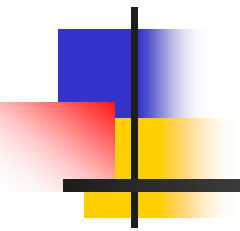


Backdoors en environnement Windows

La problématique de filtrage des flux sortants

Benjamin CAILLAT

Département sécurité du SCNA (Ministère de l'Économie et des Finances)
Enseignant- Ingénieur au Master sécurité de l'ESIEA

Site: http://benjamin.caillat.free.fr

Mail: b.caillat@security-labs.org

Préambule

- L'interconnexion réseau interne-Internet
  - Une nécessité pour les entreprises (accès web, messagerie)
  - Introduit de nombreuses problématiques de sécurité
- Le contrôle des flux entrants
  - Multitude de produits pour construire infrastructure assurant un contrôle des flux entrants efficace
  - L'intrusion sur le réseau interne depuis l'extérieur en passant par cette architecture est difficile
- Mais qu'en est-il de flux sortants ?
  - Cas d'un programme malicieux exécuté sur réseau interne ?
  - Infrastructures suffisantes pour bloquer envoi de données ?
  - Sinon quel est le niveau de compromission du SI ?
  - Quelles solutions de détection ?

Cadre et postulats (1)

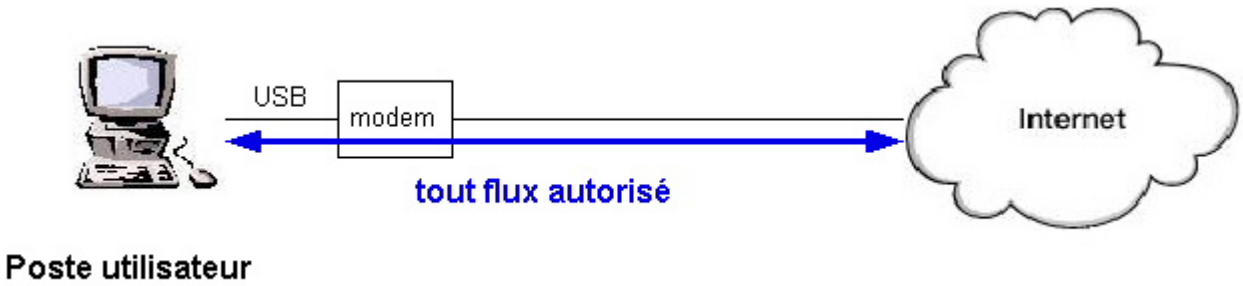
- Injection du programme malicieux :
  - Scénario implique qu'un pirate soit parvenu à placer et à provoquer l'exécution d'un programme sur réseau interne.
  - Social engineering, failles (JPG, IFRAME overflow).
- Objectif du programme malicieux.
  - Cas des codes destructeurs écarté.
  - Programme malicieux ~ backdoor cherchant à envoyer des données à un pirate sur Internet.
- Étude constituée d'une partie théorique et d'une partie pratique, basée sur des développements personnels:
  - Deux backdoors: Fratus et Parsifal.
  - Un programme de contrôle: BlackMoon.

Cadre et postulats (2)

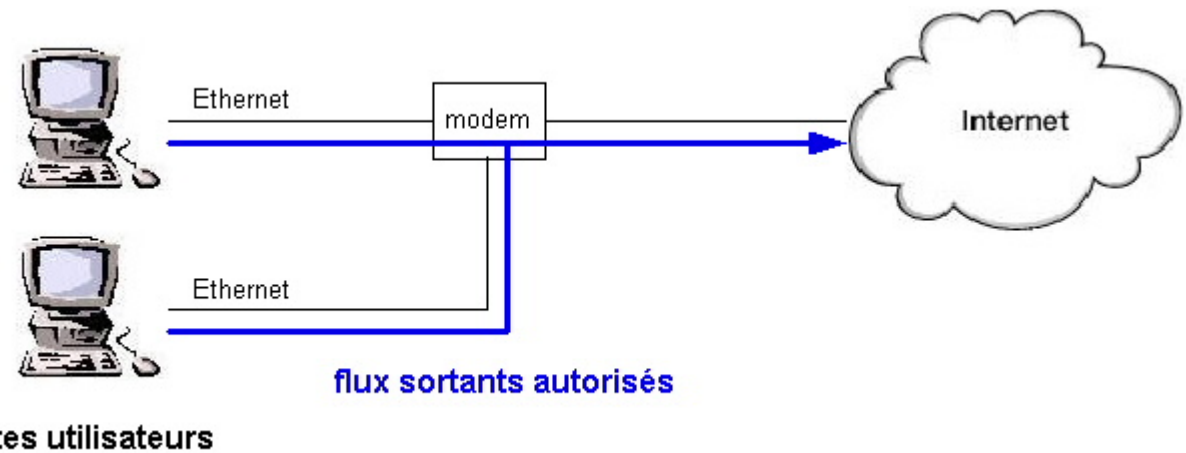
- Architecture réseau considérée.
  - Cas de postes WINDOWS ayant accès au web.
  - Interconnexion entre postes et Internet peut se faire de plusieurs manières:
    - Connexion directe.
    - Connexion via passerelles applicatives.

Cadre et postulats (3)

- Connexion directe (particulier)
  - Modem USB

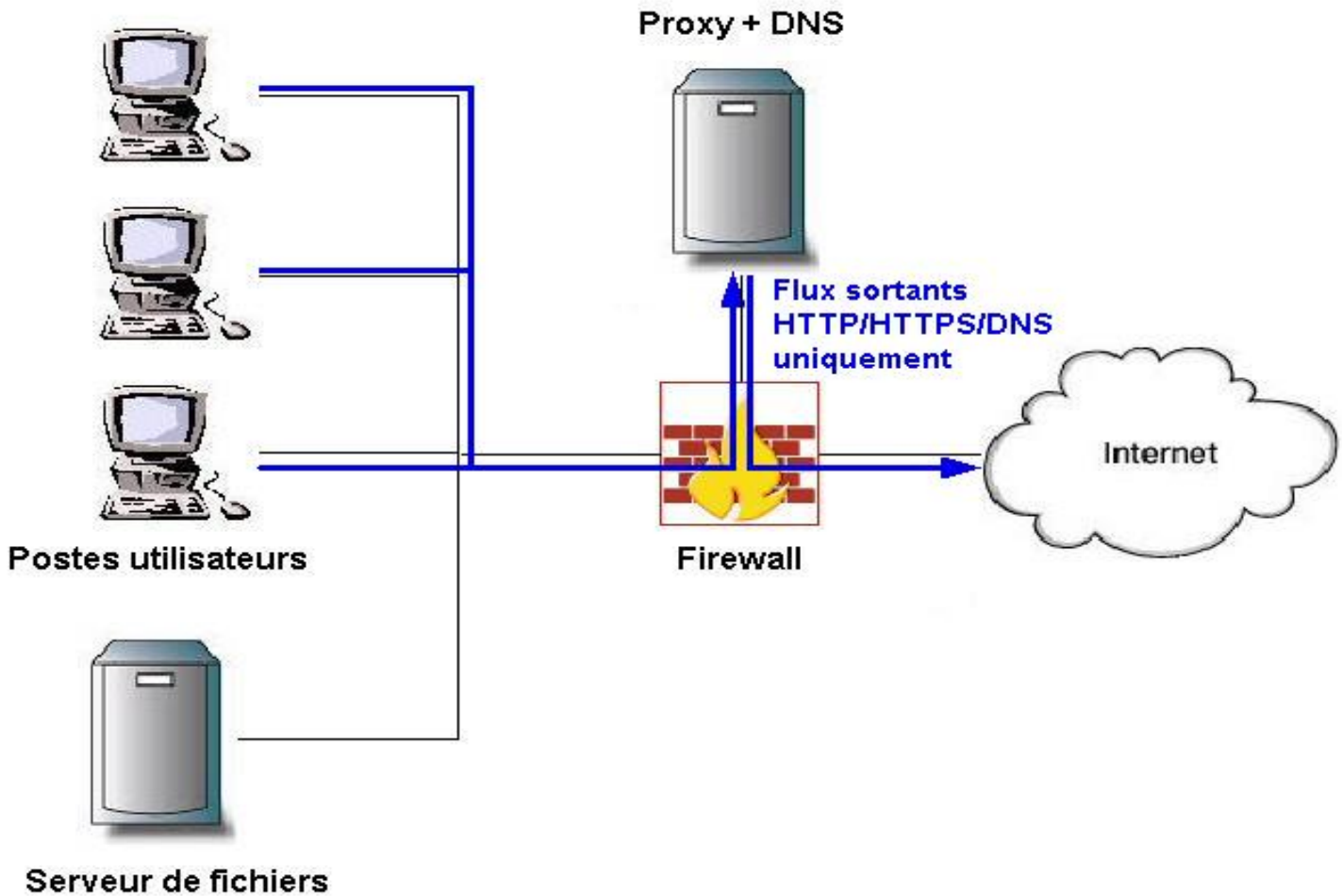


  - Modem Ethernet



Cadre et postulats (4)

- Connexion via des passerelles applicatives (entreprises)



Cadre et postulats (5)

- Matrice de flux

Sens / Connexion	Poste utilisateur vers Internet	Internet vers Poste utilisateur
Directe (USB)	Tout accès autorisé	Tout accès autorisé
Directe (Ethernet)	Tout accès autorisé	Tout accès refusé
Via passerelle	HTTP, HTTPS, DNS via proxy	Tout accès refusé

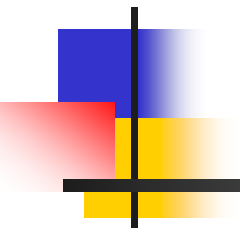
- Cas des entreprises offrant une connexion web (HTTP, HTTPS et DNS) à leurs employés

  => Considère la troisième architecture

  (Mais étude valable pour les autres architectures)

Objectif de l'étude

- Objectif général:
  - Évaluer le risque en terme de confidentialité lié à l'exécution d'un programme inconnu sur le réseau interne d'une entreprise offrant des accès Web.
- Détail:
  - Déterminer si architecture réseau (Firewall + Proxy) est suffisante pour empêcher qu'une backdoor envoie des données à un pirate.
  - Sinon: Étudier les possibilités offertes au pirate.
  - Évaluer les dispositifs de détection.

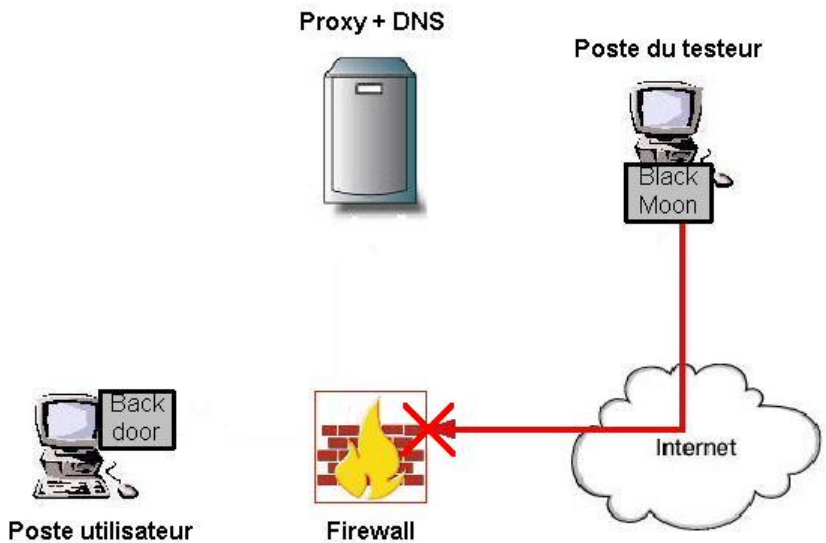


Contournement du filtrage des flux sortants

Théorie: Caractéristiques essentielles de la backdoor

Client ou serveur ?

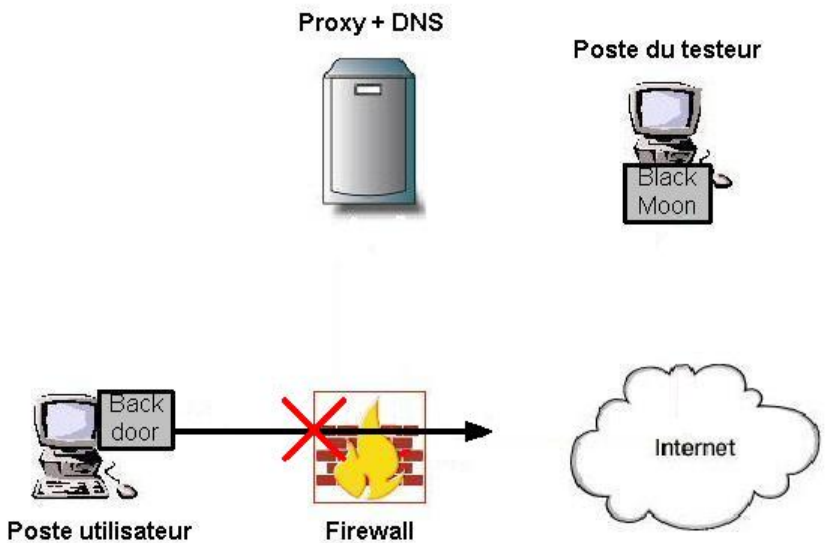
- Naturellement = backdoor ~ serveur.
- Configuration ne fonctionne pas.



=> La backdoor doit donc être la partie cliente qui initie des flux vers un serveur sur Internet.

Nature des communications ?

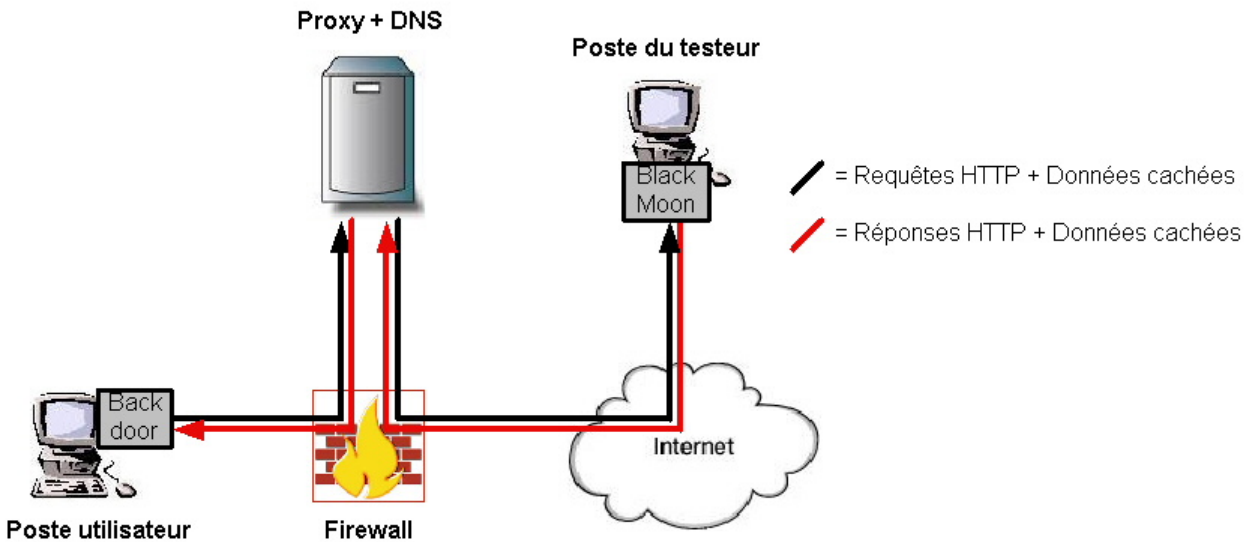
- Firewall interdit tout flux de la zone protégée vers la zone externe.
- Ouverture d'une connexion directe impossible.



=>Communication doit s'appuyer sur flux autorisés HTTP, HTTPS et DNS.

Transport des données dans HTTP (1)

- Autoriser HTTP = un client et un serveur vont être autorisés à échanger des données
- Rien ne garantit que client~navigateur et serveur~serveur web
- Principe du transport de données



- Nécessite seulement de suivre norme HTTP pour proxy

Transport des données dans HTTP (2)

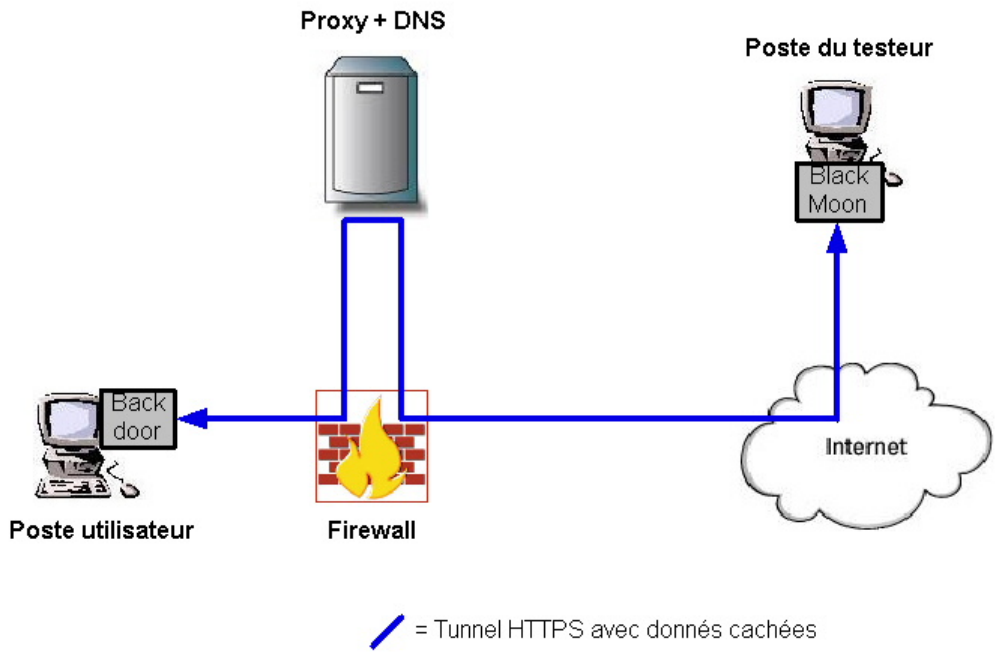
- Envoi serveur -> backdoor:
  - Aisé car réponses HTTP contiennent une partie “data”
- Envoi backdoor -> serveur:
  - Requêtes HTTP = méthode GET => pas de partie “data”
  - Faibles quantités = encodage dans header
  - Transferts importants = méthode POST

Transport des données dans HTTPS (1)

- Problème de la proxyfication des flux HTTPS.
- Principe de l'établissement d'un connexion:
  - Client initie connexion TCP avec le proxy.
  - Client envoie une requête HTTP CONNECT avec @IP et port du serveur à contacter.
  - Proxy initie une connexion TCP avec le serveur.
  - Proxy renvoie au client un message de confirmation de connexion.
  - Données TCP de tous les paquets de cette connexion sont transmis sans aucun contrôle.

Transport des données dans HTTPS (2)

- Principe du transport de données:

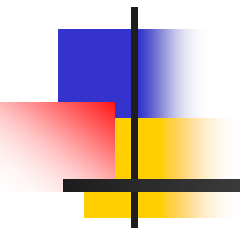


- Etablissement d'un tunnel avec la méthode "CONNECT".
- Transport des données.

Partie théorique: Conclusion

- Propriétés essentielles de la backdoor:
  - Backdoors = parties clientes
  - Communication par canaux cachés au sein HTTP, HTTPS et DNS
  - Backdoors doivent en premier lieu récupérer les paramètres de connexion réseau

=> Malgré mise en place d'une infrastructure autorisant seulement HTTP, HTTPS et DNS, communication avec le pirate est possible.



Contournement du filtrage des flux sortants

Pratique: Implémentation dans Fratus et Parsifal

Deux clients, un serveur

- Développements personnels constitués de 3 outils:
  - Fratus et Parsifal: 2 Backdoors ~ 2 parties clientes

    Applications « console » développées en C
  - BlackMoon: Partie serveur

    Application « graphique » développée en C#
- Nature des communications:
  - Basées sur des canaux cachés au sein de HTTP et HTTPS
  - DNS non traité

Transport des données dans HTTP (1)

- Deux types de requêtes:
  - Sans données => méthode « GET »
  - Avec données => méthode « POST »
- Format des requêtes

```
GET [URL] HTTP/1.0
Accept: text/html;q=[CODE]
Proxy-Connection: keep-alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;
   [BACKDOOR ID].[KEY]

POST [URL] HTTP/1.0
Accept: text/html;q=[CODE]
Proxy-Connection: keep-alive
Content-Length: [DATA LENGTH]
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0;
   [BACKDOOR ID].[KEY]
[DATA]
```

Transport des données dans HTTP (2)

- Format des réponses

```
HTTP/1.1 [CODE] OK
Content-Length: [DATA-LENGTH]
Content-Encoding: gzip
Cache-Control: private
Connection: keep-alive
[DATA]
```

- Utilisation du mode « keep-alive »

  => Connexion rétablie automatiquement en cas de fermeture

- Encodage des données

  => XOR aléatoire transmis dans header de la requête

- Problème du cache sur le proxy

  => Utilisation de la directive « Cache-Control »

- Fonctionnement en mode polling

Transport des données dans HTTPS

- Principe:
  - Établissement du tunnel via la méthode CONNECT
  - Envoi des codes/données encodés par XOR
- Difficulté de la programmation de l'établissement du tunnel sécurisé
  - Librairies Windows = objet COM ou bas niveau
  - Librairie openssl pas forcément installée

  => Pas d'établissement de tunnel crypté

Récupération des paramètres de connexion

- Mode d’exécution des backdoors complètement différent

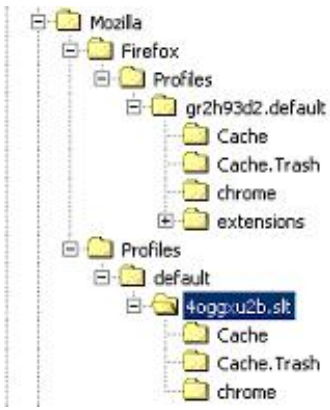
  => Principe de récupération différent pour les deux backdoors

FRATUS

- Mode d'exécution = Processus séparé
- Récupération des paramètres de connexion réseau
  - Internet Explorer: API wininet.dll, registry
  - Netscape, Firefox:

    fichier "pref.js" stocké dans

    un répertoire de nom aléatoire:



    => Necessité d'effectuer une recherche récursive du fichier
  - Problème de la configuration automatique (PAC)
- Survie de la backdoor ~ survie du processus Fratus

PARSIFAL (1)

- Mode d'exécution

  Thread injecté dans les processus navigateurs
- Récupération des paramètres de connexion réseau

  Basée sur le hooking des fonctions connect et send
- Injection présente de nombreux avantages:
  - Processus Parsifal ne s'exécute qu'un temps très court
  - Les connexions réseaux initiées par la backdoor seront vues par les FWs personnels comme provenant du navigateur
  - Supporte l'utilisation de configuration automatique (PAC)
  - Supporte l'authentification simple sur le proxy
- Mais introduit de nouveaux problèmes:
  - Si aucun navigateur lancé au moment de l'injection ?
  - Si l'utilisateur ferme le navigateur ?

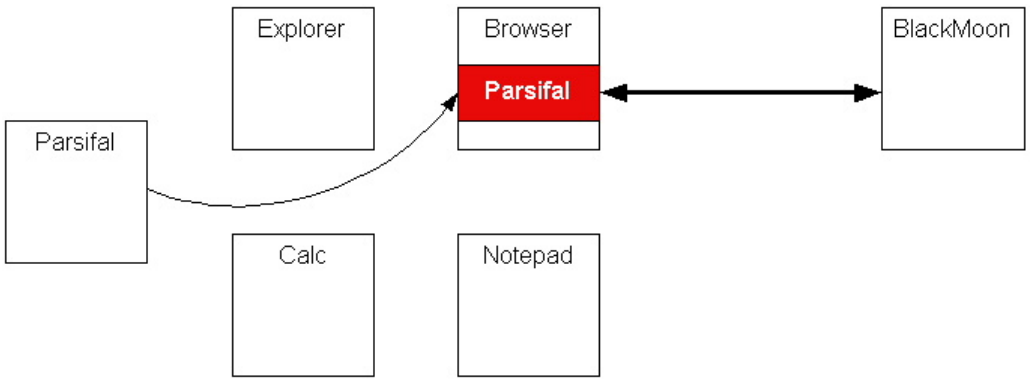
PARSIFAL (2)

- Solution 1: Poser des hypothèses
  - Postulat “un navigateur est lancé” = hypothèse contraignante
  - Postulat “Survie de la backdoor liée navigateur” = impossible

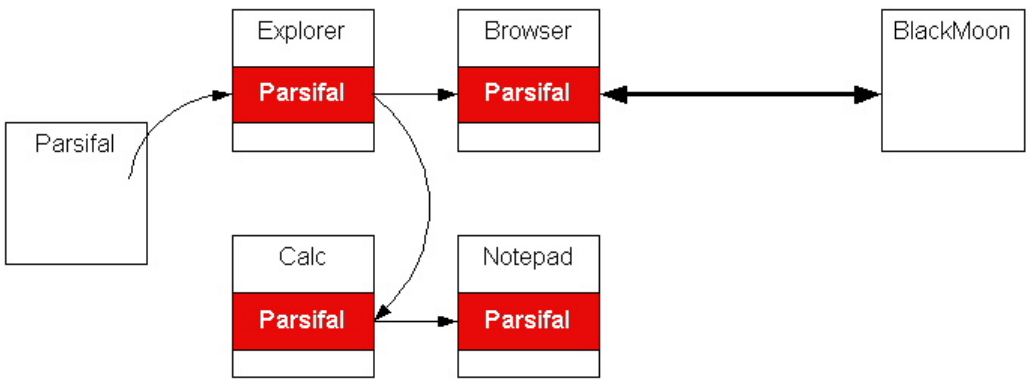
    => Solution non exploitable
- Solution 2: Adopter un comportement viral
  - Parsifal injecte tous les processus de l’utilisateur
  - Si processus est un navigateur: comportement “backdoor”
  - Sinon : recherche de processus à injecter

PARSIFAL (3)

- Trois modes de fonctionnement
  - Browser: l'injection directe

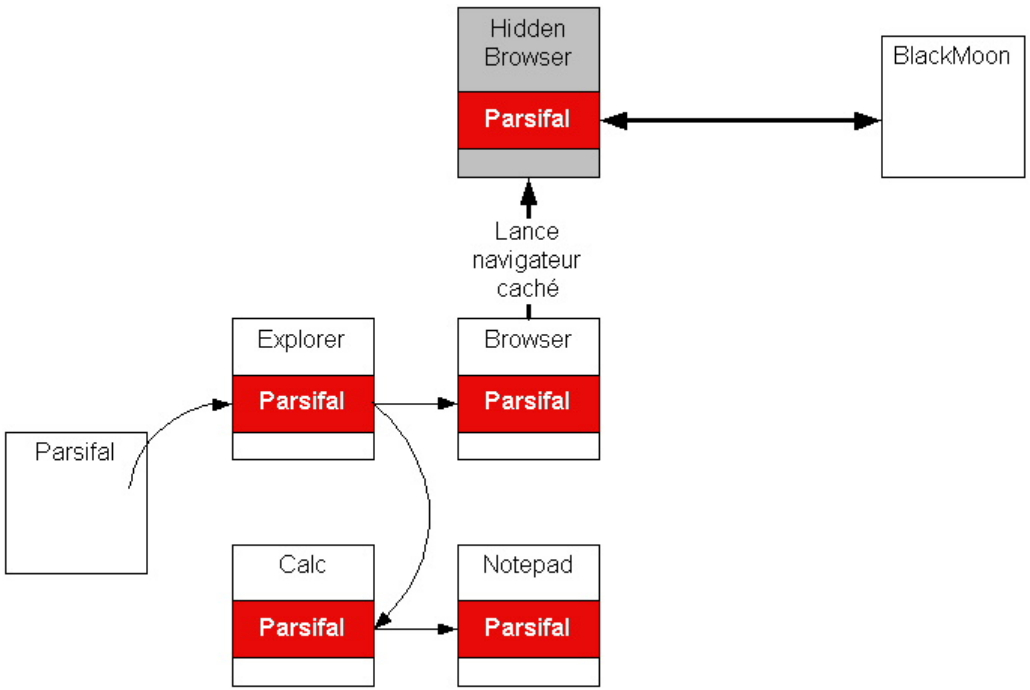


  - All: la propagation virale



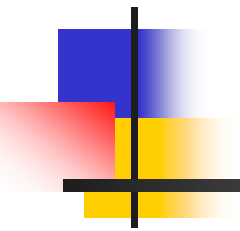
PARSIFAL (4)

- All Browser: la propagation virale + lancement de navigateur caché



Partie pratique: Conclusion

- Fratus et Parsifal = Deux backdoors:
  - Parties clientes.
  - Communiquent avec une partie serveur via des canaux cachés au sein de HTTP et HTTPS.
  - Capables de récupérer les paramètres de connexion réseau.
- Preuve que l'infrastructure considérée est donc insuffisante pour empêcher l'établissement de canaux cachés et l'envoi de données de la backdoor vers le pirate.



Fonctionnalités implémentées dans Fratus et Parsifal

La problématique de l’évolutivité

Un enjeu: l'évolutivité

- Backdoors peuvent communiquer avec partie serveur, mais que peuvent-elles réellement faire ?
- Pour montrer les capacités de telles backdoors, l'architecture de Fratus et Parsifal doit leur garantir une évolutivité maximale
- Intégration des fonctionnalités dans les backdoors pose plusieurs problèmes:
  - Nécessite de porter le code dans les deux backdoors
  - Nécessite une recompilation
  - Augmente la taille de la backdoor

Une solution: la modularité

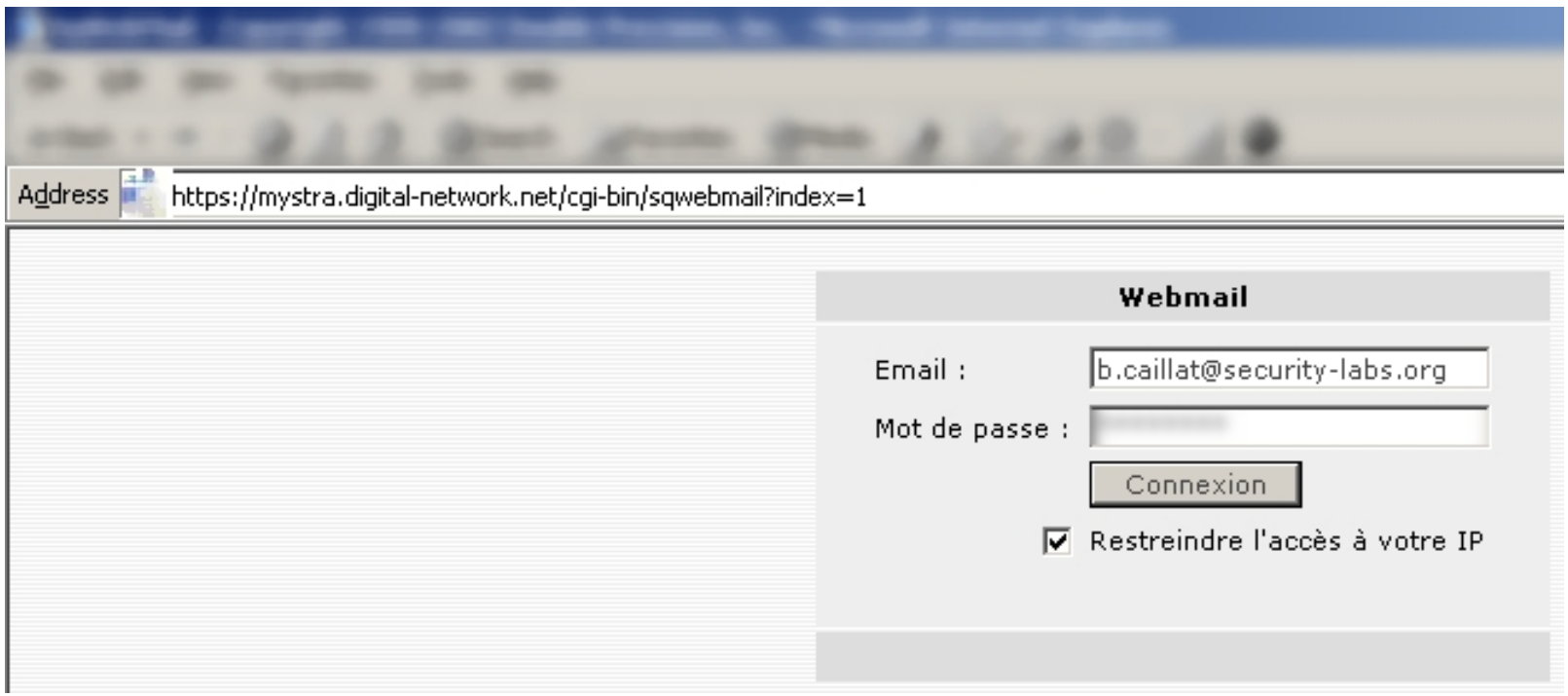
- Backdoors suivent une architecture modulaire
  - Fonctionnalités réelles déportées dans des modules (Dll)
  - Backdoors n'offrent qu'un ensemble limité de fonctionnalités permettant :
    - L'upload des modules
    - Le transfert des commandes vers les modules
    - Le transfert des résultats générés par les modules vers BlackMoon

Prérequis de la modularité

- Modularité nécéssite:
  - L'existence d'une interface de communication standard entre module et backdoors.
  - Que les backdoors supportent un ensemble minimal de fonctionnalités pour implémenter l'interface.
- Interface constituée de:
  - Quatre fonctions, dont le prototype est normalisé, exportées par la Dll.
  - Un fichier de logs dont le nom est normalisé.
  - Un mutex de synchronisation pour l'accès au fichier de logs.
- Fonctionnalités des backdoors
  - Récupération des paramètres de connexion réseau.
  - Transport de données (messages, fichiers de logs).
  - Implémentation d'un mini-shell.
  - Survie de la backdoor au cours de la session utilisateur.
  - Survie au redémarrage.

PARSIFAL: L'interception des requêtes « intéressantes » (1)

- Spécificité liée à la technique d'API hooking
  - Hook de la fonction HttpSendRequest
  - Dans IE, permet d'intercepter toutes les requêtes
  - Logge des requêtes "intéressantes"
- Exemple: Accès messagerie sécurisée



PARSIFAL: L'interception des requêtes « intéressantes » (2)

- Récupération du mot de passe dans un fichier local

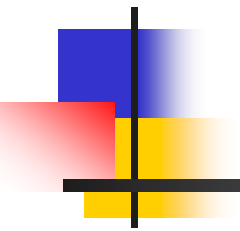
```
-----------------------------------------------
https://mystra.digital-network.net/cgi-bin/sqwebmail?empty=1

-----------------------------------------------
https://mystra.digital-network.net/cgi-bin/sqwebmail?noframes=1

-----------------------------------------------
https://mystra.digital-network.net/cgi-bin/sqwebmail
 Optionnal data         = username=b.caillat@security-labs.org&password=[illegible]&do.login=Connexion&sameip=on
```

Possibilités offertes: Conclusion

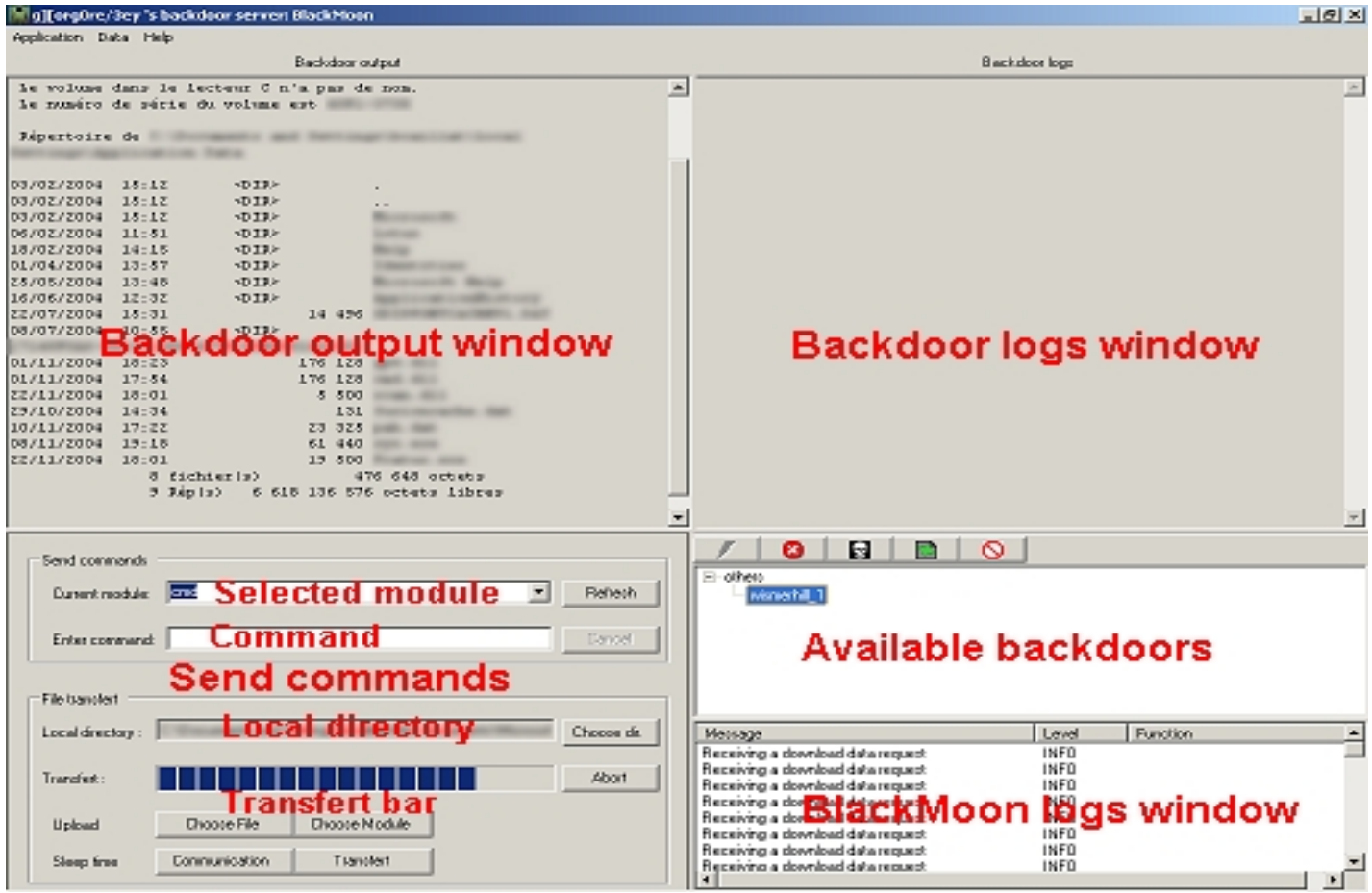
- Modularité apporte de nombreux avantages:
  - Compatibilité des modules.
  - Pas de recompilation. Seules les caractéristiques de l'interface de normalisation sont nécessaires pour développer de nouveaux modules.
  - Taille reste fixe.

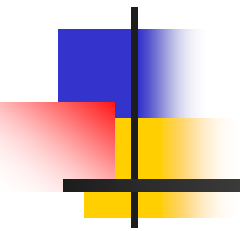


Présentation de BlackMoon

Présentation de BlackMoon

- Application graphique développée en C#



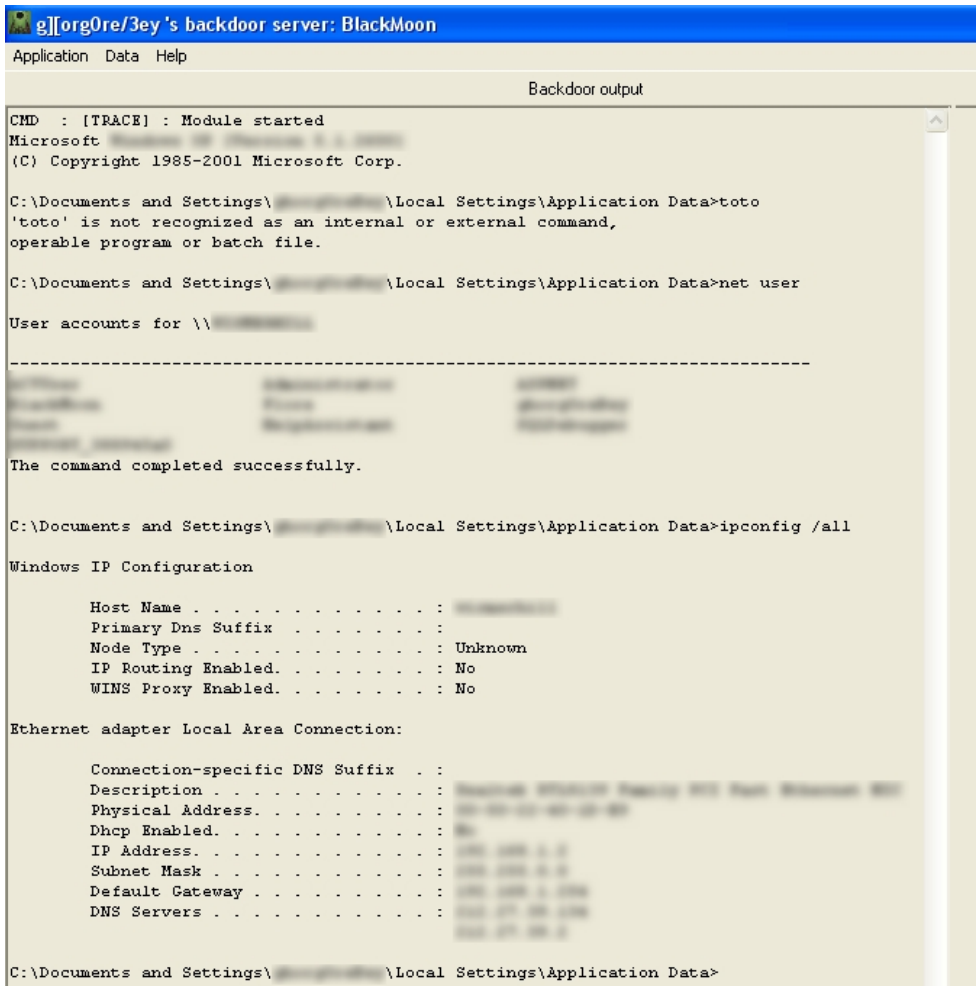


Les possibilités des backdoors

Description des modules fournis

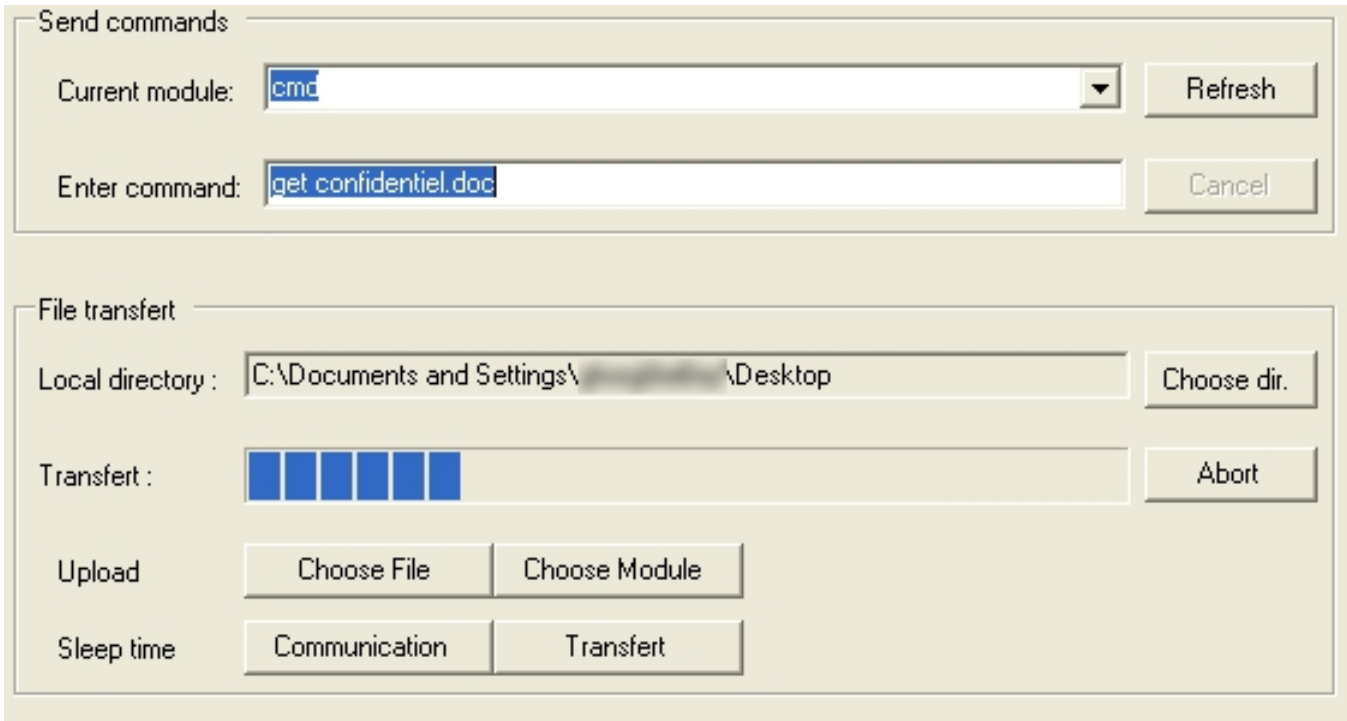
“cmd”, le contrôle de l’hôte (1)

- Equivalent de “cmd” distant



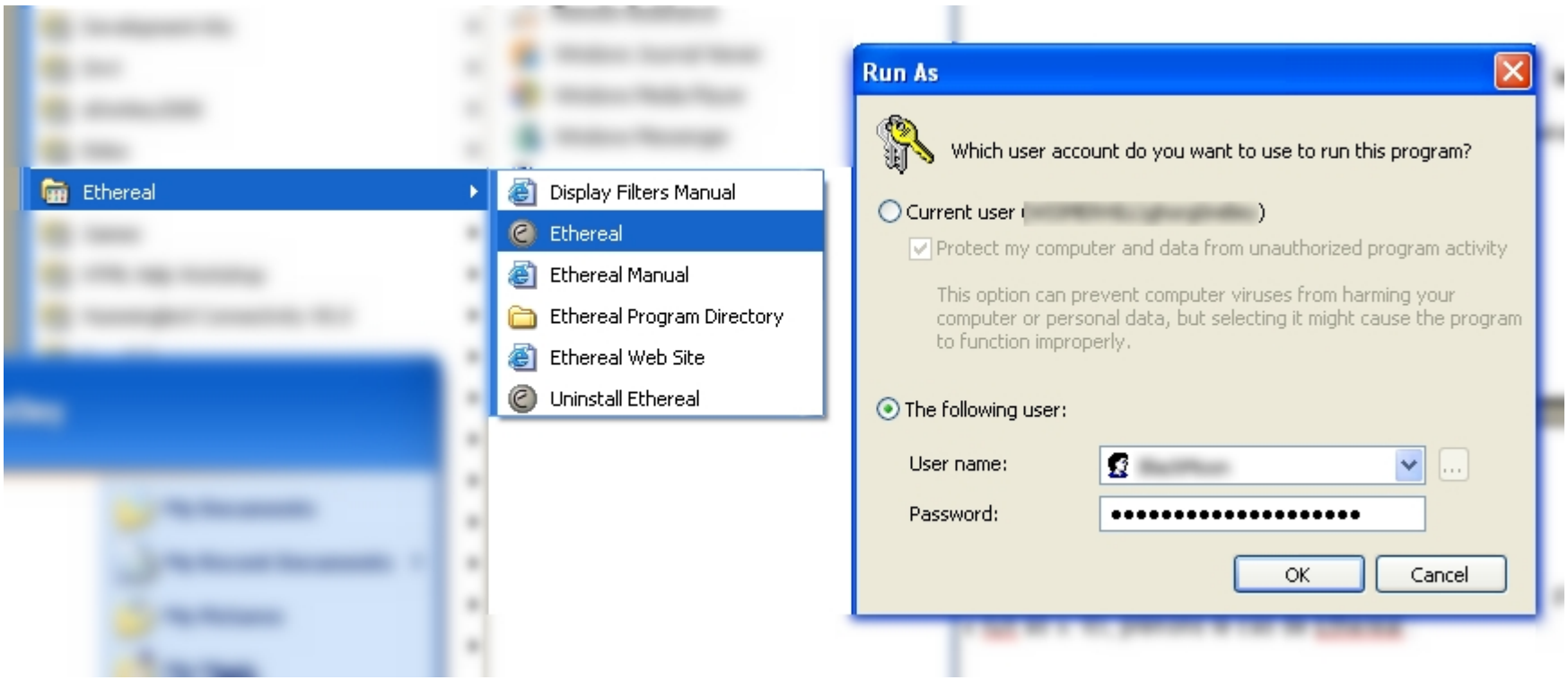
“cmd”, le contrôle de l’hôte (2)

- L’upload/download de fichiers



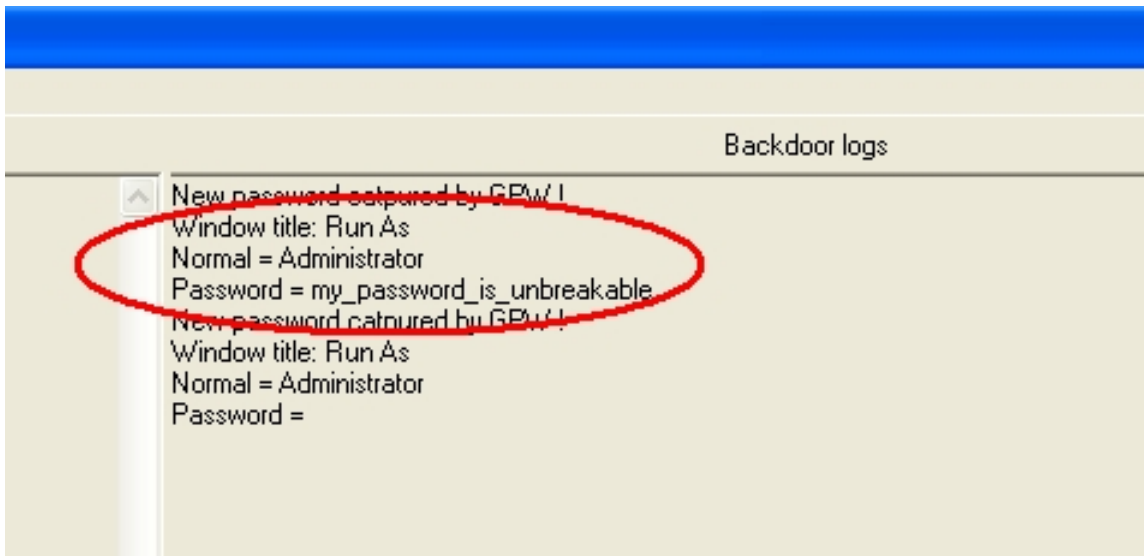
“gpw”, l’interception de mots de passe (1)

- Module de récupération de mots de passe
  Fonctionne sur une technique de wide-system hook



“gpw”, l’interception de mots de passe (2)

- Récupération du mot de passe sur BlackMoon



“scan”, la découverte du réseau interne

- Scanner de ports:
  - TCP connect()
  - UDP
  - ICMP
  - Raw TCP

- : Moins puissant que nmap (pas de fingerprint)

+: Scan depuis une machine protégée par FW personnel

L’architecture de propagation (1)

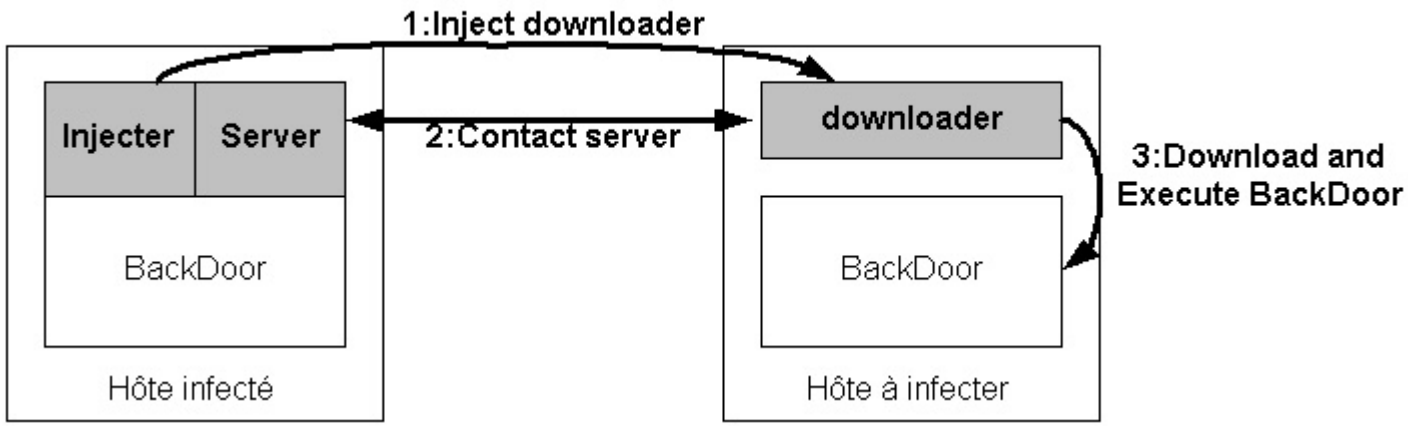
- BlackMoon construite pour gérer multi-backdoor
- Infection manuelle de plusieurs hôtes est fastidieuse

  => Necessité d’un mécanisme automatique
- Mécanisme doit être au maximum indépendant de la méthode de propagation

  => Mécanisme découpé en trois parties:
  - « downloader » télécharge backdoor depuis un serveur et l’exécute
  - « server » attend la requête du downloader
  - « injecter » permet l’exécution de « downloader » sur un hôte à infecter

L’architecture de propagation (2)

- Principe de l’infection d’un nouvel hôte



- Toutes les spécificités liées à un mécanisme de propagation sont alors confinées dans l’« injecter »

L’architecture de propagation (3)

- Présentation de “downloader”
  - Principe:
    - Récupération des adresses de fonctions (PEB)
    - Téléchargement de la backdoor (fonction URLDownloadToFile sur l’URL http://[serveur infecté] :[SRV PORT])
    - Exécution de la backdoor (WinExec sur le fichier téléchargé)
  - Caractéristiques:
    - Code en assembleur
    - Taille ~ 300 octets
    - Supporte la relocalisation
    - Pas d’adresses hardcodées => Portables sur toutes versions de Windows
    - Variantes: pas d’octets nuls, sleep infini à la fin,...

L’architecture de propagation (4)

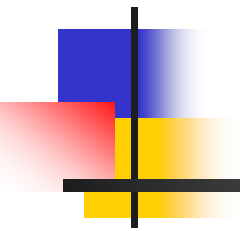
- Présentation de “srv”
  - Principe:
    - Faux serveur Web écoutant sur le port [SRV PORT]
    - Attend la requête d’un downloader et renvoie la backdoor
  - Caractéristiques:
    - Développé sous forme de module

L’architecture de propagation (5)

- “vrs” est un exemple d’injecter.
- Nécessité de trouver un mécanisme viable dans le temps.
- Présentation de “vrs”
  - Principe:
    - Recherche des disques partagés montés en RW.
    - Recherche récursive d’exécutables sur ces disques.
    - Infection sans augmentation de taille avec le « downloader ».
  - Caractéristiques:
    - Développé sous forme de module.

Possibilités des backdoors: Conclusion

- Modules fournis donne un apperçu des possibilités:
  - Récupération de mots de passe, découverte du réseau, récupération de fichiers, remplacement de fichiers, propagation sur réseau interne,...
  - On peut imaginer de nombreux autres modules:
    - Capture de mots de passes sur le réseau
    - Recherche et envoi automatique de fichiers .doc, .ppt,...
- Possibilités des backdoors illimitées !



Détection et contre-mesures (1)

Dispositifs de détection sur le client

Antivirus

- Multiples techniques de détection:
  - Analyse de signatures
  - Analyse heuristique statique
  - Analyse comportementale
- Mais problèmes des faux-positifs
- Tests:

	Fratus	Parsifal
Viruscan Enterprise 7 McAfee	NON	NON
AVG (free version)	NON	NON
Kaspersky Anti-virus Personal Pro 5.0	NON	NON

- Conclusion:

  Antivirus adaptés face aux grandes vagues virales, mais inefficaces contre développements spécifiques comme Fratus et Parsifal

Firewall personnel

- Filtrage d'accès basé sur la notion d'application
- Cas de Windows XP SP2: un "pseudo" FW personnel
- Cas des autres FWs
  - Fratus détecté, mais problème de la réaction de l'utilisateur
  - Parsifal non détecté

	Fratus	Parsifal
FW personnel Windows XP SP2	NON	NON
ZoneAlarm (free)	OUI	NON
Sygate Pro (trial)	OUI	NON
Kaspersky Anti-Hacker 1.7 (trial)	OUI	NON

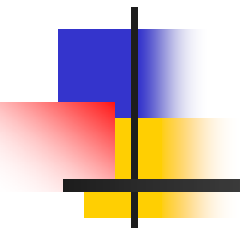
- Conclusion:

  Fratus: sécurité repose sur la réaction de l'utilisateur

  Parsifal: non détecté

Solutions complètes

- Test avec Entercept de McAfee
- Configuration ???
  - Tout flux interdit TCP
  - Autorisation de HTTP, HTTPS et DNS pour Internet Explorer
- Résultats:
  - Fratus: bloqué (indépendant de l'utilisateur)
  - Parsifal: non détecté



Détection et contre-mesures (2)

Dispositifs de détection sur le proxy

Problématiques associées (1)

- Type de dispositifs amène plusieurs questions:
  - Emplacement de la sonde
  - Critères utilisés pour distinguer un flux légitime d'un flux pirate
- Emplacement de la sonde
  - Avant

    => @IP destination est celle du proxy

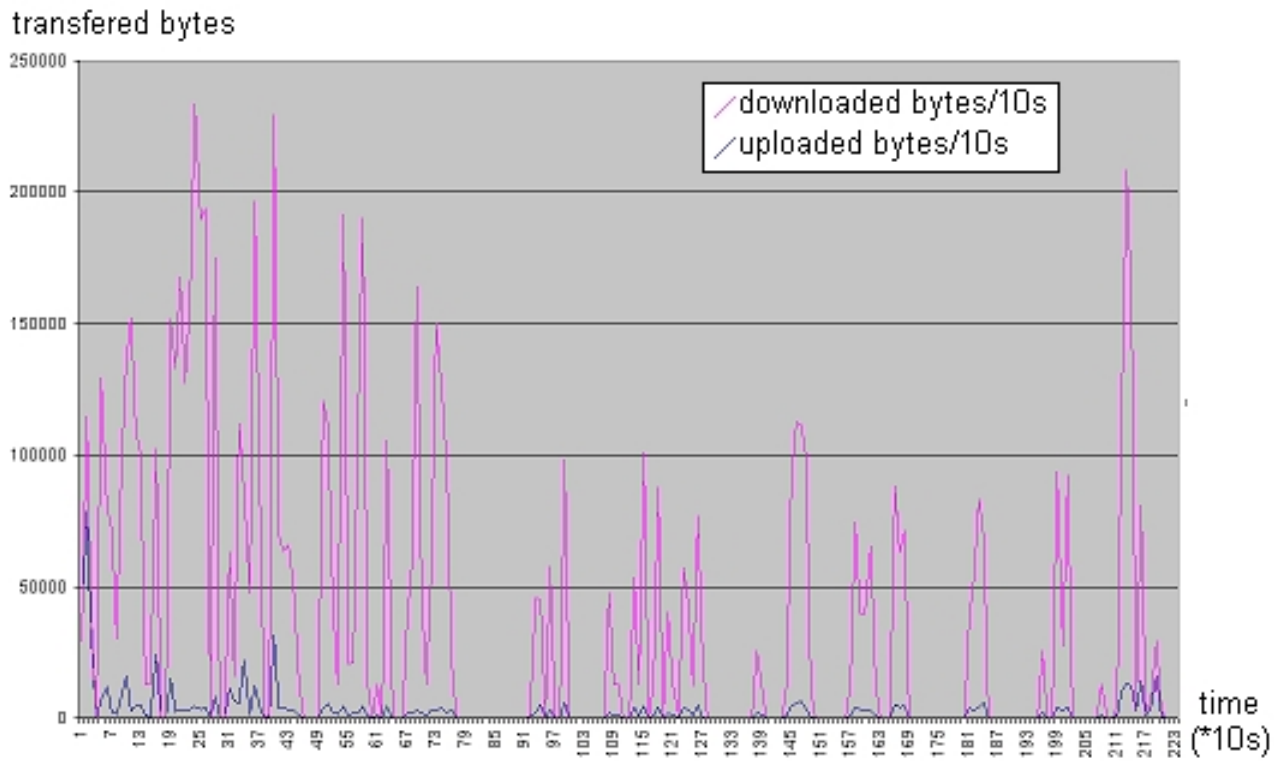
    => Nécessité d'une analyse multi-couches
  - Après

    => @IP source est celle du proxy

    => Nécessité de découvrir 1 flux parmis ceux de N clients !
  - Conclusion:
    - Sonde doit analyser les flux avant le proxy
    - Possibilité d'analyser avant et après puis de corréler les résultats

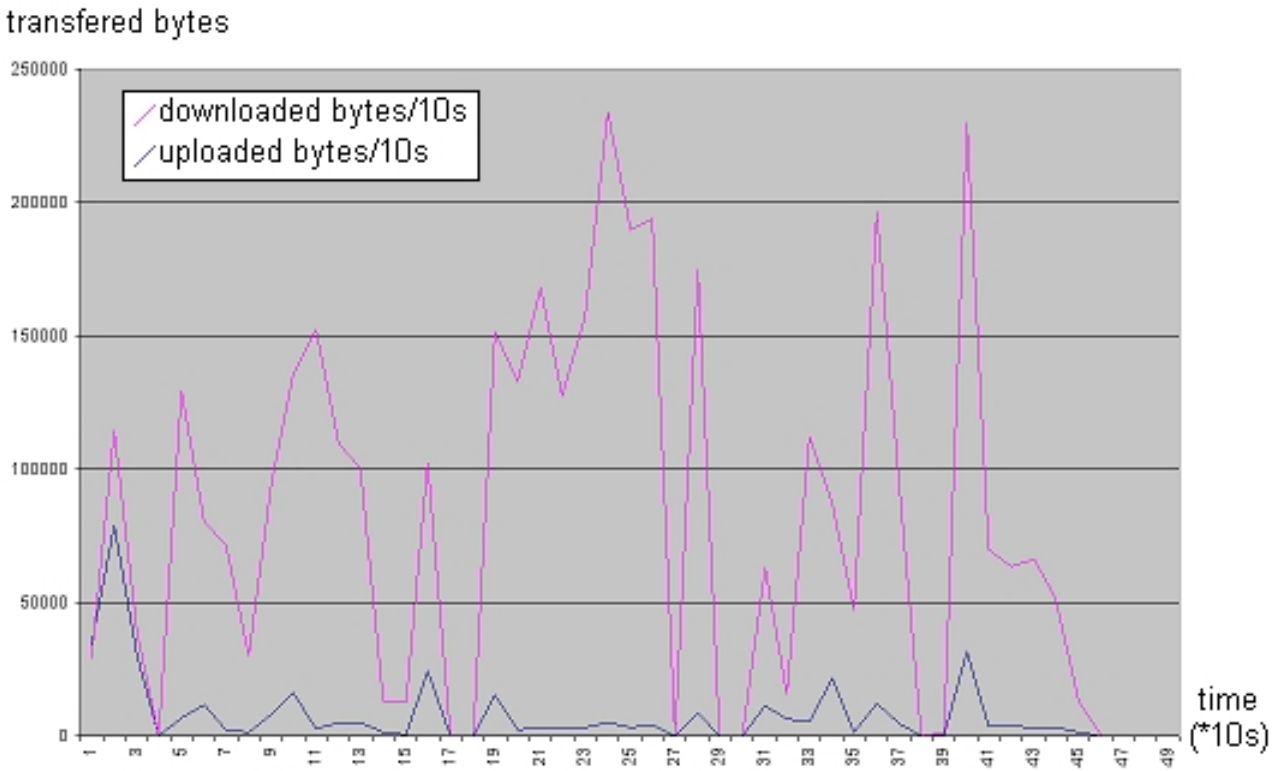
Problématiques associées (2)

- Allure du trafic généré par des accès au web



Problématiques associées (3)

- Allure du trafic généré par des accès au web (zoom)

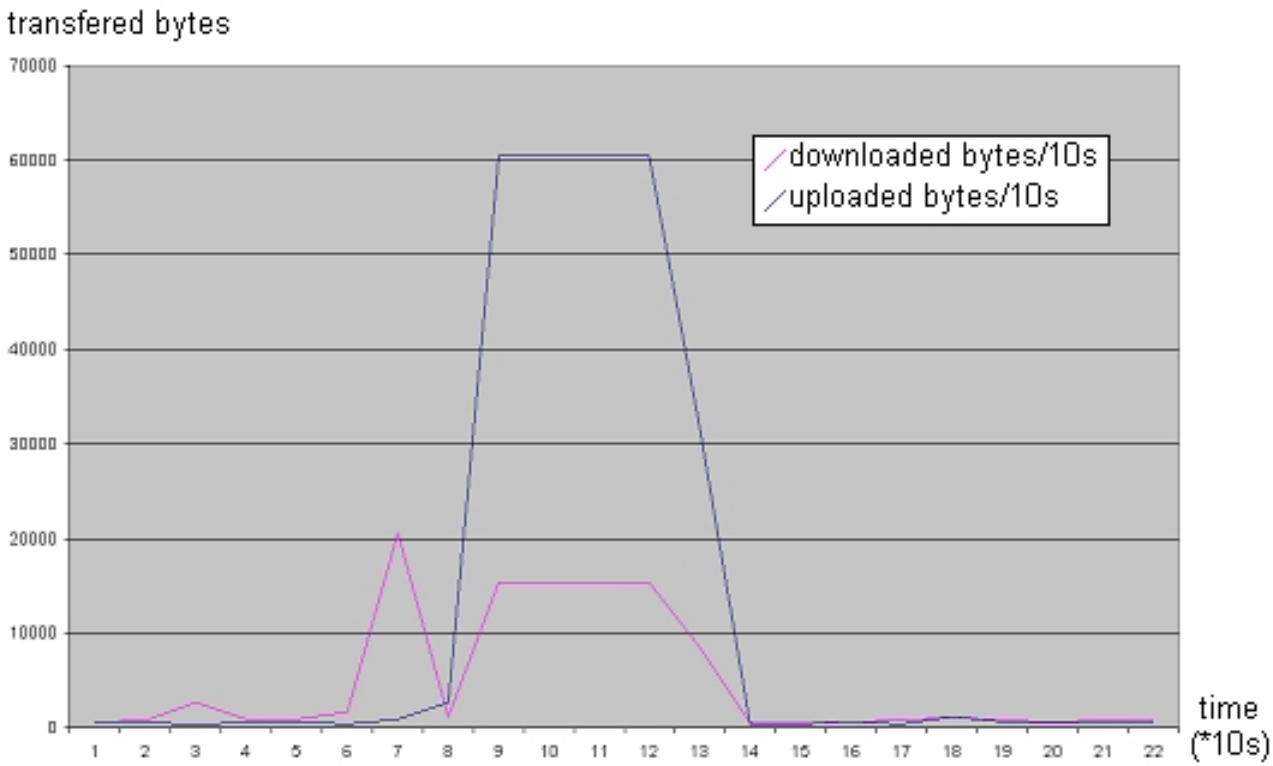


Problématiques associées (4)

- Critères de détection
  - Au niveau réseau
    - La durée des connexions
    - La régularité (quantité de données, durée entre les requêtes)
    - Le rapport upload/download
  - Au niveau applicatif
    - L'utilisation répétée de requêtes POST
    - L'accès répété à un serveur
    - L'accès répété à une même URL

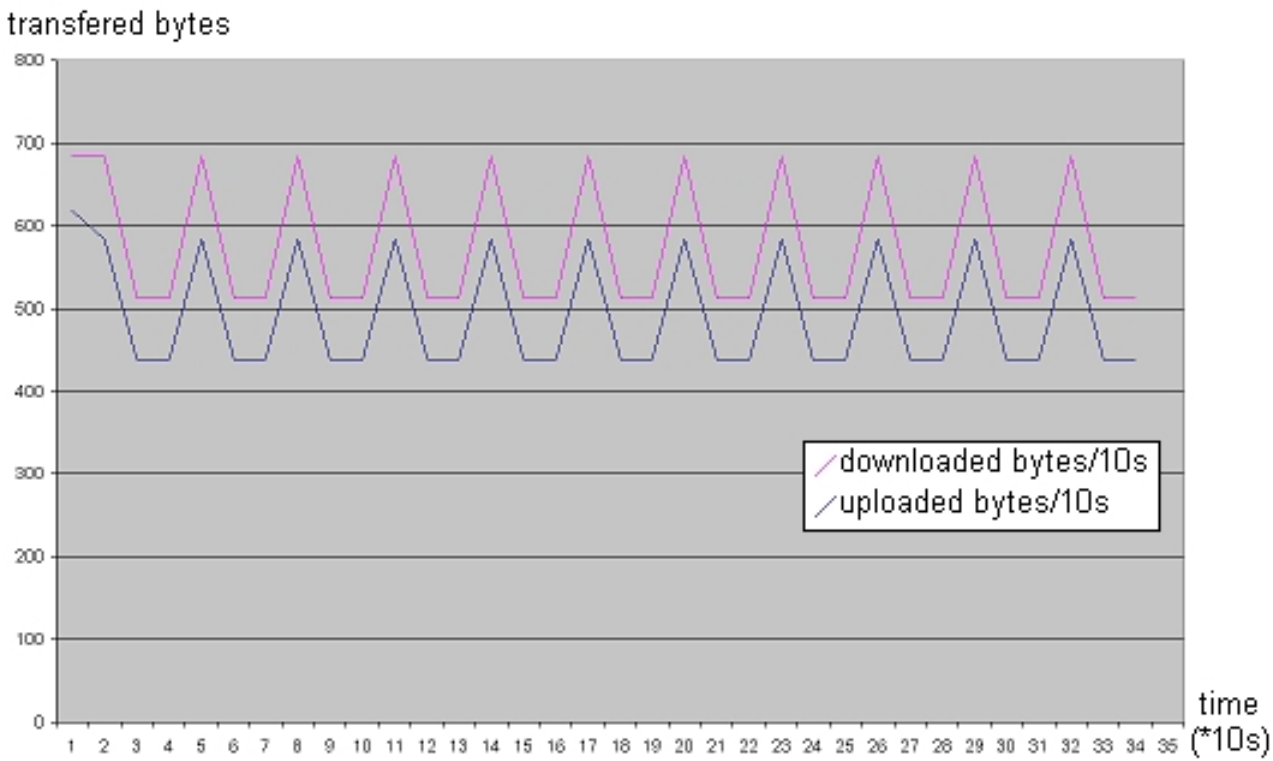
Sans mécanisme de furtivité (1)

- Allure du trafic généré par les backdoors



Sans mécanisme de furtivité (2)

- Allure du trafic généré par les backdoors (zoom)



Sans mécanisme de furtivité (3)

- Traces générées sur le proxy

  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html
  TCP_MISS/200 146 GET http://XX.XX.XX.XX/ - DIRECT/XX.XX.XX.XX text/html

- Conclusion

  Sans mécanisme supplémentaire allure du trafic généré par les backdoors très différente de celle du trafic généré par des accès web.

Mécanismes de furtivité – Niveau réseau

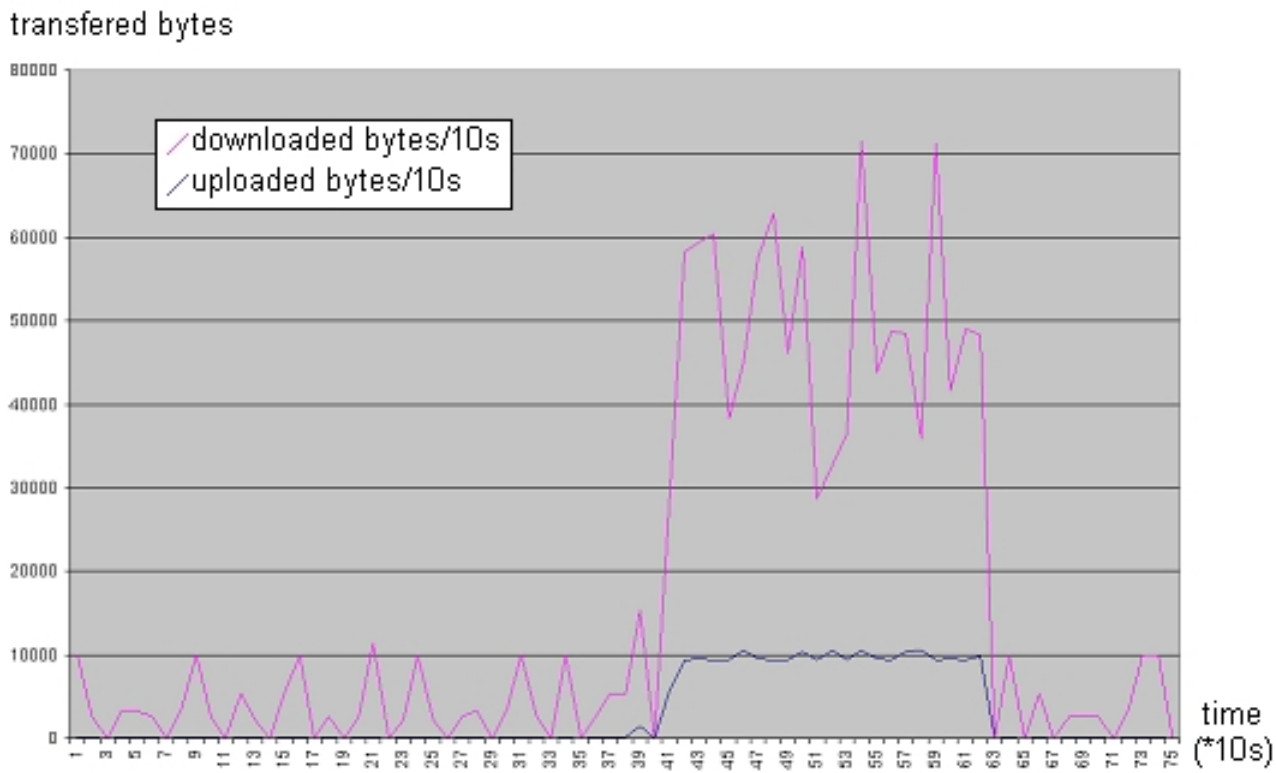
- Limitation des temps de connexion:
  - Deux time-out sur BlackMoon : recv time-out et max life-time.
  - Lors de l'expiration, la connexion est fermée.
  - Aucune perturbation, même au milieu téléchargement.
- Régularité des requêtes (fonctionnement polling).
  - Introduction d'un paramètre aléatoire pour faire varier l'attente entre deux requêtes.
- Rapport Upload/Download:
  - Certaines requêtes envoyées par BlackMoon sont vides:
    - Comportement anormal pour un serveur WEB.
    - Provoque un rapport Upload/Download suspect.
  - Renvoie de pages HTML inutiles prises de manière aléatoire dans le répertoire "trash."

Mécanismes de furtivité – Niveau protocole

- Randomization des URLs
  - Ajout d'un générateur dans la backdoor:
    - Complexe
    - Nécessite beaucoup de chaînes de caractères => augmente taille
  - Solution: Extraire les liens des pages "inutiles"
  - Avantages:
    - Simple à mettre en oeuvre, augmentation de la taille de la backdoor faible
    - Qualité de la randomization dépend du nombre de pages inutiles
    - Comportement proche navigateur: Reçoit URL puis suit lien
- Randomization des adresses des serveurs
  - Voir plus loin
- Choix du protocole
  - Choix "intelligent": GET et HTTPS

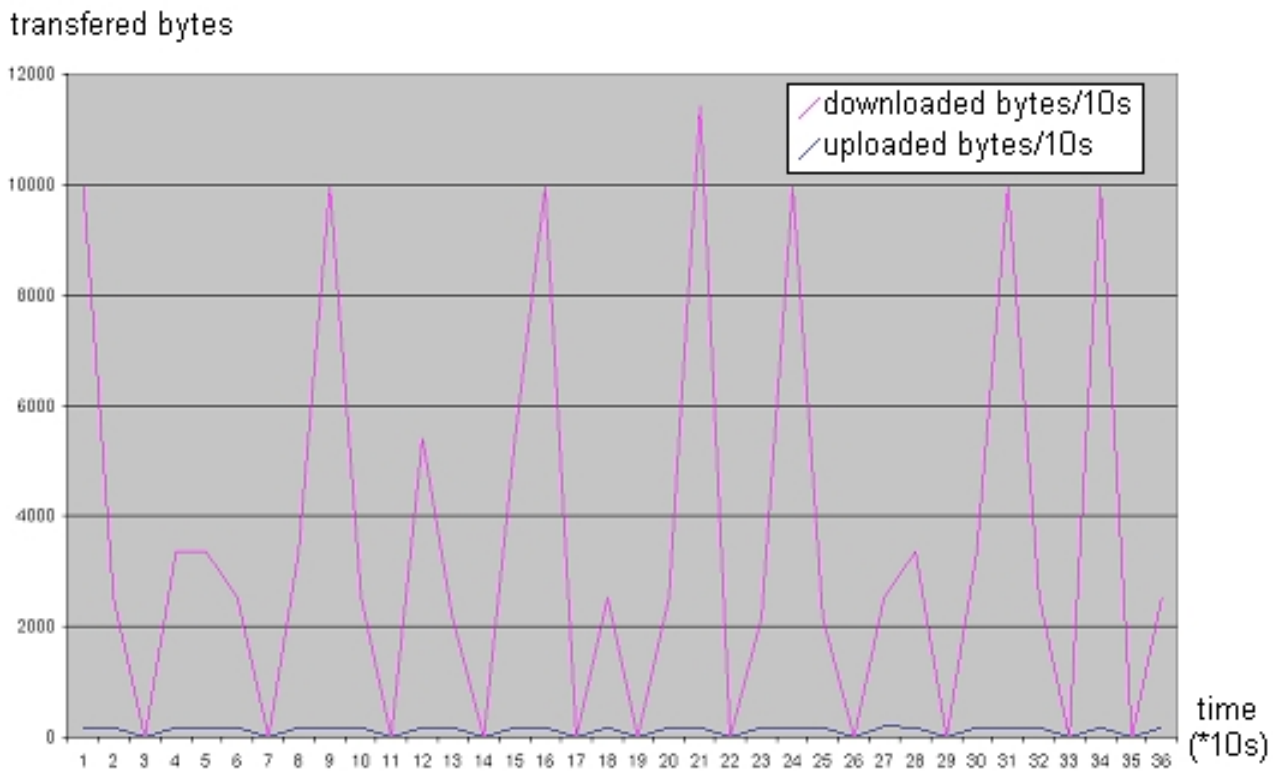
Mécanismes de furtivité – Résultats (1)

- Allure du trafic généré par les backdoors



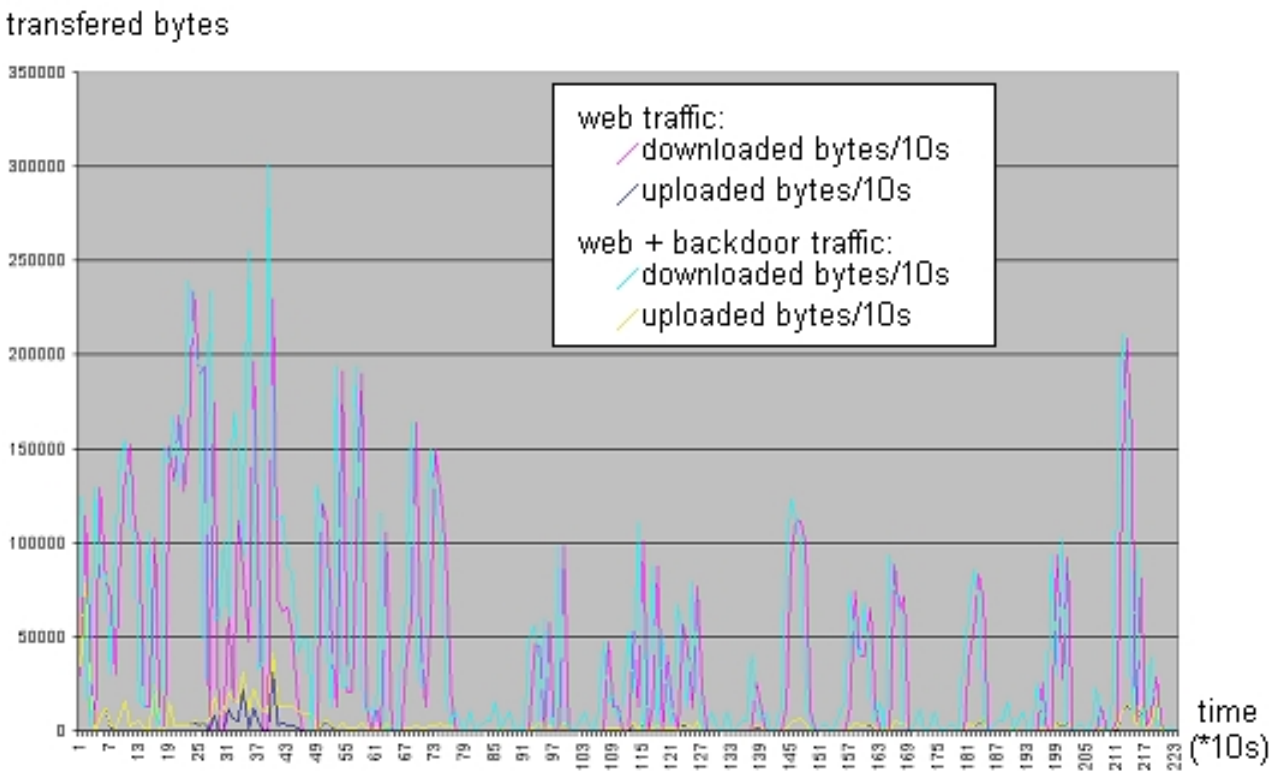
Mécanismes de furtivité – Résultats (2)

- Allure du trafic généré par les backdoors (zoom)



Mécanismes de furtivité – Résultats (3)

- Allure du trafic maintenant très proche du trafic web
- De plus, trafic “noyé” dans le trafic légitime



Tests avec détecteurs

- Constat: très peu de produits de ce type.
- Deux logiciels ~ projets en cours plutôt que produits finalisés:
  - Cctde : un freeware développé par Simon Castro qui fonctionne de pair avec snort.
  - Tcpstatflow : un freeware développé par « Fryxar ».
- Actuellement ces logiciels basés sur des seuils:
  - La durée des connexions.
  - La quantité de données transférées.
  - Le rapport upload/download.
- Conclusion
  - Critères insuffisants pour détecter les flux des backdoors lorsque mécanismes de furtivité activés.

Détection et contre-mesures - Conclusion

- Les mécanismes de détection/contrôle “classiques” sur le poste client ou sur le proxy sont insuffisants.
- Au niveau du poste client:

  Mécanisme d’analyse comportemental au niveau noyau, couplé avec un FW.

  => Complexe, problème de stabilité, d’adminstration
- Au niveau du proxy:

  Sonde analysant les flux avant et après le proxy aux niveaux 3, 4 et 7 et effectuant une consolidation de ces données.

  => Complexe

L’analyse post-attaque

A la recherche du pirate...

L’analyse post-attaque

- Principe de l’analyse post-attaque.
  - Intervient après la découverte et la gestion d’une attaque.
  - Consiste en l’analyse des différents éléments du système d’information afin de déterminer :
    - si une attaque a bien eu lieu.
    - l’étendue de l’attaque (les équipements impactés/modifiés).
    - le scénario de l’attaque (la ou les failles exploitées).
    - la cible de l’attaque (documents, données, ...).
    - l’origine de l’attaque (accumulation de preuves en vue de poursuites judiciaires).
- Cette partie se concentre sur ce dernier point.

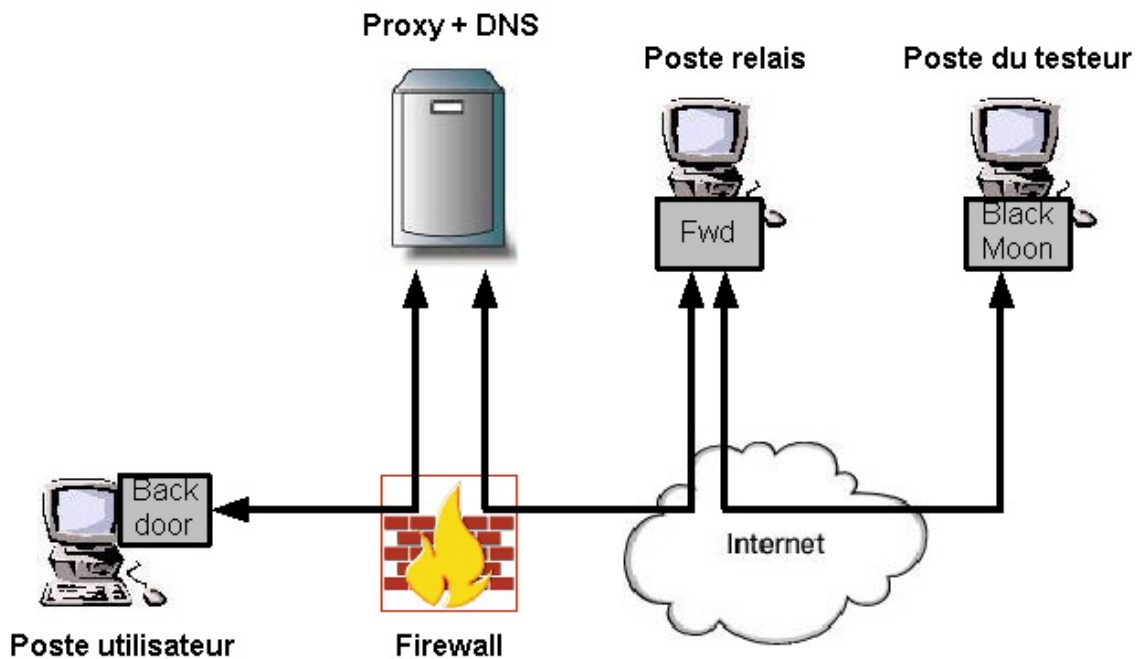
La collecte des traces

- Traces laissées/générées par les backdoors
  - Au niveau de l'hôte: la backdoor
  - Au niveau du proxy: traces des requêtes utilisées pour canaux
  - Au niveau des routeurs: traces des paquets
- Analyse des traces:
  - Traces sur routeurs à priori insuffisantes
  - Backdoor contient l'adresse de BlackMoon « hardcodée »
  - Logs sur le proxy contiennent l'adresse de BlackMoon

  => Necéssité que ni la backdoor ni les requêtes ne contiennent l'adresse de BlackMoon

Le principe de relais

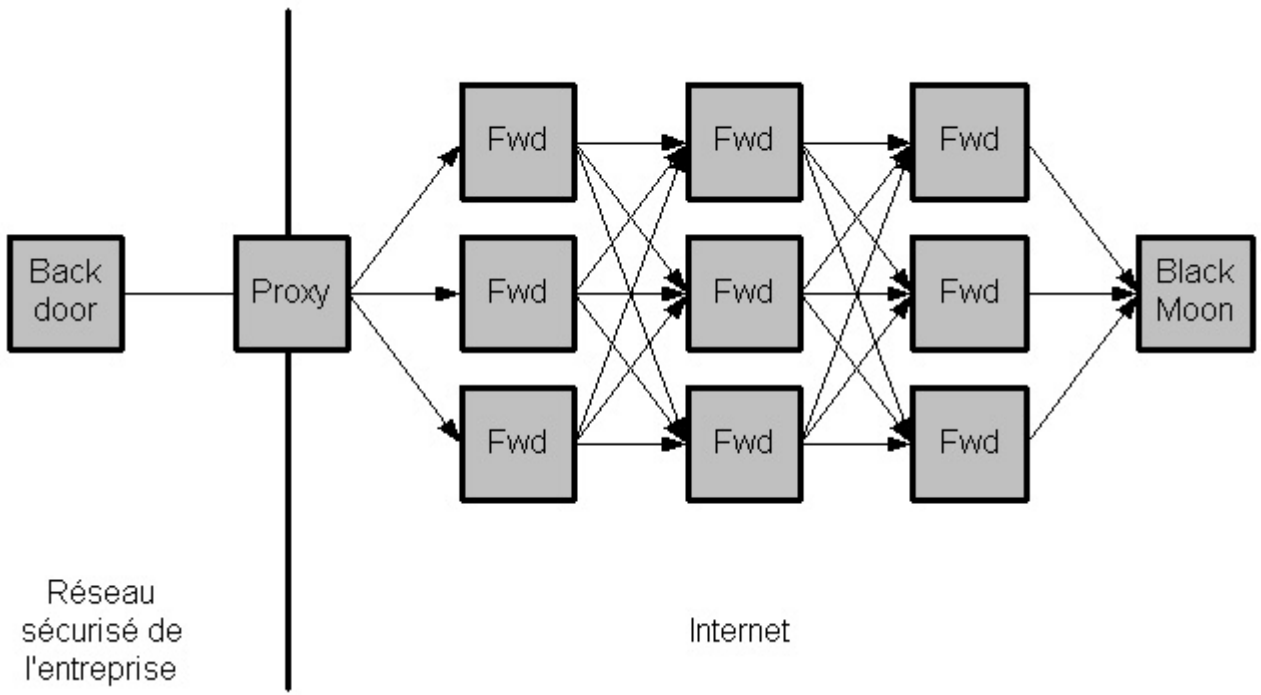
- Une solution consiste à intercaler un relais (“forwarder”) entre le proxy et BlackMoon



=> Adresse hardcodée et loggée est celle du relais

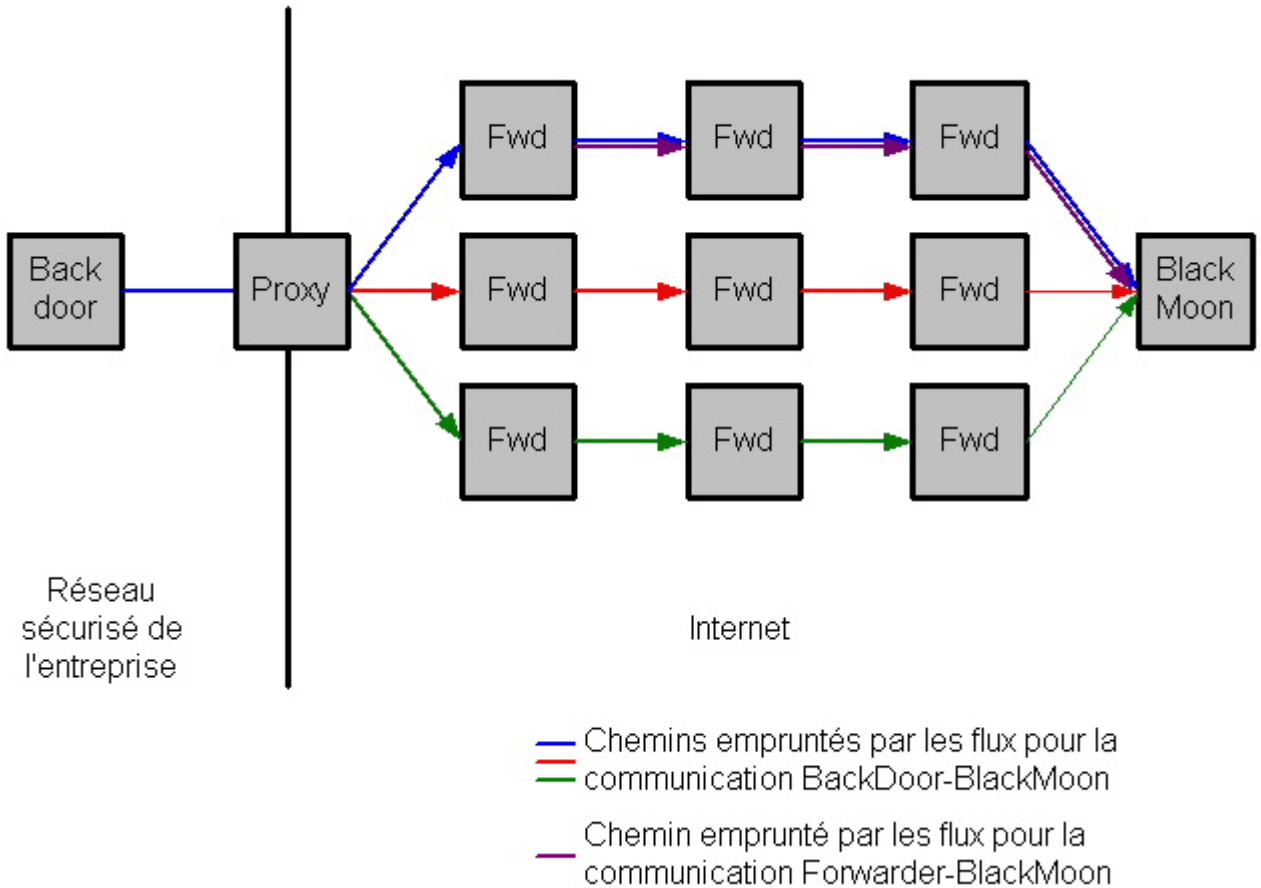
Généralisation du concept (1)

- Possibilité de chaîner plusieurs forwarders
- Problème: Aucun lien donc aucun contrôle physique du forwarder => Nécessité de redondance
- Architecture logique finale



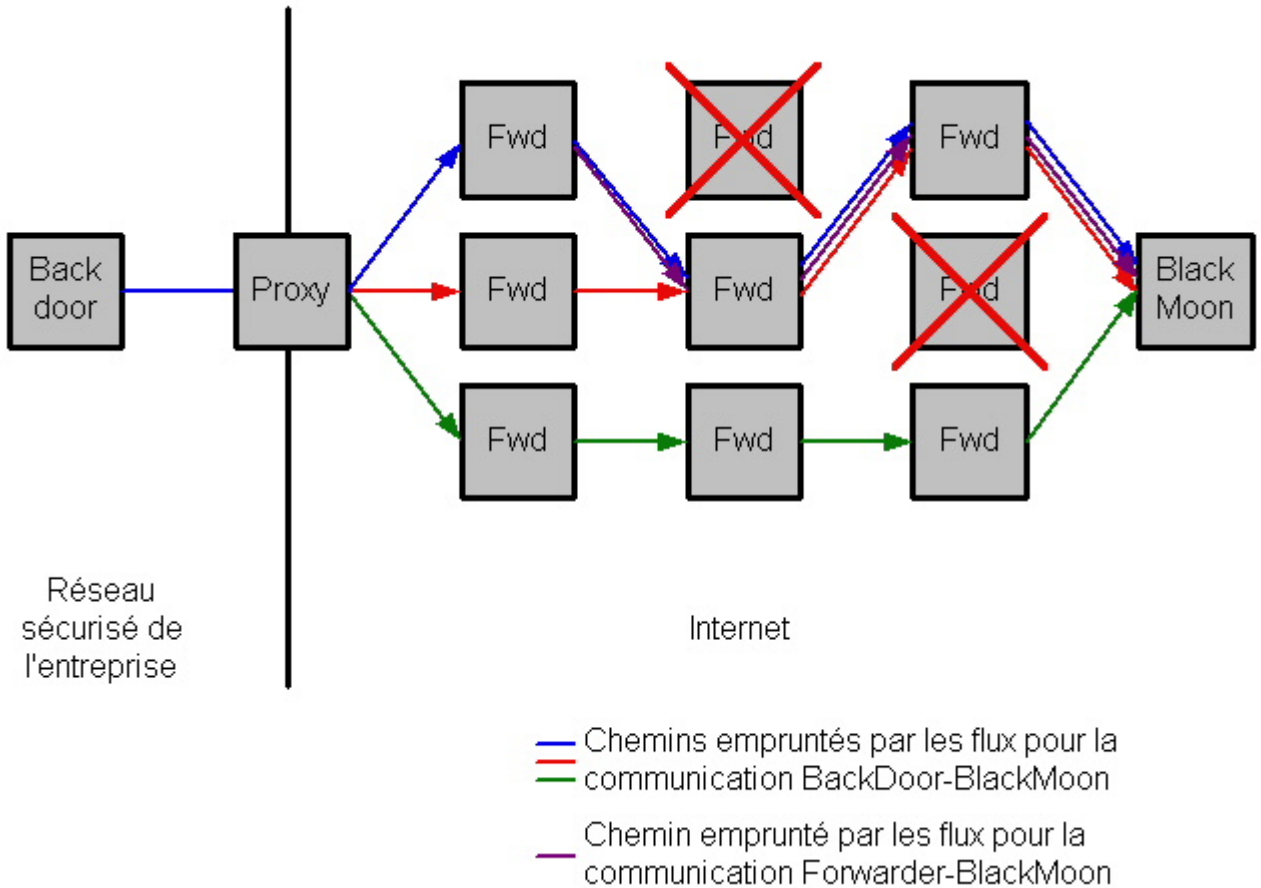
Généralisation du concept (2)

- Fonctionnement en mode normal



Généralisation du concept (3)

- Fonctionnement après arrêt de deux forwarders



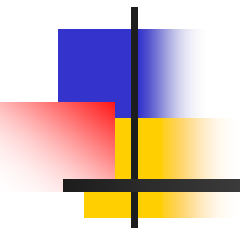
L’implémentation des relais

- Les relais sont implémentés sous forme de module :
  - Nom: le module “fwd”.
  - Développement rapide et portable sur les deux backdoors.
  - Backdoor installée sur poste relais => Possibilité de “nettoyer” le relais une fois l’attaque terminée.

Qui sont les forwarders ?

- Création de forwarders confrontée aux problèmes :
  - Trouver des machines n'ayant aucun lien avec le testeur.
  - Trouver des machines restant connectées à Internet pendant de longues durées.
  - Trouver un moyen d'infecter ces machines.
- Une solution: les réseaux peer to peer:
  - Machines restent connectées pendant de longues périodes.
  - Concept du peer to peer est l'échange de fichiers, donc la possibilité de transférer des fichiers sur d'autres machines.
- Une méthode de création de forwarder est donc l'injection de "fake" dans les réseaux peer to peer.

  => Outre le problème de violation des droits de protection des logiciels, les réseaux peer to peer représentent à ce titre un véritable danger.

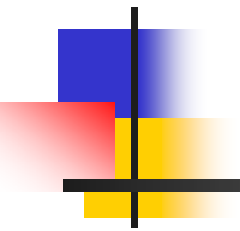


Conclusion générale

Le constat final...

Conclusion générale

- Architectures « classiques » avec accès au web insuffisantes pour empêcher évasion de données.
- Possibilités de la backdoor quasiment illimitées.
- Mécanismes de détection « classiques » inefficaces.
- Ils doivent tout de même être mis en œuvre pour détecter les implémentations simplistes.
- Mais doivent être complétés par mesures permettant :
  - d'éviter au maximum que la backdoor ne rentre sur le réseau.
  - de limiter au maximum les conséquences en cas d'intrusion.
- Situation guère satisfaisante. Dans un contexte d'espionnage industriel, espérons que des produits de détection efficaces verront prochainement le jour.



Questions / Remarques ?

Liens

- **MISC n°10,11,14**

  Une très bonne série d'articles de Valgasu sur le principe d'injection et d'API hooking

- **http://www.lsdp.net/~lotfree/doc/HTTP/tunneling_http.html**

  Un article sur le tunneling et les canaux cachés au sein du protocole HTTP

- **http://www.gray-world.net/projects/papers/html/cctde.html**

  La base « théorique » de cctde

  Soft disponible sur http://www.gray-world.net/pr_cctde.shtml

- **http://www.geocities.com/fryxar/**

  Page de téléchargement de tcpstatflow