Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
PREMIER MINISTRE

S . G . D . S . N
Agence nationale de la sécurité des systèmes d'information
CERTA

Paris, le 21 février 2000
N° CERTA-2000-INF-001

Affaire suivie par :
CERTA

NOTE D'INFORMATION DU CERTA

Objet : Le déni de service distribué

Gestion du document

Référence	CERTA-2000-INF-001-1.1
Titre	Le déni de service distribué
Date de la première version	21 février 2000
Date de la dernière version	19 juin 2000
Source(s)	
Pièce(s) jointe(s)	Aucune

TAB. 1 – *gestion du document*

Une gestion de version détaillée se trouve à la fin de ce document.

1 Introduction

La récente vague d'attaques dont ont été victimes un certain nombre de grands sites américains (« Yahoo ! » le 07/02/2000, « Buy.com », « Stamps.com », « eBay », « CNN », « Amazon » et « MSN » le 08/02/200, « ZDNet » et « E*Trade » le 09/02/2000) correspond à des attaques de type déni de service (Denial of Service ou DoS).

Ce document, de type Note d'information, a pour but de présenter ce type d'attaque, de fournir un certain nombre de pointeurs sur des documents de référence et de donner quelques pistes pour tenter de se protéger contre de telles attaques.

1.1 Le principe des attaques

Ces attaques visent à rendre inutilisable un site par une saturation de requêtes (la cible n'est pas compromise) rendue possible par la mise à contribution de centaines de machines piratées à cet effet. Les attaquants recherchent sur le réseau, au moyen d'outils classiques d'analyse, des machines présentant des failles connues qui permettent d'obtenir le privilège administrateur (root) sur ces machines (on peut penser que les campagnes de scans, qui se sont déroulées depuis la fin de l'été sur Internet, avaient pour but de recenser les machines présentant de telles faiblesses).

Une fois le privilège administrateur obtenu, les attaquants installent ensuite sur les machines compromises un logiciel d'attaque, de type client/serveur, qui est piloté depuis une machine unique sous leur contrôle. Le but du jeu étant de disposer de plusieurs centaines de machines disposant de ce logiciel d'attaque. Les outils d'attaque de

type déni de service distribué (DDoS, Distributed Denial of Service) sont disponibles sur Internet et certains ont fait l'objet d'une mise à jour récente.

Au moment choisi, il ne reste plus qu'à déclencher une attaque vers une cible unique : cela peut être un envoi massif de mèls, de demandes de recherche d'information (« Yahoo ! »), de commandes (« Amazon »), etc. Les serveurs attaqués sont alors noyés sous un flux d'information extrêmement important (un milliard de bits par seconde dans le cas de « Yahoo ! ») et ne sont plus capables de remplir leur tâche principale : c'est le déni de service.

1.2 Les parades

Il n'existe malheureusement pas de parade pour ce genre d'attaque, en effet on peut facilement détecter une attaque provenant d'une machine unique (correspondant à une même adresse) et bloquer le flux d'information en provenance de cette machine, mais il est très difficile de distinguer, lorsque le flux est réparti sur des centaines de machines, une attaque d'une demande de connexion en provenance d'un client réel. Si le logiciel d'attaque est bien fait, il génère des requêtes qui sont toutes différentes ce qui rend encore plus difficile la détection de l'attaque.

Comme l'a dit Thomas Longstaff de l'université de Carnegie Mellon :

> « *En réalité, la prévention doit plus porter sur le renforcement du niveau de sécurité des machines connectées au réseau [pour éviter qu'une machine puisse être compromise] que sur la protection des machines cibles [les serveurs Web]* ».

Il est indispensable de vérifier régulièrement que les machines ne présentent pas de failles de sécurité et ne pourront pas être utilisées en rebond dans une attaque. Il existe des outils commerciaux et du domaine public qui permettent d'effectuer de telles recherches (ne pas hésiter à contacter le CERTA pour obtenir des informations sur ces outils).

Les recommandations **CERTA-1999-REC-001** du 8 décembre 1999 restent bien sûr d'actualité, surtout celle concernant le recueil des traces : « *veiller au recueil et à la conservation de toutes les traces liées aux incidents et susceptibles de servir de preuves ultérieurement. Les journaux doivent naturellement être activés et protégés* ».

2 Systèmes affectés

2.1 Les systèmes vulnérables au déni de service

Tous les systèmes reliés à Internet sont des victimes potentielles du déni de service. Les services attaqués (courrier électronique, WEB, ICMP, etc.) sont universels.

2.2 Les systèmes vulnérables au rebond

Les systèmes suivants sont connus comme pouvant être des machines servant d'intermédiaires dans les attaques de déni de service :

- Certains Macintosh sous *MacOS 9* peuvent servir d'amplificateur de trafic. Une nouvelle version d'*Open Transport (2.6)* a été publiée pour corriger cette faille, cette nouvelle version est disponible à l'adresse suivante :
  http://asu.info.apple.com/swupdates.nsf/artnum/n11560
- Les machines Unix sont sensibles à trinoo, TFN (Tribe Flood Network), TFN2K (Tribe Flood Network 2000) et Stacheldraht ;
- Les machines sous Windows sont sensibles à TFN2K.

3 Comment détecter une attaque ?

3.1 Détecter un déni de service

Un déni de service présente sensiblement les mêmes symptômes que ceux d'un système dont le réglage n'est pas adapté à la charge en présence d'un pic d'utilisation légitime. Si après s'être assuré que les serveurs et le réseau sont correctement configurés, on constate que :

- les serveurs (messagerie, WEB, routeur, ...) sont soumis à une charge considérablement au-dessus de la moyenne ;
- le réseau véhicule un trafic anormalement important ;

– on peut alors raisonnablement penser à un déni de service.

3.2 Détecter une compromission

A l'instar des outils de détection de virus :

– tout moyen de détection de déni de service est rapidement obsolète à mesure que les sources des outils d'attaques sont modifiées ;
– un seul examen des systèmes ne suffit pas. Il convient de chercher régulièrement si les systèmes ne sont pas infectés.

3.3 Outils de détection

Le tableau suivant décrit plusieurs outils qui aident à détecter les machines compromises par certains logiciels d'attaque. Ce sont des outils de balayage de réseau (scan) pour y trouver des traces de compromissions. L'OS évoqué concerne uniquement la machine d'analyse et non pas les machines compromises.

Moyen	trinoo	TFN	TFN2K	tachelDraht
find_ddos	X	X	X	
dds (Distributed DoS tool Scanner)	X	X		X
gag (sickenscan)	X	X	X	
RID (Remote Intrusion Detection)	X	X		X

TAB. 2 – *Moyens de détection des outils de déni de service distribués*

find_ddos logiciel disponible (attention ! uniquement sous forme binaire pour Solaris avec processeur Sparc et Intel, ainsi que pour Linux pour processeur Intel) sur le site
http://www.fbi.org/nipc/trinoo.htm ;

dds logiciel disponible sous forme de source en C sur le site
http://staff.washington.edu/dittrich/misc/ddos pour les systèmes suivants :

– Linux (kernel 2.2.x) ;
– Solaris 2.6 ou plus ;
– Digital Unix 4.0 ;
– IBM AIX 4.2 ;
– FreeBSD 3.3-Release ;
– OpenBSD 2.6.

gag logiciel disponible sous la forme de source C sur le site
http://staff.washington.edu/dittrich/misc/ddos pour les systèmes suivants :

– Linux (kernel 2.2.x) ;
– Solaris 2.6 ou plus ;
– Digital Unix 4.0d ;
– IBM AIX 4 ;
– FreeBSD 3.3-Release.

RID logiciel disponible sous la forme de source C pour Solaris 2.7 sur le site
http://theoygroup.com/Software/RID .

3.4 L'approche manuelle

Une étude régulière de la configuration de chaque système peut aider à trouver les logiciels DDoS.

– les machines infectées ont probablement été configurées avec un rootkit. Il est possible de reconnaître des

programmes infectés en cherchant des chaînes de caractères spécifiques à l'aide de la commandestrings ou bien en étudiant les paquets échangés. Les documents suivants expliquent ce qu'il faut rechercher :
  - http://www.sans.org/y2k/stacheldraht.htm ;
  - http://www.cert.org/incident_notes/IN-99-07.html ;
  - http://staff.washington.edu/dittrich/misc/tfn.analysis .
- de manière générale la découverte de chevaux de Troie peut être effectuée aisément à l'aide d'outil commeTripwire mis en oeuvre *dès l'installation du système* ;
- les machines infectées sont susceptibles d'émettre des paquets IP dont l'adresse origine est falsifiée. Un tel trafic est le signe évident de l'utilisation malveillante d'une des machines du réseau interne. Si ce n'est pas déjà fait, il convient de configurer le *routeur* ou le *firewall* afin de détecter les paquets dont l'origine ne correspond pas à l'une des adresses du réseau interne ;
- les ports *1524*, *27665*, *27444*, *31335* et *37337* ont été utilisés pour prendre le contrôle à distance de machines infectées. L'utilisation régulière de logiciels commenessus ounmap permet d'identifier ports ouverts par les machines du réseau interne. L'apparition de nouveaux ports doit faire l'objet d'une enquête.

En cas de découvertes de l'intrusion en vue de commettre un déni de service, les recommandations décrites dans le document **CERTA-1999-REC-001** restent applicables.

4 Comment se protéger ?

D'une manière générale il est difficile de se protéger d'une attaque par DoS/DDoS. Néanmoins les règles élémentaires de sécurité complétées par des actions spécifiques permettent de minimiser les effets de ce type d'attaque.

4.1 Règles élémentaires

- Utilisez les dernières versions des logiciels : il est courant de posséder des versions obsolètes des systèmes d'exploitation ou des applications, possédant des trous de sécurité, exploités par les outils DDoS, qui ont depuis été résolus ;
- Souscrire à des listes de diffusion relatives à la sécurité. Tenir à jour les actions à entreprendre en cas d'urgence (conduite à tenir, téléphone..) ;
- Archivez et exploitez efficacement les journaux (stockage extérieur, régularité de l'opération) ;
- Invalidez les services inutiles, en particulier sur les machines bastions.

4.2 Règles spécifiques

- Configurez avec soin les serveurs Web : il est par exemple possible sur les serveursapache de réduire les délais d'expiration des connexions. Cette action aura pour effet d'atténuer les effets de saturation mais risque, dans le même temps, de faire échouer des connexions valides ;
- Multipliez les serveurs et les sites de façon à distribuer les requêtes entrantes : ce partage est réalisable par une méthode simple et efficace comme le *Round Robin DNS* ou par une solution hardware comme un répartiteur. Il est ainsi possible de configurer deux serveurs DNS, le second servant de secours au premier, de façon transparente, en cas de saturation (en prenant soin de dissocier clairement les adresses IP de ces serveurs) ;
- Paramétrez votre garde barrière afin d'interdire temporairement les adresses IP les plus actives à partir desquelles les attaques sont émises. Une machine *rebond* pourra souvent être réutilisée avant que son détenteur n'en soit informé et qu'il ne réagisse ...
- Paramétrez votre garde barrière afin d'interdire les paquets sortant dont la source n'est pas une adresse IP interne connue : ainsi, au mieux, vous interdirez à une de vos machine de servir de rebond et dans le pire des cas, la cible du DoS pourra identifier le réseau origine du rebond.

5 Outils d'attaque distribuée

5.1 Généralités

Par rapport aux outils traditionnels d'attaque, les outils distribués sont caractérisés par :

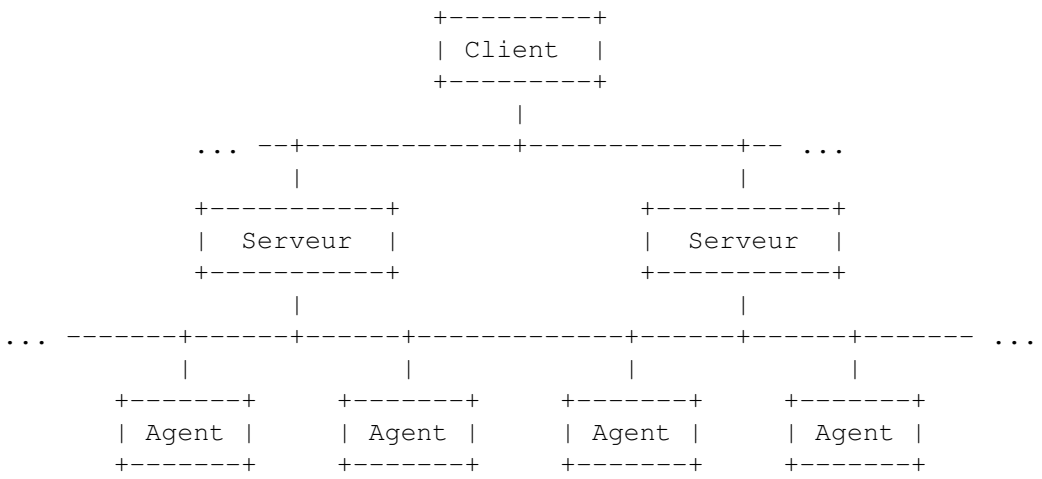
- une procédure d'installation de logiciels d'attaque (infection) fortement automatisée pour pouvoir traiter un

nombre conséquent de machine ;

– une architecture distribuée de type client/serveur semblable en cela aux logiciels d'administration ou de sécurité des réseaux modernes ;
– une attaque massive de déni de service, coordonnée, par les machines infectées contre un ou plusieurs sites.

L'efficacité du déni de service étant liée au nombre de machines compromises, le programme d'installation recherche (scan) et exploite une ou des failles liée(s) à des services installés de manière standard sur les stations : services RPC de Sun, serveur FTP, ... pour s'y introduire frauduleusement. De plus, un root kit est éventuellement installé modifiant les commandes usuelles telles `ls`, `ps`, `netstat`, ..., de manière à dissimuler les programmes et connexions réseaux illicites.

Dans les versions les plus récentes l'architecture comporte 3 types d'hôtes différents, le client, le(s) serveur(s) et les agents (terminologie du **CERT/CC**). Le schéma typique est le suivant :



L'analogie, utilisée par l'un des outils, entre les Agents et des feuilles permet bien d'imaginer l'organisation arborescente du réseau en assimilant les Serveurs à des branches.

Le client, utilisé par l'attaquant, contrôle un ou plusieurs Serveurs, lesquels communiquent avec les Agents chargés de la mise en oeuvre du déni de service. Pour éviter que les Agents et les Serveurs ne puissent être utilisés par autrui (administrateur légitime de l'hôte, autres pirates) toutes les commandes vers ces programmes nécessitent des mots de passe ou des informations supplémentaires.

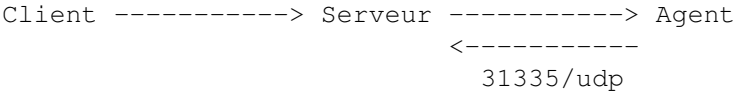
5.2 Détails sur quelques outils

Les divers programmes étant disponibles sous forme de code source, toutes les valeurs citées ci-dessous ne sont pas garanties : un pirate à la petite semaine compilera vraisemblablement les sources en l'état, mais il est à la portée de tout programmeur d'analyser le code (un peu commenté) et de modifier en conséquence les numéros de ports, les chaînes de commande, les mots de passe, ...

Pour analyse, les sources sont disponibles sur le serveur de packet storm [**?**].

5.3 trinoo

Les connexions entre les divers Agents sont résumées dans le schéma ci-dessous :



En l'absence de *rootkit*, `netstat` donne :

serveur

Protocole	Adresse locale
tcp	0.0.0.0:27665
udp	0.0.0.0:31335

Agent

Protocole	Adresse locale
tcp	0.0.0.0:27444

Les connexions sont réalisées en clair, par conséquent toutes les chaînes ci-dessous peuvent être identifiées par un *renifleur* de paquets.

5.3.1 Installation

Lorsqu'un Serveur est lancé sur une machine compromise, il réclame un premier mot de passe (`gOrave`, par défaut) et passe alors en tâche de fond. Le Serveur crée un fichier (appelé `...` par défaut) pour stocker le nom des agents que se seront fait connaître (il peut exister un second fichier `...-b` qui est une copie de secours créée après certaines commandes).

L'Agent, lorsqu'il démarre, cherche à contacter un certain nombre de serveurs (inscrits en dur dans le binaire) sur le port UDP 31335 en envoyant la chaîne `*HELLO*`. Si un Serveur est à l'écoute, il répond sur le port UDP 27444 de l'Agent par la commande `png 144adsl` (la commande `png` demande un accusé de réception à l'Agent et `144adsl` est le mot de passe par défaut entre le Serveur et l'Agent). L'Agent répond alors sur le même port que précédemment par la chaîne `PONG`. Cela termine la phase d'identification et son adresse est stockée chiffrée par le Serveur dans le fichier `...` à l'aide de l'algorithme symétrique BlowFish.

5.3.2 Attaque

Le Client se connecte sur le port TCP 27665 d'un Serveur, s'authentifie avec un mot de passe (`betaalmostdone` par défaut) et peut alors envoyer un certain nombre de commandes d'administration (liste des Agents identifiés et vérification qu'ils sont actifs, arrêt des Agents, ...). Il peut demander un déni de service sur une ou plusieurs adresses IP. Le Serveur contacte alors les Agents qui ouvrent un socket sur un port non privilégié et lancent une multitude de paquets UDP vers des ports aléatoires des cibles.

À priori, les Agents ne modifient pas l'adresse source des paquets, par conséquent les cibles seraient en mesure d'identifier les Agents.

Pour des informations plus complètes, voir trinoo.analysis de D. Dittrich (cite [**?**]).

5.4 TFN

```
        tout shell            echo_reply/icmp
Client ------------> Serveur -----------------> Agent
         distant             <-----------------
                              echo_reply/icmp
```

En l'absence de rootkit, netstat donne :

Serveur

Protocole	Adresse locale	Commentaire
raw	0.0.0.0:1	plus une éventuelle connexion telnet, SSH, ...

Agent

Protocole	Adresse locale
raw	0.0.0.0:1

Tout système possède, en standart, ce type de *socket* ouverte : ce n'est donc pas une caractéristique d'un système compromis.

5.4.1 Installation

Contrairement à trinoo la procédure d'installation est très simple :

- les *Agents* n'ont pas connaissance des *Serveurs*, donc pas d'étape d'identification ;
- les *Serveurs* ne gèrent pas non plus de fichier particulier pour stocker la liste des *Agents*, cette dernière leur étant fournie par le nom d'un fichier passé en argument.

5.4.2 Attaque

Le *Serveur* reçoit ses ordres sous forme de ligne de commande : le *Client* doit donc se connecter sur l'hôte avec un shell distant (en clair avec telnet, chiffré avec SSH, ...). Le *Serveur* ne nécessite pas de mots de passe, mais il faut systématiquement fournir un fichier des *Agents* dans la commande. Les ordres qui peuvent être transmis aux *Agents* sont :

- choix du masque pour falsifier les adresses sources (aléatoire, modification d'un, de deux ou de trois des octets de plus faible(s) poids de l'adresse de l'*Agent*) ;
- changement de la taille des paquets d'*attaque* ;
- attaques par saturation de requêtes UDP, TCP, ICMP sur plusieurs adresses ou smurf sur une seule (en précisant un ou des réseaux amplificateur) ;
- ouverture d'un shell distant sur un port donné par les hôtes des *Agents*.

La communication entre un Serveur et des Agents se fait en utilisant des messages `ICMP echo_reply` (cf. RFC 792) où le champ identifiant (16 bits) contient la commande transmise et le segment données les arguments ou messages. Les Agents ne produisent qu'un seul type de commande, un accusé de réception (`0x007B` ou `123` en décimal par défaut).

Chaque Agent en phase d'attaque peut créer jusqu'à 50 processus fils pour attaquer autant de cibles depuis l'hôte.

Pour des informations plus complètes, voir tfn.analysis de D. Dittrich ([**?**]).

5.5 Stacheldraht(fil de fer barbelé en allemand)

C'est un outil partiellement basé sur TFN (dont il reprend le code de l'Agent) mais qui a comme caractéristique de chiffrer toutes ses communications TCP. De plus, il comporte des instructions de compilation pour Linux et Solaris.

Version intiale (analysée par D. Dittrich) :

```
        16660/tcp           ------------------------------>
Client ------------> Serveur  65000/tcp et echo_reply/icmp  Agent
                            <------------------------------
```

Version 4 actuellement disponible :

```
        61111/tcp            65512/tcp et echo_reply/icmp
        ou 65512/tcp         ou 65513/tcp
Client ------------> Serveur ------------------------------>  Agent
                            <------------------------------
                                   echo_reply/icmp
```

En l'absence de *rootkit*, `netstat` donne :

Serveur

Protocole	Adresse locale	Commentaire
tcp	0.0.0.0:x	où x=16660, 61111 ou 65512
raw	0.0.0.0:1	

Agent

Protocole	Adresse locale	Commentaire
tcp	0.0.0.0:x	où x=65000, 65512 ou 65513
raw	0.0.0.0:1	

5.5.1 Installation

On trouve une phase d'installation semblable à trinoo : les Agents possèdent un fichier avec une liste de Serveurs (`.ms` par défaut) chiffré avec BlowFish ou bien se réfèrent à 2 Serveurs par défaut codés dans le binaire. L'identification mutuelle se fait par l'envoi d'un message ICMP echo_reply en clair avec comme identifiant `666` en décimal et `skillz` dans le champ de données. Tout Serveur présent répond par un `echo_reply` avec un identifiant `667` et `ficken` en données. L'Agent procède alors à un test pour savoir si il peut émettre des adresses sources complètement aléatoire en envoyant vers un Serveur un paquet ICMP echo_reply d'adresse source 3.3.3.3 avec son adresse réelle dans les données (identifiant 666). Si le message arrive au Serveur il répond via echo_reply

avec un identifiant à `1000` et `spoofworks` (`l'usurpation fonctionne` !) dans les données. En cas de succès l'Agent fabriquera des adresses sources quelconques, sinon il se limitera à modifier l'octet de poids le plus faible (ce qui permet au moins à la cible d'identifier le réseau ou sous-réseau émetteur).

Le Serveur stocke les Agents identifiés dans un fichier `.bc` chiffré avec BlowFish.

5.5.2 Attaque

Il faut utiliser un client dédié qui chiffre (à l'aide de BlowFish) tout le trafic TCP avec le Serveur. Après authentification à l'aide d'un mot de passe, les commandes suivantes peuvent être transmises aux Agents via echo_reply/icmp (la commande est un code dans l'identifiant) :

- gestion de la liste de Serveurs des Agents (ajout, retrait) ;
- liste des Agents présents et morts après un test de la connexion TCP ;
- changement de la taille, des ports cibles, pour les paquets d'attaque ;
- attaques par saturation de requêtes UDP, TCP, ICMP sur plusieurs adresses ;
- ouverture d'un shell distant sur un port donné par les hôtes des Agents ;
- mise à jour de chaque Agent en utilisant le programme rcp sur la machine hôte.

Chaque Agent a la possibilité de créer un certain nombre de processus fils (1 par défaut et 15 maximum) pour attaquer séquentiellement l'ensemble des cibles.

Pour des informations plus complètes, voir stacheldraht.analysis de D. Dittrich (cf. [**?**]).

5.6 TFN2K

Il s'agit d'une évolution de TFN, ayant pour but de diversifier les attaques réalisées et d'améliorer la furtivité suite aux vulnérabilités et signatures (binaires, trafic réseau) publiées après l'analyse de TFN. Par ailleurs le code comporte des directives pour la compilation sous Linux, Solaris et Windows.

```
        tout shell            echo_reply/icmp ou tcp ou udp
Client ------------> Serveur -------------------------------> Agent
         distant             <-------------------------------
                              echo_reply/icmp ou tcp ou udp
```

En l'absence de *rootkit*, `netstat` donne :

Serveur

Protocole	Adresse locale	Commentaire
raw	0.0.0.0:x	où x=1(icmp), 6(tcp) ou 17(udp) plus une éventuelle connexion telnet, SSH, ...

Agent

Protocole	Adresse locale
raw	0.0.0.0:1
raw	0.0.0.0:6
raw	0.0.0.0:17

5.6.1 Installation

Semblable à TFN.

5.6.2 Attaque

Le *Serveur* reçoit ses ordres sous forme de ligne de commande : le *Client* doit donc se connecter sur l'hôte avec un shell distant (en clair avec telnet, chiffré avec SSH, ...). Le *Serveur* nécessite un éventuel de mot de passe, mais il faut fournir un fichier des *Agents* où l'adresse de l'un d'eux dans la commande. Les ordres qui peuvent être transmis aux *Agents* sont :

- protocole de communication avec les Agents TCP, UDP, ICMP ou aléatoire par défaut ;
- choix du masque pour falsifier les adresses sources (aléatoire, modification d'un, de deux ou de trois des octets de plus faible(s) poids de l'adresse de l'Agent) ;
- changement de la taille des paquets d'attaque ;

– attaques par saturation de requêtes UDP, TCP, ICMP, smurf (en précisant un ou des réseaux amplificateur) ou un mélange des trois premiers ;
– attaque dite TARGA3 fabricant des paquets IP avec des protocoles, des valeurs de fragmentation et des drapeaux normaux et fantaisistes auxquels certaines (?) piles TCP/IP sont sensibles avec toutes sortes de conséquences néfastes pour la cible ;
– réalisation d'une commande shell quelconque par les Agents. Cela offre un champ immense de possibilités d'agression : envois massifs de mails, mise à jour des agents, récupération de tout fichier (dont mots de passe) sur les hôtes, activation des scripts d'un serveur HTTP cible, ... (dans ces cas, l'adresse source ne peut pas être falsifiée sans outil complémentaire) ;
– ouverture d'un shell distant sur un port donné par les hôtes des Agents.

La communication un *Serveur* et des *Agents* se fait en utilisant le champ de données d'un paquet TCP, UDP ou ICMP (choix aléatoire par défaut) en chiffrant la commande avec l'algorithme symétrique CAST-256. L'Agent est à l'écoute des trois protocoles et, en déchiffrant le champ de données, détermine si le paquet lui est destiné.

Chaque Agent en phase d'attaque peut créer jusqu'à 50 processus fils pour attaquer autant de cibles depuis l'hôte.

Pour d'autres informations, voir l'analyse d'Axent Security Team (cf. [**?**]).