

Technical overview Virtual Browser

Version 1.0 – premier semestre 2009



Sommaire

1.	Introduction	3
2.	Etat des lieux de la sécurité web	3
2.1.	Vulnérabilités de contenu	3
2.2.	Tromperie de l'utilisateur	4
2.3.	Tromperie du navigateur	4
2.4.	Rebond	4
2.5.	Vulnérabilité dans un site web	5
3.	Problématique de mobilité des applications web	5
4.	Solutions technologiques envisageables	6
5.	Architecture de la solution Virtual Browser	7
5.1.	Agent	7
5.2.	Virtual Soft-appliance	7
6.	Virtual Browser, solution de sécurité et de mobilité web	9
6.1.	Immunité du poste de travail	9
6.2.	Contrôle et cloisonnement des applications	9
6.3.	Mobilité	9
6.4.	Publication d'applications web	9
6.5.	Compatibilité du navigateur avec les applications web	9

1. Introduction

Ce document a pour objectif de décrire la solution innovante *Virtual Browser*. Cette solution est une réponse efficace à la problématique de sécurité et de mobilité des applications web. Grâce à elle, les utilisateurs peuvent surfer sur Internet en toute sérénité.

Virtual Browser est un navigateur Internet d'un genre nouveau puisque le "cœur" du navigateur ne s'exécute plus sur le poste de travail (comme c'est le cas avec les navigateurs classiques tels que Internet Explorer ou Firefox) mais dans des machines virtuelles dédiées, isolées sur un serveur, hébergé dans l'infrastructure de l'entreprise ou sur Internet « in the cloud ». Ce navigateur virtuel n'échange que les éléments de présentation (affichage, son, clavier, souris) avec le poste de travail de l'utilisateur.

2. Etat des lieux de la sécurité web

La sécurité web est un domaine de recherche encore très actif en raison de la croissance importante d'Internet, en particulier pour des opérations commerciales en ligne. Plusieurs types de vulnérabilités sont possibles, certaines directement liées au fonctionnement du web et d'Internet, d'autres plus généralistes aux systèmes d'informations.

2.1. Vulnérabilités de contenu

Le développement d'Internet et sa complexification ont mis le navigateur (qui est finalement le logiciel client d'accès à Internet et aux applications web en général) dans une situation de grande vulnérabilité. En effet, alors qu'il était à l'origine conçu pour lire des pages web relativement statiques, il est devenu en quelques années le client universel d'accès aux informations et aux services en ligne. On lui demande aujourd'hui d'exécuter des applications publiques et des applications sensibles (E-commerce, banque en ligne, webmail, intranet métier) dans un seul et même programme logiciel.

Par conséquent, le navigateur et ses extensions (plugins) sont confrontés à un grand nombre de fichiers à traiter et présentés sous un grand nombre de formats différents : documents, images, vidéos, musiques, scripts, ... Cette diversité conduit inévitablement à une grande complexité et des erreurs d'implémentation permettant d'exécuter des attaques sont régulièrement trouvées dans ces composants. En outre, le développement d'applications web de nouvelle génération (les applications web 2.0) conduit même aujourd'hui les navigateurs à exécuter du code provenant de sites web. En d'autres termes les navigateurs exécutent localement des programmes envoyés par des sources non connues. Enfin les navigateurs (et en particulier leurs extensions) ne sont pas toujours bien mis à jour par les utilisateurs, ce qui accentue le risque.

L'exploitation de ces vulnérabilités provoque souvent une infection de la machine par différents types de malwares (virus, troyen, spyware, ...) qui sont alors téléchargés lors de l'attaque (en utilisant des malwares de type Downloader).

2.2. Tromperie de l'utilisateur

L'utilisateur peut être trompé par un site malveillant cherchant la plupart du temps à divulguer des informations personnelles ou à effectuer des opérations à son insu.

- **Phishing** : Le plus souvent reçu par e-mail, les tentatives de phishing sont destinées à faire se connecter un utilisateur sur un site malveillant en faisant croire qu'il s'agit d'un site de confiance (banque, webmail, ...). L'utilisateur est ensuite incité à saisir des informations personnelles (mots de passe, informations bancaires, ...) qui seront récupérées par l'attaquant et utilisées sur le véritable site de confiance.
- **Clickjacking** : Exploitant le plus souvent les possibilités d'affichage avancé offertes par les navigateurs, le clickjacking permet de faire cliquer un utilisateur dans une page qu'il ne voit pas grâce à un jeu de superposition de pages web. L'utilisateur va alors modifier ou divulguer à son insu des informations confidentielles. Le clickjacking est également utilisé pour modifier les paramètres d'accès à la webcam par le plugin Flash, permettant alors à l'attaquant de voir et d'entendre l'utilisateur à son insu.

2.3. Tromperie du navigateur

Le navigateur peut également être trompé lors de sa connexion aux sites demandés par l'utilisateur :

- **DNS rebinding** : Exploitation du protocole DNS pour modifier les adresses IP pointées entre un site légitime et un site malveillant. Cette technique permet de contourner la protection de "same origin policy" des navigateurs afin de faire exécuter du code provenant d'un site malicieux et ayant accès à un site légitime.
- **Pharming** : Si la configuration DNS de l'utilisateur a été modifiée de manière malveillante (serveur DNS modifié, ajout d'entrées dans le fichier hosts, ...), le navigateur peut présenter à l'utilisateur un site malveillant pour un site légitime.

Ces vulnérabilités peuvent être mitigées dans le cas d'un site authentifié par un certificat (connexion HTTPS) et dans le cas où l'utilisateur n'accepte pas un certificat qui n'est pas de confiance (ce qui est malheureusement assez rare).

Note : Le mécanisme de protection de "same origin policy" présent dans les navigateurs assure qu'un script, provenant d'un site web et exécuté sur le poste, ne peut accéder ou récupérer que des documents ou fichiers provenant de la même origine. C'est une protection essentielle mais dont le contournement permet la plupart du temps d'accéder à des informations confidentielles.

2.4. Rebond

De par son fonctionnement qui le conduit à accéder (parfois en même temps) à des sites et des applications web très différents les uns des autres, le navigateur est constamment amené à suivre des liens en provenance de pages web sans en maîtriser le risque. Il peut ainsi servir de passerelle entre plusieurs sites web.

Plusieurs vulnérabilités sont alors possibles :

- **Intranet scanning** : En utilisant le plus souvent un script Javascript (même si certaines techniques permettent de s'en passer), il est possible pour un attaquant de faire scanner le réseau local de l'utilisateur par son navigateur et de récupérer les informations ainsi obtenues (IP disposant d'un serveur web, type du serveur web, application installée, ...).
- **Déclenchement d'une vulnérabilité dans un site web** : La plupart des exploitations de vulnérabilités dans un site web (XSS, CSRF) passent par un rebond sur le navigateur de l'utilisateur.

2.5. Vulnérabilité dans un site web

Des vulnérabilités spécifiques à la technologie web sont présentes dans de nombreux sites web, bien que connues depuis des années :

- **XSS (Cross Site Scripting)** : Une vulnérabilité dans un site web permettant de faire exécuter du code externe en contournant la protection de "same origin policy" des navigateurs. Peut servir à voler les cookies de session d'un utilisateur.
- **CSRF (Cross Site Request Forgery)** : Un manque de contrôle dans un site web pouvant être utilisé pour faire exécuter à son insu des actions (envoi d'email, modification de mot de passe, transfert d'argent, ...) par un utilisateur sur un site sur lequel il est authentifié.

Ces vulnérabilités sont exploitées soit par rebond à partir d'un site malveillant, soit par un lien présent dans un e-mail.

3. Problématique de mobilité des applications web

La problématique de mobilité des applications web vient à la fois du développement du nomadisme en entreprise (ouverture du système d'information pour les utilisateurs mobiles, les télétravailleurs, les clients ou les partenaires) et de la "webification" des applications (développement des portails d'entreprise, du webmail, de l'intranet, du Software-as-a-service et du Cloud Computing).

De plus en plus d'entreprises ont besoin de publier des applications vers l'extérieur ce qui pose un double problème :

- **Comment garantir que l'ouverture du système d'information se fait sans risque ?**
A partir du moment où l'application est accessible depuis l'extérieur, l'entreprise donne un accès public sur un "site web" (http ; TCP port 80) avec tous les risques que cela comporte : tentative d'intrusion directe sur le site par un attaquant ou contamination du site par un poste autorisé mais lui-même contaminé.
- **Comment s'assurer que l'application publiée sera toujours compatible avec les postes des utilisateurs alors même que l'entreprise ne les maîtrise pas toujours ?**

Lorsqu'une application web est développée, l'entreprise doit s'assurer qu'elle est compatible avec l'ensemble des navigateurs du marché (y compris sur smartphones, PDA et autres iPhones) sans pour autant maîtriser les différents plugins et composants installés par chaque utilisateur.

Cette problématique est du reste valable lorsqu'on parle d'applications grand public. Une banque en ligne, un site de E-commerce ou toute application web hébergée rencontre des risques en matière de sécurité, de disponibilité et de compatibilité.

4. Solutions technologiques envisageables

Les différentes vulnérabilités web présentées plus haut peuvent être résolues ou mitigées avec différentes technologies :

Vulnérabilités de contenus :

- Filtrage anti-virus et anti-malware sur tous les flux (y compris HTTPS).
- Mise à jour des différents composants du navigateur.
- Exécution de chaque application dans un environnement virtuel permettant de circonscrire l'impact et de démarrer chaque session dans un environnement sain.

Tromperie de l'utilisateur :

- Filtrage d'URL utilisant une liste noire de sites de Phishing connus.
- Cloisonnement de session pour que les sites légitimes ne soient exécutés que dans des sessions bien identifiées (Phishing) et qu'un site malveillant ne puisse inclure un site légitime (Clickjacking).

Tromperie du navigateur :

- Backoffice de sécurité web qui prend en compte ces différentes problématiques

Rebond :

- Cloisonnement de sessions web pour éviter le rebond d'un site malveillant vers un site légitime ou interne.

Vulnérabilité dans un site web :

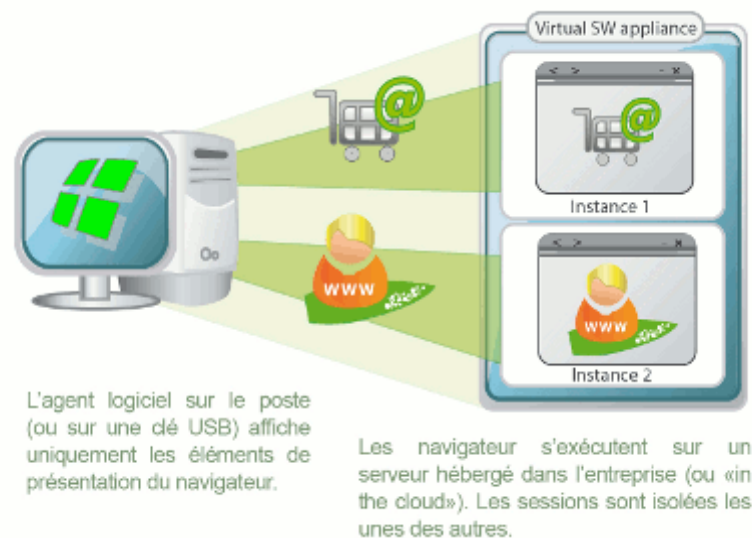
- Cloisonnement de sessions web pour éviter que la navigation sur un site malveillant ne déclenche l'exploitation d'une vulnérabilité sur un site légitime.

Dans ce contexte, vous verrez dans la suite de ce document que la solution *Virtual Browser* à elle seule permet de couvrir la plupart des risques liés au web. Adossée à un backoffice de sécurité web, elle permet de résoudre l'ensemble des problématiques évoquées en offrant une navigation sécurisée, en particulier sur les sites de confiance.

En outre la solution *Virtual Browser* permet d'apporter une solution infaillible en termes de gestion de la mobilité des applications web. En effet elle apporte à la fois les garanties de sécurité de l'application publiée mais aussi de compatibilité de cette application puisque le navigateur d'accès à l'application web est lui-même "en ligne". Les problèmes de compatibilité sont résolus puisque dans tous les cas, le poste de travail exécutera le navigateur publié. En cas de mise à jour de l'application, l'entreprise n'a qu'un seul navigateur à mettre à jour : celui qui est sur le serveur central.

5. Architecture de la solution Virtual Browser

L'architecture globale de la solution Virtual Browser est la suivante :



La solution est composée de deux parties logicielles : (1) le client (ou *agent*) et (2) le serveur (ou *virtual soft-appliance*).

5.1. Agent

L'*agent* est le composant qui est installé sur le poste de l'utilisateur et qui est chargé de réaliser la connexion à la *soft-appliance* et l'affichage déporté du navigateur :

- La connexion à la *soft-appliance* est sécurisée par une connexion SSL.
- La connexion à la *soft-appliance* sert à authentifier l'utilisateur, ouvrir les sessions, récupérer les informations de configuration, ...
- L'affichage déporté est réalisé en utilisant le protocole open source NX (NoMachine) et un serveur X local (embarqué dans l'*agent*).
- L'agent pourra fonctionner sur un grand nombre de systèmes d'exploitation desktop (Windows XP/Vista, Mac OS X, Linux, ...) et mobile (Windows Mobile, iPhone, Blackberry, ...). Dans la version 1.0 il fonctionne uniquement sous environnement Windows XP/Vista.
- L'*agent* est fourni également dans une version portable auto-exécutable pour fonctionner sur une clé usb.

5.2. Virtual Soft-appliance

La *soft-appliance* est la partie serveur de la solution, elle est utilisée pour gérer les différentes instances de *Virtual Browser*.

La *soft-appliance* est fournie par commonIT sous la forme d'une image VMware. Ce format permet de mettre facilement en place une ferme d'appliances *Virtual Browser* pour s'adapter au besoin de charge ou de localisation géographique des serveurs.

C'est via une connexion https sur la *soft-appliance* que l'administrateur configure et supervise la solution :

Gestion des utilisateurs :

- Les différents utilisateurs de *Virtual Browser* sont authentifiés sur une base locale ou sur un serveur externe (LDAP, NTLM).
- Chaque utilisateur, pourra (si l'administrateur le décide) retrouver ses éléments d'archivage de sessions (favoris, cookies, etc...) à chaque connexion.
- Lors de chaque téléchargement (upload/download), les fichiers passent par une zone dédiée à l'utilisateur sur le serveur :
 - La taille de cette zone est définie par l'administrateur.
 - Dans cette zone tampon dédiée, les contrôles de contenus sont effectués : protocole ICAP permettant d'interroger un serveur antivirus externe depuis le navigateur.

Gestion des applications et des sessions :

- L'administrateur définit les sessions qui seront accessibles sur la *soft-appliance*. Ces sessions sont caractérisées par un type de navigateur (voir plus bas), par une icône (qui sera visible sur le poste de l'utilisateur et qui lui permettra d'ouvrir la session) et par des applications web (ou des URL).
- L'administrateur définit quelles sont les applications qui ont le même niveau de sensibilité et qui auront le droit d'être exécutées dans une même session.
- Les différentes sessions de *Virtual Browser* sont lancées dans un environnement OpenVZ permettant d'isoler les applications les unes des autres dans des machines virtuelles différentes.

Gestion des navigateurs :

- A chaque session, l'administrateur associe un profil de navigateur.
- Dans le profil du navigateur, l'administrateur choisit :
 - Le moteur de rendu utilisé (en version 1.0 seul le moteur de rendu Gecko (Firefox) est disponible, les moteurs d'IE et de Safari suivront dans les prochaines versions).
 - Les plugins (choisis parmi une liste fournie et mise à jour par commonIT).
 - Si l'impression et le téléchargement (upload/download) de fichiers sont autorisés.
- Les composants des navigateurs sont mis à jour régulièrement et automatiquement.
- Le navigateur est lancé dans un environnement virtuel OpenVZ propre et redémarré à chaque ouverture de nouvelle session.
- Le navigateur virtuel bénéficie d'une interface renforcée permettant d'empêcher un accès au système local.
- Les informations de l'utilisateur (cookies, bookmark, ...) sont présentes dans un répertoire monté par le serveur dans l'environnement virtuel.
- Le filtrage des contenus via le protocole ICAP est effectué directement dans le navigateur et non pas dans le flux, ce qui permet de faire les contrôles sur les flux chiffrés https.

6. Virtual Browser, solution de sécurité et de mobilité web

La solution *Virtual Browser* offre les fonctionnalités suivantes :

6.1. Immunité du poste de travail

Par construction, *Virtual Browser* isole le navigateur web vis-à-vis du poste de travail et du réseau en permettant son exécution sur un serveur dédié. De par son architecture, *Virtual Browser* garantit qu'aucune attaque ne peut toucher le poste puisqu'aucune application web n'est exécutée localement. Si une attaque survient, elle reste isolée dans l'instance virtuelle de sa session sur le serveur et elle est détruite lorsque l'utilisateur ferme la fenêtre.

6.2. Contrôle et cloisonnement des applications

Virtual Browser permet de maîtriser totalement le comportement du navigateur et en particulier des applications et des contenus encapsulés dans les flux web chiffrés https.

Par ailleurs, sur la soft-appliance, les sessions web sont cloisonnées les unes des autres selon leur niveau de confiance. Grâce à ce mécanisme, les applications à risque, comme le surf Internet, sont exécuter dans des instances isolées des applications critiques (CRM, banques en ligne, travail collaboratif, Intranet, etc...) qui sont ainsi protégées nativement.

6.3. Mobilité

Virtual Browser peut remplacer avantageusement une solution de VPN en offrant un accès distant sécurisé aux applications web de l'entreprise sans nécessiter de contrôle d'intégrité du poste. En effet l'isolation du navigateur, qui protège le poste contre une attaque venant d'une application web, permet à l'inverse de protéger l'application web d'un éventuel virus présent sur le poste.

6.4. Publication d'applications web

Pour les entreprises proposant des services en ligne sur Internet *Virtual Browser* est une excellente solution d'accès sécurisé. En effet le serveur de l'application n'a plus besoin d'être accessible sur Internet (port 80/TCP ouvert) mais peut être configuré pour n'accepter que les connexions provenant de navigateurs virtuels hébergés dans l'infrastructure de l'entreprise. De cette manière l'entreprise se protège contre les intrusions et les attaques de type CSRF ou Man-in-the-browser.

6.5. Compatibilité du navigateur avec les applications web

L'architecture de *Virtual Browser* consiste à dédier une instance virtuelle par session web sur le serveur. De cette manière l'entreprise peut définir pour chaque application la version du navigateur qui sera utilisée. Par exemple, si une application de l'intranet n'est compatible qu'avec le moteur Firefox 3.0 utilisant le plugin java JRE 1.6, tous les utilisateurs exécuteront automatiquement ce navigateur sans même s'en rendre compte, quelque soit le terminal qu'ils utilisent.