

Rapport SAÉ 24 :Traiter des données

Département Réseaux et Télécommunications

Projet SNIFFER Ethernet - M.CAM / Mme. ESCAZUT

Par : ROUSSE Louane, MARTIN Claire, THIEBOT Alex
Nom de groupe : Rababa



TABLE DES MATIÈRES

I. Introduction.....	3
I. Présentation du projet.....	3
II. Rappel des objectifs.....	3
II. Exigences.....	4
0. Général.....	4
1. Programme.....	5
2. Base de Données.....	10
3. Site Web.....	11
III. Base de Données.....	15
a. Schéma logique de la base de données.....	15
b. Schéma de la base de données.....	16
c. Validation des 3 formes normales.....	17
IV. Site Web.....	18
1. Schéma de navigation.....	18
2. visualisation des pages du site web.....	20
V. Le code Python.....	22
I. Début du code.....	22
II. Lecture et récupération des données du fichier de configuration.....	23
III. Établissement d'une connexion entre notre programme et notre base de données...	24
IV. Filtrage et Extraction des données.....	25
V. Plan de Validation / Rapports & Anomalies.....	38
0. Général.....	38
I. Programme Python.....	39
II. Base de données.....	62
III. Site web.....	65
VI. Gestion de projet.....	79
1. RACI.....	79
2. Diagramme de GANTT.....	80
3. Définition et gestion des risques.....	81
4. Retour d'expérience.....	83
VII. Conclusion.....	84

I. Introduction

I. Présentation du projet

Suite à une première définition des exigences dans le cadre de la SAÉ 15, de la conception de nos algorithmes, du schéma de notre base de données et de la maquette de notre site web, il est maintenant temps de mettre tout cela en œuvre à travers la SAÉ 24.

Ainsi, à travers ce rapport, nous vous représenterons les exigences mises en place lors du semestre dernier avec des modifications qui auront été apportées, nous vous détaillerons ensuite la construction de notre base de données ainsi que de ses différentes tables. Suite à cela, vous y trouverez les rapports des différents tests effectués visant à s'assurer du bon fonctionnement de tout notre système. Enfin, vous y verrez l'explication de notre code python, puis de la gestion de projet, pour aboutir à notre retour d'expérience sur cette seconde partie de la SAÉ.

II. Rappel des objectifs

Notre problématique au premier semestre était, rappelons-le, la suivante :

Problématique :

Comment afficher clairement les trames des tests du sniffer Ethernet sur une page web, à partir d'un fichier brut ?

Ci-dessous les objectifs qui avaient été fixés auparavant, dans le cadre de la SAÉ 15.

Objectifs :

- 1) Réception d'un fichier en binaire brut d'un trafic Ethernet
- 2) Programme qui va traduire et décoder le fichier
- 3) Stockage et classification des données
- 4) Affichage des données lisibles sur un site web

De ce fait, notre objectif principal est donc désormais de réaliser tout cela à travers la SAÉ 24.

Ainsi dans un premier temps, nous vous rappelons dans la partie qui suit les exigences établies auparavant en ayant ajouté les nouvelles dans le but de s'assurer de la structure et des demandes auxquelles notre produit doit répondre.

I. Exigences

0. Général

SNIFFER _ETHERNET - General_001

Le système doit fonctionner sous Linux.

SNIFFER _ETHERNET - General_002

Le système doit utiliser des logiciels et librairies libres de droit pour les entreprises.

SNIFFER _ETHERNET - General_003

Le système doit fonctionner sur Raspberry PI 3.

SNIFFER _ETHERNET - General_004

Le système doit donner un résultat en 5 minutes maximum.

SNIFFER _ETHERNET - General_005

Le fichier binaire fourni en entrée doit être en .result_data .

1. Programme

SNIFFER_ETHERNET - Prog_101

Le programme doit être commenté à 70% minimum.

SNIFFER_ETHERNET - Prog_102

Les fichiers html et css doivent être validé par le W3C.

SNIFFER_ETHERNET - Prog_BinaireToAscii_101

Le programme doit lire les trames 800 et 806.

SNIFFER_ETHERNET - Prog_BinaireToAscii_102

Le programme doit différencier les trames 800 et 806.

SNIFFER_ETHERNET - Prog_BinaireToAscii_103

Le fichier des trames en entrée du programme doit être en binaire.

SNIFFER_ETHERNET - Prog - Prog_BinaireToAscii_104

Le programme doit convertir le binaire en ASCII.

SNIFFER_ETHERNET - Prog - TrameVersBdD_101

Le fichier “config” en entrée du programme doit toujours être de la même forme et être nommé “test_report.txt” en étant placé dans le dossier “/var/www/html/Configuration”.

Exemple ci-dessous :

```
*****
* T E S T   R E P O R T
=====
* Thales Alenia Space
=====
* Programme          : Unknown - .selected_conf_file missing
* Test phase         : Unknown - .selected_conf_file missing
* Tested SW          : OBSW
* Tested SW version/edition : 03.00.00
* SDB version/edition  : 04.02.01
* SGSE version/edition  : ATB 01.01.01
* Analyser tool version/edition : TRGC FULL JAVA V04.00.27
=====
* Execution test engineer    : pastoro
* Execution begin date     : "22-09-06 09-46-53"
* Execution end date       : "22-09-06 09-49-52"
* Execution status          : Nominal
=====
* Analyse checker mode      : On
* Analyse detail level      : Verbose
* Analyse date accuracy     : 10us
* Analyse start date        : 2022-09-06 08-14-40
=====
* Processor module id       : PM A
* Simulation date           : "2022-09-06 09-46-13"
* Format TM                 : BVL4000
=====
* Test                   : Vt_DEMO_power_on
* Test procedure version/edition : Revision:
*****
***** Permanent checks *****
*****
* Test case 100
*****
100.1.1      [03/04/2022 15:37:30.014560]          Start TM Archiving
(TMR)          ALL VC
100.2.1      [03/04/2022 15:37:30.015080]          Start UART
Archiving (Channel1)
100.3.1      [03/04/2022 15:37:30.015480]          Start UART
Archiving (Channel2)
```

```

100.4.1      [03/04/2022 15:37:30.015900]          Start UART
Archiving (Channel3)

100.5.1      [03/04/2022 15:37:30.016320]          Start UART
Archiving (Channel4)

100.6.1      [03/04/2022 15:37:30.016700]          Start UART
Archiving (Channel5)

100.7.1      [03/04/2022 15:37:30.017180]          Start UART
Archiving (Channel6)

100.8.1      [03/04/2022 15:37:30.017540]          Start UART
Archiving (Channel7)

100.9.1      [03/04/2022 15:37:30.017970]          Start UART
Archiving (Channel8)                      ERROR

100.10.1     [03/04/2022 15:37:30.018530]          Select transmit
mode           RAW

100.11.1     [03/04/2022 15:37:31.019390]          Wait 1000 ms

*****
* Test case 6400 conclusion
*****
* Number of actions      : 11
* Number of manual controls : 0
* Number of automatic controls passed : 0
* Number of automatic controls failed : 0
*****
* Test Vt_DEMO_power_on conclusion
*****
* Number of actions      : 251
* Number of manual controls : 13
* Number of automatic controls passed : 4
* Number of automatic controls failed : 2
* Analyse phase duration   : 00h00mn08s
*****

```

SNIFFER_ETHERNET - Prog - TrameVersBdD_107

Le programme doit remplir une base de données.

SNIFFER_ETHERNET-Prog-Fonction-Transfert_101

Concernant les fonctions transfert :

- Sont définies dans un fichier d'extension “.json”.
 - Elles doivent être fournies manuellement par l'utilisateur dans un fichier au nom de “ft.json”, dans des objets.
 - Les adresses IP et MAC concernées doivent toutes être renseignées au sein des objets json correspondants.
-

SNIFFER_ETHERNET-Prog-Fonction-Transfert_102

L'insertion des fonctions de transfert doivent correspondre au format ci-dessous :

```
{  
    "MAC": {"CLE1": "VALEUR1", "CLE2": "VALEUR2"},  
    "IP": {"CLE1": "VALEUR1", "CLE2": "VALEUR2"},  
    "PMID": {"PMIDcle1": "PMIDvaleur1", "PMIDcle2": "PMIDvaleur2"},  
    "FT_0": {"F0cle1": "F0valeur1", "F0cle2": "F0valeur0", "F0cle0": "F0valeur0"},  
    "FT_1": {"F1cle1": "F1valeur1", "F1cle2": "f1valeur2", "F1cle3": "F1valeur3"},  
    "FT_2": {"F2cle1": "F2valeur1", "F2cle2": "F2valeur2"},  
    "FT_3": {"F3cle1": "F3valeur1", "F3cle2": "F3valeur2", "F3cle3": "F3valeur3", "F3cle4": "F3valeur4", "F3cle5": "F3valeur5"},  
    "FT_4": {"F4cle1": "F4valeur1", "F4cle2": "F4valeur2", "F4cle3": "F4valeur3", "F4cle4": "F4valeur4", "F4cle5": "F4valeur5"},  
    "FT_5": {"F5cle1": "F5valeur1", "F5cle2": "F5valeur2", "F5cle3": "F5valeur3", "F5cle4": "F5valeur4"},  
    "FT_6": {"F6cle1": "F6valeur1", "F6cle2": "F6valeur2", "F6cle3": "F6valeur3", "F6cle4": "F6valeur4"},  
    "FT_7": {"0": "FT_2_label_1", "1": "FT_2_label_2"}  
}
```

SNIFFER_ETHERNET-Prog-Fonction-Transfert_103

Les fonctions de transferts doivent systématiquement être définies dans l'ordre qui suit :

→ MAC, IP, PMID, FT_0, FT_1, FT_2, FT_3, FT_4, FT_5, FT_6, FT_7

SNIFFER_ETHERNET-Prog -Fonction-Transfert_104

Le programme doit signaler l'utilisateur dans le cas où une fonction transfert est manquante.

SNIFFER_ETHERNET-Prog -Fonction-Transfert_105

Il doit toujours avoir onze lignes dans le fichier texte contenant les fonctions de transferts.

SNIFFER_ETHERNET-Prog -Fonction-Transfert_106

Une fonction de transfert est caractérisée par sa clé et par sa valeur.

2. Base de Données

Le modèle auquel la base de données doit répondre :

	bench_1	Frame Date	bench_3	bench_4	bench_5	bench_6	Frame Size
en octet	8	8	4		4		4
en bits	64	64	32	12	4	16	32
valeur fixe						0	
fonction transfert					FT_0		
commentaire		nb sec					nb octets

	MAC Dest @	MAC Source @	field_1	field_2	field_3	field_4	field_5	field_6	field_7	field_8	Soure IP @	Dest IP @	field_9	field_10	field_11	field_12	field_13	field_14	field_15	field_16	field_17	field_18
en octet	6	6	2	2	2	2	2	1	1	2	4	4	2	2	2	2					2	
en bits	48	48	16	16	16	16	16	8	8	16	32	32	16	16	16	16	3	1	1	3	3	5
valeur fixe										0x11						000		1				
fonction transfert	MAC @	MAC @									IP @	IP @					FT_7			FT_5	FT_2	
commentaire																						

	field_19	field_20	field_21	field_22	field_23	field_24	field_25	field_26	field_27	field_28	field_29	field_30	field_31	field_32	field_33	field_34	field_35	field_36	
en octet	2	2			1				1		2		1	1	2	2	2	2	n
en bits	2	14	16	4	1	1	1	1	2	6	6	10	8	8	16	16	16	16	8xn
valeur fixe				0000		0						0x00							0x0000
fonction transfert									FT_3	FT_4			FT_1						
commentaire															packet date				

SNIFFER_ETHERNET - BdD_201

Chaque table de la base de données doit avoir une clé primaire unique.

SNIFFER_ETHERNET - BdD_202

La base de données est constituée d'au minimum trois tables :

- CONFIG :

Contient les informations de configuration extraites du fichier texte (voir SNIFFER_ETHERNET-Prog-BdDVersHTML_101) pour chaque test effectué.

- TRAME800 :

Contient les données des trames de type “ARP” (Adresse Resolution Protocol)

- TRAME806 :

Contient les données des trames de type “UDP” (User Datagram Protocol)

SNIFFER_ETHERNET - BdD_203

La base de donnée doit respecter les 3 formes normales.

3. Site Web

SNIFFER_ETHERNET - Web_301

Le système doit afficher clairement le résultat de la trame sur une page web.

SNIFFER_ETHERNET - Web_302

Le site web doit être écrit en HTML, CSS et PHP.

SNIFFER_ETHERNET - Web_303

Le site web doit afficher les trames sous formes de tableaux.

SNIFFER_ETHERNET - Web_304

Les noms des tableaux doivent être modifiable via un fichier texte.

SNIFFER_ETHERNET - Web_305

Le fichier texte doit se trouver dans le même répertoire que les programmes et doit s'appeler "Labels.txt" afin d'être reconnu par le programme Python. Sa structure doit être la suivante :

```
Date
PMID
bench_3
bench_5
FrameSize
MACdest@
MACsource@
field1
field2
field3
field4
field5
field6
field7
sourceIP@
destIP@
field9
field10
field11
field14
field16
field17
field18
field20
field21
field23
field25
field26
field28
field29
field30
field32
PacketDate
nomtest
```

Il ne doit pas y avoir de ligne en plus ni de moins que sur cet exemple.

SNIFFER_ETHERNET - Web_306

Le programme doit être en php.

SNIFFER_ETHERNET - Web_307

L'utilisateur doit pouvoir sélectionner une trame en fonction de son nom de test et de sa date d'exécution.

SNIFFER_ETHERNET - Web_308

Le programme doit être intégré dans le code HTML

SNIFFER_ETHERNET - Web_309

L'utilisateur doit pouvoir sélectionner le nombre de trames qu'il souhaite voir affichées sur le site web, au choix : 10, 20, 50 ou 100 trames par page.

SNIFFER_ETHERNET - Web_310

Le site web doit afficher automatiquement 10 trames si aucun nombre n'est spécifié.

SNIFFER_ETHERNET - Web_311

Le site web doit toujours afficher la ligne du tableau contenant le noms des champs.

SNIFFER_ETHERNET - Web_312

Le site web doit avoir la possibilité de supprimer une trame.

SNIFFER_ETHERNET - Web_313

Le site peut s'adapter à différents navigateurs internet

SNIFFER_ETHERNET - Web_314

Le site web devra être ouvert manuellement à l'adresse “localhost/Site/accueil.html” saisi dans l'URL d'un navigateur Internet.

II. Base de Données

a. Schéma logique de la base de données

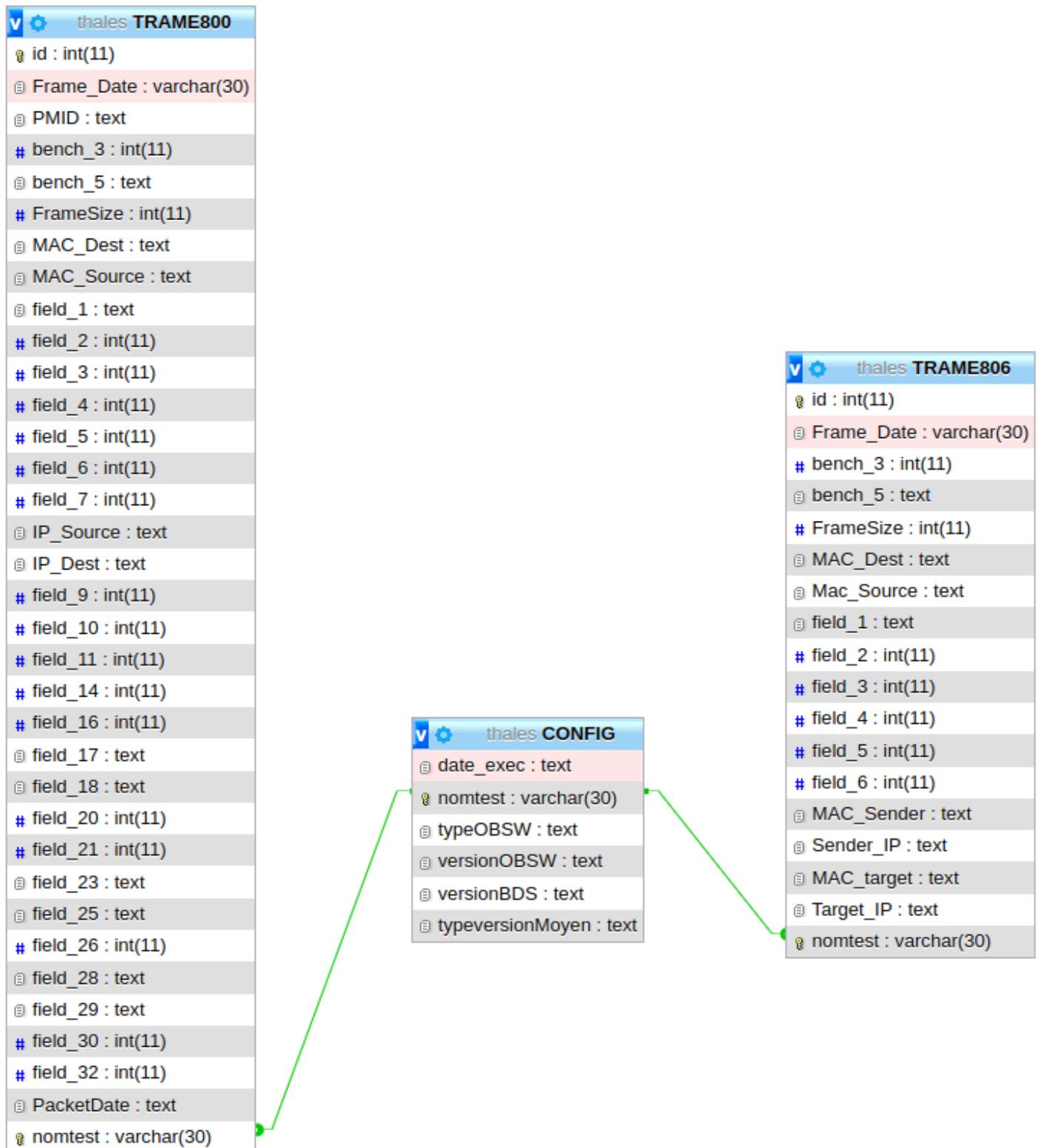
Nous avons fait le choix de créer deux tables au sein de notre base de données, à savoir la table TRAME qui aura pour rôle de récupérer toutes les informations d'une trame, puis la table CONF qui elle prendra en compte les informations principales de configuration d'un test en les récupérant par l'intermédiaire d'un fichier texte les contenant. La structure de la base est donc la suivante :

CONFIG (nomtest, date_exec, typeOBSW, versionOBSW, versionBDS, typeversionMoyen)

TRAME800 (id, Frame_Date, PMID, bench_3, bench_5, FrameSize, MAC_Dest,
MAC_Source, field_1, field_2, field_3, field_4, field_5, field_6,
IP_Source, IP_Dest, field_9, field_10, field_11, field_14, field_16, field_17,
field_18, field_20, field_21, field_23, field_25, field_26, field_27, field_28,
field_29, field_30, field_32, PacketDate, #nomtest)

TRAME806 (id, Frame_Date, bench_3, bench_5, FrameSize, MAC_Dest, MAC_Source,
field_1, field_2, field_3, field_4, field_5, field_6, MAC_Sender, Sender_IP,
MAC_target, Target_IP, #nomtest)

b. Schéma de la base de données



c. Validation des 3 formes normales

Formes Normales	Contraintes	Vérification
1NF	Tous les champs de toutes les tables ne prennent qu'une valeur atomique.	Chaque champ de la base contient uniquement qu'une information ayant un sens propre.
2NF	La 1NF est validée et aucun des champs de la table ne dépend d'une partie de clé primaire.	Nos tables ne contiennent pas de clé composée.
3NF	La 2NF et la 3NF sont validées et aucun des champs des tables ne dépendent d'un autre champ non clé.	Chaque field nous fournit une donnée qui ne dépend pas des autres données des autres fields. En d'autres mots, chaque champ comporte des données qui ont leur signification propre.

III. Site Web

1. Schéma de navigation

En ce qui concerne l'architecture du site web, vous visualiserez un schéma représentatif ci-dessous.

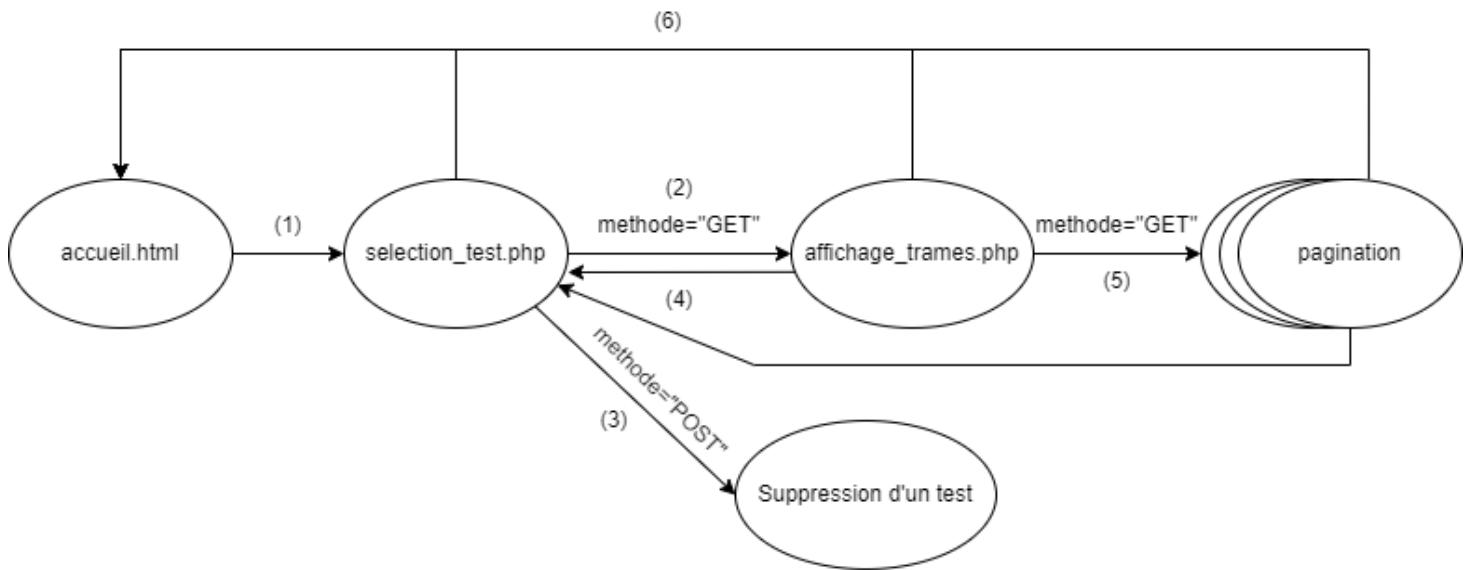


Schéma structurel de navigation de notre site web.

Pour plus de précisions :

- En (1) nous passons de la page d'accueil "accueil.html" qui est la première page nous arrivons, à notre page de sélection "selection_test.php" par l'intermédiaire d'une balise html de type <button> redirigeant vers la page de sélection, comme ci-dessous :

```
<button class="btn-first b1" onclick="window.location.href='./selection_test.php'">Entrer dans le site</button>
```

- Lorsque nous arrivons sur la page de sélection de la trame à afficher, nous rencontrons deux formulaires :
 - Le formulaire (2) consiste à sélectionner une trame via son nom et sa date d'émission. Ces informations seront retransmises via la méthode "GET" vers

la page “affichage_trames.php” qui affichera les données souhaitées en fonction de ce qui est contenu dans le tableau associatif php \$_GET[].

- Le formulaire (3) a pour but de permettre à l’utilisateur de supprimer une trame de la base de données s’il le souhaite “suppression d’un test” sur le schéma représente une fonctionnalité et non une page à part entière. La méthode utilisée pour cette transmission de données est la méthode “POST”. Nous avons préféré utiliser la méthode post car la suppression d’un test représente une manipulation sensible. En effet, cette méthode envoie les données dans le corps de la requête HTTP plutôt que de les inclure dans l’URL. Cela rend les données moins visibles et plus sécurisées, car elles ne sont pas exposées dans l’historique du navigateur ou dans les journaux du serveur.
- La liaison (4) permet de revenir sur la page de sélection des tests, si l’utilisateur souhaite en visualiser un autre.
- (5), lorsque nous arrivons sur la page “affichage_trames.php”, il est alors possible de visualiser l’entièreté des données souhaitées par l’utilisateur. Rentre cependant en compte un paramètre de pagination, qui offre le choix à l’utilisateur d’afficher 10,50 ou 100 trames par page. Ce choix sera passé en variable via l’URL en utilisant la méthode “GET”.
- La liaison (6) permet de revenir sur l’accueil, simplement en cliquant sur le logo Thalès du site.

2. visualisation des pages du site web.

Voici un visuel de notre site web.

accueil.html-1ere page :

La première page est l'accueil, celle sur laquelle l'utilisateur arrive.

selection_test.php-2eme page :

La deuxième page est celle de la sélection du test que l'utilisateur veut visionner. Les deux boutons servent respectivement à :

- sélectionner un test à supprimer
- sélectionner un test à afficher en fonction du nom du test et de sa date d'exécution.

affichage_trames.php-3eme page :

La troisième page est celle où s'affichent toutes les trames sélectionnées, par groupe de 10 par défaut.

accueil.html-1ere page:



selection_test.php-2eme page :

**PROJET SNIFFER
ETHERNET**

Nom du test et date

VT_DEMO_P1
Afficher

Supprimer un test

VT_DEMO_P1
Afficher

affichage_trames.php-3eme page :

**PROJET SNIFFER
ETHERNET**

Importer un fichier
Nombre de trames par page : 10

Frame	Frame_Date	bench_3	bench_5	FrameSize	MAC_Dest	MAC_Source	field_1	field_2	field_3	field_4	field_5	field_6	fie
numero 1	04/03/2022 15:37:01.408434	3	0	210	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	192	10719	0	64	17
numero 2	04/03/2022 15:37:01.433417	3	0	94	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	76	10720	0	64	17
numero 3	04/03/2022 15:37:01.508373	3	0	350	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	332	10721	0	64	17
numero 4	04/03/2022 15:37:01.533608	3	0	98	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	80	10722	0	64	17
numero 5	04/03/2022 15:37:01.533614	3	0	590	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	572	10723	0	64	17
numero 6	04/03/2022 15:37:01.608567	3	0	246	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	228	10724	0	64	17
numero 7	04/03/2022 15:37:01.633551	3	0	210	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	192	10725	0	64	17
numero 8	04/03/2022 15:37:01.683524	3	0	94	00:1f:29:57:94:f2	00:10:ec:01:e5:43	0x800	17664	76	10726	0	64	17
numero	04/03/2022												

Numéro de page : Page 1

IV. Le code Python

Ci-dessous vous sera expliqué le fonctionnement des différentes parties de notre programme Python, dont la majeure partie de notre code se situe dans le fichier “lire_trame.py”.

I. Début du code

Les différents modules que nous avons importé dans notre programme sont les suivants :

```
#!/usr/bin/python3
import struct
import datetime
from datetime import timedelta
import mysql.connector
import json
```

- Le module ‘struct’ en Python permet de manipuler des données binaires en les convertissant en objets Python structurés et vice versa, il sera utilisé essentiellement dans la partie “filtrage des données issues du fichier binaire brut” afin de récupérer les différents types de données qui, dans notre cas précis, sont structurées en big endian, dont le symbole est : “>”.
- Le module ‘datetime’ en Python fournit des classes et des fonctions pour travailler avec des dates, des heures et des durées de manière précise. De ce module nous avons importé également ‘timedelta’ qui est utilisée pour représenter une durée spécifique.
- Le module ‘mysql.connector’ nous a apporté les outils nécessaires pour initialiser une connexion avec une base de données dans le but d’y insérer des données précises, à l’aide de curseurs.
- Le module ‘json’ est utilisé dans la gestion des fonctions de transfert. Ce module permet de stocker des informations structurées.

Les notions venant d’être énoncées seront plus précisément décrites dans les différentes parties correspondantes de notre programme Python : lire_trame.py

II. Lecture et récupération des données du fichier de configuration.

```
# Programme qui permet de récupérer des informations d'un fichier texte

with open ("..../Configuration/test_report.txt","r") as fichier:           #lit le fichier

    ligne_voulu=[]

        mot_clé=("* Tested SW             :","* Tested SW version/edition   :","* SDB
version/edition          :","* SGSE version/edition       :","* Execution begin date
:","* Test                :")

    "si une de ces chaines de caractères se trouve quelque part sur une ligne de "ligne_fichier" alors
elle est stocké dans "ligne_voulu" ""

    for line in fichier:           #Pour chaque ligne dans le fichier
        ligne=line.rstrip()         # Suppression des caractères de saut de ligne '\n' avec .rstrip()
        for i in range(len(mot_clé)):
            if mot_clé[i] in ligne:  # On vérifie dans chaque ligne s'il y un mot clé
                #print(line)
                ligne_voulu.append(ligne) #On ajoute la ligne au tableau ligne_voulu[]

    #print(ligne_voulu)
    information_voulu=[ ]

    for i in ligne_voulu:
        liste_ligne_voulu=[ ]
        liste_ligne_voulu=i.split(":")
    #sépare ligne_voulu en morceaux séparées par un ":" et stocke le tout dans "liste_ligne"

    information_voulu.append(liste_ligne_voulu[1].strip())
    #prend la deuxième partie de "liste_ligne_voulu" et la stocke dans "information_voulu"

    # On crée les variables qui suivent pour stocker les informations :
    versionOBSW = ""
    typeOBSW = ""
    versionBDS = ""
    typeVersionMoyen = ""
    execution_begin_date = ""
    nomDuTest = ""

    if len(information_voulu) >= 1:
        versionOBSW = information_voulu[0]
    if len(information_voulu) >= 2:
        typeOBSW = information_voulu[1]
    if len(information_voulu) >= 3:
        versionBDS = information_voulu[2]
    if len(information_voulu) >= 4:
```

```

typeVersionMoyen = information_voulu[3]
if len(information_voulu) >= 5:
    date_exec = information_voulu[4] #date d'exécution
if len(information_voulu) >= 6:
    nomDuTest = information_voulu[5]

```

#On s'assure à chaque pas que le tableau 'information_voulu' contient tous les éléments requis puis on les extrait successivement dans des variables correspondantes.

```

# Affichage des variables contenant les informations
# print("versionOBSW:", versionOBSW)
# print("typeOBSW:", typeOBSW)
# print("versionBDS:", versionBDS)
# print("typeVersionMoyen:", typeVersionMoyen)
# print("date_exec:", date_exec)
# print("nomDuTest:", nomDuTest)

```

III. Établissement d'une connexion entre notre programme et notre base de données.

- Nous avons initialisé la connexion entre notre base de données SQL s'intitulant "thales" et notre programme Python, avec le code suivant, utilisant le module mysql.connector :

```
# On établit la connexion avec la base de données sur le serveur local
```

```

conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="rababa",
    database="thales"
)

```

- La variable conn représente l'établissement de la connexion.
- 'host=localhost' spécifie l'hôte sur lequel la base de données hébergée localement est en cours d'exécution : dans une base MySQL sur un serveur PHPMyAdmin.
- Les mentions 'user' et 'password' concernent le renseignement des identifiants de connexion
- Enfin nous précisons le nom de la base de données : "thales".

Nous passons ensuite à l'ajout des données venant d'être identifiées et stockées dans des variables au sein de notre base de données. Pour cela on effectue une requête sql d'insertion. Ainsi dans cet exemple nous récupérons les données du programme précédent visant à récupérer les données du fichier de configuration.

```
# Ajout des données dans la base thales :  
sql = "INSERT INTO CONFIG (date_exec, nomtest, typeOBSW, versionOBSW, versionBDS,  
typeversionMoyen) VALUES(%s, %s, %s, %s, %s, %s)"
```

- Ainsi nous insérons les données du fichier de configuration dans la table ‘CONFIG’ de notre base de données, dans les champs entre parenthèses qui correspondent aux noms des colonnes de chaque enregistrement.
- Les données à insérer sont indiquées par les marqueurs de positionnement ‘%s’ il doit avoir autant de ces marqueurs que de valeurs à insérer.

```
# Données à insérer :  
donnees = (date_exec, nomtest, typeOBSW, versionOBSW, versionBDS, typeversionMoyen)
```

- Les valeurs stockées dans données seront mises successivement dans chaque colonne d’un enregistrement.
- On exécute ensuite la requête ‘sql’ avec notamment en paramètres les données, puis on ferme la connexion une fois que l’enregistrement complet a été saisi.

```
# Exécuter la requête :  
cursor.execute(sql, donnees)  
conn.commit()  
cursor.close()  
conn.close()
```

IV. Filtrage et Extraction des données.

- Voici la majeure partie de notre programme.

Est pris en compte le traitement des deux types de trames : UDP et ARP, reconnaissables grâce à la valeur du field_1.

IV.a Définition des référentiels temporelles avec le module Python ‘datetime’.

```
# Définition des dates de référence avec le module datetime :  
  
dateRef1 = datetime.datetime(1970,1,1,0,0,0) #année, mois, jour, heure, minute, seconde  
dateRef2 = datetime.datetime(2000,1,1,12,0,0) #année, mois, jour, heure, minute, seconde
```

- Nous avons défini deux dates de références pour extraire la date d’émission pour chaque trame et la date d’émission de paquet.

- ‘dateRef1’, soit le 1er janvier 1970 sera utilisé pour référencer les dates concernant les trames du protocole ARP (0x800).
- ‘dateRef2’ soit le 1er janvier 2000 pour les dates des trames UDP (0x806).

IV.b Récupération des objets json dans le but d'appliquer les fonctions de transfert définies par l'administrateur.

```
#----- RÉCUPÉRATION DES FONCTIONS DE TRANSFERT
#----- FICHIER JSON "ft.json"
```

```
with open("ft.json") as file:
    fonctions = json.load(file)
    MAC=fonctions["MAC"]
    IP=fonctions["IP"]
    PMID_=fonctions["PMID"]
    FT_0=fonctions["FT_0"]
    FT_1=fonctions["FT_1"]
    FT_2=fonctions["FT_2"]
    FT_3=fonctions["FT_3"]
    FT_4=fonctions["FT_4"]
    FT_5=fonctions["FT_5"]
    FT_6=fonctions["FT_6"]
    FT_7=fonctions["FT_7"]
```

```
#-----
```

- Nous récupérons les objets json contenus dans le fichier ‘ft.json’, convertit en dictionnaire Python.

IV.c Fonction principale

- Dans cette partie, le programme python écrit figurera dans la fonction principale “lire_trame(input)”.

```
# Définition de la fonction principale pour lire et récupérer les données de la trame jusqu'à
FrameSize :

def lire_trame(input) :
    with open(input, "rb") as fic :
```

PMID_non_def = 0

```

bench_5_non_def = 0
MAC_non_def = 0
IP_non_def = 0
f17_non_def = 0
f18_non_def = 0
f28_non_def = 0
f29_non_def = 0
f32_non_def = 0

#On définit des variables qui nous serviront dans la suite, dans le but d'avertir l'utilisateur
#si une fonction de transfert n'a pas été définie dans le fichier "ft.json".

nbrTrame = 0
nbrTrame806 = 0
nbrTrames800 = 0

data = fic.read(28) #on lit les 28 premiers octets de l'entête de la trame
while data : #tant que l'on parcours le fichier
    # print("-----")
    # print()
    #print("Trame num",nbrTrame+1, ":")
    # print()
    # print("-----")

    # --- On récupère la date, bench_3, bench_5 et framesize :

    date1, date2, bench_3, bench456, framesize = struct.unpack('> ddlll', data)
    FrameDate = dateRef1 + timedelta(0, date2)

#on met la date au bon format en ajoutant le nbr de seconde depuis le 1er janvier 1970,
la 'dateRef1'.

    Frame_Date = FrameDate.strftime('%d/%m/%Y %H:%M:%S.%f') #j/m/a H:m:s.ms
#On définit le format de la date d'émission de la trame.

```

- Après avoir récupéré chaque donnée, nous les avons traitées au cas par cas, indépendamment les unes des autres.

```

# --- Affichage bench_3
# print('bench_3 : ', bench_3)

# --- On définit un masque et on l'applique pour récupérer bench5
masque = 0b00000000000011110000000000000000
bench_5 = (bench456 & masque) >> 16
bench_5=str(bench_5)

```

```
# --- Fonction transfert
for cle in FT_0:                      #pour les clé dans fct0
```

```

if cle == str(bench_5):      #si clé = val bench5
    bench_5_ft = FT_0.get(str(cle))
    if str(bench_5) not in FT_0:
        bench_5_ft = '<undefined>'
        bench_5_non_def+=1
    bench_5=str(bench_5)
    #print('bench5 :',bench_5 ) #valeur fixe : 0

    #print("FrameSize :", framesize)

```

```

# --- Lecture de la suite des données et stockage dans des variables :

data = fic.read(14) # On lit la partie de la trame qui nous intéresse jusqu'au field1
MACdest, MACsource, f1 = struct.unpack('> 6s6sH', data)

```

--- RÉCUPÉRATION ET AFFICHAGE ADRESSES MACS

```

# --- MAC DESTINATION
MACdest = ':'.join('{:02x}'.format(octet) for octet in MACdest)

# --- fonction transfert
for cle in MAC :
    if cle == str(MACdest) :
        MACdest_ft = MAC.get(str(cle))
        if str(MACdest) not in MAC:
            MACdest_ft = '<undefined>'
            MAC_non_def +=1
    #print('MACdest : ', MACdest_ft)

```

- Concernant la partie du code qui traite les fonctions de transfert pour chaque donnée récupérée dans une trame, nous pouvons voir ci-dessus par exemple pour associer l'adresse MAC à une fonction de transfert, nous allons parcourir le dictionnaire MAC et allons vérifier si une des clés est égale à l'adresse MAC lue. Si c'est le cas, nous lui attribuons la valeur dans la clé du dictionnaire correspondante. Si ce n'est pas le cas, alors nous renvoyons à l'utilisateur qu'il manque une association dans le fichier "ft.json" en incrémentant la variable MAC_non_def pour signifier qu'une adresse trouvée n'a pas été définie dans les fonctions de transfert.
- On fait de même pour toutes les données nécessitant une association à une fonction de transfert suivantes.

```

# --- MAC SOURCE
MACsource = ':'.join('{:02x}'.format(octet) for octet in MACsource)

# --- fonction transfert

```

```

for cle in MAC :
    if cle == str(MACsource) :
        MACsource_ft = MAC.get(str(cle))
    if str(MACsource) not in MAC:
        MACsource_ft = '<undefined>'
        MAC_non_def+=1
    #print(MACsource_ft)

```

- Ensuite, nous vérifier le “field_1”, afin de savoir qu’elle type de trame, UDP ou ARP nous sommes en train de lire, tel que :

```

# --- VERIFICATION DE LA VALEUR DU TYPE DE TRAME (ARP 0x800 ou UDP 0x806)

f1 = hex(f1)

```

- Si la valeur hexadécimale (le champs Ethernet de la trame) field_1 est égal à “0x800”, alors nous allons lire la trame en fonction d’une structure propre à ce type de trame :

```

#SI 800 : -----
if f1 == "0x800" :
    reste = framesize - 60 #trames inutiles
    data = fic.read(framesize-reste-14)

#on lit la partie de la trame du field1 jusqu'a la fin

    f2, f3, f4, f5, f6, f7, f8, sourceIP, destIP, f9, f10, f11, f12, f13_14_15_16_17_18,
f19_20, f21, f22_23_24_25_26, f27_28, f29_30, f31, f32, f33_f34, f35 = struct.unpack('>
HHHHBBHIIHHHHHHBBHBBBIH', data)
    #print(f1)

```

- On traite les adresses IPs des trames ARP :

```
# RÉCUPÉRATION ET AFFICHAGE ADRESSES IPs
```

```

-----
```

```

# --- IP SOURCE
sourceIP = ''.join(str(b) for b in struct.unpack('BBBB', struct.pack('>I',
sourceIP)))
# --- fonction transfert
for cle in IP :
    if cle == str(sourceIP) :
        sourceIP_ft = IP.get(str(cle))
    if str(sourceIP) not in IP:
        sourceIP_ft = '<undefined>'
        IP_non_def+=1

```

```

#print('source IP : ',sourceIP,' ',sourceIP_ft)
#print('sourceIP : ', sourceIP)

# --- IP DESTINATION
destIP = '.'.join(str(b) for b in struct.unpack('BBBB', struct.pack('>I', destIP)))
# --- fonction transfert
for cle in IP :
    if cle == str(destIP) :
        destIP_ft = IP.get(str(cle))
    if str(destIP) not in IP:
        destIP_ft = '<undefined>'
    IP_non_def+=1
#print('destIP : ', destIP_ft)

```

- Ci-dessous le code visant à traiter ensuite tous les champs des trames ARP :

RÉCUPÉRATION ET AFFICHAGE DES FIELDS

```

# f1
# print('f1 : ', f1)

# f2
# print('f2 : ', f2)

# f3
# print('f3 : ',f3)

# f4
# print('f4 : ', f4)

# f5
# print('f5 : ', f5)

# f6
# print('f6 : ', f6)

# f7
# print('f7 : ', f7)

# f9
# print('f9 : ', f9)

# f10
# print('f10 : ', f10)

# f11
# print('f11 : ', f11)

# on définit un masque et on l'applique pour récupérer f14
masque = 0b0000000000000001
f14 = (f13_14_15_16_17_18 >> 12 ) & masque
#print('f14 : ',f14)

```

```

# on définit un masque et on l'applique pour récupérer f16
f16 = (f13_14_15_16_17_18 >> 8) & masque
#print('f16 :',f16)

# on définit un masque et on l'applique pour récupérer f17
masque = 0b000000000000000111
f17 = (f13_14_15_16_17_18 >> 5) & masque
f17=hex(f17)
# --- fonction transfert
for cle in FT_5:                      #pour les clé dans fct0
    if cle == str(f17):      #si clé = val bench5
        f17_ft = FT_5.get(str(cle))
    if str(f17) not in FT_5:
        f17_ft = '<undefined>'
    f17_non_def+=1
    f17=str(f17)
#print('f17 : ',f17_ft)

# on définit un masque et on l'applique pour récupérer f18
masque = 0b00000000000011111
f18 = (f13_14_15_16_17_18 & masque)
# --- fonction transfert
for cle in FT_2 :
    if cle == str(f18) :
        f18_ft = FT_2.get(cle)
    if str(f18) not in FT_2:
        f18_ft = '<undefined>'
    f18_non_def+=1
    f18_str=str(f18)
#print('f18 : ',f18_ft)

# on définit un masque et on l'applique pour récupérer f20
masque = 0b00111111111111111
f20 = f19_20 & masque
#print('f20 : ',f20)

# f21
#print('f21 : ', f21)

# on définit un masque et on l'applique pour récupérer f23
masque = 0b00000001
f23 = hex((f22_23_24_25_26 >> 3) & masque)
#print('f23 : ', f23)

# on définit un masque et on l'applique pour récupérer f25
masque = 0b00000001
f25 = hex((f22_23_24_25_26 >> 1) & masque)
#print("f25 : ", f25)

# on définit un masque et on l'applique pour récupérer f26
masque = 0b00000001

```

```

f26 = f22_23_24_25_26 & masque
#print('f26 :',f26)

# on définit un masque et on l'applique pour récupérer f28
masque = 0b00111111
f28 = f27_28 & masque

# --- fonction transfert
for cle in FT_3 :
    if cle == str(f28) :
        f28_ft = FT_3.get(cle)
    if str(f28) not in FT_3:
        f28_ft = '<undefined>'
    f28_non_def+=1
    f28_str=str(f28)
    #print('f28 : ',f28_ft)

# on définit un masque et on l'applique pour récupérer f29
masque = 0b00000000000111111
f29 = (f29_30 >> 10 ) & masque

# --- fonction transfert
for cle in FT_4 :
    if cle == str(f29) :
        f29_ft = FT_4.get(cle)
    if str(f29) not in FT_4:
        f29_ft = '<undefined>'
    f29_non_def+=1
    f29_str=str(f29)
    #print('f29 : ',f29_ft)

# on définit un masque et on l'applique pour récupérer f30
masque = 0b0000001111111111
f30 = f29_30 & masque
#print('f30 : ',f30)

# f32
# --- fonction transfert
for cle in FT_1 :
    if cle == str(f32) :
        f32_ft = FT_1.get(cle)
    if str(f32) not in FT_1:
        f32_ft = '<undefined>'
    f32_non_def+=1
    f32_non_def += 1 #f32 n'est pas défini
    f32=str(f32)
    #print('f32 : ',f32_ft)

```

- On définit le champ de date d'émission du paquet, en choisissant cette fois-ci une autre date de référence comme il nous a été demandé, à savoir le 1er janvier 2000.

```

# f33_f34_f35 : PacketDate

    f35 = f35/(2**16) #secondes
    f33_f34 = f33_f34 #date, heures
    f33_34_35 = f33_f34 + f35 #date, heures.secondes

    PacketDate = dateRef2 + timedelta(0, f33_34_35)

#on met la date au bon format en ajoutant le nbr de seconde depuis dateRef

    Packet_Date = (PacketDate).strftime('%d/%m/%Y %H:%M:%S.%f') #j/m/a
H:m:s.ms

```

- Le code ci-dessous permet d'avoir le PMID pour la fonction transfert 6.

```

# ----- PMID
    #'01b' formate tout les field en tant que string binaire à 0nb chiffres
    #print(Packet_Date)
    f14F=format(f14, '01b') #'01b" formate le field 14 en tant que string binaire à 1 chiffres
    f18F=format(f18, '05b') #'05b" formate le field 14 en tant que string binaire à 5 chiffres
    f28F=format(f28, '06b')
    f29F=format(f29, '06b')
    f30F=format(f30, '010b')

    #concaténation de tous les field formatés
    MID=f14F+f18F+f28F+f29F+f30F

    PMID=hex(int(MID, 2) )
    #int(MID, 2)' convertit le string binaire 'MID' en un integer(entier). Le deuxième argument, le 2, spécifie la base en tant que 2 pour la conversion binaire.

    # fonction de transfert
    for cle in PMID_ :
        if cle == str(PMID) :
            PMID_ft = PMID_.get(cle)
        if str(PMID) not in PMID_ :
            PMID_ft = '<undefined>'
            PMID_non_def+=1

    nbrTrame800 += 1

```

- Si le field_1 est égal à la valeur '0x806', il s'agit alors d'une trame UDP, et sera traité de manière analogue à la trame ARP, comme on peut le voir ci-dessous :

```
#SI 806 : -----
```

```

else :
    reste = framesize - 42 #trames inutiles
    data = fic.read(framesize-reste-14)#on lit la partie de la trame du field1
jusqu'à la fin
    f2, f3, f4, f5, f6, MACsender, senderIP, MACtarget, targetIP =
    struct.unpack('> HHBBH6sl6sl', data)
        # print('#####')
        # print(f1)
        # print('#####')

    # f2
    # print('f2 : ', f2)

    # f3
    # print('f3 : ',f3)

    # f4
    # print('f4 : ', f4)

    # f5
    # print('f5 : ', f5)

    # f6
    # print('f6 : ', f6)

# MACsender
MACsender = ':'.join('{:02x}'.format(octet) for octet in MACsender)
# --- fonction transfert
for cle in MAC :
    if cle == str(MACsender) :
        MACsender_ft = MAC.get(str(cle))
    if str(MACsender) not in MAC:
        MACsender_ft = '<undefined>'
    MAC_non_def += 1 #Une adresse mac n'est pas définie
    #print(MACsender_ft)
    #print(MACsender)

#senderIP
senderIP = '.'.join(str(b) for b in struct.unpack('BBBB', struct.pack('>I',
    senderIP)))
#fonction transfert
for cle in IP :
    if cle == str(senderIP) :
        senderIP_ft = IP.get(str(cle))
    if str(senderIP) not in IP:
        senderIP_ft = '<undefined>'
    IP_non_def += 1 #Une adresse IP n'est pas définie
    #print('senderIP :', senderIP_ft)

# MACtarget
MACtarget = ':'.join('{:02x}'.format(octet) for octet in MACtarget)
#fonction transfert
for cle in MAC :
    if cle == str(MACtarget) :

```


IV.d. fin du programme Python

- On sort de la boucle et le programme écrit ci-dessous sera commun aux deux types de trame rencontrés. La fin de ce programme consiste à lire le reste de la trame en fonction du reste calculé grâce à la taille dans l'entête de la trame, puis on incrémente la variable “nbrTrame”.

```
#COMMUN AU DEUX ----

data = fic.read(reste) #on lit le reste de la trame
data = fic.read(28) # on lit l'en tête à nouveau
nbrTrame=nbrTrame + 1 # on compte +1 trame

#print(nbrTrame) #affiche le nombre total de trames dans le fichier binaire
#print(nbrTrame800) #affiche le nombre de trames ARP
#print(nbrTrame806) #affiche le nombre de trames UDP
```

- Enfin cette dernière partie a pour but de prévenir l'utilisateur si l'une des correspondances n'a pas été renseignée dans le fichier "ft.json" qui contient l'ensemble des fonctions de transfert.

```
# VERIFICATION DE LA DEFINITION DES CHAMPS DANS LE FICHIER JSON
print("\nNombre de trames =",nbrTrame) # 5481

if PMID_non_def != 0 :
    print("Tous les MID ne sont pas définis, complétez le fichier ft.json si possible")
    print()
    if bench_5_non_def != 0 :
        print("Tous les champs bench_5 ne sont pas définis, complétez le fichier ft.json si possible")
        print()
    if MAC_non_def != 0 :
        print("Toutes les adresses MAC ne sont pas définies, complétez le fichier ft.json si possible")
        print()
    if IP_non_def != 0 :
        print("Tous les adresses IP ne sont pas définies, complétez le fichier ft.json si possible")
        print()
    if f17_non_def != 0 :
        print("Tous les champs f17 ne sont pas définis, complétez le fichier ft.json si possible")
        print()
    if f18_non_def != 0 :
        print("Tous les champs f18 ne sont pas définis, complétez le fichier ft.json si possible")
        print()
    if f28_non_def != 0 :
        print("Tous les champs f28 ne sont pas définis, complétez le fichier ft.json si possible")
        print()
```

```
if f29_non_def != 0 :  
    print("Tous les champs f29 ne sont pas définis, complétez le fichier ft.json si  
possible")  
    print()  
if f32_non_def != 0 :  
    print("Tous les champs f32 ne sont pas définis, complétez le fichier ft.json si  
possible")  
    print()
```

V. Plan de Validation / Rapports & Anomalies

Tous les tests doivent être effectués sur Linux.

L'Objectif principal de cette partie, est de pouvoir vérifier pour pouvoir affirmer l'exactitude et la fonctionnalité de nos exigences. Ces rapports de tests s'effectuent en plusieurs parties et concernent :

- Le programme Python
- Les Fonctions de transfert
- La Base de données
- La modification des labels
- Le Site Web

0. Général

SNIFFER _ETHERNET - General_002 :

D'après plusieurs site tel que python.developpez.com, wikipedia.org :
Python est un langage open source. Il possède une Licence open source CNRI, compatible GPL, mais sans la restriction *copyleft*.
Python est donc libre et gratuit, même pour les usages commerciaux.

I. Programme Python

LIRE TRAME 800 VERSION 1

Procédure du test			
Titre : LIRE TRAME 800 VERSION 1			Test ID : P1
Résumé : Le programme peut lire chaque trame du fichier binaire	Procédure ID : 101 Date : 26/03/23	Exécuté le : Début : 26/03/23 - 18h36 Fin : 26/03/23 - 18h39	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog_BinaireToAscii_101 (partiellement)	Rédigé par : THIEBOT Alex	Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	
Description:			
Une boucle est définie pour lire chaque trame et après les avoir traitées. Il y a ensuite l'ajout d'une variable "Nbrtrames" qui s'incrémente à chaque tour de boucle pour obtenir le nombre total de trame dans le fichier binaire..			
Conditions initiales :			
Tous les affichages provenant de la fonction print() doivent être commentés. Le fichier binaire «ethernet_result.data» contenant les trames 800 uniquement doit être dans le même fichier que le programme et mis en entrée (le fichier binaire correspondant se trouvant dans le fichier Vt-DEMO-power_on.rep)			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Décommenter la ligne qui permet de compter les trames		R.A.S.
II.	Décommenter la ligne qui permet d'afficher le nombre de trames		R.A.S.
III.	Lancer le programme	La variable "nbrTrame" doit avoir la valeur 5481.	R.A.S.

Exécution du test			
1		PASS	R.A.S.
2		PASS	R.A.S.
3		PASS	R.A.S.
Durée d'exécution : moins de 10 secondes			
Compte rendu :			
Le programme Python est capable de lire et afficher les données de chaque trame dont le field_1 est égal à '0x800'.			

LIRE TRAME 800 VERSION 2

Procédure du test			
Titre : Lire Trames 800 version 2			Test ID : P1
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le programme peut lire chaque trames du fichier binaire	102 Date : 26/03/23	Début : 06/04/23 - 14h05 Fin : 06/04/23 - 14h07	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog_BinaireToAscii_101 (partiellement)	Rédigé par : THIEBOT Alex	Exécuté par : ROUSSE Louane Vérifié et approuvé par : THIEBOT Alex	
Description: Une boucle est définie pour lire chaque trame et après les traiter. Ajout d'une variable "Nbrtrames" dans la boucle pour les compter.			
Conditions initiales : Tous les affichages (print) doivent être commentés. Le fichier binaire ethernet_result.data contenant les trames 800 uniquement doit être dans le même fichier que le programme et mis en entrée (le fichier binaire se trouve dans le fichier Vt-DEMO-power_on.rep). V2 du test avec le programme final.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Décommenter la ligne qui permet de compter les trames		R.A.S.
II.	Décommenter la ligne qui permet d'afficher le nombre de trames		R.A.S.
III.	Lancer le programme	"nbrTrames" doit avoir la valeur 5481.	R.A.S.
Exécution du test			

I.	Effectué ligne 308	PASS	R.A.S.
II.	Effectué ligne 311	PASS	R.A.S.
III.		PASS	R.A.S.

Durée d'exécution : moins de 10 secondes

Compte rendu :

Le programme est en capacité de lire toutes les trames du fichier binaire dans son intégralité et de nous restituer le nombre de trames. Ce test concerne la version finale de notre programme, une fois achevé.

LIRE TRAME 800 ET 806 VERSION 1

Procédure du test			
Titre : LIRE TRAME 800 ET 806 VERSION 1			Test ID : P2
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le programme peut lire chaque trame du fichier binaire	103 Date : 26/03/23	Début : 06/04/23 - 14h11 Fin : 06/04/23 - 14h12	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog_BinaireToAscii_101 SNIFFER_ETHERNET - Prog_BinaireToAscii_102	Rédigé par : THIEBOT Alex	Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	

Description :

Une boucle est définie pour lire chaque trame et après les traiter. Ajout d'une variable "nbrTrames" dans la boucle pour les compter.

Conditions Initiales :

Tous les affichages (print) doivent être commentés. Le fichier binaire contenant les trames 800 ET 806 doit être dans le même fichier ethernet_result.data que le programme et mis en entrée (le fichier binaire se trouve dans le fichier Vt-DEMO-mem_observability-ethernet.rep).

Étapes du test

N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Décommenter la ligne qui permet de compter les trames	Le programme doit être capable de compter le nombre de trames dans le fichier binaire.	R.A.S.
II.	Décommenter la ligne qui permet d'afficher le nombre de trames	Le programme nous affiche le nombre exacte de trames contenues dans le fichier	R.A.S.

		binaire.	
III.	Lancer le programme	“nbrTrames” doit avoir la valeur 5298 pour le test ‘Vt-DEMO-mem_observability-ethernet’	R.A.S.
Exécution du test			
1.	Effectué ligne 308	PASS	R.A.S.
2.	Effectué ligne 311	PASS	R.A.S.
3.		PASS	R.A.S.
Compte rendu : Le programme peut en effet lire chaque trame du fichier binaire.			

DIFFÉRENCIATION DES TRAMES 800 ET 806

Procédure du test			
Titre : Différenciation des trames 800 et 806			Test ID : P3
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le programme peut lire chaque trame du fichier binaire	104 Date : 06/04/23 - 13h23	Début : 12/04/23 - 13h23 Fin : 12/04/23 - 13h28	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog_BinaireToAscii_101 SNIFFER_ETHERNET - Prog_BinaireToAscii_102	Rédigé par : THIEBOT Alex	Exécuté par : THIEBOT Alex Vérifié et approuvé par : MARTIN Claire	
Description:			
Une boucle est définie pour lire chaque trame puis une boucle pour les trames 800 est définie et une autre pour les trames 806 pour traiter les deux types de trames différemment. Ajout d'une variable "nbrtrames800" dans la boucle des trames 800 et d'une variable "nbrtrames806" dans la boucle des trames 806 pour les compter.			
Conditions initiales :			
Tous les affichages (print) doivent être commentés. Le fichier binaire contenant les trames 800 ET 806 doit être dans le même fichier ethernet_result.data que le programme, et mis en entrée (le fichier binaire se trouve dans le fichier Vt-DEMO-mem_observability-ethernet.rep).			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Décommenter les lignes qui permettent de compter les trames	Le nombre de trames peut être compté	R.A.S.
II.	Décommenter les lignes X qui permet d'afficher le nombre de trames	Le nombre de trames peut-être affiché	R.A.S.

III.	Lancer le programme	Le programme Python est capable d'afficher le nombre exact de trames ARP et UDP comprises dans le fichier binaire brut.	R.A.S.
------	---------------------	---	--------

Exécution du test

1.		PASS	R.A.S.
2.		PASS	R.A.S.
3.		PASS	R.A.S.

Compte rendu :

Le programme compte les trames 800 et 806 et renvoie le résultat. Il peut différencier les trames 800 et 806.

RAPIDITÉ D'EXÉCUTION

Procédure du test			
Titre : RAPIDITÉ D'EXÉCUTION			Test ID : P4
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le programme doit s'exécuter en moins de 5 minutes.	105 Date : 18/05/23	Début : 23/05/23 - 16h12 Fin : 23/05/23 - 16h16	PASS
Exigence.s validée.s par le test : SNIFFER _ETHERNET - General_004 SNIFFER _ETHERNET - General_003	Rédigé par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description:			
Le temps d'exécution entre le lancement du programme et la fin de l'affichage des trames sur le site web doit être inférieur à 5 minutes.			
Conditions initiales :			
Le site web, le programme et les bases de données doivent être finalisés. Le tout doit être exécuté sur le raspberry.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Démarrer le programme avec en entrée le fichier binaire contenant les trames 800 et 806. Mettre en marche le chronomètre.	Le programme doit s'exécuter sans problème. Si des problèmes sont rencontrés, arrêter le test et les résoudre.	R.A.S.
II.	Arrêter le chronomètre lorsque les bases de données sont remplies, c'est-à-dire lorsque le programme python sera terminé.	Le temps doit être inférieur à 5 minutes pour que le test soit validé.	R.A.S.
Exécution du test			

1		PASS	R.A.S.
2		PASS	R.A.S.
Durée d'exécution : 1 minute 13			
Compte rendu : Notre programme entier a bien un temps d'exécution inférieur à cinq minutes sous linux sur le raspberry. Le test est validé.			

CONNEXION À LA BASE DE DONNÉE

Procédure du test v1			
Titre : Connexion du programme python à la base de données		Test ID : P5	
Résumé : Le programme Python doit se connecter à la base de données.	Procédure ID : 106 Date : 06/05/23	Exécuté le : Début : 06/05/23 - 10h23 Fin : 06/05/23 - 10h35 Exécuté par : MARTIN Claire	Résultat final : FAIL
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog - TrameVersBdD_107	Rédigé par : Claire MARTIN	Vérifié et approuvé par : ROUSSE Louane	
Description:			
Le programme Python doit être capable d'établir un lien avec la base de données "thales" située sur un serveur PhpMyAdmin, et doit ainsi être capable d'y ranger les données dans chacun des champs correspondants.			
Conditions initiales :			
Le programme Python est finalisé, la base de données est entièrement configurée et prête à être mise en service.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Importer le module 'MySQL connector' sur python	Il ne doit pas y avoir d'erreurs et Python doit être capable d'utiliser le module "MySQL connector" pour transférer les données requises dans la table de la base de données correspondante.	R.A.S.
II.	Exécuter le programme python sans erreurs	Vérifier que le programme est capable de transmettre des données dans la table de	R.A.S.

		la base de données	
Exécution du test			
1.		FAIL	
2.		FAIL	Une erreur de type 'ModuleNotFoundError' ou no module named 'mysql' est survenue, ce qui est une erreur indiquant que le module sql ne s'est pas installé ou accessible depuis Python.
Solution requise : Il faut pouvoir installer le module MySQL connector avec le module 'pip' inclus dans python3.			
Compte rendu : Notre code Python ne peut pas établir de lien avec notre base de données, dû à une mauvaise importation du module mysql connector. Il faut pouvoir l'installer avec le module 'pip' et la commande 'pip install' inclus dans Python3, qui est une commande utilisée pour installer des modules, des packages Python ainsi que de les mettre à jour.			

Procédure du test			
Titre : Connexion du programme python à la base de données v2			Test ID : P6
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le programme doit se connecter à la base de données.	107 Date : 06/05/23	Début : 18/05/23 - 14h43 Fin : 18/05/23 - 14h56	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog - TrameVersBdD_107	Rédigé par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description:			
Le programme Python doit être capable d'établir un lien avec la base de données "thales" située sur un serveur PhpMyAdmin, et doit ainsi être capable d'y ranger les données dans chacun des champs correspondants.			
Conditions initiales :			
Le programme Python est finalisé, la base de données est entièrement configurée et prête à être mise en service.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Importer le module 'MySQL connector' sur python	Il ne doit pas y avoir d'erreurs et Python doit être capable d'utiliser le module "MySQL connector" pour transférer les données requises dans la table de la base de données correspondante.	R.A.S.
II.	Exécuter le programme python sans erreurs	Vérifier que le programme est capable de transmettre des données dans la table de la base de données	R.A.S.
Exécution du test			

1.		PASS	R.A.S.
2.		PASS	R.A.S.

Durée d'exécution : 1 minute 20

Compte rendu :

Notre code Python peut être effectué sans erreurs tout en ayant importé le module MySQL connector et est capable d'insérer des données dans la base de données "thales" sur un serveur PhpMyAdmin, la connexion entre notre code Python et notre base de données peut donc avoir lieu.

FONCTIONS DE TRANSFERT

Procédure du test			
Titre : Fonctionnement du fichier “ft.json”			Test ID : P7
Résumé : Les correspondances décrites dans les fonctions transfert s'appliquent correctement.	Procédure ID : 108 Date : 25/05/23	Exécuté le : Début : 25/05/23 - 14h24 Fin : 25/05/23 - 14h26	Résultat final : FAIL
Exigence.s validée.s par le test : SNIFFER_ETHERNET-Prog-Fonction-Transfert_102 SNIFFER_ETHERNET-Prog-Fonction-Transfert_103	Rédigé par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description: Lors de l'exécution du programme, les valeurs désignées doivent être remplacées par celle correspondante dans la table de fonction de transferts, contenue dans un objet json dans un fichier “.json” et reconvertis en dictionnaire lors de leur passage dans le code Python.			
Conditions initiales : Le fichier texte contenant les fonctions transfert doit se trouver dans le même dossier que le programme. Le fichier texte doit être rempli selon les exigences désignées.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Compléter le fichier “ft.json”	Le fichier “ft.json” contenant l'intégralité des fonctions de transferts doit être rempli.	R.A.S.
II.	Exécuter le programme Python	Les données doivent être correctement associées	Erreur recensée
Exécution du test			
1		PASS	R.A.S.

2		FAIL	Le programme s'interrompt et affiche une erreur comme quoi il n'a pas trouvé la correspondance entre la fonction de transfert.
---	--	-------------	--

Compte rendu :

Le programme s'interrompt, une correspondance entre une des données et une fonction de transfert n'a pas pu être établie. Pour remédier à cela, il faudrait que si l'utilisateur a oublié par mégarde de notifier une correspondance dans le fichier "ft.json", que le programme continue de tourner et alerte l'utilisateur à la fin qu'il manque une ou plusieurs correspondances dans le fichier .json n'a pas pu être retrouvée, en indiquant de quelle fonction de transfert il s'agit.

Procédure du test			
Titre : Fonctionnement du fichier “ft.json”			Test ID : P7
Résumé : Les correspondances décrites dans les fonctions transfert s'appliquent correctement.	Procédure ID : 109 Date : 25/05/23	Exécuté le : Début : 27/05/23 - 9h45 Fin : 27/05/23 - 9h50	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET-Prog-Fonction-Transfert_102 SNIFFER_ETHERNET-Prog-Fonction-Transfert_103	Rédigé par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description: Lors de l'exécution du programme, les valeurs désignées doivent être remplacées par celle correspondante dans la table de fonction de transferts, contenue dans un objet json dans un fichier “.json” et reconvertis en dictionnaire lors de leur passage dans le code Python.			
Conditions initiales : Le fichier texte contenant les fonctions transfert doit se trouver dans le même dossier que le programme. Le fichier texte doit être rempli selon les exigences désignées. A été rajouté une gestion d'oubli. Si le client a oublié de renseigner une correspondance au sein de la fichier “ft.json”, à la fin de l'exécution, un message d'avertissement sera affichée aux yeux du client.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Compléter le fichier “ft.json”	Le fichier “ft.json” contenant l'intégralité des fonctions de transferts doit être rempli.	R.A.S.
II.	Exécuter le programme Python	Les données doivent être correctement associées	R.A.S.
Exécution du test			
1		PASS	R.A.S.
2		PASS	Le programme ne s'interrompt

			pas, et en cas d'oubli, il le notifie à son utilisateur.
--	--	--	--

Compte rendu :

Le programme ne s'interrompt plus et associe les fonctions de transfert aux données qui sont mutuellement présentes. En cas d'oubli, le client recevra un appel que son fichier "ft.json" contenant toutes les fonctions de transfert n'est pas complet. Ce sera ensuite à lui d'aller le compléter correctement pour ne plus recevoir de messages d'erreurs.

OUVERTURE AUTOMATIQUE DU SITE

Procédure du test			
Titre : Ouverture automatique du site dans un navigateur web			Test ID : P8
Résumé : Une fois le programme exécuté, le site web est automatiquement ouvert	Procédure ID : 110 Date : 01/06/23	Exécuté le : Début : 04/06/23 - 14h56 Fin : 04/06/23 - 14h59	Résultat final : <div style="background-color: red; color: white; padding: 5px; text-align: center;">FAIL</div>
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Web_314	Rédigée par : Louane Rousse	Exécuté par : Claire Martin Vérifié et approuvé par : Louane Rousse	
Description:			
A la fin de l'exécution du programme, le site web affichant les trames est automatiquement ouvert.			
Conditions initiales :			
Le site ne doit pas contenir de trames.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Exécuter le programme	Une fois exécutée, la page doit s'ouvrir toute seule.	La page ne s'ouvre pas, aucun code d'erreur n'est cependant affiché.
Exécution du test			
1		FAIL	Code d'erreur dit que l'url est bon, mais la page ne s'ouvre pas.

Durée d'exécution : 1 minute 27 secondes

Compte rendu :

Après différents essais, le résultat reste le même. La page ne s'ouvre pas.
Abandon de la fonctionnalité, le site web devra être ouvert manuellement.

GESTION DES ERREURS

Procédure du test			
Titre : Gestion des erreurs			Test ID : P9
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le site web doit signaler les erreurs si elles surviennent.	111 Date : 10/04/23	Début : 10/04/23 - 20h12 Fin : 10/04/23 - 20h16	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET-Prog -Fonction-Transfert_104	Rédigée par : MARTIN Claire	Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	
Description:			
Le site web doit afficher un message correspondant à une erreur pouvant impacter le test et donner un résultat erroné. Il vérifie que chaque field qui ont besoin d'une fonction transfert aient leur fonction transfert rempli.			
Conditions initiales :			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
1	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du PMID.	le message “Tous les MID ne sont pas définis, complétez le fichier ft..json si possible” devrait s'afficher.	R.A.S.
2	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du bench_5.	le message “Tous les champs bench_5 ne sont pas définis, complétez le fichier ft..json si possible” devrait s'afficher.	R.A.S.
3	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle des adresses MAC.	le message “Toutes les adresses MAC ne sont pas définis, complétez le fichier	R.A.S.

		ft..json si possible" devrait s'afficher.	
4	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle des adresses IP.	le message "Toutes les adresses IP ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.
5	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du field 17.	le message "Tous les champs f17 ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.
6	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du field 18.	le message "Tous les champs f18 ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.
7	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du field 28.	le message "Tous les champs f28 ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.
8	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du field 29.	le message "Tous les champs f29 ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.
9	vérifier que le fichier json contenant les fonctions transfert (ft.json) ne contienne pas celle du field 32.	le message "Tous les champs f32 ne sont pas définis, complétez le fichier ft..json si possible" devrait s'afficher.	R.A.S.

Exécution du test

1		PASS	R.A.S.
2		PASS	R.A.S.
3		PASS	R.A.S.
4		PASS	R.A.S.
5		PASS	R.A.S.
6		PASS	R.A.S.
7		PASS	R.A.S.
8		PASS	R.A.S.
9		PASS	R.A.S.

Durée d'exécution : 1 min 27, le temps que le programme python s'exécute.

Compte rendu :

Le site web affiche les erreurs.

II. Base de données

COUPLE DE CLÉS PRIMAIRES

Procédure du test			
Titre : Contrainte de clé			Test ID : BD1
Résumé : La base de donnée doit accepter que les tables TRAME800 et TRAME806 aient toutes deux des id commençant à 0	Procédure ID : 201 Date : 20/04/23	Exécuté le : Début : 20/04/23 - 10h00 Fin : 20/04/23 - 10h08	Résultat final : FAIL
Exigence.s validée.s par le test : SNIFFER_ETHERNET - BdD_201 SNIFFER_ETHERNET - BdD_203	Rédigée par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description: Il faut que le site puisse afficher l'emplacement de chaque trame dans le fichier binaire brut. Pour cela, nous utilisons le champ id des tables TRAME800 et TRAME806.			
Conditions initiales : La base de données contient entre autres deux tables : TRAME800 et TRAME806. Les deux possèdent un champ 'id' qui est noté comme leur clé primaire. Par ailleurs, la table TRAME800 est déjà remplie grâce à un premier test effectué.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Exécution du programme	Pas de message d'erreur	Code d'erreur
II.			

III.			
Exécution du test			
1		FAIL	La contrainte d'intégrité de clé primaire

Compte rendu :
Nous obtenons une erreur d'intégrité sur la primaire : lors de l'ajout des trames 806 dans la base de donnée, l'indice du premier tuple est le même que celui dans la table TRAME800 (c'est à dire 0), un code d'erreur est généré.

Procédure du test			
Titre : Contrainte de clé			Test ID : BD1
Résumé : La base de donnée doit accepter que les tables TRAME800 et TRAME806 aient toutes deux des id commençant à 0	Procédure ID : 202 Date : 20/04/23	Exécuté le : Début : 22/04/23 - 13h20 Fin : 22/04/23 - 13h25	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - BdD_201 SNIFFER_ETHERNET - BdD_203	Rédigée par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	
Description: Il faut que le site puisse afficher l'emplacement de chaque trame dans le fichier binaire brut. Pour cela, nous utilisons le champ id des tables TRAME800 et TRAME806.			
Conditions initiales : La base de données contient entre autres deux tables : TRAME800 et TRAME806. Les deux possèdent un champ 'id' qui est noté comme leur clé primaire. Elles possèdent également un champ 'nomdetest'. Par ailleurs, la table TRAME800 est déjà remplie grâce à un premier test effectué.			

Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Changement de la structure des deux tables	Le couple (id,nomdetest) devient clé primaire dans chacune des tables	R.A.S.
II.	Exécution du programme	Aucune erreur lors de l'exécution	R.A.S.
III.			
Exécution du test			
1		PASS	R.A.S.
2		PASS	R.A.S.
3		PASS	R.A.S.
Compte rendu : En ayant choisi d'ajouter le nom du test en clé primaire au même titre que le champ id, deux tests différents gérant deux types de trames pourront être affichés sur le site La table accepte maintenant l'ajout de plusieurs types de trames.			

VALIDATION DES 3 FORMES NORMALES

→ Voir la rubrique “Base De Données” page 15 pour la validation des trois formes normales.

III. Site web

TEST DES DIFFÉRENTES INTERFACES POSSIBLES

Procédure du test			
Titre : Test des différentes interfaces possibles			Test ID : SW1
Résumé : Le site peut s'adapter à différentes interfaces.	Procédure ID : 301 Date : 08/05/23 Rédigée par : THIEBOT Alex	Exécuté le : Début : 27/05/23 - 15h49 Fin : 27/05/23 - 15h53	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Web_301		Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	
Description: L'affichage du site web doit s'adapter en fonction des différents appareils potentiellement utilisés. Le tableau doit toujours être visible clairement et les fonctionnalités doivent toujours fonctionner.			
Conditions initiales : Le site web doit être opérationnel sur l'écran test. Le test est validé si toutes les étapes sont validées.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Il faut se rendre dans l'onglet Outils de développement sur un navigateur internet depuis la page du site web à tester. garder le même navigateur pour toutes		R.A.S.

	les étapes.		
II.	Tester avec un ordinateur portable		R.A.S.
III.	un petit écran		
IV.	un grand écran		R.A.S.
Exécution du test			
1		PASS	Test effectué sur google chrome.
2		PASS	R.A.S.
3		FAIL	Problème avec le petit écran sur selection_test.php les boutons se superposent.
4		PASS	R.A.S.
Solution requise : Il faut changer la "width" de la <div class="pourladiv> de 50% à 70%			
Compte rendu :			
Le site web s'adapte aux interfaces d'un grand écran et d'un ordinateur portable. Mais le petit écran présentait un problème, on a donc changé la largeur de la boîte contenant les 2 boutons "Nom du test et date" et "supprimer un test" de 50% à 70% afin de résoudre le problème d'adaptation. → Le problème a été résolu, le site web s'adapte aux différentes interfaces.			

ADAPTABILITÉ DU SITE WEB

Procédure du test			
Titre : Test différents navigateurs internet			Test ID : SW2
Résumé :	Procédure ID :	Exécuté le :	Résultat final :
Le site peut s'adapter à différents navigateurs internet.	302 Date : 08/05/23	Début : 27/05/23 - 16h02 Fin : 27/05/23 - 16h05	PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Web_313	Rédigée par : THIEBOT Alex	Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	

Description:

Les fonctionnalités et l'affichage du site web doivent s'adapter en fonction des différents navigateurs de recherche utilisés. Le tableau doit toujours être visible clairement et les fonctionnalités doivent toujours fonctionner.

Conditions initiales :

Le site web doit être opérationnel au niveau de l'affichage et des fonctionnalités sur l'écran test.

Test validé si toutes les étapes sont validées.

Étapes du test

N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Ouvrir le site sur Google Chrome et s'assurer que toutes les fonctionnalités fonctionnent + que l'interface est lisible.		R.A.S.
II.	Ouvrir le site sur Firefox et s'assurer que toutes les fonctionnalités fonctionnent + que l'interface est lisible.		R.A.S.
III.	Ouvrir le site sur Bing et s'assurer que toutes les fonctionnalités fonctionnent +		R.A.S.

	que l'interface est lisible.		
IV.	Ouvrir le site sur Internet Explorer et s'assurer que toutes les fonctionnalités fonctionnent + que l'interface est lisible.		R.A.S.
V.	Ouvrir le site sur Opera et s'assurer que toutes les fonctionnalités fonctionnent + que l'interface est lisible.		R.A.S.

Exécution du test

1		PASS	
2		PASS	Certaines fonctionnalités sur la scrollbar marchent partiellement (taille et couleur).
3		PASS	
4		FAIL	Certaines fonctionnalités HTML/CSS ne marchent pas(taille, couleur,position).
5		PASS	R.A.S.

Compte rendu :

Presque tous les navigateurs web sont capables d'afficher un rendu lisible des différentes pages du site web. Sauf Internet Explorer qui présente des problèmes au niveau de la lisibilité de certains textes. Le test est validé car il marche sur la majorité des navigateurs web.

VALIDATION W3C

Procédure du test			
Titre : Validation W3C			Test ID : SW3
Résumé : Le code du site web doit être validé par le W3C.	Procédure ID : 303 Date : 08/05/23	Exécuté le : Début : 01/06/23 - 14h34 Fin : 01/06/23 - 14h43	Résultat final :
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Prog_102	Rédigée par : THIEBOT Alex	Exécuté par : MARTIN Claire Vérifié et approuvé par : ROUSSE Louane	PASS
Description: Tester et repérer les erreurs du code du site web via le site du World Wide Web Consortium. Il ne doit rester aucune erreur.			
Conditions initiales : Le site web doit être à sa version finale.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Pour chaque page du site web, vérifier leur code HTML sur le site du W3C (https://validator.w3.org/).	Le HTML ne doit présenter aucune erreur (en rouge).	Les remarques (en orange) peuvent être laissé si elle n'empêchent pas le fonctionnement du site web
II.	Pour chaque page du site web, vérifier leur code CSS sur le site du W3C (https://jigsaw.w3.org/css-validator/).	Le CSS ne doit présenter aucune erreur (en rouge).	
Exécution du test			
1		PASS	R.A.S.

2		PASS	R.A.S.
3			
Durée d'exécution :			
Compte rendu : Les pages du site web respectent la W3C, ainsi que le CSS.			

CHANGEMENT DES LABELS

Procédure du test			
Titre : Changement des labels			Test ID : SW4
Résumé :	Procédure ID : 304 Date : 12/04/23	Exécuté le : Début : 12/04/23 - 9h47 Fin : 12/04/23 - 9h51	Résultat final : PASS
Exigence.s validée.s par le test :	Rédigée par : ROUSSE Louane	Exécuté par : THIEBOT Alex Vérifié et approuvé par : ROUSSE Louane	
Description: Le site web ouvre un fichier texte "Labels.txt" contenant un nom de champ par ligne. Ces noms correspondent aux labels situés sur les colonnes sur le site web.			
Conditions initiales : Le site web doit afficher un tableau avec le nom des champs sur la première ligne. Un fichier texte doit être rempli avec les noms des champs correspondants.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Ouvrir le site web sur un navigateur internet.	Affichage lisible. Si illisible, arrêter le test et corriger le problème.	R.A.S.
II.	Ouvrir le fichier texte contenant le nom des fields.	Le fichier doit être rempli. Si ce n'est pas le cas, le remplir en suivant le schéma du site.	R.A.S.
III.	Noter le nom du 1er field et le modifier depuis le fichier texte. enregistrer les modifications du fichier texte et réactualiser la page.	Le nom du field modifié dans le fichier texte doit avoir changé et être le même sur le site web. Si ce n'est pas le cas, test non	R.A.S.

		validé.	
Exécution du test			
1	Le site internet est lisible.	PASS	R.A.S.
2	Le fichier texte a été rempli pour ce test.	PASS	R.A.S.
3	Dès lors du changement d'un nom de colonne dans le fichier texte, la modification s'applique instantanément après rafraîchissement de page.	PASS	R.A.S.
Durée d'exécution :			
Compte rendu : Le nom des champs peut être changé grâce à un fichier texte "Labels.txt".			

SUPPRIMER LES TRAMES

Procédure du test			
Titre : Supprimer trame			Test ID : SW5
Résumé : Le site web a la possibilité de supprimer une trame.	Procédure ID : 305 Date : 18/05/23	Exécuté le : Début : 20/05/23 - 15h34 Fin : 20/05/23 - 15h38	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Web_312	Rédigée par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : THIEBOT Alex	
Description:			
Le site web permet de supprimer une trame si l'utilisateur le souhaite.			
Conditions initiales :			
Le site web doit être ouvert et déjà afficher au moins une trame.			
Étapes du test			
N°	Étape	Description Pass/Fail Criteria	Résultat attendu
I.		Cliquer sur le bouton intitulé "Supprimer une trame"	Le bouton lié à la requête de suppression d'une trame dans la base de donnée est fonctionnel
II.		Vérifier que le site web n'affiche plus la trame une fois supprimée	Le site WEB n'affiche plus la trame dans la sélection une fois le nom de test associé est supprimé
Exécution du test			
1.			PASS
			R.A.S.

2.		PASS	Afin d'obtenir le résultat attendu, il faut rafraîchir la page deux fois.
----	--	------	---

Durée d'exécution : Moins d'une seconde

Compte rendu :

Le site permet la suppression d'une trame , il suffit de rafraîchir la page pour voir le résultat. Le test est validé.

AFFICHER UNE TRAME EN FONCTION DE SON NOM DE TEST ET DE SA DATE

Procédure du test			
Titre : afficher trame fonctionnalités			Test ID : SW6
Résumé :	Procédure ID : 306 Date : 18/05/23	Exécuté le : Début : 20/05/23 - 15h40 Fin : 20/05/23 - 15h42	Résultat final : PASS
Exigence.s validée.s par le test :		Exécuté par : MARTIN Claire Rédigée par : MARTIN Claire	Vérifié et approuvé par : THIEBOT Alex
Description:			
Le site sera en capacité d'afficher des trames selon deux critères : sa date d'exécution et son nom. En effet, un même test peut être effectué à deux dates différentes : le client doit être en mesure de pouvoir choisir de visualiser les trames à une date précise.			
Conditions initiales :			
Le site web doit être ouvert et contenir au moins une trame.			
Étapes du test			
N°Étape	Description Pass/Fail Criteria	Résultat attendu	Remarques
I.	Sélectionner un nom de test		
II.	Sélectionner une date		
III.	appuyer sur "exécuter" ?	seulement les trames correspondantes doivent s'afficher.	
Exécution du test			
1		PASS	R.A.S.

2		PASS	R.A.S.
3		PASS	R.A.S.
Durée d'exécution : Moins d'une seconde après avoir appuyé sur le bouton			
Compte rendu : Le site permet d'afficher une trame selon deux critères : sa date d'exécution et son nom. Aucune erreur n'a été détectée lors des tests.			

AFFICHER UN NOMBRE DE TRAME PAR PAGE AU CHOIX (10,50,100)

Procédure du test			
Titre : choisir nombre trame affichées			Test ID : SW7
Résumé : Le site web permet de choisir combien de trames sont affichées.	Procédure ID : 307 Date : 18/05/23	Exécuté le : Début : 20/05/23 - 15h43 Fin : 20/05/23 - 15h45	Résultat final : PASS
Exigence.s validée.s par le test : SNIFFER_ETHERNET - Web_309	Rédigé par : MARTIN Claire	Exécuté par : MARTIN Claire Vérifié et approuvé par : THIEBOT Alex	
Description: L'utilisateur peut sélectionner le nombre de trames qu'il veut voir afficher par page, entre 10, 50 et 100.			
Conditions initiales : Le site web doit être ouvert et une trame déjà présente.			
Étapes du test			
N°Étape	Description	Résultat attendu	Remarques
1	Noter combien de trames sont affichées automatiquement.	10 trames doivent être affichées.	si ce n'est pas le cas, continuer le test. Il ne sera pas entièrement validé.
2	Sélectionner "afficher 10 trames" et exécuter.	Le site doit afficher 10 trames.	
3	Sélectionner "afficher 20 trames" et exécuter.	Le site doit afficher 20 trames.	
4	Sélectionner "afficher 50 trames" et exécuter.	Le site doit afficher 50 trames.	
5	Sélectionner "afficher 100 trames" et exécuter.	Le site doit afficher 100 trames.	
Exécution du test			

1		PASS	R.A.S.
2		PASS	R.A.S.
3		PASS	R.A.S.
4		PASS	R.A.S.

Durée d'exécution : Moins d'une seconde

Compte rendu :

Nous pouvons choisir parmi les options des <select>, l'affichage de 10,20,50,100 trames par pages.

V. Gestion de projet

Cette partie vous renseignera sur la manière dont nous nous sommes organisés pour aller au bout de ce projet. Vous pourrez visualiser les différentes tâches que nous avions définies et affinées au fur et à mesure de notre avancement, la définition des rôles de chacun, ainsi que les différents risques identifiés et les solutions apportées.

1. RACI

Ainsi nous avons mis en place un RACI qui est un tableau ayant pour but de se mettre en accord sur les différentes missions assignées à chacun de nous. Étant un petit groupe, il était important pour nous de nous tenir mutuellement informés de toutes nos avancées et de vérifier le travail de chacun afin de s'entraider.

		ROUSSE Louane	THIEBOT Alex	MARTIN Claire
Réécriture des exigences	X			
Planification des tâches	X	R	A I	I
Définition des risques	X	I	R	A I
Structuration du code Python	X	R	I	A I
Application des filtres Python et récupération des données	X	R	A I	C I
Conception et schématisation de la base de données	X	R	A I	C I
Installation d'un serveur de bases de données	X	I	A I	R
Création des tables et de ses caractéristiques	X	A I	I	R
Ranger les données dans les tables correspondantes	X	A I	I	R
Écriture du programme gérant les fonctions de transfert	X	C I	A I	R
Écriture du programme de récupération des données du fichier de configuration	X	I	R	A I
Écriture du programme pour modifier le nom des labels	X	R	A I	I
Installation et configuration d'un serveur web	X	I	A I	R
Concevoir une interface web pour visualiser les données	X	I	R	A I
Rendre le site dynamique	X	I	R	C I
Écriture du rapport	X			
Réalisation d'un support de présentation orale				

Légende :

	En groupe
X	Fait
R	Effectue la tâche
A	Vérifie la tâche
C	Aide R
I	Est tenu informé

2. Diagramme de GANTT

Ce schéma est représentatif de l'évolution de notre organisation des tâches au cours de ce semestre.

Comme indiqué dans la légende, les différentes nuances de bleu indiquent les ajustements du planning au cours du projet.

DIAGRAMME DE GANTT

TITRE DU PROJET			SAE - 24					NOM DU GROUPE		Rababa																	
CHEF DE PROJET			Louane Rousse					DATE		09/06/23																	
TITRE DE LA TÂCHE	PROPRIÉTAIRE DE LA TÂCHE	DATE DE DÉBUT	DATE LIMITE	DURÉE EN JOURS	TÂCHE TERMINÉE (EN %)	JANVIER		FEVRIER		MARS				AVRIL			MAI			JUIN							
						4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Réécriture des exigences	Groupe	23/01/23	12/02/23	19	100 %																						
Planification des tâches	L	23/01/23	12/02/23	19	100 %																						
Définition des risques (précisément)	A	23/01/23	12/02/23	19	100 %																						
Structuration du code Python	L	30/01/23	07/04/23	67	100 %																						
Réécupération des données et application des filtres Python	L	06/02/23	14/04/23	68	100 %																						
Conception et schématisation de la base de données	L	06/02/23	12/02/23	6	100 %																						
Création des tables et ses caractéristiques	C	03/04/23	01/05/23	28	100 %																						
Ranger les données dans les tables correspondantes	C	15/04/23	21/05/23	36	100 %																						
Installation d'un serveur de base de données	C	15/04/23	21/05/23	36	100 %																						
Ecriture du programme gérant les fonctions transferts	C	15/04/23	28/05/23	43	100 %																						
Ecriture du programme de récupération des données du fichier de configuration	A	06/02/23	26/02/23	20	100 %																						
Ecriture du programme pour modifier le nom des labels	L	12/04/23	19/04/23	7	100 %																						
Installation et configuration d'un serveur web	C	08/05/23	03/06/23	25	100 %																						
Conception de l'interface web (visualisation des données)	A	05/06/23	21/06/23	16	100 %																						
Dynamisation du site	A	15/05/23	03/06/23	18	100 %																						
Ecriture du rapport	Groupe	23/01/23	09/06/23	136	100 %																						
Réalisation d'un support de présentation orale	Groupe	05/06/23	21/06/23	16	10 %																						

Légende :

Temps initialement prévu	
Rajout de temps n°1	
Rajout de temps n°2	

3. Définition et gestion des risques

Après réflexion des risques que nous pourrions rencontrer au cours de ce second semestre, nous les avons mesurés et classés dans une matrice des risques, du risque le plus mineur au plus majeur.

	Risques	Importance	Solution	Importance du risque après solution
Techniques	Vitesse d'exécution du raspberry Pi	2	Cela ne relève pas de nos compétences	2
	Difficulté en programmation	2	Pratiquer et s'aider de ressources en lignes	1
	Difficulté à traiter les fichiers	2	Se documenter	1
	Manque de méthodologie	2	Planifier les objectifs et les tâches de chacun, organiser le code	1
	Gestion des bases de données	3	Se documenter	1
	Dysfonctionnement ou perte du Raspberry Pi	3	Garder une copie du projet sur un ordinateur et/ou un disque dur	1
Humains	Mauvaise estimation du temps	1	Travailler régulièrement	1
	Procrastination	1	Alterner travail productif et repos	1
	Maladie	1		1
	Mauvaise gestion du temps	2	Planifier les tâches à l'avance	1
	Avis divergents sur la réalisation du projet	2	Communiquer	1
	Planning surchargé	3	S'organiser et se fier au diagramme de Gantt	2
	Pas de motivation	3	Commencer une tâche, même simple	1
	Hors-sujet	3	Poser des questions au responsable du projet	1
	Abandon de la part d'un membre du groupe	3	Réorganisation entre les membres restants	2

La légende de ce diagramme est la suivante :

	Risque mineur
	Risque modéré
	Risque important

Matrice des risques :

Nous avons classé les risques définis ci-dessous dans la matrice des risques suivante, les classant par importance et par probabilité.

Risques humains		Sévérité			
		Léger	Mineur	Modéré	Majeur
Probabilité	Certain		Procrastination		Planning surcharge
	Probable	Mauvaise estimation du temps	Maladie	Mauvaise gestion du temps	
	Possible				Pas de motivation
	Peu probable			Avis divergents sur la réalisation du projet	Hors-sujet, Abandon de la part d'un membre du groupe

Risques techniques		Sévérité		
		Léger	Mineur	Majeur
Probabilité	Certain	Vitesse d'exécution du raspberry Pi		
	Probable		Difficulté en programmation	Gestion des bases de données
	Peu probable	Modifier le nom des colonnes sur le site web	Manque de méthodologie, Traiter les fichiers	Dysfonctionnement, perte du Raspberry Pi

4. Retour d'expérience

Claire

Réaliser ce projet fut pour moi très enrichissant et intéressant, malgré le fait que nous étions un petit groupe avec au départ des connaissances peu avancées sur les différents sujets et thèmes auquel a touché cette SAÉ (tels que les différents langages de programmation, les bases de données, le langage php). Cependant, nous avons pu aboutir à un projet fonctionnel même si loin d'être parfait, et je suis satisfaite du travail que nous avons accompli et de notre évolution au cours de ce projet. Notamment, je trouve que la gestion du temps a été un challenge en plus pour nous, il a fallu apprendre, se renseigner, faire un très grand nombre d'essais, d'ailleurs si c'était à refaire et pour la prochaine fois, je prendrai davantage de notes à chaque essai effectué pour pouvoir fournir plus de tests et de meilleure qualité. C'était pour moi l'un de mes premiers projets importants prenant en compte toutes les procédures d'un projet réel et professionnel (définition des exigences, gestion de projet, conception d'algorithme etc.) j'apprends de mes erreurs et j'en tire des leçons pour les futurs projets à venir, mais je suis globalement contente de ce que nous avons effectué durant cette S.A.É. Et pour finir, merci à mes camarades Louane et Alex pour leur contribution et leur implication !

Louane

Malgré les difficultés que nous avons pu rencontrer, je suis fière de ce que nous avons accompli. Ce projet aura été une expérience intéressante et formatrice, et m'aura prouvé que réaliser de grande chose est possible avec une bonne organisation et de la persévérance. Il m'aura aussi permis de mettre en application toutes les ressources apprises en classe et d'en comprendre leur réel usage, ce que je trouve très important et intéressant.

Pour finir, j'aimerai remercier Claire et Alex pour leur implication dans le projet !

Alex

Réaliser ce projet de sniffer Ethernet a été pour moi une expérience très enrichissante et intéressante (quoique un peu longue). C'est le premiers projets importants prenant en compte toutes les procédures d'un projet réel et professionnel, comme la définition des exigences, la gestion du projet et la conception d'algorithmes. J'ai pu mettre en pratique les connaissances acquises en classe, notamment sur les langages de programmation, le HTML/CSS ...Je suis content du travail accompli et de l'évolution au cours de ce projet. Cette expérience m'a permis de comprendre l'importance du travail continue et de l'organisation dans la réalisation de projets complexes. En conclusion, je suis fier de ce que nous avons réalisé lors de ce projet de sniffer Ethernet. J'ai pu acquérir de nouvelles compétences techniques. Les erreurs commises durant ce projet vont me servir de leçons pour mes futurs projets. Pour finir, j'aimerai remercier Claire et Louane pour leur implication dans le projet !

VI. Conclusion

Pour conclure ce projet, nous aimerais remercier Madame escazut, Monsieur Gautero et Monsieur Cam pour nous avoir aidé tout au long de ce grand projet. Cette SAÉ fut une expérience très enrichissante qui nous a permis d'exploiter toutes nos compétences, mais aussi d'apprendre des autres.

Nous sommes fiers d'avoir concrétisé notre travail réalisé au premier semestre en vous présentant un programme répondant à notre problématique et aux besoins du client.

En effet, au cours de ce second semestre nous avons mis en œuvre les exigences et algorithmes conçus lors du premier semestre en ajoutant notamment des exigences pour affiner la définition du projet.

De ce fait, nous avons pu concevoir un programme Python qui a pu lire, traiter et extraire les données d'un fichier en binaire brut, ainsi que les données de son fichier texte de configuration.

Nous avons stocké ensuite ces mêmes données dans une base de données sur un serveur local et les avons triées pour pouvoir différencier les différents types de trames, et afin de permettre au site WEB d'afficher plus efficacement les trames d'un test sélectionné par l'utilisateur afin qu'il puisse procéder à l'analyse et l'interprétation de ces résultats.

C'est ainsi que s'achève ce projet qui nous aura accompagné tout au long de cette année, et nous remercions encore une fois les enseignants et l'entreprise Thalès Alenia Space pour cette opportunité.

MARTIN Claire
ROUSSE Louane
THIEBOT Alex