INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

centre de recherche SOPHIA ANTIPOLIS – MÉDITERRANÉE

Détection d'intrusions : Extraction et Utilisation de Connaissances

Réunion ARC Sésur

Collaboration :

LIRMM – LGI2P/EMA – INRIA Sophia

Plan

Introduction

Problématique – Etat de l'art

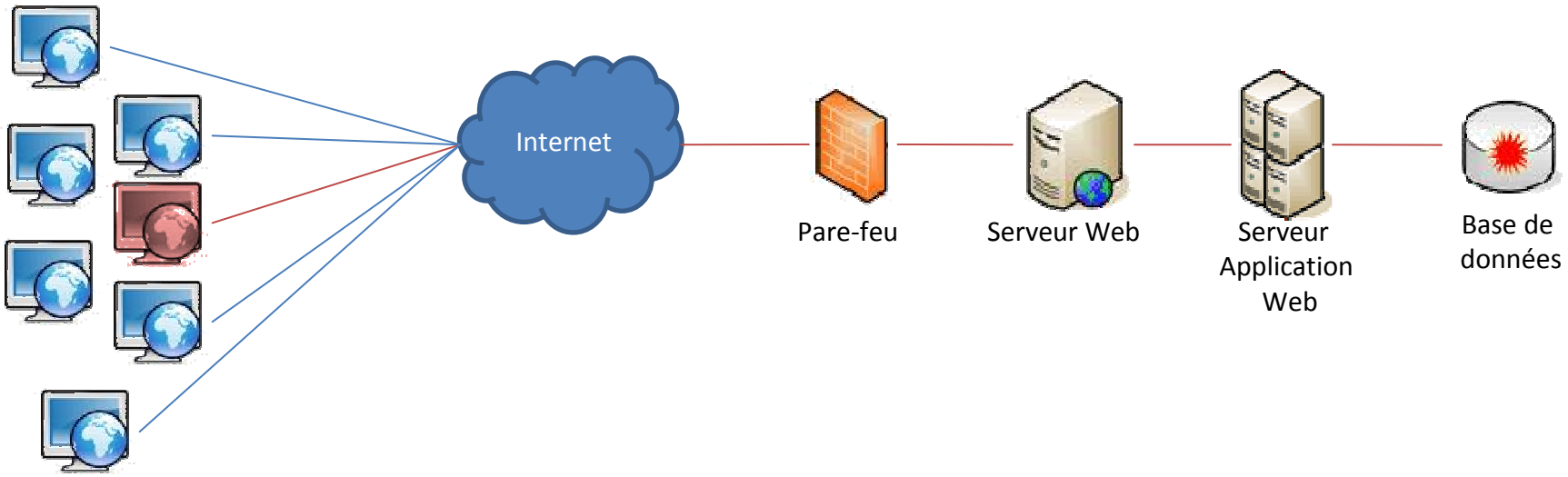
Apprentissage sur les non attaques – base de signatures statistiques

Apprentissage sur les attaques – approche par mot et par descripteurs

Conclusion

Introduction

- Dommages (yahoo, google, …)
- Permanent, évolution (pays, Pentagone)
- Diversité des attaques



Introduction

Exemple : sql injection

URL :

http://webmail.mailSite.fr/
cgi-bin/getInfor?
leemail=valemail&motdepasse=password

Select email, motdepasse, nom, prenom
From Clients
Where email = *valemail* **and** motdepasse = password ;

Valemail = victime@mailSite.fr
Motdepasse = 'x' **or** '1' = '1'

Introduction

- Un exemple d'URL :

- La requête associée :

Select email, mdp, nom, prenom
From Clients
Where lemail = *valemail* **and** mdp = *password* ;

- L'attaque (SQL INJECTION) :

valemail = victime@mailSite.fr
passwd = 'x' **or** '1' = '1'

Introduction

http://www.google.fr/search?hl=fr&rlz=1T4GFRC_frFR207FR208&q=.cgi&meta=
http://images.google.fr/images?hl=fr&rlz=1T4GFRC_frFR207FR208&q=.cgi&ie=UTF-8&oe=UTF-8&um=1&sa=N&tab=wi
http://groups.google.fr/groups?hl=fr&rlz=1T4GFRC_frFR207FR208&q=.cgi&ie=UTF-8&oe=UTF-8&um=1&sa=N&tab=ig
http://news.google.fr/news?hl=fr&rlz=1T4GFRC_frFR207FR208&q=.cgi&ie=UTF-8&oe=UTF-8&um=1&sa=N&tab=gn
http://www.google.fr/search?hl=fr&q=sdi&ie=UTF-8&oe=UTF-8&um=1&sa=N&tab=iw
http://maps.google.fr/maps
http://www.google.fr/search?hl=fr&q=maps&meta=
http://www.google.fr/search?hl=fr&q=scholar&meta=
http://scholar.google.com/
http://scholar.google.com/scholar?q=intrusion+detection+data+mining&hl=fr&lr=&lr=
http://scholar.google.com/scholar?hl=fr&lr=&cluster=13562338393205821716
http://groups.google.fr/groups/dir?lnk=od&hl=fr&sel=16823695
http://groups.google.fr/groups/dir?hl=fr&sel=16823695,67188904,16823684,16823694,16823683
http://groups.google.fr/group/soswindows?lnk=gschg&hl=fr
http://news.google.fr/?ned=fr&topic=n
http://video.google.fr/videosearch?q=genre:../../../etc/passwd
http://earth.google.fr/earth4.html
http://books.google.fr/books?id=7NA0UD5wbI4C&pg=PA1&ots=FO9STQKEzv&dq=ids&sig=p3BmwwqjEMozRgoNdXsfj6tjn4M
http://blogsearch.google.fr/blogsearch?hl=fr&q=ids&lr=
http://directory.google.com/Top/World/Fran%C3%A7ais/Informatique/
http://code.google.com/projects.html
http://fr.search.yahoo.com/search?p=ids&fr=yfp-t-501&ei=UTF-8&meta=vc%3D
http://fr.news.search.yahoo.com/search/news?p=ids&fr=yfp-t-501&ei=UTF-8&meta=vc%3D
http://ee.my.yahoo.com/config/my_init?.intl=fr&.partner=my&.from=i
http://fr.maps.yahoo.com/broadband#mvt=m&trf=0&lon=3.887701&lat=43.610042&mag=5
http://fr.movies.yahoo.com/anecdotes-films/index.html?b=60
http://shopping.yahoo.fr/b/a/sbs/124901/17925290.html?keyword1=besace
http://fr.search.cars.yahoo.com/bin/search/photos_gallery_fr/cars/?m=y&p=cat:photos_cars+Alpina
http://fr.groups.yahoo.com/group/forum_apm/
http://yahoo.fr.immostreet.com/recherche.htm?euro=1&idtt=1&idtypebien=1&lang=fr&nb_pieces=2&pays=fr&photo=1&cp=34000&ci=&
http://fr.astrology.yahoo.com/weekly/20070702/coq.html
http://fr.mobile.yahoo.com/go;_ylt=AsKpwyG2DzclQYZYEyB9EkbntAcJ
http://fr.voyage.yahoo.com/p-page_d_accueil-595670-action-reservation-reserver-location
http://fr.videogames.games.yahoo.com/r/psp/fighting/dragon_ball_z_shin_budokai_2/6b4ab0.html
http://fr.news.yahoo.com/pcinpact/20070703/ttc-perian-1-0-le-couteau-suisse-pour-qu-c2f7783_2.html
http://fr.sports.yahoo.com/02072007/70/gp-de-france-hamilton-pas-inquietant.html
https://login.yahoo.com/config/login_verify2?.slogin=select*fromclientwhereclient=name&.intl=fr&.src=&.bypass=&.partner=&.done=http%3a//edit.yahoo.com/config/eval_profile%3f.done=http%3a//fr.mail.yahoo.com/%26.scrumb=0&pkg=&owd=
http://edit.yahoo.com/config/edit_account?.child=&.src=&.scrumb=u591QPtgl3.&.done=http%3a//edit.yahoo.com/config/eval_profile%3f.done=http%3a//fr.mail.yahoo.com/%26.scrumb=u591QPtgl3.

Problématique

Définition
(chaine de caractères bien formée) Soit c une chaine de caractères. C est dite bien formée si elle ne possède pas de caractères interdits par la norme RFC 1738

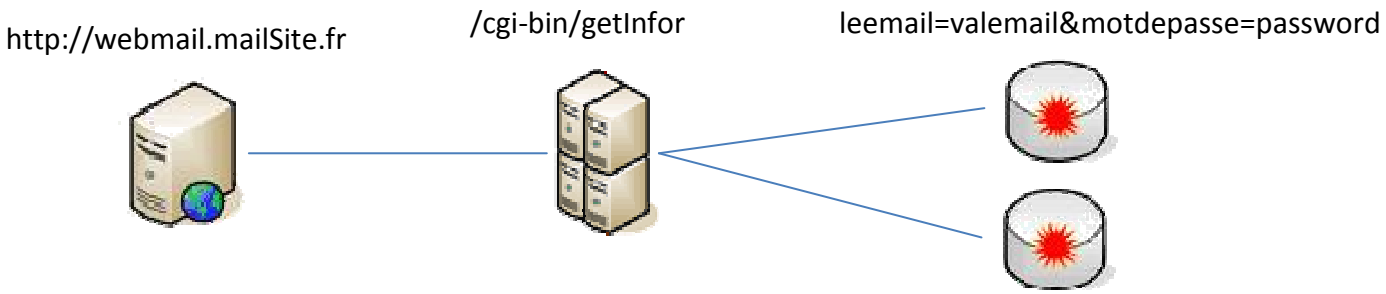
Définition
(URL) Soit UR une chaine de caractères correspondant à un domaine. Soit URI une chaine de caractères correspondant à un chemin. URL = < UR, URI, {p}>

Définition
(paramètres) Soit PARAM un ensemble de paramètres tel que p ∈ PARAM est un couple p = (variable, valeur)

Problématique

Exemple

URL = < (http://webmail.mailSite.fr), (/cgi-bin/getInfor),
{(leemail, valemail), (motdepasse, password)} >



Problématique

Définition
(base de données d'URL) Soit DB une base de données où chaque entrée est définie par le tuple :

$$e = < id, URL >$$

Définition
(la fonction attack, DB+, DB-) Soit attack(e) une fonction qui retourne true si l'entrée e ∈ DB est une attaque et false autrement.

$$\forall e \in DB+, attack(e) = false$$
$$\forall e \in DB-, attack(e) = true$$

Problématique

Problématique
La problématique consiste à la caractérisation de la fonction attack.

Contraintes, difficultés
1. Volume important des données à traiter
2. Diversité des url
3. Pas de formes standards pour les attaques
4. Caractérisation des url
5. Faux négatifs et positifs
6. Sans expertise
7. Données disponibles sous la forme de flots de données

Etat de l'art

Approche Comportementale (détection d'anomalies)

- Apprentissage sur des données normales
- Recherche de déviations
- ADAM, PHAD, NIDES, MINDS

Approche par signature (détection d'abus)

- Apprentissage sur des données labélisées attaques et non attaques
- Distinguer les attaques
- ID3, IDS-ANN, IDES, ISOA

Contributions

Comment caractériser une attaque ?
Deux approches : signature de non attaques – signatures d'attaques

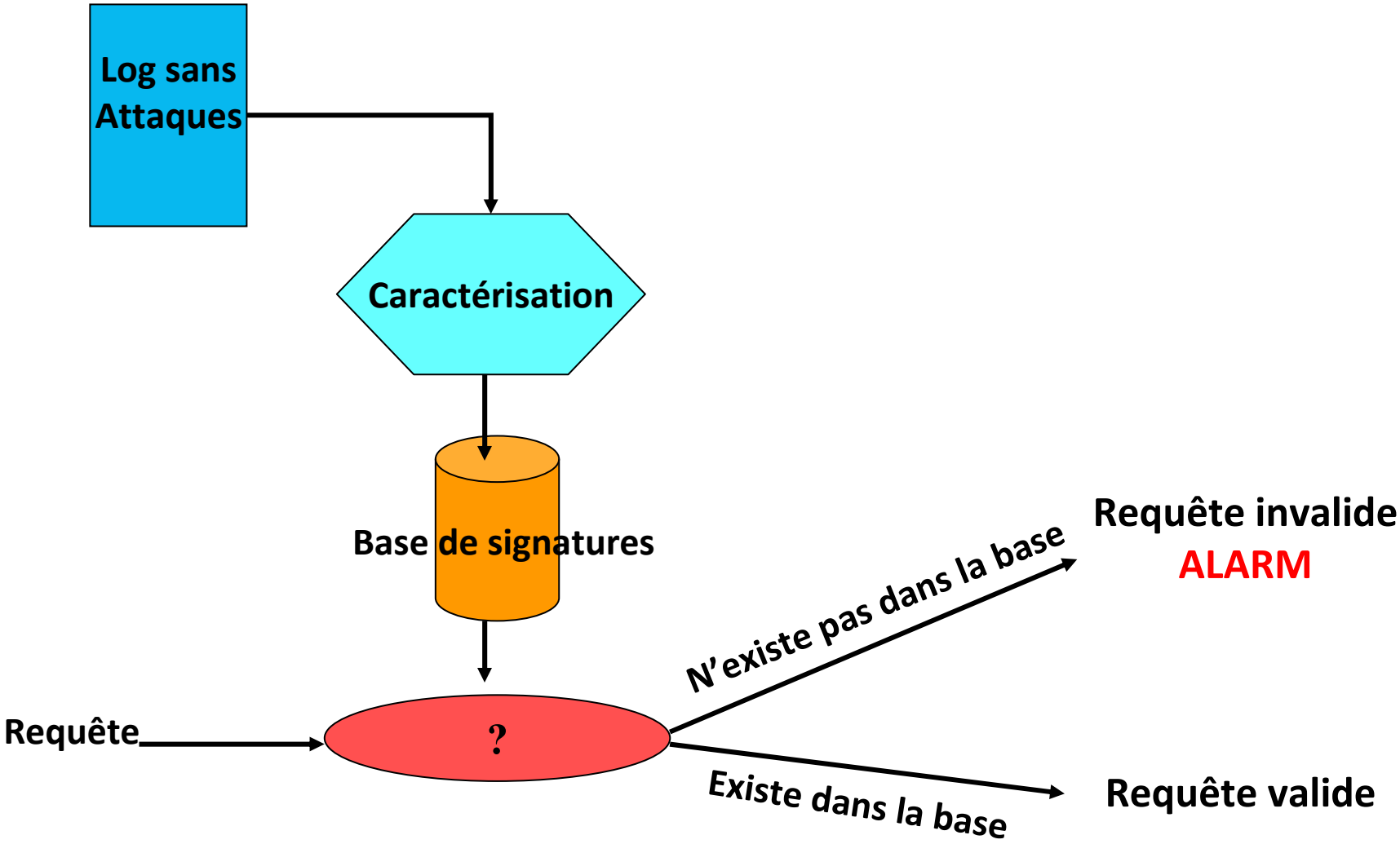
Détection d'anomalies :

non attaques plus nombreuses
Comment caractériser un comportement normal ?
Apprentissage
Mise en production : représentation efficace
Comment faire évoluer les signatures ?

Détection d'abus :

uniquement en cas d'alarmes
Comment caractériser une attaque ?
Apprentissage
Mise en production : en cas d'alerte
Comment faire évoluer les signatures ?

Première approche – sans attaques



Sélection des paramètres

Quid des paramètres
Calculés sur l'URI et sur la valeur de chaque variable de la requête

URI : ensemble de caractères séparés par / ou \
Requête : var1=val1&var2=val2&…&varn=valn

Calculés sur les chaînes originales et sur les chaînes décodées

Sélection des paramètres

Encodage des chaînes

%25 = %; %73 = s; %2573 = s

Exemple : **SELECT * FROM `admin`**

SELECT%2B%2A%2BFROM%2B%2560admin%2560

%2573%2565%256C%2565%2563%2574%2B%252A%2B%2566%2572%256F%256D%2B%2560%2561%256D%2569%256E%2560

Les paramètres

Obtenus à partir de la chaîne originale

- Nombre de %00 (NULL)

  *Location=/etc/passwd**%00**dfgf*

  Pas observés dans les normaux – dans 20% des attaques
- Nombre de caractères codés
- Nombre de caractères codés qui ne nécessitent pas d'encodage (*a-z, A-Z, 0-9*)
- Nombre de caractères encodés deux fois
- Nombre de fois maximal où un caractère est encodé

Remarque : nombre de caractères encodés (é, à, â, ..) peu fréquent dans les requêtes normales

Les paramètres

Obtenus à partir de la chaîne décodée

- Longueur de la chaîne
- Nombre de ‘ ; * / \ < > ..\ ../ |
- Nombre d’apparition des lettres de l’alphabet, nombres, espaces, caractères spéciaux
- Indicateur sur la longueur (buffer overflow)

De nombreux essais et … 21 paramètres

Génération d'un vecteur

Pour URI

/%63alendrier%27%3B%2A/../../calendrier.php

→ 0 4 1 0 1 35 1 1 1 4 0 0 0 2 0 0 23 0 0 0 0

Pour Query arg. #1

3649439%5C4%7C%20%258A8

→ 0 4 0 1 2 13 0 0 0 0 1 0 0 0 0 1 0 9 1 1 0

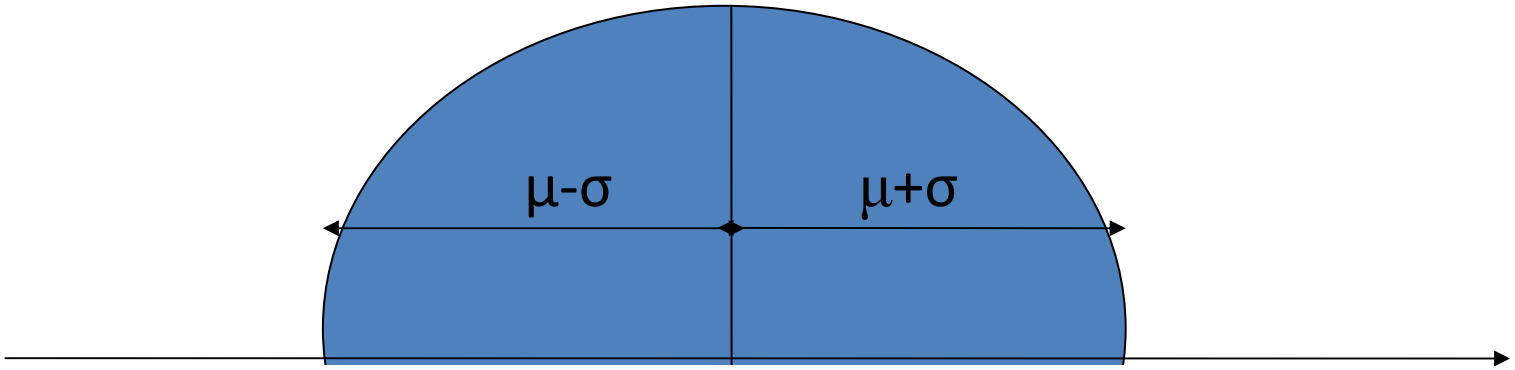
Pour Query arg. #2

qsddff

→ 0 0 0 0 0 6 0 0 0 0 0 0 0 0 0 0 0 6 0 0 0 0

Génération de la distribution

Calcul de la moyenne (μ) et de l'écart type (σ) pour tous les paramètres



Fichier URI

-0.1540 0.1658
-0.0032 0.0032
-0.0007 0.0007
.....

ier Query

104 0.2136
278 0.029
920 19.1362

Création des signatures

Pour chaque entrée, assigner un 1 si la valeur du paramètre rentre dans la distribution et un 0 autrement

Exemple de fichier de signatures

00110110101111O111100
00101010110101010101O
01010101010101010100100

Utilisation de ces signatures pour tester de nouvelles requêtes

Test des requêtes (1/2)

Séparation de la partie URI et de chaque valeur de paramètre

Génération de la signature ($\mu-\sigma \leq val \leq \mu+\sigma$)

Poids de la partie URI	**Poids de la partie QUERY**
0.166667	
0.166667	
0.000000	
0.000000	0.00000
0.000000	0.00555

Test des requêtes (2/2)

Pour l'URI :

Pour chaque signature, le nombre de bits différent est compté et la somme est divisée par le nombre de paramètres (21) de l'URI ($0 \leq weight \leq 1$).

Le minimum de tous les poids est calculé : il caractérise la différence de l'URI par rapport aux signatures

Test des requêtes (2/2)

Pour les valeurs des requêtes :

Calcul des poids pour chaque argument

Procédure répétée pour chaque argument

On conserve le maximum de tous les minimums !

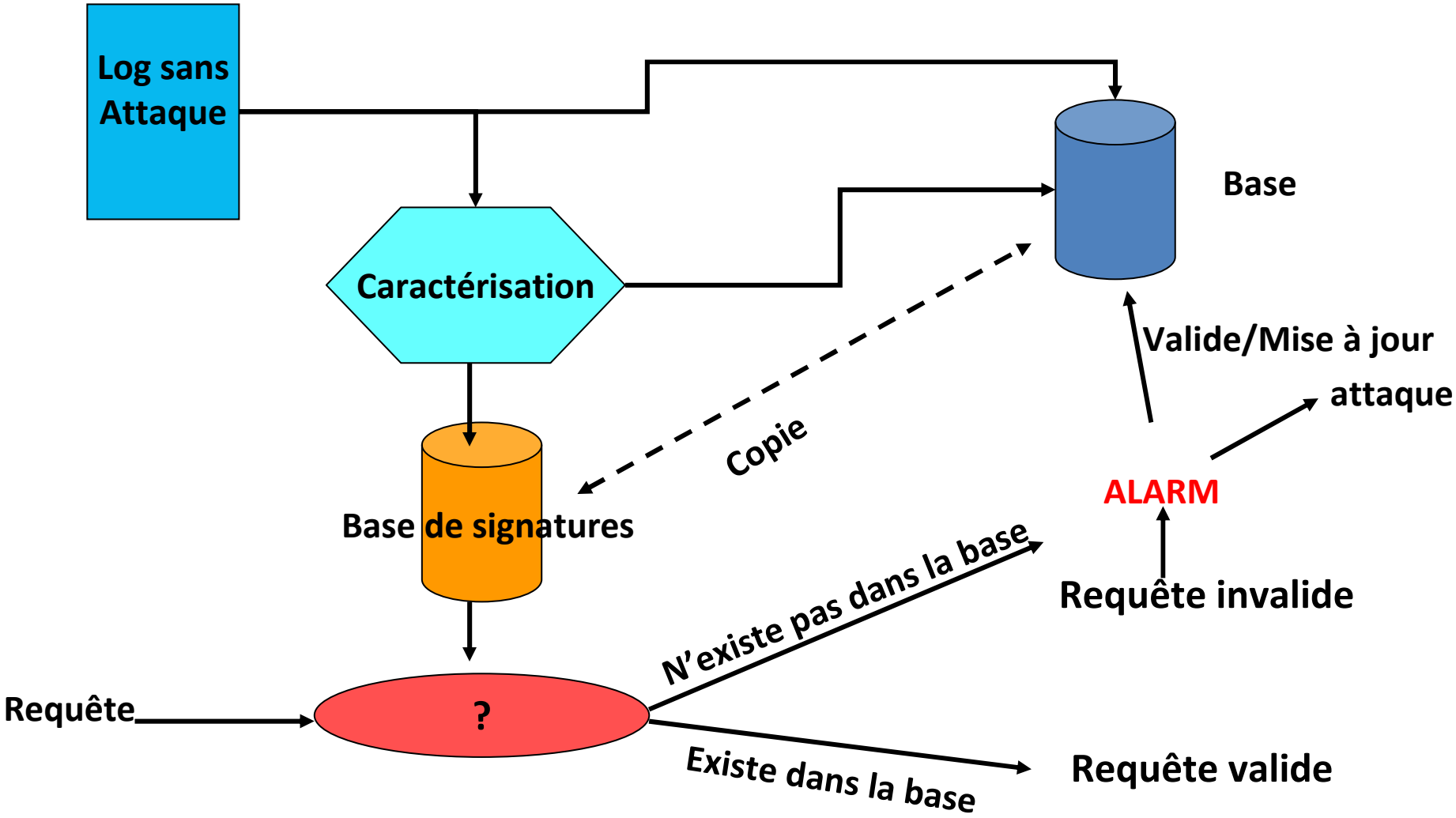
Exemple :

Si une requête a 3 arguments (arg1,arg2,arg3)

Pour les 2 premiers (weight(arg1)=0, weight(arg2)=0)
Pour le dernier (weight(arg3)>0)

Attaque dans l'argument 3

Mise à jour des signatures



Expérimentations 1

Apprentissage de :

1740978 requêtes valides
504497 arguments de requêtes
1740978 URI

Test avec 5603 requêtes invalides et 432853 requêtes valides

Détection de requêtes valides : 99.99%
Détection des requêtes d'attaques : 90.08%

Matrice de confusion	**Valides**	**Attaques**
Valides Prédites	432821	554
Attaques prédites	32	5044

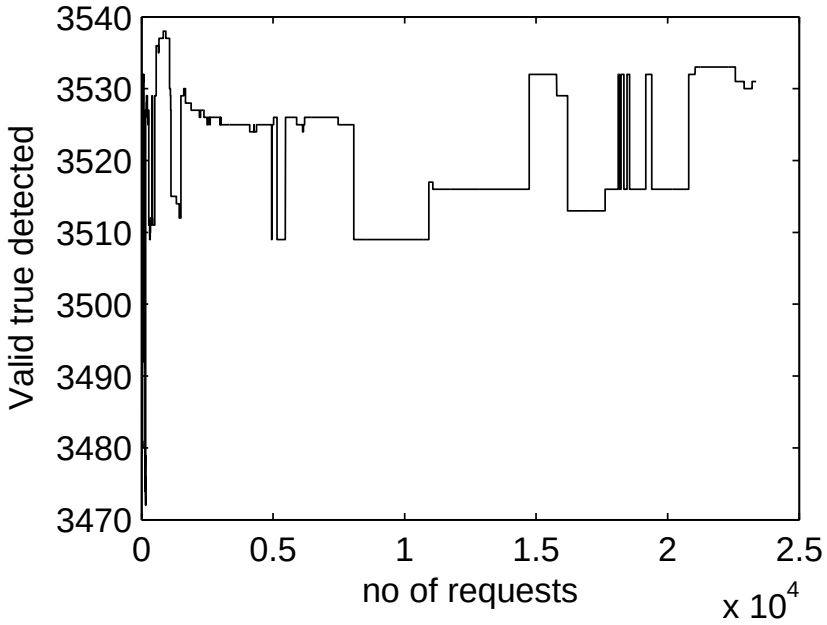
Expérimentations 2

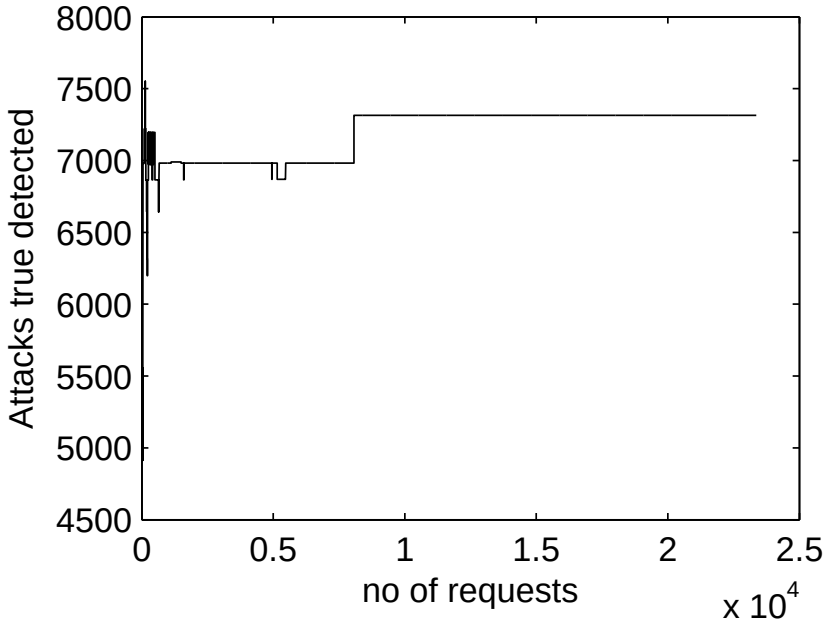
Combien de requêtes doivent être apprises avant de tester l'approche ?

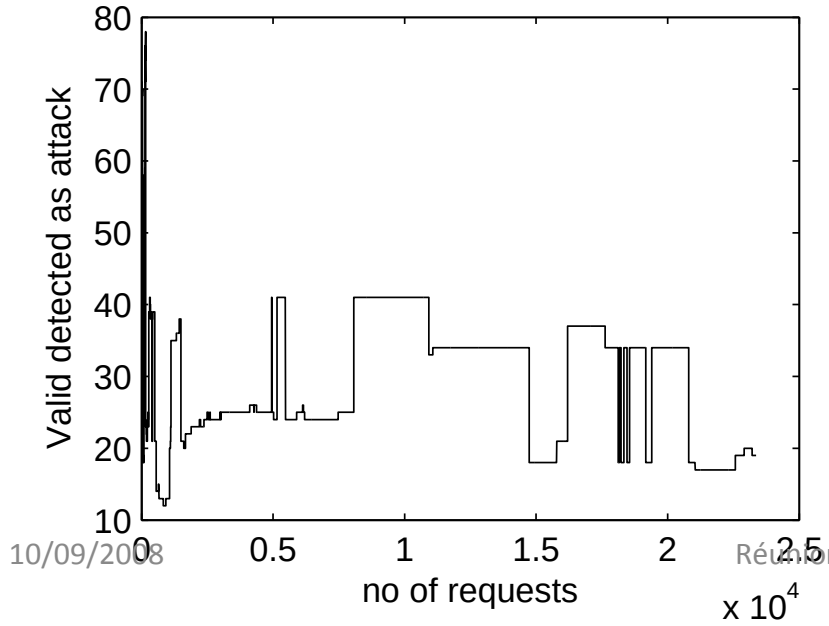
- 23367 requêtes pour l'apprentissage
- 1 requête à la fois – Test avec 3550 attaques et 7618 requêtes valides
- A chaque fois, la signature est mise à jour lorsqu'une nouvelle requête est apprise

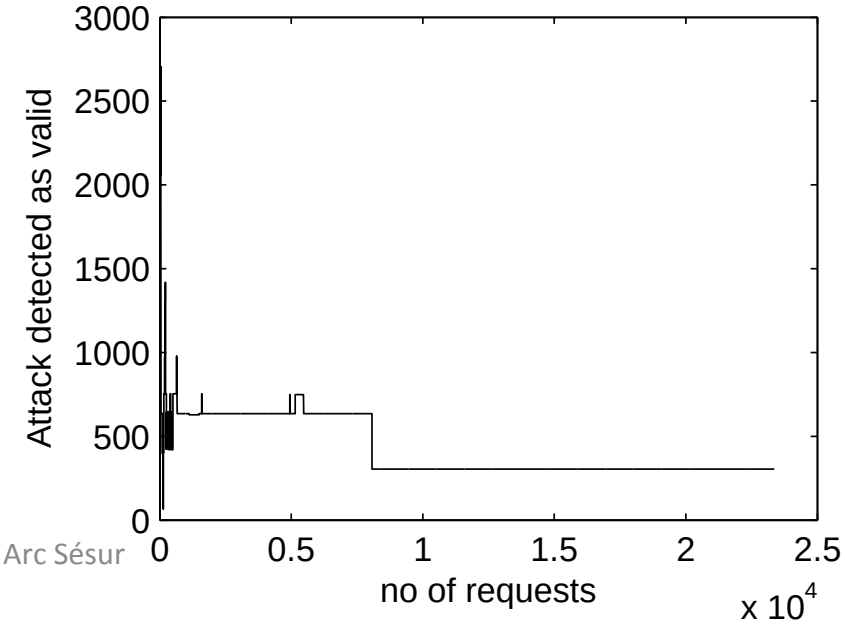
Le nombre de fausses alarmes devient constant après avoir appris # 7500 requêtes

La procédure nécessite 6.82 secondes pour mettre à jour les signatures après avoir appris 23367 requêtes


Valid true detected
3540
3530
3520
3510
3500
3490
3480
3470
0
0.5
1
1.5
2
2.5
no of requests
x 10^4


Attacks true detected
8000
7500
7000
6500
6000
5500
5000
4500
0
0.5
1
1.5
2
2.5
no of requests
x 10^4

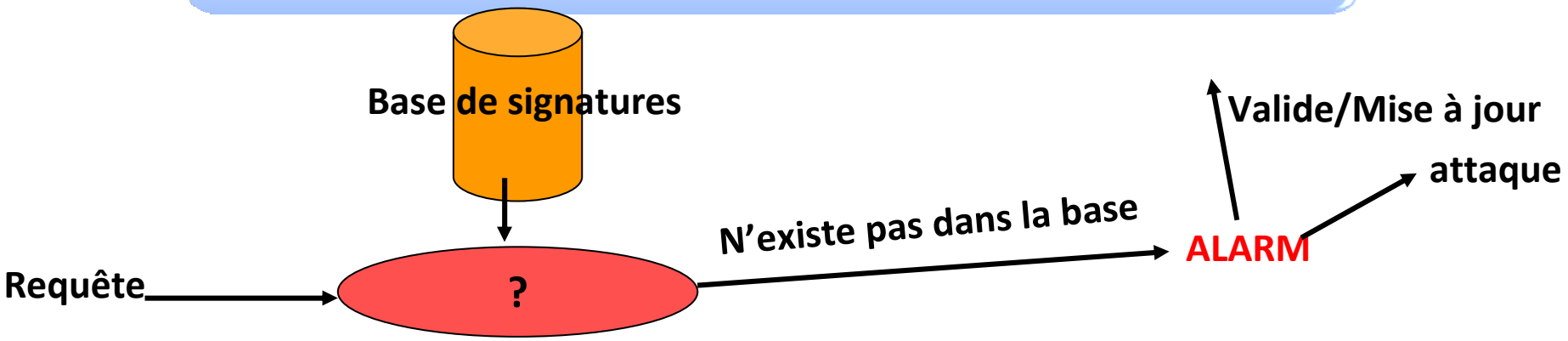

Valid detected as attack
80
70
60
50
40
30
20
10
0
0.5
1
1.5
2
2.5
no of requests
x 10^4


Attack detected as valid
3000
2500
2000
1500
1000
500
0
0
0.5
1
1.5
2
2.5
no of requests
x 10^4

Premières conclusions

Capable de détecter en temps réel des attaques par rapport aux signatures statistiques (plusieurs centaines, milliers en 1 seconde)

Trop d'alarmes ?



Secondes approches – avec attaques

Apprentissage des attaques

Idée : avant de lancer une alarme, regarder si cela correspond à une attaque dans une base de signatures d'attaques
Problème : comment caractériser l'attaque ?

Principes :

- Signature
- Text Mining
- Classification supervisée

Approche par Mots

Objectif

Relation entre les lettres formant des mots pour caractériser les attaques.

Principes

- Séparateurs : tout sauf lettres et chiffres
- Mots : ensemble de caractères entre les séparateurs.

Exemple

http://host/cgi-bin/vuln.cgi?file=../../../../etc/passwd

http	host	cgi	bin	vuln	file	etc	passwd

Approche par Mots

Algorithme
1.Extraction des mots
2.Construction des vecteurs de mots
3.Retenir que les mots les plus pertinents (Information Mutuelle)
4.Application de l'algorithme de classification

Exemple
http://host/cgi-bin/vuln.cgi?file=../../../../etc/passwd
http://host/cgi-bin/vuln.cgi?file=paper.pdf

http	host	cgi	etc	passwd	paper	pdf	Classe		paper	pdf
1	1	2	1	1	0	0	A		0	A
1	1	2	0	0	1	1	Ā		1	Ā

Approche par Mots

Expérimentations

•Jeux de données ECML/PKDD 2007 Discovery Challenge

Classe	Lignes	URL	URI	Query	Body
XSS	93546	1513	1513	1206	207
SSI	93188	1545	1545	1317	228
XPathInjection	115470	1865	1865	1591	274
SQLInjection	127835	2090	2090	1782	308
PathTransversal	151778	2518	2518	1932	334
OSCommanding	179961	2984	2984	2417	280
LDAPInjection	115468	1865	1865	1591	274
Valid	2187479	35006	35006	22309	6653

•Classifieurs

- •Naïve Bayes (NB)
- •C.45

•JAVA/WEKA

Approche par Mots

Paramètres d'expérimentations

- Nombre d'instances par classe : 1500 → 300
- Nombre de mots retenus : 300, 200 et 100
- 15 essais par classifieurs soit 30 essais

Protocole d'expérimentations

- Validation croisée
- Matrice de confusion

	i	$\bar{i}$
$\hat{i}$	VP_i	FP_i
$\hat{\bar{i}}$	FN_i	VN_i

Évaluation des performances

Précision	Rappel	F-mesure
$\frac{VP_i}{VP_i+FP_i}$	$\frac{VP_i}{VP_i+FN_i}$	$\frac{(2\times Rappel \times Precision)}{(Rappel+Precision)}$

Approche par Mots

Meilleur Résultat

300 instances, 300 mots et L'algorithme C.45 P_{valide} = 0.654

a	b	c	d	e	f	g	h	classified
291	1	1	2	3	0	0	1	a = Valid
52	238	4	0	4	0	1	1	b = PathTraversal
8	0	290	0	0	0	0	1	c = SqlInjection
81	2	3	212	0	1	1	0	d = LdapInjection
13	3	3	0	280	0	1	0	e= OsCommanding
0	0	0	0	0	300	0	0	f = XPathInjection
0	1	0	0	0	0	299	0	g = XSS
0	0	1	0	0	0	0	298	h = SSI

Approche par Descripteurs

Objectif

- Relation entre les mots et les caractères pour une meilleure caractérisation

Principes

- Descripteur : ensemble de mots pertinents et de caractères.
- Descripteur représente une URI ou une valeur
- Descripteur : remplacement des mots non pertinents par le symbole « @ »

Exemple

../../../../etc/passwd => N../etc/passwd (normalize)
../../fichier1/fichier2/etc/passwd => ../../@/@/etc/passwd (update Word)
../../@/@/etc/passwd => N../N@/etc/passwd
Jaro-Winkler(« /N../file1 », « N/..file2 ») #1 => descr. commun+2

Approche par Descripteurs

Algorithme
1.Extraction des mots pertinents
2.Génération des descripteurs
3.Construction des vecteurs de descripteurs
4.Regrouper avec similarité (JARO-WINKLER)
5.Retenir que les descripteurs les plus pertinents (Information Mutuelle)
6.Application de l'algorithme de classification

Exemple
http://host/cgi-bin/vuln.cgi?file=../../../../etc/passwd
http://host/cgi-bin/vuln.cgi?file=paper.pdf

@://@/(	N../etc/passwd	paper.pdf	classe	classe
	1	0	A	A
	0	1	Ā	Ā

Approche par Descripteurs

Paramètres d'expérimentations

- Nombre d'instances par classe : 1500 → 300
- Nombre de mots retenus : 300, 200 et 100
- Degré de similarité : 0.5 → 1.0
- 90 essais par classifieurs soit 180 essais

Protocole d'expérimentations

- Validation croisée
- Matrice de confusion

Évaluation des performances

1. Précision
2. Rappel
3. F-mesure

Approche par Descripteurs

Le meilleur du NB

a	b	c	d	e	f	g	h	classified
1500	0	0	0	0	0	0	0	a = Valid
0	1500	0	0	0	0	0	0	b = PathTraversal
0	0	1499	1	0	0	0	0	c = SqlInjection
0	0	0	1500	0	0	0	0	d = LdapInjection
0	0	2	0	1477	2	0	19	e= OsCommanding
0	0	0	0	0	1500	0	0	f = XPathInjection
0	0	0	0	0	0	1500	0	g = XSS
0	0	0	0	138	0	0	1362	h = SSI

Approche par Descripteurs

Le meilleur du C.45

a	b	c	d	e	f	g	h	classified
1500	0	0	0	0	0	0	0	a = Valid
0	1499	0	1	0	0	0	0	b = PathTraversal
0	0	1497	2	0	0	1	0	c = SqlInjection
0	0	0	1500	0	0	0	0	d = LdapInjection
0	0	7	2	1488	0	3	0	e= OsCommanding
0	0	0	0	0	1500	0	0	f = XPathInjection
0	0	0	0	0	0	1500	0	g = XSS
0	0	0	0	144	0	0	1356	h = SSI

Approche par Descripteurs

NB Sans Similarité

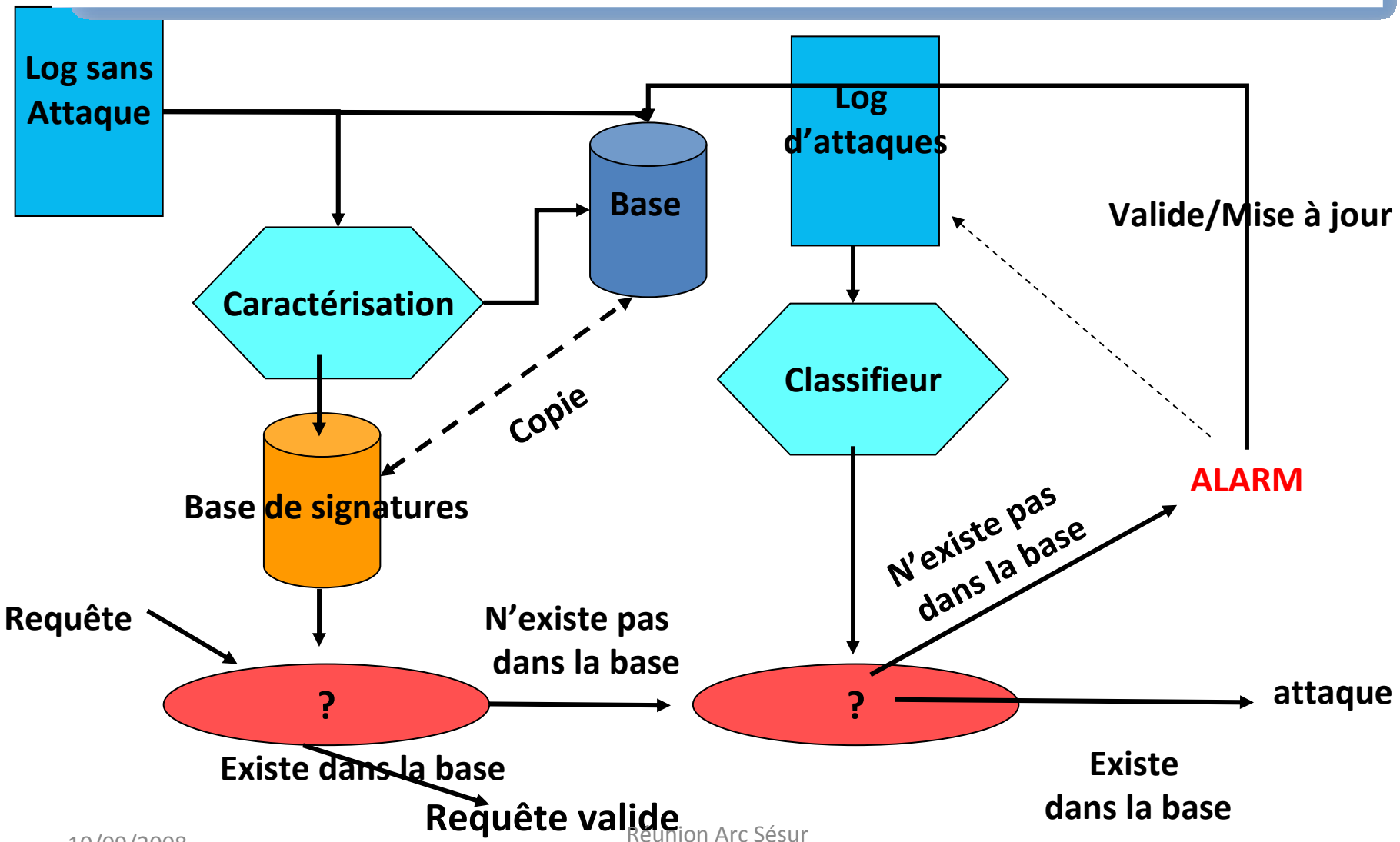
a	b	c	d	e	f	g	h	classified
937	0	0	0	0	0	563	0	a = Valid
0	1493	0	2	0	1	4	0	b = PathTraversal
3	3	1435	1	0	2	56	0	c = SqlInjection
13	2	4	1366	1	5	108	1	d = LdapInjection
20	0	2	3	696	10	97	672	e= OsCommanding
10	2	3	3	0	1374	108	0	f = XPathInjection
34	9	25	11	3	29	1379	10	g = XSS
5	0	0	0	408	3	8	1076	h = SSI

Approche par Descripteurs

C.45 Sans Similarité

a	b	c	d	e	f	g	h	classified
1004	0	151	0	0	0	345	0	a = Valid
49	1254	81	0	0	0	116	0	b = PathTraversal
221	0	613	0	0	1	665	0	c = SqlInjection
97	0	139	1040	0	0	224	0	d = LdapInjection
90	0	140	0	664	0	227	379	e= OsCommanding
82	0	140	1	0	1036	241	0	f = XPathInjection
266	0	435	0	0	0	799	0	g = XSS
62	0	92	0	435	0	111	800	h = SSI

Approche générale



Conclusions

- Possibilité de détecter en temps réel des attaques
- Mise à jour en des signatures de non attaques (en dynamique)
- Mise à jour du classifieur d'attaques (utilisation de clustering sur stream ?)