Détection du pharming côté client vers une redirection du GET HTTP

Sommaire

6.1 Seconde proposition : vers une redirection du GET HTTP 114
6.1.1 Fonctionnement général 114
6.1.2 Conditions d'expérimentation 116
6.1.2.1 Couples de pages légitimes 116
6.1.2.2 Couples de pages légitimes-contrefaites 118
6.1.3 Implémentation : aperçu général 118
6.1.4 Vérification de l'adresse IP du domaine visité 119
6.1.4.1 Implémentation : points spécifiques 119
6.1.4.2 Résultats 123
6.1.4.3 Synthèse IP 124
6.1.5 Analyse et comparaison du code source des pages webs 125
6.1.5.1 Méthodes étudiées : introduction de nouvelles techniques d'analyse du code source des pages webs 126
6.1.5.2 Implémentation : points spécifiques 128
6.1.5.3 Résultats des analyses pour la définition des techniques de comparaisons les plus pertinentes 130
6.1.5.4 Résultats d'analyses des taux d'échec de récupération des pages Défaut et Référence : 134
6.1.5.5 Résultats des Analyses pour l'élaboration de la méthode de comparaison finale 135
6.1.5.6 Problèmes rencontrés 139
6.1.5.7 Synthèse HTML 141
6.1.6 Temps de traitement 143
6.2 Limitations 143
6.2.1 Vérification de l'adresse IP du domaine visité 143
6.2.2 Analyse et comparaison du code source des pages webs 144
6.2.3 Intégration de l'approche dans le navigateur client 145
6.3 Synthèse du chapitre 146

Notre proposition de détection du pharming côté client se base sur une étude du code source HTML de la page web, combinée à des requêtes DNS.

Dans le Chapitre 5, nous avons présenté une première solution de détection du pharming au sein de laquelle la page web visitée est comparée à une page web dite de référence, grâce à la substitution du nom de domaine par une adresse IP au sein de l'URL légitime. Bien que les résultats obtenus au travers de cette première proposition se soient avérés encourageants, ils ont également mis en exergue un certain nombre de faiblesses de l'approche proposée.

Dans ce chapitre, nous développons donc une seconde approche qui améliore la proposition précédente. Nous y expliquons notamment les modifications majeures apportées pour la récupération de la page de référence, désormais basée sur une redirection de la requête HTTP vers une adresse IP déterminée (c.-à-d. l'adresse IP de référence). Nous y détaillons également la combinaison de multiples techniques d'analyse du code source de la page web, afin de déterminer un seuil de décision. Les tests réalisés dans cette seconde proposition portent sur 328 URLs de login légitimes, évaluées depuis 11 localisations géographiques réparties sur 5 continents, ainsi que sur 75 nouveaux couples de pages légitimes-contrefaites.

Ensuite, nous explicitons les limitations associées aux deux approches proposées ainsi que les verrous techniques demeurant à l'issue de notre étude (cf. section 6.2).

Ce chapitre fait partie de nos contributions : cette seconde proposition a été publiée et présentée à la conférence *Network and System Security* (NSS) en Septembre 2011 [GL11].

6.1 Seconde proposition : vers une redirection du GET HTTP

Notre seconde proposition a été élaborée avec le quadruple objectif : de pallier les défauts et problèmes majeurs rencontrés dans la première approche, d'augmenter la base de résultats, d'améliorer les techniques de comparaison et enfin, de définir un seuil de décision.

Les verrous demeurant à l'issue de notre première proposition sont :

- un taux d'échec trop élevé pour la récupération de la page de référence PageRef,
- la difficulté d'identification et d'élimination des pages d'erreur,
- et enfin, l'absence d'identification de l'adresse IPdef, utilisée pour la récupération de la page par défaut PageDef.

6.1.1 Fonctionnement général

En premier lieu, nous nous sommes attachés à répondre au problème majeur du taux d'échec de récupération de PageRef.

Si nos hypothèses sur les causes probables de ce taux d'erreur (cf. section 5.3.5.3) s'avèrent exactes, nous devons trouver un moyen de transmettre l'information du FQDN demandé, lors de la requête de l'URL de référence.

La piste idéale se révèle être la réécriture du paquet afin de conserver l'URL d'origine intacte (et donc transmettre le FQDN), tout en imposant l'adresse du serveur destination.

En conséquence, le fonctionnement de notre seconde proposition se déroule de la manière suivante (cf. figure 6.1) :

Pour chaque page de login, le FQDN de l'URL visitée est extrait afin de générer deux requêtes DNS : l'une est envoyée au serveur DNS par défaut (DNSdef), l'autre est adressée au serveur DNS de référence (DNSref). DNSdef retourne plusieurs adresses IP, dont celle utilisée pour l'affichage de la page web dans le navigateur (IPdef), tandis que DNSref retourne une ou plusieurs adresses IP (IPref), incluant ou excluant IPdef.

Dans le cas où l'adresse IPdef est incluse dans la réponse IPref, le site est considéré comme légitime. Dans le cas contraire, nous procédons à l'analyse de la page web :

- la page web visitée (PageDef), telle qu'affichée dans le navigateur, est récupérée à partir de IPdef.

– La page de référence (PageRef) est récupérée grâce à l'envoi d'une requête GET HTTP, utilisant l'URL originale, à destination de IPref.
Considérons l'exemple suivant : l'URL visitée par l'Internaute est `https://www.amazon.com/gp/yourstore?ie=UTF8&ref_=pd_irl_gw&signIn=1`. La page affichée a été récupérée depuis le serveur web possédant l'adresse IP : 72.21.210.250 (IPdef), adresse connue grâce à la technique de récupération de PageDef utilisée dans cette approche (cf. section 6.1.4). Le serveur DNS de référence interrogé retourne l'adresse IP : 207.171.166.252 (IPref). La page de référence est alors récupérée en envoyant un GET HTTP, contenant l'URL `https://www.amazon.com/gp/yourstore?ie=UTF8&ref_=pd_irl_gw&signIn=1`, à l'adresse IP destination IPref : 207.171.166.252.

Les codes sources HTML des deux pages webs (PageDef et PageRef) sont ensuite comparés (cf. figure 6.1), grâce aux techniques énoncées en sections 5.2.2 et 6.1.5. Enfin, la légitimité de la page web visitée est déterminée en comparant le pourcentage de similitude obtenu entre les deux pages (PageDef et PageRef) à un seuil de décision pré-défini.

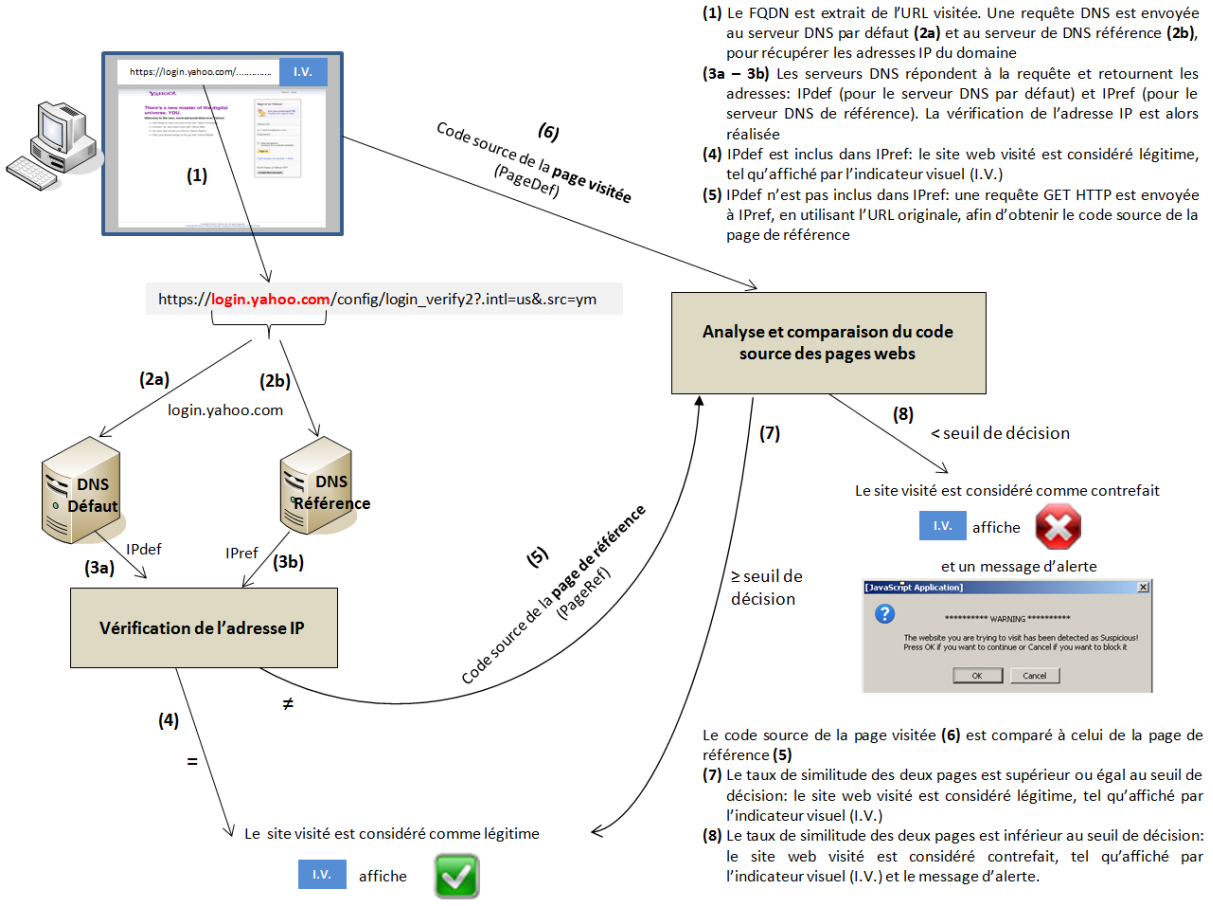


FIGURE 6.1 – Seconde approche : fonctionnement général

Définition du serveur de référence : Au vu des résultats obtenus précédemment, nous avons élaboré un nouveau scénario de définition du serveur de référence. L'utilisateur aurait alors le choix entre deux propositions : 1/ le choix du DNSref est laissé à son appréciation parmi une liste de serveurs pré-définis (= choix exposé dans la première proposition), ou 2/ le programme sélectionne et utilise - de manière automatique et aléatoire - un serveur DNSref, parmi une liste de serveurs pré-définis. Cette sélection, qui serait renouvelée à chaque vérification d'une page de login, est proposée grâce aux résultats très similaires obtenus auprès des différents serveurs DNS de référence. L'avantage de ce deuxième scénario - qui serait celui recommandé - serait de proposer une meilleure résistance aux attaques (cf. section 4.1.2.1), grâce au choix aléatoire effectué pour DNSref, renouvelé à chaque vérification.

Tableau 6.1 – Répartition des 328 URLs de login légitimes par secteur d'activités

Catégories	Quantité	Pourcentage
Banques	204	62%
Autres (administration, assurances, logiciels, jeux, FAI, industrie, vidéos, photos, informations)	71	22%
e-commerce	44	13%
Réseaux sociaux	5	2%
email	4	1%

6.1.2 Conditions d'expérimentation

Sur le même principe que celui utilisé lors de notre première proposition (cf. section 5.3.2), nous avons évalué l'efficacité de notre proposition à partir de deux types de comparaisons :

- des comparaisons de couples de pages de login légitimes récupérées depuis plusieurs localisations géographiques et plusieurs serveurs DNS (c.-à-d. le serveur DNS par défaut de l'utilisateur ainsi que nos trois serveurs DNS de référence : OpenDNS, GoogleDNS et DNSAdvantage). L'ensemble des couples de pages légitimes ont été collectées entre Mars et Juin 2011.
- des comparaisons de couples de pages de login légitimes-contrefaites, visuellement très similaires. L'ensemble des pages contrefaites (et légitimes associées) utilisées ici ont été récupérées entre Décembre 2010 et Juin 2011.

Les tests effectués se sont ensuite déroulés en deux temps :

- Une première analyse qui a deux objectifs : 1/ Mesurer l'efficacité de cette seconde proposition vis-à-vis de la précédente, tant au niveau des résultats IP que de l'analyse sur le code source HTML dans son intégralité, et 2/ Explorer de nouvelles techniques de comparaison pour n'en retenir que les plus pertinentes. Pour la suite du chapitre, cette première analyse sera nommée *Analyses pour la définition des techniques de comparaison les plus pertinentes*.
- Une deuxième analyse visant à définir le seuil de décision ainsi que la méthode de comparaison finale des pages webs (à partir des techniques de comparaison retenues en première analyse). Pour la suite du chapitre, cette deuxième analyse sera nommée *Analyses pour l'élaboration de la méthode de comparaison finale*.

6.1.2.1 Couples de pages légitimes

Nous avons établi un panel de 328 URLs de login décomposé de la manière suivante :

- **224 nouvelles URLs (dont 2 à FQDN commun)**. Ces 224 URLs ont été sélectionnées en utilisant les mêmes critères que précédemment, à savoir : diversifier les secteurs d'activités, multiplier les TLDs ainsi que les langages des pages webs.
- **et les 104 URLs à FQDN uniques utilisées dans la première proposition**. L'intégration de ces 104 URLs a pour objectif d'avoir un premier indicateur de la variabilité temporelle des résultats obtenus précédemment (cf. sections 5.3.4.2 et 5.3.5.2).

Classification des 328 URLs de login : Nous avons classifié les 328 URLs sélectionnées selon les 5 secteurs d'activités identifiés dans la première proposition, à savoir : banques, réseaux sociaux, e-commerce, email et autres (cf. section 5.3.2.1 pour plus de détails sur les classifications). La répartition des catégories, par ordre d'importance, reste identique à l'étude précédente, avec une nette prédominance des sites bancaires (cf. tableau 6.1).

Les 328 URLs sélectionnées sont issues de 48 TLDs différents, divisables en 2 catégories : les cc-TLD et les g-TLD. Pour une meilleure lisibilité, les TLDs sélectionnés ont été regroupés par continent (cf. tableau 6.2). A nouveau, la répartition des TLDs reste à peu près similaire à l'étude précédente, même si leur nombre a plus que doublé. On constate également l'apparition d'un nouveau g-TLD : *COOP* pour *Cooperative*.

Tableau 6.2 – Répartition des 48 TLDs des 328 URLs de login légitimes

	Pourcentage	TLD
Europe	38.1%	AD, AT, BA, BE, BG, CY, CZ, DE, DK, EE, FI, FR, GR, IT, LU, LV, NL, NO, PL, RO, SE, UK
Asie	5.8%	GE, ID, IL, IN, JO, JP, MV, MY, PH, PK, SG, TR
Océanie	4.3%	AU, NZ
Amérique du Sud	3.4%	AR, BR, CL, CO, MX
Afrique	0.9%	ZA
Amérique du Nord	0.6%	CA, HN
Commercial	44.8%	COM
Network	1.2%	NET
Organization	0.6%	ORG
Cooperative	0.3%	COOP

Analyses pour la définition des techniques de comparaison les plus pertinentes : Pour comparer nos deux propositions et définir les techniques de comparaison les plus pertinentes, nous avons testé nos 328 URLs de login sur 11 sites géographiques, répartis sur 5 continents. Nous avons ainsi collecté les résultats issus de 4 serveurs DNS (c.-à-d. le serveur DNS par défaut de l'utilisateur - vérifié différent des serveurs DNS de référence -, et nos 3 serveurs de référence).

La distribution géographique des tests effectués est la suivante (cf. figure 6.2) :

- Europe : France (3 tests en Île de France et 1 test dans le Sud de la France), Belgique
- Amérique du Nord : États-Unis, Mexique
- Amérique du Sud : Vénézuela
- Asie : Chine, Turquie
- Afrique : Sénégal

A noter que pour certaines comparaisons effectuées (cf. section 6.1.5), nous avons parfois restreint nos analyses à 6 localisations, principalement par manque de temps.

Dans un second temps, depuis une même localisation (située en Île de France), nous avons testé les 328 URLs à 10 reprises, sur une période de 3 mois (entre Avril et Juin 2011). Ceci afin d'avoir une seconde mesure de la variabilité temporelle des résultats.

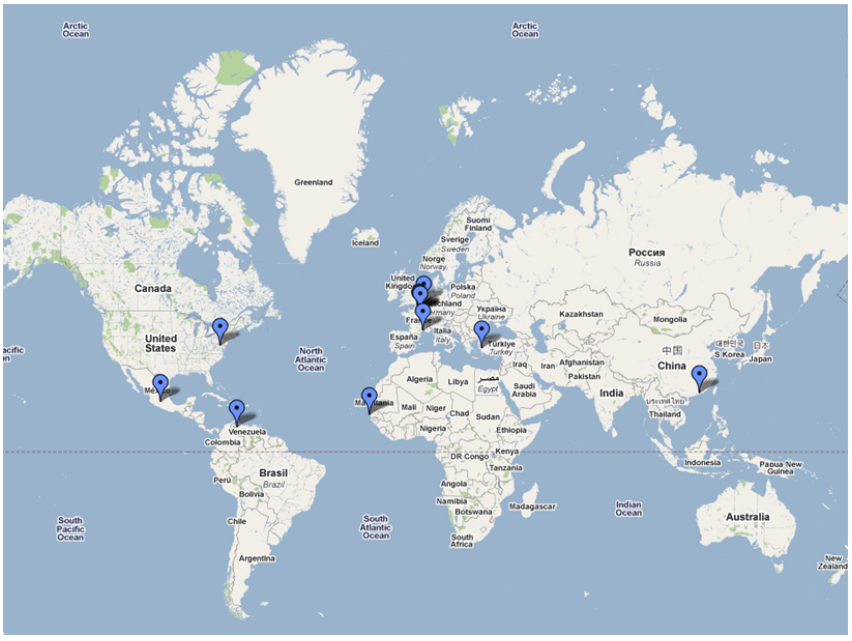


Figure 6.2 – Seconde approche : répartition géographique des tests

A noter que, comme dans la première proposition, nous avons rencontré des pertes de connectivité importantes (rupture de courant électrique et ou Internet) sur les localisations du Vénézuela et du Sénégal. Ceci peut expliquer des taux d'erreur sensiblement plus élevés sur ces localisations.

Analyses pour l'élaboration de la méthode de comparaison finale : Enfin, pour élaborer une méthode de comparaison basée sur les techniques les plus pertinentes et déterminer un seuil de décision pour la comparaison des pages webs, nous avons choisi d'étalonner notre solution sur les résultats issus d'une seule localisation et d'un seul couple DNSdef-DNSref : Bruxelles(Belgique) / DNSdef-GoogleDNS. Puis, une fois défini le seuil de décision, nous avons vérifié l'efficacité de notre méthode de détection sur 5 autres localisations : Mexico(Mexique), Montpellier(France-Sud), Samoreau(France-IdF), Dakar(Sénégal) et Shenzhen(Chine). Pour chaque localisation, nous n'avons utilisé qu'un seul couple DNSdef-DNSref, respectivement DNSdef- : GoogleDNS, OpenDNS, DNSAdvantage, OpenDNS et GoogleDNS.

6.1.2.2 Couples de pages légitimes-contrefaites

Les couples de pages légitimes contrefaites, visuellement très similaires, ont été récupérées selon la méthode exposée en section 5.3.2.2.

L'ensemble des pages collectées nous ramène aux 5 secteurs d'activité exposés précédemment.

Analyses pour la définition des techniques de comparaison les plus pertinentes : Nous avons étudié la variabilité des résultats, par rapport à la première proposition, sur 75 nouveaux couples de pages légitimes-contrefaites.

Analyses pour l'élaboration de la méthode de comparaison finale : Puis, pour la définition de notre seuil de décision, nous avons étalonné notre méthode de détection sur 55 couples de pages légitimes-contrefaites (extraits des 75 couples précédents). Nous avons ensuite vérifié notre méthode de détection sur un deuxième jeu de 58 couples de pages légitimes-contrefaites, répartis de la façon suivante : 20 couples de pages - non utilisés dans l'étalonnage - sont issues des 75 couples précédents, auxquels ont été rajoutés 38 nouveaux couples de pages.

6.1.3 Implémentation : aperçu général

La nouveauté - et la difficulté - majeure de cette seconde proposition réside dans la récupération de la page de référence, effectuée via une redirection du GET HTTP vers une adresse IP définie.

A cet effet, nous avons exploré deux pistes/scénarios d'implémentation visant à intervenir à un niveau inférieur du modèle OSI (en comparaison des fonctions de récupération de pages utilisées dans la première approche) :

- Une première piste a consisté en l'exploration de l'utilisation d'une technique d'écoute des paquets. L'objectif visé étant d'écouter la requête de la PageDef, afin de générer la requête de la PageRef.
- Une seconde piste a consisté en la réécriture complète de la requête GET HTTP, au travers de l'écriture de sockets.

Ces deux pistes sont détaillées en section 6.1.4.

L'implémentation de la première piste a automatiquement induit des modifications majeures dans l'ordonnancement initial de nos programmes. En effet, la nécessité d'un modèle de paquet et la génération de la nouvelle requête GET HTTP qui devaient être regroupés au sein d'un même programme, ont fortement impacté notre technique - initialement très séparée - de récupération des informations Défaut (adresse IP et page web) et Référence (cf. section 6.1.4.1).

Au travers de la seconde piste, l'implémentation est revenue à son organisation initiale, présentant ainsi le même type de découpage que dans la première proposition.

Au passage, on peut noter que l'intérêt de connaître l'adresse IPdef s'est révélé ici devenir une véritable nécessité, et ce quelle que soit la piste retenue.

Concernant l'analyse du contenu du code source de la page web, nous avons également exploré de nouvelles pistes visant à une meilleure différenciation des pages légitimes et contrefaites. L'objectif est triple : la définition d'un seuil de décision, l'éventuelle optimisation/diminution du temps de traitement, et l'éventuelle détection des zones de codes majoritairement impactées par la contrefaçon (pour en déduire les techniques les plus pertinentes à utiliser).

En complément des techniques de détection étudiées dans notre première proposition (c.-à-d. approches par caractères et par mots, appliquées au code source complet), nous avons donc exploré deux nouvelles familles de pistes (détaillées en section 6.1.5) : 1/ application de la méthode de calcul sur des sous-parties du code source complet (c.-à-d. dans son intégralité), et 2/ analyse des balises contenues dans le code source.

A noter que pour une meilleure lisibilité, le *code source complet* sera désormais nommé *code complet* dans la suite de ce chapitre. A ne pas confondre avec la page web complète (qui inclut des images, des fichiers de scripts complémentaires, etc.) que nous n'avons pas explorée dans notre étude.

6.1.4 Vérification de l'adresse IP du domaine visité

L'objectif de cette section est de déterminer si la relative staticité des adresses IP associées aux FQDN des pages de login, se confirme sur davantage d'URLs (vs. les résultats obtenus lors de la première proposition).

En complément, cette section détaille les modifications majeures apportées à notre implémentation afin de répondre aux deux problèmes intrinsèquement liés que sont : l'identification de l'adresse IPdef utilisée pour récupérer PageDef, et la mise en œuvre de l'implémentation de la redirection du GET HTTP.

Les résultats de tests obtenus ici se concentrent sur l'étude des URLs à FQDN unique, c.-à-d. 327 des 328 URLs sélectionnées.

6.1.4.1 Implémentation : points spécifiques

Premier scénario d'écoute des paquets : Une première piste étudiée pour la réécriture du GET HTTP vers une adresse IP choisie, a été l'utilisation de l'API *pcap* (pour *packet capture*), couramment utilisée par les outils de supervision du trafic réseau. La piste évoquée consiste alors à utiliser un modèle de paquet (c.-à-d. le premier GET HTTP envoyé pour la récupération de PageDef) pour générer le GET HTTP de PageRef.

Le fonctionnement envisagé pour notre programme devient alors le suivant : 1/ Un premier programme contenant la requête DNS vers DNSdef (cf. algorithme 4), et 2/ Un second programme contenant : la requête DNSref, la récupération des pages webs et le calcul des scores en utilisant les approches par caractères et par mots (cf. algorithme 5).

Algorithme 4 Seconde approche avec scénario *pcap* : DNS par défaut

Entrées: le fichier *.txt* contenant la liste des n URLs de login.
Sorties: 1 fichier *DNSqueries_default.txt* contenant les n réponses du serveur par défaut.

1: **pour** $i = 0$ **à** n **faire**
2: Requête DNS auprès du serveur par défaut pour le FQDN(i).
3: Sauvegarde des adresses IP (*IPdef*) retournées pour le FQDN(i) (un même fichier pour les n URLs).
4: **fin pour**

Pour notre implémentation, nous avons testé deux librairies Java *jNetCap* [Tec] et *Jpcap* [UoC]. Les difficultés majeures associées à cette approche et implémentation résident dans les étapes 2, 3 et 9 de l'algorithme 5 :

- Dans l'étape N° 2, si DNSdef a retourné plusieurs adresses IP, nous n'avons pas connaissance de l'adresse qui est utilisée pour la récupération de PageDef en étape N° 4.
- L'écoute et l'interception du paquet à l'étape N° 3 doit se faire grâce à l'utilisation d'un filtre ciblé sur l'adresse IP destination IPdef. En effet, un filtrage par port applicatif ne peut être suffisant (à moins d'interdire toute autre navigation Internet simultanée, ce qui est inenvisageable en dehors d'un environnement expérimental). Or, l'utilisation de la fonction habituelle *url.openConnection()*

pour la récupération de PageRef ne nous donne pas connaissance de cette adresse IP. Il faudrait alors envisager un filtrage sur l'ensemble des adresses IP retournées par DNSdef pour le FQDN concerné, espace dont la taille est variable et changeante à chaque requête DNS.

– Enfin l'étape 9 - facilement réalisable pour une page HTTP classique - s'est avérée bloquante pour une page de login HTTPS, puisque nos différents essais à ce propos se sont révélés insatisfaisants. En effet, en préambule du GET HTTP, l'ensemble des échanges liés à l'établissement de la connexion sécurisée doivent être réalisés. Ceci revient à effectuer une implémentation complète de SSL, ce qui représente un codage relativement lourd et complexe.

Algorithme 5 Seconde approche avec scénario *pcap* : DNS de référence

Entrées: le fichier *.txt* contenant la liste des n URLs de login, et le fichier *DNSqueries_default.txt* créé par le programme précédent (cf. algorithme 4).

Sorties: 1 fichier contenant les n réponses du serveur de référence, 3 fichiers contenant les scores des n pages (2 pour l'approche par caractères et 1 pour l'approche par mots), et n fichiers contenant les codes sources des pages webs de référence.

1: **pour** $i = 0$ **à** n **faire**
2: Récupération de l'adresse IPdef dans le fichier *DNSqueries_default.txt*.
3: Écoute du paquet à suivre.
4: Récupération et sauvegarde du code source de la page web par défaut en utilisant l'URL `https://FQDN(i)/arborescence1/.../arborescenceX/fichier.html`. (1 fichier par URL, nommé *date_heure_def_FQDN*(i)*.txt*)
5: Calcul du score de PageDef(i) avec l'approche par caractères, puis sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs).
6: Requête DNS auprès du serveur de référence pour le FQDN(i).
7: Sauvegarde des adresses IP (*IPref*) retournées pour le FQDN(i) (un même fichier pour les n URLs).
8: Récupération de la première adresse IP *IPref*, différente de *IPdef*.
9: Création du nouveau paquet pour la récupération et la sauvegarde du code source de la page web de référence, en utilisant l'URL `https://FQDN(i)/arborescence1/.../arborescenceX/fichier.html`. Le nouveau paquet créé est envoyé à destination de *IPref* (1 fichier par URL, nommé *date_heure_nomDNSref_FQDN*(i)*.txt*).
10: Calcul du score de PageRef(i) avec l'approche par caractères, et sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs).
11: Calcul du taux de similitude PageDef(i)/PageRef(i) avec l'approche par mots, et sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs).
12: **fin pour**

Second scénario d'écriture de socket : Une seconde piste étudiée a été la réécriture de la socket qui contient le GET HTTP. En effet, la fonction *url.openConnection()*, classiquement utilisée pour la récupération d'une page web, est une version simplifiée d'une socket présentant les arguments *host* (contenant le FQDN de l'URL demandée) et *port* (contenant le port destination, c.-à-d. 443 pour le HTTPS). Une fois la socket ouverte, les paramètres du GET HTTP peuvent alors être spécifiés.

Notre implémentation repose sur 4 librairies Java : *java.net.Socket* pour l'écriture des sockets, ainsi que *java.security.**, *javax.net.ssl.** et *javax.net.SocketFactory* pour l'établissement de la connexion HTTPS (c.-à-d. l'utilisation d'un modèle de socket SSL pré-établi).

Pour générer la requête de PageDef nous utilisons la version basique de la fonction *socket*, tandis qu'une deuxième version plus élaborée (permettant de spécifier l'adresse IP du serveur destination) est utilisée pour récupérer PageRef.

Pour illustrer nos propos, considérons deux exemples d'implémentation de la récupération de PageDef et PageRef, pour l'URL `https://login.yahoo.com/config/mail?&.src=ym&.intl=fr`, basés sur l'utilisation de socket :

– L'algorithme 6 illustre un exemple de récupération de PageDef : les étapes 1 à 3 permettent la définition de l'URL et du port destination associé, utilisés pour la requête. Puis, les étapes 4 et 5 créent et ouvrent la socket HTTPS pour le FQDN requis. Enfin, les étapes 6 à 10 spécifient le contenu du GET HTTP, à savoir : le chemin d'accès du fichier demandé.

Algorithme 6 Exemple de requête HTTP à base de socket, pour la récupération de PageDef

```
1: String FQDN = "login.yahoo.com";
2: String file = "/config/mail?&.src=ym&.intl=fr";
3: int portdest = 443;
4: SocketFactory socketFactory = SSLSocketFactory.getDefault();
5: s = socketFactory.createSocket(FQDN, portdest);
6: OutputStream out = s.getOutputStream();
7: PrintWriter outw = new PrintWriter(out, false);
8: outw.print("GET " + file + " HTTP/1.0\r\n");
9: outw.print("Accept : text/plain, text/html,text/*\r\n");
10: outw.print("\r\n");
```

– L'algorithme 7 illustre un exemple de récupération de PageRef : les étapes 1 à 4 permettent la définition de l'URL, de l'adresse IP et du port destination, utilisés pour la requête. Puis, les étapes 5 et 6 créent et ouvrent la socket HTTPS avec le serveur possédant l'adresse IP destination spécifiée précédemment. Enfin, les étapes 7 à 12 spécifient le contenu du GET HTTP, à savoir : le chemin d'accès du fichier ainsi que le FQDN demandé.

Algorithme 7 Exemple de requête HTTP à base de socket, pour la récupération de PageRef

```
1: String FQDN = "login.yahoo.com";
2: String IPref = "217.146.187.123";
3: String file = "/config/mail?&.src=ym&.intl=fr";
4: int portdest = 443;
5: SocketFactory socketFactory = SSLSocketFactory.getDefault();
6: s = socketFactory.createSocket(IPref, portdest);
7: OutputStream out = s.getOutputStream();
8: PrintWriter outw = new PrintWriter(out, false);
9: outw.print("GET " + file + " HTTP/1.0\r\n");
10: outw.print("Accept : text/plain, text/html, text/*\r\n");
11: outw.print("Host : " + FQDN + "\r\n");
12: outw.print("\r\n");
```

Choix du scénario : De ces deux scénarios, la solution retenue est donc celle présentant le scénario le plus favorable, à savoir : l'écriture de socket.

Notre implémentation, dont l'ordonnancement est très similaire à celle de la première proposition (cf. section 5.3.3), est illustrée par les algorithmes 8 et 9.

A noter que dans cette nouvelle implémentation, nous avons également supprimé la notion d'"heure" dans les noms de fichiers contenant PageDef et PageRef. Ceci afin d'éviter tout problème de traitement ultérieur des pages (cf. section 5.3.3.2), traitement devenu indispensable pour tester les nouvelles techniques de comparaisons de pages étudiées en section 6.1.5.1.

Identification de l'adresse IPdef et choix de l'adresse IPref : Un autre avantage de la fonction *socket* est de pouvoir identifier avec certitude l'adresse IP utilisée pour la récupération de la page par défaut. En effet, reprenons l'exemple utilisé précédemment, à savoir la récupération de PageDef associée à l'URL `https://login.yahoo.com/config/mail?&.src=ym&.intl=fr`. Alors que notre serveur DNSdef a retourné les adresses IP 217.12.8.76 et 217.146.187.123 pour le FQDN `login.yahoo.com`, l'affichage de la socket utilisée pour récupérer PageDef retourne les éléments suivants : `Socket[addr=login.yahoo.com/217.12.8.76,port=443,localport=55271]`. On y retrouve aisément les informations de : FQDN, adresse IP associée, port destination et port source. L'adresse IPdef est alors extraite grâce à une analyse du champ *addr* de la socket affichée.

Algorithme 8 Seconde approche avec scénario *socket* : DNS par défaut

Entrées: le fichier *.txt* contenant la liste des n URLs de login.

Sorties: 1 fichier contenant les n réponses du serveur par défaut, 1 fichier contenant les scores (approche par caractères) des n pages par défaut, 1 fichier contenant les n adresses IPdef utilisées pour récupérer les n pages par défaut, et n fichiers contenant les codes sources des pages webs par défaut.

1: **pour** $i = 0$ **à** n **faire**
2: Requête DNS auprès du serveur par défaut pour le FQDN(i).
3: Sauvegarde des adresses IP (*IPdef*) retournées pour le FQDN(i) (un même fichier pour les n URLs).
4: Récupération et sauvegarde du code source de la page web par défaut `https://FQDN(i)/arborescence1/.../arborescenceX/fichier.html` en créant une socket HTTPS basique (1 fichier par URL, nommé *date_def_FQDN*(i)*.txt*).
5: Extraction[1] et sauvegarde de l'adresse IP (*IPdef*) utilisée pour récupérer Pagedef.
6: Calcul du score (approche par caractères) de la page(i) récupérée, puis sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs).
7: **fin pour**

[1] cf. explications sur la méthode d'extraction dans le paragraphe "Identification de l'adresse IPdef" de la section en cours.

Algorithme 9 Seconde approche avec scénario *socket* : DNS de référence

Entrées: le fichier *.txt* contenant la liste des n URLs de login et le fichier contenant les n adresses IPdef utilisées pour récupérer les n pages par défaut.

Sorties: 1 fichier contenant les n réponses du serveur de référence, 1 fichier contenant les scores (approche par caractères) des n pages de référence, 1 fichier contenant les scores (approche par mots) des n pages défaut vs. les n pages de référence, et n fichiers contenant les codes sources des pages webs de référence.

1: **pour** $i = 0$ **à** n **faire**
2: Requête DNS auprès du serveur de référence pour le FQDN(i).
3: Sauvegarde des adresses IP (*IPref*) retournées pour le FQDN(i) (un même fichier pour les n URLs).
4: Lecture de l'adresse IPdef utilisée pour récupérer la page par défaut du FQDN(i) et comparaison avec les résultats retournés par DNSdef :
5: **si** DNSref a retourné une seule adresse IP **alors**
6: l'adresse IPref utilisée pour récupérer PageRef est l'adresse IP retournée par DNSref.
7: **sinon si** DNSref a retourné plusieurs adresses IP **alors**
8: l'adresse IPref utilisée pour récupérer PageRef est la première adresse IP trouvée différente de IPdef[1].
9: **finsi**
10: Récupération et sauvegarde du code source de la page web de référence `https://FQDN(i)/arborescence1/.../arborescenceX/fichier.html` en créant une socket HTTPS avancée, destinée à l'IPref sélectionnée précédemment (1 fichier par URL, nommé *date_nomDNSref_FQDN*(i)*.txt*).
11: Calcul du score (approche par caractères) de la page(i) récupérée, puis sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs)
12: Calcul du taux de similitude des PageDef et PageRef de la page(i) avec l'approche par mots, puis sauvegarde du résultat dans un fichier *.txt* (1 fichier pour les n URLs)
13: **fin pour**

[1] cf. explications détaillées et exemple dans le paragraphe "Identification de l'adresse IPdef" de la section en cours.

A noter que pour améliorer notre seconde proposition (vs. la première proposition), nous avons choisi d'utiliser une adresse IPref obligatoirement différente de l'adresse IPdef dès lors que cela s'avérait possible. En effet, dans le cas où DNSref retourne plusieurs adresses IP, l'adresse IPref utilisée est désormais la première adresse IP trouvée différente de IPdef.

Par exemple, supposons que l'adresse IPdef utilisée pour récupérer la page web par défaut est

Tableau 6.3 – Comparaison des réponses du DNS par défaut vs. 3 serveurs DNS de référence, sur 11 localisations géographiques

	Taux de convergence avec les adresses IP retournées par le serveur DNS par défaut (min ≤ moyenne ≤ max)	Écart-Type [1]	Intervalle de confiance à 95% [1]
OpenDNS	82.62% ≤ **86.29%** ≤ 92.66%	2.44%	[84.85% ; 87.73%]
GoogleDNS	85.80% ≤ **87.72%** ≤ 93.60%	2.51%	[86.24% ; 89.20%]
DNSAdvantage	84.36% ≤ **86.51%** ≤ 87.50%	1.05%	[85.89% ; 87.13%]

[1] entre localisations

199.59.148.83. DNSref retourne les adresses IP : 199.59.148.11, 199.59.148.10 et 199.59.148.83. L'adresse IPref sélectionnée est alors 199.59.148.11.

6.1.4.2 Résultats

Cette section s'inscrit dans les analyses pour la définition des techniques de comparaison les plus pertinentes. Elle se décompose en deux temps : une étude multi-localisations et une étude temporelle depuis une même localisation (pour plus de détails sur les échantillons et zones de tests associées, cf. section 6.1.2).

6.1.4.2.1 Variabilité des adresses IP utilisées par les pages de login :

Etude multi-localisations : Le tableau 6.3 indique les taux de convergence des adresses IP (c.-à-d. nous retrouvons au minimum 1 adresse *IPdef* dans les adresses *IPref*) obtenus auprès des différents serveurs DNS de référence. Ceux-ci confortent et améliorent les conclusions de la première approche (cf. section 5.3.4.2). En effet, les résultats obtenus ici nous indiquent que les adresses IP utilisées par les sites de login sont fortement convergentes : les moyennes des taux de convergence, entre serveur DNS par défaut et serveurs de référence, varient de 86 à 87% (contre 81 à 82% dans la première proposition). De plus, l'écart-type (entre localisations) s'est considérablement réduit : 1.05 à 2.51%, contre 3.62 à 5.00% précédemment. Enfin, l'intervalle de confiance à 95% (c.-à-d. l'incertitude d'estimation, entre localisations) varie de 84 à 89%. Nous constatons que les 3 DNS de référence donnent toujours le même type de résultats, avec un léger avantage de convergence pour DNSAdvantage (vs. OpenDNS dans la première proposition).

Cette analyse qui inclut les 104 URLs à FQDN unique de la première proposition, précédemment testées 3 à 5 mois plus tôt sur 11 localisations, tend à indiquer une certaine stabilité temporelle des résultats.

De par les améliorations apportées dans notre seconde proposition, nous sommes désormais en mesure d'analyser les adresses IP utilisées pour récupérer PageDef et PageRef. Une deuxième analyse menée sur les adresses IP (cf. tableau 6.4) nous indique alors que 77 à 79% des FQDNs interrogés ont retourné une seule et même adresse IP (c.-à-d. lorsque la réponse du DNS défaut est comparée à la réponse du DNS de référence), avec des écart-types oscillant entre 0.97 et 2.55%. Par conséquent, ceci nous indique que dans 77 à 79% des cas, les deux pages webs (PageDef et PageRef) ont été récupérées auprès du même serveur web.

Etude multi-temporelle depuis une même localisation : Depuis une même localisation, les résultats obtenus sur une période de 3 mois (cf. tableau 6.5) nous indiquent que le taux de convergence des adresses IP associées aux pages de login sont très stables. En effet, les écart-types sont inférieurs à 1%, pour des taux de convergence moyens situés entre 86 et 87%. A noter que lors de l'analyse multi-localisations, le site géographique testé ici avait indiqué un taux de convergence moyen oscillant entre 85.80 et 86.75%, selon le serveur DNS de référence utilisé.

Les analyses sur le taux d'unicité des adresses IPdef et IPref utilisées pour récupérer les pages webs associées (cf. tableau 6.6), nous indiquent que 77 à 78% des FQDNs interrogés (avec un écart-type maximum de 1.02%) ont retourné une seule et même adresse IP. A noter que lors de l'analyse

Tableau 6.4 – Comparaison des adresses IPdef et IPref utilisées pour récupérer les 328 pages webs légitimes, sur 11 localisations géographiques

	Taux d'unicité des adresses IPdef et IPref (min ≤ moyenne ≤ max)	Écart-Type [1]	Intervalle de confiance à 95% [1]
OpenDNS	75.61% ≤ **78.08%** ≤ 84.40%	2.34%	[76.70% ; 79.46%]
GoogleDNS	76.52% ≤ **79.14%** ≤ 84.76%	2.55%	[77.63% ; 80.65%]
DNSAdvantage	75.84% ≤ **77.75%** ≤ 79.01%	0.97%	[77.18% ; 78.32%]

[1] entre localisations

Tableau 6.5 – Comparaison des réponses du DNS par défaut vs. 3 serveurs DNS de référence, pour une même localisation géographique, sur une période de 3 mois

	Taux de convergence avec les adresses IP retournées par le serveur DNS par défaut (min ≤ moyenne ≤ max)	Écart-Type [1]	Intervalle de confiance à 95% [1]
OpenDNS	85.08% ≤ **86.33%** ≤ 88.09%	0.93%	[85.75% ; 86.91%]
GoogleDNS	85.40% ≤ **86.69%** ≤ 87.77%	0.78%	[86.20% ; 87.18%]
DNSAdvantage	85.71% ≤ **86.97%** ≤ 88.79%	0.88%	[86.39% ; 87.52%]

[1] entre les différentes dates de tests

multi-localisations, cette localisation avait retourné un taux d'unicité moyen variant de 76.52 à 78.35%, selon le serveur DNS de référence utilisé.

6.1.4.2.2 Taux d'échec des requêtes DNS : Précisons tout d'abord que les requêtes DNS effectuées ne comportaient qu'un seul essai de résolution par FQDN.

Pour l'étude multi-localisations, nous obtenons des taux d'échec moyens des requêtes DNS très faibles, variant de 0.08% à 0.44% (cf. tableau 6.7). En effet, nous obtenons généralement entre 0 et 2 erreurs par site géographique (sur les 327 URLs testées). Deux exceptions sont à noter toutefois : 1/ L'une des localisations située en Île de France nous retourne jusqu'à 11 erreurs auprès du serveur DNS par défaut, mais aucune erreur auprès des serveurs de référence. 2/ La localisation située en Chine nous retourne jusqu'à 7 erreurs auprès de GoogleDNS.

Pour l'étude multi-temporelle, nous obtenons des taux d'échec moyens des requêtes DNS nuls auprès des serveurs DNS de référence, contre un taux moyen d'échec de 3.57% auprès du serveur DNS par défaut (cf. tableau 6.8). Ce taux d'échec notablement plus élevé auprès du serveur DNS par défaut (vs. les résultats de l'étude multi-localisations), s'explique par le fait que l'étude multi-temporelle se concentre sur le site géographique observé comme le plus défavorable lors de l'étude multi-localisations. En effet, sur cette localisation nous notons entre 4 et 15 erreurs auprès du serveur DNS par défaut, selon la date des tests. Nous constatons par ailleurs que ces erreurs se produisent sur des URLs différentes selon les dates à laquelle les tests sont effectués.

6.1.4.3 Synthèse IP

Au travers des résultats obtenus sur les 327 URLs testées dans cette seconde proposition, nous confirmons la staticité des adresses IP associées aux pages de login étudiées et ce, tant en terme de répartition géographique que temporelle.

Nous confirmons également que les serveurs DNS de référence interrogés donnent le même niveau de résultats en terme de convergence des adresses IPdef vs. IPref (86 à 87% en moyenne).

Parce que nous sommes désormais en mesure d'analyser l'adresse IP utilisée pour récupérer PageDef et PageRef, nous constatons également que dans une large majorité des cas (77 à 79% en moyenne), cette adresse IP est unique. Ceci semble donc confirmer que la vérification de l'adresse IP associée à une page de login est un critère pertinent pour la détection du pharming. D'autant plus que les taux d'échec

Tableau 6.6 – Comparaison des adresses IPdef et IPref utilisées pour récupérer les 328 pages webs légitimes, pour une même localisation géographique, sur une période de 3 mois

	Taux d'unicité des adresses IPdef et IPref (min ≤ moyenne ≤ max)	Écart-Type [1]	Intervalle de confiance à 95% [1]
OpenDNS	75.91% ≤ **77.29%** ≤ 78.05%	0.71%	[76.85% ; 77.73%]
GoogleDNS	76.22% ≤ **77.68%** ≤ 78.96%	1.02%	[77.05% ; 78.31%]
DNSAdvantage	76.52% ≤ **78.08%** ≤ 78.08%	1.00%	[75.90% ; 78.70%]

[1] entre les différentes dates de tests

Tableau 6.7 – Taux d'échec des requêtes DNS auprès des 4 serveurs DNS, sur 11 localisations géographiques

	Taux d'échec des requêtes DNS (min ≤ moyenne ≤ max)	Écart-Type [1]
Défaut	0% ≤ **0.44%** ≤ 3.35%	0.99%
OpenDNS	0% ≤ **0.11%** ≤ 0.61%	0.21%
GoogleDNS	0% ≤ **0.36%** ≤ 2.13%	0.67%
DNSAdvantage	0% ≤ **0.08%** ≤ 0.30%	0.14%

[1] entre localisations

des requêtes DNS observés auprès des serveurs DNS de référence sont quasi-nuls (entre 0 et 0.36%). Néanmoins, ceci peut être tempéré par des taux d'échec parfois plus élevé auprès des serveurs DNS par défaut (jusqu'à 3.57% en moyenne dans nos tests). Il reste donc indispensable d'utiliser une autre technique d'analyse, en combinaison avec la vérification de l'adresse IP, pour déterminer la légitimité d'une page web.

Enfin, il est à noter que la difficulté rencontrée dans la première approche, à savoir la réinitialisation du cache DNS Java, demeure un problème. A ce stade, nous n'avons trouvé aucune solution Java satisfaisante. Il pourra donc s'avérer nécessaire d'évaluer la portabilité de ce problème dans d'autres langages, en cas d'implémentation réelle de notre solution.

6.1.5 Analyse et comparaison du code source des pages webs

Les résultats obtenus lors de la première approche - bien qu'encourageants - nous ont amenés à conclure qu'aucune des deux méthodes évaluées (approche par caractères et par mots, appliquées au code complet) n'était suffisamment aboutie pour différencier de manière fiable (c.-à-d. sans faux-positifs et/ou faux-négatifs) les sites légitimes des sites contrefaits. En effet, la fenêtre de recouvrement entre le score maximum obtenu lors des comparaisons des sites légitimes-contrefaits, et le score minimum obtenu lors des comparaisons de sites légitimes, est de 11%.

Ces mêmes résultats laissent également entrevoir de meilleures performances avec l'approche par mots, c.-à-d. davantage d'écart entre les scores obtenus pour les comparaisons des pages légitimes-contrefaits vs. les scores obtenus pour les comparaisons de sites légitimes.

De plus, nous avons constaté que dans le cadre des comparaisons de sites légitimes, nos résultats étaient parfois impactés par des pages d'erreur récupérées en lieu et place de la page légitime réellement attendue.

Dans cette seconde proposition, nous concentrons donc nos efforts sur l'élaboration d'une méthode de détection plus aboutie et la définition d'un seuil de décision. De plus, nous cherchons à éliminer toute page d'erreur récupérée en lieu et place d'une page légitime. Enfin, nous évaluons l'efficacité de notre nouvelle méthode de récupération de PageRef, afin de voir si les taux d'échec sont diminués.

Tableau 6.8 – Taux d'échec des requêtes DNS auprès des 4 serveurs DNS, sur une même localisation géographique, durant 3 mois

	Taux d'échec des requêtes DNS (min ≤ moyenne ≤ max)	Écart-Type [1]
Défaut	0% ≤ **3.57%** ≤ 4.57%	1.19%
OpenDNS	**0%**	-
GoogleDNS	**0%**	-
DNSAdvantage	**0%**	-

[1] entre les différentes dates de tests

Les tests effectués dans cette section portent sur l'intégralité des 328 URLs sélectionnées (cf. section 6.1.2).

6.1.5.1 Méthodes étudiées : introduction de nouvelles techniques d'analyse du code source des pages webs

En complément des méthodes utilisées lors de la première approche, tant pour améliorer nos performances de détection que pour focaliser nos analyses aux zones de codes les plus pertinentes, nous nous sommes intéressés à deux nouvelles familles de pistes pour l'analyse du contenu des pages webs : 1/ application des méthodes de calcul à des sous-parties du code source complet, et 2/ analyse des balises contenues dans le code source.

En effet, à partir de la structure type d'un code source HTML rappelée en figure 6.3, nous avons scindé le code HTML des pages webs récupérées en plusieurs sous-parties. Ce découpage s'est effectué selon deux critères/hypothèses :

- La recherche d'une exhaustivité des zones analysées, par rapport au contenu global du code source récupéré. Ceci afin d'analyser la pertinence des différents éléments du code dans la différenciation d'un site légitime d'un site contrefait.
- La mise en exergue des éléments qui nous apparaissent comme les plus sensibles : les liens et les balises présents dans le code source. En effet, dans le cas d'une corruption a minima d'une page web, les premiers éléments modifiés seront les liens. Sinon, dans le cas d'une modification plus conséquente, nous supposons que l'attention portée à l'ordonnancement et/ou la présence des balises pourrait s'avérer révélateur d'une contrefaçon. En effet, toute modification de code (ajout, suppression, modification - hors substitution de taille identique) modifiera automatiquement le nombre et/ou l'emplacement des balises. A vérifier toutefois que les pages légitimes récupérées depuis plusieurs sources (c.-à-d. grâce aux informations fournies par plusieurs serveurs DNS) ne soient pas également impactées par des problèmes d'ordonnancement de balises liés à la localisation. En effet, si le serveur DNS de référence interrogé nous renvoie vers une page web hébergée dans une zone géographique très éloignée de notre point de téléchargement, nous serions peut-être confrontés à des modifications de la structure HTML, telles qu'évoquées en section 5.2.2.

Au final, nous avons donc porté notre analyse sur 7 zones de code source explicitées ci-après.

6.1.5.1.1 Application de la méthode de calcul au code source complet : Tel que vu dans la première proposition, nous avons appliqué nos méthodes de calcul par caractères et par mots à l'intégralité du *Code complet*. A noter que pour les pages légitimes, il s'agit en fait du *Code complet sans en-tête HTTP*. Comme son nom l'indique, cette sous-version *Code complet sans en-tête HTTP* contient le code source intégral, auquel a été retiré l'en-tête HTTP présent lors de la récupération de la page web effectuée par notre programme. Cette sous-version n'a donc d'intérêt que pour une application aux pages légitimes récupérées depuis les informations fournies par différents serveurs DNS. En effet, les pages contrefaites et légitimes associées ayant quant à elles été récupérées depuis un navigateur web, elles sont dépourvues de cet en-tête HTTP. Notons que cette sous-version a vu son apparition suite à la volonté d'élimination des pages d'erreur récupérées en lieu et place des pages légitimes attendues

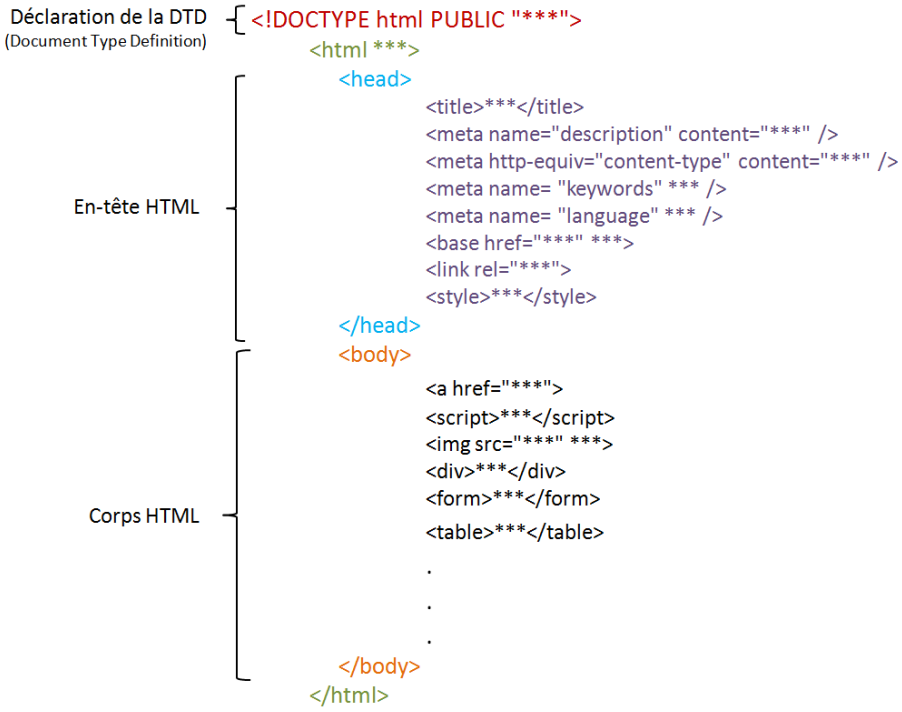


FIGURE 6.3 – Structure type du code source d'une page HTML

(pour plus de détails, cf. section 6.1.5.2).

6.1.5.1.2 Application de la méthode de calcul sur des sous-parties du code source complet : Le découpage du code source des pages webs en sous-parties s'est quant à lui intéressé à 4 zones : *Body*, *Head complet*, *Head contenu* et *Liens*.

Les sous-parties *Head complet* et *Body* correspondent respectivement à l'en-tête HTML incluant la déclaration de la DTD (Document Type Definition), et au corps de la page web. L'utilisation de la DTD, en amont du code source de la page web, n'est pas obligatoire mais elle est présente dans la quasi-totalité des pages légitimes récupérées. Située en préambule du document HTML, elle est interprétée et utilisée par le navigateur web du client, afin d'obtenir un meilleur rendu d'affichage de la page web. De par sa présence quasi-permanente et notre volonté d'exhaustivité dans les analyses menées, nous avons jugé utile de l'intégrer à nos comparaisons.

La sous-partie *Head contenu* correspond au contenu de l'en-tête HTML (incluant la DTD), hors balises. En effet, nous nous sommes aperçus que dans bon nombre de pages webs, le contenu de l'en-tête HTML contenait peu de "contenu réel" en dehors des balises et de leurs attributs.

Enfin, la sous-partie *Liens* contient les URLs relatives à un lien ou une image affichés dans la page web.

Sur chacune de ces sous-parties de code source, nous avons appliqué les deux techniques de comparaison : approches par caractères et par mots.

6.1.5.1.3 Analyse des balises contenues dans le code source complet : Cette analyse s'est portée sur l'extraction de 13 balises - considérées comme les plus pertinentes - contenues dans le code source complet.

En effet, le nombre de balises potentiellement utilisables en HTML étant tellement important, nous avons choisi de concentrer nos efforts sur celles que nous avons considéré comme les plus pertinentes, parmi une liste de plus de 90 balises. Ainsi nous avons sélectionné :

– 5 balises structurelles représentatives des paragraphes, sections, sauts de ligne, tableaux et fenêtres incorporées à la page.

- 2 balises relatives aux formulaires, typiquement utilisés pour des zones de saisie de login et mot de passe.
- 2 balises descriptives de la page qui contiennent le titre ou un descriptif de la page.
- 2 balises relatives aux scripts exécutés sur le poste client.
- 2 balises relatives aux liens et images qui ont un grand rôle dans l'aspect de la page, et/ou son appartenance à un domaine/arborescence de site.

A ces balises, nous avons appliqué deux types de calculs de scores qui réutilisent un peu le principe de l'approche par caractères : l'un de ces scores est dit par *occurrence*, tandis que l'autre est dit par *localisation*. Le principe de l'analyse par *occurrence* repose sur l'hypothèse suivante : si une partie de code est ajoutée ou supprimée, le nombre de balises est forcément impacté (à moins d'effectuer une substitution de balises). Le principe de l'analyse par *localisation* repose sur une deuxième hypothèse : si une modification est apportée à l'intérieur d'une balise (en rajoutant ou supprimant du contenu), ou de manière plus globale au code, la prise en compte du numéro de ligne où se trouve chaque balise sera forcément impacté.

Ainsi, basés sur ces hypothèses, le calcul de score par *occurrence* retourne, pour chaque balise, un score représentatif du nombre de fois où cette dernière apparaît dans le code complet :

$$\text{Score occurence balise(x)} = \sum_{i=1}^{n} présence(balise(x), i)$$

où x représente l'une des 13 balises étudiée

et n correspond au nombre de lignes du fichier étudié[1]

Sur le même principe, le calcul de score par *localisation* retourne, pour chaque balise, un score représentatif du nombre de fois où une balise apparaît, ainsi que la ligne où elle se situe dans le code complet :

$$\text{Score localisation balise(x)} = \sum_{i=1}^{n} présence(balise(x), i) \cdot i$$

où x est l'une des 13 balises étudiées,

n correspond au nombre de lignes du fichier étudié[1]

et i est le numéro de ligne où la balise a été trouvée dans le code source complet

.

6.1.5.2 Implémentation : points spécifiques

L'implémentation du GET HTTP permettant de récupérer PageRef - envoyé à destination d'une adresse IP pré-définie -, ainsi que la sélection de l'adresse IPref utilisée pour récupérer PageRef sont détaillées dans la section 6.1.4.1.

A noter que, dans le cas des pages légitimes récupérées depuis différentes localisations, les calculs de score utilisant les approches par caractères ou par mots, appliquées au code complet, ont été réalisés au téléchargement des pages webs. Tous les tests effectués sur le code modifié ou des sous-parties de celui-ci ont été réalisés après tests, selon les techniques décrites ci-après.

Les traitements réalisés sur couples de pages légitimes et couples de pages légitimes-contrefaites sont identiques.

6.1.5.2.1 Elimination des pages d'erreur et application de la méthode de calcul au code source complet : Une analyse plus détaillée des scores obtenus lors des comparaisons de pages légitimes effectuées dans la première approche, nous indiquent que bon nombre de pages délivrent un taux de similitude de l'ordre de 99.95 à 99.99%. Après investigations sur le contenu de ces pages, nous nous sommes aperçus que l'une de leurs différences majeures se résume à un horodatage. Celui-ci est présent dans l'en-tête HTTP incorporé dans le contenu de la page (cf. exemple en figure 6.4) dès lors que cette dernière est récupérée par un GET HTTP (tel qu'effectué par notre programme Java).

De plus, nous sommes confrontés à une difficulté conséquente, à savoir que certaines pages légitimes sont indisponibles pour diverses raisons (site en maintenance, page déplacée, etc.). Par conséquent, il nous arrive de récupérer des pages d'erreur en lieu et place des pages légitimes attendues. La comparaison effectuée en est alors faussée, ce qui impacte nos résultats.

1. Un exemple de fichier étudié est disponible en figure 6.7

```
HTTP/1.1 200 OK
Date: Sun, 03 Apr 2011 23:37:09 GMT
Server: Apache
Pragma: no-cache
Expires: 0
P3P: CP="NON CUR ADM DEV PSA OUR LEG STA"
Set-Cookie: sid25050000=test; path=/cgi; domain=banking.nordlb.de; secure
Cache-control: private, no-cache, no-store
Content-Length: 5038
Connection: close
Content-Type: text/html; charset=iso-8859-1
```

FIGURE 6.4 – Exemple d'en-tête HTTP trouvé en préambule du code source d'une page web légitime

TABLEAU 6.9 – Exemples de codes d'en-tête HTTP, en préambule d'une page web légitime

Code HTTP	Catégorie	Signification	Quantité d'URLs associées sur le site de Bruxelles	
200	Succès	Page récupérée avec succès	282	85.98%
301	Redirection	Page déplacée de façon permanente	1	12.20%
302		Page déplacée de façon temporaire	37	
303		La réponse à cette requête est ailleurs	2	
400	Erreur du client	La syntaxe de la requête est erronée	1	0.91%
403		Authentification refusée	2	
500	Erreur du serveur	Erreur interne du serveur	3	0.91%

Théoriquement, toute indisponibilité de page web se doit d'être proprement spécifiée dans l'en-tête HTTP via un code de connexion HTTP approprié. Des exemples de codes HTTP courants que nous avons rencontrés sont spécifiés dans le tableau 6.9. A titre d'exemple, ce tableau mentionne également la proportion d'URLs associées à ces codes, sur une des localisations testées : Bruxelles.

Pour résoudre ces problèmes, nous avons donc créé une sous-version du code complet, intitulée *Code complet sans en-tête HTTP* qui contient le code source de la page web, hors en-tête HTTP.

Pour la suite des analyses, toutes nos comparaisons de pages légitimes effectuées sur le code complet se basent désormais sur ce nouveau fichier *Code complet sans en-tête HTTP* qui exclut l'en-tête HTTP.

6.1.5.2.2 Application de la méthode de calcul sur des sous-parties du code source complet : Le découpage du code source des pages webs en 4 sous-parties s'effectue, à raison d'un fichier créé par sous-partie, grâce aux techniques suivantes :

- le *Body* est récupéré par extraction de l'intégralité du contenu placé entre les balises <BODY> et </BODY>
- Le *Head complet* est obtenu par extraction de l'intégralité de la DTD indiquée dans l'élément <!DOCTYPE>, ainsi que du contenu de la page placé entre les balises <HEAD> et </HEAD>.
- *Head contenu* est récupéré par extraction du contenu de la DTD indiqué dans l'élément <!DOCTYPE>, ainsi que du contenu (hors balises) placé entre les balises <HEAD> et </HEAD> (par ordre de balises de même type). La figure 6.5 montre un exemple de fichier *Head contenu* créé.

```
<!doctype html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="fr" dir="ltr"
class="html-ltr firefox">
<head>
<meta name="description" content="Le site de MaBanque">
<script>Monscript</script>
<meta name="keywords" content="MaBanque, argent, prêt, livret A">
<title>Page de MaBanque</title>
</head>
...
```

(a) Code source

```
html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"
xmlns="http://www.w3.org/1999/xhtml" lang="fr" dir="ltr" class="html-ltr
firefox"
Page de MaBanque
description Le site de MaBanque
keywords MaBanque, argent, prêt, livret A
Monscript
```

(b) Fichier de HEAD CONTENU créé

FIGURE 6.5 – Exemple de fichier créé pour la sous-partie HEAD CONTENU, à partir du code source complet d'une page web

– *Liens* est obtenu par extraction du contenu des balises <A ...> et <IMG>, tel qu'illustré en figure 6.6.

```
<html>
<head>
<title>Bienvenue sur MaBanque</title>
</head>
<body>
<img src="/images/Logo_MaBanque.jpg">
<br>Vous êtes ici sur la <b>page d'accueil</b>.
<br><a href="login.php">Se connecter</a>.
</body>
</html>
```

(a) Code source

```
IMG : images/Logo_MaBanque.jpg
URL: login.php
```

(b) Fichier de LIENS créé

FIGURE 6.6 – Exemple de fichier créé pour la sous-partie LIENS, à partir du code source complet d'une page web

6.1.5.2.3 Analyse des balises contenues dans le code source complet : Dans un premier temps, nous créons un fichier qui extrait l'ensemble des balises ouvrantes contenues dans le code complet, en association avec le numéro de ligne où elles apparaissent. La figure 6.7 illustre un exemple de fichier créé.

```
<html>
<head>
<title>Bienvenue sur MaBanque</title>
</head>
<body>
<img src="/images/Logo_MaBanque.jpg">
<br>Vous êtes ici sur la <b>page d'accueil</b>.
<br><a href="login.php">Se connecter</a>.
</body>
</html>
```

(a) Code source

```
1 html
2 head
3 title
5 body
6 img
7 br
7 b
8 br
8 a
```

(b) Fichier de balises créé

FIGURE 6.7 – Exemple de fichier créé pour l'analyse des balises, à partir du code source complet d'une page web

A partir de ce fichier, nous générons ensuite deux fichiers de scores (un par occurrence et un par localisation) : le premier ne tient compte que des noms de balises, tandis que le second tient également compte du numéro de ligne.

Les 13 balises précédemment sélectionnées sont identifiées par la recherche des champs suivants :

– les 5 balises structurelles : <p> pour les paragraphes, pour les sections,
 pour les sauts de ligne, <table> pour les tableaux et <iframe> pour les fenêtres incorporées à la page.
– les 2 balises relatives aux formulaires : <form> et <input>.
– les 2 balises descriptives de la page : <title> et <description>.
– les 2 balises relatives aux scripts exécutés sur le poste client : <script> et <noscript>.
– les 2 balises relatives aux liens et images : <a ...> (incluant <a href>, <a target>, etc.) et <img>.

Nous avons également envisagé d'extraire la balise <base href>, qui permet d'indiquer un chemin absolu commun aux liens de la page, mais nos analyses manuelles sur quelques dizaines de page ont montré une faible utilisation de celle-ci.

A noter que certaines balises peuvent contenir des attributs (p.ex. des indications de couleur, d'alignement ou de taille), placés entre guillemets, à l'intérieur du champ de balise. Dans ce cas, nous extrayons la balise sans ses attributs.

6.1.5.3 Résultats des analyses pour la définition des techniques de comparaisons les plus pertinentes

Cette section s'inscrit dans les analyses pour la définition des techniques de comparaisons les plus pertinentes. Elle porte sur l'application des approches par caractères et par mots aux code complet et

Tableau 6.10 – Taux de similitude des 328 pages légitimes avec l'approche par caractères appliquée au code complet, sur 11 localisations géographiques

	APPROCHE PAR CARACTÈRES sur code complet		
	Taux de similitude avec la page issue de IPdef (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]
OpenDNS	99.67% ≤ **99.83%** ≤ 99.91%	0.07%	[99.79% ; 99.87%]
GoogleDNS	99.70% ≤ **99.84%** ≤ 99.89%	0.05%	[99.81% ; 99.87%]
DNSAdvantage	99.73% ≤ **99.84%** ≤ 99.90%	0.06%	[99.81% ; 99.87%]

[1] entre localisations

Tableau 6.11 – Taux de similitude des 328 pages légitimes avec l'approche par mots appliquée au code complet, sur 11 localisations géographiques

	APPROCHE PAR MOTS sur code complet		
	Taux de similitude avec la page issue de IPdef (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]
OpenDNS	98.58% ≤ **98.86%** ≤ 99.07%	0.14%	[98.72% ; 99.00%]
GoogleDNS	98.34% ≤ **98.84%** ≤ 99.04%	0.19%	[98.65% ; 99.03%]
DNSAdvantage	98.55% ≤ **98.90%** ≤ 99.02%	0.14%	[98.76% ; 99.04%]

[1] entre localisations

sous-parties du code complet, ainsi qu'aux calculs de scores par occurrence et localisation appliqués aux balises.

6.1.5.3.1 Analyses sur le code complet : Les analyses sur le code complet portent sur les pages légitimes issues des 11 localisations, ainsi que sur celles issues d'une même localisation sur une période de 3 mois. En comparaison, elles s'intéressent également aux résultats obtenus sur les 75 couples de pages légitimes-contrefaites. Pour plus d'informations sur les échantillons et zones de tests, cf. section 6.1.2.

Taux de similitude des pages légitimes - étude multi-localisations : Les tableaux 6.10 et 6.11 indiquent les taux de similitude obtenus entre les pages PageDef et PageRef, en utilisant les deux approches par caractères et par mots. On constate que ces résultats confirment et améliorent les conclusions tirées lors de la première approche, à savoir que : le degré de convergence des pages est extrêmement élevé (entre 98 et 99% en moyenne) avec des écart-types très faibles (inférieurs à 0.20%). De plus, on peut noter que quel que soit le serveur DNS de référence interrogé pour récupérer IPref, les résultats des comparaisons sont très similaires, c.-à-d. les 3 serveurs DNS de référence sont comparables.

Ces excellents résultats peuvent notamment s'expliquer par la forte proportion d'URLs (77 à 78%, cf. section 6.1.4.2) qui ne retournent qu'une seule et même adresse IP pour le FQDN interrogé, et ce quel que soit le serveur DNS interrogé.

De plus, on peut également observer que la suppression de l'en-tête HTTP du code source complet a permis d'améliorer les résultats obtenus. Ceci est d'autant plus significatif avec l'approche par mots, dont les résultats sont désormais de niveau quasi-identique à ceux obtenus avec l'approche par caractères.

Taux de similitude des pages légitimes - étude multi-temporelle depuis une même localisation : Depuis une même localisation, les tableaux 6.12 et 6.13 nous amènent aux mêmes conclusions que celles observées lors de l'étude multi-localisations, à savoir : des degrés de convergences des PageDef et PageRef extrêmement élevés (97 à 99% en moyenne), des écart-types très faibles (inférieurs à 0.30%), et des résultats quasi-identiques quel que soit le DNS de référence utilisé. Ceci semble donc être vecteur d'une bonne stabilité des résultats dans le temps.

TABLEAU 6.12 – Taux de similitude des 328 pages légitimes avec l'approche par caractères appliquée au code complet, pour une même localisation sur une période de 3 mois

	APPROCHE PAR CARACTÈRES sur code complet		
	Taux de similitude avec la page issue de IPdef (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]
OpenDNS	99.64% ≤ **99.81%** ≤ 99.89%	0.09%	[99.75% ; 99.87%]
GoogleDNS	99.62% ≤ **99.84%** ≤ 99.90%	0.08%	[99.79% ; 99.89%]
DNSAdvantage	99.65% ≤ **99.85%** ≤ 99.91%	0.08%	[99.80% ; 99.90%]

[1] entre localisations

TABLEAU 6.13 – Taux de similitude des 328 pages légitimes avec l'approche par mots appliquée au code complet, pour une même localisation sur une période de 3 mois

	APPROCHE PAR MOTS sur code complet		
	Taux de similitude avec la page issue de IPdef (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]
OpenDNS	96.89% ≤ **97.36%** ≤ 97.60%	0.25%	[97.20% ; 97.52%]
GoogleDNS	96.88% ≤ **97.44%** ≤ 97.65%	0.29%	[97.26% ; 97.62%]
DNSAdvantage	96.88% ≤ **97.49%** ≤ 97.66%	0.24%	[97.34% ; 97.64%]

[1] entre localisations

Taux de similitude des pages légitimes-contrefaites : Concernant les résultats obtenus sur les pages légitimes-contrefaites (cf. tableau 6.14), on constate que - tel qu'aperçu lors de la première approche - la méthode par mots se démarque nettement. En effet, le taux de similitude moyen observé est plus faible avec l'approche par mots (43%, contre 76% avec l'approche par caractères), même si les valeurs minimales (0 à 10%) et maximales (96 à 99%) sont de niveau comparable pour les deux approches.

De plus, bien que les intervalles de confiance à 95% des deux approches nous retournent des valeurs maximales (50 ou 83% selon l'approche) bien en deçà des valeurs minimales obtenues sur les comparaisons de pages légitimes (96% au minimum), on peut constater que le taux de faux-négatifs reste bien trop élevé avec l'approche par caractères. En effet, un éventuel seuil de décision fixé à 90 ou 96% laisse apparaître un facteur 10 entre les taux de faux-négatifs résiduels obtenus avec les deux approches (57 vs. 5% avec un seuil décisionnel à 90%, ou 17 vs. 1% avec un seuil décisionnel à 96%).

6.1.5.3.2 Analyses sur les sous-parties du code complet : Les analyses sur les sous-parties du code complet portent exclusivement sur les 75 couples de pages légitimes-contrefaites. En effet, de par les résultats de comparaisons extrêmement élevés obtenus sur l'analyse du code complet des pages légitimes, nous avons jugé inutile de détailler plus avant les résultats de même calibre obtenus sur les sous-parties de code complet associées.

A contrario, il apparaît nettement plus intéressant de focaliser ces analyses sur les pages légitimes-contrefaites, afin d'essayer d'identifier des zones révélatrices des contrefaçons perpétrées.

Avec l'approche par caractères (cf. tableau 6.15), on peut constater que les sous-parties *Liens* et *Body* semblent être, en moyenne, les plus affectées par des changements. Néanmoins, aucune des 4 sous-parties étudiées ne se distingue vraiment. Toutes conduisent à des taux de faux-négatifs trop élevés, et ce quel que soit le seuil décisionnel envisagé (de 21 à 40% pour un seuil à 90%, et de 13 à 22% avec un seul de décision à 96%).

En étudiant les résultats obtenus avec l'approche par mots (cf. tableau 6.16), on constate également que les sous-parties *Liens* et *Body* sont, en moyenne, les plus affectées par les changements. La partie *Liens* est d'ailleurs celle qui obtient le score moyen le plus bas. Ceci peut notamment s'expliquer par

Tableau 6.14 – Taux de similitude du code complet de 75 couples de pages légitimes-contrefaites

	Taux de similitude entre pages légitimes et contrefaites (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]	Taux de faux-négatifs si le seuil de décision est 90%	96%	Approche déterminante pour % couples de sites
CARACTÈRES	10.83% ≤ **76.91%** ≤ 99.93%	27.12%	[70.78% ; 83.05%]	57.33%	17.33%	0%
MOTS	0% ≤ **43.45%** ≤ 96%	33.11%	[35.96% ; 50.95%]	5.33%	1.33%	100%

[1] entre couples de pages

Tableau 6.15 – Taux de similitude des sous-parties du code complet de 75 couples de pages légitimes-contrefaites, en utilisant l'approche par caractères

	APPROCHE PAR CARACTÈRES **sur sous-parties du code complet**					
	Taux de similitude entre pages légitimes et contrefaites (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]	Taux de faux-négatifs si le seuil de décision est 90%	96%	Approche déterminante pour % couples de sites
BODY	12.10% ≤ **68.68%** ≤ 100%	28.20%	[62.26% ; 75.11%]	26.67%	16.00%	6.67%
HEAD COMPLET	11.05% ≤ **76.57%** ≤ 100%	23.32%	[71.29% ; 81.85%]	40.00%	22.67%	16.00%
HEAD CONTENU	12.41% ≤ **72.49%** ≤ 100%	26.23%	[66.55% ; 78.43%]	34.67%	20.00%	28.00%
LIENS	0.55% ≤ **58.96%** ≤ 99.87%	33.37%	[51.36% ; 66.56%]	21.33%	13.33%	20.00%
CODE COMPLET[2]	10.83% ≤ **76.91%** ≤ 99.93%	27.12%	[70.78% ; 83.05%]	57.33%	17.33%	0%

[1] entre couples de pages
[2] ces résultats issus du tableau 6.14 sont reportés ici pour une meilleure lisibilité et comparaison des résultats

les contrefaçons de pages webs effectuées à minima - afin de rendre la détection de contrefaçon plus difficile - qui ne change que quelques liens (typiquement des zones de login) de la page originale.

De plus, on constate que la sous-partie *Liens* est celle qui conduit au taux de faux-négatifs résiduel le moins élevé : si le seuil décisionnel est établi à 90%, le taux de FNR s'avère identique à celui observé sur code complet (c.-à-d. 5.33%). Il devient alors nul si le seuil est fixé à 96%.

6.1.5.3.3 Analyses sur les balises : Les analyses sur les balises portent sur les 75 couples de pages légitimes-contrefaites, ainsi que sur les pages légitimes issues de 6 localisations : Bruxelles, Mexico, Montpellier, Samoreau, Dakar et Shenzhen.

On peut constater, sans surprise, que l'analyse des balises par occurrence et localisation tend à indiquer très peu de changements sur le nombre et l'emplacement des balises contenues dans les pages webs légitimes. En effet, les taux de similitude moyens observés sur 6 localisations varient de 98 à 100% pour des écart-types de 0 à 12% (cf. tableau 6.17 et 6.18).

Concernant l'analyse des pages légitimes et contrefaites, on constate cette fois que les résultats obtenus sont nettement plus bas (vs. ceux obtenus avec les pages légitimes) tant en terme d'occurrence que de localisation : les taux de similitude sont respectivement de 31 à 75% (hors balise <title>) et 16 à 59% (cf. tableaux 6.19 et 6.20). Néanmoins, quelle que soit la balise observée, les écart-types sont très importants : 24 à 45% (hors balise <title>) pour l'occurrence, et 38 à 43% pour la localisation. Ceci rend difficilement exploitables les résultats des balises comme seuls critères de décision.

A noter que les balises descriptives (<title> et <description>) sont non pertinentes pour l'analyse par occurrence et/ou localisation, car toujours présentes/identiques ou absentes.

6.1.5.3.4 Synthèse des méthodes les plus pertinentes : Au vu des résultats d'analyses sur l'étude des méthodes les plus pertinentes, on peut retenir que :

- L'approche par caractères, qu'elle soit appliquée au code complet ou sous-parties du code complet, ne permet pas d'aboutir à des résultats concluants. En effet, le taux de faux-négatifs résiduel potentiel demeure trop important. Cette méthode n'est donc pas retenue pour l'élaboration de notre technique de comparaison finale.
- L'approche par mots appliquée au code complet donne des résultats très intéressants mais, le taux

Tableau 6.16 – Taux de similitude des sous-parties du code complet de 75 couples de pages légitimes-contrefaites, en utilisant l'approche par mots

	APPROCHE PAR MOTS sur sous-parties du code complet					
	Taux de similitude entre pages légitimes et contrefaites (min ≤ moyenne ≤ max)	Écart-Type[1]	Intervalle de confiance à 95%[1]	Taux de faux-négatifs si le seuil de décision est 90%	96%	Approche déterminante pour % couples de sites
BODY	0% ≤ **44.35%** ≤ 97%	33.31%	[36.76% ; 51.94%]	13.33%	1.33%	93.33%
HEAD COMPLET	0% ≤ **53.33%** ≤ 100%	35.58%	[45.28% ; 61.39%]	24.00%	6.67%	84.00%
HEAD CONTENU	0% ≤ **54.40%** ≤ 100%	36.31%	[46.18% ; 62.62%]	24.00%	8.00%	72.00%
LIENS	0% ≤ **40.86%** ≤ 96.00%	31.20%	[33.76% ; 47.97%]	5.33%	-	80.00%
CODE COMPLET	0% ≤ **43.45%** ≤ 96%	33.11%	[35.96% ; 50.95%]	5.33%	1.33%	100%

[1] entre couples de pages
[2] ces résultats issus du tableau 6.14 sont reportés ici pour une meilleure lisibilité et comparaison des résultats

Tableau 6.17 – Taux de similitude par occurrence des balises de 328 pages légitimes de 6 localisations

	Balises												
	title	description	a	img	script	noscript	p	br	table	div	iframe	form	input
Moyenne[1]	99%	-	99%	99%	99%	98%	99%	99%	99%	99%	100%	99%	99%
Écart-type[1]	5%	-	6%	5%	6%	10%	7%	6%	7%	5%	0%	7%	7%

[1] entre couples de pages

de faux-négatifs résiduel - bien que faible - n'est pas nul. Cette technique ne peut donc se suffire à elle-même pour déterminer la légitimité d'une page web.
- L'analyse des sous-parties du code complet, en utilisant l'approche par mots, laisse apparaître que la sous-partie *Liens* est la plus pertinente. Elle permet en effet d'obtenir le score moyen le plus bas et le taux de FNR résiduel le plus intéressant. Une attention particulière est donc à accorder à cette sous-partie *Liens* dans l'élaboration de notre méthode de comparaison finale.
- L'analyse des balises - tant en terme d'occurrence que de localisation - nous démontre une nette différence entre les taux de similitude obtenus sur pages légitimes vs. pages légitimes-contrefaites. Néanmoins, ces résultats sont tempérés par des écart-types très élevés obtenus sur pages légitimes-contrefaites. L'intégration des scores des balises dans la méthode de comparaison finale peut donc s'avérer intéressante, mais non suffisante.
- Enfin, les taux de similitude minimum obtenus sur pages légitimes laissent entrevoir un seuil de décision maximum de 96% (cf. tableaux 6.10 à 6.13). En effet, les résultats de nos études multi-localisations et multi-temporelles sur code complet sont toutes supérieures à cette valeur. De plus, les intervalles de confiance à 95% associés sont supérieurs à 97%.

6.1.5.4 Résultats d'analyses des taux d'échec de récupération des pages Défaut et Référence :

En comparaison des résultats obtenus lors de la première approche, nous constatons que notre nouvelle technique de récupération de PageRef (c.-à-d. basée sur une URL non modifiée, envoyée à destination d'une adresse IP définie) a permis de diminuer considérablement le taux d'échec de récupération de page associée. En effet, nous observons désormais que le taux d'échec constaté lors de l'étude multi-localisation est de l'ordre de 4% en moyenne (cf. tableau 6.21), contre 22% en moyenne lors de la première approche, rejoignant ainsi les taux d'échec rencontrés pour PageDef.

A noter que pour l'étude multi-localisations, un site géographique nous retourne des taux d'échec plus élevés que la moyenne : 15% constatés au Vénézuéla. Ces échecs s'expliquent notamment par les pertes de connectivité rencontrées sur ce site.

Deux autres sites géographiques (Samoreau et Shenzhen) nous retournent des taux d'échec d'environ 6 et 7%, taux constatés tant pour PageDef que PageRef. Une étude plus poussée sur les raisons de ces échecs nous amène à deux constats : 1/ sur une même localisation, les URLs concernées sont les mêmes,

Tableau 6.18 – Taux de similitude par localisation des balises de 328 pages légitimes de 6 localisations

	Balises												
	title	description	a	img	script	noscript	p	br	table	div	iframe	form	input
Moyenne[1]	99%	-	99%	99%	99%	98%	99%	99%	99%	99%	100%	99%	99%
Écart-type[1]	7%	-	9%	8%	8%	12%	9%	8%	8%	8%	1%	9%	9%

[1] entre couples de pages

Tableau 6.19 – Taux de similitude par occurrence des balises de 75 couples de pages légitimes-contrefaites

	Balises												
	title	description	a	img	script	noscript	p	br	table	div	iframe	form	input
Moyenne[1]	100%	-	75%	74%	72%	57%	68%	61%	62%	77%	31%	85%	71%
Écart-type[1]	0%	-	29%	26%	34%	42%	36%	38%	45%	27%	46%	24%	29%

[1] entre couples de pages

que la page web soit récupérée auprès de IPdef ou de l'une des trois adresses IPref, et 2/ ces erreurs sont dues à des échecs d'échanges SSL réalisés en préambule de la récupération de la page web.

Enfin, les 8 sites géographiques restants retournent des taux d'échec inférieurs à 4%.

L'étude multi-temporelle porte sur le site géographique de Samoreau(France-IdF) qui nous a retourné des taux d'échec d'environ 6% lors de l'étude multi-localisations. On constate ici que les taux d'échec demeurent relativement stables dans le temps, autour de 6% en moyenne (cf. tableau 6.22). A nouveau, une investigation plus poussée sur ces taux d'échec indique : 1/ une constance des URLs concernées sur une même date (c.-à-d. les pages web récupérées depuis IPdef ou l'une des 3 IPref ont toutes échoué), 2/ les URLs concernées sont assez variables d'une date de test à une autre, et 3/ ces erreurs sont dues à des échecs d'échanges SSL réalisés en préambule de la récupération de la page web. A ce stade, nous n'avons pas eu le temps de revenir à une éventuelle correction de notre implémentation pour supprimer ces erreurs. De plus, le caractère assez changeant des URLs concernées d'une date à l'autre laisse perplexe.

Les résultats obtenus ici confirment donc nos hypothèses sur les causes supposées des taux d'erreur de récupération de PageRef, tels qu'observés dans la première approche. La seconde approche, telle qu'elle a été conçue, permet donc de revenir à des taux d'échec raisonnables. Toutefois, on constate que des erreurs résiduelles demeurent sur certaines localisations, erreurs engendrées par des problèmes d'établissement de la connexion SSL.

6.1.5.5 Résultats des Analyses pour l'élaboration de la méthode de comparaison finale

Au travers des conclusions tirées à l'issue de la section 6.1.5.3, nous avons donc choisi d'élaborer notre méthode de comparaison finale en tenant compte des éléments suivants :

- un seuil de décision qui ne peut excéder 96%,
- l'application de la méthode par mots au code complet,
- l'application de la méthode par mots au sous-fichier *Liens*,
- et l'utilisation des scores par occurrence et localisation associés aux balises.

A noter que nous nous sommes concentrés ici sur 5 des 13 balises étudiées précédemment, balises que nous considérons comme les plus importantes, à savoir : les 2 balises relatives aux liens et images (<a> et <img>), la balise de <script> et les 2 balises de formulaires (<form> et <input>). L'ensemble des 13 balises nous a en effet donné le même type de résultats tant sur pages légitimes-contrefaites que sur pages légitimes. Par manque de temps, nous avons concentré nos efforts sur les 5 balises les plus assujetties aux attaques.

Notons également qu'il n'y a aucun recoupement entre les pages utilisées pour l'étalonnage et la vérification des résultats obtenus ici.

Tableau 6.20 – Taux de similitude par localisation des balises de 75 couples de pages légitimes-contrefaites

	Balises												
	title	description	a	img	script	noscript	p	br	table	div	iframe	form	input
Moyenne[1]	59%	-	49%	44%	43%	31%	43%	44%	46%	51%	16%	47%	46%
Écart-type[1]	39%	-	39%	38%	37%	38%	42%	39%	43%	41%	32%	40%	38%

[1] entre couples de pages

Tableau 6.21 – Taux d'échec de récupération des pages webs légitimes en utilisant IPdef ou IPref, sur 11 localisations géographiques

URL utilisant l'adresse IP fournie par le DNS	**Taux d'échec** de récupération des pages webs (min ≤ moyenne ≤ max)	Écart-Type[1]
Défaut	1.52% ≤ **4.52%** ≤ 15.55%	4.13%
OpenDNS	1.52% ≤ **4.43%** ≤ 15.85%	4.15%
GoogleDNS	1.52% ≤ **4.68%** ≤ 15.85%	4.13%
DNSAdvantage	1.83% ≤ **4.63%** ≤ 15.55%	2.35%

[1] entre localisations

6.1.5.5.1 Étalonnage : L'étalonnage de notre méthode de comparaison finale s'est donc effectué sur 55 couples de pages légitimes-contrefaites et les pages légitimes issues de Bruxelles(Belgique) / couples de pages DNSdef et GoogleDNS (pour plus de détails, cf. section 6.1.2).

Après analyse des scores intermédiaires (sur balises par occurrence, balises par localisation, code complet et *Liens*) obtenus tant sur pages légitimes que sur pages légitimes-contrefaites, nous nous sommes aperçus que l'application de la méthode par mots sur la sous-partie *Liens* pouvait s'avérer fortement intéressante sur pages légitimes-contrefaites, tandis qu'elle pouvait s'avérait parfois pénalisante sur pages légitimes. En effet, en dehors du problème d'horodatage des pages éliminé précédemment, l'essentiel des divergences entre pages légitimes résident dans des modifications de liens, utilisées pour l'affichage de contenus dynamiques tels que des images.

De ce constat, nous avons donc introduit 3 méthodes de calcul :

1. le score final est obtenu en accordant un poids identique aux : balises par occurrence, balises par localisation et code complet. Cette méthode sera nommée **(B) : Balises + Code complet**.
2. le score final est calculé en accordant un poids identique aux balises vs. le code complet. Ainsi le score est déterminé à partir de : balises par occurrence, balises par localisation et (2 × code complet). Cette méthode sera nommée **(C) : Balises + Code complet × 2**.
3. le score final est obtenu en accordant un poids identique aux : balises par occurrence, balises par localisation, *Liens* et code complet. Cette méthode sera nommée **(D) : Balises + Liens + Code complet**.

Précisons qu'à des fins de comparaisons, les 3 méthodes sont systématiquement comparées entre elles, ainsi qu'à la méthode par mots appliquée au code complet seul. Cette dernière est nommée **(A) : Code complet**.

A noter que dans les tableaux qui suivent, les meilleurs résultats (c.-à-d. les plus intéressants pour une décision de légitimité des pages) apparaissent en gras afin d'améliorer leur visibilité.

Pages légitimes - Bruxelles : Le tableau 6.23 sur pages légitimes indique clairement que la méthode **(B) : Balises + Code complet** est la plus intéressante sur pages légitimes. En effet, c'est la méthode qui délivre le taux de similitude moyen le plus élevé (99.61%), l'écart-type le plus faible (0.92%) et le taux de similitude minimum le plus élevé (90.96%).

Pages légitimes-contrefaites - 55 couples de pages : Le tableau 6.24 sur pages légitimes-contrefaites délivre un message plus complexe. En effet, le taux de similitude moyen le plus faible (44.67%) est

Tableau 6.22 – Taux d'échec de récupération des pages webs légitimes en utilisant IPdef ou IPref, pour une même localisation sur une période de 3 mois

URL utilisant l'adresse IP fournie par le DNS	**Taux d'échec** de récupération des pages webs (min ≤ moyenne ≤ max)	Écart-Type[1]
Défaut	5.49% ≤ **6.52%** ≤ 7.32%	0.60%
OpenDNS	5.49% ≤ **6.28%** ≤ 7.01%	0.56%
GoogleDNS	5.79% ≤ **6.55%** ≤ 7.93%	0.66%
DNSAdvantage	5.49% ≤ **6.28%** ≤ 7.01%	0.46%

[1] entre localisations

Tableau 6.23 – Résultats d'étalonnage des méthodes de comparaisons finales sur 328 pages légitimes récupérées à Bruxelles

	Taux de similitude Moyen	Écart-Type[1]	Taux de similitude Minimum	Quantité de Faux-positif[2]
(A) : Code complet	99.02%	2.08%	74.00%	2
(B) : Balises + Code complet	**99.61%**	**0.92%**	**90.96%**	**0**
(C) : Balises + Code complet × 2	99.46%	1.17%	86.72%	1
(D) : Balises + Liens + Code complet	99.12%	2.27%	82.72%	4

[1] entre couples de pages
[2] pour un seuil de décision à 90%

obtenu avec l'approche **(A) : Code complet**, tandis que l'écart-type le plus faible (20.83%) et le taux de similitude maximum le plus faible (87.24%) sont obtenus avec l'approche **(D) : Balises + Liens + Code complet**. Par ailleurs, en comparaison avec l'étalonnage réalisé sur pages légitimes, la seule méthode qui ne présente aucun espace de recouvrement est la méthode **(B) : Balises + Code complet** : le taux maximum obtenu sur pages légitimes-contrefaites est de 89.37%, tandis que le taux minimum sur pages légitimes est de 90.96%. Il apparaît donc qu'un seuil de décision fixé à 90% semble le meilleur choix. En effet, il permettrait dans les deux cas d'avoir des taux de faux-négatifs (c.-à-d. pages contrefaites déclarées légitimes à tort) et de faux-positifs (c.-à-d. pages légitimes déclarées contrefaites à tort) nuls.

On peut toutefois remarquer que l'approche **(C) : Balises + Code complet × 2** délivre des résultats relativement proches de la méthode choisie.

Il est également notable que les 3 approches proposées délivrent de bien meilleurs résultats que la seule approche par mots appliquée au code complet.

Au final, suite à cet étalonnage, nous retenons la méthode **(B) : Balises + Code complet**, associée à un seuil de décision de **90%** :

$$TS_{final} = \text{Moyenne}\ (TS_{Balises\_occurence},\ TS_{Balises\_localisation},\ TS_{Code\_complet})$$

où TS est le taux de similitude

6.1.5.5.2 Test de l'efficacité de la méthode de comparaison retenue : Les tests sur l'efficacité de notre méthode de comparaison finale se sont effectués sur 58 couples de pages légitimes-contrefaites et les pages légitimes issues de 5 localisations : Mexico, Montpellier, Samoreau, Dakar et Shenzhen, récupérées à partir des adresses retournées par différents serveurs DNSdef et DNSref (pour plus de détails, cf. section 6.1.2).

Tableau 6.24 – Résultats d'étalonnage des méthodes de comparaisons finales sur 55 couples de pages légitimes-contrefaites

	Taux de similitude Moyen	Écart-Type[1]	Taux de similitude Maximum	Quantité de Faux-négatif[2]	Espace de recouvrement[3]
(A) : Code complet	**44.67%**	32.49%	96%	2	22%
(B) : Balises + Code complet	59.00%	21.64%	89.37%	**0**	-
(C) : Balises + Code complet × 2	55.42%	23.46%	87.53%	**0**	1%
(D) : Balises + Liens + Code complet	54.57%	**20.83%**	**87.24%**	4	5%

[1] entre couples de pages
[2] pour un seuil de décision à 90%
[3] avec les résultats sur pages légitimes, cf tableau 6.23

Pages légitimes - étude multi-localisations : Le tableau 6.25 sur pages légitimes issues de 5 localisations indique très clairement que l'approche **(B) : Balises + Code complet** délivre les meilleurs résultats tant en terme de taux de similitude moyen, que de faux-positifs, et ce quelle que soit la localisation. Concernant les écart-types et le taux de similitude minimum, cette même approche délivre les meilleurs résultats sur, respectivement, 4 et 3 localisations.

Nous nous sommes alors intéressés aux faux-positifs afin de comprendre les raisons de ces mauvaises détections. Nous identifions alors 2 pages qui posent problème :

– la page du site `twitter.com` qui contient une grande part de contenu dynamique, telles que des images publicitaires. On peut d'ailleurs voir un aperçu de cette page web où apparaît un bandeau d'images dynamiques sur la figure 6.8. Ainsi, bien que les balises soient inchangées, beaucoup de liens sont partiellement modifiés ce qui impacte à la baisse le score du code complet (et donc le score global de la méthode). Précisons toutefois que la méthode de calcul finale retenue délivre le score le plus élevé (88.62%) pour cette page, par rapport aux 3 autres méthodes. Cette page correspond à 1 faux-positif observé sur 3 localisations : Montpellier, Dakar et Mexico.

Figure 6.8 – Page de login `twitter.com` incluant un bandeau d'images dynamiques

– la page du site `www.myspace.com` pose également problème sur la localisation de Mexico. Nous constatons en effet que la PageDef récupérée est écrite en espagnol, tandis que la PageRef récupérée est écrite en anglais. Ainsi, bien que les occurrences de balises soient inchangées, le texte contenu est très différent. Ceci impacte à la baisse à la fois le score obtenu pour le code complet et la localisation des balises. Précisons toutefois que la méthode de calcul finale retenue délivre le score le plus élevé (71.39%) pour cette page, par rapport aux 3 autres méthodes.

Pages légitimes-contrefaites - 58 couples de pages : Le tableau 6.26 sur pages légitimes-contrefaites délivre le même type de message que lors de l'étalonnage. A savoir que les meilleurs résultats

Tableau 6.25 – Résultats des méthodes de comparaisons finales sur 328 pages légitimes de 5 localisations

	Localisation	Taux de similitude Moyen	Écart-Type[1]	Taux de similitude Minimum	Quantité de Faux-positif[2]
A	Mexico	98.66%	4.85%	33.00%	4
	Montpellier	98.80%	3.84%	43.00%	3
	Samoreau	98.89%	2.14%	74.00%	2
	Dakar	98.98%	2.28%	71.00%	2
	Shenzhen	98.94%	2.21%	73.00%	2
				Total	*13*
B	Mexico	**99.47%**	2.23%	66.33%	2
	Montpellier	**99.54%**	**1.36%**	82.51%	1
	Samoreau	**99.57%**	**1.01%**	**90.90%**	0
	Dakar	**99.59%**	**0.99%**	**89.96%**	1
	Shenzhen	**99.58%**	**0.89%**	**91.00%**	0
				Total	***4***
C	Mexico	99.36%	**1.91%**	**74.50%**	2
	Montpellier	99.41%	1.41%	**85.38%**	1
	Samoreau	99.43%	1.25%	86.67%	1
	Dakar	99.44%	1.28%	85.22%	1
	Shenzhen	99.42%	1.19%	86.50%	1
				Total	*6*
D	Mexico	98.89%	3.26%	66.33%	6
	Montpellier	99.04%	2.39%	85.25%	5
	Samoreau	99.03%	2.52%	83.67%	5
	Dakar	99.06%	2.51%	81.47%	5
	Shenzhen	99.07%	2.32%	82.25%	5
				Total	*26*

[1] entre couples de pages
[2] pour un seuil de décision à 90%

seraient obtenus avec l'approche **(D) : Balises + Liens + Code complet**. Toutefois, tel que vu précédemment, cette méthode délivre des résultats catastrophiques sur les pages légitimes en produisant 26 faux-positifs. Il apparaît alors ici que la méthode choisie **(B) : Balises + Code complet** ne délivre que 3 sites faux-négatifs.

Nous nous sommes intéressés à ces faux-négatifs afin de comprendre les raisons des mauvaises détections. Nous identifions alors 3 pages qui posent problème :

- les pages des sites `westernunion` et `scotiabank` présentent le même cas de figure. En effet, le chemin de base des URLs est modifié en début de page web puis, toutes les URLs suivantes sont modifiées en chemins relatifs (alors que les pages légitimes utilisent des chemins absolus). Ainsi, de par notre élimination des *Liens* dans la méthode de comparaison finale, ces changements sont moins visibles. Toutefois, ils apparaissent très nettement dans le score obtenu sur le code complet.
- la page du site `rbc` présente un autre cas de figure, totalement invisible pour l'utilisateur si utilisé dans une attaque de pharming (cf. figure 6.9). On constate en effet que la (quasi-)seule modification de cette page - par rapport à son pendant légitime -, est l'inclusion d'un script (20 lignes ajoutées par rapport à une page de 502 lignes, cf. figure 6.10) utilisé dans le cadre d'une redirection des zones de saisie des champs login et mot de passe. Ainsi, les modifications apportées sont uniquement visibles dans le score des balises par localisation, ce qui affecte insuffisamment le score final avec la méthode retenue.

Pour compléter cette analyse des faux-négatifs, la section 6.1.5.7 s'intéresse à étudier la quantité de modifications qu'il faut apporter à une page web, afin que celles-ci soient détectées par notre méthode de calcul finale.

6.1.5.6 Problèmes rencontrés

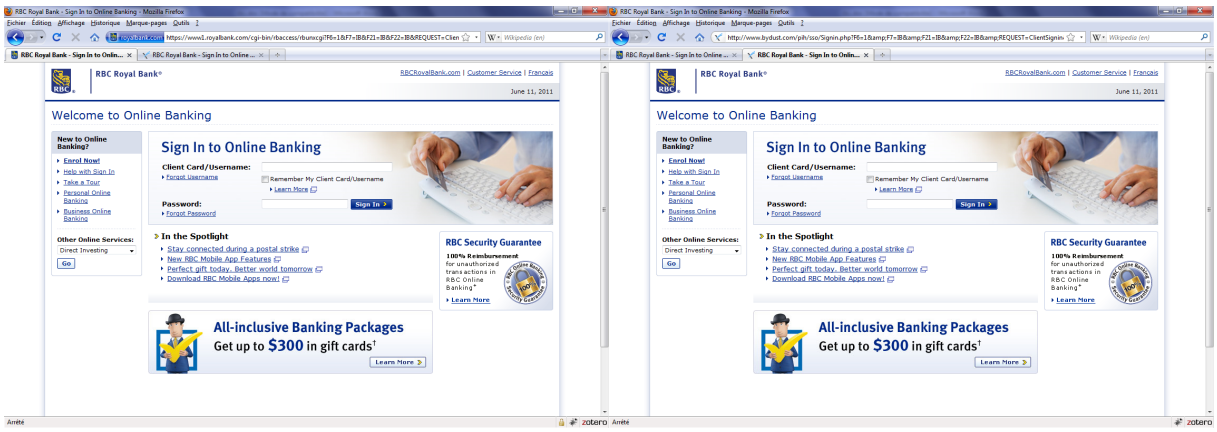
6.1.5.6.1 Elimination des pages d'erreur : La technique d'élimination des pages d'erreur, telle qu'expliquée en section 6.1.5.2 n'est pas infaillible. En effet, il nous est arrivé de rencontrer des pages d'erreur improprement codées dans l'en-tête HTTP. Par exemple, alors que l'en-tête HTTP indiquait un code "200

Tableau 6.26 – Résultats des méthodes de comparaisons finales sur 58 couples de pages légitimes-contrefaites

	Taux de similitude Moyen	Écart-Type[1]	Taux de similitude Maximum	Quantité de Faux-négatif[2]
(A) : Code complet	**47.21%**	33.10%	96.00%	4
(B) : Balises + Code complet	52.47%	27.13%	94.03%	**3**
(C) : Balises + Code complet × 2	51.15%	27.77%	93.01%	**3**
(D) : Balises + Liens + Code complet	49.58%	**25.87%**	**92.51%**	1

[1] entre couples de pages
[2] pour un seuil de décision à 90%

OK", le contenu de la page faisait mention d'une indisponibilité de site. Par conséquent, dès lors que nous obtenions des résultats de comparaisons de pages légitimes surprenants, nous nous sommes assurés à chaque fois manuellement qu'aucune des deux pages récupérées ne correspondait pas à ce cas de figure. Si tel était le cas, le couple de page était éliminé des résultats d'analyses.



(a) Site légitime (b) Site contrefait

Figure 6.9 – Captures d'écran du site légitime RBC (`https://https://www1.royalbank.com/cgi-bin/rbaccess/...`) et d'une contrefaçon (`http://www.bydust.com/pih/sso/Signin.php?...`), récupérées le 11 Juin 2011

```
<script language="JavaScript" type="text/javascript">
<!--
  function checkFields()
            {
            if (document.rbunxcgi.K1.value == '')
            {
                alert ( 'Please Enter Your Client Card/Username' );
    document.rbunxcgi.K1.focus();
    return false;
            }

            if (document.rbunxcgi.Q1.value == '')
            {
              alert ( 'Please Enter Your Password' );
    document.rbunxcgi.Q1.focus();
              return false;
            }

            }
          //-->
</script>
```

Figure 6.10 – Script utilisé dans le cadre d'une redirection des champs login et mot de passe, ajouté à la page légitime RBC

6.1.5.6.2 Création des sous-parties du code complet et analyse des balises : Une difficulté associée à notre technique d'extraction des balises est la présence d'erreurs syntaxiques au sein du code source (p.ex. utilisation de caractères spéciaux inappropriés, balises mal fermées, etc.). Ces types d'erreurs ont été rencontrés aussi bien dans les couples de pages légitimes que dans les couples de pages légitimes-contrefaites. Dans ce dernier cas, les couples de pages posant problèmes ont été éliminés et remplacés par d'autres (car ils nécessitaient trop de corrections manuelles). Dans le cas des couples de pages légitimes, nous avons été amenés à corriger manuellement 3 couples de pages qui comportaient des erreurs (p.ex. des apostrophes remplacées par des guillemets, des "http" manquants en amont d'une URL, des balises mal orthographiées, etc.). A noter que les corrections manuelles apportées ne faussent aucunement les résultats de comparaison obtenus, puisque nous avons rencontré (et corrigé) exactement les mêmes erreurs dans les deux pages : PageDef et PageRef.

Précisons également que ces erreurs ne posent aucun problème aux navigateurs webs actuels qui savent s'adapter et délivrer un affichage correct de la page. A contrario, notre programme développé pour l'analyse des balises n'est actuellement pas capable de faire cette correction automatique. Par conséquent, à chaque erreur de syntaxe dans le nom d'une balise ou à chaque oubli de balise de fermeture (ces erreurs sont d'autant plus fréquentes dans les pages contrefaites), le programme de création des sous-fichiers échoue et nous ne pouvons effectuer le calcul de score global. Notre implémentation se devra donc d'être perfectionnée afin de ne pas être mise en échec par un attaquant qui trufferait volontairement sa page d'erreurs syntaxiques, en vue de mettre en défaut notre programme d'analyse.

6.1.5.6.3 Phase d'établissement de la connexion sécurisée : A ce stade, nous conservons quelques cas d'échec de récupération des pages webs PageDef et/ou PageRef associés à des échecs d'établissement de la connexion sécurisée SSL/TLS (cf. section 6.1.5.4). Nous n'avons toutefois pas eu le temps de revenir à notre implémentation pour corriger ces erreurs.

6.1.5.7 Synthèse HTML

6.1.5.7.1 Résultats obtenus : Les résultats sur l'analyse et la comparaison de pages webs obtenus dans cette seconde proposition confirment et améliorent les résultats de la proposition précédente.

En outre, ils attestent que l'approche par mots est plus appropriée que l'approche par caractères pour l'analyse des codes sources des pages webs.

Les nouvelles techniques de comparaisons développées ici, ainsi que les améliorations apportées par l'élimination de l'en-tête HTTP (qui introduisait une part variable sur les pages légitimes), permettent d'améliorer les résultats de comparaisons. Elles permettent également d'aboutir à une méthode de comparaison basée sur les analyses de balises par occurrence et par localisation, associées à l'analyse du code complet. Elles révèlent également un intérêt pour une analyse spécifique des *Liens* inclus dans les pages webs, mais celle-ci s'avère difficilement exploitable sur pages légitimes. Enfin, elles permettent d'aboutir à un seuil de décision de 90%.

Il est important de souligner que l'ensemble de ces résultats doivent être tempérés par la quantité de pages testées ainsi que les faux-positifs et/ou faux-négatifs résiduels. En effet, bien que les résultats obtenus lors des deux approches soient convergents, ils nécessitent d'être confirmés sur davantage d'URLs. De plus, la méthode de calcul finale retenue délivre quelques résultats erronés. Elle nécessite donc des analyses plus poussées afin d'être améliorée.

Un des verrous majeurs à l'issue de la première proposition concernait un taux d'échec trop élevé pour la récupération de PageRef. Ce problème a été résolu dans la seconde proposition, ce qui confirme les causes probables d'échec supposées précédemment. Notons toutefois que quelques erreurs subsistent concernant la phase d'établissement de la connexion SSL/TLS.

Il est également important de souligner que la nouvelle technique de récupération des pages webs développée ici, permet désormais d'identifier et de distinguer clairement les adresses IPdef et IPref utilisées.

6.1.5.7.2 Seuil de détection des modifications : De par les résultats obtenus précédemment ainsi que les analyses portées sur les cas de faux-positifs et faux-négatifs résiduels, nous nous sommes intéressés à déterminer la part minimale de page web devant être modifiée pour aboutir à une détection

Tableau 6.27 – Estimation des seuils de détection des modifications apportées au code source, d'après l'exemple de la page web Paypal

	Seuil de détection[1] des modifications, exprimé[2] en MOTS	en CARACTÈRES	Score méthode finale	Scores intermédiaires Balises occurrence	Balises localisation	Code complet	Equivalence script RBC[3]
Ajout - début code	+2.95 %	+1.52%	89.41%	99.09%	72.15%	97%	0.75
Ajout - fin code	+17,49%	+9.61%	89.99%	96.15%	91.81%	82%	4
Modification	29.90%	1.28%	89.67%	100%	100%	69%	N/A
Suppression - début code	−4.78%	−2.36%	89.70%	100%	74.09%	95%	0.5
Suppression - fin code	−47.70%	−11.86%	89.37%	92.22%	87.88%	88%	2.5

[1] pour obtenir un taux de similitude final < 90% avec la méthode retenue
[2] exprimé en % par rapport à la taille du code légitime
[3] exprimé en quantité de scripts de 20 lignes, cf. figure 6.10

par la méthode de comparaison finale retenue (c.-à-d. le score obtenu doit être inférieur au seuil de décision de 90%).

Ainsi, nous avons pris en exemple le code source de la page légitime Paypal `https://www.paypal.com` constitué de[1] : 22485 caractères, 109 lignes de codes et 983 mots. A savoir que la notion de "mots" indiquée ici correspond au découpage effectué par notre approche par mots. A ne pas confondre avec la notion de mots telle qu'interprétée par l'esprit humain (cf. section 5.2.2.4.2).

A partir de la page source légitime récupérée, nous avons créé 5 pages contrefaites, incluant l'un des 5 changements suivants : *modifications* en cours de code, *ajout* de code (en *début* ou *fin de page*), *suppression* de code (en *début* ou *fin de page*).

Concernant les *ajouts* et les *suppressions* de code, nous nous sommes placés dans 2 cas : le plus favorable (changements en début de page) et le plus défavorable (changements en fin de page). En effet, des ajouts et/ou suppressions réalisés en début de code induiront automatiquement des modifications majeures sur les localisations des balises. A contrario, des ajouts et/ou suppressions effectués en fin de code auront un impact moindre sur les balises (impact variable selon le type de modifications apportées), laissant ainsi tout le poids de la détection à l'analyse du code complet.

L'*ajout* a consisté à incorporer un script utilisé à des fins malveillantes, autant de fois que nécessaire, jusqu'à aboutir à une détection. Pour ce faire, nous avons utilisé un script à notre disposition : celui rencontré dans la page contrefaite RBC (cf. figure 6.10).

Les *modifications* ont été réalisées grâce au rajout d'un caractère par mot.

Les changements apportés en *début de page* ont été effectués dès les premières lignes de code, juste après la déclaration de la DTD.

Enfin, les changements apportés en *fin de page* ont été réalisés en partance des dernières lignes de code.

En conséquence, le tableau 6.27 délivre les seuils de détection résultant de ces 5 comparaisons de pages légitimes-contrefaites. Pour l'*ajout*, nous aboutissons à une détection dès 1.52 à 9.61% de caractères (ou 2.95 à 17.49% de mots) insérés, fonction de la zone où sont réalisés les changements. Pour les *modifications*, la détection se fait dès 1.28% de caractères (ou 29.90% de mots) modifiés. Enfin, la *suppression* est détectée dès 2.36 à 11.86% de caractères (ou 4.78 à 47.70% de mots) ôtés, selon la zone impactée.

A noter que les estimations indiquées ici sont toutes relatives. En effet, dans le code ajouté/modifié/supprimé, tout est fonction de la proportion de balises vs. contenu. Plus la part de balises sera importante, plus le seuil de détection sera abaissé, et réciproquement. Enfin, de véritables pages contrefaites utilisent rarement un seul type d'altération de contenu. Elles mixent généralement ajouts et/ou modification et/ou suppression. Les chiffres indiqués ici sont donc à interpréter comme des valeurs extrêmes de détection.

1. en date du 28 Juillet 2011.

Tableau 6.28 – Temps de traitement moyens relevés sur 11 localisations, portant sur 328 URLs de login légitimes

	Temps de traitement moyen par URL (min ≤ moyenne ≤ max)	Écart-type[1]
Défaut	2 sec. ≤ **4 sec.** ≤ 7 sec.	2 sec.
OpenDNS	1 sec. ≤ **3 sec.** ≤ 6 sec.	2 sec.
GoogleDNS	2 sec. ≤ **3 sec.** ≤ 5 sec.	1 sec.
DNSAdvantage	1 sec. ≤ **5 sec.** ≤ 12 sec.	4 sec.
Moyenne globale	***3.5 sec.***	

[1] entre localisations

6.1.6 Temps de traitement

La dernière partie de notre analyse a consisté à estimer le temps de traitement associé à notre seconde approche, c.-à-d. le temps nécessaire pour la récupération des pages et les calculs de scores (utilisant les approches par caractères et par mots). Le tableau 6.28 donne une estimation moyenne des temps de traitement relevés sur 11 localisations, pour la récupération et les calculs de score des pages légitimes.

On peut constater que le temps de traitement moyen relevé par page est de 3.5 sec. en moyenne, toutes localisations confondues. Ceci semble présager d'un temps de traitement "raisonnable" en environnement utilisateur.

Néanmoins des études plus poussées devront être menées sur le temps de traitement associé à la méthode finale retenue. A noter toutefois que les zones de traitement les plus consommatrices de temps sont celles qui concernent la récupération des pages.

6.2 Limitations

Dans cette section, nous nous intéressons aux limitations et verrous techniques demeurant à l'issue de notre étude. En effet, bien que les résultats obtenus dans les sections 5.3 et 6.1 soient globalement favorables, de nombreuses questions et/ou problématiques demeurent en suspens.

6.2.1 Vérification de l'adresse IP du domaine visité

Filtrage des requêtes DNS : Lors des tests réalisés sur pages légitimes, nous avons rencontré des problèmes liés à la vérification de l'adresse IP sur 3 localisations (c.-à-d. Japon et France(Bretagne) dans la première approche, et Australie dans la seconde approche). Les données résultantes, inexploitables, n'ont donc pas été incluses dans les résultats de tests énoncés précédemment. Sur ces 3 localisations, nous avons été confrontés à un filtrage des requêtes DNS. En effet, seules étaient autorisées les requêtes DNS à destination du serveur DNS par défaut (tel que préconisé dans certaines études [SRM07]). A noter que 2 des 3 localisations concernées ont effectué les tests depuis un réseau d'entreprise, tandis que la troisième a effectué les tests depuis un réseau public (c.-à-d. un hôtel). La mise en œuvre de filtrage des requêtes DNS peut donc s'avérer bloquante pour l'utilisation de notre proposition. Néanmoins, à ce jour nous n'avons pas été confrontés à ce problème dans des tests effectués depuis des réseaux personnels.

Établissement de la connexion sécurisée : Tel qu'énoncé dans nos résultats de tests, nous avons observé des erreurs résiduelles lors des échanges SSL, échanges indispensables à la récupération des pages de login. Bien que nous ayons implémenté une solution visant à accepter tous les certificats proposés par les serveurs webs interrogés, quelques erreurs SSL demeurent. Néanmoins, parce que l'implémentation est fortement dépendante du langage utilisé et que les erreurs constatées sont parfois changeantes, nous n'avons pas encore exploré plus avant cette problématique. Elle devra cependant être étudiée avec attention en cas d'implémentation réelle.

6.2.2 Analyse et comparaison du code source des pages webs

Identification des pages de login : Un des postulats de notre proposition est de s'adresser aux pages de login exclusivement. En effet, les résultats obtenus sur pages webs au sens large sont si peu satisfaisants que nous avons choisi de nous concentrer sur les pages de login, principales cibles des attaques de phishing/pharming. Ce postulat induit automatiquement une nouvelle problématique, à savoir : lors de la navigation web de l'utilisateur, comment savoir que l'analyse doit être effectuée. Autrement dit, comment savoir que l'utilisateur consulte une page de login. Une réponse simple semblerait être : dès lors que la connexion s'effectue en mode sécurisé (c.-à-d. en HTTPS), l'analyse doit être effectuée. Néanmoins, certaines pages de login s'affichent en HTTP dans le navigateur client (p.ex. Facebook), même s'il y a bien utilisation d'une connexion sécurisée (de manière totalement transparente pour l'utilisateur) lors de l'envoi des login et mot de passe. Ces pages seraient donc exclues, à tort, de nos analyses. Une autre solution pourrait consister à faire rechercher les zones de login (p.ex. via la recherche de balises de type <form>, <input>) à notre moteur de détection avant son exécution. Néanmoins, là aussi certains sites seront exclus de la comparaison. En effet, si la zone de login apparaît dans une nouvelle fenêtre ouverte spécifiquement à cet effet, le code source de la page web ne contiendra qu'un lien hypertexte servant à la redirection.

Redirection de contenu ou de site web : Le point précédent introduit une des limitations de notre approche. En effet, notre comparaison de pages webs s'intéresse exclusivement au code source des pages principales récupérées. Autrement dit, si la page principale est décomposée en sous-fenêtres, nous ne comparons que les liens hypertexte conduisant à ces sous-fenêtres, sans analyser leurs contenus réels. Ceci peut être considéré comme une vulnérabilité de notre approche.

Lors d'une éventuelle implémentation de notre proposition, une précaution particulière devra être prise concernant le choix du moment où s'exécute notre moteur de détection. En effet, tel qu'évoqué dans le Chapitre 3, certaines URLs de login procèdent à une redirection automatique. Par conséquent, si notre moteur de détection est lancé trop tôt, la comparaison de pages ne s'exécutera pas sur la page réellement visitée par l'utilisateur.

Authentification du site web visité : Il est important de souligner que notre approche ne vise en aucun cas à assurer l'authentification du site web visité. Elle ne se substitue donc aucunement à l'utilisation des techniques d'authentification existantes, ni à la nécessité de vigilance de l'utilisateur sur la validité du certificat utilisé par le site de login visité.

Il est donc tout à fait possible que 2 pages visitées amènent à des scores identiques ou très similaires - particulièrement avec l'analyse des balises ou l'approche par caractères -, alors que celles-ci sont totalement décorrélées. Ce point pourrait donc conduire à une nouvelle vulnérabilité de notre proposition, à savoir : le taux de similitude des pages comparées se révèle élevé, alors que les pages comparées n'ont aucun rapport. Néanmoins ceci peut être tempéré par le fait que le but premier des attaquants est de leurrer un maximum d'utilisateurs, avec des imitations visuelles (quasi-)parfaites des pages légitimes usurpées. On peut donc estimer que l'utilisateur sera à même d'identifier un affichage de page qui ne correspondrait pas à ses attentes.

Pages d'erreur : Tel que nous l'avons détaillé en section 6.1.5.5, les pages d'erreur improprement codées au niveau de l'en-tête HTTP conduiront automatiquement à des taux de similitude des pages comparées très faibles. Si cela concerne PageDef, il ne peut s'agir d'un réel problème puisque l'utilisateur sera informé de l'indisponibilité du site via la page affichée dans son navigateur. Toutefois, si cela concerne PageRef, notre moteur de détection délivrera une décision erronée.

Pages de login avec contenu dynamique : Une page de login présentant une forte proportion de contenu dynamique pose problème. En effet, des liens hypertexte renouvelés trop fréquemment - ou à différents moments selon la localisation -, peuvent conduire à des faux-positifs. La page du site `twitter.com` en est un exemple flagrant (pour plus de détails, cf. section 6.1.5.5). Par conséquent, plus les pages de login incluront du contenu dynamique, moins notre moteur de détection sera efficace. A ce jour, on peut cependant remarquer que seul le cas de l'URL `twitter.com` a réellement présenté ce problème parmi 328 URLs testées. A contrario, on peut noter que la page `login.yahoo.com` qui contient

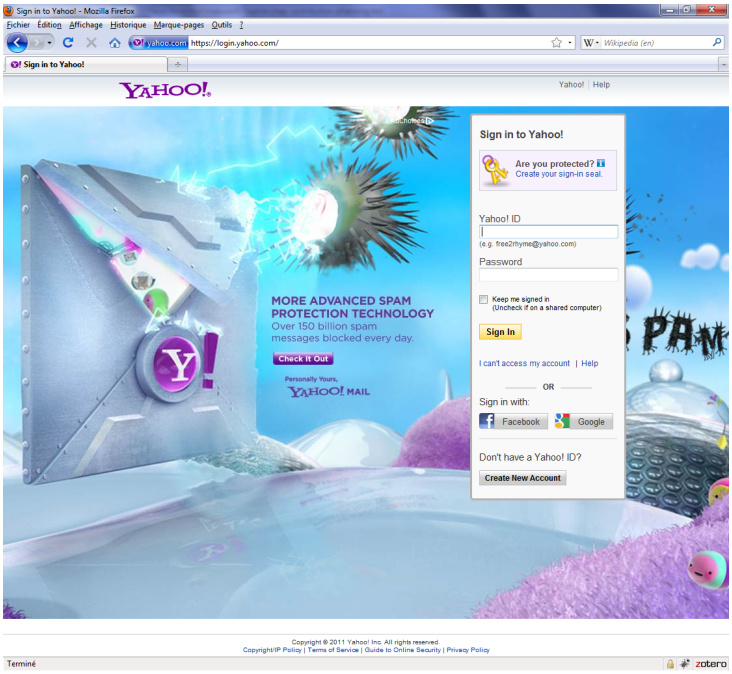


FIGURE 6.11 – Page `login.yahoo.com` incluant du contenu dynamique

elle-aussi du contenu dynamique (via une image de fond, cf. figure 6.11) retourne des taux de similitude très bons, de l'ordre de 99%.

Localisation du DNSref vs. l'utilisateur : Enfin, une autre limitation associée au calcul de score des pages webs peut résider dans les divergences de localisations qui hébergent PageDef et PageRef. De par l'interrogation de 2 serveurs DNS différents (DNSdef et DNSref), nous ne pouvons garantir d'aboutir à des serveurs webs géographiquement proches pour la récupération des 2 pages webs.

Parce que le langage de la page affichée dans le navigateur est automatiquement adapté selon les préférences de l'utilisateur, il est nécessaire que la page de référence le soit également. Nous avons vu que dans nos tests un site géographique a posé problème (cf. section 6.1.5.5, cas de l'URL `www.myspace.com` sur la localisation de Mexico). Toutefois, dès lors que notre approche sera intégrée dans le navigateur client, les 2 pages demandées (PageDef et PageRef) devraient toutes deux bénéficier des mêmes indications de préférence.

6.2.3 Intégration de l'approche dans le navigateur client

L'intégration de notre proposition dans le navigateur client de l'Internaute n'a pas encore été réalisée à ce jour. Pour rappel, notre solution vise à s'intégrer sous la forme :

- d'un indicateur visuel très simple et binaire indiquant le niveau de confiance envers la page web visitée,
- et d'une notification active effectuée via l'affichage d'un message d'alerte de type pop-up, dès lors que le site visité semble suspicieux.

Le choix d'un indicateur visuel très simple et d'un message de notification actif sont basés sur les conclusions d'études précédentes, qui précisent les types d'alertes les plus efficaces auprès des utilisateurs (cf. explications détaillées dans le Chapitre 2).

Un aperçu schématique de l'intégration visuelle envisagée est visible en figure 6.12. En complément, un menu de configuration du serveur DNS de référence devra être ajouté, afin de proposer à l'Internaute les choix de configurations DNSref détaillés en section 6.1.1.

Cette intégration visuelle dans le navigateur client amène à plusieurs nouvelles vulnérabilités et/ou limitations, à savoir :

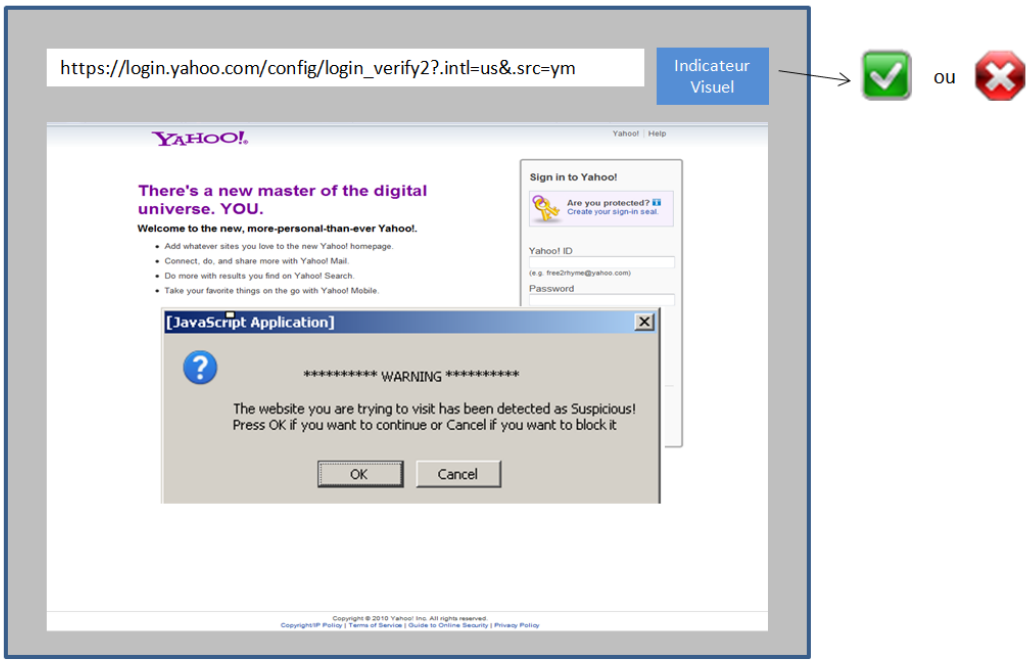


FIGURE 6.12 – Schéma d'intégration visuelle envisagée pour notre proposition de détection du pharming

- notre proposition se voit exposée aux vulnérabilités intrinsèques du navigateur et, plus globalement, à celles de la machine de l'Internaute,
- la corruption éventuelle de l'indicateur visuel,
- et la corruption éventuelle de la configuration du serveur DNS de référence.

Nous devrons donc, à minima, sécuriser le stockage de l'information de configuration du serveur DNS de référence sur la machine client. Une piste éventuelle pourrait être de sauvegarder cette information de manière chiffrée et signée par l'utilisateur par exemple. Ce point devra faire l'objet d'une analyse plus poussée.

A noter que nous devrons également apporter une attention toute particulière à d'éventuels problèmes de réinitialisation de cache DNS au sein du navigateur client et/ou du système d'exploitation. Ceci afin de s'assurer que nos requêtes sont belles et bien générées.

Enfin, nous n'avons pas pu explorer plus avant l'intégration de nos développements menés en Java (langage nécessaire pour la génération des requêtes DNS) au sein d'une barre d'outils en Javascript (langage recommandé pour un meilleur rendu visuel, mais ne pouvant satisfaire aux besoins exigés par les requêtes DNS). Néanmoins, d'après les problèmes rencontrés à ce propos dans le Chapitre 3, il est fort probable que la tâche s'annonce difficile, voire qu'elle nous conduise à des changements de langage.

6.3 Synthèse du chapitre

Ce chapitre ainsi que le précédent ont présenté deux solutions visant à proposer une méthode de détection des attaques de pharming côté client. Ces solutions reposent toutes deux sur une combinaison de la vérification de l'adresse IP du domaine visité, associée à une analyse de contenu des pages webs de login.

Les résultats obtenus dans la seconde proposition exposée dans ce chapitre ont confirmé la staticité des adresses IP associées aux pages de login entrevue dans la première approche (cf. Chapitre 5). Ils ont également indiqué une convergence des résultats en provenance de 3 serveurs DNS de référence différents, permettant ainsi d'aboutir à une meilleure robustesse de la solution. Concernant l'analyse du

code source des pages webs, l'approche par mots s'est révélée être la plus intéressante. Combinée à des techniques d'analyses des balises, elle a permis d'aboutir à la définition d'un seuil de décision. Enfin, la nouvelle technique de récupération de la page de référence a permis de revenir à des taux d'erreurs raisonnables.

A l'issue des résultats convergents et prometteurs obtenus dans nos deux propositions - certes sur un nombre restreint d'URLs -, une implémentation et intégration dans le navigateur client peut être envisagée (moyennant une étude préalable sur le(s) langage(s) les plus approprié(s) à utiliser). Celle-ci peut se présenter, par exemple, sous la forme d'une intégration aux techniques déjà utilisées pour la détection du phishing, à savoir : les barres d'outils intégrées dans le navigateur du client.

Offrir une telle fonctionnalité de détection des attaques de pharming perpétrées sur le réseau client peut s'avérer souhaitable pour améliorer la sécurité globale de la navigation web de l'Internaute, en complément des techniques d'authentification et de protection déjà disponibles et/ou en cours de déploiement côté réseau Internet (p.ex. HTTPS, DNSSEC).

Néanmoins, de nombreuses questions et limitations associées à nos deux propositions restent à résoudre. De plus, la technique d'analyse de comparaison des pages webs se doit d'être perfectionnée. Enfin, des tests sur davantage d'URLs devront être menés.