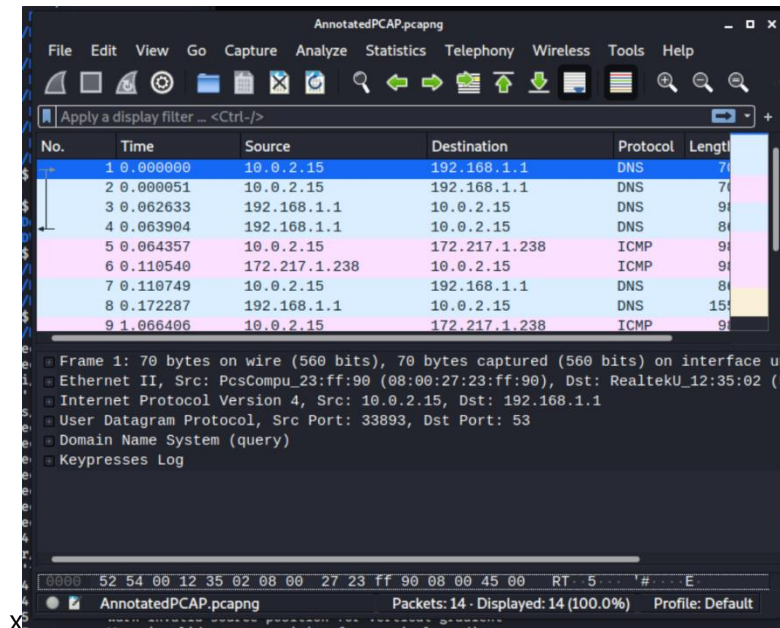
If you click on yes, it will prompt you to select a .zip file. If you do not select a .zip file, it will not let you continue.



Once the user selects a project, two views will be shown, the Timeline View and the Packet View.


Packet View

The packet view consists of the ECEld version of Wireshark that is available at <https://github.com/ARL-UTEP-OC/eceld-wireshark>.

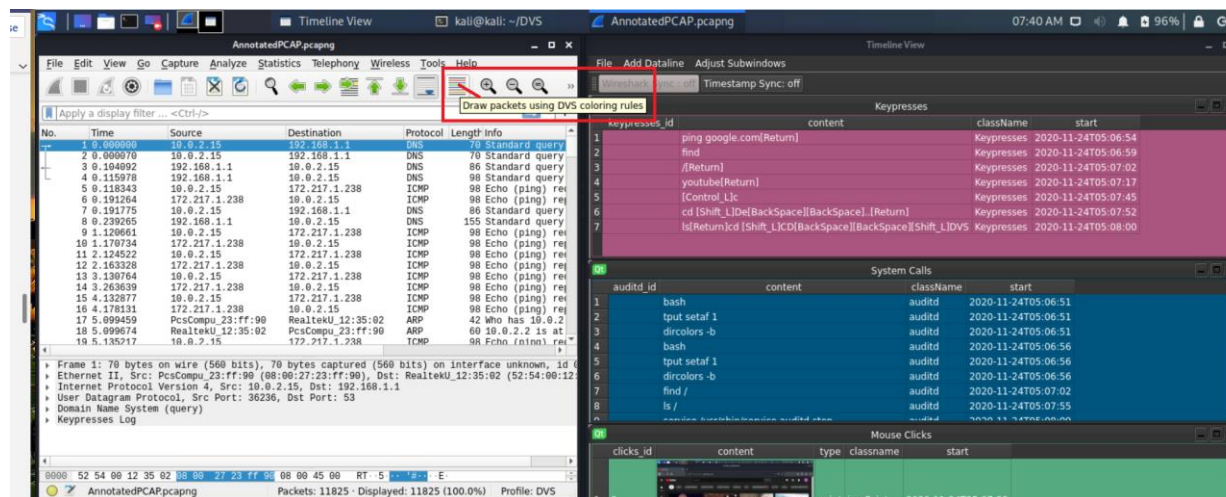


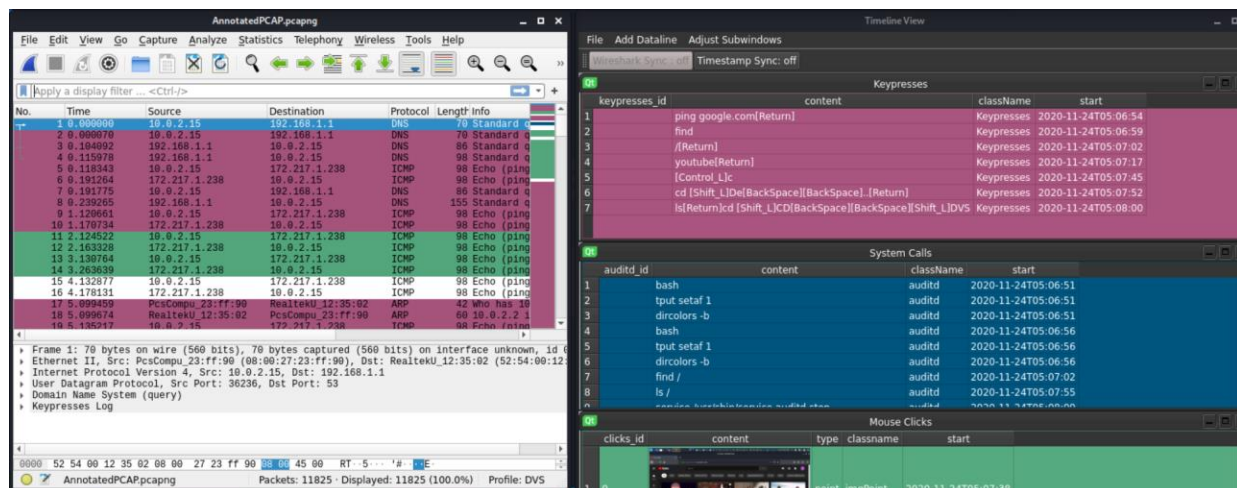
Wireshark Coloring Rules

The system will generate a color rule file that can be read by Wireshark. In order to colorize the packets in the Wireshark View to correspond to active data lines on the Timeline View, the user

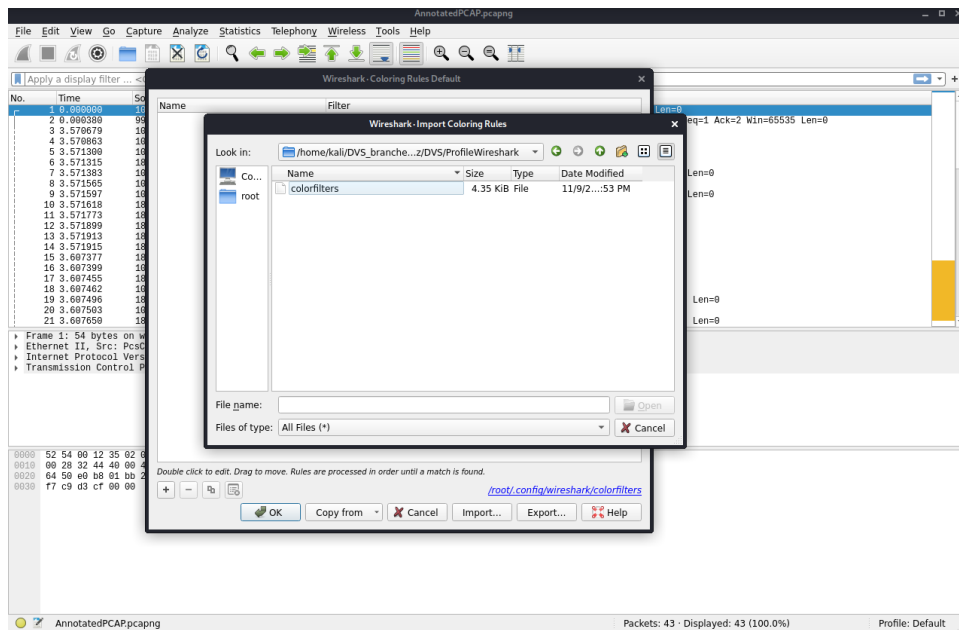
can click on the  toolbar button and refresh the page by pressing "CTRL+R".


Enabling/Disabling this button colors and de-colors packets accordingly.

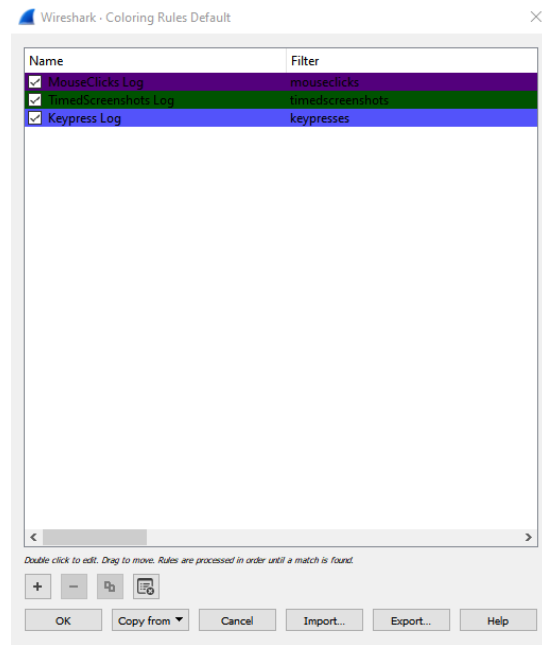




Alternatively, the user can import this file manually into Wireshark. This file is located in the profile directory within Wireshark (./eceld-wireshark/wireshark-3.2.0/profiles/DVS/colorfilters) **OR** in the DVS directory (./DVS/GUI/PacketView/colorfilters)



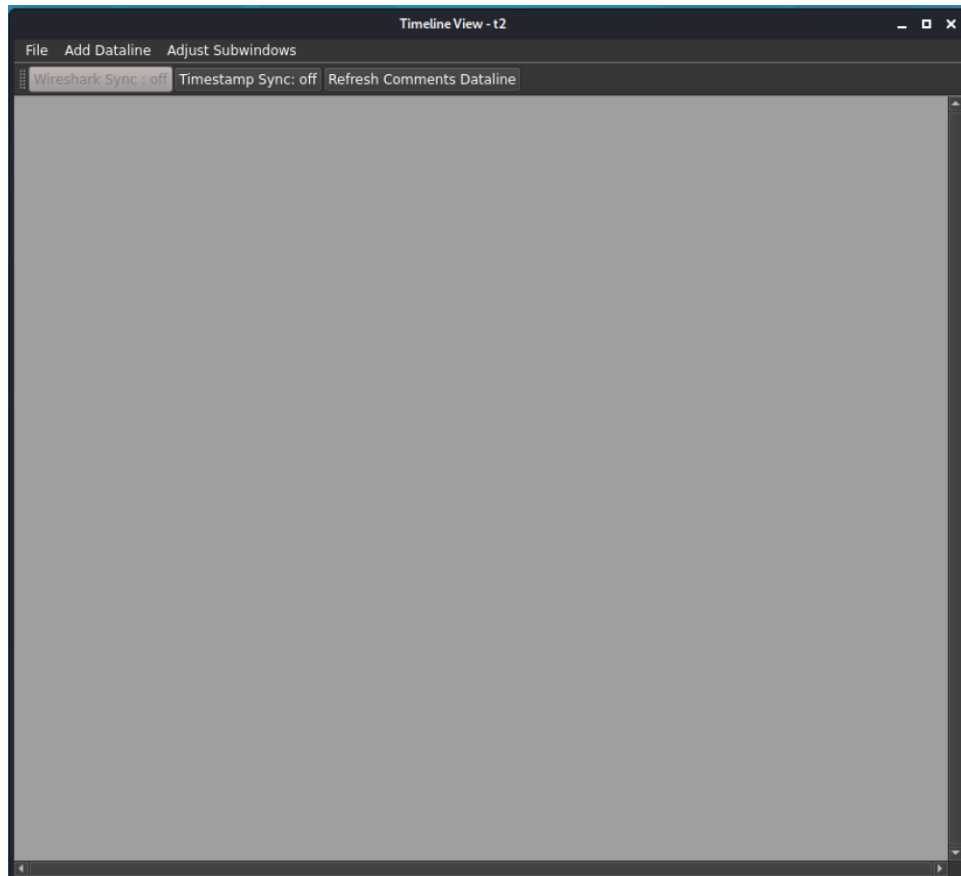
If there are any current rules on the dialog, clear them using this button  Then the desired colorfilter file can be selected and applied:



Currently, the DVS on a Windows OS supports only the manual import of the color profile.

Timeline View

The Timeline View looks like the following:



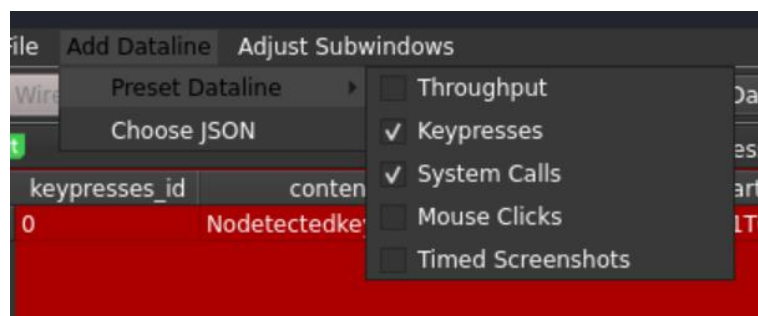
Adding Datalines

The “Add Dataline” menu option has two options: Preset Dataline and Choose JSON .

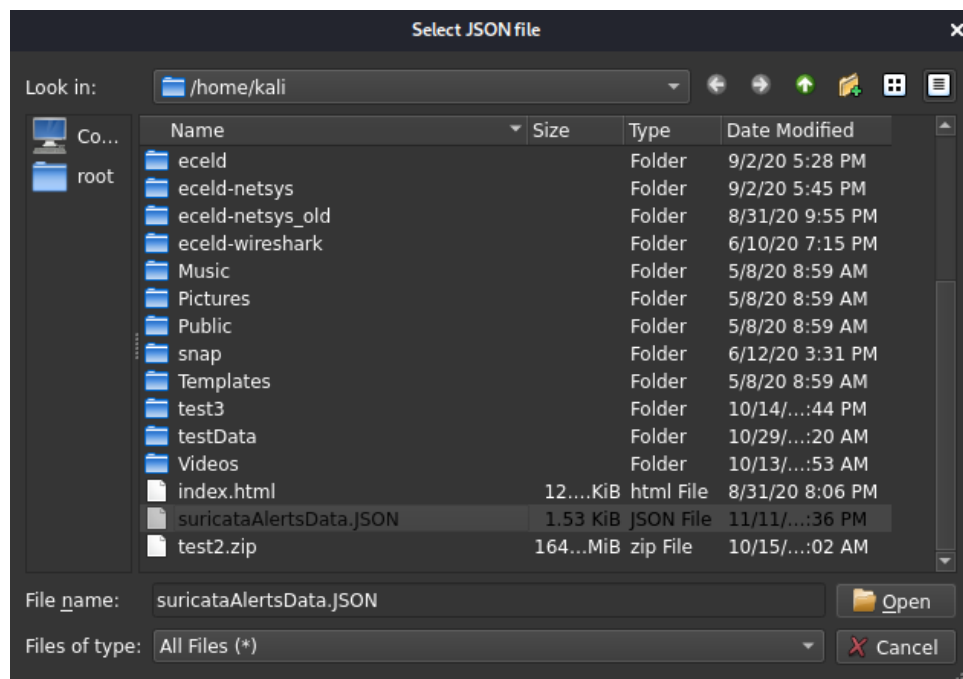
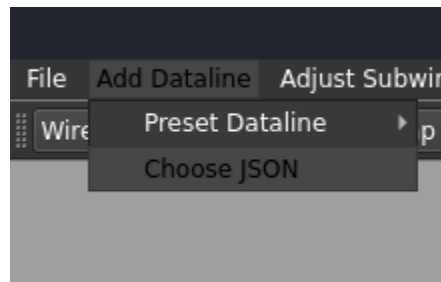
When selecting the Preset Dataline option, a list of the predefined datalines will become visible.

These datalines will use the data from the project created in the start window. When a dataline is open, the checkbox will be checked. When the dataline is closed, the checkbox is unchecked.

This feature allows for one view of each dataline to be available at once.



When selecting the Choose JSON option, the user can select a JSON file to open it on the timeline view. According to the JSON data, the correct render will be used to display the data.



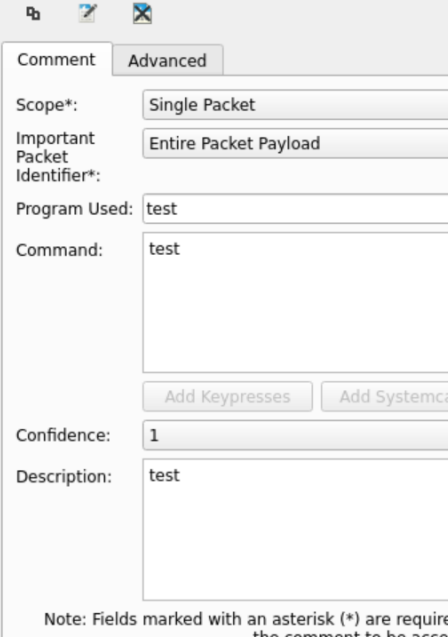
As an example, the user can select a new JSON file like the Suricata Alerts Data that will be rendered as a text table display.

Suricata					
	suricata_id	suricata_rule_id	content	className	start
1	0	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:51
2	1	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:51
3	2	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:51
4	3	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:51
5	4	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:53
6	5	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:54
7	6	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:54
8	7	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:55
9	8	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:55
10	9	[1:2:0]	Test dns_query option	suricataAlert	2020-10-07T02:02:55

The user can also select the packet comment dataline that will display all packet comments added on wireshark.

[illegible]

The user can add packet comments by going to the wireshark view and going to Edit > Packet Comment...



The image shows the 'Wireshark - Packet 1 Comment' dialog box. It has a title bar with standard window controls. Below the title bar is a toolbar with three icons: a document, a pencil, and a trash can. The dialog is divided into two tabs: 'Comment' and 'Advanced'. The 'Comment' tab is active, showing several fields: 'Scope*' with a dropdown menu set to 'Single Packet', 'Important Packet Identifier*' with a dropdown menu set to 'Entire Packet Payload', 'Program Used:' with a text box containing 'test', 'Command:' with a text box containing 'test', 'Confidence:' with a dropdown menu set to '1', and 'Description:' with a text box containing 'test'. Below these fields are two buttons: 'Add Keypresses' and 'Add Systemcalls'. At the bottom of the dialog is a note: 'Note: Fields marked with an asterisk (*) are required for the comment to be accepted.' Below the note are three buttons: 'Delete Comment', 'OK', and 'Cancel'.

Wireshark - Packet 1 Comment

Comment Advanced

Scope*: Single Packet

Important Packet Identifier*: Entire Packet Payload

Program Used: test

Command: test

Add Keypresses Add Systemcalls

Confidence: 1

Description: test

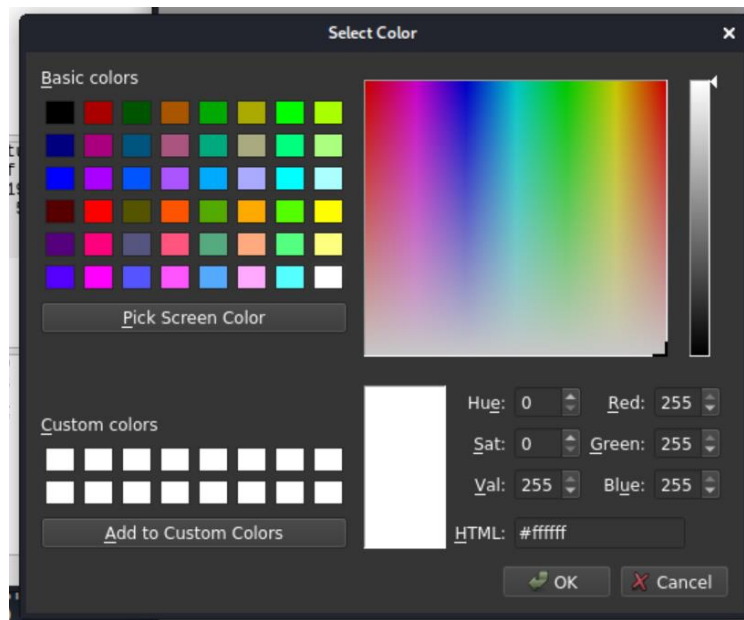
Note: Fields marked with an asterisk (*) are required for the comment to be accepted.

Delete Comment OK Cancel

Once the comment is added, it will appear in the packet comment view in the Timeline view.

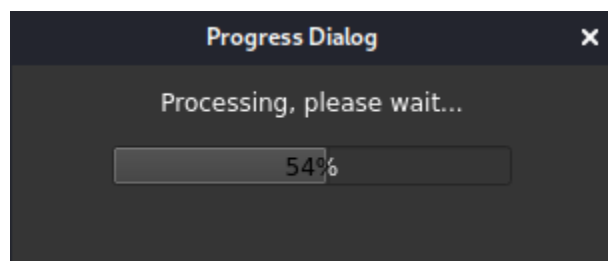
Color Selection

Once each dataline is selected, the user will be asked to select a color to assign to the dataline. The color selector looks like the following:

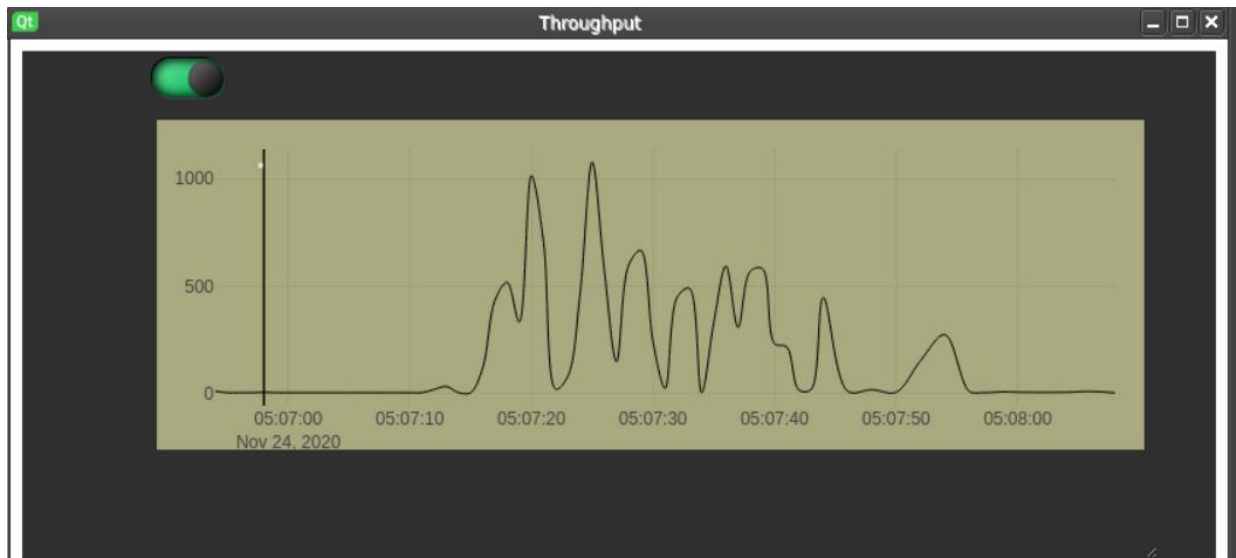
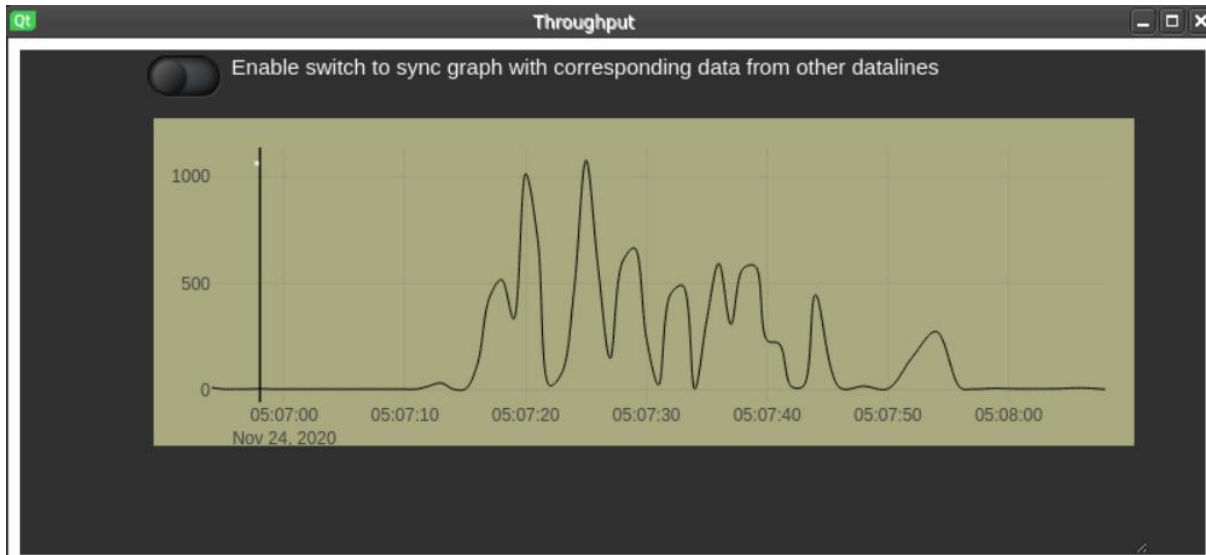


Throughput

The first dataline option is throughput. When the user selects this option, a progress dialog will be visible indicating that the dataline is loading.



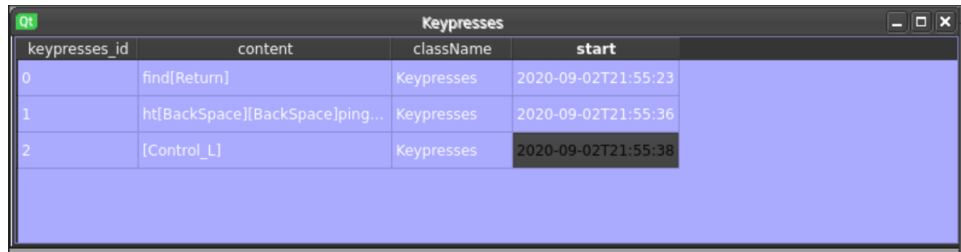
The following dataline will be shown in the timeline view.



This dataline is a dash graph render displaying the tshark network throughput data in time intervals.

Keypresses

The second dataline option is keypresses. When the user selects this option, the following dataline will be shown in the timeline view:

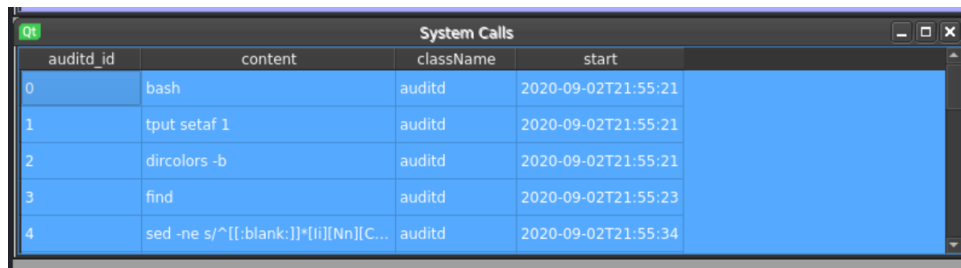


keypresses_id	content	className	start
0	find[Return]	Keypresses	2020-09-02T21:55:23
1	ht[BackSpace][BackSpace]ping...	Keypresses	2020-09-02T21:55:36
2	[Control_L]	Keypresses	2020-09-02T21:55:38

This dataline is a text render in a table display containing the parsed data from the keypresses JSON file. This table includes the id of the keypress (keypresses_id), the content, className, and the timestamp.

System Calls

The third dataline option is the system calls. When the user selects this option, the following dataline will be shown in the timeline view:



auditd_id	content	className	start
0	bash	auditd	2020-09-02T21:55:21
1	tput setaf 1	auditd	2020-09-02T21:55:21
2	dircolors -b	auditd	2020-09-02T21:55:21
3	find	auditd	2020-09-02T21:55:23
4	sed -ne s/^([[:blank:]]*)[Ii][Nn][C]...	auditd	2020-09-02T21:55:34

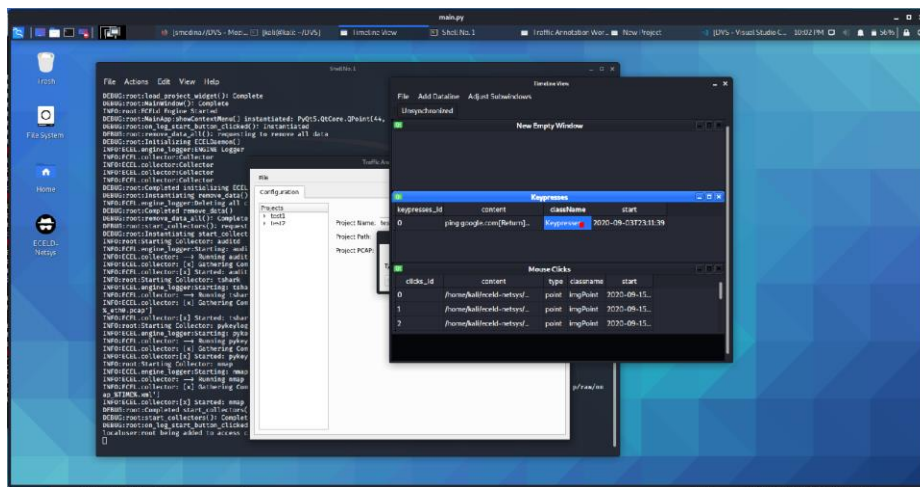
This dataline is a text render in a table display containing the parsed data from the auditdData JSON file. This table includes the auditd id, the system call, className and timestamp in which the system call was used.

Mouse Clicks

The fourth dataline option is the mouseclicks. When the user selects this option, the following dataline will be shown in the main window:

clicks_id	content	type	classname	start
0		point	imgPoint	2020-09-02T...
1		point	imgPoint	2020-09-02T...

This table includes the click id, the screenshot, the type, className and the timestamp of the click. The user can click on any of the images to view it in a new window.



Timed Screenshots

The last option is the Timed screenshots. When the user selects this option, the following dataline will be shown in the main window:

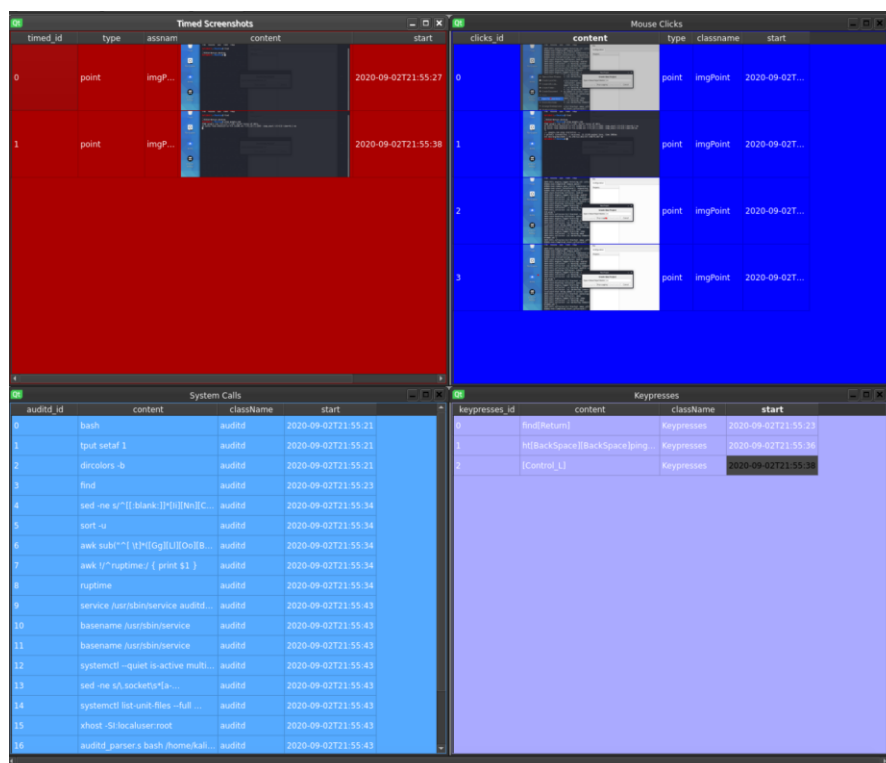
timed_id	type	assnam	content	start
0	point	imgP...		2020-09-02T21:55:27
1	point	imgP...		2020-09-02T21:55:38

This table includes all the screenshots that were timed, the `timed_id`, the `type`, `className` and the timestamp of the screenshot. The user can click on any of the images to view it in a new window.

The user is able to open, close and minimize each dataline window individually as well as displaying a combination of the datalines available at the same time.

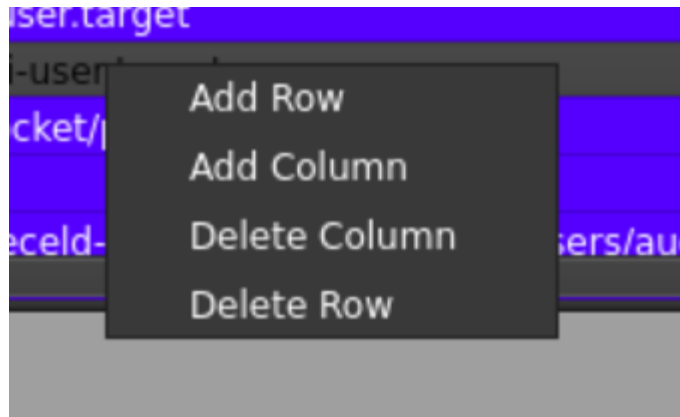
Adjust Subwindows

This option allows the user to readjust the windows in a tile formatting. The following is a sample of what happens when there are 4 windows open and this option is selected:



Context Menu

The context menu appears when the user right clicks on the datalines. The user is given 4 menu options: add row, add column, delete row, and delete column.



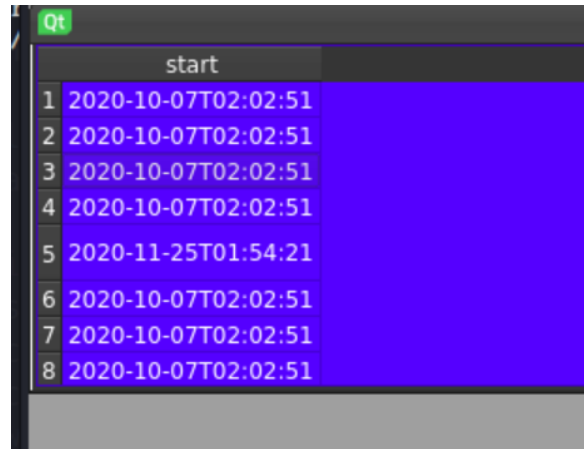
"Add row" adds a row to the table where the user right clicks. The row added has the timestamp of when the row was added. The user can edit the rows added

	auditd	2020-10-07T02:02:51
	auditd	2020-10-07T02:02:51
		2020-11-25T01:54:21
	auditd	2020-10-07T02:02:51
	auditd	2020-10-07T02:02:51

"Add Column" adds an empty and editable column anywhere the user right clicks on.

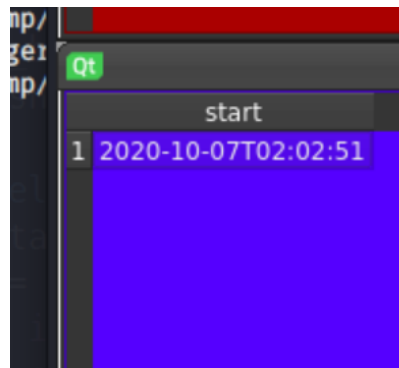
	className	4	start
	auditd		2020-10-07T02:02:51
	auditd		2020-10-07T02:02:51
	auditd		2020-10-07T02:02:51
	auditd		2020-10-07T02:02:51
			2020-11-25T01:54:21
	auditd		2020-10-07T02:02:51
	auditd		2020-10-07T02:02:51

"Delete Column" allows the user to delete any column.



	start
1	2020-10-07T02:02:51
2	2020-10-07T02:02:51
3	2020-10-07T02:02:51
4	2020-10-07T02:02:51
5	2020-11-25T01:54:21
6	2020-10-07T02:02:51
7	2020-10-07T02:02:51
8	2020-10-07T02:02:51

"Delete Row" allows the user to delete any row.



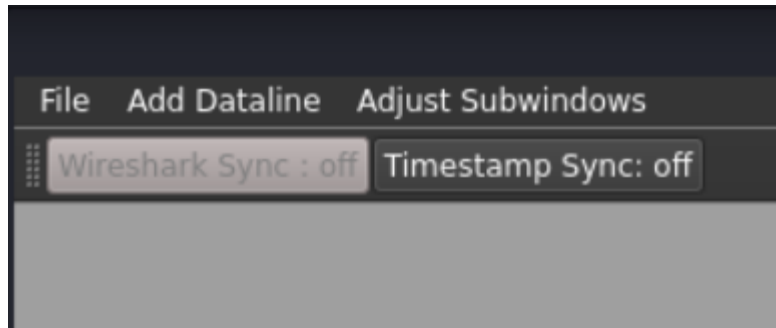
	start
1	2020-10-07T02:02:51

```
New Row Added in the SystemCalls Dataline
New Column Added in the SystemCalls Dataline
Column removed Added in the SystemCalls Dataline
Column removed Added in the SystemCalls Dataline
Column removed Added in the SystemCalls Dataline
Column removed Added in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline
Row removed in the SystemCalls Dataline

```

Buttons to enable/disable synchronization

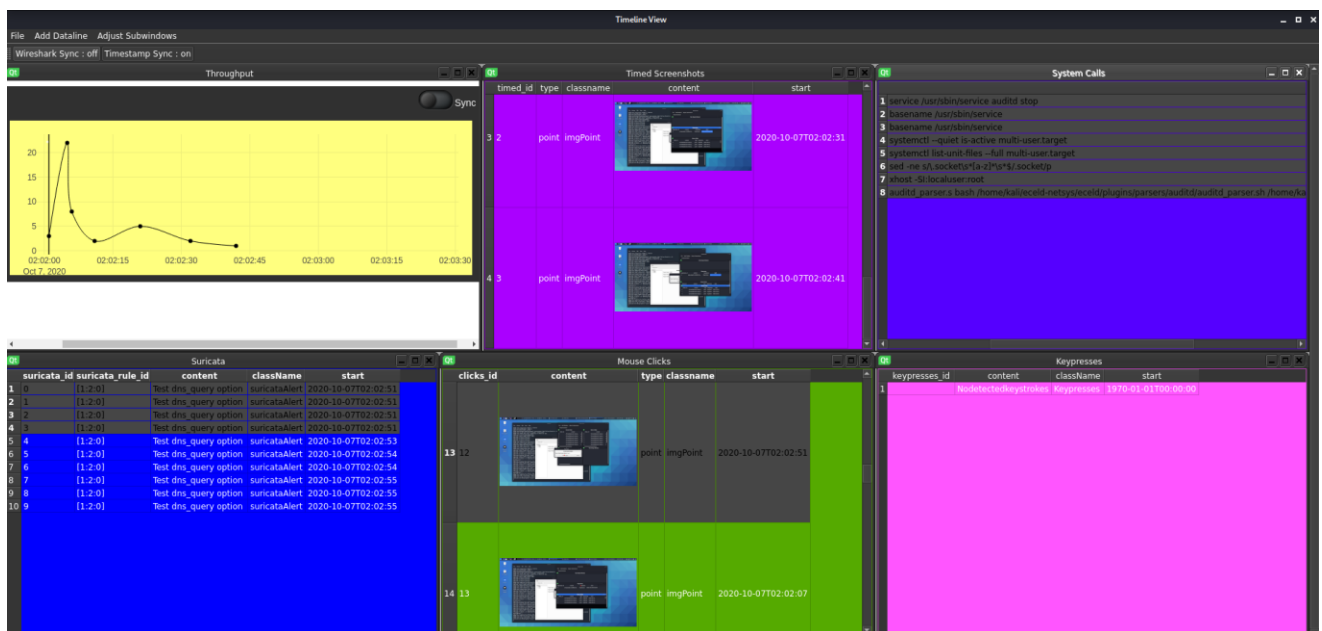
This feature will allow the user to synchronize views. The following are the sync buttons:



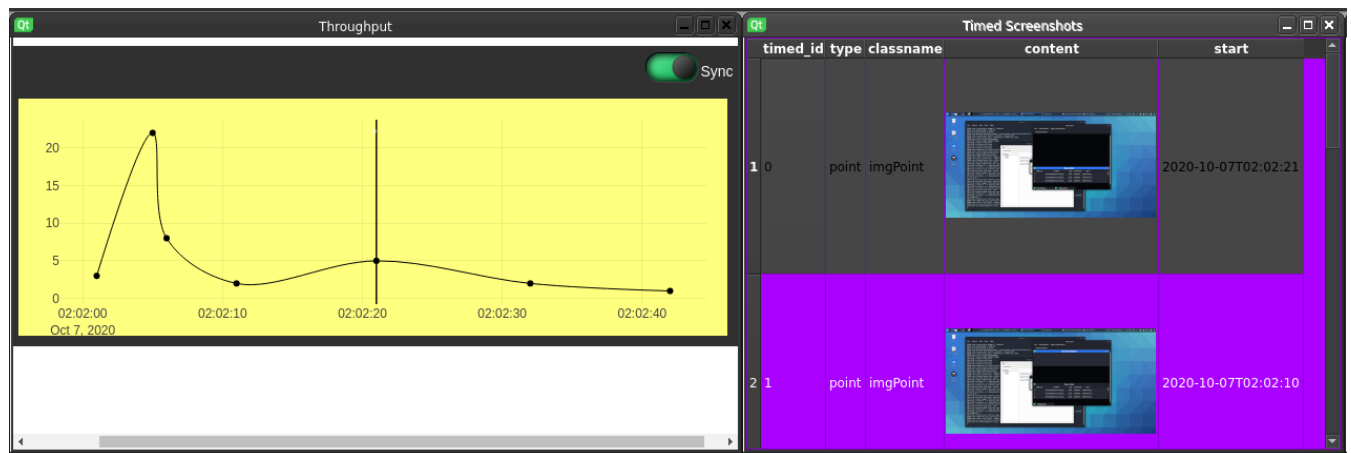
Timestamp Sync

When the user clicks on "Timestamp Sync: off" it will be updated to "Timestamp Sync: on." All the datalines in the timeline view will be synchronized according to the timestamp, highlighting the rows in the tables were the system finds a matching timestamp.

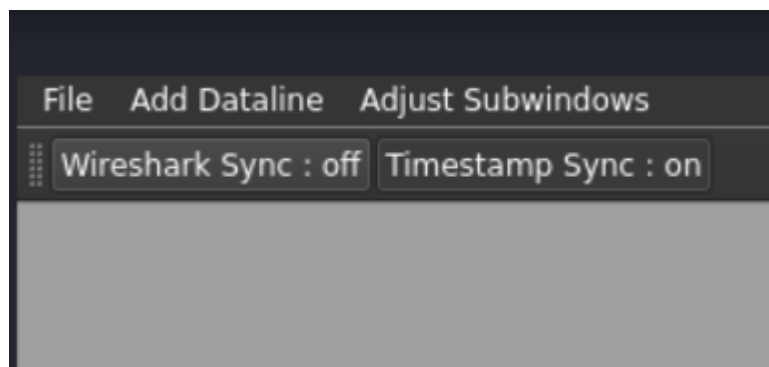
The user can select any row of any dataline to sync the datalines according to the timestamp selected. The following shows the sync behavior:



The Throughput has a sync toggle button, that when activated will send a signal to the system, this will sync the throughput when the user selects a timestamp from the other datalines.



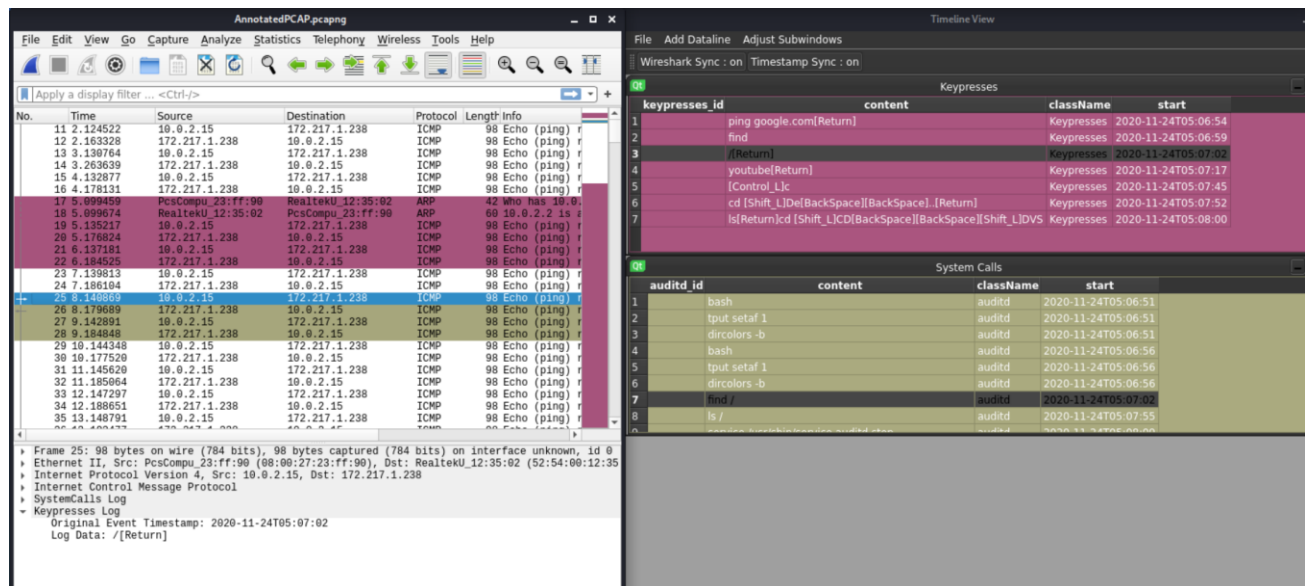
Initially, the Wireshark button is disabled and can only become clickable when Timestamp sync has been turned on:



It will become disabled whenever Timestamp sync is turned off again.

Wireshark Sync

As mentioned previously, Wireshark Sync can only be turned on if Timestamp sync is on. When the user clicks on "Wireshark Sync: off" it will be updated to "Wireshark Sync: on." Now, clicking a packet on the Wireshark view will highlight the corresponding data in the Timeline View and vice versa:

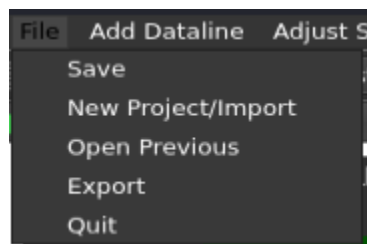


File Drop-Down Menu

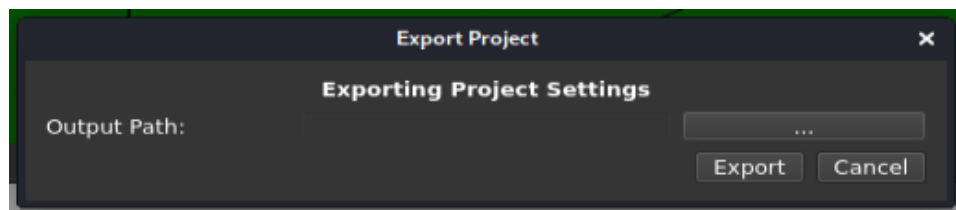
The DVS system now has more options for the file drop-down menu, however for iteration 5 only 2 functionalities work: Import, Export. If the user clicks on any other option the system will simply ignore the request. These include: Save, Open Previous.

File Drop-Down Menu (Export)

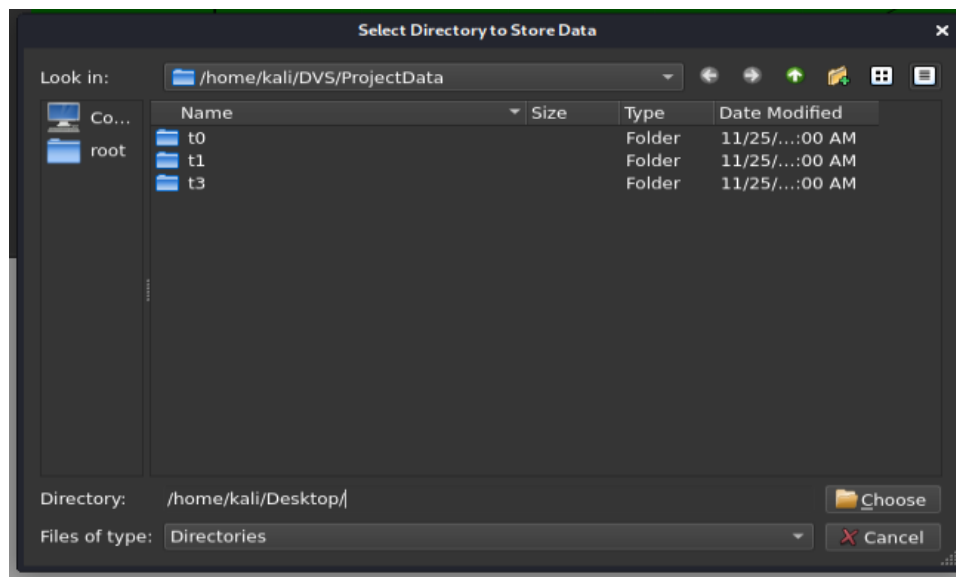
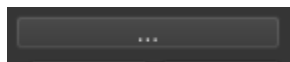
The DVS system is now able to export a working project to a directory that you choose, you will need to navigate to the tool bar on top and select the "File" tab.



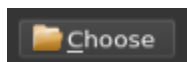
A drop down will show with different types of functionalities, to export click the "Export" selection.



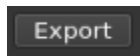
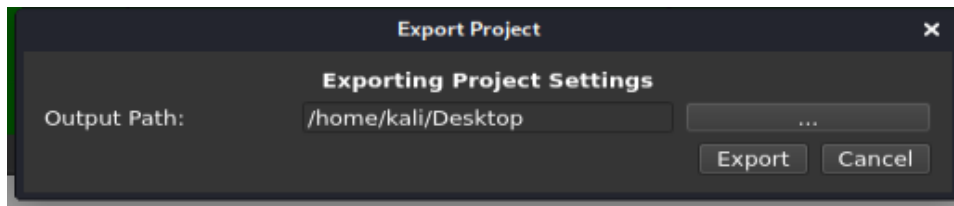
The DVS system will then display a pop-up window “Export Project” that will prompt the user to provide information about the location to export to. For this iteration, the user would be able to click on the image below to choose the directory.



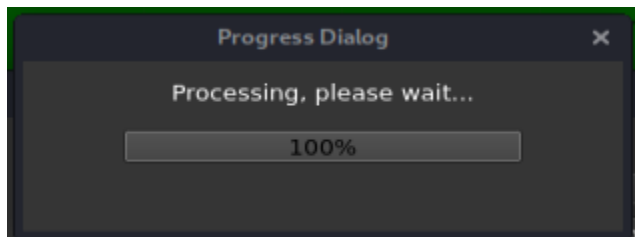
Once you manually enter the directory or navigate to the directory, click on the “choose” button showed in the image below.



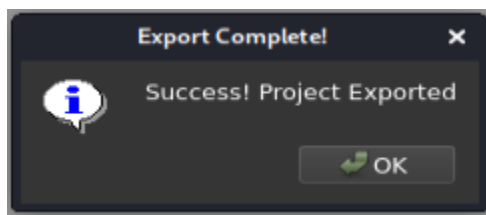
This will then populate the Output Path to the directories path you selected, now click on the export button.



While the DVS is exporting the project, it will provide the user with a progress dialog.



When finished, the DVS system will display a pop-up window notifying the user that the project has been exported.



Wireshark View

The DVS system uses a modified version of eceld-Wireshark in order to properly apply the sync and colorization functionalities. These are the files that were modified in the following directories:

- **/eceld-wireshark/wireshark-3.2.0/ui/qt/:**
 - packet_list.cpp
 - coloring_rules_dialog.cpp
 - coloring_rules_dialog.h

-
- main_window_slots.cpp
 - main_window.ui
 - **/eceld-wireshark/wireshark-3.2.0/:**
 - CMakeOptions.txt
 - Include build option "Build qtshark" ON

Appendix

Keywords

DVS - Data Visualization System

ECEld – Evaluator-Centric and Extensible Logger

ECEld-NetSys