Legond-Aubry Fabrice
fabrice.legond-aubry@lip6.fr

SSI

Sécurité des Systèmes Informatiques

Concepts

Note: une partie des slides est extrait du cours de sécurité de S. Naktin du CNAM

Utilisation des outils

Plan de cours

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

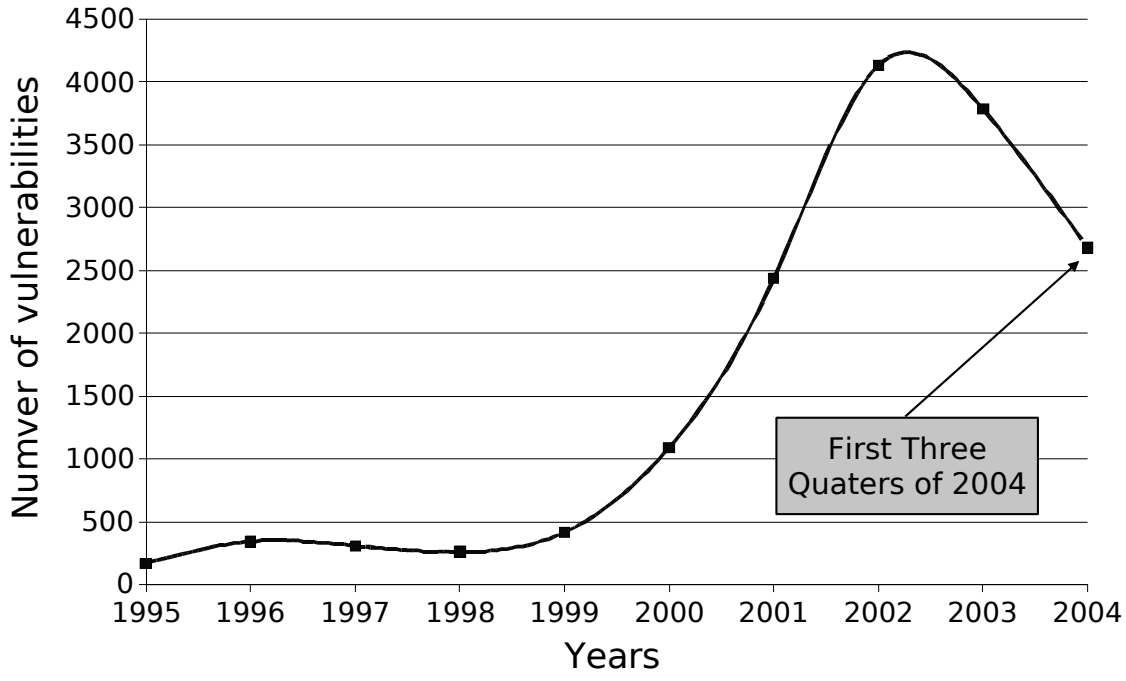
Les certificats

Authentification des personnes

Statistiques

Introduction

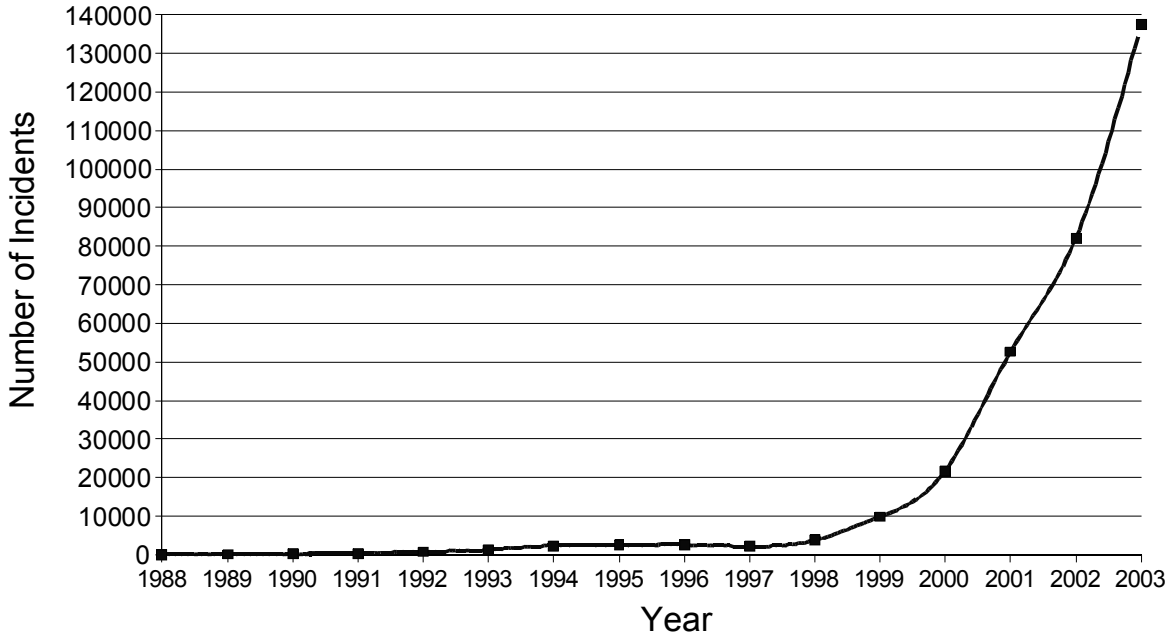
Vulnerabilities



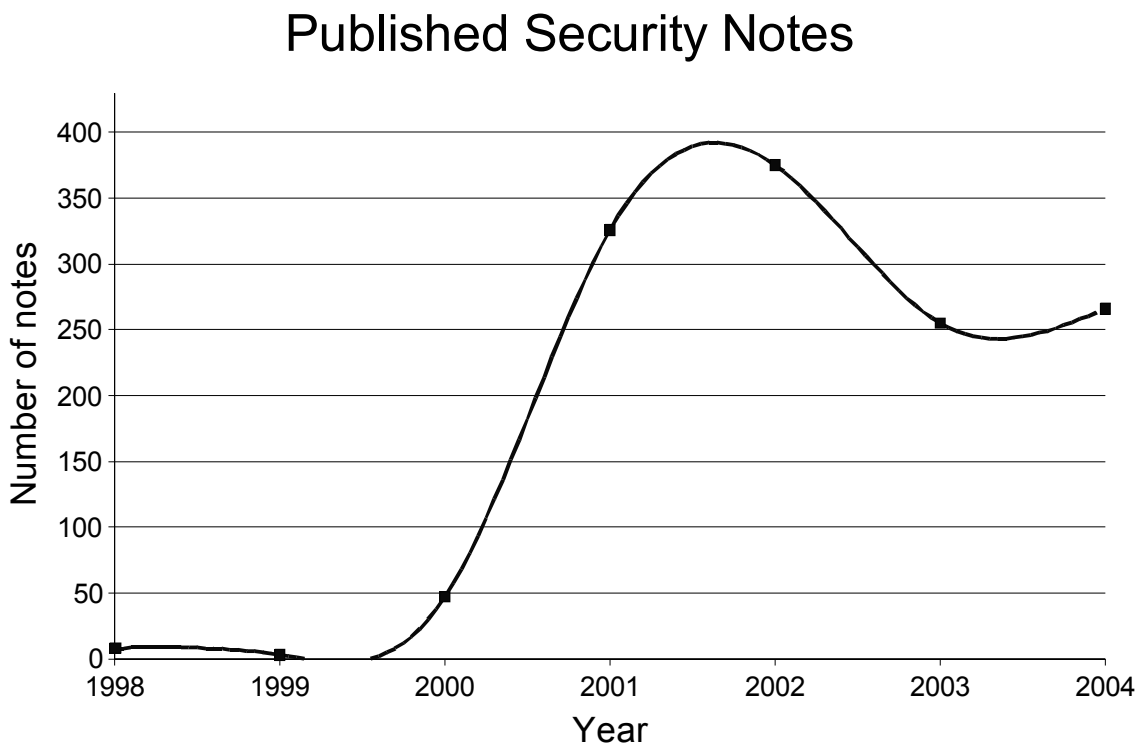
Statistiques

Introduction

Incidents



Statistiques



Introduction

L'homme et l'ordinateur

- Importance croissante du rôle de l'ordinateur
  - Dans la diffusion de l'information
  - via des systèmes techniques de plus en plus complexes, dans des domaines de plus en plus variés.
- Dans le passé, l'informatique était concentrée sur
  - les effets possibles d'une erreur de programmation
  - la validation de processus critique (transport, énergie...)
- Actuellement, on se concentre sur
  - le détournement possible des nouvelles technologies de l'information
  - la communication par soit des pirates / groupes étatiques / groupes industriels d'informations
- Faut-il craindre avec raison les effets pervers d'une informatisation trop rapide de la société?
  - OUI !!!!

Introduction

Ce qui peut arriver à votre machine

- Refuser de faire quoi que ce soit (plantage)
- Faire trop tôt ou trop tard ce qu'il devait faire (mauvaise réactivité)
- Accomplir des actions différentes de celles attendues (« bug »)
- Etre attaquer par un pirate avec pour conséquence (attaque)
  - La destruction de vos données
  - La transformation de votre écran en une œuvre d'art minimaliste
  - L'inondation de la planète de messages pornographiques
  - L'espionnage de votre comportement et la vente de ces informations
  - L'utilisation de votre machine comme relai d'attaque
  - L'implantation d'un module de surveillance étatique

Conséquences

- Dans la majorité des cas, des conséquences assez bénignes:
  - "retaper" deux ou trois fois la même chose, suite à la « perte d'un fichier »
  - Perdre des emails, des photos, des textes
  - Subir de la publicité
  - Réinstallation d'une machine chez le particulier
- Mais pour une entreprise, des conséquences considérables:
  - la paralysie des serveurs Web,
  - le vol de sommes considérables,
  - la faillite d'une entreprise qui ne peut plus facturer
- Dans le futur ?
  - l'échec d'un tir de fusée
  - la création d'embouteillages monstrueux,
  - une panne de courant paralysant une métropole
  - une panne paralysant les transports ferroviaires d'un grande capitale

Pourquoi est-ce si important ?

Introduction

- Dommages potentiels importants
  - Indisponibilités des systèmes
  - Manipulation(s) des données/systèmes
  - Destruction des données/systèmes
  - Si la puce de votre carte d'identité crash existerez-vous encore ?
- Coûts importants de remise en marche
  - Financiers (remise en marche, ventes manquées)
  - Temporels (remise en marche, configuration)
  - Humains (juristes, informaticiens)
  - Matériels (serveurs de sauvegarde, redondance, sécurité)
- Coûts importants de maintient en état
  - Analyses des données de surveillance
  - Analyses des machines

La sécurité : un problème critique !

Introduction

- La sécurité des machines et des réseaux
  - Doit être la première préoccupation de toute entité utilisant l'informatique
    - Entité = gouvernement, entreprise, particulier, associations, …
    - Laisseriez-vous ouverte votre porte d'appartement ?
  - Si ignoré, des nombreux problèmes financiers
  - Doit dépendre d'ingénieurs spécialistes
- Nécessite une définition
  - Des risques de sécurité
  - D'un plan et d'une politique de sécurisation
    - architectures, droits, organisations, acteurs
  - D'un plan de réaction aux attaques

Sécurité : centres de préoccupations

Introduction

- Sécurité locale (sur une machine)
  - Sécurisé l'OS
  - Sécurisé les services
  - Sécurisé l'accès à la machine (boot)
- Sécurisé réseau (services réseaux)
  - Topologie et architectures des serveurs
  - Les protocoles exploités
  - Le contrôle d'accès par firewall
- **Les utilisateurs !!!!**
  - Education comportementale
    - Peur de l'ordinateur (ligne 14, régulateurs vitesses, …)
    - La non-compréhension des outils informatiques
  - Identifications des maillons faibles (personnes à risques)
- Le cadre juridique et sociale (politique ?)

Conclusion

Introduction

- Besoin en spécialistes en sécurité pour définir :
  - Quels types d'attaques l'entreprise peut subir
  - Quels moyens de protections sont à mettre en place
  - Quels mesures (contre-mesures) utiliser en cas d'attaque
  - Quels sont les contrôles sont à effectuer et à quelle fréquence
- Il n'y a aucun système sûr à 100%
- Il faut savoir choisir son degré de protection en fonction de ses besoins
  - ADEQUATION DES MOYENS ET DES OBJECTIFS
  - C'EST UNE OBLIGATION JURIDIQUE

Conclusion

Introduction

Il existe donc une probabilité raisonnable de pouvoir cohabiter et même collaborer avec les ordinateurs.

Il suffit de prendre le temps de savoir ce que nous voulons en faire et comment.

Lorsque le problème est bien posé, les solutions techniques existent déjà souvent et, dans le cas contraire, seront inventées.

Bibliographie

Introduction

- S. Natkin, « *Protocoles de Sécurité de l'Internet* », Dunod, 2002
- B. Schneier, « *Cryptologie appliquée* », Thomson publishing, 2001
- J. Stern ,*« La science du secret »*, Odile Jacob Ed, 1998
- D. Stinson, « *Cryptologie: théorie et pratique* », Thomson publishing, 2003
- D. B. Chapman & E. D. Zwicky, *« La sécurité sur Internet — Firewalls »*, O'Reilly , 1996
- Microsoft Security Team, « *Sécurité Windows* », Microsoft Press, 2005
- Les livres de poches d'Isaac Asimov (« *les robots* »)
- S. Garfinkel & G. Spafford , *« Practical UNIX & Internet Security »*, seconde Édition, 1996
- B. Schneier, « *Secrets and Lies, Digital Security in a Networked World* », J. Wiley and sons ed, 2000

Sites WEB

Introduction

- Commentcamarche : introduction
- Le site du cnam (G. Florin, S. Natkin) pour le module sécurité

  (de nombreuses informations sont extraites de ce cours)
- http://www.ssi.gouv.fr/ Serveur thématique sur la sécurité des systèmes d'information du Secrétariat Général de la Défense Nationale
- http://fr.wikipedia.org/ section sécurité
- http://www.cru.fr/ Comité Réseau des Universités
- http://www.urec.fr/ Unité Réseau du CNRS
- http://www.hsc.fr/ Hervé Schauer Consultants
- https://www.clusif.asso.fr/index.asp Club de la Sécurité des Systèmes d'Information Français
- www.renater.fr, www.cert.org, www.securite.org, www.ouah.org, www.securityfocus.org
- Cours sur le web :
  - Michel Riguidel (http://perso.enst.fr/~riguidel/UESecur)
  - Stephane Naktin au CNAM

Plan de cours

Utilisation des outils

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

Les certificats

Authentification des personnes

Concepts et Terminologie

- La **sûreté** de fonctionnement d'un SI correspond au degré de confiance que peut accorder les utilisateurs dans le service délivré
  - **Disponibilité (avaibility)** : capacité à être prêt à délivrer le service (dans les meilleures conditions)
  - **Fiabilité (reliability)** : continuité de service (pas d'arrêt)
- La **sécurité** de fonctionnement d'un SI se décompose en deux thèmes
  - **Sécurité (safety)** : évitement des situations catastrophiques
    - Celles qui sont considérées comme inacceptables pour les utilisateurs
  - **Sécurité (security)** : préservation de la confidentialité et de l'intégrité des informations
    - la lutte contre les fautes intentionnelles (virus, bombes logiques, chevaux de Troie, etc.)

Concepts de sûreté

- Défaillance : Service délivré ≠ Service spécifié
- Erreur : état du système susceptible d'entrainer une défaillance
- Faute : Cause de l'erreur
- Relations erreurs/fautes/défaillances:
  - L'erreur est la manifestation de la faute sur le système
  - La défaillance est l'effet d'une erreur sur le service

Concept de sûreté : Faute

Concepts et Terminologie

- Une faute devient active lorsqu'elle produit une erreur
- Une faute active est :
  - soit une faute dormante activée par le traitement
  - soit une faute externe
- Une faute interne peut passer cycliquement de l'état dormant à actif, …

Concept de sûreté : Erreur

Concepts et Terminologie

- Temporaire par nature
- Latente ou détectée
  - Latente, tant qu'elle n'a pas été reconnue
  - Détectée, soit par les mécanismes de détection et
- traitement d'erreurs, soit par son effet sur le service (défaillance)

1 erreur ➠ propagation d'autres erreurs dans d'autres parties du système (effet papillon)

Concept de sûreté : Défaillance

Concepts et Terminologie

- Une défaillance survient lorsqu'une erreur traverse l'interface Système/Utilisateur et altère le service délivré par le système
- Dans un système constitué d'un ensemble de composants, la conséquence de la défaillance d'un composant est :
  - une faute interne pour le composant englobant,
  - une faute externe pour les composants avec lesquels il interagit
- Type de défaillance
  - Fautes franches (*fail stop*) : arrêt pur et simple
  - Omissions : perte de messages
  - Temporaires : déviations temporelles / spécifications
  - Byzantin : comportement aléatoire ou malveillant

De l'erreur à la défaillance

... ➠ défaillance ➠ faute ➠ erreur ➠ défaillance ➠ ...

Concepts et Terminologie

- Une erreur est susceptible de provoquer une défaillance, mais ne la provoque pas nécessairement (ou pas immédiatement)
  - Parce qu'il y a une redondance interne suffisante pour que le système continue de fournir le service
  - Parce que la partie erronée de l'état n'est pas utilisée pour telle ou telle fonction
- Une erreur est latente tant qu'elle n'a pas provoqué de défaillance
- Le temps entre l'apparition de l'état d'erreur et la défaillance est le délai de latence
  - plus le délai de latence est long, plus la recherche des causes d'une défaillance est difficile

Propriétés de sécurité

Avant de pouvoir effectivement développer des applications sécurisées, vous devez comprendre les concepts fondamentaux de la sécurité.

Les 5 piliers de la sécurité sont

Authentification

Non répudiation

Intégrité

Confidentialité

Auditabilité

Propriété de sécurité: l'authentification

C'est la propriété qui assure la reconnaissance sûre de l'identité d'une entité

- L'authentification protège de l'usurpation d'identité
- Signature = Authentification
  - Authentification: Première idée contenue dans la notion habituelle de signature
  - le signataire est le seul à pouvoir réaliser le graphisme (caractérisation psychomotrice)
- Entités à authentifier:
  - une personne
  - un programme qui s 'exécute (processus)
  - une machine dans un réseau

Propriété de sécurité: la non répudiation

C'est la propriété qui assure que l'auteur d'un acte ne peut ensuite dénier l'avoir effectué

- Signature = Authentification+Non répudiation :
  - Seconde idée contenue dans la notion habituelle de signature
  - le signataire s'engage à honorer sa signature
  - engagement contractuel/juridique, on ne peut pas revenir en arrière
- Deux aspects spécifiques de la non répudiation dans les transactions électroniques:
  - ***a) La preuve d'origine :*** Un message (une transaction) ne peut être nié par son émetteur.
  - ***b) La preuve de réception :*** Un récepteur ne peut ultérieurement nier avoir reçu un ordre s'il ne lui a pas plu de l'exécuter alors qu'il le devait juridiquement.
- Exemple: Exécution d'ordre boursier, de commande, …

Concepts et Terminologie

Propriété de sécurité: l'intégrité

C'est la propriété qui assure qu'une information n'est modifiée que par des entités habilitées (selon des contraintes précises)

- Exemples :
  - Une modification intempestive (même très temporaire) est à interdire sur une écriture comptable validée
  - Le code binaire des programmes ne doit pas pouvoir être altéré
  - Les messages de l'ingénieur système doivent pouvoir être lus et non modifiés

Concepts et Terminologie

Concepts et Terminologie

Propriété de sécurité: la confidentialité

C'est la propriété qui assure qu'une information ne peut être lue que par des entités habilitées (selon des contraintes précises)

- Exemples :
  - Un mot de passe ne doit jamais pouvoir être lu par une autre personne que son possesseur
  - Un dossier médical ne doit pouvoir être consulté que par les malades et le personnel médical habilité
  - On ne doit pas pouvoir intercepter le contenu d'un courrier

Concepts et Terminologie

Propriété de sécurité: l'auditabilité

C'est la propriété qui assure la capacité à détecter et à enregistrer de façon infalsifiable les tentatives de violation de la politique de sécurité.

- **Audit** : Examen méthodique d'une situation relative à un produit, un processus, une organisation, réalisé en coopération avec les intéressés en vue de vérifier la conformité de cette situation aux dispositions préétablies, et l'adéquation de ces dernières à l'objectif recherché [définition ISO, d'après la norme AFNOR Z61-102]
- **Auditabilité** : Garantir une maîtrise complète et permanente sur le système et en particulier pouvoir retracer tous les événements au cours d'une certaine période.

Plan de cours

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

Les certificats

Authentification des personnes

Attaques sur l'authentification

- Déguisement (Mascarade)
  - Pour rentrer dans un système, on essaye de piéger des usagers et de se faire passer pour quelqu'un d'autre (usurpation d'identité)
- Exemple:
  - simulation d'interface système sur écran,
  - simulation de terminal à carte bancaire

Attaques sur l'intégrité

Types d'attaques

- **Intégrité des données : Modification de messages, de données**
  - Attribution par une personne non autorisée (usager , agent autorisé) d'avantages illicites
  - Comment ? En modifiant un fichier, un message, ...
  - Le plus souvent cette modification est réalisée par un programme et devient aussi une attaque sur l'intégrité des programmes
  - Ex : modification des données sur un serveur Web
- **Intégrité des protocoles : Répétition ("replay")**
  - Espionnage d'une interface, d'une voie de communication (téléphonique, réseau local) pour capter des opérations (même cryptées elles peuvent être utilisables)
  - Répétition de l'opération pour obtenir une fraude.
  - Exemple: Plusieurs fois la même opération de crédit d'un compte bancaire.

Attaques sur l'intégrité

Types d'attaques

- **Intégrité des programmes**
  - Les modifications à caractère
    - *Frauduleux* : Pour s'attribuer par programme des avantages (virement des centimes sur un compte)
    - *De sabotage* : Pour détruire avec plus ou moins de motivations des systèmes ou des données
  - Type de modifications
    - Infections informatiques à caractère unique
      - Bombe logique ou cheval de Troie
      - Introduction d'un comportement illicite avec un trigger
    - Infections auto reproductrices
      - Virus (reproduction rapide), Ver (reproduction lente, dormant)
      - Vecteur d'infection : secteur amorçage, infection fichier, macros virus, mutation

Attaques sur la confidentialité

Types d'attaques

- Les attaques ayant pour but le vol d'informations via un réseau par espionnage des transmissions de données (espion de ligne, accès aux données dans des routeurs et des serveurs Internet)
- Analyse de trafic : On observe le trafic de messages échangés pour en déduire des informations sur les décisions de quelqu'un.
  - Exemples: augmentation des transactions sur une place financière
  - Exemple: le début de concentration militaire entraîne un accroissement de trafic important.
- Inférence : On obtient des informations confidentielles à partir d'un faisceau de questions autorisées (et d'un raisonnement visant à faire ressortir l'information).

Attaques sur la disponibilité

Types d'attaques

- **Attaque par violation de protocole**
  - Erreur très rare en fonctionnement normal et non supportées par le protocole
  - Envoie de données non prévues (trames malformées, séquence non prévues)
- **Attaque par saturation**
  - Envoi de messages trop nombreux provoquant un écroulement des systèmes et réseaux
  - Exemple : « Distributed Denial Of Service »

Attaques sociales

- Dans la majeure partie des cas le maillon faible est l'utilisateur lui-même !
- Par méconnaissance ou duperie, l'utilisateur va ouvrir une brèche dans le système.
- Comment ?
  - En donnant des informations (mot de passe par exemple) au pirate informatique
  - En exécutant une pièce jointe
  - En discutant sur du chat
  - En ramassant une disquette/CD et en l'insérant dans un lecteur
- Aucun dispositif de protection ne peut protéger l'utilisateur contre les arnaques
  - seuls bon sens, raison et un peu d'information sur les différentes pratiques peuvent lui éviter de tomber dans le piège !

Attaques sociales

- Déroulement :
  - Une phase d'approche (ou d'accroche ☺)
    - permettant de mettre l'utilisateur en confiance
      - en se faisant passer pour une personne de sa hiérarchie, de l'entreprise, de son entourage ou pour un client, un fournisseur, sa banque, …
  - Une mise en alerte
    - afin de le déstabiliser et de s'assurer de la rapidité de sa réaction
      - prétexte de sécurité, d'une situation d'urgence
  - Une diversion (une situation permettant de rassurer l'utilisateur et d'éviter qu'il se focalise sur l'alerte)
    - Phase optionnelle
    - Ex: un remerciement, un courrier électronique, un site web, une redirection vers le site web de l'entreprise
- http://www.securityfocus.com/infocus/1527
- http://www.cert.org/incident_notes/IN-2002-03.html

Plan de cours

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

Les certificats

Authentification des personnes

Etapes pour une politique de sécurité

1. Définition de la politique
   - Règles concernant les ressources informatiques

     Ressources immatériels / données
   - Règles concernant les ressources physiques

     Documents papiers, accès aux bâtiments
2. Identification des vulnérabilités
   - En mode fonctionnement normal (définir tous les points faibles)

     Ex: Arrivé/Départ de personnel, sortie de documents, entrée d'appareils électroniques, les passes des « TECHNICIENS DE SURFACES » !
   - En cas d'apparition de défaillances un système fragilisé est en général vulnérable

     Ex: Lecteur de carte HS, Contrôle biométrique inopérant, Serveur Kerberos HS
   - C'est dans un de ces moments intermédiaires qu'une intrusion peut le plus facilement réussir

     Ex: arrêt d'un firewall pour effectuer un transfert de données

Etapes pour une politique de sécurité

3. Évaluation des probabilités associées à chacune des menaces

   Ex: Danger des CDs de musique, des clefs USB

   Ex: Autoriser les accès aux sites de cracks

4. Évaluation du coût d'une intrusion réussie

   Ex: Le coût d'un vol d'information grâce à un iPod 30go ?

   (beaucoup de base de données tiennent sur 30go ...)

5. Choix des contre mesures

   Ex: Interdiction de boot sur des supports externes

   Ex: Qui peut accéder physiquement aux machines ?

6. Évaluation des coûts et de **l'adéquation** des contres mesures

   Ex: Mettre des contrôles rétiniens en dehors d'entité confidentiel défense est hors la loi.

   Ex: Pénétrer une machine à l'origine de l'attaque est interdit

7. Décision

   Application et mis en place des solutions

Cohérences des moyens

- La réalisation d'une politique de sécurité résulte de la mise en œuvre cohérente de:
  - Moyens physiques
    - architecture des bâtiments, systèmes de contrôle d 'accès, destructeurs de documents…
  - Moyens informatiques
    - Contrôles de services et des machines
  - Règles d'organisation et moyens procéduraux
    - règles de fonctionnement qui doivent être respectées

Cohérences des moyens

- Les moyens doivent être « complets »:
  - dans le cadre des hypothèses considérées, quoi qu'il arrive la politique est respectée
- Les moyens doivent être non contradictoires et raisonnablement contraignants
  - Ils ne doivent pas constituer un obstacle à la réalisation des fonctions opérationnelles de l 'organisation considérée
  - Ex: les procédures trop complexes sont souvent contournées
- Les moyens doivent être homogènes par rapport aux risques et aux attaques considérés
  - Ex: il est inutile de chiffrer tous les documents informatiques s'ils partent en clair dans les poubelles
- Le respect des procédures est un des points essentiels de l'efficacité
  - Elles doivent donc être comprises et acceptées par toutes les personnes concernées.

Principe de mise en œuvre

- Assurer la mise en œuvre d'une politique de sécurité consiste à garantir que, à chaque instant, toutes les opérations sur les objets (ressources) ne sont réalisables et réalisées que par les entités (physique ou informatique) habilitées.
- La base de la réalisation de la sécurité sont
  - **le confinement**: L'ensemble des objets sont maintenus dans des domaines étanches, l'accès ce fait via un guichet protégé.
  - **le principe du moindre privilège:** Pour qu'un système fonctionne en sécurité il faut donner à ses utilisateurs exactement les droits dont il ont besoin pour s'exécuter, **ni plus ni moins**.

Les politiques de sécurité

Dans le détail des politiques

- Il existe différentes méthodes pour créer des politiques de sécurités
  - **MEHARI** (*MEthode Harmonisée d'Analyse de RIsques*), CLUSIF, https://www.clusif.asso.fr/fr/production/mehari/
  - **EBIOS** (*Expression des Besoins et Identification des Objectifs de Sécurité*), DCSSI, http://www.ssi.gouv.fr/fr/confiance/ebios.html
  - La norme ISO 17799
- Il existe différents standards de sécurité
  - Défini par le ministère de la défense US
    - http://www.dia.mil, http://www.radium.ncsc.mil/ (orange book)
  - Défini par la NSA (Compartemented Mode Workstation)
    - http://www.nsa.gov/
- Il existe différents standards de validation de politiques
  - CC-EAL-5 (Common Criteria Evaluation Assurance Level 5), ISO 15408
    - http://niap.nist.gov/cc-scheme/index.html

Utilisation des outils

Plan de cours

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

Les certificats

Authentification des personnes

Quelques définitions

- **Décrypter ou casser un code** c'est parvenir au texte en clair sans posséder au départ ces informations secrètes. C'est l'opération que doit réaliser Estelle pour retrouver M.
- **L'art de définir des codes (de chiffrement) est la cryptographie.** Un spécialiste en cryptographie est appelé cryptographe.
- **L'art de casser des codes est appelé cryptanalyse ou cryptologie**. Un spécialiste en cryptanalyse est appelé cryptanalyste.
- **Un crypto-système** est l'ensemble des deux méthodes de chiffrement et de déchiffrement utilisable en sécurité.

Ce que permet la cryptographie

- 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,...)
  - Empêcher un décodage par hasard
  - Empêcher une attaque par force brute
  - Empêcher l'utilisation méthode inédite de décodage
  - Empêcher la lecture avant codage ou après décodage
- Ne pas sous-estimer les autres méthodes de sécurité sous prétexte que la cryptographie est omnipotente

Chiffrement

Le chiffrement est donc une transformation d'un texte pour en cacher le sens.

- Les acteurs : A(lice) et B(ob) – les gentils, Estelle (l'Espionne)
- Bob, doit transmettre à Alice, un message M ∈ MESSAGES_A_ENVOYER.
- M est un message dit « en clair » (non chiffré).
- Estelle écoute la voie de communication pour connaître M.
- Bob, construit un texte chiffré C ∈ MESSAGES_CHIFFRES.
- $C= E_k(M)$ ou $C = \{M\}^{E}_{k}$
- La fonction E_k dépend d'un paramètre k appelé clef de chiffrement.
- La possibilité de chiffrer repose donc sur la connaissance de l'algorithme de chiffrement E et de la clef k de chiffrement.

Déchiffrement

Le déchiffrement est l'opération inverse permettant de récupérer le texte en clair à partir du texte C chiffré.

- Il repose sur la fonction $D_{K'}$ de MESSAGES_CHIFFRES dans MESSAGES_A_ENVOYER telle que
  - $M= D_{K'}(C)$ ou $C = \{M\}^{D}_{K'}$
- On doit avoir l'idempotence !! ➔ $D_{K'}(E_k(M))= M$
  - K, K' sont des secrets, D et E des «algorithmes» publics
- $D_{K'}$ est donc une fonction inverse à gauche de E_k.
  - Note: il peut y avoir symétrie (D=crypter, E=décrypter)
- Pour que ces opérations assurent la confidentialité du transfert entre Alice et Bob, il est nécessaire qu'au moins une partie des informations E, D, k, K soit ignorée du reste du monde.

Efficacité du chiffrement

- L'efficacité dépend
  - Du Secret de la clé
  - De la difficulté à deviner la clé ou a les essayer toutes : lié à la taille de la clé
  - De la difficulté de l'inversion de l'algorithme de chiffrement sans connaître la clé (*cassage*)
  - De la longueur de la clé
  - De l'existence de portes par derrière (pour les gouvernements...) ou d'autres moyens plus faciles de déchiffrement
    - D'où l'importance de l'accès ou non au code source
  - 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

Un outil: Crypto-systèmes symétriques

- Tels que soit **k=K**, soit la connaissance d'une des deux clefs permet d'en déduire facilement l'autre.
- Conséquences :
  - Dichotomie du monde : les bons et les mauvais
  - Multiplication des clefs (un secret n'est partagé que par 2 interlocuteurs), donc pour N interlocuteurs N*(N-1)/2 couples
- La qualité d'un crypto système symétrique s'analyse par rapport à des propriétés statistiques des textes chiffrés et la résistance aux classes d'attaques connues.
- En pratique tant qu'un crypto système symétrique n'a pas été cassé, il est bon, après il est mauvais.

Les outils de la sécurité

CS symétriques: exemples

- Substitution mono alphabétique
  - Pour chaque lettre, on associe une lettre de substitution
  - Attaquable par la connaissance du % d'utilisation des lettres
  - Attaquable par la connaissance de la structure du message
- Substitutions de polygrammes
  - Pour chaque lettre, on associe des groupes de lettres de substitution
- Par transposition
  - On écrit le texte dans un tableau de n colonnes puis on écrit les colonnes
- CS symétriques modernes :
  - Combinaison complexe d'opérations de transposition et de substitution sur des chaines de bits (opérateurs arithmétiques) prenant comme paramètre tout ou partie de la clef.
  - Fonctionne par blocs (ECB) ou en chaîne (CBC)

Les outils de la sécurité

CS symétriques: l'ancêtre DES

- Créé en 1978 par IBM
- Caractéristiques
  - Cryptage par bloc de 64 bits (0/1)
  - Utilise une clef de 56 bits (0/1)
  - 19 étages (étapes) d'opérations de logique combinatoire
  - Chaque étape est son propre inverse
- Performances excellentes (car basé sur des opérations logiques simples).
- Peu sécurisé car il existe des algorithmes de cassage efficace.
- Il existe des faiblesses dans le DES connu depuis longtemps par la NSA : il ne résiste pas à la cryptanalise différentielle

CS symétriques: Caractéristiques

- Les nouveaux
  - 3DES : succession de 3 DES en cascade avec 2 clefs K1 et K2 de 56bits ➔ DES_{K1},DES^{-1}_{K2},DES_{K1}
  - IDEA : longueur de clef élevé (128bits)
  - Blowfish, SAFER, AES, AES256, TWOFISH, CAST5
  - RC2, RC3, RC4, RC5, Shipjack (secret)
- Caractéristiques :
  - Débits importants
    - IDEA: 880 Kb/s (logiciel sur 386 33Mhz), 55Mb/s (circuits)
  - Une seule clef (donc partagée)
  - Clef de « petite taille »

Un outil: Crypto-systèmes asymétriques

- Tels que la connaissance de k (la clef de chiffrement) ne permet pas d'en déduire celle de K(la clef de déchiffrement). [K' ≠ k]
- Un tel crypto-système est dit asymétrique, la clef k est appelée la **clef publique**, la clef K est appelée la **clef privée**.
- Fondement théorique : montrer que la recherche de K à partir de k revient à résoudre un problème mathématique notoirement très compliqué, c'est à dire demandant un grand nombre d'opérations et beaucoup de mémoire pour effectuer les calculs.
- **RSA** (l'algorithme le plus utilisé à l'heure actuel) la déduction de K à partir de k revient à résoudre le problème de factorisation d'un grand nombre. Un problème sur lequel travaille les mathématiciens depuis plus de 2000 ans,
- On estime que le plus rapide ordinateur que l'on puisse construire utilisant la meilleure méthode connue met plus de 1000 ans pour retrouver la clef privée d'un système RSA utilisant un modulo de 1024 bits (ordre de grandeur de la taille des clefs).
- Il existe RSA, DSA, ElGammal (ancien DSA), ELG

CS asymétrique : « l'ancêtre RSA »

- **Chiffrement**
  - La clé publique est un couple d'entiers: **K = (e, n)**
  - Le chiffrement se fait au moyen de l'élévation à la puissance e modulo n: **E_K (M) = M^e mod n**
- **Déchiffrement**
  - La clé secrète est un couple d'entiers: **k = (d, n)**
  - Le déchiffrement se fait au moyen de l'élévation à la puissance d modulo n: **D_k (M) = M^d mod n**
- Utilisation intensive des grands nombres
  - Il existe des algorithmes pour calculer les puissances avec modulo sur des grands nombres

CS asymétrique RSA : Les clefs

- **Détermination de n**
  - Trouver **deux entiers premiers** p et q très grands
  - **Calculer n = p q**
  - p et q doivent rester secrets: La sécurité du système repose sur la difficulté de factoriser un grand entier n en deux entiers premiers p et q.
  - n doit avoir une longueur supérieure à 512 bits.
- **Détermination de e**
  - Calculer **z = (p-1) (q-1)**
  - Choisir un entier **e premier avec z.**
- **Détermination de d**
  - Choisir un entier d tel que : **e * d =1 mod z**
  - d inverse de e dans l'arithmétique mod z

La clé privée est (d,n). La clé publique est (e,n).

CS asymétrique RSA : Exemple

Les outils de la sécurité

- **Exemple:**
  - P=7, Q=3 ➔ N = P*Q = 21
  - Z= (P – 1) * (Q – 1) = 6 * 2 = 12
  - Choisir e premier avec z: e=5
  - Choisir d tel que d*e=1 [z]

    d*5=1 mod 12 ➔ d=17
- Clef publique (d=17,n=21)
- Clef privée (e=5,n=21)
- Cryptage, décryptage du nombre 19 (M)
  - **E_K (19) = M^e [n] = 19^5 [21] = 2 476 099 [21] = 117909 * 21 + 10 [21] = 10 [21]**
  - **D_k (10) = 10^{17} [21] = 100 000 000 000 000 000 [21] = 4 761 904 761 904 761 * 21 + 19 [21] = 19**

CS asymétrique RSA : performances

Les outils de la sécurité

- L'algorithme précédent est en O(3t) multiplications
- Multiplications sur 512 Bits= 64 multiplication en moyenne sur 32 bits. 192 multiplications pour l'élévation à la puissance.
- **Utiliser des longueurs de clés de plus en plus importantes**
  - Valeurs utilisées 512 bits, 640 bits, 1024 bits (considéré comme sûr pour plusieurs années), 2048 bits
- **Utiliser des circuits intégrés de cryptage de plus en plus performants**
  - Actuellement une dizaine de circuits disponibles.
  - Vitesse de cryptage de base pour 512 bits: de 10 à 100 Kb/s
  - Évolution en cours de l'ordre de 1 Mb/s
- **Remarque**: Compte tenu de la complexité des traitements le DES est environ toujours 10 à 100 fois plus rapide que le RSA.

CS asymétrique RSA : Conseils

- Ne jamais utiliser une valeur de n trop petite.
  - Actuellement un calcul en parallèle utilisant quelques milliers d'ordinateurs pendant quelques mois permet de factoriser des nombres d'une centaine de chiffres (400 bits)
  - Utiliser des n=1024 ou 2048 bits selon que la protection recherchée est de plus ou moins de cinq ans.
- Ne pas utiliser une clef secrète trop courte.
- N'utiliser que des clefs fortes, c'est à dire telles que p-1 et q-1 ont un grand facteur premier.
- Ne pas chiffrer des blocs trop courts (les compléter toujours a n-1 bits), de façon à détruire toute structure syntaxique [padding]
- Ne pas utiliser un n commun à plusieurs clefs si ces clefs peuvent être utilisées pour chiffrer un même message.
- Si une clef secrète (d,n) est compromise, ne plus utiliser les autres clefs utilisant n comme modulo.
- Ne jamais chiffrer ou authentifier un message provenant d'un tiers sans le modifier (ajouter quelques octets aléatoires par exemple).

Fonction à sens unique

C'est une fonction f(M) facile à calculer mais telle qu'il est extrêmement difficile de déduire M de f(M).

- Exemple:
  - Calcul modulo n (dans un anneau fini)
  - M^2 est facile à calculer modulo n (ou M^e)
  - $\sqrt{M}$ est difficile à calculer (log M)
  - CRC

Fonction de hachage

Une fonction de hachage h est une fonction qui à un message M de longueur quelconque fait correspondre un message H(M) (notée aussi $\{M\}^H$) de longueur constante.

- L'intérêt d'une fonction de hachage est que M peut être arbitrairement grand alors que $\{M\}^H$ a une longueur donnée.
- Fonction **NON BIJECTIVE**
  - elle est destructrice si taille M > longueur de $\{M\}^H$
  - Elle caractérise le bloc de données
  - Fonction a sens unique : difficulté pour retrouver M à partir de $\{M\}^H$
- Terminologie
  - Résumé, fonction de contraction, digest, empreinte digitale, ...
- Exemple:
  - « Hash codes » des systèmes de fichiers
  - codes détecteurs d'erreurs

Fonction de hachage : univers

- Les entités sont distinguables les une des autres par l'emprunte (le résultat de la fonction)
- Parmi les champs du schéma on peut trouver un sous ensemble caractérisant chaque entité. Cet identifiant est appelé : **clé**
- On appelle **univers des clés** (**U**) : l'ensemble des valeurs possibles des clés.
- En général, toutes les clés ne sont pas utilisées. On notera **K** l'ensemble des clés utilisées.
- **ATTENTION :** Les clés ne sont pas le résultat d'une fonction de hachage. Elles résultent d'un choix du programmeur.

Fonction de hachage : création

- On appelle **Fonction de hachage** :

Une fonction qui détermine la place d'une entité uniquement d'après sa clé

- H **:** clés ➔ [0..m-1] avec [0..m-1] = Espace des empruntes
- C'est la composition de deux fonctions :
  - 1) **Fonction de codage**
    - Clés ➔ entiers
  - 2) **Fonction d'adressage**
    - entiers ➔ [0..m-1]
- Caractéristiques :
  - Une bonne fonction de hachage doit faire intervenir tous les bits de la clé.
  - Une bonne fonction de hachage doit briser en sous chaînes des bits de la clé.

Fonction de hachage sécurisé

- f(M) telle que f est une fonction de hachage par rapport à M
- f est à collision faible difficile: il est calculatoirement difficile de trouver M significatif tel que f(M)=K
  - Difficulté de trouver un bloc ayant un signature K
- f est à collision forte difficile: il est calculatoirement difficile de trouver M et M' tel que f(M)=f(M')
  - Difficulté de trouver deux blocs ayant la même signature
- Elle est avec clef si son calcul dépend d'une information secrète (la clef K)
- Les algorithmes de hachages
  - Sécurisé : MD5 (emprunte 128bits), SHA1, SHA2 (256/384/512), TIGER
  - Ne plus utiliser MD5 et SHA1. Des pairs de fichiers a empruntes identiques ont été crées. Ils ne sont plus sécurisés.
  - Non sécurisé : MD2, MD4, CRC32

Les outils et la politique

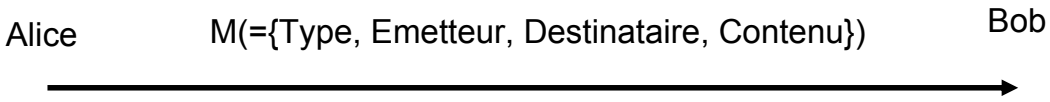
- Pour des raisons d´état, besoin de décoder certaines données/messages
- Limitation par exemple de la taille des clés pour permettre une attaque force brute avec de gros moyens de calcul (officiellement ils « font de l'algèbre »)
- Mouvements sur Internet de démonstrations de craquages massivement parallèles pour pousser à l'autorisation de clés suffisamment grandes
- USA : limitation à 40 bits des clés des systèmes symétriques à l'exportation
- En France autorisation personnelle sans déclaration de clés sur 128 bits max (cryptage symétrique). Décret n99-199 du 17 mars 1999
  - www.scssi.gouv.fr
- Pacte Ukusa datant de 1948 sous contrôle de la NSA : USA + GB + Canada + Australie + Nouvelle-Zélande associés pour espionner les communications mondiales.

RENSEIGNEZ VOUS !!!

Plan de cours

Notations

- Pour chaque échange de messages, on a:
  - Type, Emetteur, Destinataire, Contenu
  - Type ➔Sémantique du message (but)
  - Emetteur ➔ expéditeur du message (identifié par @IP)
  - Destinataire ➔ récepteur du message (identifié par @IP)
  - Contenu ➔ Informations nécessaires au message
- Alice envoie a Bob le message M :



Notations

- Cryptage symétrique
  - $\{M\}^{SYM}_{clef}$ pour crypter et décrypter.
- Fonctions de hachage et signature
  - $\{M\}^{H}_{meth}$ calculer le résumé avec la méthode « meth »
- Signature d'un bloc d'informations M par Alice :
  - $\{M\}^{SIG}_{alice} = \{\{M\}^{H}\}^{SYM}_{CLEF}$

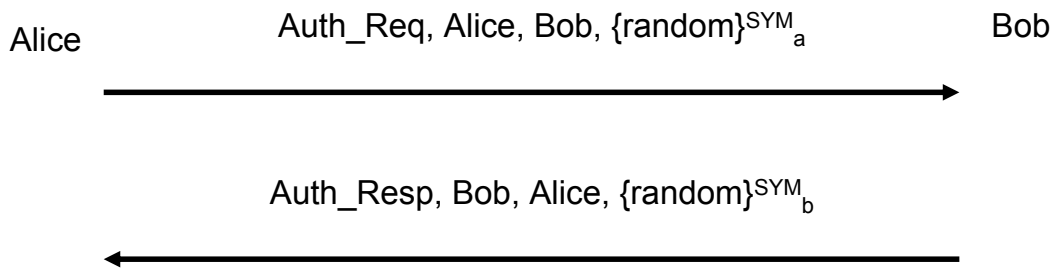
Authentification par un CS symétrique (solution 1)

Utilisation des CS symétriques

- Partage des clefs privées. Connaissances des participants :
  - Alice connait
    - sa propre clef A
    - Date de validité : Date début a / Date fin a
    - la clef B de Bob
  - Bob connait
    - sa propre clef B
    - Date de validité : Date début b / Date fin b
    - la clef A de Alice
- Protocole permettant à Alice d'authentifier Bob

Authentification par un CS symétrique (solution 1)

Utilisation des CS symétriques



- Assure aussi la confidentialité et l'intégrité
- Possibilité d'utiliser une clef unique pour identifier le couple Alice/Bob
- Peu extensible ➔ trop de clefs à échanger entre les acteurs !

Utilisation des CS symétriques

Authentification par un CS symétrique (solution 2)

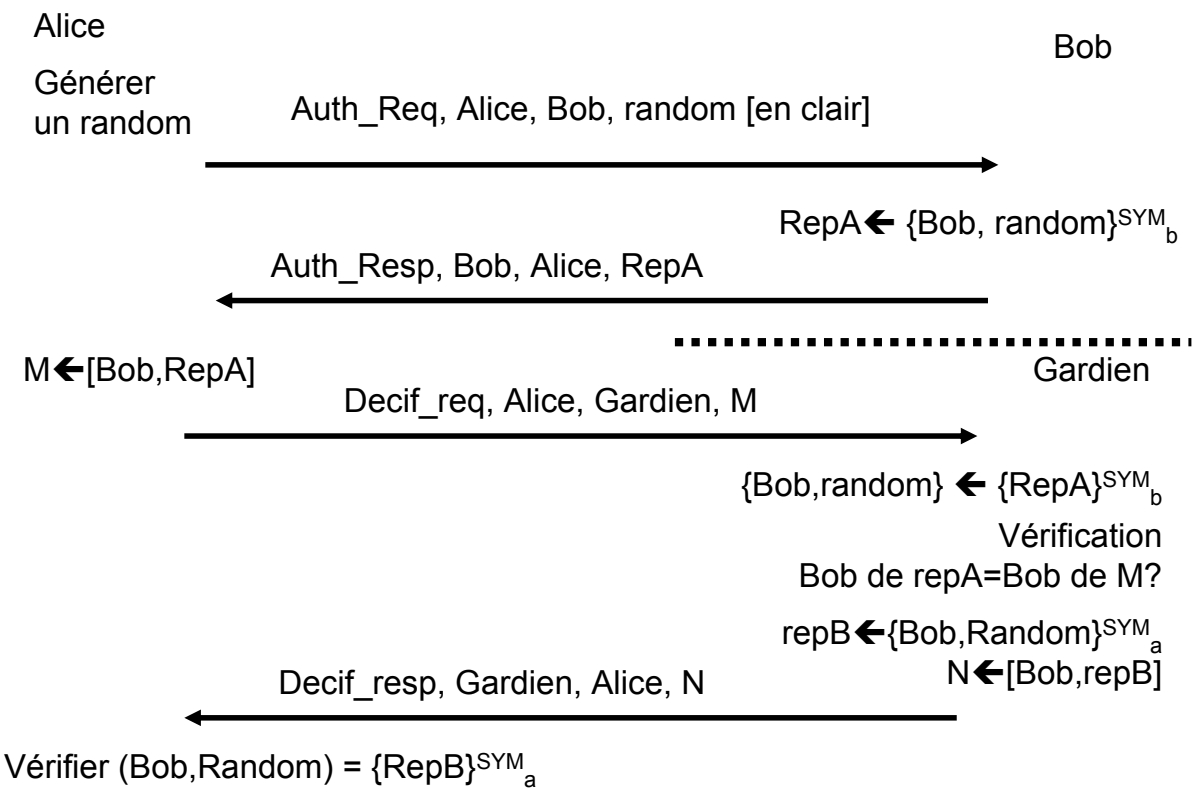
- Utilisation d'un gardien des clefs
  - Exemple : kerberos [ie cerbere], le gardien des enfers
  - Chaque participant connaît sa clef et celle du gardien G
  - La gardien connait toutes les clefs (a,b,c)
  - Les informations (clefs) sont protégées en intégrité et confidentialité
  - Protection par une clef connue du gardien seulement (en général)
- Dans un système à clef privée:
  - Le gardien est un point faible
  - Il doit conserver toutes les clefs !!!
  - S'il est compromis, tout le monde l'est !!!
- Alice connait sa clef A, date de validité : date début a / Date fin a
- Bob connait sa clef B, date de validité : date début b / Date fin b
- Rappel: Le gardien connait toutes les clefs

Utilisation des CS symétriques

Authentification par un CS symétrique (solution 2)

- But de l'authentification par gardien des clefs :
  - Protocole permettant à Bob de prouver à Alice qu'il est Bob
  - Bob détient un secret sur lequel repose l'authentification
  - Bob ne doit pas révéler le secret à Alice
  - Il existe un tiers fiable qui a authentifié Bob (gardien des clefs ou annuaire de certificats)

Authentification par un CS symétrique (solution 2)



Authentification par un CS symétrique : Kerberos

- **Motivations :**
  - développé au MIT pour le projet Athena
  - protéger les serveurs partagés des accès non autorisés depuis les stations de travail (plusieurs milliers)
- **Principes directeurs**
  - mode d'exécution client-serveur
  - vérification de l'identité d'un « client » (utilisateur sur une station)
  - contrôle du droit d'accès à un serveur pour le client
  - fournit au client une clé d'accès (*ticket*) pour le serveur
- Gestions des clefs:
  - Utilisation du cryptage symétrique
  - **clef différente pour chaque serveur**
  - **clef valide pour une période de temps finie**

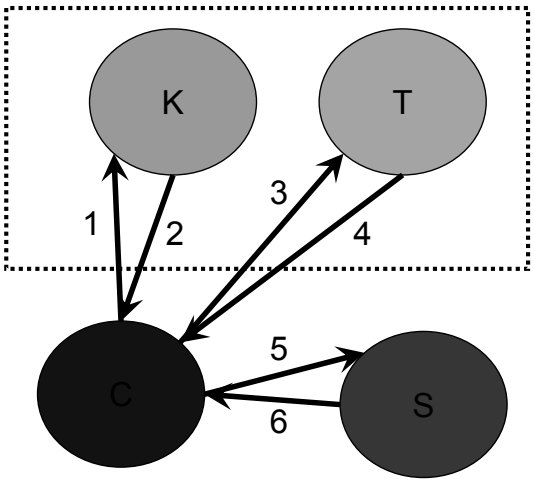
Authentification par un CS symétrique : Kerberos

- **Principe de fonctionnement : certificats « infalsifiables »**
- **Ticket** : caractérise une session entre un client C et un serveur S
  - **$Tcs = \{S, C, adr, Td, life, Kcs\}_{Ks}$**
  - **adr : adresse IP du client**
  - **Td : heure de début de session**
  - **life : durée maximale de vie la de session**
  - **Kcs : clé de session partagée par C et S**
  - **Ks : clé permanente du serveur S**
- **Authentifieur** : caractérise une autorisation pour le client à un instant t
  - **$Acs(t) = \{C, adr, t\}_{Kcs}$**
  - **généré par le client**
  - **permet une authentification « permanente » par le serveur**

Authentification par un CS symétrique : Kerberos

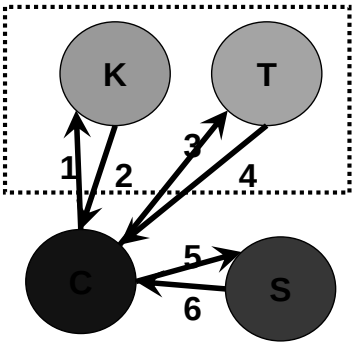
- **Architecture**

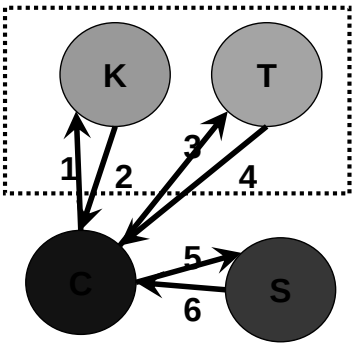
1. accès au serveur Kerberos K : authentification du client
2. Retour d'un ticket pour accéder au serveur de ticket
3. Accès au serveur de ticket T : contrôle d 'accès au serveur S
4. retour d 'un ticket pour accéder au serveur S
5. accès au serveur S



Authentification par un CS symétrique : Kerberos

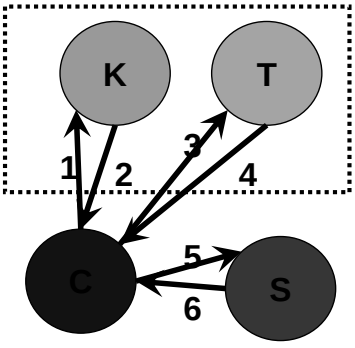
- (1) Demande par C d'un TGT (Ticket Granting Ticket) à K
  - Dé/Chiffré avec le mot de passe utilisateur
  - Message M1: tgt_req, C, K, {C, T}
  - K génère une clé de session KCT pour chiffrer le dialogue entre C et T
  - K génère un ticket TGTct pour autoriser l'accès du client C au serveur T
  - TGTct = {T, C, adr, td, life, KCT}$^{SYM}_{T}$
  - K connaît la clé T (de T)
- (2) Récupération du TGT par C
  - Message M2: tgt_resp, K, C, {{KCT}$^{SYM}_{C}$, TGTct}
  - C déchiffre { KCT }$^{SYM}_{C}$ à l 'aide de sa clef C et mémorise la clé KCT
  - C mémorise le ticket TGTct (sans pouvoir le déchiffrer)

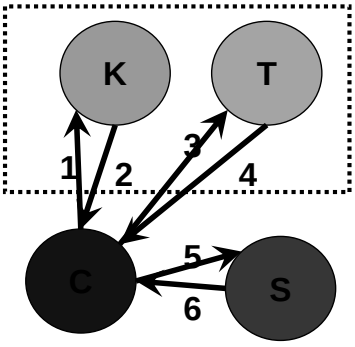




Authentification par un CS symétrique : Kerberos

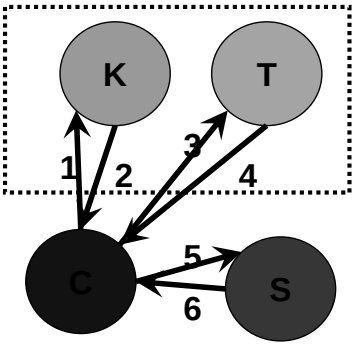
- (3) Demande d'un « *Service Ticket* » à T à l'instant **t1**
  - C construit un authentifieur : Act(t1) = {C, adr, t1}$^{SYM}_{KCT}$
  - message M3 : st_req, C, T, { Act(t1) ; TGTct ; S }
  - T déchiffre le ticket TGTct à l'aide de sa clé T, vérifie sa validité, et récupère ainsi la clé de session KCT
  - T déchiffre l'authentifieur Act à l 'aide de la clé de session KCT (obtenue dans TGTct) et récupère l'identification du client
  - T contrôle le droit d 'accès du client C au serveur S
  - T génère une clé de session **KCS** pour chiffrer le dialogue entre C et S et génère un ticket **STcs** pour autoriser l'accès du client C au serveur S
  - **STcs** = {S, C, adr, td, life, **KCS**}$^{SYM}_{S}$
  - T connaît la clef S du serveur S
- (4) Obtention du ticket
  - message M4 : st_resp, T, C, {KCS, **STcs**}$^{SYM}_{KCT}$

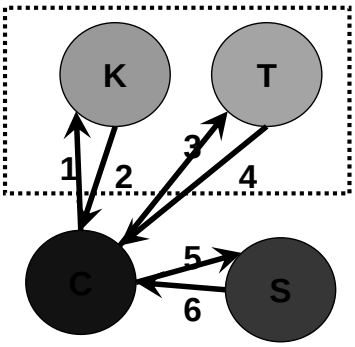




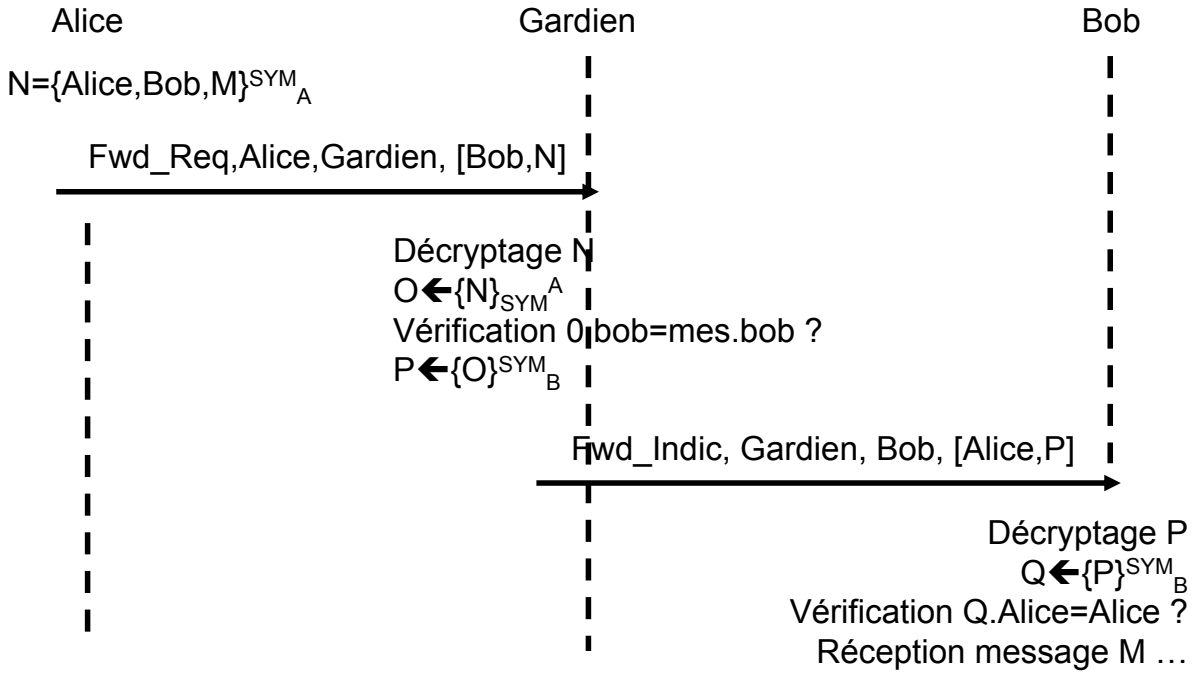
Authentification par un CS symétrique : Kerberos

- (5) Requête au serveur S
  - Rappel : message M4
    st_resp, T, C, $\{KCS, STcs\}^{SYM}_{KCT}$
  - C déchiffre le message M4 à l 'aide de la clé KCT
  - C mémorise le ticket STcs (sans pouvoir le déchiffrer) et KCS
  - C construit un authentifieur :
    $Acs(t2) = \{C, adr, t2\}^{SYM}_{KCS}$
  - Message M5: serv_req, C, S, {requête, STcs, Acs}
- (6) Traitement de la requête
  - S déchiffre le ticket STcs à l 'aide de sa clef S, vérifie sa validité (temporelle, authentification,...)
  - S récupère la clé de session KCS
  - S déchiffre l'authentifieur Acs(t2) à l 'aide de la clé de session KCS et vérifie sa validité (temporelle)





Confidentialité avec un CS symétrique



- Le message M peut être une clef privée (symétrique) de session
- On évite ensuite le gardien comme intermédiaire (maillon faible, goulot)

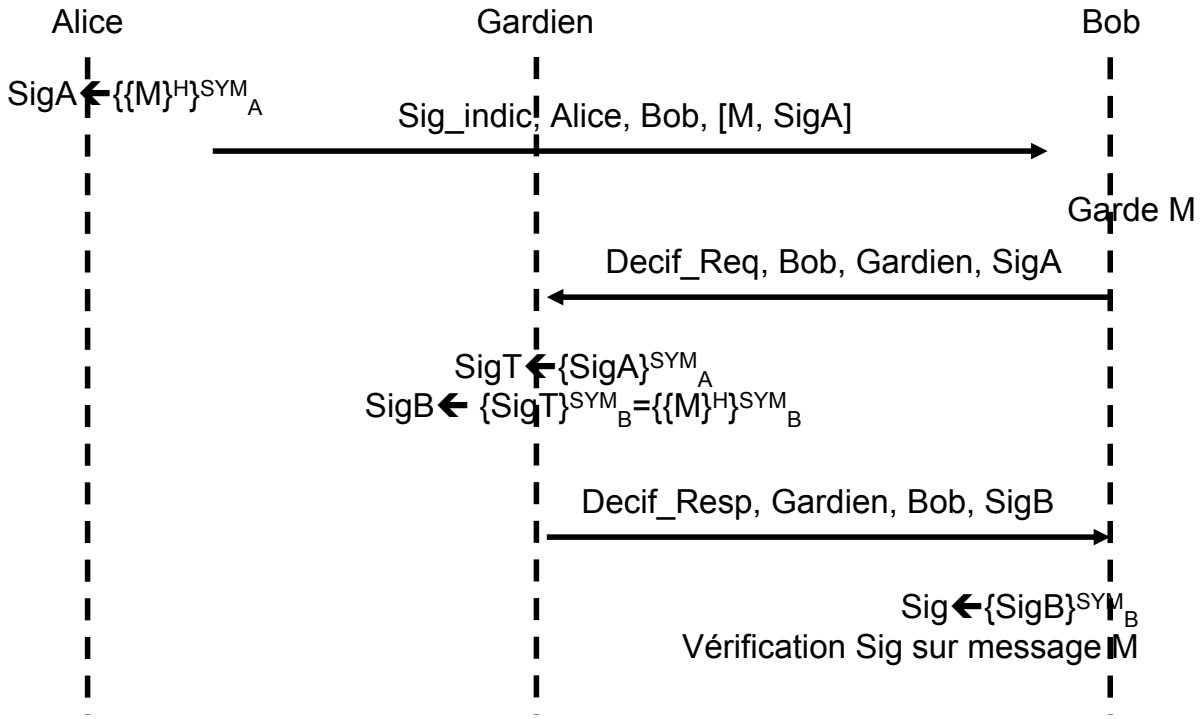
Intégrité et signature

- Alice doit envoyer à Bob un message, tel que Bob puisse contrôler que le message n'a pas été modifié et a bien été créé par Alice.
- On pourrait utiliser le principe de confidentialité basé sur **la possibilité de générer des données correctes par les usagers autorisés** détenteurs du secret.
  - L'intégrité ne peut être mise en cause que par les détenteurs du secret.
  - **Problème**: La vérification de l'intégrité est alors **coûteuse** si les données sont **longues [cryptage nécessaire].**
- **Solution**: Chiffrer une information **courte** caractéristique du message grâce à une **fonction de hachage à sens unique.**

Signature

- Signature
  - Une signature manuscrite idéale est réputée posséder les propriétés suivantes:
  - La signature **ne peut-être imitée et authentifie** le signataire.
  - Elle prouve que le signataire a délibérément signé le document.
  - La signature appartient à un seul document (elle **n'est pas réutilisable**).
  - Le document signé ne peut être partiellement ou totalement **modifié**
  - La signature peut être **contrôlée et** ne peut-être **reniée**.
- Base de la signature numérique: une fonctions de hachage
  - H sécuritaire et d 'une fonction à sens unique f avec brèche.
  - La signature est composée de $f^{-1}(\{M\}^H)$
  - Seul le signataire sait calculer f^{-1}
  - Tout le monde peut calculer H et f et donc pour M donné vérifier la signature
  - Si H est a collision faible, on ne pourra pas coller une fausse signature sur un document à créer
  - Si H est à collision forte difficile Estelle ne pourra pas fabriquer 2 documents, un que Bob peut signer, l'autre pas, ayant le même résumé donc la même signature

Intégrité et signature avec un CS symétrique



Utilisation des CS symétriques

Plan de cours

Introduction

Concepts et Terminologie

Types d'attaques

Les politiques de sécurité

Les outils de la sécurité

Utilisation des CS symétriques

Utilisation des CS asymétriques

Autres Utilisations des CS

Les certificats

Authentification des personnes

Utilisation des outils

Notations

- Pour chaque échange de messages, on a:
  - Type, Emetteur, Destinataire, Contenu
  - Type ➔Sémantique du message (but)
  - Emetteur ➔ expéditeur du message (identifié par @IP)
  - Destinataire ➔ récepteur du message (identifié par @IP)
  - Contenu ➔ Informations nécessaires au message
- Alice envoie a Bob le message M :

Alice M(={Type, Emetteur,Destinataire, Contenu}) Bob

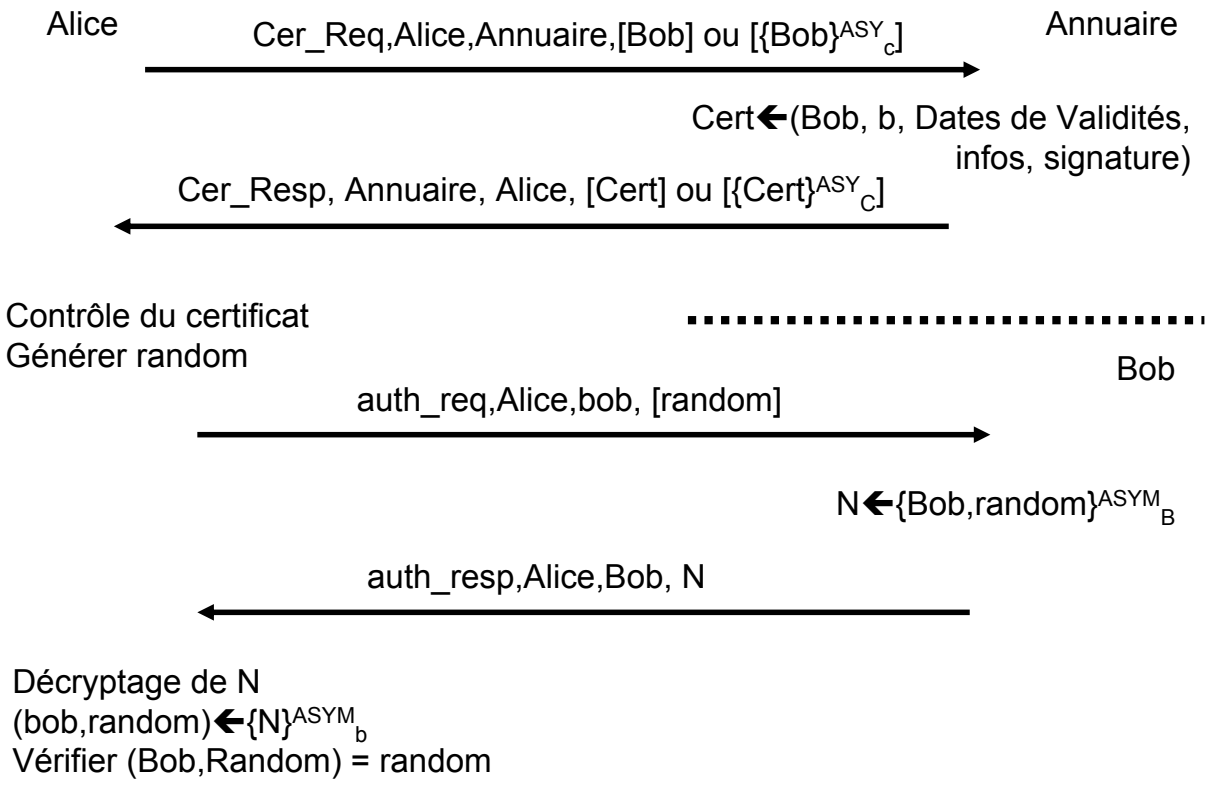
Notations

- Cryptage asymétrique
  - Si on crypte avec l'un, on décrypte avec l'autre
  - « clef » (minuscule) est la clef publique
  - « CLEF » (majuscule) est la clef privée
  - $\{M\}^{ASYM}_{clef}$ est le symétrique de $\{M\}^{ASYM}_{CLEF}$
- Fonctions de hachage et signature
  - $\{M\}^{H}_{meth}$ calculer le résumé avec la méthode « meth »
- Signature d'un bloc d'informations M par Alice :
  - $\{M\}^{SIG}_{alice} = \{\{M\}^{H}\}^{ASYM}_{CLEF}$

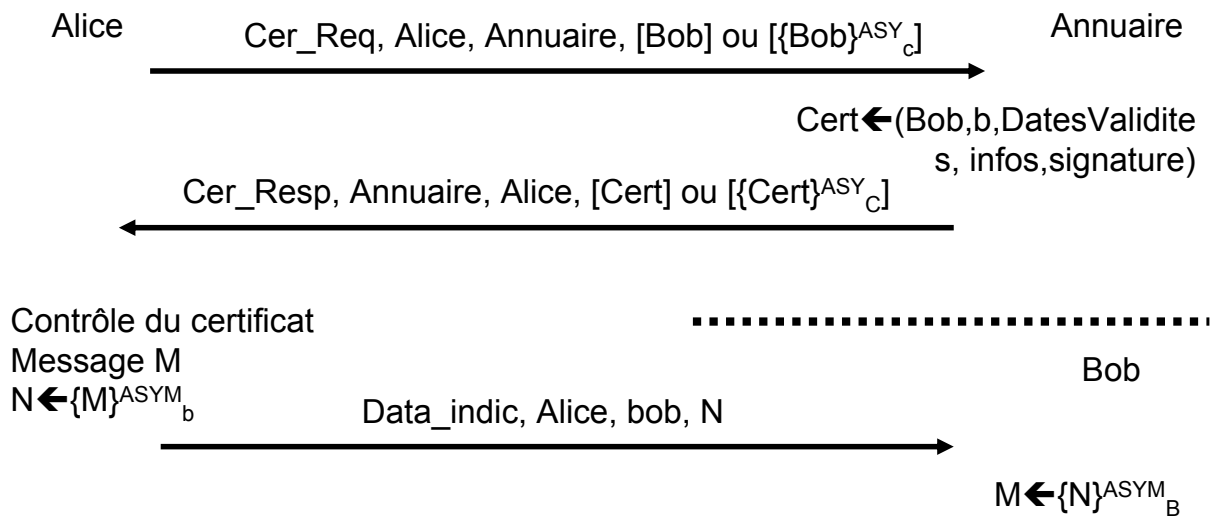
Authentification par un CS asymétrique

- Systèmes à clefs publiques: Annuaire de clefs
  - L'annuaire possède les clefs publiques des membres
  - L'annuaire a sa clef (partie publique « c » et privée « C »)
- Les informations de l'annuaire sont protégées en intégrité
- Chaque participant connaît sa clef privée, sa clef publique et la clef publique de l'annuaire
  - Alice connaît sa clef privée A / publique a, et c
  - Bob connaît sa clef privée B / publique b, et c
  - L'annuaire connaît C / c, le certificat de a (donc clef a) et de b (donc la clef b)

Authentification par un CS asymétrique (solution 2)



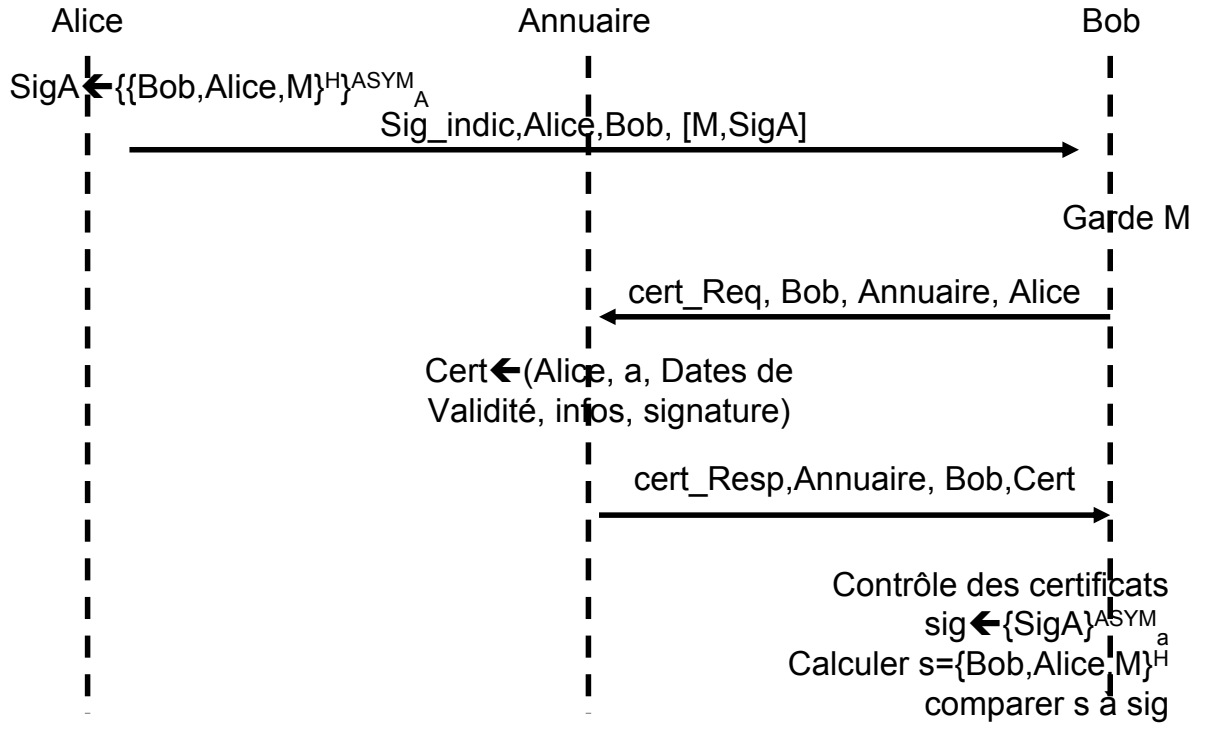
Confidentialité avec un CS asymétrique



- Rappel: très peu utilisé car très lent !!
- On utilise le cryptage asymétrique pour échanger une clef de session
  - Principe de SSH

Utilisation des CS asymétriques

Intégrité et signature avec un CS asymétrique



Utilisation des CS asymétriques

Intégrité de flots de messages

- Un flot d'échanges de longue durée doit être caractérisé par une **connexion**.
  - Problème de rejeu ("replay")
  - Un message dupliqué **peut être inséré dans un flot par un usager malveillant**
  - Il peut être correct du point de vue de la connexion, séquence et signature mais menacer l'intégrité de l'application
- Rejeu possible d'un message
  - D'une ancienne connexion
  - De la connexion courante
- Intégrité du flot de message
- Utilisation d'un **Nonce** (Used Only Once), qui distingue chaque message:
  - Numéro de séquence sur un modulo grand (sur 32 ou 64 bits)
  - Estampillage par horloge (horodatage)
  - Nombre aléatoire

Considérations ad-hoc: Stockages des clefs

- Clef publique de l'autorité, ne doit pas pouvoir être modifiée
  - Dans le code en dur
  - sur un support fiable (carte à puce)
- Clef privée de l'utilisateur, ne doit pas pouvoir être lue:
  - sur un support confidentiel (carte à puce) ou un fichier chiffré avec un mot de passe (local au poste ou sur disquette)
  - SSH ➔ clef privée droit 700, clef sur stick USB
- Certificat de l'utilisateur:
  - Annuaire+support local ou carte ou disquette
  - Annuaire central+version locales (cache, annuaire privé)

Considérations ad-hoc: Utilisation des clefs

- Plus on utilise une clef plus elle est vulnérable !
  - Clef utilisée pour chiffrer une suite de transfert de fichier
  - Clef utilisée pour chiffrer un numéro de carte bleue
- Plus elle sert a protéger des données pérennes, plus elle doit être fiable
  - Signature électronique d'un article de presse
  - Signature électronique d'un testament
- *On peut utiliser des canaux très lents mais très fiables pour véhiculer des clefs qui seront utilisées sur des voies plus rapides et moins fiables*

Plan de cours

Autres Utilisations des CS

Partage de secret: protocoles à seuil

- Certaines opérations sont suffisamment sensibles pour devoir **engager la responsabilité de plusieurs** personnes.
- On peut faire **vérifier l'identité** de plusieurs usagers simultanément possesseurs d'un **mot de passe** pour engager une action. [ex: attaque nucléaire]
- Mais cette approche peut ensuite être encore raffinée en souhaitant **donner une part de responsabilité plus importante** selon un grade:
  - Ex : Il suffit de la présence du responsable financier pour ouvrir le coffre ou de trois chefs de service ou ...
- **Le problème du partage d'un secret:**
  - Comment diviser une clé d'accès représentée par **une valeur numérique V en k parts**
  - De telle façon qu'un groupe de porteurs de t+1 parts peuvent reconstituer la clé alors qu'un groupe de porteurs de t parts ne le peuvent pas.
  - Les porteurs de parts **doivent pouvoir reconstituer V** dans un système informatique d'autorisation sans jamais connaître V.

Autres Utilisations des CS

Partage de secret: protocoles à seuil (Shamir)

- V **valeur numérique** entière
  - On **génère aléatoirement t valeurs entières** a_1 , a_2 , ... , a_t
  - **On leur associe un polynôme** dont le terme constant est **V** :
- $P(x) = a_t.x^t + a_{t-1}.x^{t-1} + ... + a_1 x + V$
- Une part du **secret est un couple (x_i,$P(x_i)$)** x_i non nul
  - les parts sont générées par des x_i différents
- Pour éviter une possible **attaque force brute** par un groupe de porteurs agissant par essais et erreurs pour compléter leur connaissance
  - on choisit un entier premier N grand
  - les calculs sont faits en arithmétique modulo N
- Pour retrouver le secret, niveau 2^{nd} ➔ résolution d'un système à (t+1) valeurs inconnues (a_1 , a_2 , ... , a_t ,V) avec (t+1) équations [parts]

Notarisation par un CS asymétrique

Autres Utilisations des CS

- But: non répudiation d'une action (commande) entre A et B
- Moyen: Utilisation d'un tiers de confiance (N)
- A connait la clef publique de N (kn), la clef publique de B (kb), sa clef KA/ka
- B connait la clef publique de N (kn), la clef publique de A (ka), sa clef KB/kb
- N connait la clef publique de A (ka), la clef publique de B (kb), sa clef KN/kn

Notarisation par un CS asymétrique

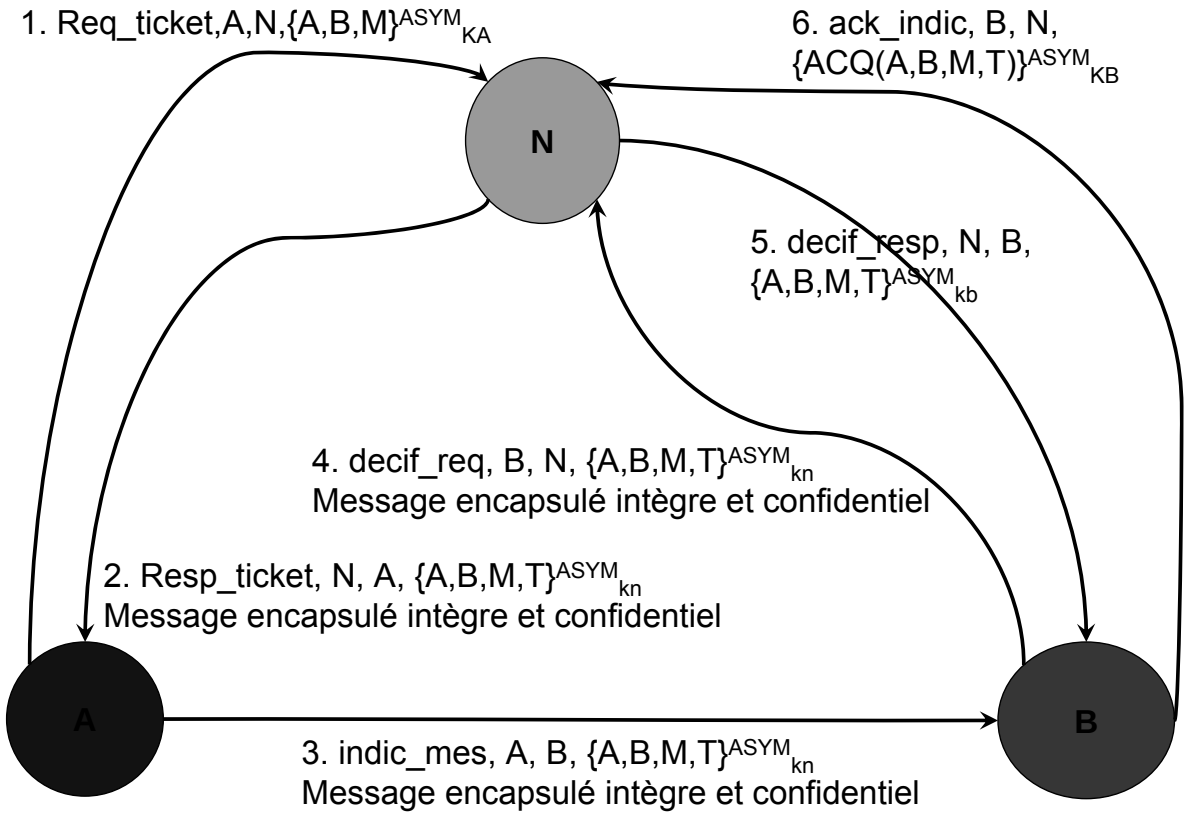
Autres Utilisations des CS

1. **A souhaite envoyer le message M** au destinataire B de façon notarisée (notariée ?)
   - il envoie au notaire $\{A,B,M\}^{ASYM}_{KA}$
2. **Le notaire qui reçoit la transaction** peut la décoder puisqu'il connaît **ka**.
   - Il date la transaction T et la journalise
   - Enregistrement de A , B , M , T
   - La transaction ne pourra pas ensuite être reniée par A.
3. **Le notaire possède une clé secrète personnelle KN** qu'il utilise pour signer la transaction
   - $M= \{A,B,M,T\}^{ASYM}_{KN}$, $T=\{A,B,M,date\}^{H}$
   - Il renvoie le message M en réponse à A qui va la conserver pour preuve de la notarisation effectuée.

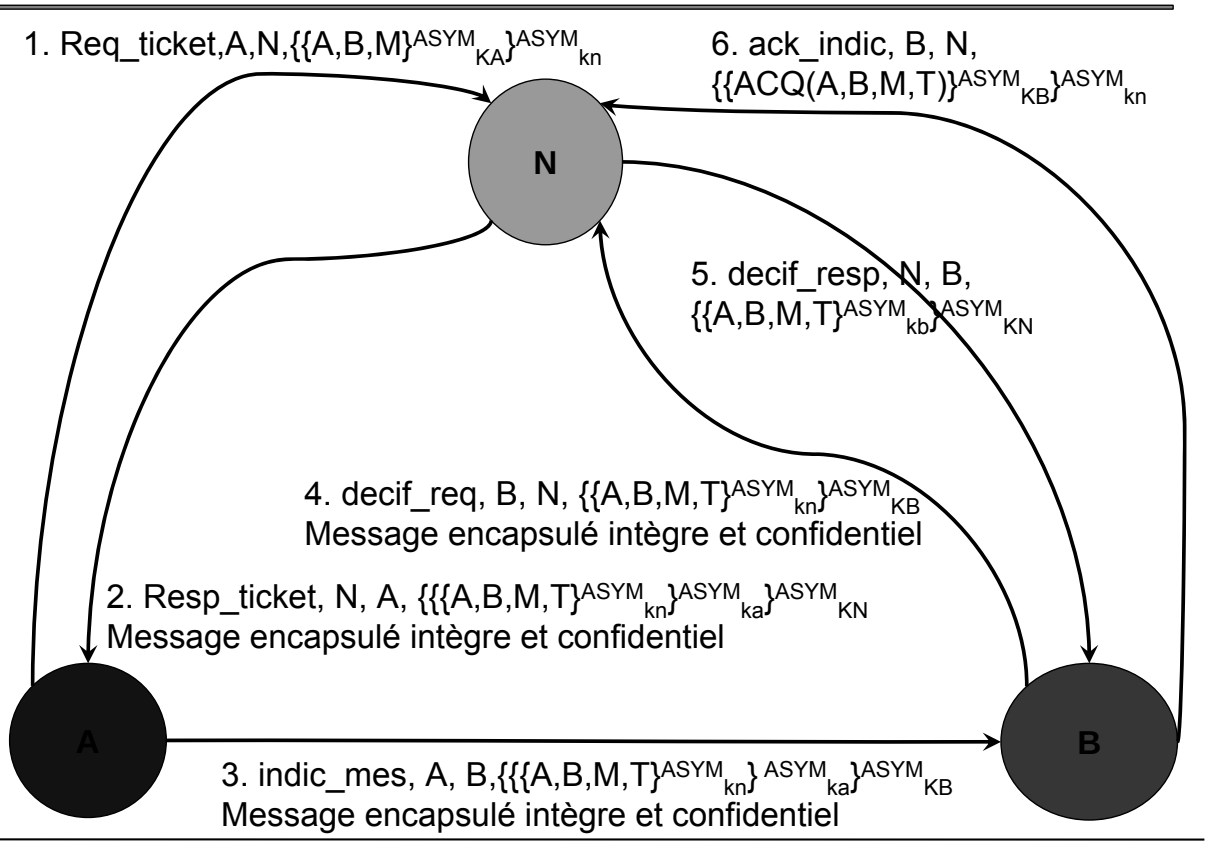
Notarisation par un CS asymétrique

4. **L'émetteur A envoie alors la transaction** à son destinataire B sous la forme M signée par le notaire (il ne peut avoir modifié celle ci entre temps).
   - B ne peut encore interpréter les informations mais il enregistre M pour preuve de la requête de A.
5. **Pour connaître M, B demande au notaire** le déchiffrement de M
   - Le notaire envoie à B la transaction chiffrée avec la clé de kb soit $\{A,B,M,T\}^{ASYM}_{kb}$.
   - Seul B peut la lire ➔ confidentialité l'intégrité
   - Pour l'authenticité, il faudrait $\{\{A,B,M,T\}^{ASYM}_{kb}\}^{ASYM}_{KC}$
6. **Pour terminer complètement le protocole** il faut que le notaire dispose d'une preuve de remise à B soit une réponse $\{ACQ(A,B,M,T)\}^{ASYM}_{KB}$ que le notaire enregistre.

Notarisation par un CS asymétrique



Notarisation par un CS asymétrique



Plan de cours

Certificats et cryptages asymétriques

- Rappel : Cryptage asymétrique
  - PB : Tout repose sur la confiance dans la provenance de la clef publique.
  - Attaque du type Man-In-The-Middle : Celui qui souhaite écouter les messages de votre correspondant vous remet une fausse clef publique pour cette personne.
  - Exemple SSH ne gère pas les certificats
- Solution : une autorité est chargée de signer les clefs publiques
  - Cette autorité s'appelle l'autorité de certification (« Certificate Authority » ou CA)

Les autorités de certifications (CA)

- AC: autorité de certification
  - Norme de représentation des certificats X509
  - Norme de protocole d'accès: LDAP
  - Elle chiffre (avec sa clef privée) une empreinte de
    - L'identité de son titulaire, personne, serveur ou application (*Distinguished Name of Subject*)
    - Sa (celle du titulaire) clef publique
    - Les informations relatives à l'usage de cette clef : période de validité, type des opérations possibles, etc …
  - L'ensemble est appelé certificat X509.
  - Les certificats X509 font l'objet d'une norme : ITU-T X509 international standard V3 1996, RFC2459

Les autorités de certifications (CA)

- Rôle :
  - Vérifie les demandes de certificats (Certificat Signing Request)
  - Génère les certificats et les publie
  - Génère les listes de certificats révoqués (Certificat Revocation List)
- Vérifie (par l'intermédiaire d'une autorité d'enregistrement) :
  - l'identité des demandeurs de certificats et les éléments de la demande
  - Recueille et vérifie les demandes de révocation

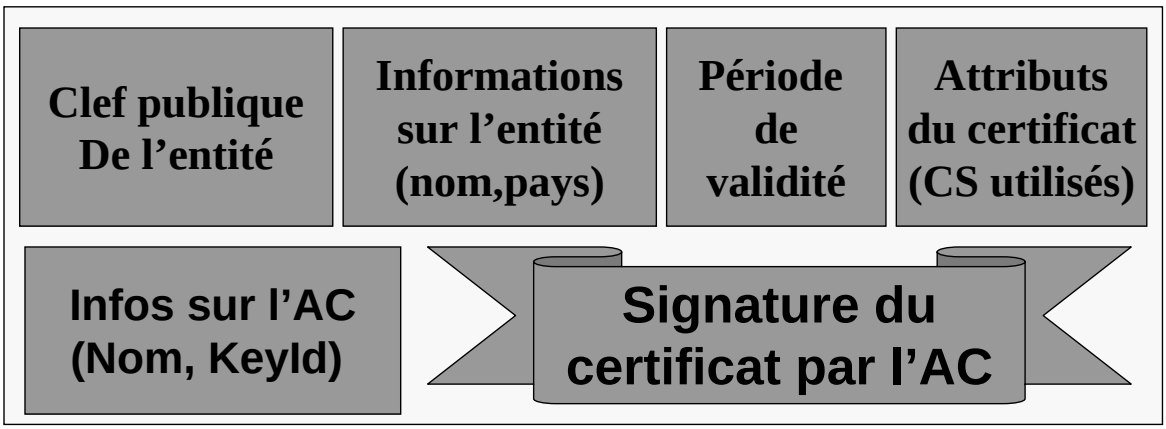
Les autorités de certifications (CA)

- Contrôle des certificats
  - Toutes entités impliquées dans un schéma à clef publique doit détenir la clef publique de l'autorité de certification.
  - Tout accès à un certificat doit être contrôlé:
    - Vérifier que la signature est valide
    - Vérifier que la date courante est dans la période de validité
    - Vérifier la clef publique du certificat en vérifiant la signature qui y a été apposée à l'aide de la clef publique de l'autorité de certification (CA).
  - Pour éviter les rejeux de certificats invalidés le serveur d'annuaire doit :
    - Soit s'authentifier
    - Soit dater et signer sa réponse
    - Soit transmettre périodiquement des listes de révocation datée et signées
- **On ne fait confiance qu'aux clefs signées !!!**
- **Attention, il existe des certificats auto-signés (sans CA officiel)**

Utilisation des certificats

- Les certificats sont composés de deux éléments :
  - d'informations sur l'entité propriétaire du certificat
  - d'informations sur l'entité émettrice du certificat (l'annuaire, l'autorité de certification, ...)
- Les informations sur l'entité propriétaire du certificat sont
  - le nom,
  - la clef publique de l'entité à identifier,
  - des informations supplémentaires
- Les informations sur l'entité émettrice du certificat
  - Date de validité du certificat,
  - Le but de la clef
  - L'emprunte du certificat (faite par l'annuaire)
  - Le nom de l'entité émettrice (annuaire)
  - Des informations concernant les algorithmes utilisés

La structure des certificats



- Le certificat établit un lien fort entre le nom (DN) de son titulaire et sa clé publique
  ➔ AUTHENTIFICATION FORTE
- Protocoles : TLS/SSL, S/MIME, VPN, Java, …
- Usages : Horodatage, Signature, E-commerce, E-vote, E-Administration, …

Exemple de certificat

Certificate:
Data:
Version: 3 (0x2)
Serial Number: 13805 (0x35ed)
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=FR, O=CNRS, CN=CNRS-Standard
Validity
Not Before: Apr 24 14:09:48 2006 GMT
Not After : Apr 24 14:09:48 2008 GMT
Subject: C=FR, O=CNRS, OU=UMR7606, CN=src.lip6.fr/emailAddress=postmaster@lip6.fr
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (1024 bit)
Modulus (1024 bit):
00:ec:29:c5:24:d6:4d:e4:b5:31:71:46:2f:15:64:
…
a6:ee:85:31:22:de:74:d8:d1:5f:8a:32:e0:b3:d7:
84:e4:8f:ab:66:92:ad:f8:eb
Exponent: 65537 (0x10001)

X509v3 extensions:
X509v3 Basic Constraints: critical
CA:FALSE
Netscape Cert Type:
SSL Client, SSL Server
X509v3 Key Usage: critical
Digital Signature, Non Repudiation, Key Encipherment
X509v3 Extended Key Usage:
TLS Web Server Authentication, TLS Web Client Authentication
Netscape Comment:
Certificat serveur CNRS-Standard
X509v3 Subject Key Identifier:
79:F7:B4:D3:D8:E9:B8:ED:3C:A1:85:A6:DD:FA:68:CC:74:8C:82:1F
X509v3 Authority Key Identifier:
keyid:67:59:A5:E5:07:74:49:03:EF:05:CF:CC:2E:A4:18:D5:10:C8:9E:3C
DirName:/C=FR/O=CNRS/CN=CNRS
serial:02
X509v3 Subject Alternative Name:
DNS:src.lip6.fr
X509v3 CRL Distribution Points:
URI:http://crls.services.cnrs.fr/CNRS-Standard/getder.crl
Signature Algorithm: sha1WithRSAEncryption
54:a4:1c:c2:21:fd:06:9b:df:bd:50:4b:d2:ae:e0:3f:46:64:

Les certificats: Aspects Juridiques

- Usages : Horodatage, Signature, E-commerce, E-vote, …
- Validité de l'écrit électronique
  - Reconnaissance juridique de la signature électronique
  - Obligation de dématérialisation des procédures
  - E-Administration (déclaration d'impôts)
- Le cadre est défini par
  - La loi du 13 mars 2000
  - Le décret du 30 mars 2001
  - Le décret du 18 avril 2002
  - L'arrêté du 31 mai 2002.

Création de certificat (Annuaire publique)

Les certificats

Alice

Annuaire
Autorité de certification (AC)

Obtention du certificat de l'AC

Génération d'une clef publique A/a
Génération d'une clef privée MP
Génération d'un certificat (Alice, A, Date)
Stockage (sûr) $\{(Alice, A, Date)\}^{SYM}_{MP}$
C← $\{(Alice, a, Date)\}^{ASYM}_{ac}$

certCreation_req, Alice, AC, C

Décryptage de C
D← $\{(Alice, a, Date)\}^{ASYM}_{AC}$
Contrôle EXTERNE de l'id d'Alice
MAJ de l'annuaire de certificat
Cert(Alice)=$(Alice,a,Date,\{\{Alice,a,Date\}^{H}\}^{ASYM}_{ac}$

certCreation_resp, AC, Alice, [Cert(Alice)]

Création de certificat (Annuaire privé)

Les certificats

Alice

Annuaire
Autorité de certification (AC)

certCreation_req, Alice, AC, ∅

Contrôle EXTERNE de l'id d'Alice
Génération d'une clef publique A/a
Génération d'une clef privée MP
Génération d'un certificat (Alice, A, Date)
Génération C←$\{(Alice, A, Date)\}^{SYM}_{MP}$

certCreation_resp, AC, C

Par fichier, disquette, carte à puce

MP par voie confidentielle
Par fichier, disquette, carte à puce

MAJ de l'annuaire de certificat
Cert(Alice)=$(Alice,a,Date,\{\{Alice,a,Date\}^{H}\}^{ASYM}_{ac}$

Révocations de certificats

- CRL = « certificat revocation list »
- Les CRL : la liste des certificats révoqués, liste signée par la CA
  - Similaire à l'opposition des CB/chèque en cas de vol
  - Pas encore de CRL incrémentale (le certificat contient une url du fichier de crl)
  - La révocation est une limite théorique au modèle des PKIs.
- Les navigateurs doivent vérifier par eux-même les CRL
  - Mal implémenté ➔ souvent non vérifié

Plan de cours

Authentification des personnes

Rappels & généralités

- Le contrôle d'accès est la base des mécanismes informatiques:
  - Il permet de spécifier la politique dans le domaine de l'informatique.
  - Il définit la façon dont le système contrôle ces droits.
  - Il devrait, en théorie, encapsuler toutes les autres techniques informatiques
  - Pour l'instant ce n'est pas le cas.
- Principe du **moindre privilège** :
  - Un objet ne doit disposer que des droits qui lui sont strictement nécessaires pour réaliser les tâches qui lui sont dévolues.
- Utilisation de politique obligatoire :
  - La politique doit le moins possible dépendre des utilisateurs en tant que personne, mais reposer sur les rôles de la politique de sécurité du système d'information.

Authentification des personnes

Méthodes d'authentification des personnes

- L'**authentification** = **vérification de l'identité** d'une entité.
- L'une des mesures les plus importantes de la sécurité:
  - Impossible d'assurer la confidentialité, l'intégrité, la non répudiation sans la garantie de l'identité de l'entité soumettant une requête.
- L'authentification devrait être assurée en continue (pas une fois pour toute à l'ouverture d'un objet ou en début de session)
- Un personne peut quitter son poste en le laissant ouvert :
  - procédure de déconnexion automatique
  - procédure d 'authentification périodique
- Une entité informatique peut être corrompue
  - une substitution peut avoir lieu (surtout en réseau, nécessité de protocoles de sécurité)
- L'authentification des personnes peut se faire par trois méthodes:
  - Ce que connaît l'utilisateur (Mot de passe),
  - Ce que détient l'utilisateur (carte...),
  - Ce qu'est l'utilisateur (Méthode biométrique)

Authentification par connaissance

Authentification des personnes

- Le mot de passe, le code confidentiel
  - Technique la plus simple et la plus répandue
- Problèmes bien connus:
  - Si le mot de passe est simple il peut être trouvé par une attaque par dictionnaire
  - Si le mot de passe est compliqué l'utilisateur le note pour s'en souvenir !
  - La frappe du mot de passe peut être publique
  - Les mots de passe doivent être stockés (point sensible)
- Quelques parades:
  - **Ne jamais utiliser** son login, son nom, le nom de son chien, son n°de tél., un mot d'un dictionnaire...
  - Utiliser chiffres et lettres avec des caractères spéciaux au moins 6 à 7 caractères, mais trouver un moyen mémotechnique
  - **Obliger l'usager à changer** régulièrement de mot de passe.
  - **Surveiller les tentatives d'accès** illicite par comptage (les afficher).
  - **Prévenir l'usager des connexions** précédentes sur son compte en affichant la date et l'heure (par exemple du dernier accès).

MDP: l'exemple d'Unix/Linux crypt()

Authentification des personnes

- On ne stocke pas les MDP dans /etc/passwd ou /etc/shadow
  - Utilisation d'une fonction à sens unique crypt()
    - L'inverse n'existe pas.
    - Propriété : Si p=p' alors crypt(p)=crypt(p')
    - Propriété : Si crypt(p')=crypt(p) alors p=p'
    - Combinatoire importante: attaque par force brute difficile
    - Alteration pat un paramètre (SALT) pour introduire des différences entre les entités
      - crypt(p, Salt)
- Habituellement, 30% des mots de passe sont devinables

Authentification des personnes

Authentification par objet

- Un secret matérialisé physiquement
  - La clé traditionnelle
  - Une carte magnétique, à code barre, à puce
  - Un stick USB
  - Un porte-clefs générateur de clef temporaire
- Technique simple, répandue.
- Les problèmes :
  - la perte, le vol du support
  - la duplication (plus ou moins facile mais toujours possible)
  - Nécessite souvent l'intervention humaine

Authentification des personnes

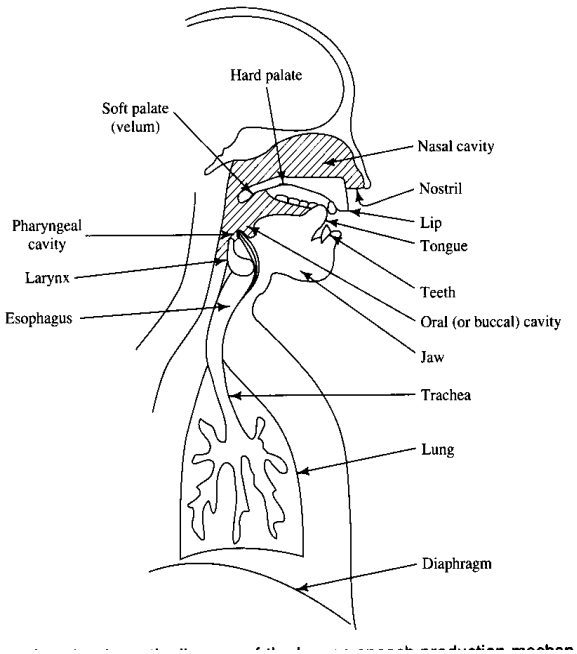
Authentification par l'utilisateur lui-même

- **Les méthodes bio métriques**
- Une solution en rapide développement
  - peut-être très efficace
  - souvent onéreuse,
  - peut-être difficile à accepter dans certains cas par l'utilisateur
- Nécessité d'études approfondies (analyse de la variabilité) du caractère utilisé
  - à l'intérieur du groupe humain des usagers autorisés
  - ou dans une population quelconque
- Incertitudes des techniques bio métriques
  - La variabilité intra-individuelle
  - La variabilité inter-individuelle
- Conduit à deux types d'erreurs possibles:
  - Le rejet à tort d'un individu autorisé
  - L'acceptation à tort d'une personne non autorisée.

Quelques techniques biométriques

- L'empreinte digitale
- la vascularisation de la rétine, de l'iris
- la voix
- la géométrie de la main, du visage
- dynamique de la signature
- dynamique de la frappe clavier
- empreinte génétique
- Thermographie faciale

BIOMETRIQUE – Reconnaissance de voix



Reconnaissance

- The speaker
- Par un code (dépendance au texte)
- Par le timbre (indépendance au texte)

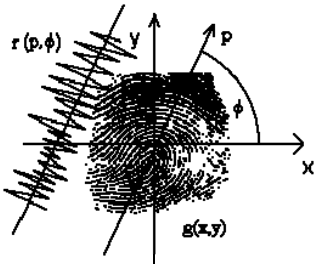
Facile à gérer

Enregistrement possible

Sensibilité aux bruits parasites

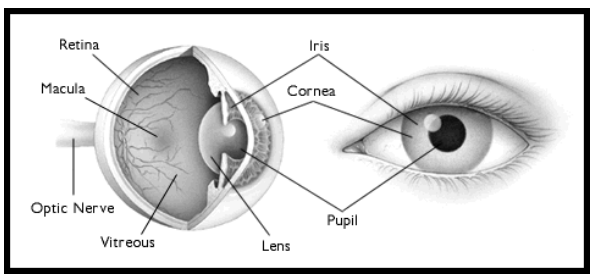
Importance des erreurs (toux)

BIOMETRIQUE – Empruntes digitales



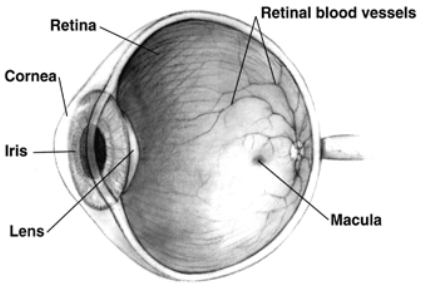
- Reconnaissance géométrique du doigt
  - Très connue et exploité
  - Petite taille des dispositifs
  - Faibles coûts des dispositifs
  - Analyse rapide, faible taux de rejet
  - Assez bien implanté
    - police
    - ordinateurs portables (IBM)
    - Bientôt (déjà) carte d'identité à puce
  - Problèmes
    - Doigts sales ou coupés
    - Besoin de la coopération de l'utilisateur

BIOMETRIQUE – Reconnaissance de l'iris



- Reconnaissance de la géométrie de l'iris
  - Grande quantité d'information
  - Reconnaît les vrais jumeaux
  - Très peu de rejet
  - Reconnaissance à distance
  - Problèmes
    - Non différenciation photo/humain
    - Non différenciation fausse iris/humain
    - Coût élevé

BIOMETRIQUE – Reconnaissance rétinienne

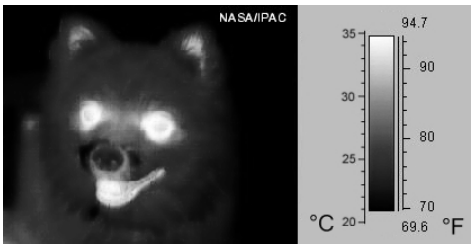


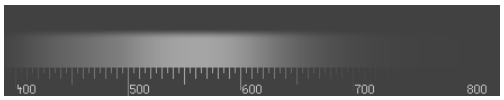
- Reconnaissance rétinienne
  - Vaisseaux sanguins
  - Peu de facteurs de variations (ie peu de maladies)
  - Meilleur taux de réussite
  - Problèmes:
    - Très cher
    - Intrusif donc peu populaire
      Qui veut coller son oeil dans l'objectif ?
    - Devient moins efficace avec le temps (âge de la personne)

BIOMETRIQUE – Reconnaissance faciale

- Reconnaissance de la géométrie faciale
  - Distance entre les yeux, la bouche, le nez, ...
  - Facile à gérer
  - Identification à distance
  - Utile pour l'analyse de foule
  - Problème :
    - Identification impossible des vrais jumeaux
    - Sensible aux problèmes du visages (maladies, accidents)
    - Sensible aux lunettes, piercing
    - Maquillage, masque, perruques ➔ échec

BIOMETRIQUE - Thermographie





- Etude du spectre électromagnétique
  - IR
  - cartographie de la chaleur du visage
  - Bon taux de reconnaissance
  - Reconnaît les vrais jumeaux
  - Problèmes:
    - Coût très élevé
    - Expérimentale

MULTIMETRIQUES

- Biométriques seules sont insuffisantes
- Combinaison de différentes techniques
  - Biométriques
    - Reconnaissance par emprunte digitale
  - Méthodes traditionnelles
    - Mot de passe, Smart Cards avec des informations
- Forte taux de réussite en authentification
- Très résistant
  - Nécessite le cassage des n méthodes
- Coût raisonnable

Authentification des personnes

Authentification – “One Time Passwords”

- Principes :
  - Utilisation de plusieurs mots de passe
  - Une fois utilisé, le mot de passe n’est plus utilisable (Utilisable une et une seule fois)
  - L’utilisateur a une clef privée (le secret) connue par le serveur qui permet de générer le mot de passe
  - L’utilisateur et le serveur on un calculateur logiciel ou matériel
- Génération du mot de passe
  - Le serveur génère un défi et une solution (le mot de passe)
  - Le serveur transmet le défi à l’utilisateur
  - L’utilisateur génère la propre solution à partir du défi
  - L’utilisateur la donne au serveur
  - Si les solutions correspondent, l’utilisateur est authentifié
- Peut être implanté sur des systèmes embarqués
  - Smart Cards, PDAs, portables, Keys Holders, etc...

Authentification des personnes

Authentification – “One Time Passwords”

- Désavantages
  - L’utilisateur doit
    - posséder l’algorithme ou une appareil capable de le faire
    - un liste pré-imprimée de mots de passe
  - Dangereux en cas d’utilisation de la liste !
- Exemple
  - Challenge: 5241
    - Rechercher du MDP 5241 dans la liste
    - Rappel : La liste est générée à partir de la clef secrète
  - Réponse: CLAD ROY TOP BAD CAKE MATH
    - Quand la liste est expirée, on change de clef et on régénère une liste
- Difficile à gérer
- Outil les plus connus: S/Key (Neil Haller, “The S/KEY one-time password system”, 1994), OPIE, SecurID, CryptoCard

Authentification bi-factuelle

- Comme pour l'authentification multimétriques
- On combine deux méthodes :
  - "Quelque chose que vous connaissez" (secret)
    - ie un mot de passe
  - "Quelque chose que vous avez" (appareil)
- L'appareil stocke ou fait tourner un algorithme de génération de clefs uniques configurer pour l'utilisateur
- La plus part des «systèmes OPT » utilise la double identification

Legond-Aubry Fabrice
fabrice.legond-aubry@u-paris10.fr

SSI

Sécurité des Systèmes Informatiques

Sécurité réseau

Plan de cours

Introduction

Introduction

Attaques de niveau 1

Attaques niveau 2: ethernet

Attaques niveau 3: IP/ICMP

IPSEC - VPN

Attaques niveau 4: TCP

SSL/TLS – Parefeu - NAT

Analyse

Vocabulaire

Introduction

- Attaque par eMail
  - Hoax : fausse information, rumeur destiné à saturer la messagerie
  - Mail bombing : Attaque consistant à générer beaucoup de mail pour saturer un serveur de mail
  - Spamming : Envoie de courriers non sollicités à but commercial
  - Phishing : Envoie de courriers permettant le détournement d'informations
- Attaque Réseau TCP/IP
  - Spoofing : Forger un message réseau faux et/ou malformé
  - Flooding : Inondation en vu de saturer une machine
  - Smurfing : Equivalent du flooding mais sur tout un réseau
  - Hijacking : Détournement d'un connexion
  - Sniffing : Ecoute des communications en vu d'obtenir des informations
  - Replay : Le rejeu
  - Denial Of Service : Déni de service

Vocabulaire

- Attaque logicielle
  - Buffer Overflow : Attaque par débordement
  - Service poisoning : Corruption d'un service (détournement de son comportement)
- Logiciel de contrôle ou de nuisance
  - Backdoor : Logiciel de prise de contrôle à distance
  - Keystroke Monitoring : Logiciel de surveillance de la machine
  - Trapdoor : Logiciel permettant d'obtenir des droits privilégiés suite à une action particulière

Les problèmes de sécurité dans la pratique

- Internet n'a pas été conçu avec un objectif de sécurité
  - Internet est basé sur la confiance
  - Chaque niveau traversé par vos données offre des moyens d'attaques
- Internet est né avec les unix
  - il n'y a pas un UNIX mais une multitude d'implémentations différentes qui présentent toutes des caractéristiques propres
- Il existe de nombreux problèmes de sécurité dans la plus part des systèmes informatiques actuels.
- Au niveau physique et liaison (ethernet)
  - Sniffers qui écoute le réseau

Les problèmes de sécurité dans la pratique

Introduction

- Au niveau réseau (IP)
  - IP Spoofing et Smurfing
- Au niveau transport (TCP)
  - SYN Flooding
  - Au niveau applicatif (service réseau)
  - Déni de services (Deny of Services)
  - Buffer Overflows
- Attaque au niveau des services
- Attaque au niveau des personnes

Rappels

Introduction

- Un intrusion peut durer moins de quelques seconde ou s'étendre sur plusieurs semaines
- Elle est faciliter par l'existence d'outils pré-programmer (script-kiddies)
- Une bonne protection passe par le triumvirat : Parefeu, Antivirus, Antiespion
- Permet l'accès aux informations ou servir de base d'attaque
  - Une série d'intermédiaire complexifie de façon exponentielle la localisation de la source
  - Une attaque extérieure peut être impossible à déposer devant les tribunaux
- Un très bon site: www.ouah.org
- On rentre dans les détails gores (« gory-details »)
  - Ne faites pas ces expériences sur le réseau de votre FAC !!!
  - C'est détectable, dangereux et juridiquement PUNISSABLE !!!
  - Même si l'administration est gentille, plus de compte !!!
    - c'est dure de travailler sans compte informatique

Plan de cours

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

Attaques de niveau 3: IP/ICMP

IPSEC - VPN

Attaques de niveau 4: TCP

SSL/TLS – Parefeu - NAT

Analyse

Ecoute des supports physiques

- Ethernet physique
  - Plug sur le câble
    - Introduction de bruit
    - Ecoute du support physique
  - Emissions électromagnétiques des câbles
- Moyens de lutte
  - Blindage des câbles, cage de faraday
  - Filtrage électrique
  - Ne pas connecter les machines vers l'extérieur
  - Contrôle des câbles par du matériel spécifique

Réseau sans fils 802.11

- Le 802.11 aka WiFi (Wa fa)
  - C'est un standard de réseau sans fil
  - Il existe une pléthore de protocoles
  - On parle de l'alphabet 802.11 (802.11a, 802.11b, ...)
  - Les seuls a retenir sont 802.11g, 802.11i, 802.11x
- Le 802.11 soulève des problèmes
  - Qui existent déjà !!!! ➔ Mise en exergue des problèmes filaires !
  - Périmètre de sécurité
    - Portables d'inconnus entre dans le périmètre
    - Equivalent à une prise de la taille d'une sphère de 80m de rayon
- Sans cryptage (confidentialité) et sans authentification
  ➔ SUICIDAIRE !!!
- Ecoute et brouillage possible

802.11: Moyens de lutte

- Contrôle d'accès (efficacité limitée)
  - Spatial: mesures et calibrage des puissances du signal des bornes
  - Par adresse: Contrôle d'accès des adresses MAC
  - Pas de diffusion du SSID !
- Confiner ce réseau dans un réseau spécial externe
  - Eviter les accès IP sur le réseau interne
- Confidentialité: Le WEP une coquille vide ➔ A JETER
  - Faiblesse du chiffrement, pas de gestion des clefs
  - airsnort.shmoo.com, www.cr0.net:8040/code/network/aircrack (dans beaucoup de distrib linux)
- Confidentialité et authentification ➔ la seule solution valable, ENCORE EN DEPLOIMENT !
  - **802.11i WPA2 (et PAS WPA simple ou de WEP) ➔ confidentialité**
  - **802.11x avec un serveur radius (www.freeradius.org) et des modules EAP**
- Audit
  - Journalisation des adresses inconnues (MAC et IP)
  - Journalisation des scan
  - Détection des réseaux pirates internes et externes
  - Recherche de signal en bordure (istumbler.net et autres) et triangularisation
- Saturation hertzienne de la zone couverte

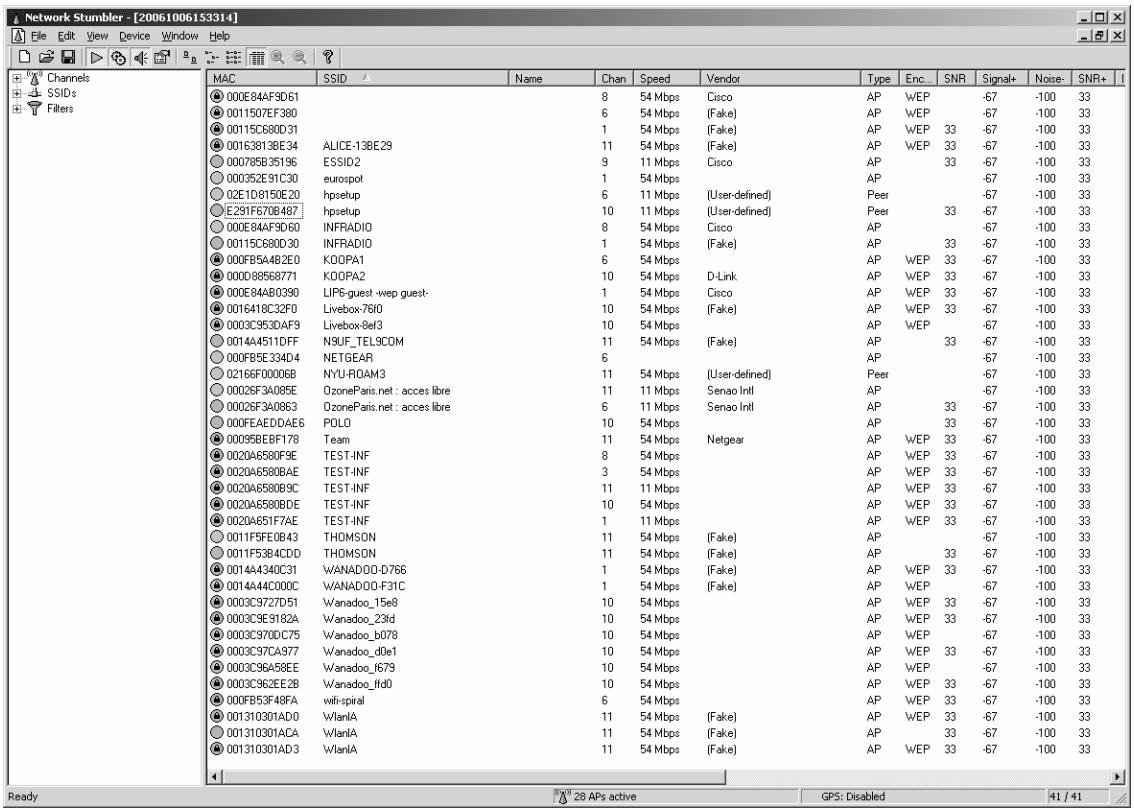
Exemple netstumbler

NETSTUMBLER

Exemple netstumbler

- Permet de connaître les PA de la zone
- Permet de connaître les réseaux non sécurisés

Les AP WiFi autour de mon bureau



Plan de cours

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

Attaques de niveau 3: IP/ICMP

IPSEC - VPN

Attaques de niveau 4: TCP

SSL/TLS – Parefeu - NAT

Analyse

Rappels : Ecoute réseau (sniffing)

- But:
  - Collecte d'informations sur les données circulant sur le réseau
  - Analyse à posteriori des trames
  - Attaque de décryptage sur les données (analyse différentielle)
  - Captures des mots de passe en clair (POP3, TELNET, IMAP, ...)
- Fonctionne avec tous les protocoles de niveau supérieur
  - La capture s'effectue au niveau 2 avec du matériel classique
  - La capture s'effectue au niveau 1 avec du matériel spécialisé

Ecoute ethernet: fonctionnement

- Ethernet c'est 90% des réseaux locaux
  - Prix ridicules, déploiement aisé, grande variété de matériel
  - Ethernet est un support DIFFUSANT !!!! ➔ facilité d'écoute !!!!!
- Il faut passer la carte en mode "Promiscuous"
  - Permet à la carte de capturer tout ou partie des paquets qui transitent sur le réseau local
    - Même si les paquets sont non destiné à l'adresse IP de la machine qui écoute
    - Ne permet pas la capture hors du réseau local
  - Peut être filtrer par les routeurs
- Accès simple
  - Librairie PCAP
  - Raw socket: packet_socket = socket(PF_PACKET, int socket_type, int protocol);
- Utilisation de logiciels d'écoute
  - Très facile à utiliser
  - TCPDUMP (Linux/Windows), ethereal/PCAP (Windows/Linux)
  - Network Associates Sniffer (Windows)

Détection locale des écoutes

- Détection possible des cartes en mode "promiscous"
- Vérification locale
  - Un rootkit peut cacher des informations
  - Un rootkit peut en cacher un autre
  - Ne pas utiliser les programmes de la machine !
  - Télécharger ses propres programmes compilés !!!
  - Ifconfig:

```
pollux 14:46 >ifconfig eth0
  eth0  Link encap:Ethernet  HWaddr 00:C0:4F:24:27:E7
        inet addr:132.227.64.49  Bcast:132.227.64.255
  Mask:255.255.255.0
  UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
   RX packets:9566866 errors:44 dropped:0 overruns:0 frame:4
   TX packets:7763589 errors:0 dropped:0 overruns:0 carrier:0
   collisions:0 txqueuelen:100
  Interrupt:14 Base address:0xcc00
```

  - Un processus root inconnu est en cours d'exécution (ps)
  - Vérifier les comportements des programmes avec lsof et strace

Détection distante des écoutes (antisniff)

- En théorie c'est impossible
  - les nœuds sont passifs ➔ ils ne transmettent rien
- Dans la pratique, c'est parfois possible
  - Difficile à détecter !!
- Méthode de ping 1
  - Si la machine qui a pour @IP 132.227.64.234 et pour @MAC aa:bb:cc:dd:ee:ff est suspectée d'écouter le réseau.
  - On émet une demande ICMP "echo request" en modifiant l'adresse MAC (ie: aa:bb:cc:dd:ee:f0)
  - Si la machine répond, elle était en mode d'écoute
    - Le mode « promiscuous désactive » le filtre de l'@ MAC et répondra à l'@IP sans vérifier l'adresse MAC

Antisniff: autres méthodes

- Méthode du ping 2 : Variation sur la première méthode
  - On ne génère plus un paquet ICMP encapsulé dans ethernet
  - On utilise une ouverture de connexion TCP
  - On utilise le service "echo" (non ICMP)
  - On génère un paquet IP provoquant un message ICMP
- Méthode ARP
  - On attend quelques minutes sans émettre (vider les caches)
  - J'envoie une requête arp unicast ("non broadcast") à une autre machine en fournissant mon @IP et mon @MAC
  - Ensuite je demande un "echo request" la machine suspectée
  - Si elle répond sans émettre de requête ARP, la machine écoutait le réseau

Antisniff: autres méthodes

- Méthode du DNS
  - Les machines qui écoutent peuvent faire des requêtes DNS
  - Faire un ping et vérifier les demande DNS arrivant de machine non connues (@IP mauvaise) !!
- Méthode de routage-source
  - Envoyer un paquet à une machine intermédiaire en demandant son acheminement à la machine suspectée
  - Si la machine intermédiaire ne fait pas suivre le paquet et si la machine suspectée répond, elle écoute le réseau

Antisniff: autres méthodes

- Méthode de leurre
  - On génère du traffic POP, TELNET, FTP ... avec des comptes fictifs (sans réel droits)
  - On vérifie si des login sont effectués sur ces comptes
- Méthode de la latence
  - On génère un traffic ethernet important
    - Il sera filtré par les machines normales
    - Il sera capturé par les machines en écoute
  - On ping les machines et on mesure leur temps de réponse
- Outils: antisniff, CPM (check promiscusous method), neped

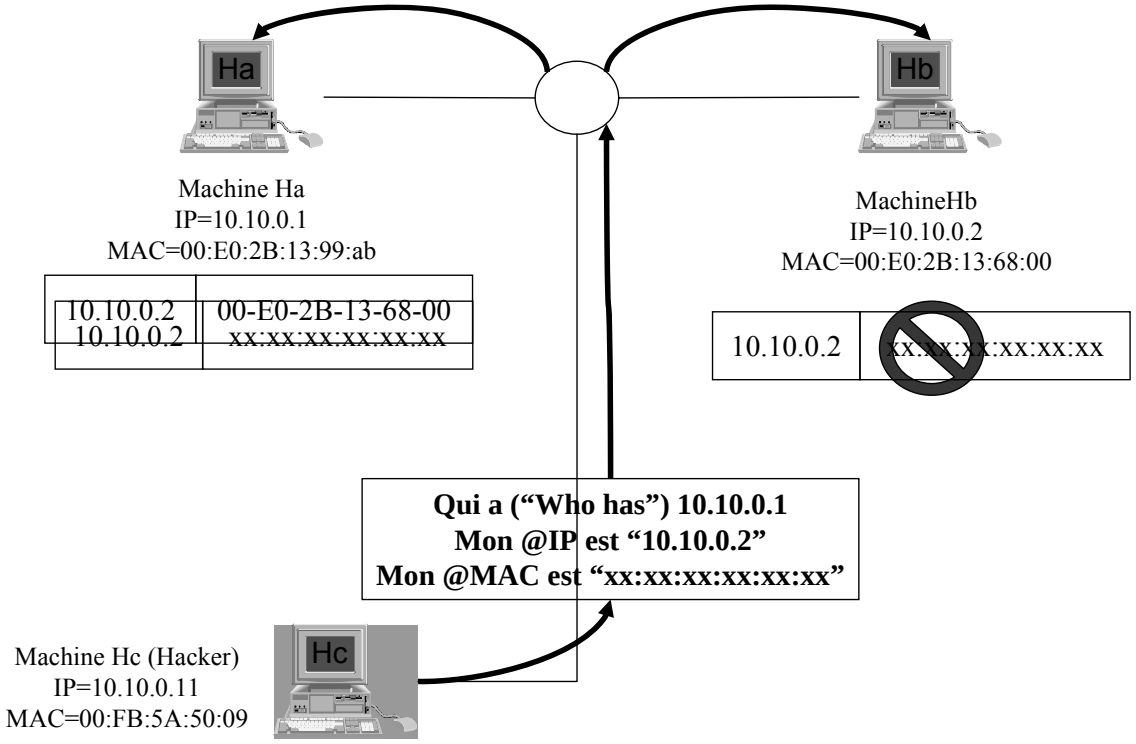
Lutter contre les écoutes

- Eviter la capture de mot de passe
  - Eviter l'authentification et les données en clair
    - Utiliser le cryptage sur les couches basses (SSL,IPSEC)
    - Utiliser l'encpasulation applicative tunneling SSH, stunnel
  - Utiliser des mots de passe à utilisation unique (OTP)
- Recherche systématique des machines inconnues
  - Découverte du réseau (HP openview, netdisco, ...)
- Limiter la connectivité des machines aux machines connues
  - ACL avec adresses ethernet au niveau des switchs
  - VMPS sur les switch (correspondance @MAC/vlan)

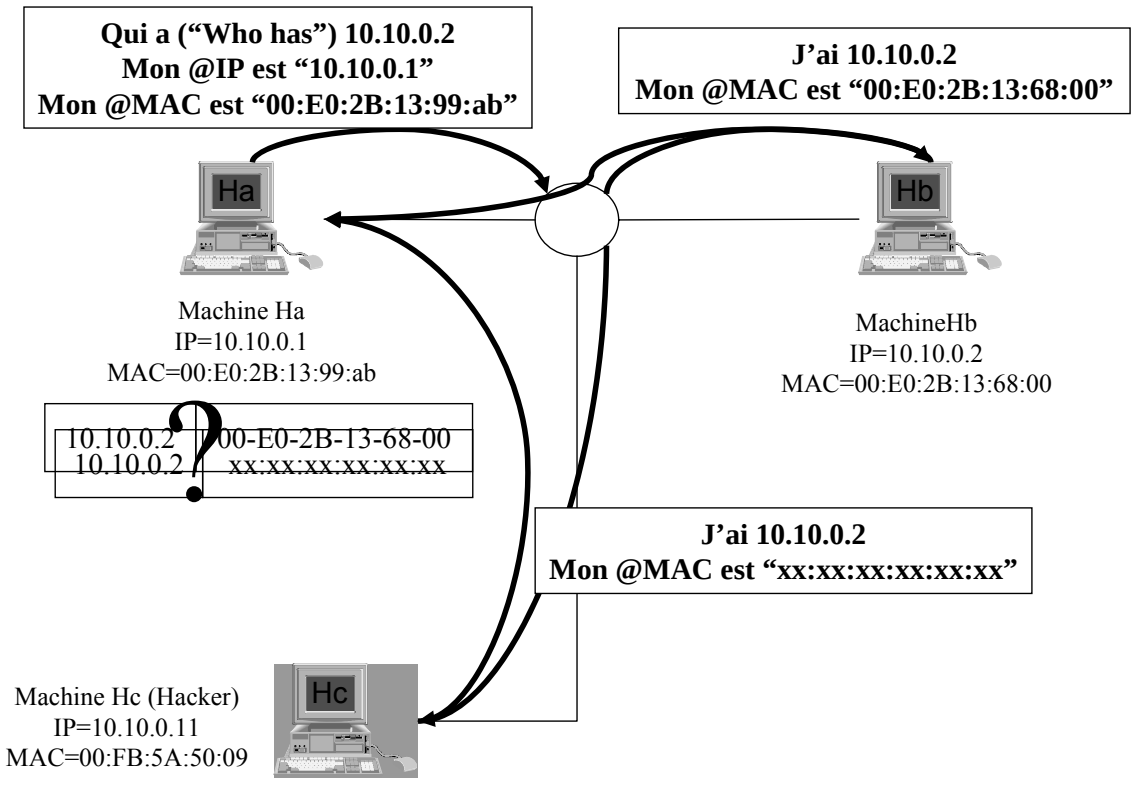
Attaques ARP

- ARP-RARP ➔ lien entre @MAC et @ IP
- ARP maintient un cache des machines qui diffusent sur le réseau
- But des attaques ARP:
  - Détourner le traffic réseau vers sa machine
  - En particulier, remplacer le couple (@MAC/@IP) du routeur par sa propre machine
  - Déni de service
- Moyen:
  - Poisoning : créer de fausses entrées dans les caches ARP
  - Flooding: Saturer les tables ARP
  - Cloning: imiter l'adresse MAC d'une autre machine
  - Spoofing: Forger de fausses réponses ARP
- Utilitaires:
  - http://web.syr.edu/~sabuer/arpoison/
  - http://ettercap.sourceforge.net/ (logiciel d'attaque ARP, SSH, tueur de connexion)
  - http://www.thehackerschoice.com/releases.php (génère des fausses réponses ARP)

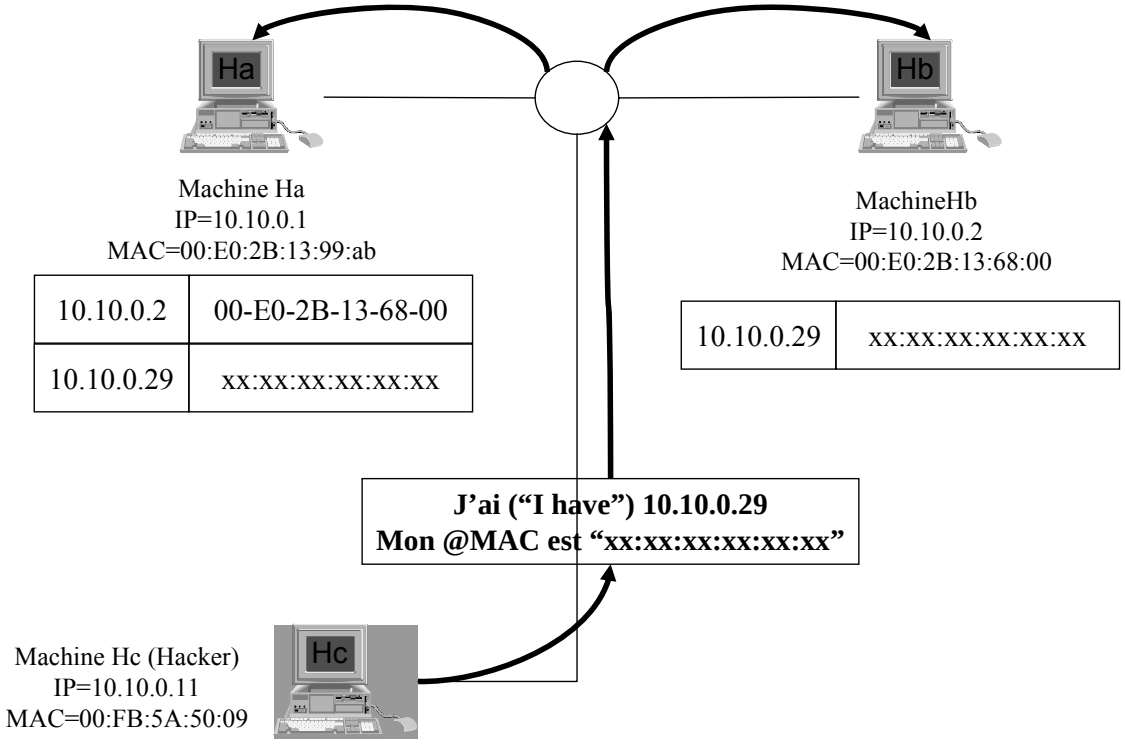
Attaque ARP : Poisoning par diffusion



Attaque ARP : Poisoning par requête



Attaque ARP : Poisoning par diffusion de réponse



Plan de cours

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

Attaques de niveau 3: IP/ICMP

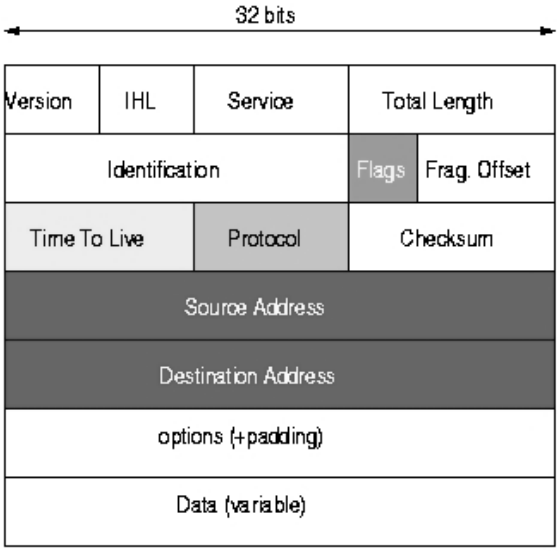
IPSEC - VPN

Attaques de niveau 4: TCP

SSL/TLS – Parefeu - NAT

IDS et Analyse

Le protocole IP



- IP suppose que
  - Les @ IP source et destination sont fiables
  - Le TTL est utilisé pour éviter que des paquets circulent indéfiniment dans le réseau (ghost packet)
  - Protocoles: ICMP=1, IGMP=2, TCP=6, UDP=17, RAW=255
  - Checksum ➔ contrôle CRC
  - Permet des options qui peuvent poser problème
    - Source Routing
    - Bit DF/MF
  - Données: taille max. 64ko

Attaque IP : Spoofing

- En générale, il s'agit d'une attaque aveugle !
- Rappel: Internet est basé sur la confiance.
- Modification du champ de l'@IP source et de certaines options
- But:
  - Usurpation d'identité ➔ on se fait passer pour une autre machine
  - Passer à travers les filtres IPs de certaines machines
- La (véritable) source:
  - peut recevoir les réponses si elle est local (sniffing)
  - NE peut PAS recevoir de réponse si elle est distante
    - Sauf cas exceptionnel (IP source routing)
- A Coupler avec des attaques TCP (cf. prochaine section)

Attaque IP : Spoofing

- Détection: Peu évidente
  - Analyser les logs du parefeu
    - Connexions inhabituelles, violation des ACL, paquets rejetés
- Lutte contre le spoofing IP:
  - Ne pas se limiter à des ACL basés sur les @IP
  - Implanter des règles strictes sur les routeurs
    - Tout paquet provenant de l'extérieure ne peut avoir une @IP source interne
    - Tout paquet provenant de l'extérieure ne peut avoir une @IP source non attribué ou non routable
      - Constitution d'une liste noire des @IP non attribuée à partir des liste de l'IANA
    - Tout paquet allant à l'extérieur ne peut avoir @IP source n'appartenant pas à votre réseau
  - Associer les @MAC avec les @IP pour les machines critiques
    - ie: LES SERVEURS (en particulier ceux d'authentifications) !!
    - PB: les entrées ARP statiques sont parfois mal supportées
  - Sonde de monitoring des couples (@MAC,@IP)
    - Détection des cas de divergences avec la conformité et des trames ARP anormales
    - Alerte + Engagement des contres mesures (détection, analyse, blocage)

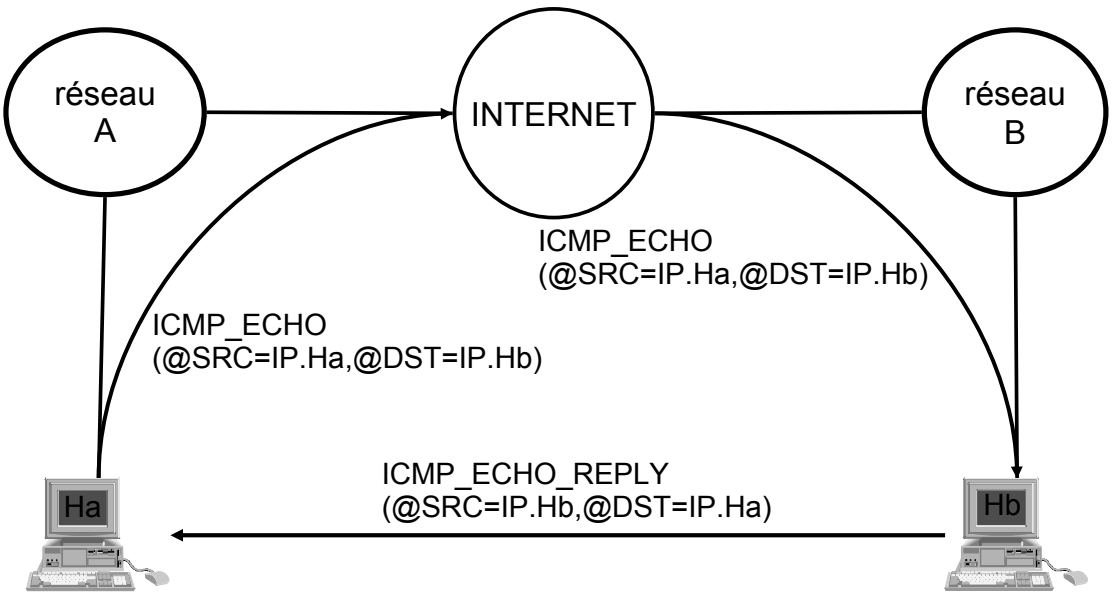
Les protocoles ICMP/IGMP

- Le protocole ICMP (Internet Control Message Protocol)
  - Le protocole ICMP est utilisé par tous les routeurs
  - Il est utilisé pour signaler une erreur sur une machine connectée
- Message ICMP = Type (8 bits), Code (8 bits), Checksum (16 bits), Message (taille variable)
- Couples Type / Message :
  - 0: Echo reply, 8: Echo request
  - 3: Destination Unreachable,
  - 5: redirect, 6: alternate host address, …

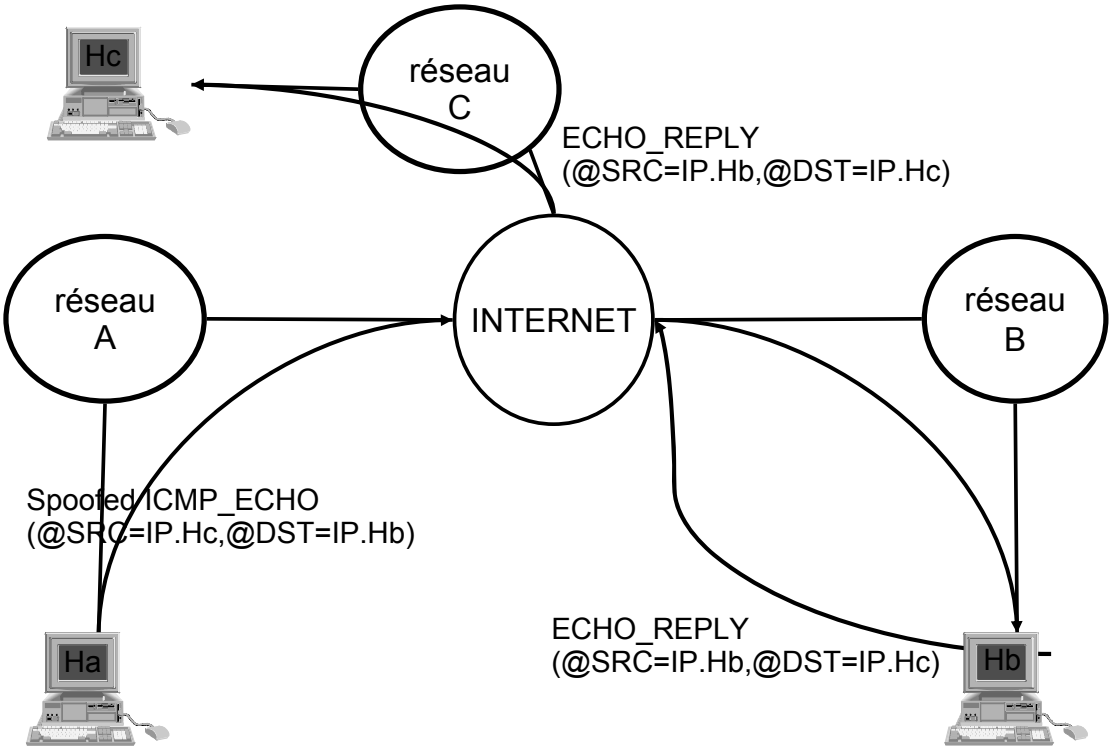
Attaque ICMP: ECHO Flooding

- But du ICMP Flooding
  - Rendre un ou plusieurs services inaccessible
  - Rendre un réseau totalement inaccessible
- Moyen
  - La machine cible passe tout son temps à répondre à des requêtes ICMP
  - Ping Of Death sous windows !!
    - « ping –f cible.reseau.fr 5242880 »
- Conséquences
  - Surcharge machine et du réseau
  - DENI DE SERVICE (D.O.S.)

Attaque ICMP: ECHO Flooding



Attaque ICMP: ECHO Flooding



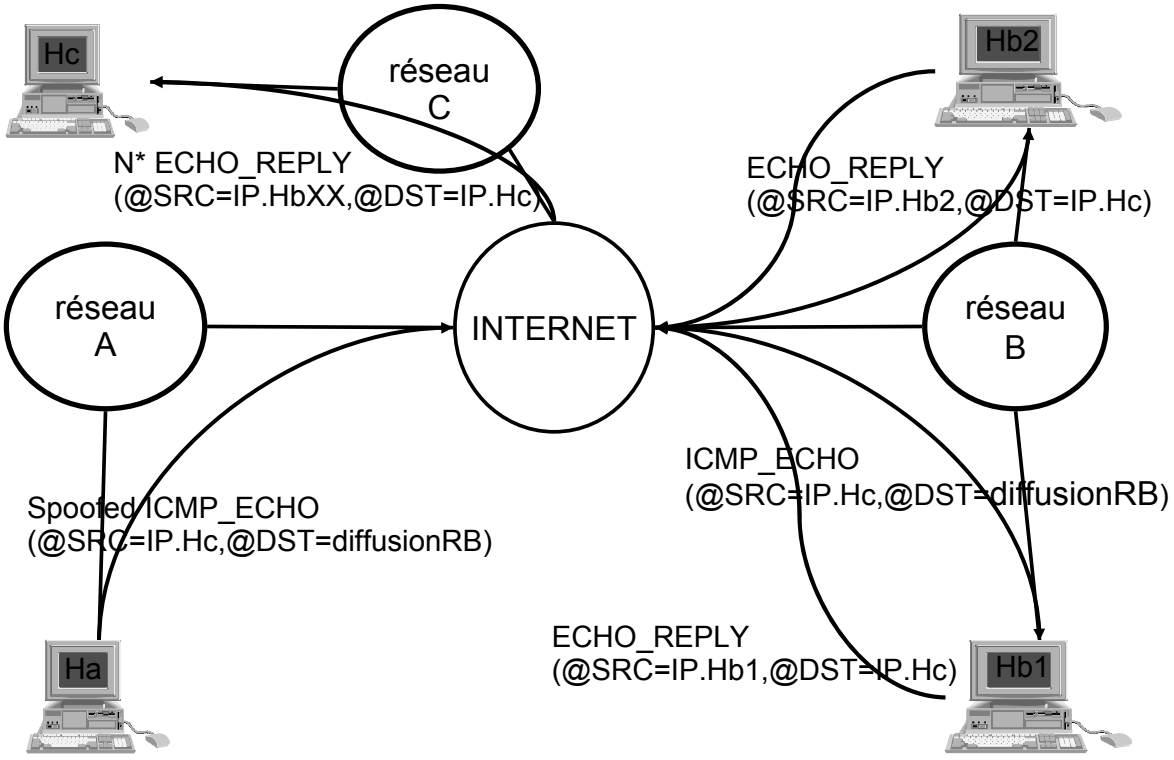
Attaque ICMP: ECHO Flooding

- Moyen de détection:
  - Observer la métrique réseau. En cas de FLOOD
    - Réseau lent sans raison interne à la machine
    - Temps de réponse important voir expiration de délais
    - Les services internes peuvent devenir inaccessibles
  - Examiner les logs des firewalls
    - Haut taux de ICMP_ECHO/REPLY
- Lutte anti FLOOD:
  - Désactiver le forward ICMP_ECHO sur les routeurs
    - Pb: plus de ping du tout.
    - Incompatibilité avec les certains logiciels
  - Utiliser des contrôles de débit
    - Ex: 3 ICMP_ECHO / REPLY par second
    - Liste noire dynamique (interdiction pendant “x” minutes)
  - N’autoriser les ping que sur le sous-réseau
    - Pb en cas d’attaque interne
  - Couplage contrôle de débit avec contrôle ip

Attaque ICMP: ECHO Smurfing

- But:
  - Déni de service sur le réseau ou la machine cible
- Moyen:
  - Utiliser un réseau comme réflecteur
  - Faire une demande de ping avec
    - Pour adresse source l'adresse du réseau (ou de l'hôte) à attaquer
    - Pour adresse cible l'adresse de diffusion du réseau réflecteur
  - Un seul paquet envoyé, toutes les machines du réseau réflecteur répondent (amplification)
  - Les réponses seront envoyées à la cible

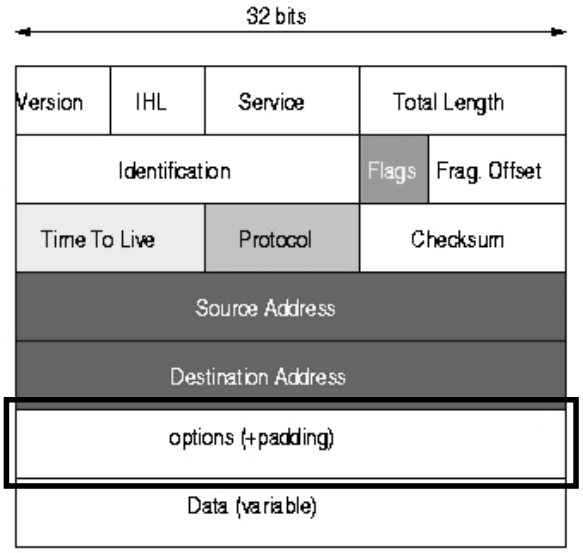
Attaque ICMP: ECHO Smurfing



Attaque ICMP: ECHO Smurfing

- Conséquences
  - Identiques au flooding mais amplifiées
  - La cible subit une charge très importante
- Si votre réseau à une grosse bande passante, il devient idéal comme réflecteur.
- Le réflecteur peut avoir un réseau lent.
- Les deux réseaux peuvent être la cible.

Le protocole IP



- IP suppose que
  - Les @ IP source et destination sont fiables
  - Le TTL est utilisé pour éviter que des paquets circulent indéfiniment dans le réseau (ghost packet)
  - Protocoles: ICMP=1, IGMP=2, TCP=6, UDP=17, RAW=255
  - Checksum ➔ contrôle CRC
  - Permet des options qui peuvent poser problème
    - Source Routing
    - Bit DF/MF
  - Données: taille max. 64ko
- IP: RFC 781, 791, 1349

Attaque IP : Source Routing

Attaques niveau 3: IP/ICMP

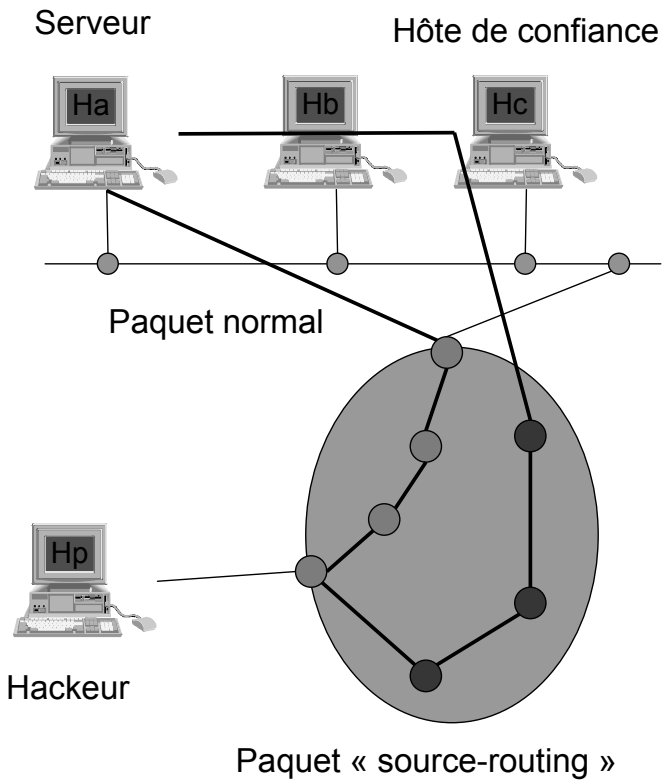
- But:
  - Cacher une connexion TCP désynchronisé
  - Dissimuler d'autres types d'attaques
  - Obtenir une réponse dans le cas d'une attaque IP-Spoofing
- Moyen:
  - L'option "source routing" permet de spécifier un chemin de retour imposé pour la réponse
  - Surpasse ("over ride") la configuration par défaut des routeurs

Attaque IP : Source Routing

Attaques niveau 3: IP/ICMP

- Comment ?
  - L'attaquant fixe l'option de "source routing" dans la paquet IP
  - L'attaquant impose la route de retour à la réponse de la cible. Il s'agit donc d'une suite d'@IP par lequel doit transiter le paquet.
  - La dernière @IP de la route doit être sur le même sous réseau que l'adresse @IP destination
  - La première machine hôte de retour doit être de confiance par rapport à la machine cible

Attaque IP : Source Routing



- Le serveur
  - Accepte les connexions
  - Accepte le source-routing sur des machines de confiance
- La machine de confiance
  - Fait suivre le paquet
  - Doit accepter le source-routing ou être corrompue.
- L'attaquant
  - Reçoit la réponse malgré l'attaque IP spoofing

Attaque IP : Source Routing

- Solution: c'est simple !!!!
  - Désactiver la fonction de "source-routing" sur les paquets IP
    - Sur les toutes les machines, sur tous les routeurs
    - Option devenue quasi-inutile, désuète et dangereuse
    - Malheureusement, ce n'est pas toujours le réglage par défaut
  - Tous les paquets vont être détruits par les machines
  - Abandonner la notion de machine de confiance
    - Lorsqu'elle est uniquement basée sur l'@IP
    - Abandonner les ~/.rhost, ~/.shost
    - Il faut de l'authentification forte

Plan de cours

IPSEC - VPN

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

Attaques de niveau 3: IP/ICMP

IPSEC - VPN

Attaques de niveau 4: TCP

SSL/TLS – Parefeu - NAT

IDS et Analyse

IPSEC - VPN

IPSEC - VPN

- Permet d'éviter le spoofing @IP et le sniffing
- IP-Secure / Développé par l'IETF (RFC 2401, 2402, 2406, 2409, 2411)
  - Utilisé pour implanter les VPN (Virtual Private Network)
  - Solution niveau 3 (réseau) ➔ IP
  - Pas une solution de niveau "applicatif" comme SSH
- IPSEC fournit :
  - La communication est crypté de bout en bout
  - L'authentification forte, confidentialité et intégrité
  - Indépendant de TCP/UDP, repose sur IP
- Il est supporté :
  - en natif par beaucoup de système et par la grande majorité des routeurs
  - par des drivers/applications sur beaucoup d'autres systèmes
- Mode IP-Sec: 2 mode de transport
  - Payload ➔ seulement les données sont cryptées 'encapsulation des données)
  - Tunnel ➔ toute la communication est encryptée (encpasulation totale)
- Mode IP-Sec: 2 protocoles
  - AH ➔ qui ne permet que l'authentification forte et intégrité, pas de confidentialité
  - ESP ➔ qui permet la confidentialité

IPSEC-VPN: Modes

IPSEC - VPN

- Le mode “Payload”/Transport
  - Facile à utiliser
  - Peut être utiliser sur tous les réseaux et routeurs
  - Les données ne peuvent être décryptées (Encryptions fortes)
- Le mode “Tunnel”
  - Plus puissant que le mode “Payload”
  - Requière le support de tous les routeurs sur le chemin
    - Nécessité des ressources CPU sur les routeurs
    - Nécessité pour les routeurs de pouvoir décrypter les entêtes

IPSEC-VPN: Protocoles

IPSEC - VPN

- Le protocole AH (Authentification Header)
  - Fournit l’authentification de provenance des données
  - L’intégrité des données (sans mode connecté)
  - Un protection (optionnelle) contre le rejeu
- Le protocole ESP (Encaspulating Security Payload)
  - Fournit la confidentialité du trafic
- AH et ESP peuvent servir de support pour faire de l’authentification
  - Mieux vaut utiliser le mode tunnel en ESP avec authentification externe
- Le protocole SA (Security Association)
  - Décrit les associations de sécurités possible (entre AH et ESP)
  - Le mode utilisé (tunnel ou payload)
  - Les algorithmes de cryptage
  - Les SA ne fonctionnent que sur un canal unidirectionnel ➔ 2 SA pour TCP, ...
- Problèmes:
  - Difficultés avec la fragmentation IP
  - Surcoût engendré par le protocole ESP

IPSEC-VPN:

- Trame IP simple:

Orig IP header	data

- AH mode transport et mode tunnel (datagrammes authentifiés sauf champs mutables)

Orig IP header	AH	data

New IP header	AH	Orig IP header	data

- ESP mode transport et mode tunnel (rouge = crypté, gris = authentifié)

Orig IP header	ESP header	data	ESP trailer	ESP auth

New IP header	ESP header	Orig IP header	data	ESP trailer	ESP auth

IPSEC - VPN

IPSEC-VPN: Performances

- L'IPSEC est très couteux mais :
  - De nouvelles implantations existent
  - Les coûts sont en fortes baisses

Mesures de performances IPSEC	
Pas d'IPSEC	315 kb/s
IPSEC avec AH et ESP	47 kb/s
IPSEC avec AH	26 kb/s
IPSEC avec mode transport ESP	26 kb/s
IPSEC avec mode tunnel ESP	26 kb/s
IPSEC ESP-part AH et le mode tunnel ESP	19 kb/s
IPSEC avec le mode transport ESP et AH	20 kb/s
IPSEC avec le mode tunnel ESP et AH	18 kb/s

IPSEC - VPN

Plan de cours

Introduction

Attaques de niveau 1

Attaques niveau 2: ethernet

Attaques niveau 3: IP/ICMP

IPSEC - VPN

Attaques niveau 4: TCP

SSL/TLS – Parefeu - NAT

IDS et Analyse

Le protocole TCP

16 Bits Source Port					16 Bits Destination Port			
32 Bits Sequence Number								
32 Bits Acknoledgement Number								
Head Length	6 bits Res	urg	ack	psh	rst	syn	fin	16 Bits Window Size
16 Bits TCP Checksum					16 Bits Urgent Pointer			
Options if any								Padding
Optional Data Area								

- Le protocole TCP est un mode connecté best effort
- Un paquet TCP
  - Port Source/Destination (@service)
  - « Sequence number » et « Ack. Number » ➔ gestion de l'ordre des paquets
  - Acknowledgment field ACK ➔ trame d'acquittement
  - Connection opening (SYN) ➔ trame d'ouverture de connexion (synchronisation)
  - Connection end (FIN) ➔ « graceful logoff »
  - Connection Reset (RST) ➔
  - « forced/brutal logoff »
  - Checksum ➔ CRC
  - Options
  - Puis viennent les données
- TCP: RFC 1180, 768(UDP), 793 (TCP), 3168 (TCP), 792(ICMP), 950(ICMP)

Le procotole TCP – Ouverture de connexion

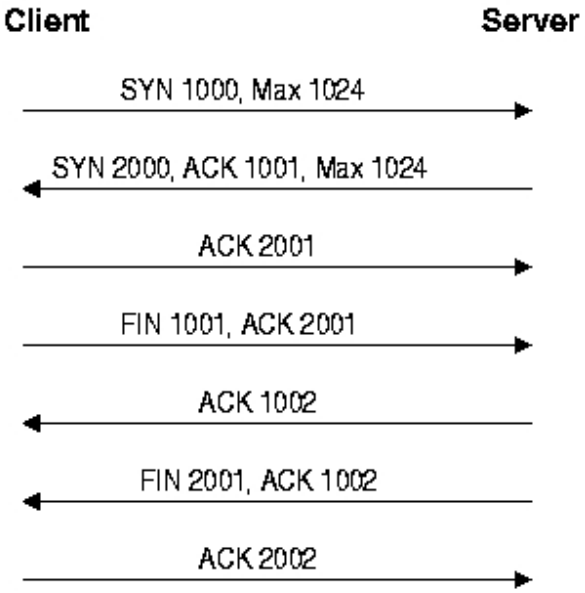
- Négociation en 3 phases (3 ways handshake)
- Le client envoie un premier paquet au serveur
  - Ce paquet contient le drapeau SYN mis à 1
  - Le client envoie un numéro de séquence de départ
  - La taille de sa fenêtre de réception
- Le serveur répond avec une trame d'acquittement d'ouverture
  - Le drapeau ACK (acquittement) est mis à 1
  - Le drapeau SYN (synchronisation) est mis à 1
  - Le serveur met un numéro d'acquittement égale au numéro du client +1
  - Le serveur met un numéro de séquence
  - La taille de sa fenêtre de réception
- Sur réception de cet acquittement, le client répond par une trame dont
  - Le drapeau ACK (acquittement) est mis à 1
  - Le numéro d'acquittement est égale au numéro de séquence du serveur +1

Le protocole TCP – Fermeture de connexion

- Fermeture brutale
  - Une des parties envoie une trame avec
    - Le drapeau RST mis à 1
    - Le numéro de séquence d'acquittement égale au dernier numéro reçut +1 (souvent ignoré)
- Fermeture élégante ou la déconnexion concertée (« graceful logoff »)
  - La partie qui souhaite se déconnecté envoie une trame avec:
    - Le drapeau FIN mis à 1
    - Le numéro d'acquittement inchangé
    - Le numéro de séquence incrémenté de 1
    - Attente de la réponse
  - La partie qui reçoit la demande de déconnexion, renvoie une trame de confirmation
    - Le drapeau ACK et FIN sont mis à 1
    - Le numéro d'acquittement est incrémenté de 1
    - Attente de la confirmation ou de timeout
  - La partie qui souhaite se déconnecté envoie une trame avec:
    - Le drapeau ACK mis à 1
  - Fermeture de la connexion par l'initiateur de la déconnexion
  - Fermeture de la connexion par le receveur de la demande

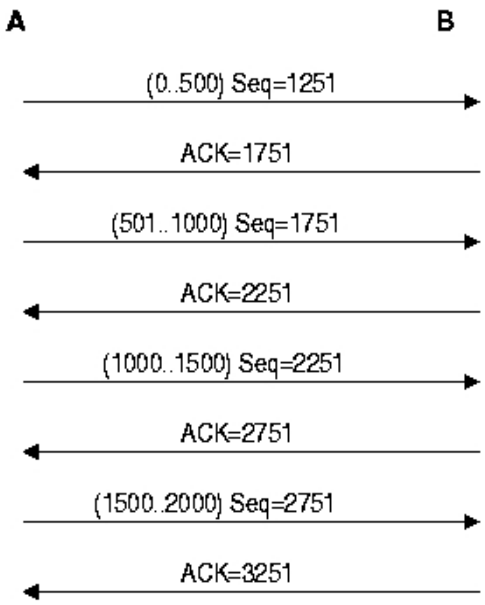
TCP - Example

- Ouverture et Fermeture de connexion



TCP – Sequence Numbers

- Le numéro de séquence correspond aux nombres d'octets envoyés depuis l'ouverture de la connexion.



Attaque TCP: SYN Flooding

Attaques niveau 4: TCP

- But :
  - Attaque de déni de service
  - Empêcher la machine d'accepter de nouvelles connexions
- Principe :
  - Créer des connexions semi-ouvertes
  - La machine cible doit attendre la fin du processus de connexion
  - Elle maintient ses ressources à disposition (non libération)
  - On sature la liste d'attente (waiting-list)
- Conséquences :
  - Tous les services TCP deviennent indisponibles
- Caractéristiques :
  - Les paquets TCP SYN ne sont pas forcement tracés
  - Le réseau n'est pas saturé de trames ICMP facilement identifiables comme dans les attaques ECHO flooding ou ECHO smurfing

Attaque TCP: SYN Flooding

Attaques niveau 4: TCP

- La machine A envoie une demande de connexion TCP à la machine B
  - paquet SYN
  - La trame TCP est encapsulé dans un paquet IP avec une fausse adresse (par ex. C)
- La machine B répond à la demande
  - Paquet SYN/ACK
  - Attend une confirmation d'ouverture qui n'arrivera jamais
- La connexion reste semi-ouverte pour un temps limité
  - dépendent du système
  - très grand devant la capacité des réseaux
- La machine A recommence avec
  - un grand nombre de demandes fabriquées
  - de fausses adresses (sources variées)

Attaque TCP: SYN Flooding

- Les solutions ne sont pas toujours à la portée des admins !
- Implantation d'une pile TCP/IP adaptée
  - Limitation de l'impact de l'attaque
  - Attendre le ACK final avant d'allouer toute ressource
    - Utiliser en premier lieu dans le noyau linux 2.x.x
- Protection implantée dans les routeurs
  - Le routeur ne transmet pas la trame SYN à la machine B
  - Il attend lui-même la réponse (SYN/ACK) de A
  - ouvre ensuite la connexion avec B et lui fait suivre le paquet SYN/ACK
  - La table d'attente des connexions est gérée par le routeur
  - Il utilise un algorithme LRU (Least Recent Used) pour vider les connexions en attente d'ouverture
  - Mode dégradé

Attaque TCP: Connection Reset

- But:
  - Forcer la déconnexion entre deux machines
    - Efficace contre les protocoles à connexions de longues durées
    - Sauvegarde, Transferts FTP, Border Gateway Protocol (RFC 1771)
- Supposition :
  - La connexion est déjà ouverte
- Moyen:
  - Utilisation du TCP spoofing et l'IP spoofing
  - Forger une fausse trame TCP avec le flag RST et une bonne valeur de numéro de séquence
    - Le numéro de séquence de la trame RST doit être dans la fenêtre de réception

Attaque TCP: Connection Reset

- Méthode 1
  - La fenêtre de réception fait environ 16Ko
  - Le nombre de numéro de séquence est sur 32 bits
  - Pour trouver une valeur valide, il faut essayer
    - 2^32/16384 ~ 260000 possibilités
    - En cas de sniffing, il suffit d'anticiper la valeur !!!
    - En aveugle sur une ligne de 100ko/s, paquets de 53 octets
      ➔ ~1800 essais / secondes
  - Si la fenêtre est large (satellite, grande capacité)
    ➔ encore moins de trames à générer
- Méthode 2 :
  - Certaines piles TCP/IP interprètent l'arrivé d'une deuxième trame SYN (ouverture de connexion TCP) comme une demande de réouverture suite à un crash.
    ➔ fermeture de la connexion
  - On applique la même méthode que précédemment

Attaque TCP : low-rate

- But:
  - Faire baisser la capacité de trafic sur les serveurs
- Moyen:
  - Engendrer des pertes de trames sur les routeurs
  - Engendrer des pertes de trames sur les serveurs
  - Faire baisser la taille des fenêtres d'émission/réception

Attaque TCP : [remote] land-attack

- But:
  - Déni de service
- Moyen:
  - Envoyer une trame ICMP echo request malformée
  - ou envoyer une trame TCP SYN malformée
  - Peut être effectué en interne ou en externe
  - Les adresses IP sources et destinations appartenant à la même machine
  - Les ports sources et destinations identiques
  - Exemple créer avec le programme HPing2
    @IP externe de la victime: 63.24.122.59
    @IP interne du routeur cible: 192.168.1.1
    hping2 -A -S -P -U 63.24.122.59 -s 80 -p 80 -a 192.168.1.1
- Testé sur des routeurs linksys WRT54GS, Cisco catalyst, …
  - Date: décembre 2005

Attaque TCP : Desynchronized Connection

- But:
  - Perturber la connexion entre deux machines
  - Empêcher deux machines de discuter
  - Etre capable de s'insérer entre les deux machines
    "THE MAN IN THE MIDDLE"
- Principe:
  - Créer une situation où les deux machines sont incapables de communiquer
  - Faire en sorte que les machines ne soient plus d'accord sur les numéros de séquence
- Conséquence:
  - Il est possible de lire/modifier les données échangées entre les deux hôtes
    - L'attaquant (man in the middle) pourra recevoir les paquets, les modifier et les faire suivre
    - Très dangereux pour les applications E-Business et pour le vol de numéro de cartes
  - Très utilisé pour les connexions en clair (telnet/rlogin)

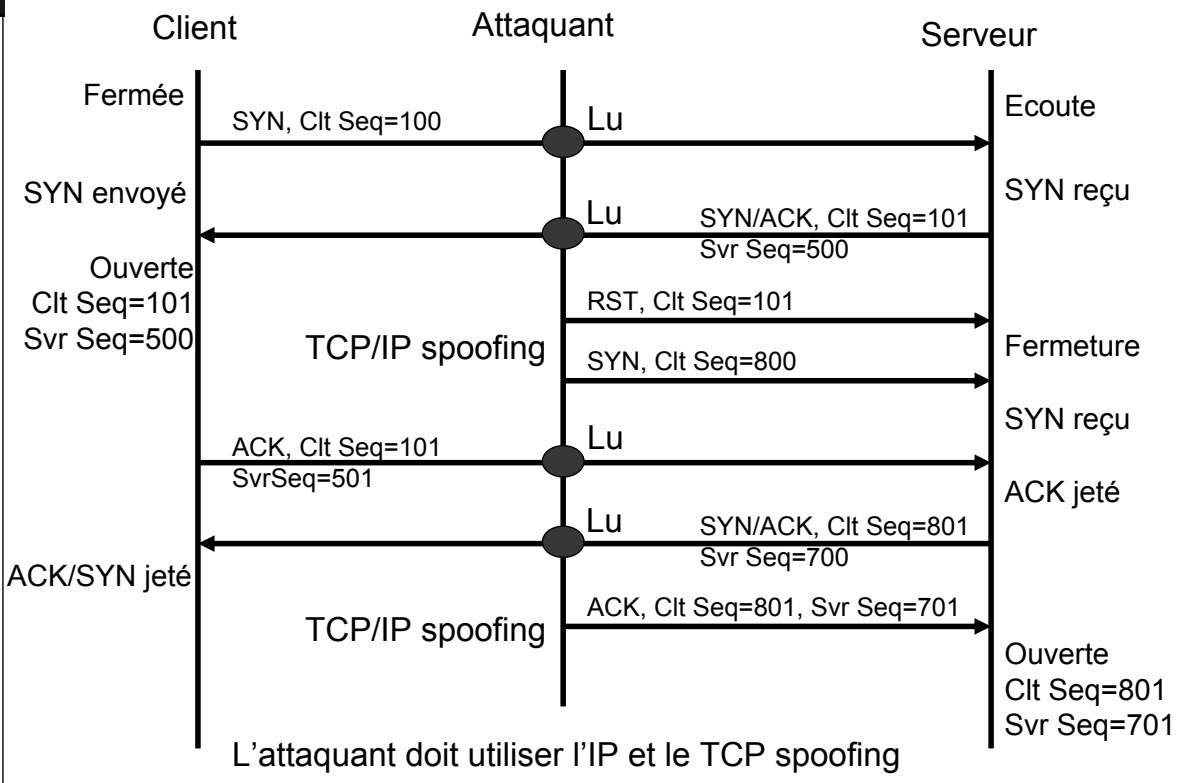
Attaque TCP : Desynchronized Connection

Attaques niveau 4: TCP

- Désynchronisation à la négociation de la connexion (early desynchronization)
- L'attaquant surveille le réseau et attend un paquet SYN/ACK de la part du serveur durant l'ouverture de la connexion
- Le vrai client accepte le paquet ACK/SYN et passe en mode connecté
- Pendant ce temps, l'attaquant annule la connexion du serveur
  - cf l'attaque TCP "connection reset"
  - envoie un trame RST (reset)
    - Le paquet a les même paramètres que le datagramme IP ACK/SYN
  - envoie une trame SYN (ouverture de connexion)
    - Le numéro de séquence est différent ➔ éviter le chevauchement de numéro de séq.
- Le serveur ferme la connexion et en ouvre une nouvelle
  - génère un nouveau numéro de séquence
  - Envoie un paquet SYN/ACK au vrai client
- L'attaquant intercepte le datagramme IP SYN/ACK et envoie une trame ACK au serveur
- Le client et le serveur sont désynchronisé

Attaque TCP : Desynchronized Connection

Attaques niveau 4: TCP

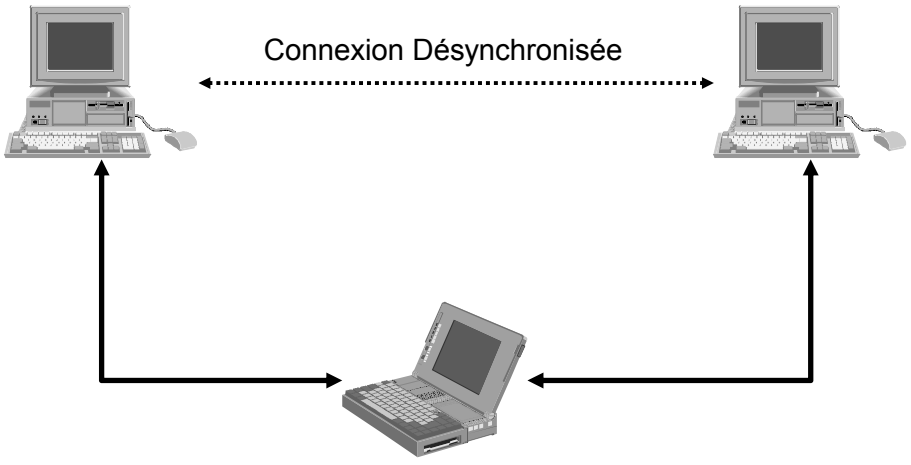


Attaque TCP : Desynchronized Connection

- Désynchronisation par envoie de messages « vides »
- Utile seulement si le protocole accepte les trames « vides »
  - Trame vide = trame sans effet sur l'application
  - Ex: trame IAC NOP sur telnet, trame NOP sur FTP
- L'attaquant surveille l'échange sans interférer
- Au moment opportun, l'attaquant envoie un grand nombre de trame vide
  - Le but est de changer le numéro de séquence du client en injectant le coût des trames vides
    - Si la trame vide fait T octets
    - Le serveur aura sa valeur incrémenté : SVR_ACK ← CLT_SEQ + T
    - Le client sera toujours à CLT_ACQ
- L'attaquant fait la même chose sur le client

Attaque TCP : « Man In The Middle » par désynchronisation

- But: attaque de l'homme au milieu



Attaque TCP : « Man In The Middle » par désynchronisation

Attaques niveau 4: TCP

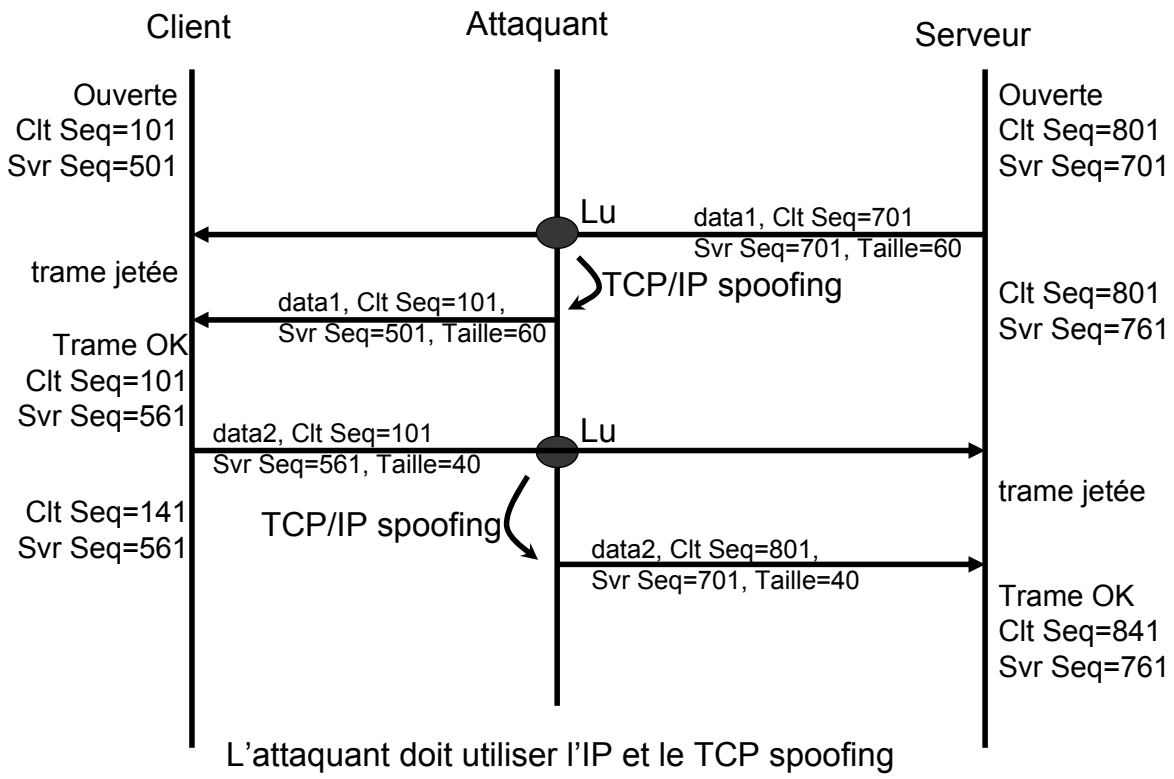
- Nous supposons les deux machines désynchronisées
- Du côté serveur
  - Le serveur n'accepte plus les données du client
  - Les trames du client sont jetées (mauvais numéro de séquence)
  - L'attaquant garde une copie des trames du client (Sniffing)
  - L'attaquant attend la demande de retransmission du paquet envoyé par le serveur au client
  - L'attaquant copie le contenu des paquets du client et les fait suivre au serveur
    - Utilisation de l'IP spoofing (on insère @IP du client comme IP source)
    - On utilise le bon numéro de séquence (celui attendu par le serveur)

Attaque TCP : « Man In The Middle » par désynchronisation

Attaques niveau 4: TCP

- Nous supposons les deux machines désynchronisées
- Du côté client
  - Le client n'accepte plus les données du serveur
  - Les trames du serveur sont jetées (mauvais numéro de séquence)
  - Le client rejettent toutes les réponses du serveur
  - L'attaquant intercepte les données envoyées par le serveur et destinée au client
  - L'attaquant copie le contenu des paquets du serveur et les fait suivre au client
    - Utilisation de l'IP spoofing (on insère @IP du serveur comme IP source)
    - On utilise le bon numéro de séquence (celui attendu par le client)
- Résultat :
  - L'attaquant échange des données avec le client et le serveur
  - Permet éventuellement d'ajouter des données
  - De filtrer les commandes et leurs résultats

Attaque TCP : « Man In The Middle » par désynchronisation



Attaque TCP : « Tempête » de désynchronisation

- Tempête d'acquittement ("The ACK Storm")
- Si les deux clients sont désynchronisés
- Quand un client reçoit une trame TCP invalide
  - Il envoie un ACK à son pair (le serveur)
  - Le numéro d'acquittement est faux !
- Quand le serveur reçoit la trame d'acquittement invalide
  - Il envoie un ACK à son pair (le client)
  - Le numéro d'acquittement est lui aussi faux !
- Le ping pong c'est bien mais qui va perdre ?
  - En théorie cela ne s'arrête pas
- La désynchronisation doit être de courte durée !
  - Il faut couper la connexion au plus tôt

Attaque TCP : « Tempête » de désynchronisation

Attaques niveau 4: TCP

- Après la tempête, le beau temps ! ☺
- IP est un service non sûr
  - Des ACK vont se perdre dans la tempête
  - Certains paquets ne vont pas être retransmis car il ne contiennent aucune données (ACK pur)
  - Plus le réseau est saturé, plus il y a de pertes !
- La tempête va se calmer toute seule
  - Auto-stabilisation de l'effet de tempête
- Génère du trafic non sollicité ➔ Peut rendre l'attaquant détectable !!
- Comment l'attaquant peut-il empêcher la tempête ?
  - L'attaquant peut minimiser l'effet "tempête"
    - Génération d'un ACK correct juste après le rejet d'un paquet par le pair
  - Dans la pratique ➔ problème de synchronisation
  - C'est compliqué. Cela demande aussi un peu de chance.

Attaque TCP : Attaques par désynchronisation

Attaques niveau 4: TCP

- Solutions :
  - Empêcher les usagers inconnus de se connecter au réseau local
    - Elimination des portables !
  - Détection des comportement suspects
    - Taux d'erreur, de perte et de retransmission élevés
  - Détection de la charge réseau
    - Détection des tempêtes d'ACK
  - Utiliser des connexions applicatives sécurisées
    - Ex: SSH
    - Les paquets peuvent être désynchronisés et interceptés mais pas interprétés (confidentialité) ou modifiés (intégrité)

Plan de cours

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

Attaques de niveau 3: IP/ICMP

IPSEC - VPN

Attaques de niveau 4: TCP

Parefeu – NAT - SSL/TLS

IDS et Analyse

Politiques de filtrages

- Le filtrage est un des outils de base de la sécurité. IL EST NECESSAIRE !
- Filtrage optimiste : PERMIT ALL
  - Tout est permis à part quelques services (ports)
  - Facile à installer et à maintenir
    - Seulement quelques règles à gérer
  - Sécurité faible
    - Ne tient pas compte des nouveaux services pouvant être ouvert
    - Ex: un utilisateur ouvre un serveur ftp personnel, …
- Filtrage pessimiste : DENY ALL
  - Rejet systématique
    - Exception : services spécifiques sur machines spécifiques
    - Ex: Autorisations explicites pour les services HTTP, SMTP, POP3, …
  - Plus difficile à installer et à maintenir
    - En particulier pour certains services (ex: FTP)
  - Sécurité forte
    - Les nouveaux services doivent être déclarés
- Prendre en compte les connexions entrantes et les connexions sortantes

Filtrage sur les routeurs

- Installer des règles sur les routeurs pour empêcher certains trafic de passer par les routeurs
  - Utilisation des Access Lists (Cisco, …)
  - Le filtrage peut être fait sur les critères suivants :
    - Par protocoles ([ethernet], IP, ICMP, TCP, UDP, ...)
    - Par adresses (suivant le protocole)
    - Par les numéros de port TCP/UDP (HTTP, SMTP, POP3, ...)
    - Par masque d'adresse
    - Par les interfaces d'accès
    - Structure et/ou contenu des paquets
  - Attention à l'ordre des règles
    - La première qui correspond est celle sélectionnée (Fist Matching, First Applied!)
  - Une politique doit être installée

Filtrage sur les machines clientes

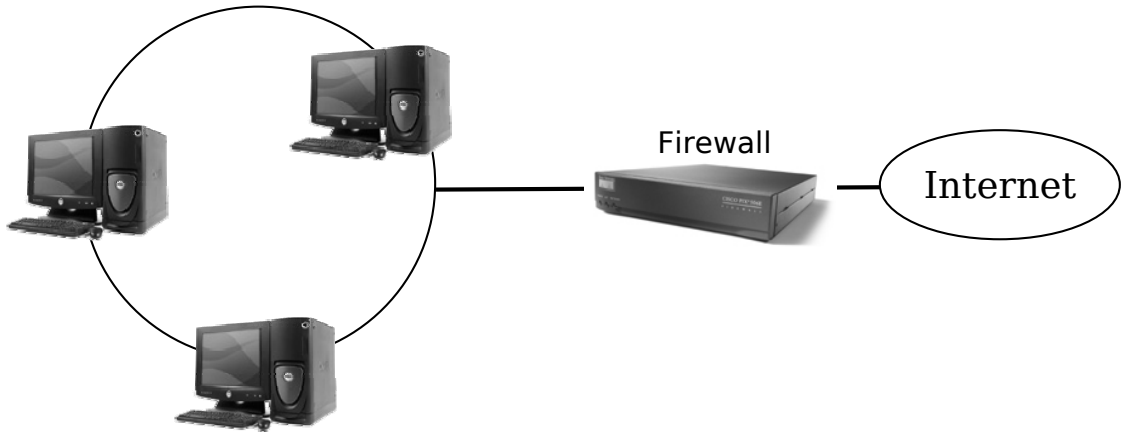
- Le filtrage doit aussi être fait aux niveaux des machines
- Les même type de politiques peuvent être appliquées
  - Optimistes ou Pessimistes
  - Les mêmes critères de filtrages peuvent être appliqués
  - De nouveau critères peuvent être ajoutés
    - Contrôle des utilisateurs et des applications
- Problèmes :
  - Difficultés de maintenance (prévoir un déploiement automatique)
  - Une MAJ doit être déployée
- Doit être adaptables sur les portables (migration)

Firewalls / Routeur

- Différence entre un routeur et un firewall
  - Un firewall ne fait pas de “IP FORWARDING”
  - Un firewall peut faire du routage au niveau applicatif
    - Existence de mandataires HTTP, POP3, etc …
- Les mandataires peuvent être intelligent
  - Filtrage par le contenu (informations)
  - La forme des paquets
- Implantation
  - Un matériel spécialisé (Cisco PIX, ...)
  - Une machine simple avec plusieurs cartes réseaux + logiciels
    - Firewall 1 (Checkpoint), Raptor, Shorewall (Linux), ...

Architecture avec Firewall sans routeur

- Modèle avec double réseau
  - Pas de routage IP
  - Filtrage applicatif seulement

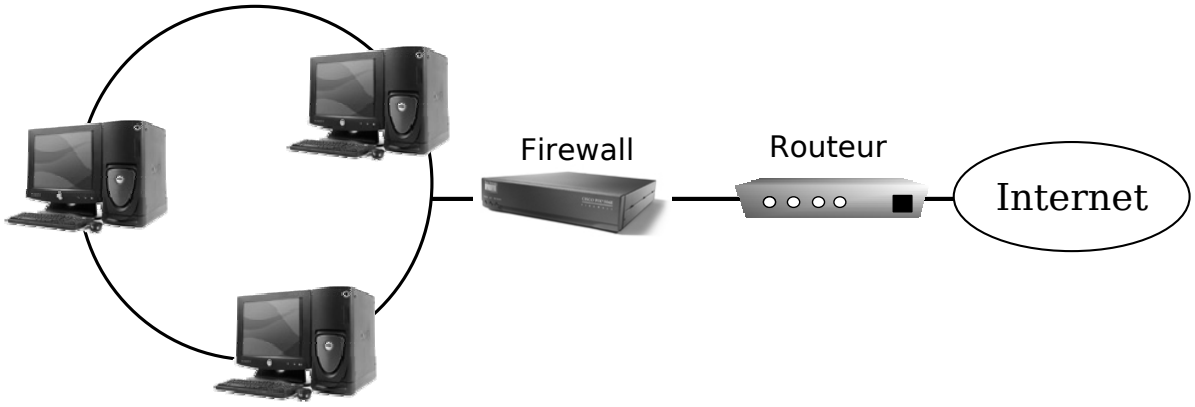


Architecture avec Firewall sans routeur

- On donne des adresses IP *privées* aux machines du réseau
  - Exemple : 10.1.1.1, 10.1.1.2
- Les serveurs ont *aussi* une adresse IP publique
  - Moyen: utilisation d'alias pour les cartes réseau
    - ifconfig eth0 10.1.1.4
    - ifconfig eth0:0 132.227.64.200
- Les clients ne peuvent pas dialoguer directement avec l'extérieur
- Passage par des **mandataires** internes
  - Ok pour certains services : smtp, nntp, web, ftp
- Plus compliqué ou impossible pour d'autres (sessions telnet)

Architecture avec Firewall et routeur

- Modèle avec Firewall et routage
- Le firewall est la seule machine visible à l'extérieur
  - Le firewall effectue le contrôle d'accès
  - Le routeur effectue le routage (translation d'adresse) ➔ NAT

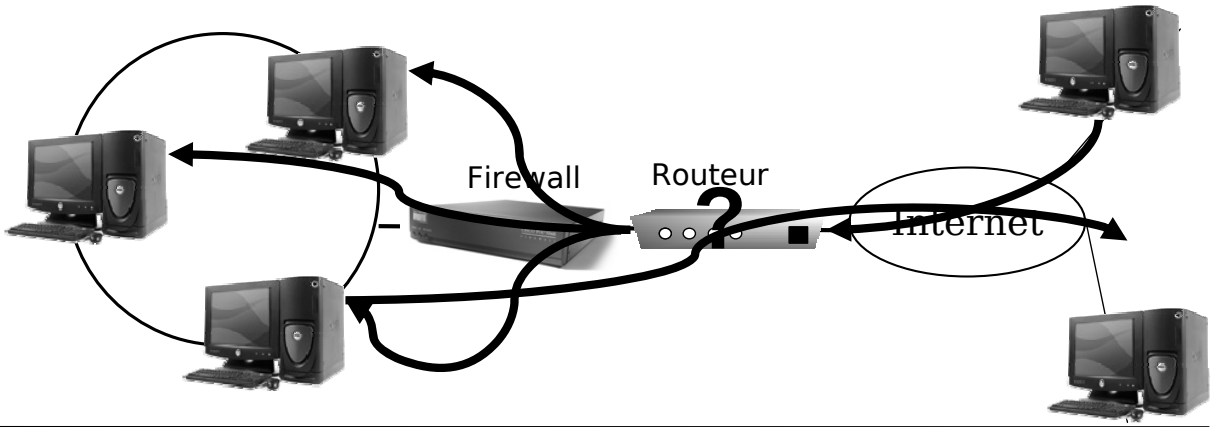


NAT

- Idée : le client fait passer ses communications par le routeur
  - Le routeur « déguise » les paquets pour faire croire qu'il en est l'émetteur
    - Le serveur distant répond au routeur
    - Le routeur fait suivre les réponses au client
  - C'est un exemple de NAT (Network Address Translation)
- Exemple :
  - Le poste 10.1.2.3 démarre une session telnet (TCP, port 23) en direction de 220.6.7.8
  - Le routeur remplace l'adresse d'origine (10.1.2.3) par sa propre adresse, et fait suivre à l'extérieur
  - Le site extérieur répond au routeur
  - Le routeur remplace l'adresse de destination (la sienne) par celle du demandeur 10.1.2.3
  - Le demandeur a obtenu sa réponse !

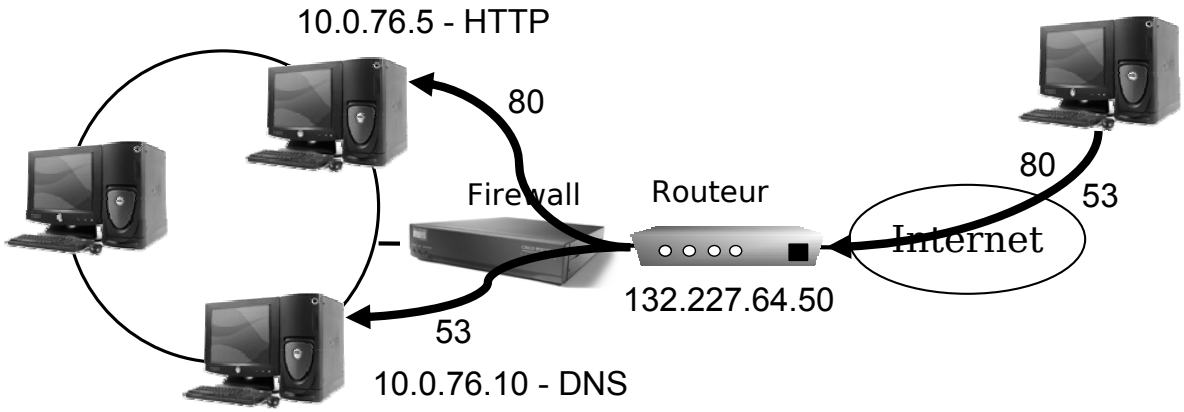
Pourquoi faire du NAT ?

- Protéger ses machines clientes
  - Les connexions sortantes sont possibles
  - Les connexions entrantes sont interdites



Pourquoi faire du NAT ?

- Parfois, l'entreprise n'a qu'un nombre limité d'adresses IP
  - Elle veut déployer plusieurs services
  - Elle vaut faire du load-balancing
  - Le routeur oriente les paquets en fonction d'une politique précise

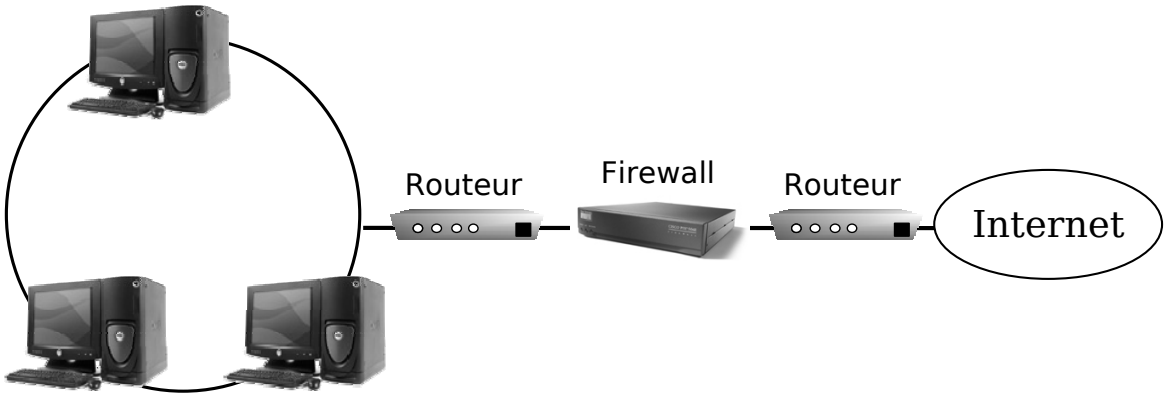


NAT n'est pas la solution ultime !

- Le NAT protège les machines clientes
  - Mais ils sont contournables !
- Méthodes :
  - Ping tunneling: Encapsulation des trames TCP dans les paquets ICMP
  - TCP traversing
  - Synchronous SYN
  - TCP/UDP Hole punching

Architecture avec Firewall et double routeurs

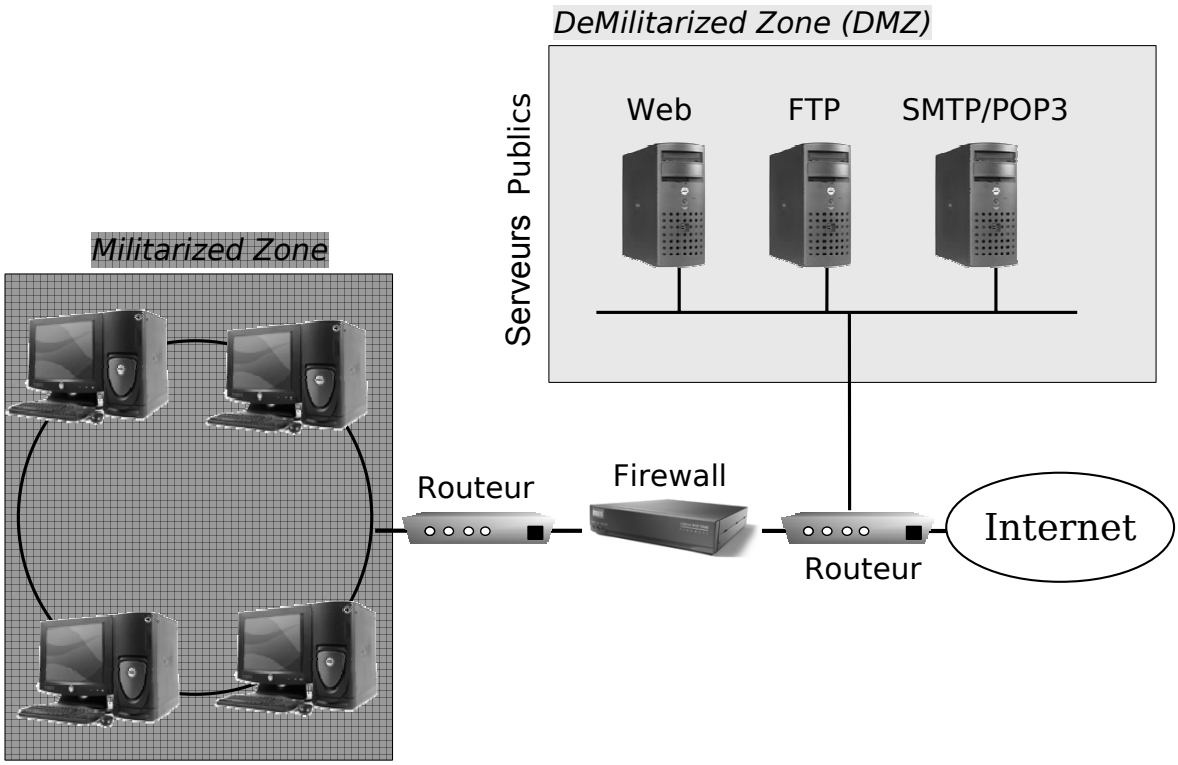
- Architecture à double routeur
  - Un routeur pour les connexions entrantes
  - Un routeur pour les connexions sortantes
  - Le firewall contrôle les accès entrants et sortants



Firewall et zone démilitarisée (DMZ)

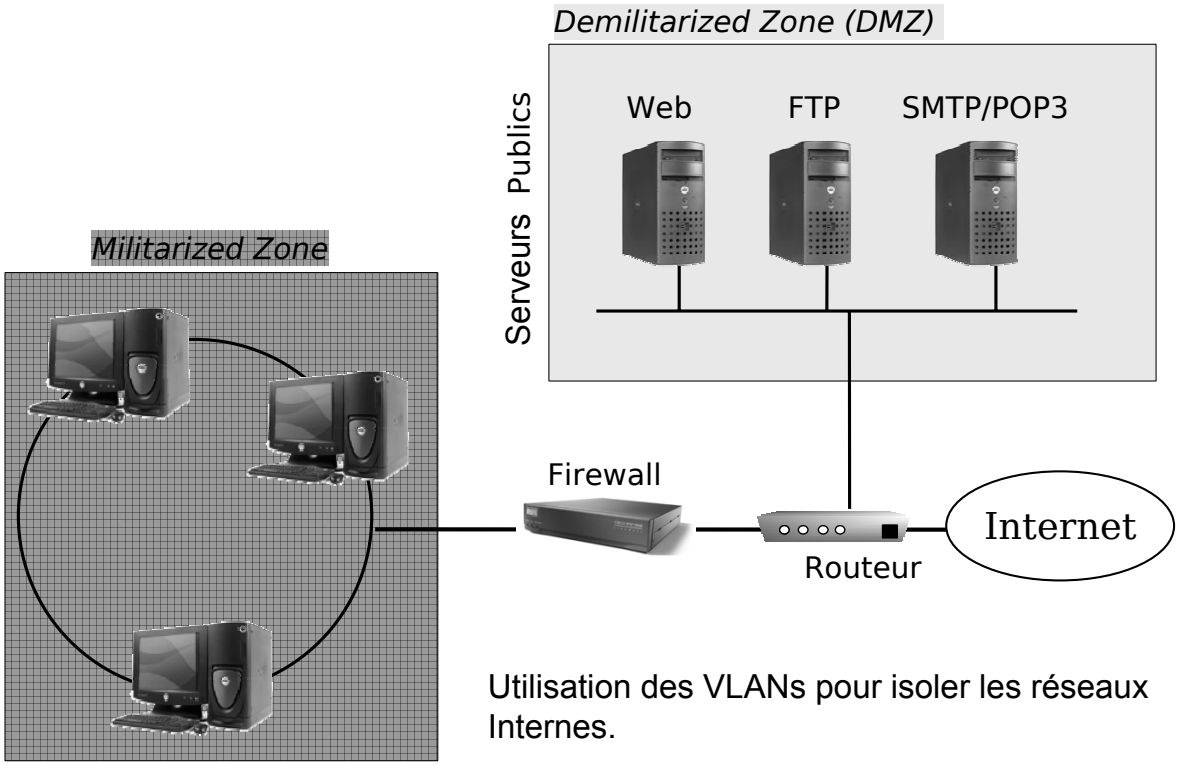
- ***Architecture à « DMZ »***
  - Découpage du réseau interne en 2 zones isolées
- Serveurs accessibles de l'extérieur, situés en « zone démilitarisée »
- Postes clients inaccessibles de l'extérieur (situé en « zone militarisée »)
- Variantes principales
  - Utilisation de deux routeurs
  - Utilisation d'un routeur « à 3 pattes »

Firewall et zone démilitarisée : 2 routeurs



Parefeu – NAT - SSL/TLS

Firewall et zone démilitarisée : Routeur à 3 pattes



Utilisation des VLANs pour isoler les réseaux Internes.

Parefeu – NAT - SSL/TLS

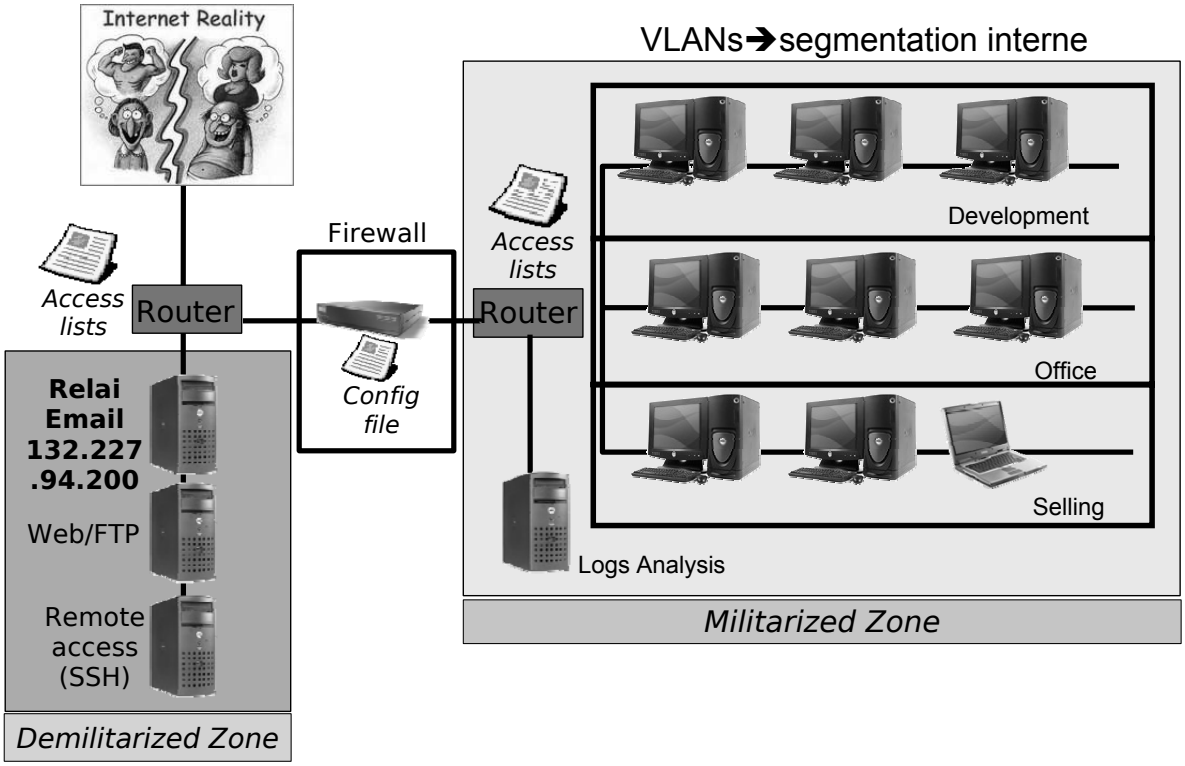
Firewall et zone démilitarisée

- ***Avantages-Inconvénients***
- Routeur à 3 pattes
  - plus économique, 1 seule machine à gérer
- 2 routeurs
  - Configuration plus simple
  - Meilleure sécurité
  - 2 étapes successives pour « braquer » le réseau interne

Utilisation des VLANs

- Le VLAN permet un regroupement d'un ensemble de machines
  - Permet l'isolement du groupe de machine
  - Regroupement par thème (pas par unité physique)
- Niveau de VLAN
  - Niveau 1 ➔ port physique sur le routeur
  - Niveau 2 ➔ adresse MAC de la machine
  - Niveau 3 ➔ sous-réseau IP ou port du service

Firewall, VLAN et zone démilitarisée



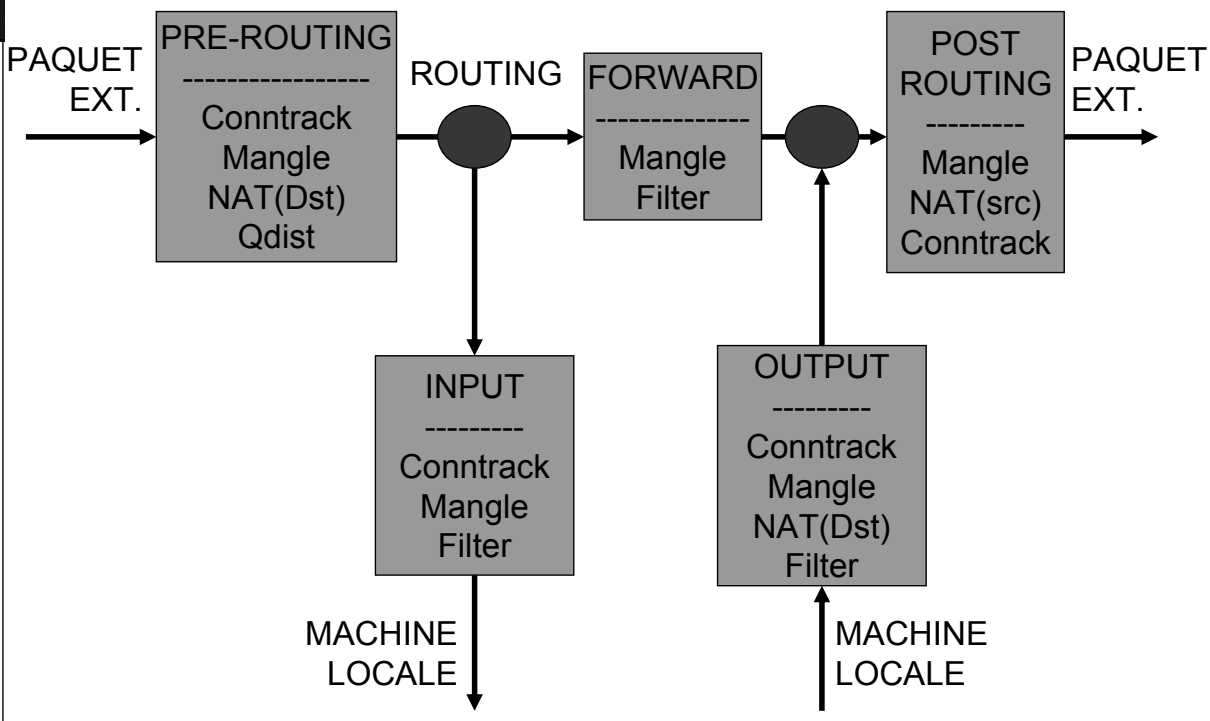
Format général et expressions des règles

- Principe : Des règles regroupées en chaînes
- Une chaine est
  - Un ensemble de règles
- Un règle est constituée
  - D'une cible ➔ en cas de concordance à qui dois-je faire suivre le paquet
  - De filtres sur l'adresse source [IP/Port] ➔ qui envoie le paquet
  - De filtres sur l'adresse destination [IP/Port] ➔ qui envoie le paquet
  - D'options (comme l'état de la connexion, l'utilisateur)
    - Ex: La connexion a-t-elle été établie (ESTABLISHED) auparavant ?
- Applicable la grande majorité des firewalls
  - Iptables, ipfw, cisco ACL

Format général et expressions des règles

- Règles de filtrage
  - SMTP-ENTRANT-1 : Autoriser tous les paquets TCP entrants, @IP destination 132.227.94.200 port 25
  - SMTP-ENTRANT-2: Autoriser tous les paquets TCP provenant du port 25 de 132.227.94.200.
  - SMTP-SORTANT-1: Autoriser les paquets émis depuis 132.227.94.200 vers port 25 autre machine
  - SMTP-SORTANT-2: Autoriser les paquets émis depuis le port 25 d'une machine vers 132.227.94.200 et **ayant le bit ACK à 1**
- Problème sur les routeurs sans état :
  - La dernière règle n'empêche pas le passage de paquets «bidouillés» en direction du serveur de courrier
  - Laisse passer les tentatives d'attaques par saturation (DOS, déni de service)
- Solution : routeurs avec état
  - Le routeur mémorise les connexions TCP établies
  - Refuse les paquets qui n'en font pas partie, ou ne sont pas le début d'une nouvelle connexion autorisée

Iptables: Firewall + NAT !



Iptables: les types de table & les chaînes

- Types de table :
  - Conntrack ➔ Permet de gérer les connexions (iptable est statefull)
  - Mangle ➔ Permet la modification des options des paquets
  - Filter ➔ Permet le contrôle des paquets (sources, destinations)
  - Qdist ➔ pour faire de la QoS
  - NAT dst ➔ Changer l'adresse [IP/Port] de destination d'un paquet
  - NAT src ➔ Changer l'adresse [IP/Port] source d'un paquet
- Chaînes cibles préexistantes pour les tables :
  - INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING (f° de la table)
  - ACCEPT ➔ Le paquet est accepté
  - DROP ➔ /dev/null
  - LOG ➔ Le paquet est tracé dans syslog [fabrique kernel]. Non bloquant !
  - REJECT ➔ Le paquet est rejeté et le firewall renvoie une erreur ICMP
  - RETURN ➔ Renvoie le paquet dans la chaîne précédente juste après l'endroit du branchement. Comportement par défaut à la fin d'une chaîne.
  - QUEUE ➔ Place le paquet dans l'environnement utilisateur

Iptables: manipulation des chaînes

- Création de la chaîne utilisateur « blacklist »

  iptables -N blacklist
- Suppression de la chaîne utilisateur « blacklist »

  iptables -X blacklist
- Vider une chaîne « chain » ou toutes les chaînes

  iptables -F [chain]
- Fixer le comportement par défaut de la chaîne « blacklist »

  iptables -P blacklist DROP
- Ajouter une règle « rule » à la chaîne « chain »

  iptables -A chain rule
- Insérer une règle « rule » à la chaîne « chain » après la position « num »

  iptables -I chain [num] rule
- Effacer une règle « rule » de la chaîne « chain » en position « num »

  iptables -D chain [num] [rule]
- Remplacer une règle « rule » en position « num » de la chaîne « chain »

  iptables -R chain [num] [rule]

Iptables : Un exemple de script

- Un firewall basique pour une machine cliente simple

#Tout interdire en entrée par défaut

Iptable –t filter –P INPUT DROP

#Autoriser les connexions entrantes en loopback

Iptables –t filter –A INPUT –i lo –s 127.0.0.0/8 –d 127.0.0.0/8 –j ACCEPT

#ACCEPTE LES CONNEXIONS déjà ETABLIES

Iptables –t filter –A INPUT –i lo -m state --state ESTABLISHED –j ACCEPT

#ACCEPTE LES NOUVELLES CONNEXIONS ENTRANTES SUR LE PORT 80

Iptables –t filter –A INPUT –i eth0 –m state --state NEW –p TCP --dport www –m limit --limit 1/s –j ACCEPT

#ACCEPTE LE PING AVEC ANTIFLOOD

Iptables –t filter –A INPUT –p icmp –icmp-type echo-request –m limit --limit 1/s –j ACCEPT

Iptables: Un exemple de script

- Ajout d'une black liste en entrée

 # créer une nouvelle chaîne

Iptable –N blacklist

 # fixer le comportement par défaut

Iptable –t filter –P blacklist RETURN

#On ajoute les adresses interdites

Iptables –t filter –A blacklist –i eth0 –s 10.0.0.0/8 –j DROP

Iptables –t filter –A blacklist –i eth0 –s 192.168.0.0/16 –j DROP

 # On ajoute un saut vers cette règle dans la règle INPUT du firewall

Iptables –t filter –I INPUT 3 –j blacklist

Iptables: Un exemple de script

#Ne pas faire suivre les paquets

Iptable –t filter –P FORWARD DROP

#Interdire toutes les connexions sortantes par défaut

Iptable –t filter –P OUTPUT DROP

#Autoriser les connexions sortantes en loopback

Iptables –t filter –A OUTPUT –o lo –s 127.0.0.0/8 –d 127.0.0.0/8 –j accept

#ACCEPTE LES CONNEXIONS déjà ETABLIES

Iptables –t filter –A OUTPUT –i lo -m state --state ESTABLISHED –j ACCEPT

#ACCEPTE LES NOUVELLES CONNEXIONS SORTANTES SUR LE PORT 22

Iptables –t filter –A OUPUT –m state --state NEW –p TCP –dport ssh -j ACCEPT

Iptables: les extensions

- Il existe de nombreux modules
  - « state » pour examiner l'état de la connexion
  - « string » pour examiner le contenu du paquet
  - « recent » pour gérer les derniers paquets reçus
  - « condition » pour déclencher un règle en fonction d'une variable dans /proc/net/ipt_condition/variable
  - « iplimit » pour limiter le nombre de connexions par @IP
  - « length » pour limiter le taille de certains paquets (ex: icmp)

Iptables : shorewall

- Il existe des scripts préfabriqués pour faire des firewalls « facilement »
- Beaucoup d'options prédéfinies !
- Voir les fichiers de configurations dans /etc/shorewall et leurs commentaires.
- Efficace en terme de temps de déploiement et de sécurité !
- Fichier /etc/shorewall/interfaces
  - Choix des interfaces et des options/zones associées
- Fichier /etc/shorewall/rules

  ACCEPT net fw icmp 8 - - 3/sec:10

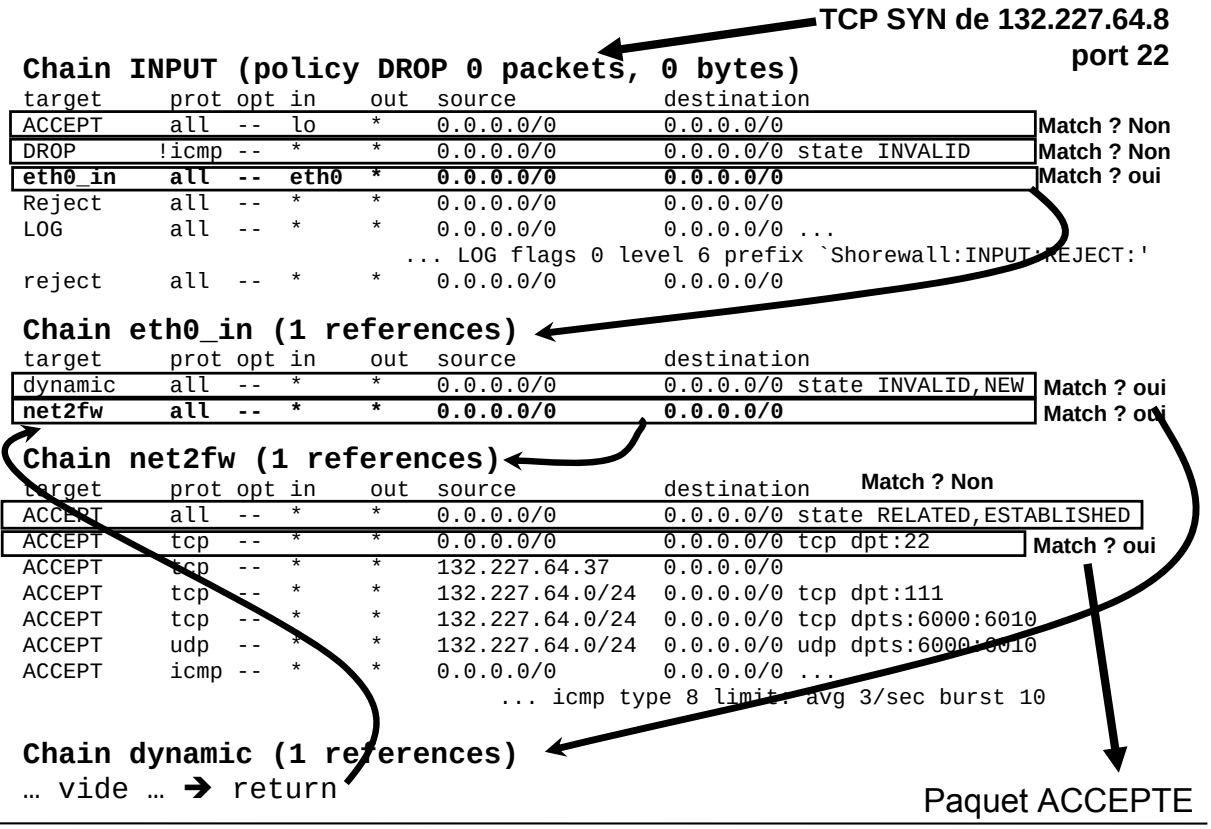
  ACCEPT net fw tcp 22 -
- Lancement : **/etc/init.d/shorewall restart**
- Génération automatique des règles iptables. Pour les voir:

  /etc/init.d/shorewall status

Iptables: Exemple de code Shorewall

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
target     prot opt in     out     source               destination
ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
DROP       !icmp --  *      *       0.0.0.0/0            0.0.0.0/0 state INVALID
eth0_in    all  --  eth0   *       0.0.0.0/0            0.0.0.0/0
Reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0
LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0 ...
                                    ... LOG flags 0 level 6 prefix `Shorewall:INPUT:REJECT:'
reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0

Chain eth0_in (1 references)
target     prot opt in     out     source               destination
dynamic    all  --  *      *       0.0.0.0/0            0.0.0.0/0 state INVALID,NEW
net2fw     all  --  *      *       0.0.0.0/0            0.0.0.0/0

Chain net2fw (1 references)
target     prot opt in     out     source               destination
ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0 state RELATED,ESTABLISHED
ACCEPT     udp  --  *      *       132.227.64.37        0.0.0.0/0
ACCEPT     tcp  --  *      *       132.227.64.37        0.0.0.0/0
ACCEPT     udp  --  *      *       224.0.1.1            0.0.0.0/0 udp dpt:123
ACCEPT     tcp  --  *      *       132.227.64.0/24      0.0.0.0/0 tcp dpt:111
ACCEPT     tcp  --  *      *       132.227.64.0/24      0.0.0.0/0 tcp dpts:6000:6010
ACCEPT     udp  --  *      *       132.227.64.0/24      0.0.0.0/0 udp dpts:6000:6010
ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0 tcp dpts:63000:64000
ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0 tcp dpt:22
ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0 ...
                                         ... icmp type 8 limit: avg 3/sec burst 10
net2all    all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

Iptables: Voyage d'un paquet



ipfw

- IPFW est l'ancêtre de iptables
- Il est encore très présent (FreeBSD, MacOS)
- Il a très bonne réputation.
- Fortement intégré au noyau:
  - Le changement d'une option oblige à la recompilation du noyau
  - Tout un ensemble d'option sont disponibles
    - IPFIREWALL_VERBOSE, IPDIVERT, …
- Déploiement du firewall: fichier /etc/rc.conf

```
firewall_enable="YES"
firewall_script="/etc/firewall/fwrules"
natd_enable="YES"
natd_interface="tun0"
natd_flags="-dynamic"
```

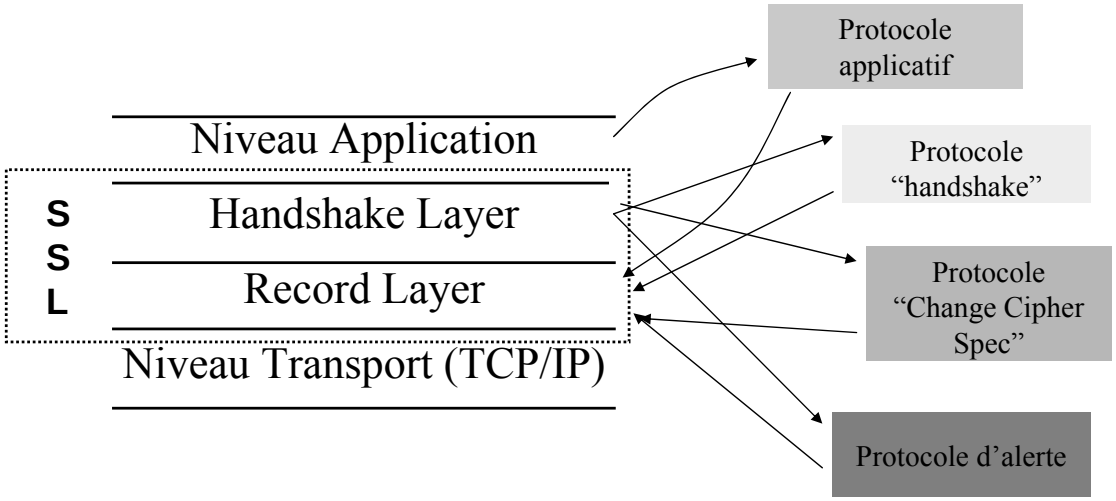
Ipfw : fichier de règles

```
# Define the firewall command (as in /etc/rc.firewall) for easy reference. fwcmd="/sbin/ipfw"

# Force a flushing of the current rules before we reload.
    $fwcmd -f flush

# Allow all data from localhost.
    $fwcmd add allow ip from any to any via lo0

# Basically give user root "GOD" privileges. $pif ➔ network card
    $fwcmd add allow tcp from me to any out via $pif setup keep-state uid root

# Allow out ping
    $fwcmd add allow icmp from any to any out via $pif keep-state

#Allow out Time
    $fwcmd add allow tcp from any to any 37 out via $pif setup keep-state

# This rule enforces the block all by default logic.
    $fwcmd add deny log all from any to any out via $pif

# Deny all inbound traffic from non-routable reserved address spaces
    $fwcmd 00300 deny all from 192.168.0.0/16 to any in via $pif #RFC 1918 private IP
    $fwcmd 00301 deny all from 172.16.0.0/12 to any in via $pif #RFC 1918 private IP
    $fwcmd 00302 deny all from 10.0.0.0/8 to any in via $pif #RFC 1918 private IP
    $fwcmd 00303 deny all from 127.0.0.0/8 to any in via $pif #loopback
    $fwcmd 00304 deny all from 0.0.0.0/8 to any in via $pif #loopback
    $fwcmd 00305 deny all from 169.254.0.0/16 to any in via $pif #DHCP auto-config
    $fwcmd 00306 deny all from 192.0.2.0/24 to any in via $pif #reserved for docs
    $fwcmd 00308 deny all from 224.0.0.0/3 to any in via $pif #Class D & E multicast
    .....
```


SSL et TLS

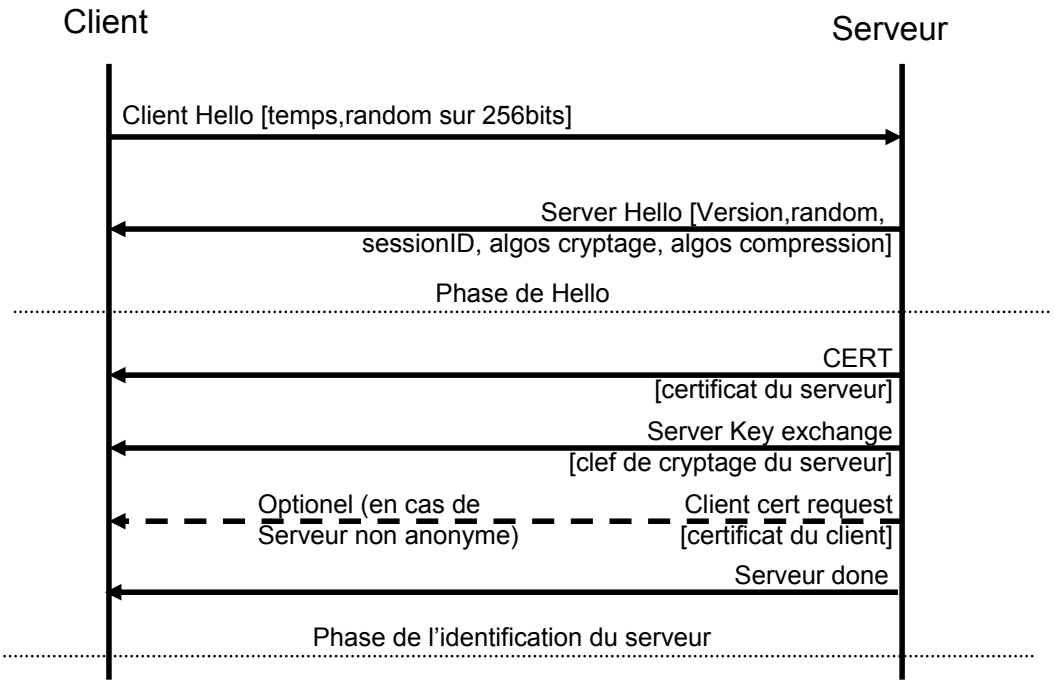
- SSL = « Secure Socket Layer »
  - Surcouche logicielle à TCP/IP
  - Utilise un protocole particulier et doit donc être sur un port particulier
  - S'insère entre la couche TCP et la couche applicatif
  - Nécessite une adaptation de l'application
  - Permet l'identification des clients (Par exemple pour un intranet)
- TLS = « Transport Secure Layer »
  - Reprend tous les concepts SSL. Plus générique et plus clair que SSL.
  - TLS n'impose pas de méthode de chiffrement spécifique
  - Utilise les mêmes couches (une couche de négociation et une couche de création de paquets)
  - Utilise les mêmes protocoles (handshake, alert, cipher change)
  - RFCisé : 2246/3546 (TLS), 2487 (SMTPs), 2595 (POPs, IMAPs), 2817/2818 (HTTPs)
- Technologie SSL plus connue et plus exploitée que TLS

Transport (/ Secure) Socket Layer

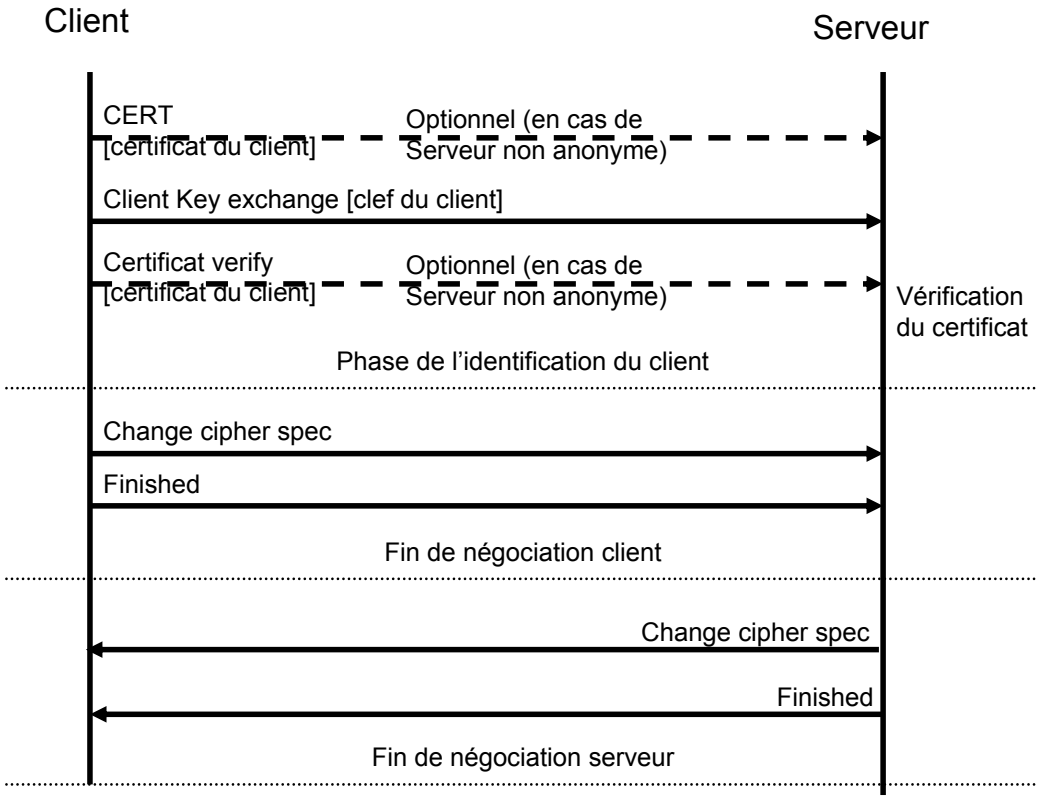


Transport (/ Secure) Socket Layer

- Négociation SSL



Transport (/ Secure) Socket Layer



Transport (/ Secure) Socket Layer

- **DES, Triple-DES, MD5, RSA, SHA-1.**
- **DSA.** Digital Signature Algorithm
- **KEA.** Key Exchange Algorithm. Un algorithme utilisé pour l'échange de clé par le gouvernement US.
- **RC2 and RC4.** Un algorithme de cryptage Rivest dévelopé pour le "RSA Data Security".
- **RSA key exchange.** Un algorithme d'échange de clefs pour SSL basé sur l'algorithme RSA.
- **SKIPJACK.** Un algorithme de cryptage à clef symétrique implanté par la société FORTEZZA et utilisé par le gouvernement US (voir FORTEZZA Cipher Suites.)

Plan de cours

IDS et Analyse

Introduction

Attaques de niveau 1

Attaques de niveau 2: ethernet

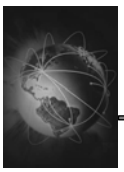
Attaques de niveau 3: IP/ICMP

IPSEC - VPN

Attaques de niveau 4: TCP

SSL/TLS – Parefeu - NAT

IDS et Analyse



« fwlogwatch » : Analyse de log firewall

- N'analyse que les paquets tracés
  - Nécessiter de bien définir les paquets à tracer
  - Les paquets jetés (drop/reject) sont à tracer
  - Certains paquets acceptés doivent être tracés
    - Ouverture de connexion (TCP SYN)
  - Ne capte pas toute l'activer réseau de la machine !
- « fwlogwatch » :
  - Il génère des rapports
  - Il peut surveiller l'évolution des logs
    - Générer des alertes via des scripts (EMAIL !)

Analyse de log firewall

- Fichier de configuration « /etc/fwlogwatch.conf » :

```
verbose = yes                                   # be verbose

resolve_hosts = yes                             # try to get hostname &
servicename
resolve_services = yes

parser = n                                      # set parser to iptable

src_ip = on                                     # show all informations
dst_ip = on
protocol = on
src_port = on
dst_port = on
tcp_opts = on
data_amount = yes
start_times = yes
end_times = yes
duration = yes

html = yes                                      # set ouput to html
output = /var/www/fwlogwatch/index.html
sender = root@src.lip6.fr
recipient = root@src.lip6.f
```


fwlogwatch : Exemple

fwlogwatch HTML

- C'est joli les premières fois
  - Mais on se lasse vite
- Une fois en place
  - il faut prendre le temps chaque jour
  - Il faut être persévérant
- Si on ne le regarde plus, autant arrêter la génération des comptes rendus

Sonde IDS

- IDS = « Intrusion Detection System »
- Utilité des sondes IDS ?
  - Il est impossible de se protéger contre toutes les attaques
  - Détecter les attaques non bloquées
  - Empêcher l'espionnage de son réseau en le détectant précocement
  - Collecter le maximum d'information sur un attaque et sur les attaque (« know your ennemy »)
- Pour que la sonde soit efficace
  - Doit percevoir son environnement
  - Doit limiter ses interactions avec son environnement
    - Eviter les détections de la sonde

IDS et Analyse

Sonde IDS

- Type de sonde IDS
  - Sonde réseau ➔ analyse des trames
  - Sonde de machine ➔ analyse les événements machines
  - Sonde applicative ➔ analyse un service particulier (peut être une combinaison de plusieurs machines)
- La détection se fait par
  - Des signatures (comportementales)
  - La détection d'anomalies
- Certains IDS ont des capacités préprogrammées de réactions

IDS et Analyse

Sondes IDS de type réseau

- NIDS = « Network based IDS »
- Type de sonde très employée
- Caractéristiques :
  - Elle écoute sur des points stratégiques du réseau
    - Doit recevoir tout le trafic du réseau (dorsale)
    - Penser à la brancher sur le routeur d'entrée et annuler l'isolation (ex: VLAN)
  - On peut segmenter les écoutes (plusieurs sondes)
  - Une sonde peut et doit être fortement sécurisée
    - Contrôle d'accès fortement limité
    - Hardened kernels
  - Une bonne configuration la rend difficilement détectable

Sondes IDS de type réseau

- Avantages
  - Peu de sondes bien placées peuvent surveiller un large réseau
  - Le déploiement de sonde a peu d'impact sur le réseau existant
    - Peu d'effort de configuration pour déployer une sonde
  - Les sondes sont camouflables et bien protégées
- Inconvénients
  - Surcharge possible de la sonde (réseau, CPU)
    - Certaines sondes utilisent un matériel spécifique
  - Certains commutateur bas de gammes n'offre pas de ports de surveillance (pas de copie du trafic)
  - Pour l'instant pas d'analyse de données cryptées
    - Il faudrait connaître toutes les clefs et les protocoles de son réseau
  - Difficulté à savoir si une attaque à réussi ou non

Sondes IDS de type Machine

IDS et Analyse

- Sonde IDS de type machine ➔ HIDS (Host based IDS)
- Caractéristiques
  - Analyse l'activité système de la machine sur laquelle elle est déployée
    - « *System Integrity Verifier* » ➔ Vérification des modifications apportées sur les fichiers du système
    - « *Log file monitor* » ➔ Vérification des traces systèmes
  - Permet de déterminer les activités suspectes de certains processus
  - Certains sondes peuvent transmettre les informations à un concentrateur
  - Certains sondes peuvent générer des messages réseaux (SNMP) ou des emails

Sonde IDS de type Machine

IDS et Analyse

- Avantages
  - Détecte des attaques non détectables sur le réseau
  - Peut être utilisé dans des environnements « cryptés »
  - Peut être utilisé sur des réseaux commutés
- Inconvénients
  - Les bases doivent être mise à jour sur les machines
  - Les sondes peuvent être corrompues/désactivées par l'attaquant
  - Pas de vision globale (scans réseaux) car elle est déployée sur une machine « cliente »
  - La sonde machine consomme du CPU et de la mémoire sur la machine « cliente »
  - Difficulté à détecter les dénis de services

Sondes IDS de type applicative

- Très proches des sondes machines
- Souvent confondues avec les sondes machines
- Caractéristiques
  - Encapsule une application
  - Surveille une application et ses évènements
  - Spécialiser pour chaque type d'application
- Avantages
  - Très fine granularité pour analyser des comportements anormaux
  - Analyse les données après le décryptage
- Inconvénients
  - Sensibilité extrême aux attaques
  - Facilement corruptible

IDS et Analyse

Sondes IDS : analyse par signatures

- Déclenchement sur la détection d'une suite prédéfinie d'évènements
- Méthode peu coûteuse et efficace
- Avantages
  - Peu de fausses alarmes
- Inconvénients
  - Il faut mettre à jour la base de signature
  - Ne détecte pas toujours toutes les variantes des attaques

IDS et Analyse

Sondes IDS : analyse par détection d'anomalies

IDS et Analyse

- Identification de comportements anormaux
- Un bon complément à l'analyse par signature
- Nécessité de définir une (la?) normalité
- Fonction d'apprentissage des comportements normaux
  - Analyse statistique des comportements
- Détection de déviance vis-à-vis de la normalité
- Avantages
  - Détection d'attaques non encore identifiées
- Inconvénients
  - Produits de nombreuses fausses alarmes
  - Nécessite un apprentissage couteux et long

Sondes IDS : réponses automatiques

IDS et Analyse

- L'administrateur n'est pas toujours présent
- Les réponses passives automatiques peuvent être :
  - Envoie de notifications et déclenchement d'alarmes ➔ NECESSAIRE !
    - Email, sms, ... aux responsables
  - Envoie de trames SNMP
  - Utilisation de plug-in
  - Archivage automatique sur un support sûr
- Les réponses actives peuvent être
  - Changer l'environnement interne ➔ PEUT ETRE DANGEREUX ET ETRE EXPLOITE !
    - Injection de trames RST pour couper l'attaque
    - Reconfiguration des routeurs et firewall pour bloquer @IP
    - Reconfiguration des routeurs et firewall pour bloquer les protocoles
    - Bloquer totalement le réseau dans les cas extrêmes
  - Collecte automatique d'informations sur la source ➔ réponse la plus efficace
    - Whois, nmap, traceroute
    - Changer le niveau de logging de l'IDS, activer d'autres type de sonde
  - Attaquer l'attaquant ! ➔ réponse la plus dangereuse (légalement et techniquement)
    - Interdit par la loi et peut attaquer quelqu'un d'innocent

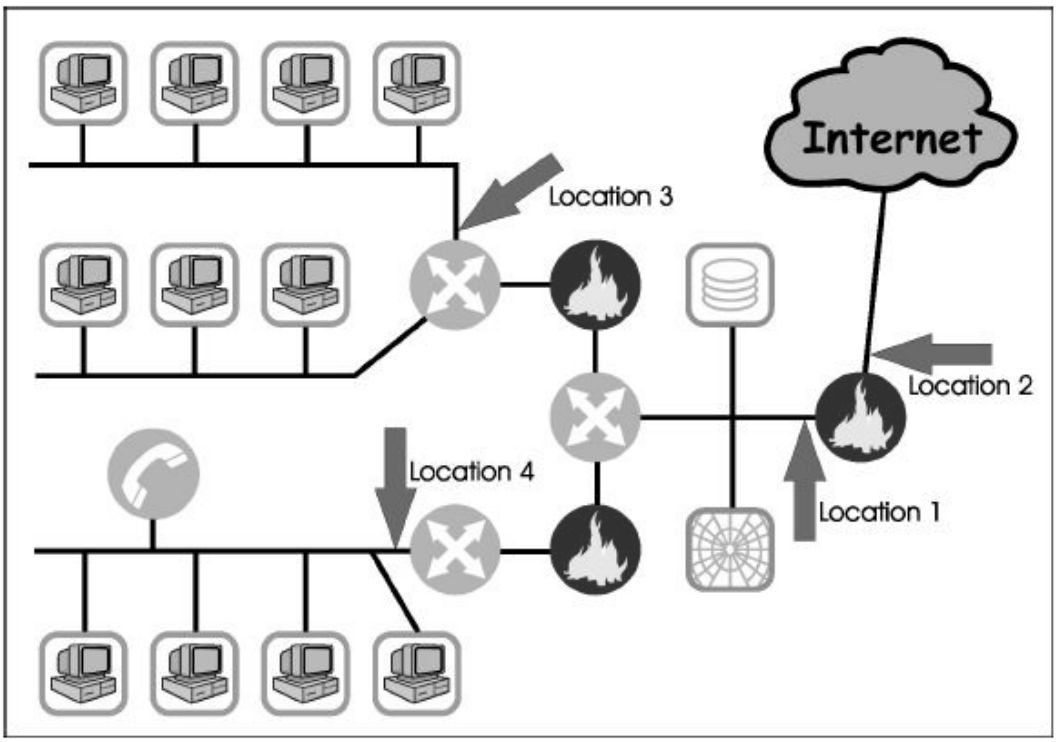
Sondes IDS : limitations ?

- Etre conscient des limitations des IDS
- Difficultés d'estimer les ressources nécessaires
  - Ressources CPU/Réseau
  - Ressources humaines pour traiter les alarmes et les MAJ des IDS
- Génération de faux positifs coûteux pour les administrateurs systèmes
- Les IDS mêmes considérés comme temps-réels mettent parfois plusieurs minutes à réagir ➔ dû à la charge de la machine
  - Temps extrêmement long en informatique
- Latence entre la mise à jour des bases et le déploiement
  - Attaque non détectée possible entre la publication et la MAJ de l'IDS
- Les réponses automatiques
  - Sont souvent ineffectives contre les hackers expérimentées
  - Peuvent gêner le trafic légal
- Les IDS ne sont pas forcement protégés contre les attaques
- Les IDS n'ont pas toujours de GUI et d'outils d'analyse efficaces

Sondes IDS : Localisation

- Localisations de déploiement
  - Derrière chaque firewall externe
  - Dans le réseau DMZ
  - Devant le firewall frontale (point d'entrée)
  - Sur les dorsales internes des sous-réseaux
- Ce sont des conseils !
  - Vous devez adapter votre stratégie IDS à
    - Votre réseau
    - Aux ressources financières
  - C'est un domaine encore en phase de R&D

Sondes IDS : Localisation



NIDS payants

- Network ICE, BlackICE, Win
  - Auto-update, TCPIP/ARP, SMNP
- Network Associate, Cybercop Monitor, Win
  - Product update, TCPIP, pas SNMP
- Cisco, Netranger, Solaris
  - MAJ par CD, TCPIP, SNMP
- ISS, RealSecure, Win/Unix/Linux
  - MAJ par HTTPs, TCPIP, SNMP
- AXENT, Omniguard, Win/AIX/Unix/HP/Solaris
  - MAJ par HTTPs, pas SNMP
- . . .

NIDS gratuits

- Les applications libres
  - Snort, Prelude, BroIDS, hogwash ➔ Sondes de type réseau
  - Nessus, ettercapNG ➔ dans le domaine de l'audit de vulnérabilités
  - IDSwakup ➔ permet de générer du trafic réseau anormal
  - Nmap, dsniff, kismet ➔ scan réseau
  - Argus ➔ Nework logger
- Elles font partie des références incontestables dans leur domaine.
  - performances parfois supérieures à celle d'applications commerciales souvent vendues très cher.
- Problème :
  - La société éditrice de Snort vient d'être rachetée
  - L'auteur de nessus ne veut plus publier ses sources
- Ils sont quand même à déployer car incontournables !
  - En attendant mieux ou leurs récupérations open source
- Il faut déployer PLUSIEURS IDS SIMULTANEEMENT !!!!!

IDS snort : configuration

- Installation
  - Edition du fichier /etc/snort/snort.conf
  - Modifier la variable HOME_NET pour définir le périmètre de confiance (ex: votre réseau)
  - L'extérieur est définit comme ! HOME_NET
  - Activation possible du log dans une DB ou un fichier binaire (bainard)
    - Variable « output »
    - Permet des débits plus importants que le texte pur
  - Activer tous les plugins (*preprocessors*) et les règles (*include*) qui vous intéressent
    - Préprocesseurs: sfPortscan, stream4, httpinspect, rpcdecode
    - En cas de réseau exposé, activer les règles BleedindEdge
    - Les règles de bases sont à MAJ sur Arachnids
    - Fixer les options
- /etc/init.d/snort start

IDS snort : outils

IDS et Analyse

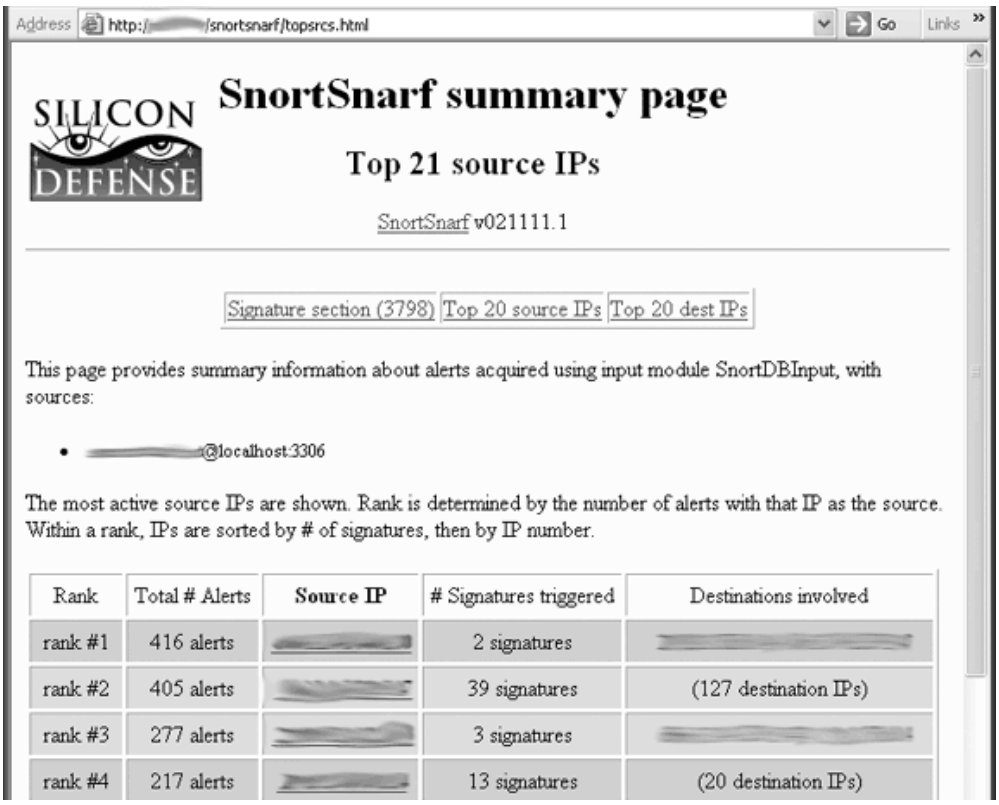
- Snort ne fait que tracer les attaques
  - Pas d'email
  - Pas de modification des règles de firewall
    - Pour iptables ➔ module snortsam ou snort-inline
- Pour emailer les alertes snorts
  - Swatch, IDSCenter, logsurfer
- Analyse et gestion des logs pour snort
  - Snortsnarf produit un rapport HTML à partir des logs snort
  - ACID produit un rapport HTML à partir d'une BD snort
  - Cerebus pour analyser les logs
  - Autres: 5n0r7, SnortReport, SnortBot, SnortPHP, snort_stat.pl (livré avec snort)

IDS snort : outils

IDS et Analyse

- GUI pour snort
  - HenWen (MacOSX), IDSCenter (Win), SnortCenter (Linux/Win)
- Maintenance des logs snort:
  - Guardian, logsnorter, snortlog
- Outils
  - Getcontact, Hogwas Signature
- Configuration snort
  - IDS Policy Manager, Snort Webmin Module
- Mise à jour des règles snort
  - oinkmaster
- Pour obtenir les modules Snort : http://www.snort.org/downloads.html

IDS snort : Exemple snortsnarf

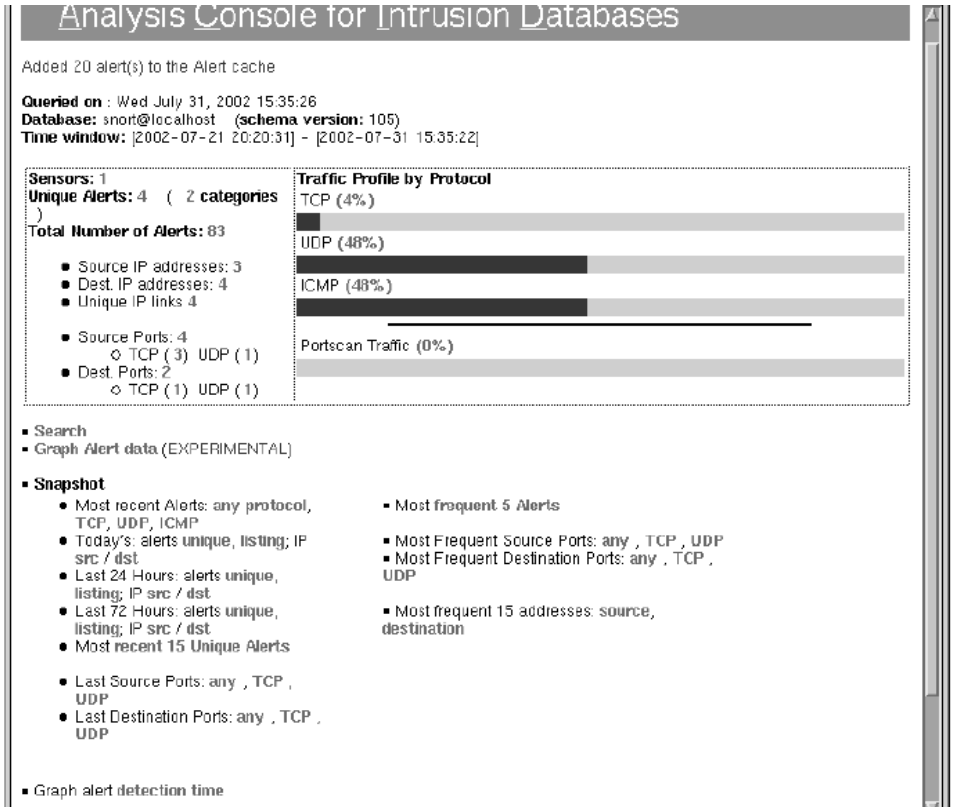


IDS snort : Exemple snortsnarf

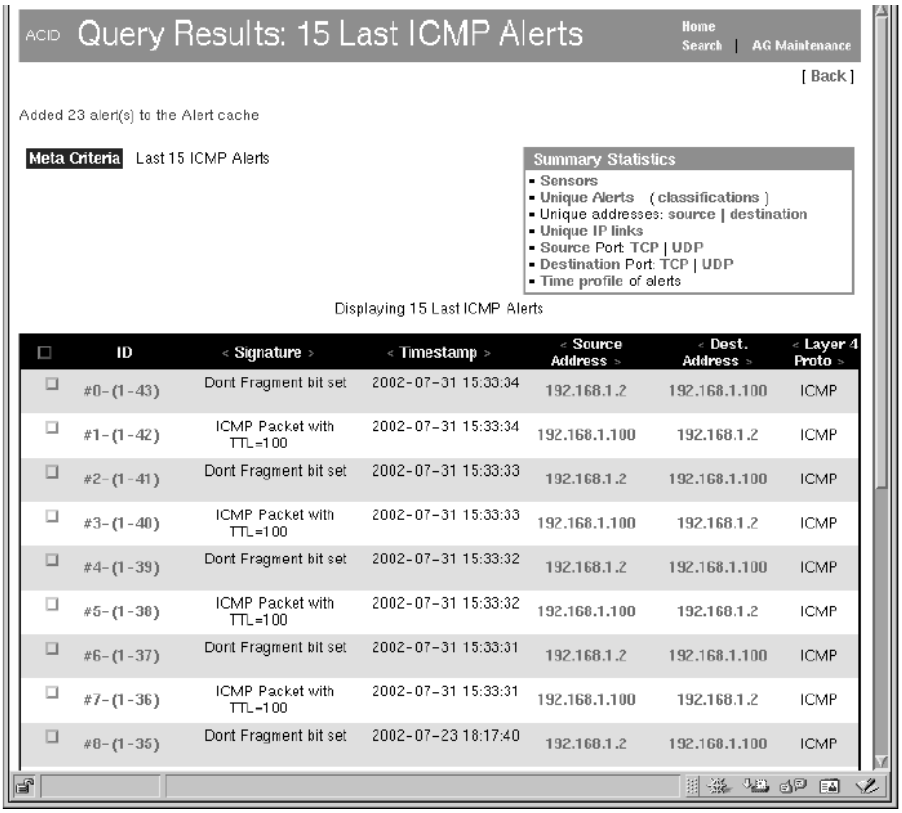
Snortsnarf HTML

- Comme fwlogwatch
  - c'est joli les premières fois ➔ Mais on se lasse vite
  - Une fois en place ➔ il faut prendre le temps chaque jour
- Si on ne le regarde plus, autant arrêter la machine
- Exemples :
  - Summary : connexion BO
  - Summary : trafic hors norme (UDP 53 vers 139 !!)
  - Top 20 source IP (211.137.96.156) : Host zombie typique
    - 1 type d'attaque, répéter régulièrement
  - Top 20 source IP (216.136.86.44) : Scan de proxy web/socks
  - Top 20 destination IP ➔ machines les plus ciblées

IDS snort : Exemple ACID



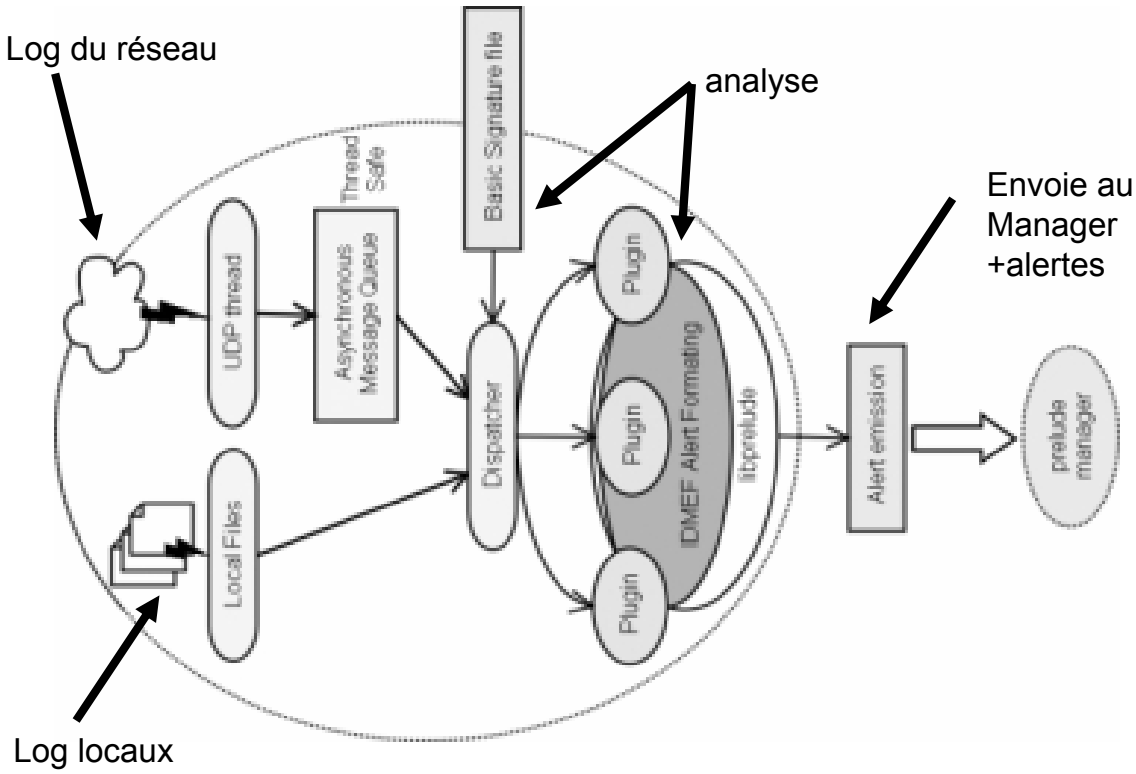
IDS snort : Exemple ACID



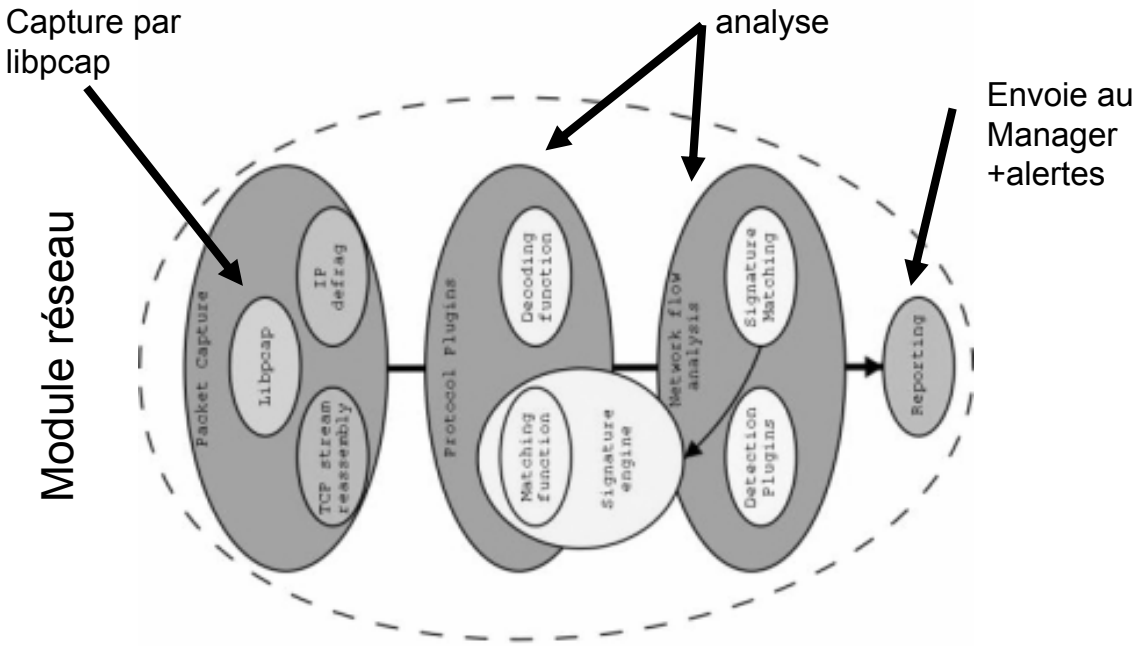
IDS: Prelude

- « Prelude » est similaire à snort
- Il est composé de 3 type de serveurs
  - Un serveur manager ➔centralise les informations collectées par les sondes
    - « prelude-manager », fichiers « /etc/prelude-manager/* »
  - Un serveur sonde NIDS ➔ collecte des informations réseaux en un point du réseau
    - « prelude-nids », fichiers « /etc/prelude-nids/* »
  - Un serveur sonde LML ➔ collecte des informations sur des machines (les logs)
    - « prelude-lml », fichiers « /etc/prelude-lml/* »
  - Un serveur de contre-mesure ➔ gére les actions de contre-mesures en fonction de données collectés par les managers
    - « prelude-cm-agent »
- Les communications entre les sondes et le manager est sécurisé

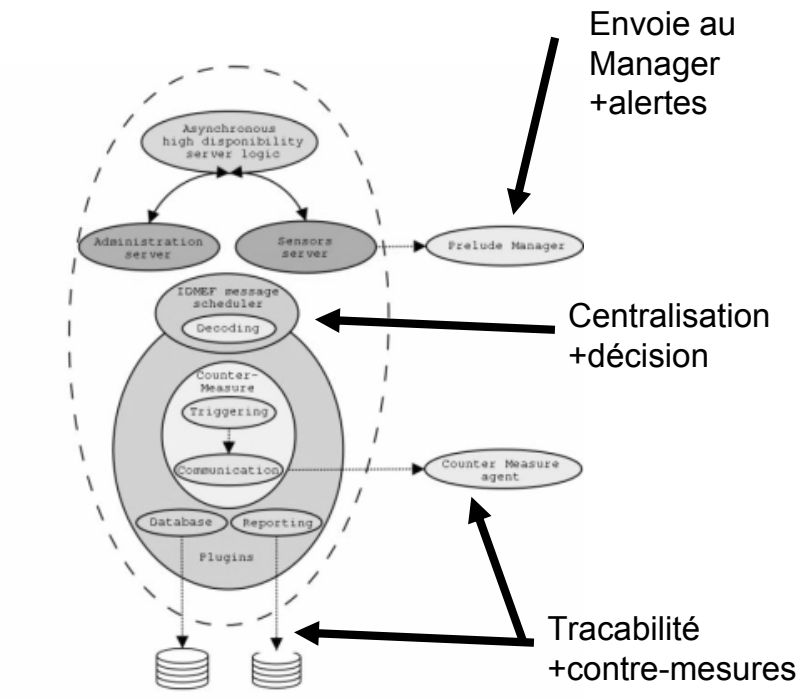
IDS: serveur « prelude-lml »



IDS: serveur « prelude-nids »



IDS: serveur « prelude-manager »



IDS Prelude : ajout d'une sonde

IDS et Analyse

- Pour l'installation, les paquetages nécessaires sont
  - « prelude-manager », « prelude-tools »
  - « prelude-nids », « prelude-lml »
  - Eventuellement une BD (mysql) et « prelude-cm-agent »
- Prelude n'accepte pas les données de sources inconnues
  ➔ il faut gérer un échange de clefs
- Pour chaque serveur, pensez au fichier /etc/prelude-xxx/prelude-xxx.conf
- Pour ajouter une sonde, il faut du côté « manager »

```
# manager-adduser
Generated one-shot password is "sa17bh20".
This password will be requested by "sensor-adduser" in order to connect.
Please remove the first and last quote from this password before using it.
- Waiting for install request from Prelude sensors...
```


IDS Prelude : ajout d'une sonde

IDS et Analyse

- Du coté sonde : « -u » uid, « -s » nom sonde, « -m » ip manager

```
# sensor-adduser -s sensor-lml -m 127.0.0.1 -u 0

Now please start "manager-adduser" on the Manager host
where you wish to add the new user.
Please remember that you should call "sensor-adduser"
for each configured Manager entry.
Press enter when done.
Please use the one-shot password provided by the
"manager-adduser" program.

Enter registration one shot password : sa17bh20
Please confirm one shot password : sa17bh20

connecting to Manager host (127.0.0.1:5553)...
Succeeded.

Username to use to authenticate : sensor-lml
Please enter a password for this user : password
Please re-enter the password (comfirm) : password
Register user “sensor-lml" ? [y/n] : y
Plaintext account creation succeed with Prelude Manager.
```

IDS Prelude : résultats collectés

- Lancer le manager (« prelude-manager ») et les sondes (NIDS et LML)
- Le fichier de log est « /var/log/prelude-manager/prelude.log »
- Il existe, comme pour snort, des générateurs de rapport
- Au démarrage de prelude-nids

```
************************************************************
* Heartbeat: ident=1
* Analyzer ID: 1041315032505060971
* Analyzer model: Prelude NIDS
* Analyzer version: 0.8.6
* Analyzer class: NIDS
* Analyzer manufacturer: The Prelude Team http://www.prelude-ids.org
* Analyzer OS type: Linux
* Analyzer OS version: 2.6.12-12mdk-i686-up-4GB
* Node[unknown]:
* Process: pid=26218 name=prelude-nids
* Creation time: 0xc76582cb.0x94c4300 (2006-01-04 00:10:03.581+0100)
*
************************************************************
```

- Même type de message au démarrage de prelude-lml

IDS et Analyse

IDS Prelude : résultats collectés (NIDS)

```
*****************************************************************************
* Alert: ident=3923
* Classification type: bugtraqid
* Classification: BAD-TRAFFIC IP Proto 103 (PIM)
* Classification URL: http://www.securityfocus.com/bid/8211
* Creation time: 0xc7658fe6.0xc659400 (2006-01-04 01:05:58.774+0100)
* Detection time: 0xc7658fe6.0xc656a00 (2006-01-04 01:05:58.774+0100)
* Process: pid=26372 name=prelude-nids path=/usr/bin
* Impact severity: medium
* Impact completion: NULL
* Impact type: other
* Impact description: Detection of a non-standard protocol or event
*** Source information ************************************************
* Source spoofed: unknown
* Node[unknown]:
* Addr[ipv4-addr]: 132.227.64.15
*** Target information ************************************************
* Target decoy: unknown
* Node[unknown]:
* Addr[ipv4-addr]: 224.0.0.13
*** Additional data within the alert  ***************************************
* Ethernet header: 0:10:d:3d:c4:0 -> 1:0:5e:0:0:d [ether_type=ip (2048)]
* Ip header: 132.227.64.15 -> 224.0.0.13 [hl=20,version=4,tos=192,len=38,id=26960,ttl=1,prot=103]
* Payload header: size=18 bytes
* Payload Hexadecimal Dump: 20 00 c9 b0  00 01 00 02  00 69 00 14 00 04 00 00 15 cb
* Detection Plugin Name: SnortRules
* Detection Plugin Author: The Prelude Team
* Detection Plugin Contact: prelude-devel@prelude-ids.org
* Detection Plugin Description: Snort signature parser.
* Snort rule ID: 2189
* Snort rule revision: 1
*****************************************************************************
```

Protocole de routage multicast CISCO

IDS et Analyse

IDS Prelude : résultats collectés (LML)

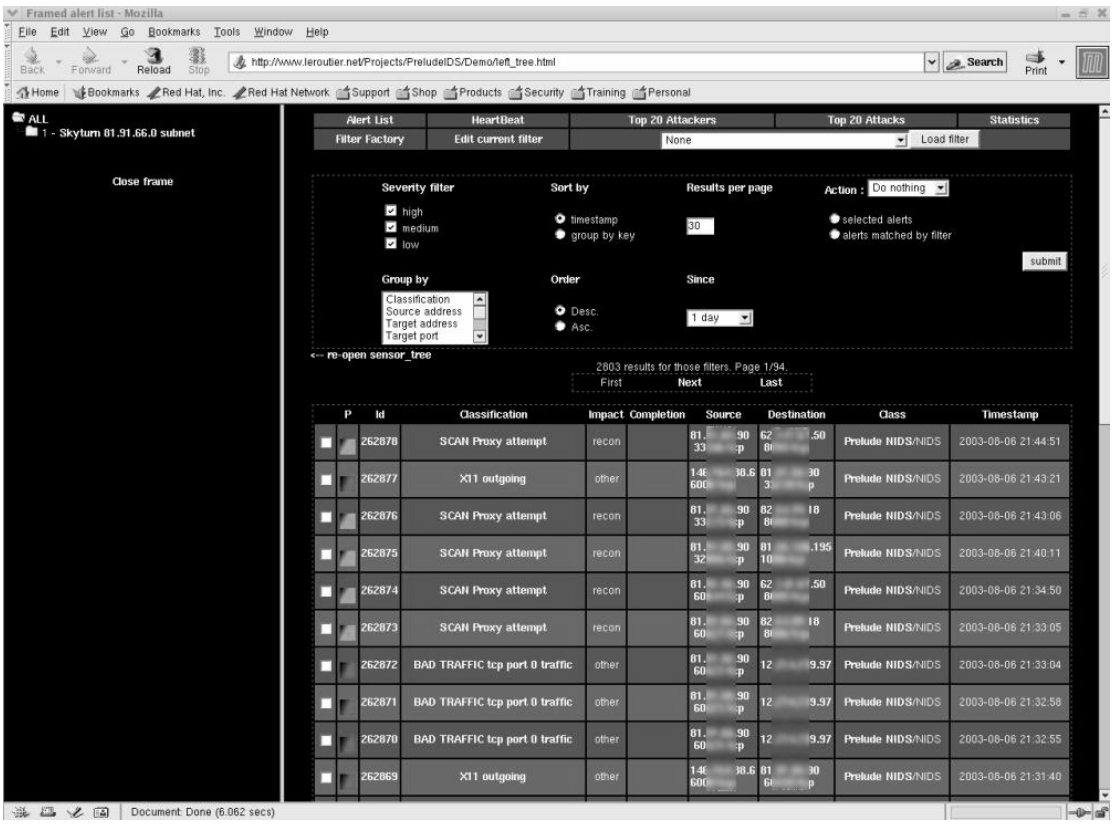
IDS et Analyse

```
********************************************************************************
* Alert: ident=4042
* Classification type: unknown
* Classification: SSH Remote root logging failed
* Classification URL: unknown
* Creation time: 0xc7659e1c.0x8b7a000 (2006-01-04 02:06:36.544+0100)
* Detection time: 0xc7659e1c.0x0000000 (2006-01-04 02:06:36.000+0100)
* Analyzer ID: 630008679108729663
* Analyzer model: Prelude LML
* Analyzer version: 0.8.6
* Analyzer class: HIDS
* Impact severity: medium
* Impact completion: failed
* Impact type: admin
* Impact description: Someone tried to login as root from 132.227.64.30:34689 using the password method
*** Source information ************************************************************
* Source spoofed: unknown
* Node[unknown]:
* Addr[ipv4-addr]: 132.227.64.30
* Service: port=34689 protocol=tcp
*** Target information ************************************************************
* Target decoy: unknown
* Service: port=22 protocol=tcp
* Target decoy: unknown
* Node[unknown]: name:eos.lip6.fr
* Addr[ipv4-addr]: 132.227.64.45
* Process: pid=0 name=sshd
*** Additional data within the alert  *************************************
* Log received from: /var/log/messages
* Original Log: Jan  4 02:06:36 eos sshd[27082]: Failed password for root from 132.227.64.30 port 34689 ssh2
********************************************************************************
```

Tentative de login Ssh en « root »

IDS Prelude : rapport HTML (piwi)

IDS et Analyse



Détection de vulnérabilité

IDS et Analyse

- « nessus », « ettercap » permettent de faire de la détection de vulnérabilité (des attaques)
- « nmap » est un bon complément
- « IDSwakeup » génère du trafic anormal pour déclencher une réaction des IDS
- A faire régulièrement ➔ Au moins une fois par mois

Legond-Aubry Fabrice
fabrice.legond-aubry@u-paris10.fr

SSI

Sécurité des Systèmes Informatiques

Sécurité des applications

Plan de cours

Introduction

Introduction

Attaques génériques de services

Attaques spécifiques de services

Backdoors / Rootkits / Trojans

Outils génériques de protection

Audit et check-list

Des attaques variées

Introduction

- Quand un programme interagit avec son environnement
  - Il est menacé
  - Une attaque peut avoir lieu sur chaque point d'interaction
  - Les accès disques, les paramètres d'exécution, les sockets, la mémoire sont des points d'interactions
  - Selon les points d'interactions des attaques distantes et/ou locales sont possibles
  - Les programmes « suid » sont des cibles de choix !
- Les services externes/publiques sont très exposés !
- Rappel: http://www.ouah.org, http://www.phrack.org
- Les sites WEB dynamiques sont les plus exposés
  - Attaques par des scripts php, CGI
  - Attaques par injection ➔ Attaque Virus HTML
  - Cross scripting

Des attaques variées

- On constate des classes d'attaques sur les services. Elles sont « génériques »
  - Le « buffer overflow » est la plus connue
  - L'attaque par chaîne de format
  - Valeurs hors normes sur les services [local et distant]
  - Par virus, cheval troie, ...
- Les attaques génériques sont une première étape vers le compromission de la machine. Elles permettent
  - l'accès à la machine
  - l'installation des modules noyaux, de backdoor, de rootkit
  - l'utilisation de la machine à des fins malveillantes
- D'autres services peuvent subir des compromissions spécifiques pour détourner leur utilisation
  - DNS cache poisoning
  - Bounce FTP server attack

Une journée ordinaire de M^r ROOT

- **Voici les avis publié le 10 janvier 2006 :**

[cert-renater] [AVIS CERTR : OpenBSD: Patch fixes suid /dev/fd access check]
[cert-renater] [AVIS CERTR : Mandriva Linux: Updated xpdf packages fix several
[cert-renater] [AVIS CERTR : RedHat: httpd security update]
[cert-renater] [AVIS CERTR : Debian: New smstools packages fix format string v
[cert-renater] [AVIS CERTR : Ubuntu: sudo vulnerability]
[cert-renater] [AVIS CERTR : Gentoo: VMware Workstation Vulnerability in NAT n
[cert-renater] [AVIS CERTR : SCO: LibXpm Integer Overflow Vulnerability]
[cert-a] [AVIS CERTA : Vulnérabilité dans ClamAV]
[cert-a] [AVIS CERTA : Vulnérabilité du module mod ssl dans Apache 2]
[cert-a] [AVIS CERTA : Multiples vulnérabilités dans postgreSQL]
[cert-a] [AVIS CERTA : Vulnérabilité dans auth ldap pour Apache]

Les méthodes de protections

- Des protections contre ces attaques existent
  - Des outils de contrôle dynamiques
    - libsafe, lids, systrace, TCPwrapper
  - Des outils d'isolation et de virtualisation
    - vmware, vserver, uml, chroot
  - Des audits automatiques de code
    - Algorithmes d'analyse du code source
    - Vérification des « include » dans les scripts, échappement des chaînes de caractères
  - Limiter services et contrôler les services
    - Eliminer les mots de passes en clair, PKI
- Un système 100% sûr n'existant pas, il faut prévoir
  - Des outils d'audit
    - Forensics Analysis (log, disque dur, …)
    - Détection des Root kit
    - Vérification de l'intégrité des fichiers
  - Des check-lists de sécurité à vérifier régulièrement
  - Des pots de miels

Plan de cours

Attaques génériques de services

Attaques par execv

- Attaque très simple basée sur un appel exec mal protégé
  - Des programmeurs imprudents : Pas de chemin absolu !
  - `execve (« ls »,NULL,NULL)`
- Il suffit de faire en sorte que le programme exécute son propre « ls »
  - Un exec modifié c'est mieux !
- Si le programme appartient à « root » et a le bit suid à 1, le « ls » sera exécuté avec les droits « root »
- Exemple

```
echo « xterm & ls » > /tmp/ls
EXPORT PATH=/tmp:$PATH
./programme_mal_ecrit
```

- Solution :
  - Faire un setuid, setgid lors d'appel comme system, exec, ...
  - Fixer des chemins absolus
  - Purger l'environnement (PATH !!!)
  - Eviter les programmes externes ➔ il y a toujours un autre moyen

Attaques génériques de services

Attaques par débordement

- Terme anglais : « buffer overflow » ou « bof »
- Principes
  - Etre capable d'exécuter un code arbitraire à travers un autre programme
    - Le code arbitraire est souvent un « code SHELL »
    - Le programme doit être « suid » pour être intéressant
- But
  - Obtenir un accès maximum sur un système
    - « Root » sur linux/unix; « Administrator » sur XP
- Conséquences
  - Le code « shell » peut être utilisé pour modifier n'importe quoi sur la machine
    - Installer des traps, ouvrir un terminal « root »
    - Télécharger des rootkits et les exécuter

Attaques par débordement

- C'est un problème majeur
  - Il n'est pas ENCORE démodé !! Mais en perte légère de vitesse.
  - Des nombreux (tous ?) services ont déjà été affectés
    - HTTP (apache, iis), ftp (proftpd, ftpd)
    - imap (TOUS) , smtp (sendmail), syslog
  - De nombreuses commandes ont déjà été affectées
    - Tous les systèmes UNIX/linux/Windows
    - Eject, rsync, rdist, cvs, mount
  - Laisse peu de traces (arrêt du service)
- Les serveurs Web sont particulièrement exposés
  - De nombreux scripts CGI, pages php/asp/jsp
  - Ils sont mal conçus/pensés par des personnes non spécialistes
- **Essentiellement du à de mauvaises habitudes de programmation**
  - **Pourrait être facilement évité !**

Attaques par débordement - Solutions

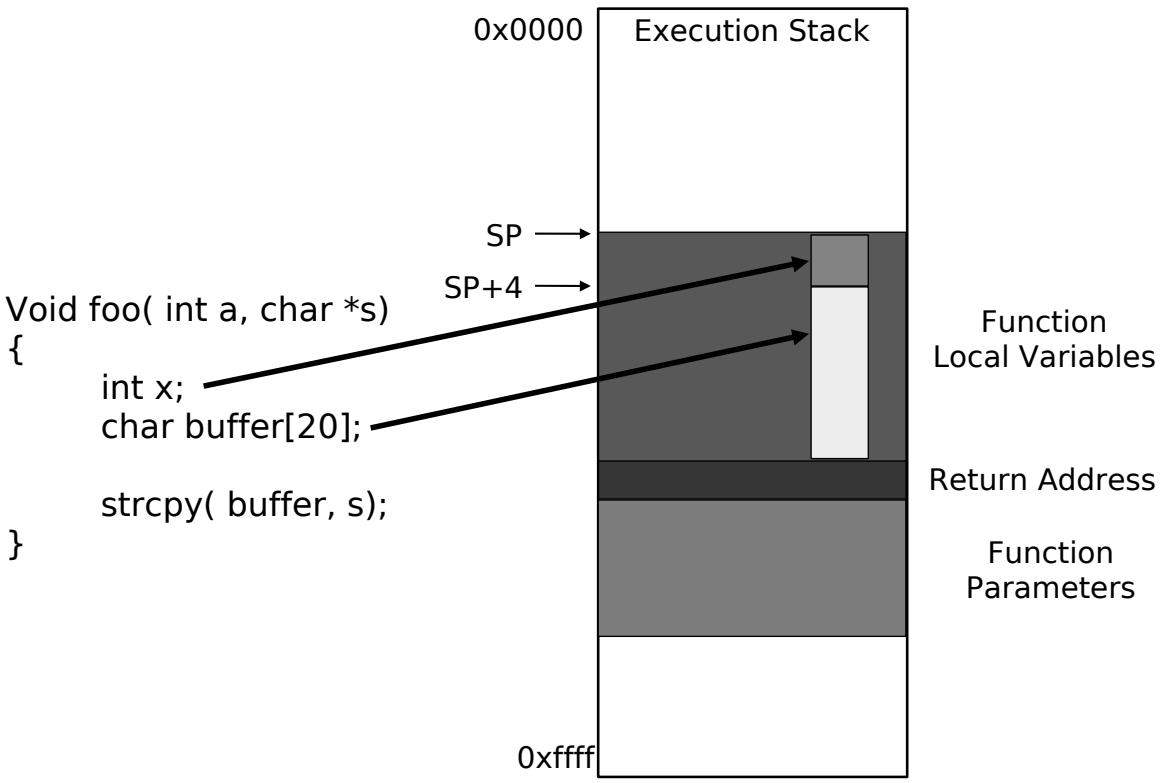
- Une bonne programmation c'est :
  - Eviter les manipulations sans contrôle
    - **BANNIR strcpy(), get(), strcat()**
    - **ATTENTION AUX sprintf(), vsprintf(), scanf()**
    - Utilisation de strcpy() ➔ strncpy()
    - Utilisation de gets() ➔ fgets()
  - Utiliser des librairies spéciales
    - **Vérification systématiques des bornes**
  - Utiliser des versions spéciales de malloc()
    - Vérification des allocations et des bornes
  - Utiliser des librairies de debug pour vérifier le programme
    - Dbmalloc, checker, LeakTracer, …
  - Purger l'environnement d'exécution !
- Modification du noyau
  - Création de nouvelles limitations sur le processus
  - Interdiction d'invocation de fork() et/ou d'exec()

Attaques par débordement

- Concerne les programmes s'exécutant avec les droits « root » (user/group) et avec le bit suid
  - Exécuter par des utilisateurs normaux mais devant conserver les droits « root »
- Sous MacOS X, on peut corrompre des programmes ayant comme propriétaire « admin » ou « root »
- Prologue ➔ trouver ces programmes

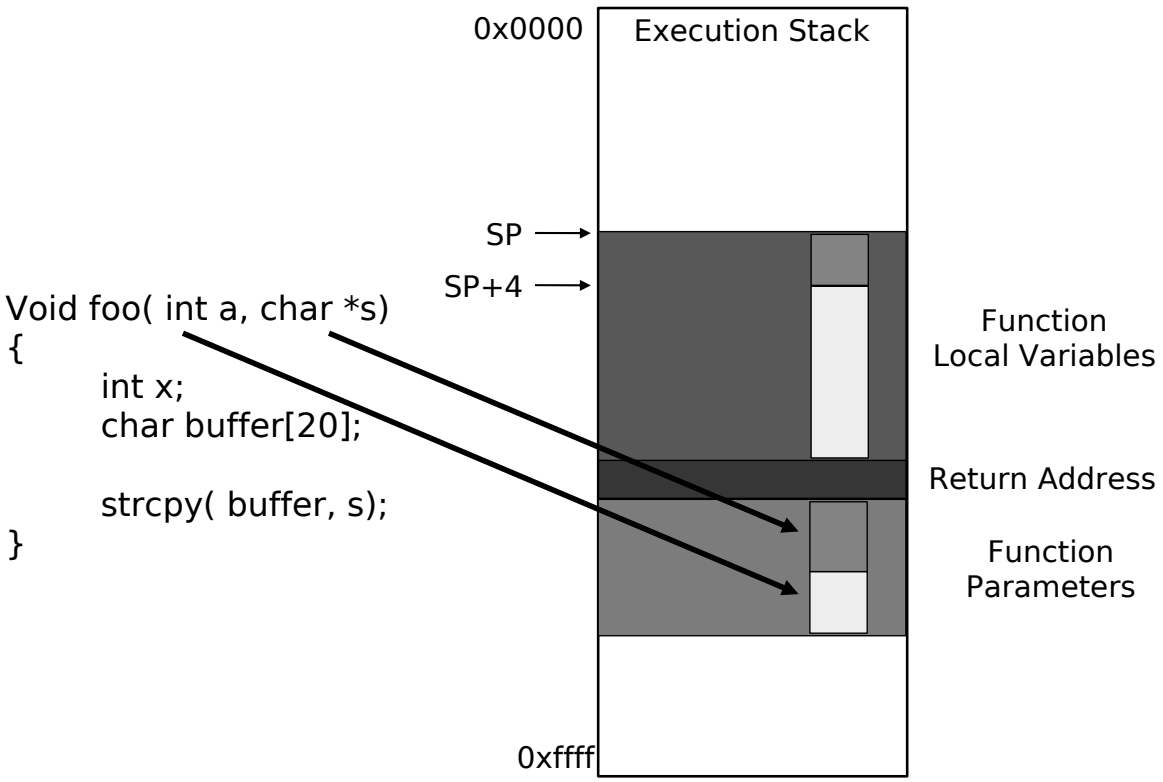
```
find /bin -user root -perm +a=s > suid.lst
find /sbin -user root -perm +a=s >> suid.lst
find /usr/bin -user root -perm +a=s >> suid.lst
find /etc -user root -perm +a=s >> suid.lst
find /var -user root -perm +a=s >> suid.lst
echo "see in suid.lst for the list..."
```


Attaques par débordement de Pile : principes



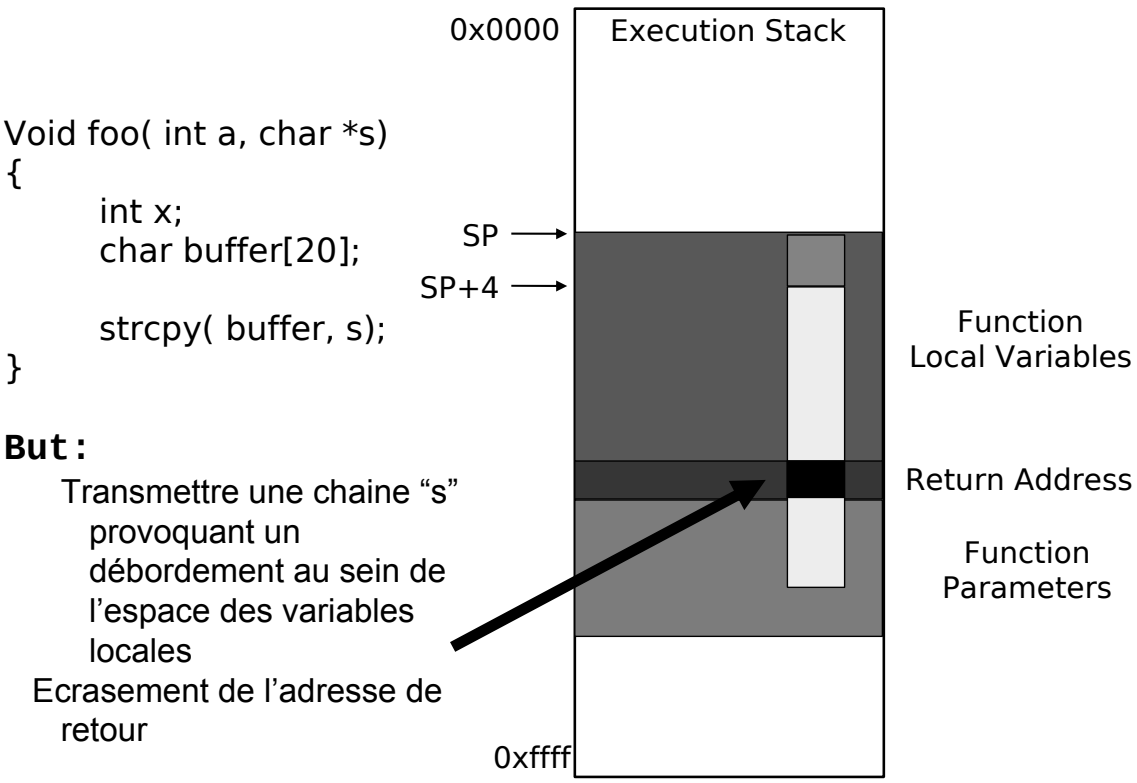
Attaques par débordement de Pile : principes

Attaques génériques de services



Attaques par débordement de Pile : principes

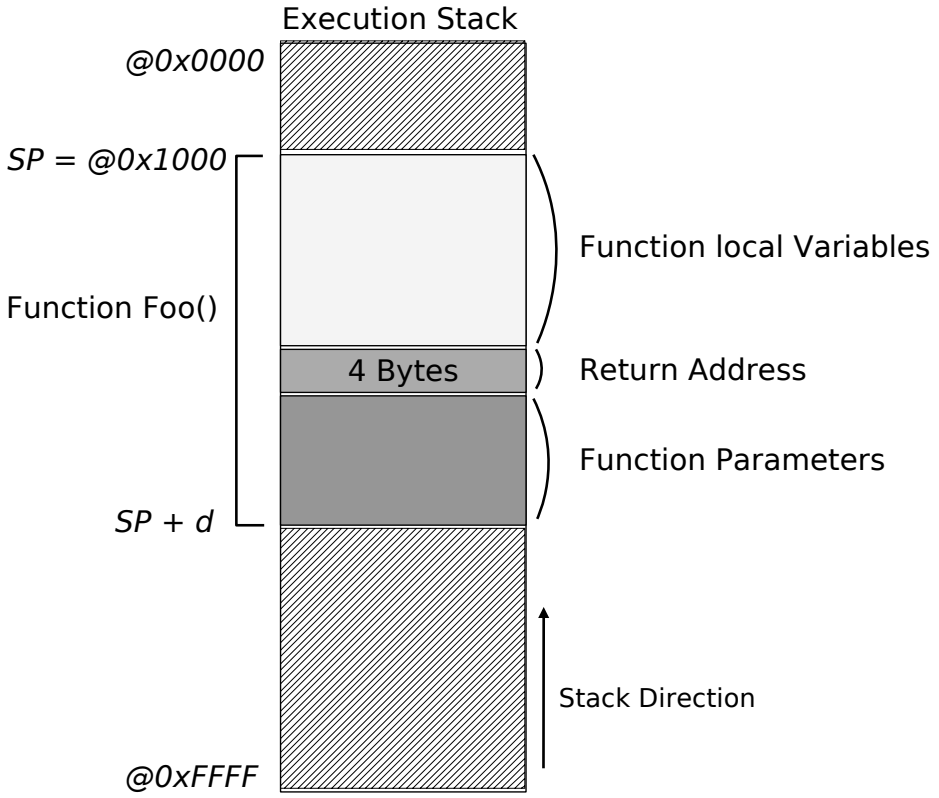
Attaques génériques de services



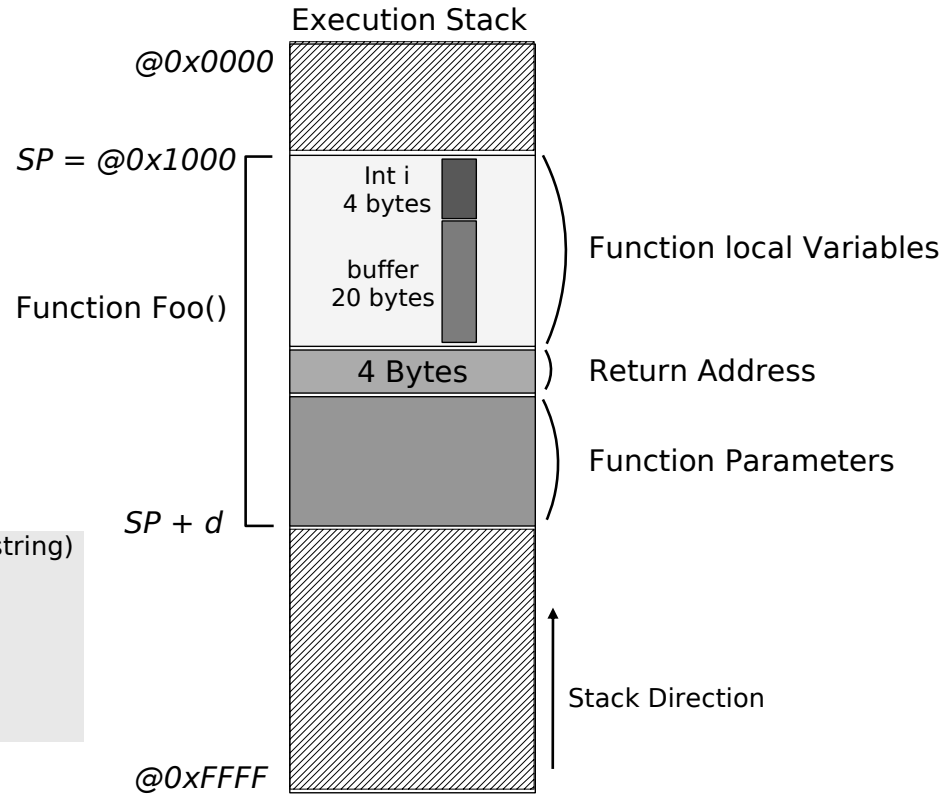
But :

- Transmettre une chaine “s” provoquant un débordement au sein de l’espace des variables locales
- Ecrasement de l’adresse de retour

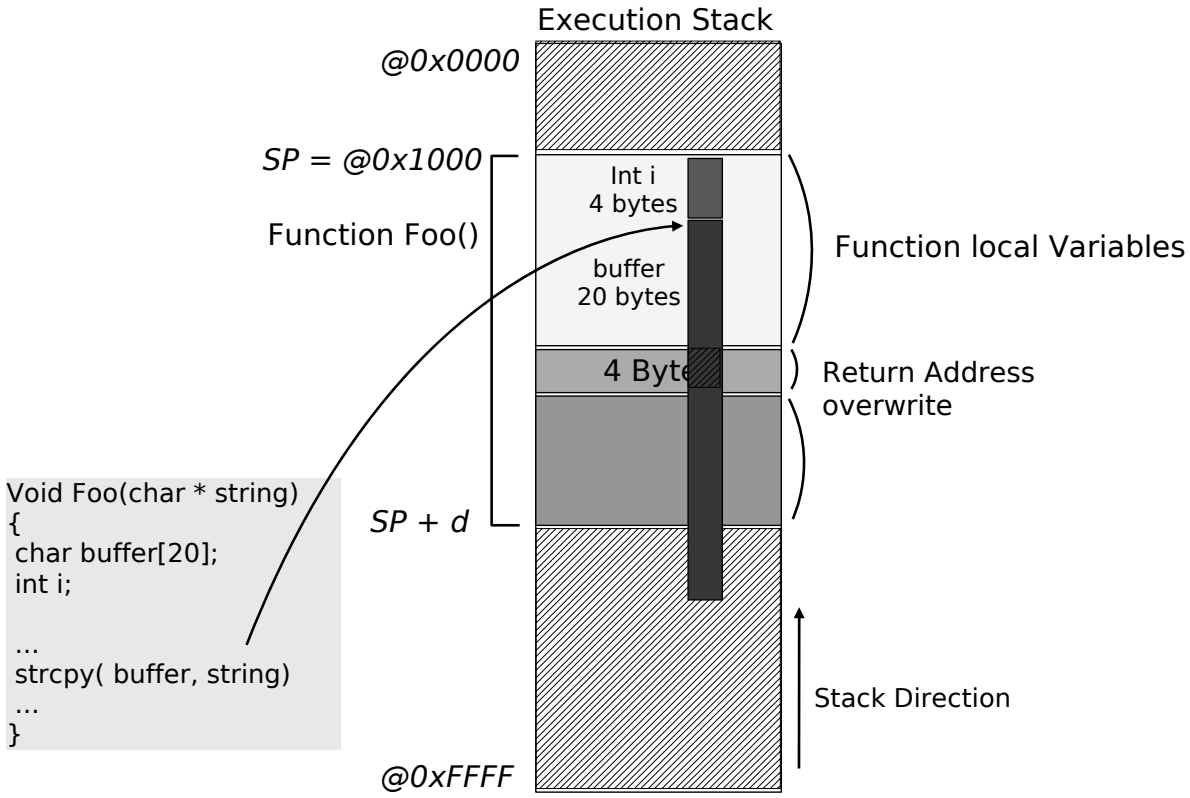
Attaques par débordement de Pile : déroulement sur x86



Attaques par débordement de Pile : déroulement sur x86



Attaques par débordement de Pile : déroulement sur x86




```
Void Foo(char * string)
{
 char buffer[20];
 int i;

 ...
 strcpy( buffer, string)
 ...
}
```


Attaques par débordement de Pile : déroulement sur x86

Créer un chaîne “string” (**~200 o.**) telle que :

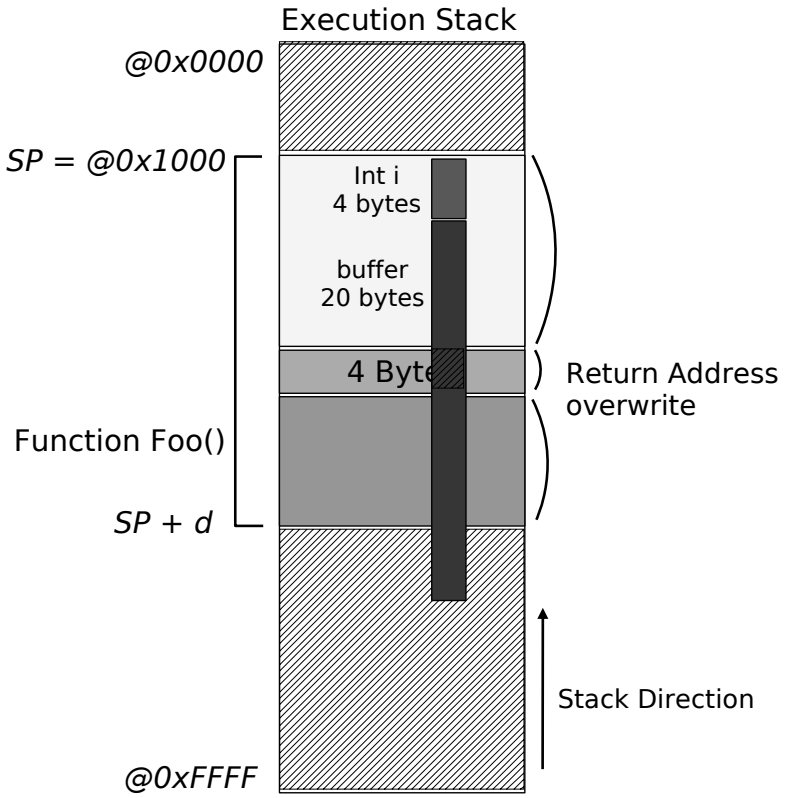
a) L’adresse de retour pointe sur un code forgé par nous
b) Contienne un code à exécuter

La première difficulté est de déterminé où est stockée l’adresse de retour.

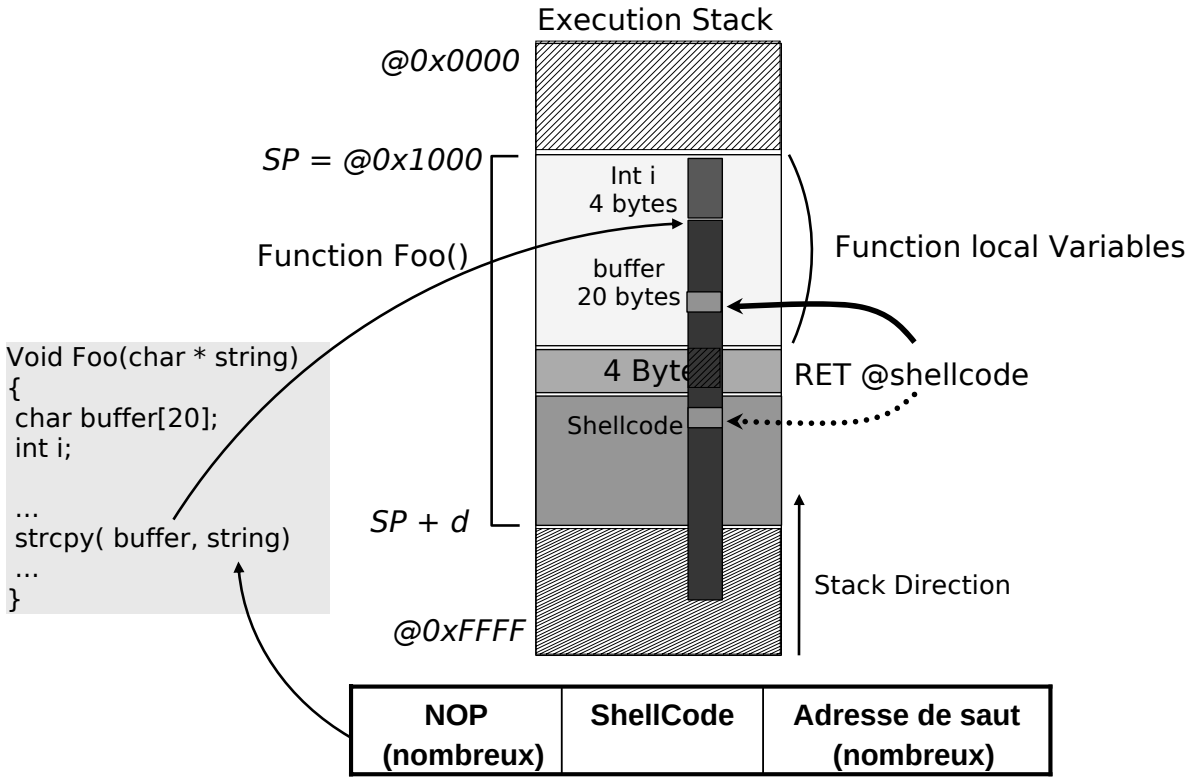
C’est beaucoup plus facile si on a accès au code source.

La deuxième difficulté est de déterminer l’adresse où sera stocké le code injecté.

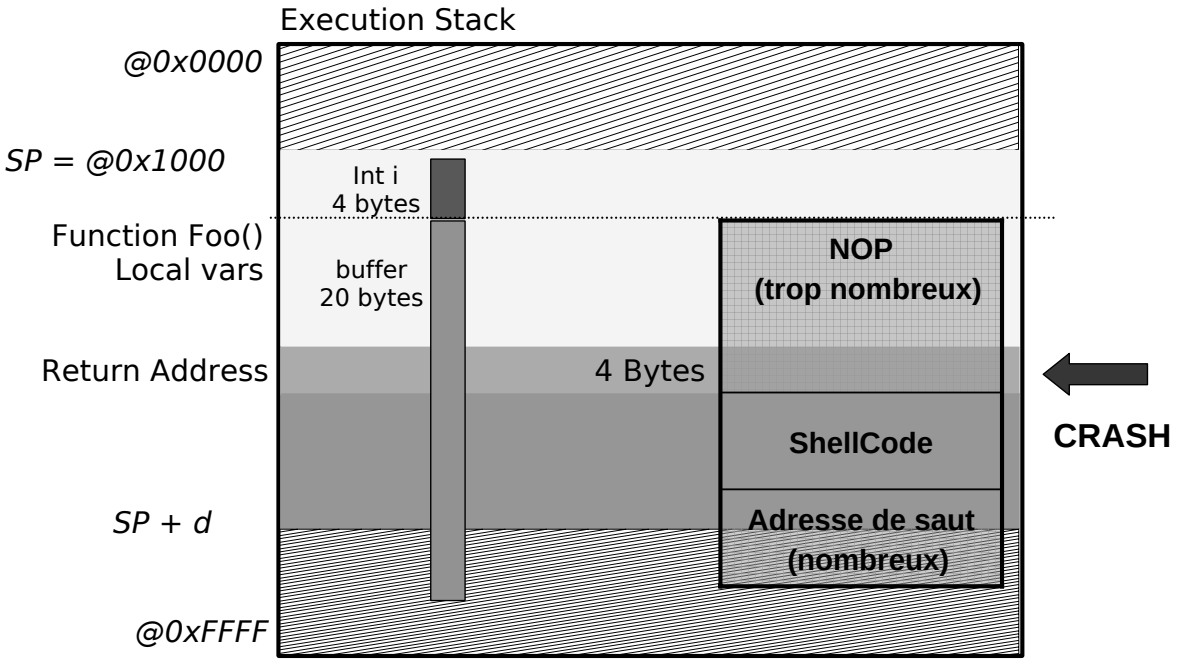
On peut se donner de la marge en remplissant d’instructions sans effet (NOP).



Attaques par débordement de Pile : déroulement sur x86

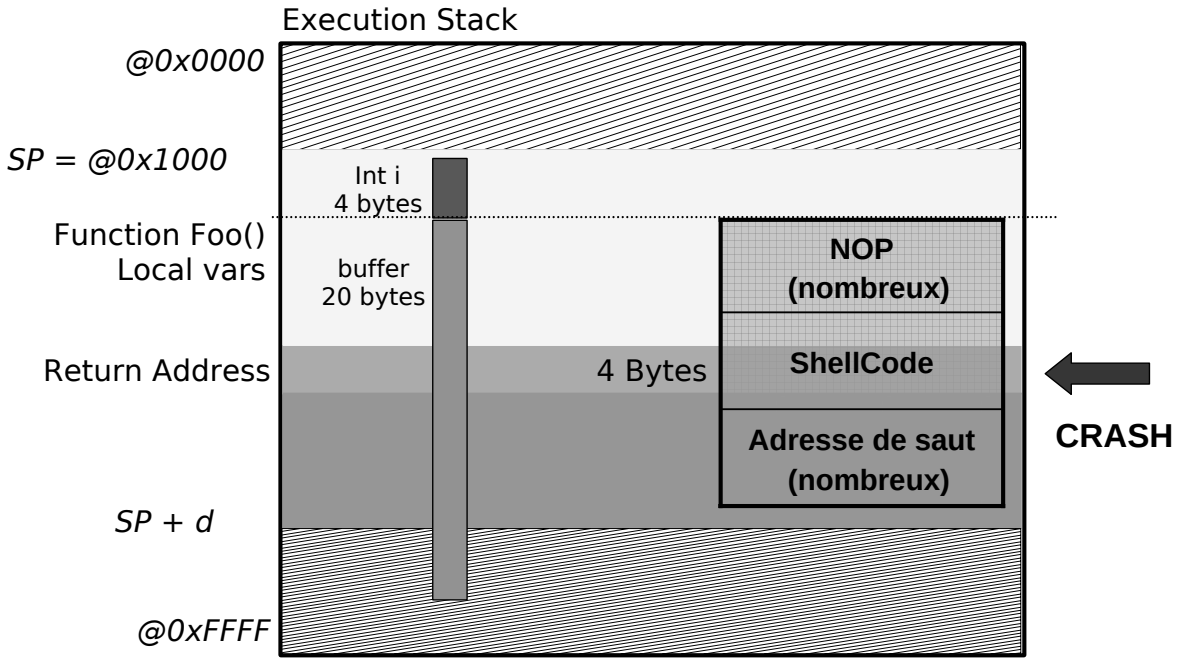


Attaques par débordement de Pile : déroulement sur x86



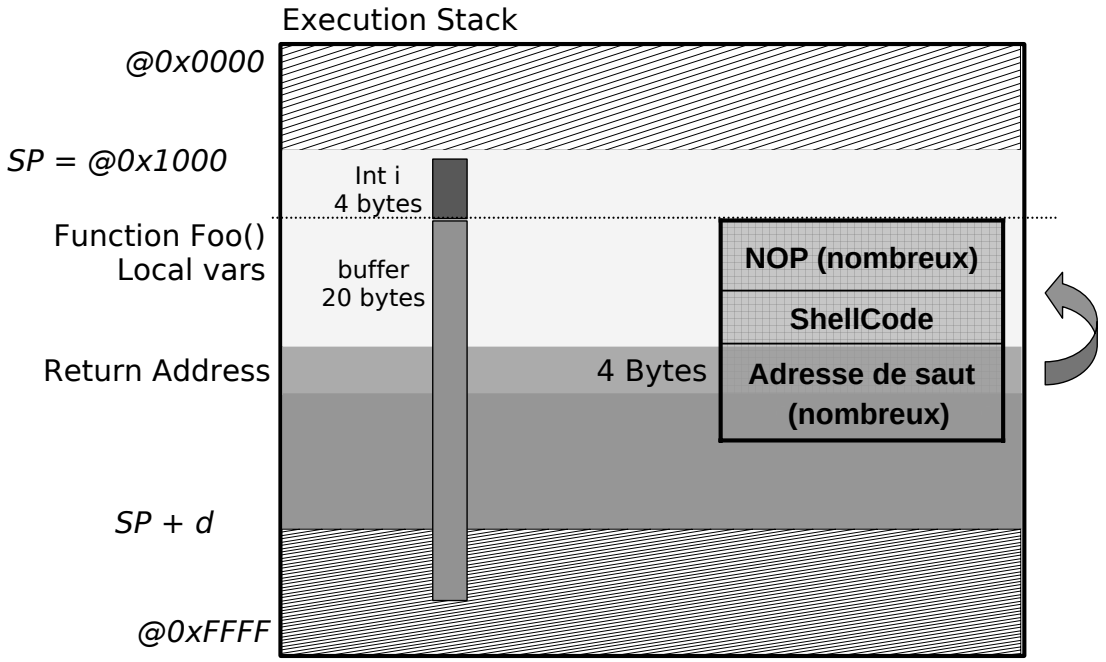
Attaques par débordement de Pile : déroulement sur x86

Attaques génériques de services



Attaques par débordement de Pile : déroulement sur x86

Attaques génériques de services



Attaques génériques de services

Attaques par débordement de Pile : le ShellCode

- On désigne sous le terme de shellcode un code permettant la création d'un shell.
  - En unix/linux, il s'agit de récupérer le code assembleur généré par l'appel à une commande exec sur un shell (sh/bash/csh) et exit (en cas d'échec)
  - On écrit le code C

```
#include <stdio.h>
void main() {
  char *name[2]; name[0] = "/bin/sh"; name[1] = NULL;
  execve(name[0], name, NULL);
}
```

  - On obtient le code assembleur par

```
gcc -o shellcode -ggdb -static shellcode.c
gdb shellcode
```

    Ou `gcc -S -o example1.s example1.c`
- Le code ne doit contenir aucune adresse absolue (JUMP absolu)
- Le shellcode sera chargé dans l'application par une chaîne spécialement forgée
  - On doit éliminer les caractères 0x00 (NULL) dans le code assembleur !!!
  - Il existe des shellcodes qui résistent au toupper() et tolower()
- Cela fonctionne car la pile (stack) est en lecture/écriture

Attaques génériques de services

Attaques par débordement de Pile : exemple de ShellCode

System Call setuid()

Arguments of execve()

System Call execve()

```
xor %eax,%eax   # remplace mov %eax, $00
xor %ebx, %ebx
xor %ecx,%ecx
mov $0x17, %eax          # interruption setuid(0, 0)
int $0x80
xor %eax,%eax
xor %edx,%edx
push %ebx                #sauver ebx
push $0x68732f6e
push $0x69622f2f
mov %esp, %ebx
lea (%esp, 1), %edx      # LoadEffAdr.  "/bin/sh"
push %eax                # sauver eax, ebx
push %ebx
lea (%esp, 1), %ecx
mov $0xb, %eax           #interruption syst execve
int $0x80
xor %eax, %eax           #  en cas d'échec
mov $0x1, %eax           # interruption syst exit()
int $0x80
```

*int execve(const char *filename, char *const argv [], char *const envp[]);*

Attaques génériques de services

Attaques par débordement sur le TAS

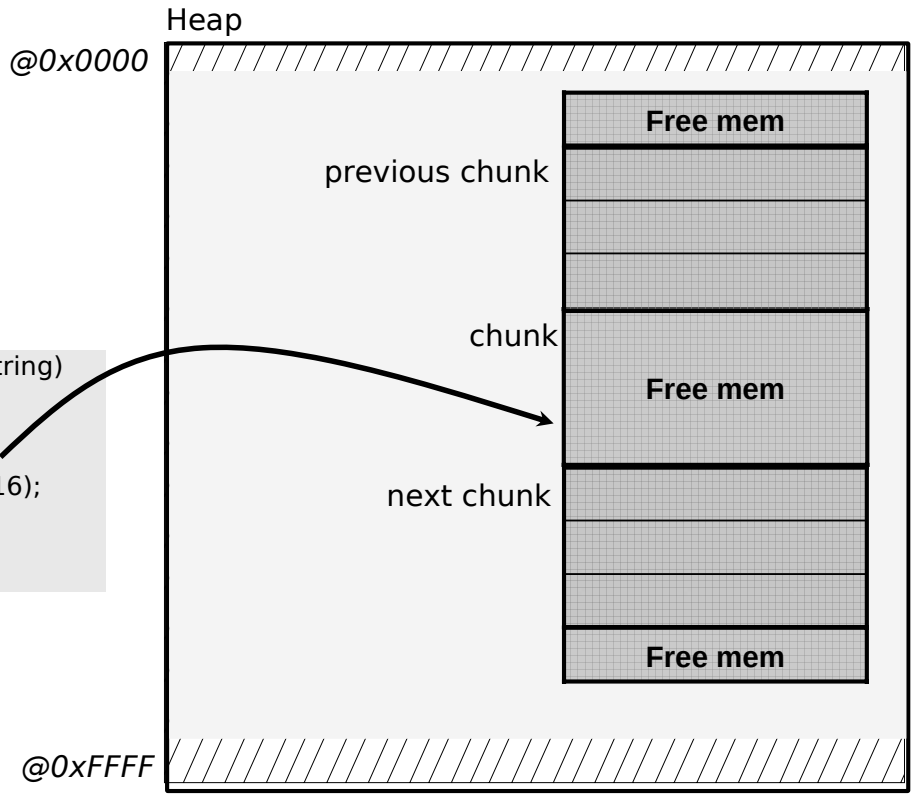
- De plus en plus de développeur d'OS fournissent des patches pour les bofs
  - Empêche l'exécution de code dans la pile
  - ➔ développement des « Heap Overflow » (« hof »)
- C'est un « overflow » qui s'attaque aux données allouées dynamiquement (TAS)
- Le TAS sert à stocker les données dynamiques manipulées
  - Bloc de données « malloc »
  - Les images d'un viewer, les fichiers MP3 d'un lecteur de musique
- Un Hof est plus complexe à mettre en place qu'un bof
  - Nécessite une connaissance approfondie du système
    - Comment sont allouer les blocs de mémoire ?
- Ce type d'attaque est essentiellement basée sur la fabrication de fichiers de données malformés
  - L'attaquant doit provoquer le traitement du fichier
  - L'attaquant doit attendre le traitement du fichier
- Exemple
  - Faille GDI+, Faille WMF, Faille WMV

Attaques génériques de services

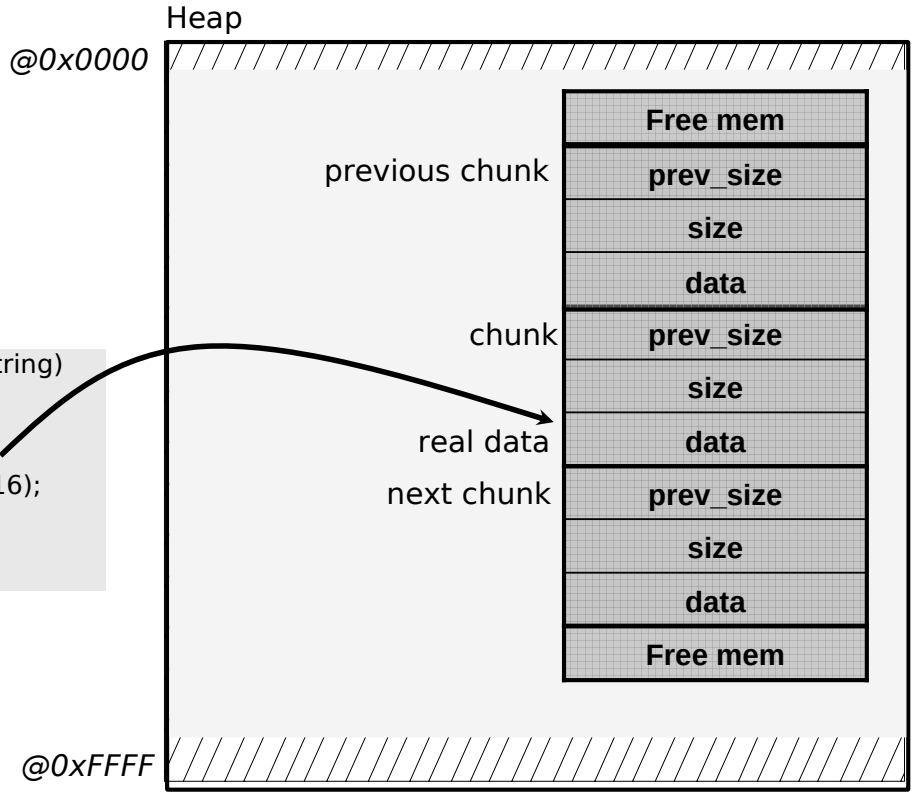
Attaques par débordement sur le TAS

- Les données du TAS peuvent contenir des pointeurs
  - Ils peuvent être corrompus
  - Les données peuvent être corrompues
- Exemple :
  - Corruption d'une structure utilisée pour un setuid
  - Corruption d'une chaîne de nom de fichier
  - Corruption d'un pointeur de fonction
    - Pour pointer sur une fonction libc
    - Pour pointer sur un ShellCode contenu sur le tas, la pile

Attaques par débordement sur le TAS : GNU C

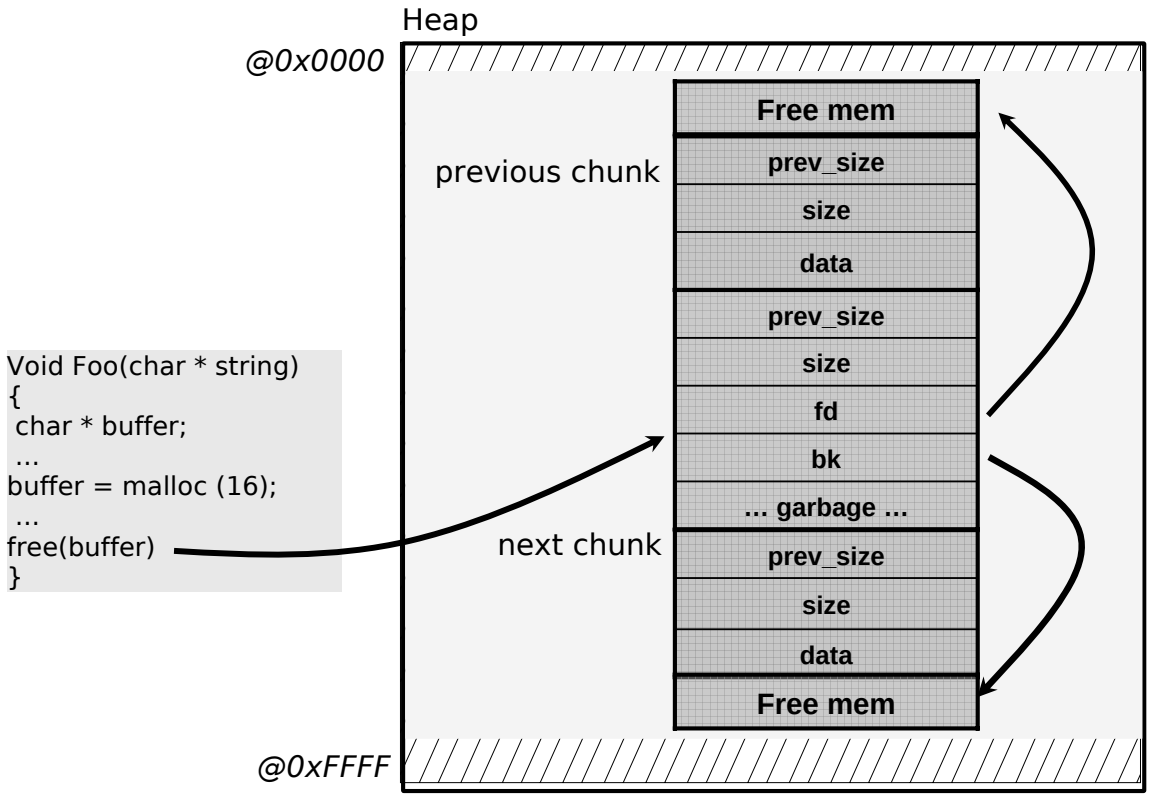


Attaques par débordement sur le TAS : GNU C



Attaques par débordement sur le TAS : GNU C

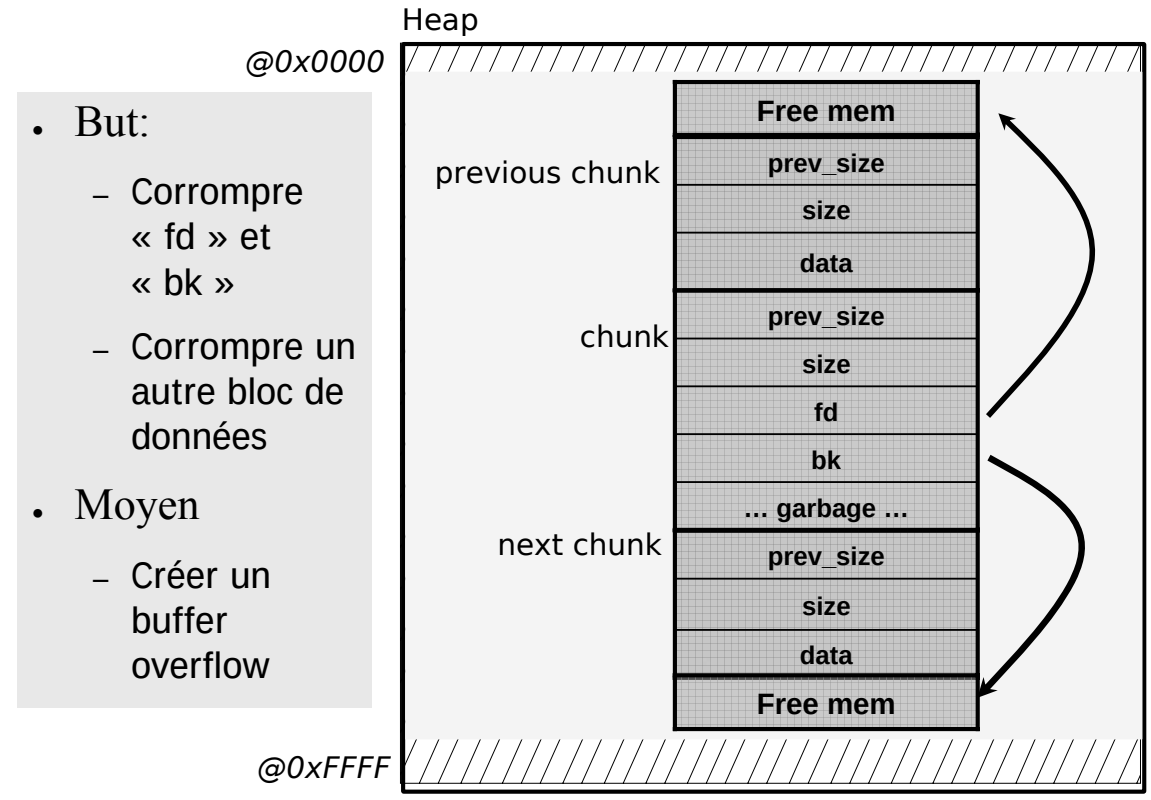
Attaques génériques de services



Attaques par débordement sur le TAS : GNU C

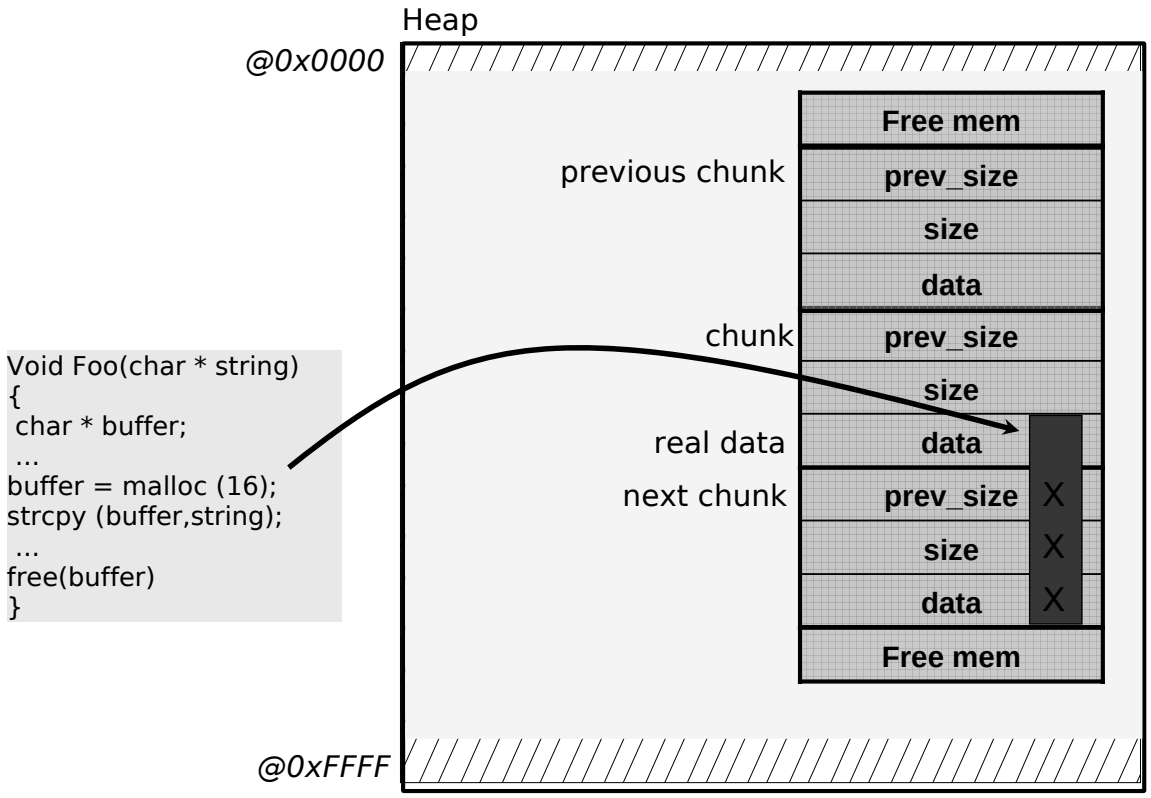
Attaques génériques de services

- But:
  - Corrompre « fd » et « bk »
  - Corrompre un autre bloc de données
- Moyen
  - Créer un buffer overflow



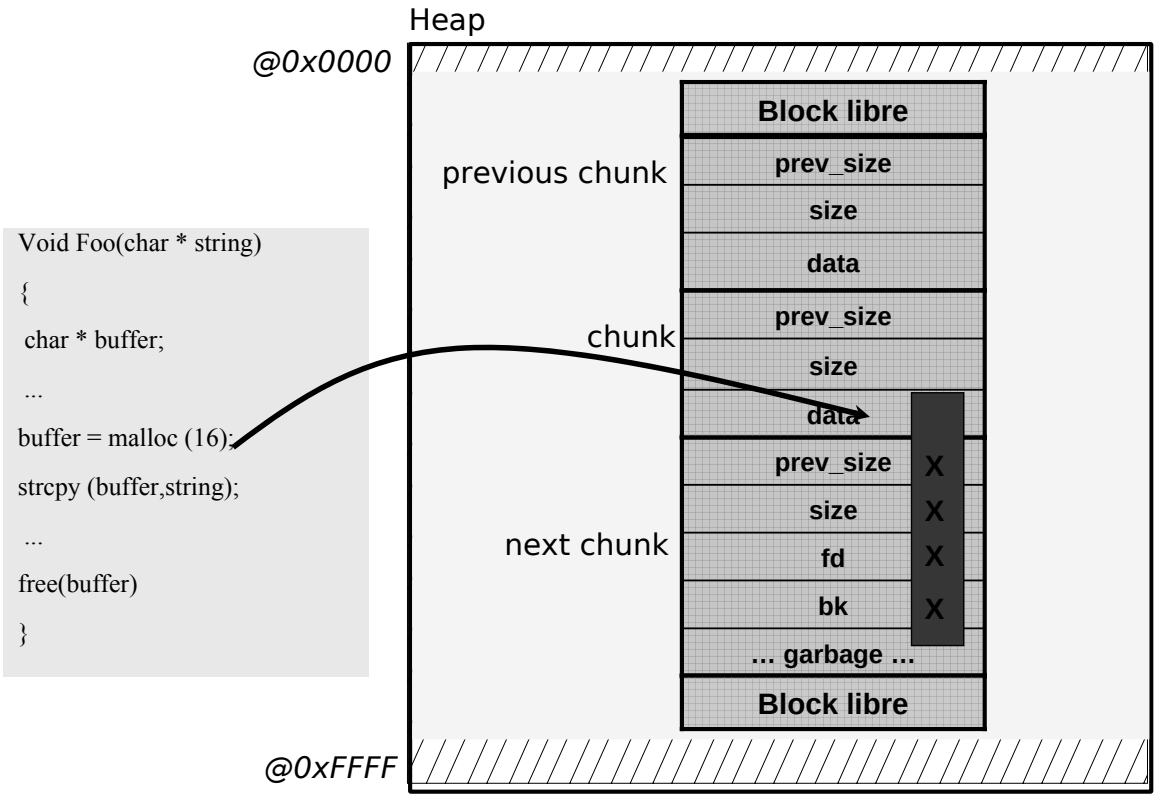
Attaques par débordement sur le TAS : GNU C

Attaques génériques de services



Attaques par débordement sur le TAS : GNU C

Attaques génériques de services



Attaques par débordement sur un espace réduis

- En cas d'espace local trop faible pour stocker le code
  - Il est possible de stocker le code ailleurs en mémoire
    - En particulier dans l'environnement du processus
    - Le processus hérite de l'environnement
  - De faire pointer l'adresse de retour sur ce code
- Il existe de nombreuses variantes
  - ∃ des variantes moins détectables par les IDS !
    - Voir les codes polymorphes (auto-mutable)
- Le ShellCode peut aussi servir à (quelques idées en l'air):
  - Copier /bin/sh en /tmp/monshell et faire un suid
  - Ouvrir un shell connecté à une socket réseau

Attaques par corruption de « chaîne de texte»

- « String Format attack »
- Caractéristiques
  - Type d'attaque découverte en 2000 (récent)
  - Utilise la pile ou le tas qui sont R/W
  - Reproduire l'effet des « bof » (dépassement)
  - Exploitation des formats de sprintf, printf, et consorts
    - En particulier utilisation du format « %n »
- Il s'agit la aussi de mauvaises habitudes de programmation
  - Méthode incorrecte: printf(string);
    - Un exemple très simple d'instruction dangereuse
    - Si string = « %s%s%s%s%s », 99% de chance de crash
  - Méthode correcte: printf("%s", string);

Attaques par corruption de « chaîne de texte»

- Il existe des exemples plus complexes

```
{
char outbuf[512];
char buffer[512];
sprintf (buffer, "ERR Wrong command: %400s", user);
……
sprintf (outbuf, buffer);
}
```

- Code apparemment anodin
  - user="%497d\x3c\xd3\xff\xbf<nops><shellcode>"
  - Lors du premier sprintf ➔ copie normale dans buffer
  - Lors du second sprintf ➔ le contenu de buffer est interprété
    - %497d ➔ prend la première valeur de la pile et génère une chaîne
    - La taille : 497 caractères + longueur(« ERR Wrong command : »)
    - C'est supérieur à 512 ➔ dépassement de capacité avec le formatage de la chaîne
- Le but ultime ➔ reproduire un « bof » avec le formatage printf
  - Cf. exemple précédent

Attaques par fichier interposé

- Exploitation de droits mal définis sur des fichiers
- Peut être couplée avec une attaque de type « race condition »
- Le meilleur exemple : « /tmp »
  - Un grand nombre de programme suid y stocke des données temporaires
  - N'importe qui peut y écrire
  - On peut créer des liens vers des fichiers « root »
- Le but
  - utiliser une application ayant les droits root pour modifier les fichiers root via des liens fabriqués
  - Si l'application ne vérifie pas correctement la nature du fichier ➔ problème
- Exemple (désuet) :
  - mktemp crée des fichiers temporaires unique mais pas imprédictibles
  - Faire un lien de /tmp/fichier_temporaire vers /.rhosts
  - Si l'application ajoute une ligne « + + » au fichier, c'est gagné
  - Après il ne reste plus qu'à faire « rsh localhost –l root »

Attaques génériques de services

Attaques par « course de vitesse »

- Principe :
  - Système multitâches (de nombreux processus)
  - Profiter du système à un instant où il est vulnérable
    - Exploit moins difficile à réaliser que l'attaque par débordement
    - « Il suffit » de réunir les conditions pour que cela arrive
  - Changer les conditions d'exécutions pendant l'exécution d'un programme
    - Manipuler les flux de données, insérer des données dans le programme
  - Mauvaises suppositions sur des liens de causalité
    - Ex sur les fichiers ➔ vérification des droits, ouverture, lecture
  - Comme d'habitude ➔ les programmes « suid » sont les cibles !
- But :
  - Obtenir les droits « root »
  - Accéder aux informations manipulées par le service
  - Empêcher le fonctionnement correct du service
  - Bloquer le travail des autres utilisateurs

Attaques génériques de services

Attaques par « course de vitesse »

- TOCTTOU =« Time Of Check To Time Of Use »
  - Exploitation de la latence une mesure et son utilisation
    - Applicable aux fichiers (en particulier aux fichiers sur /tmp)
    - Les scripts de par leur lenteur (langage interprété) y sont très sensibles
  - Obtention des informations par logiciels simples et analyse des logs produits
    - strace (appels systèmes), ltrace (appels dynamiques), nm (liste des symboles externes)

Attaques par « course de vitesse » : exemple « passwd »

- Erreur sur HP/UX et SunOS, avec « passwd »
  - Programme Suid puisque doit modifier /etc/passwd
- Déroulement normal des opérations
  1. Ouverture et lecture du fichier « /etc/passwd » ➔ obtention de l'entrée utilisateur
  2. Création et ouverture d'un fichier « ptmp » dans le répertoire du fichier « passwd »
  3. Ouvrir à nouveau le fichier « /etc/passwd » et copier le contenu dans « ptmp » (en mettant à jour l'utilisateur).
  4. Fermer le fichier « passwd », « ptmp »
  5. Renommer (pour remplacer) le fichier « passwd » par « ptmp »
- Le programme « passwd » peut travailler sur un fichier spécifié

Attaques par « course de vitesse » : exemple « passwd »

- Créer un fichier ayant un format compréhensible par plusieurs services (« passwd » et « rlogin »)

  localhost account :::::
- Cette ligne est valide pour les fichiers « rhost » et « passwd »
- Exécution de passwd (programme suid) dont on va contourner le fonctionnement normal
- Comment ?
  - En s'insérant entre les manipulations de « passwd »

Attaques par « course de vitesse » : exemple « passwd »

A. On utilise un répertoire pointé par un lien symbolique.
Lien ~attaquant/link ➔ ~attaquant/tmprep
Création du fichier « ~attaquant/link/.rhosts » avec « localhost acount ::::: »

1. Ouverture et lecture du fichier « ~attaquant/link/.rhosts »
➔ obtention de l'entrée utilisateur
   A. L'attaquant change le lien symbolique : ~attaquant/link ➔ ~cible
2. Création et ouverture d'un fichier « ~attaquant/link/ptmp » dans le répertoire du fichier passé en paramètre « ~attaquant/link »
   A. « passwd » est root, il créer le fichier « ptmp » chez la cible
   MAJ du lien symbolique : ~attaquant/link ➔ ~attaquant/tmprep
3. Ouvrir à nouveau le fichier « ~attaquant/link/.rhosts » et copier le contenu dans « ptmp » (en mettant à jour l'utilisateur).
   A. La ligne « localhost account ::::: » est copiée
   MAJ du lien symbolique : ~attaquant/link ➔ ~cible
4. Fermer le fichier « passwd », « ptmp »
   A. La fermeture n'utilise pas le lien symbolique mais travaille sur le descripteur
5. Renommer (pour remplacer) le fichier
« ~attaquant/link/.rhosts » par le fichier« ptmp »
   A. La copie replace le fichier « .rhost » du répertoire cible par le ptmp

Attaques par « course de vitesse » : exemple « tmpwatch »

- « tmpwatch »
  - Utilisé dans beaucoup de distribution linux
  - Purge régulièrement les /tmp des fichiers
  - Souffre du bugs de « courses de vitesse »
  Sur les anciennes versions
- Course à l'effacement
  - « tmpwatch » ➔ lstat, vérifie la date, unlink
  - Système multitâche ➔ interruption possible entre « lstat » et « unlink »
- 1ère Attaque possible
  - On enlève le fichier après le « lstat » et on crée un lien avant le « unlink »
- 2ème Attaque possible
  - Les fichiers « mktemp » sont uniques mais pas imprédictibles
  - « tmpwatch » peut être maintenu en état d'exécution (remplissage de tmp)
  - On se débrouille pour créer un fichier tmp qui sera utilisé par un programme cible
  - Les cibles privilégiées sont : browser web, mailer, ...
  - On utilise « tmpwatch » pour effacer le fichier crée et le remplacer par un autre
- Un exemple
  - « logrotate » stocke ses scripts post/pre-rotate dans des fichier « /tmp/logrotate.XXXXXX »

Attaques par « course de vitesse » : Solution au TOCTOU

- Eviter l'utilisation de fichier autant que possible (surtout dans les scripts shell)
  - Le bit « s » est interdit sur les scripts ➔ TROP DANGEREUX !
- Lors de la création d'un fichier
  - utiliser O_EXCL|O_CREAT (O_EXCL ne fonctionne pas sur NFS v1 et v2)
  - Appliquer des droits MINIMA avec open (pas après la création)
  - Ils ne doivent pas être mis dans un endroit manipulable par un attaquant
    - Eviter autant que possible les répertoires partagés (/tmp)
  - Pour les fichiers temporaires ➔ utiliser mkstemp()
    - « mktemp » ne fixe pas les droits
    - « tempnam » utilise la variable d'environnement TMP

Attaques par « course de vitesse » : Solutions au TOCTOU

- Changer les droits du processus
  - Apprendre à utiliser setuid/getuid/setreuid
- Utiliser les fonctions fchown, fstat, fchmod
  - Elles travaillent sur les handles que VOUS avez ouverts
  - Eviter chown, lstat, ... qui travaillent sur des noms
  - Ne pas utiliser « access » pour vérifier les droits
- Faites attention aux manipulations du FS pendant un parcours en récursion
- Utilisation des verrous POSIX (« fcntl »)
- Méthode de verrouillage obligatoire
  - Pour chaque lecture/écriture, on vérifie le verrou
  - Problème en cas de crash du service

Plan de cours

Introduction

Attaques génériques de services

Attaques spécifiques de services

Backdoors / Rootkits / Trojans

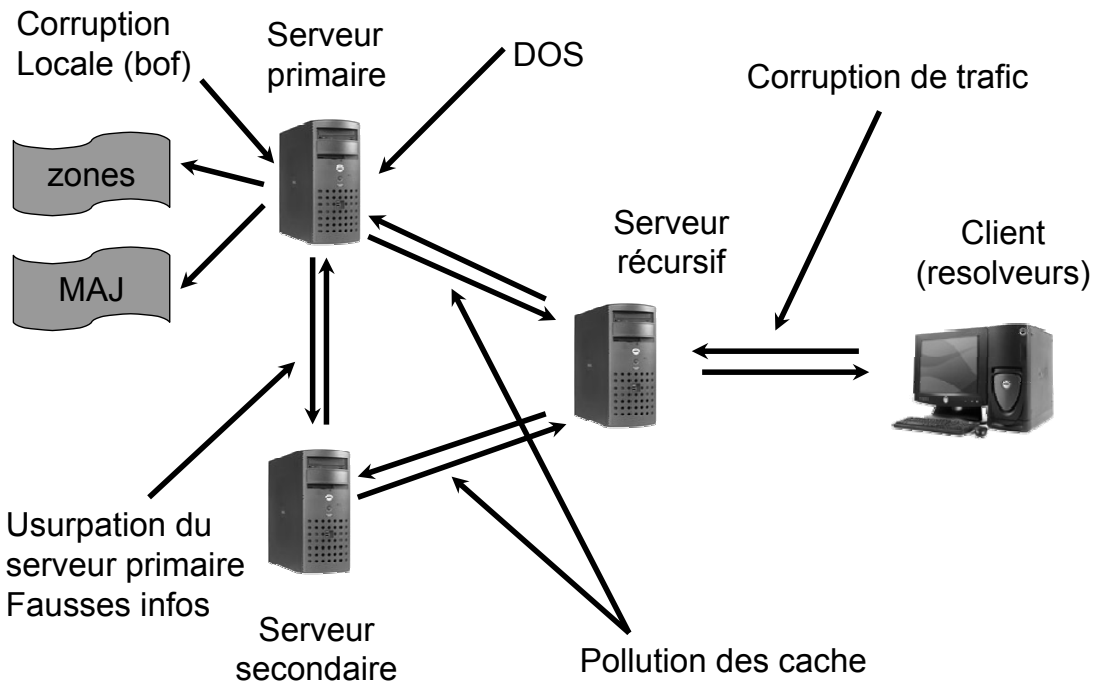
Outils de protection

Audit et check-list

Attaques sur DNS cache poisoning par spoofing

- DNS est un service de désignation !
- Le serveur DNS supporte tout les autres services.
- Une attaque du DNS permet de détourner une machine de sa cible !
  - Mise en place de faux sites WEB
  - Détournement d'informations
  - Attaques « Man In the Middle »
- Le DNS est vulnérable (RFC 3833)
  - Il ne repose quasiment que sur IP et UDP pour l'authentification de l'hôte pair

Attaques sur DNS cache poisoning



Attaques sur DNS cache poisoning par spoofing

- Le dialogue entre les serveurs DNS se passent de port 53 à port 53
- Le serveur traitent de nombreuses requêtes en UDP pour des raisons de performances
  - Il faut les distinguer
  - Pour cela, on utilise un identifiant de requête ID sur 16bits

ID	Options	Question	Réponse	divers

- Il est possible de s'insérer entre un client et son serveur DNS local
  - On sniffe la requête
  - On génère une fausse réponse avec l'ID sniffée
  - On l'envoie au client avant le serveur
  - La réponse du serveur sera ignorée

Attaques spécifiques de services

Attaques sur DNS cache poisoning par spoofing

- Si, on ne peut pas sniffer, l'attaquant peut essayer de prédire l'ID pour engendrer une attaque
- Sous Windows 95 ➔ L'ID est le nombre de req. DNS en cours !
- Bind ancienne version ➔ Nombre aléatoire puis incrémental
  - Méthode 1
    - Il suffit d'une requête sur un DNS « sniffable » pour obtenir le point de départ !
  - Méthode 2
    - L'attaquant demande une IP inexistante (ex: inconnu.domaine.com)
    - Le DNS cible fait une requête « ns1.domain.com »
    - L'attaquant génère une dizaine de réponses spoofées venant (soi-disant) de « ns1.domaine.com »
    - ID allant de 200 à 210
    - Si on obtient une réponse c'est qu'on a deviné l'ID
- Et si c'est aléatoire ?

Attaques spécifiques de services

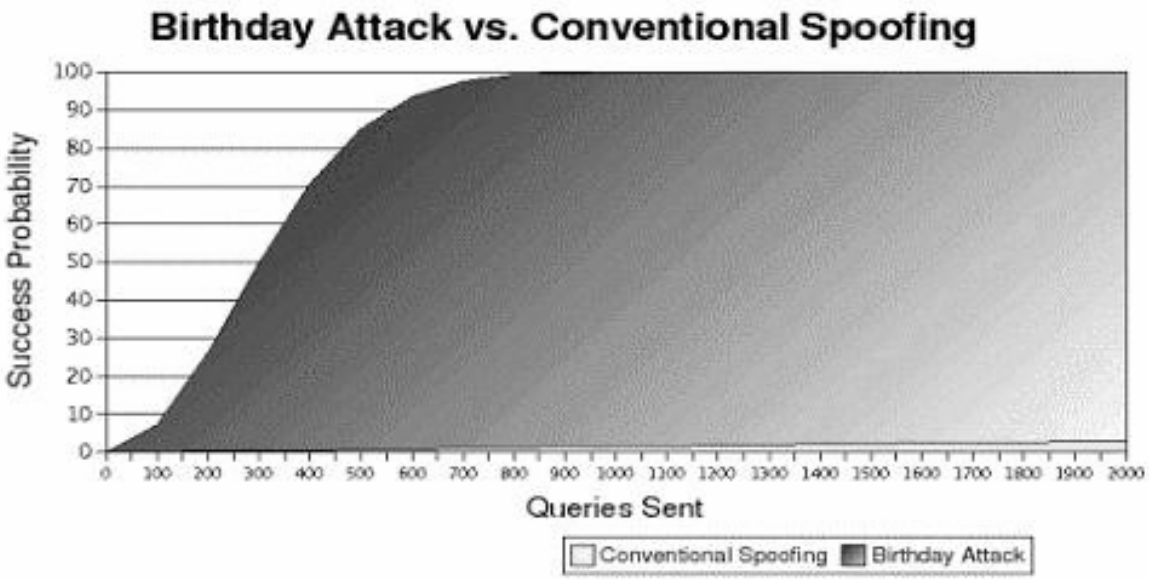
Attaques sur DNS : birthday attack

- Il est difficile de deviner l'ID d'une requête ($1/2^{16}$)
  - 1 requête et X réponses spoofées en temps très limité ➔ $p = X / 2^{16}$
- Birthday paradox attack !!!
  - Attaque basé sur un paradoxe apparent :

« sur une classe de 23 élèves ou plus, la probabilité que 2 élèves soient nés le même jour est supérieure à ½ »

- Technique appliquée au DNS
  1. Envoi de N requêtes à un serveur cache portant sur la même demande (www.exemple.com) associés à N IDs différents
  2. Transfert des N requêtes vers le serveur autoritaire du domaine exemple.com
  3. DoS sur le serveur autoritaire pour le ralentir
  4. Envoi de N réponses forgées associées à N IDs différents par l'attaquant
- Si N messages (~300) , t=le nombre de possibilités (2^{16})
  - la probabilité de succès de l'attaque $1-(1-1/t)^{N(N-1)/2}$
  - « p=.4956 » soit ~1/2

Attaques sur DNS : protection DNSsec



Attaques sur DNS : solutions

- Améliorer l'aléatoire de l'ID (espace des nombres)
- Split-Split DNS
  - FIREWALL : Interdire les IPs de votre domaine comme source sur votre point d'accès internet (paquets provenant de l'extérieur!)
  - Un serveur responsable du domaine
    - déclaré et accessible de l'extérieur
    - N'autorisé aucune requête récursive (hors domaine)
  - Un serveur cache DNS privé
    - Autoriser requêtes récursives sur votre domaine seulement
- Déploiement de DNSSec

Attaques spécifiques de services

Attaques sur DNS : solution DNSsec et TSIG

- Sécurité des données et des transactions (MAJ)
- Architecture de distribution des clefs
  - Clefs utilisées par DNSsec
  - Clefs stockées dans le DNS sécurisé utilisées pour d'autres applications (IPsec, SSH)
- Sécurité des transactions (TSIG, RFC 2845)
  - Le transfert de zones
  - Les MAJ dynamiques (DNS Dynamic Updates)
  - Le canal entre serveur récursif et client
  - Authenticité forte, intégrité, protection rejeu
  - Pas de confidentialité

Attaques spécifiques de services

Attaques sur DNS : solution DNSsec et TSIG

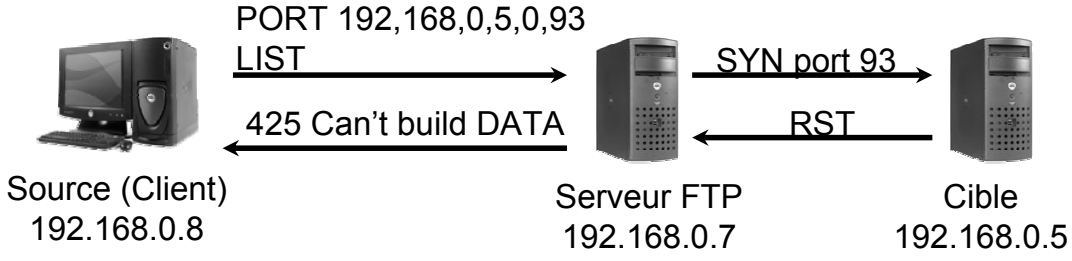
- Sécurité des données (DNSSec, RFC 4033 à 4035)
  - DNSSec assure une chaîne de confiance
  - Chaque serveur a une clef
  - Chaque serveur peut identifier de manière forte les serveurs des sous-domaines de confiance
  - Inclus un protocole de MAJ des clefs
  - Ajoute deux types d'entrées
    - SIG ➔ pour les signatures et KEY ➔pour les clefs privées

Attaques FTP : FTP servers bounce

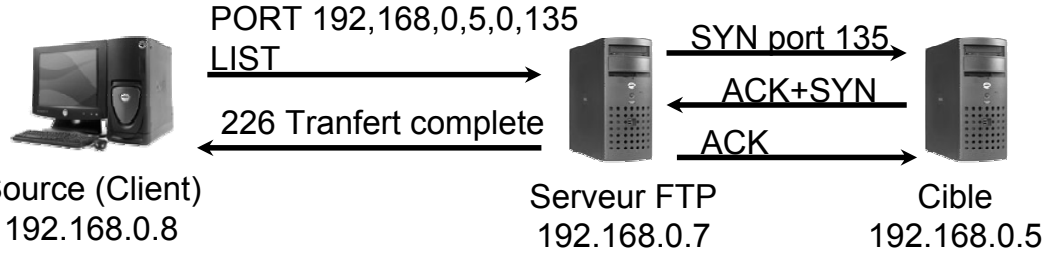
- Le protocole FTP sépare le canal de contrôle (21) et le canal de téléchargement.
  - Il est possible « d'imposer » au FTP une adresse spécifique
  - PORT aa,bb,cc,dd,pp,qq ➔ ip(aa.bb.cc.dd), port (pp,qq)
- Cette option permet de contourner les limitations de téléchargement sur les IP
  - Fichiers protégés par la loi sur l'exportation US
- Mais il permet plus !

Attaques FTP : FTP servers bounce

- La commande PORT
  - permet de tester le réseau



  - Permet d'ouvrir une connexion sur une machine



Attaques spécifiques de services

Attaques FTP : FTP servers bounce

- « nmap » inclus même ce type de scan

```
# nmap -v -b anonymous:anon@192.168.0.7 192.168.0.5

Resolved ftp bounce attack proxy to 192.168.0.7 (192.168.0.7).
Attempting connection to ftp://anonymous:anon@192.168.0.7:21
Connected:Login credentials accepted by ftp server!
Initiating TCP ftp bounce scan against 192.168.0.5 at 20:37
… … … …
Scanned 1663 ports in 9 seconds via the Bounce scan.
Host 192.168.0.5 appears to be up ... good.
Interesting ports on 192.168.0.5:
(The 1659 ports scanned but not shown below are in state: closed)
PORT STATE SERVICE
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
6969/tcp open acmsoda
MAC Address: 00:11:43:43:A8:34 (Dell (WW Pcba Test))
Nmap finished: 1 IP address (1 host up) scanned in 20.602 seconds Raw
packets sent: 2 (68B) | Rcvd: 1 (46B)
```

- A partir de là, il est possible d'envoyer un fichier sur un serveur
  - Il suffit de disposer d'un fichier contenant des commandes
  - Buffer overflow sur des services locaux non accessibles de l'extérieurs
- Utilisation de 127.0.0.1 sur des implantations FTP buggées

Attaques spécifiques de services

Attaques WEB : CGI scripts

- Attention aux variables reçues par les scripts
- C'est le danger des scripts !
- Exemple PERL « nmap.pl » :

```
#!/usr/bin/perl
# Simple CGI script to let web users run
# an nmap scan from their web browser
# using a GET request
$server = $ENV{'QUERY_STRING'};
@scan = `nmap $server`;
foreach $line (@scan) { print "$line"; }
```

} ← DANGER

- Exemple « nmap.pl » : %3B=« ; » et %6C=« l »

http://server.com/nmap.pl?w%77w.ya%68%6Fo.com%3B%6Cs

Attaques WEB : CGI scripts

- Exemple PERL « display.pl » :

```
#!/usr/bin/perl
$file = $ENV{'QUERY_STRING'};
open(myfile, "$file");
@myfile = <myfile>;
foreach $line (@myfile) { print "$line"; }
close(myfile);
```

DANGER

- Si le nom de fichier est « ls| » ➔ exécution de ls
  - ➔Exécution de code arbitraire
  - ➔Si le serveur est root, alors danger !!!
- Si `open(f, « /toto/$file")` alors « ../bin/ls| »
- Exemple « display.pl » :

`http://server.com/display.pl?%6Cs%7C`

Attaques WEB : CGI scripts

- « Poison NULL byte »
  - Exemple PERL :

```
$pageurl= $realpath . $DATA{ 'adPath' } . ".html";
open(FILE,"$pageurl")||die "can't open $pageurl: $!\n";
@lines= <FILE>;
close( FILE );
```

  - Si ‘adPath=/../../../../../etc/passwd%00’, on peut faire pointer $pageurl sur le fichier /etc/passwd
- Variables non initialisées et par défaut
  - Exemple PHP :

```
if($user && $password)
{
    $ok=check_password($user,$password);
    // Returns 1 if password matches that of the user
}
```

  - On suppose ici que ok vaut 0 par défaut mais

`http://server.com/ex.php?ok=1`

Attaques spécifiques de services

Attaques WEB : CGI scripts

- Solution : Filtrage systématique des données
  - Suppression «\0» dans les variables
  - «Escaping» des caractères dangereux (`«;» devient «\;»`)
  - La liste des caractères est **`&;` `'\"|*?~<>^()[]{}$\n\r espace`**
- Solution : Initialiser « à la main » toutes vos variables !
- Eviter les appels systèmes
- Certains langages ont des modules de sécurité
  - Perl « taint » module (« perl –T »)
  - PHP stocke les variables CGI dans $_POST, ...

Attaques spécifiques de services

Attaques WEB : Cross-Site Scripting

- CSS = « Cross Site Scripting »
- CSS renommé en XSS pour éviter la confusion
  - CSS = « Cascading Style Sheet »
- Exploitation de la dynamicité des sites pour organiser des attaques « CSS »
  - Elles interviennent lorsqu'un attaquant arrive a obtenir des informations sur un utilisateur
    - Vol de comptes, modifier des données utilisateurs,
  - Elles sont basées sur la création de liens malicieux
    - Qui collecte les informations
    - Renvoi sur un lien officiel pour camoufler l'attaque
  - Les liens malformés contiennent
    - Des renvois vers du HTML, JavaScript, VBScript, ActiveX, Flash
- Quelques exemples:

  http://archives.neohapsis.com/archives/vuln-dev/2002-q1/0311.html
  http://www.cgisecurity.com/archive/php/phpNuke_cross_site_scripting.txt
  http://www.cgisecurity.com/archive/php/phpNuke_CSS_5_holes.txt
  http://www.cgisecurity.com/archive/php/phpNuke_2_more_CSS_holes.txt

Attaques spécifiques de services

Attaques WEB : Cross-Site Scripting

- Un exemple ? Script de recherche
  - Lors d'une recherche, on réaffiche souvent le texte cherché
  - Si on oublie d'appliquer le filtrage sur la chaîne transmise

http://www.example.com/search.pl?text=**<script>alert(document.cookie)</script>**

  - Lors de l'affichage du résultat, une pop-up apparaîtra
- Un autre exemple : les forums
  - Poster un message contenant des balises <SCRIPT>, <OBJECT>, <APPLET>, <EMBED>
  - Utiliser comme url d'image de son avatar de forum un script PHP !
  - Attaque par image
    - Configurer son propre serveur WEB, pour que l'extension PHP ne soit plus « .php » mais « .jpg »
    - Référencer cette image comme Avatar, et récupérer les informations « URL referer »
- Voir les éléments DOM affichables !
  - « document.cookie », « document.location.replace»
- Solution : cf. avant

Attaques spécifiques de services

Attaques HTTP : mauvaise configuration HTTP

- Listing automatique des répertoires
  - Obtention de codes sources abandonnés
  - Facilite l'intuition de noms de fichiers
- Suivi de lien symbolique
  - Permet l'extension de la visibilité sur le seveur
  - Mieux vaut utiliser des aliases
- Server side include
  - Insertion de valeurs par le serveur
  - Danger ➔ <!--#exec cgi="/cgi-bin/baratin.pl" -->
- Homepages personnelles
  - Danger ➔ les utilisateurs ne savent pas programmer !
- Attention en cas de MAJ par FTP !
  - On ajoute des trous de sécurités

Attaques spécifiques de services

Attaques HTTP : response splitting

- Le but est de faire de l'empoisonnement de cache WEB
- Le moyen est d'essayer de s'insérer entre les entêtes HTTP et le contenu du fichier HTML.
  - En particulier lors des redirections.
- Exemple basique en JSP (voir VOS tp mdoc !!)

```
<%
response.sendRedirect("/by_lang.jsp?lang="+
request.getParameter("lang"));
%>
```

  - Addresse forgée

```
/redir_lang.jsp?lang=foobar%0d%0aContent-
  Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-
  Type:%20text/html%0d%0aContent-
  Length:%2019%0d%0a%0d%0a<html>Shazam</html>
```


Attaques spécifiques de services

Attaques WEB cache poisoning : attaques HTTP

- Résultat :

```
HTTP/1.1 302 Moved Temporarily
Date: Wed, 24 Dec 2003 15:26:41 GMT
Location: http://10.1.1.1/by_lang.jsp?lang=foobar
Content-Length: 0


HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 19
<html>Shazam</html>


Server: WebLogic XMLX Module 8.1 SP1 Fri Jun 20 23:06:40 PDT
2003 271009 with Content-Type: text/html
```

- Si vous passez par un proxy cache
  - Vous pouvez corrompre le proxy cache
  - Vous pourrez voler de l'information
  - Détourner le trafic des clients du proxy vers une autre @IP

Attaques SQL : SQL injection

- Obtention d'informations dans la base
  - Requête SQL
    - mySQL="SELECT LastName, FirstName, Title, Notes, Extension FROM Employees WHERE (City = ‘ " & strCity & "‘)"
  - Valeur des variables
    - strCity="‘) UNION SELECT OtherField FROM OtherTable WHERE (‘‘=‘"
  - On obtient la requête
    - SELECT LastName, FirstName, Title, Notes, Extension FROM Employees WHERE (City = ‘‘) UNION SELECT OtherField From OtherTable WHERE (‘‘ = ‘‘)

Attaques SQL : SQL injection

- Injection d'informations dans la base
  - Requête SQL
    - SQLString = "INSERT INTO TableName VALUES ('" & name & "', '" & email & "', '" & phone & "')"
  - On remplit les champs avec
    - Name: ' + (SELECT TOP 1 FieldName FROM TableName) + '
    - Email: blah@blah.com
    - Phone: 333-333-3333
  - L'appel à la requête SQL donne
    - INSERT INTO TableName VALUES (" + (SELECT TOP 1 FieldName FROM TableName) + ", 'blah@blah.com', '333-333-3333')

Attaques : Trojan / Virus

- Les trojans et les virus sont des programmes :
  - Malicieux et dormants
  - Qui permettent d'obtenir des droits root
  - Qui peuvent se répliquer
  - Qui peuvent s'ajouter à un programme valide
- Moyen :
  - Attendre que root exécute le Trojan / Virus
  - Emuler (en apparence) un logiciel d'authentification et transmettre les informations
    - Ex: login, ssh
    - Enchaîne le trojan programme avec le vrai
  - Flouer l'utilisateur par un programme par email
  - Automatisation d'une attaque sur un service

Plan de cours

Portes dérobées (Backdoors)

Backdoors/Rootkits

- But
  - Pouvoir revenir sur une machine même après sécurisation
  - Revenir en laissant le moins de traces possibles
  - Revenir rapidement (sans avoir à exploiter une faille de sécurité)
- Portes dérobées simples
  - Vol de mot de passe faible (password cracklib)
  - Rhosts (++), shosts, clefs ssh
  - Démons login, telnetd, sshd, rlogind modifiés
  - Portes dérobées ponctuelles (activée via crond)
  - Portes dérobées via des librairies systèmes (ex: crypt.c)

Portes dérobées (Backdoors)

Backdoors/Rootkits

- Moyens :
  - Ajouter un compte root de préférence au milieu du fichier
    - Ne pas modifier le compte root local
  - Activer un compte avec un UID/GID 0 (sync)
  - SUID une copie de votre shell favori
    - Eviter de le mettre dans /tmp (purge tmpwatch)
  - Modifier un service xinetd ou inted

    ```
    daytime stream tcp nowait /bin/sh sh –i
    ```
  - Ajouter une entrée dans la crontab

    ```
    0 0 * * * /usr/bin/trojancode
    ```
    - Activer un compte pendant une minute
  - Un alias sendmail (dans le fichier /etc/sendmail)

    ```
    decode: "|/usr/bin/uudecode"
    echo "+ +" | /usr/bin/uuencode /root/.rhosts | mail
      decode@target.com
    ```

Portes dérobées (Backdoors)

Moyens (suites)

- Insertion d'un hook dans un programme suid

```
Obtenir les parametres;
si un des paramètres à une valeur spéciale
     créer un xterm root
sinon
     faire le traitement habituel
```

- Utilisation de /dev/kmem pour changer uid/gid
- Le mail régulier de /etc/passwd ou ypcat par crond
- Installation d'un service (root) qui peut
  - utiliser un canal encrypté (pour éviter le sniffing) ➔ un second SSH !!
  - écouter sur un port TCP >1024 ou sur un port ouvert sur le firewall (ex: 25, 110, ...)
    Service FTP/WEB, xterm par tcp
  - écouter sur un port UDP
    Ne laisse pas de trace de connexion !
    Souvent, on laisse le port UDP 53 (DNS) ouvert ...
  - capturer les paquets ICMP
    Les firewalls laissent souvent passer les paquets ICMP echo request
    L'echo request permet le transport d'informations (voir la partie sécurité réseau et le p2p)

Backdoors/Rootkits

De l'art du camouflage !!

- Ces modifications laissent des traces visibles !!
- Dans tous les cas, si on modifie le système :
  - « last » affiche les dernières connexions
  - « ls », « ps » affiche les fichiers et les processus
  - « netstat », « lsof » affiche les connexions ouvertes
  - « ifconfig » affiche si la carte est en mode promiscuous
  - La lecture des droits sur les /dev/tty* ayant les droits root
  - Les logs contiennent plein d'infos

Backdoors/Rootkits

Backdoors/Rootkits

De l'art du camouflage !!

- L'objectif suivant est donc de camoufler ces traces aux utilisateurs (root inclus) !!
- But :
  - Camoufler au système (et aux utilisateurs) la présence d'éléments indésirables
- Moyen :
  - Manipuler les fichiers de logs
  - Remplacement des fichiers sur la machine
  - Insertion de modules noyaux modifiant le comportement du système !
    LKM =« Loadable Kernel Modules »

Backdoors/Rootkits

Camouflage : Manipulation des fichiers de logs utmp

- Pour vérifier la présence d'un utilisateur sur un machine
  - Utilisation des commandes « who », « w » , « finger »
  - Ces fichiers utilisent le log /var/run/utmp
  - Il est possible de modifier le fichier pour effacer l'utilisateur !
- Voir le source code pour modifier « utmp »

  http://www.phrack.org/show.php?p=25&a=6
- Utilisation de la structure utmp définie dans utmp.h
- Si on a un accès root, on peut lire les enregistrements et éliminer les logs dangereux !

Camouflage : Manipulation des programmes

Backdoors/Rootkits

- Modification des programmes de base unix (core utils)
  - «ls», «du», « find »
  - «crontab», «killall», «kill»
  - «netstat», «ps», «ifconfig», «pidof», «top»
- Modification des démons de log
  - « syslogd » (noyau), « tcpd » (connexions)
- Ajout de faux devices dans « /dev » pour les processus modifier et leur configuration
- Purge partielle des fichiers de log (/var/log)

Camouflage : Loadable Kernel Modules (LKM)

Backdoors/Rootkits

- Les modifications précédentes sont « assez » facilement détectables
  - Voir la section outils de protections
  - Tripwire, utilisation d'une copie des utilitaires
- Le plus efficace est d'agir au niveau noyau
  - Utilisation de noyaux chargeables
  - Permet un contrôle quasiment complet du système
    - Les appels systèmes peuvent être interceptés
    - /usr/include/sys/syscall.h (execve, sync, stty, …)
  - Permet un camouflage plus efficace
    - Tous les programmes sont affectés sans être modifiés !
  - Complexe et extrêmement « système dépendant »

Backdoors/Rootkits

Camouflage : Loadable Kernel Modules (LKM)

- Plusieurs possibilités s'offrent à l'attaquant
  - Insérer un nouveau module noyau
  - Modifier un module noyau
- Caractéristiques de l'insertion
  - Création à l'avance
  - Nombre de manipulations locales restreintes
  - Nécessite le camouflage
- Caractéristiques du patch
  - Empreinte système plus faible
  - Nécessite des manipulations locales
  - Peut être adapté au système « à la volée »

Backdoors/Rootkits

Camouflage : patch des Loadable Kernel Modules

- Chaque module noyau contient des fonctions standards
  - init_module, cleanup_module
- Lors du chargement, le système exécute le code « sys_init_module » qui
  - Copie le code de l'espace utilisateur vers le noyau le code du module
  - Exécute la fonction « init_module »
- Le nom des fonctions internes est stocké dans un entête ELF.

```
# objdump -t monmodule.ko
…
00000000 g     F .init.text     000000a1 init_module
00000000 g     F .exit.text     00000044 cleanup_module
…
```

- Il faut alors injecter le code supplémentaire
  - Possible car le code ELF est « rerlocatable » ➔ déplacable
  - Par défaut ce type de code permet le partage de code entre les modules ➔ Voir les modules iptables !!!!

Camouflage : patch des Loadable Kernel Modules

Backdoors/Rootkits

- Voici un code simple (fichier codeAInjecter.c)

```
#define MODULE
#define __KERNEL__
#include <linux/module.h>
#include <linux/kernel.h>
int inje_module (void) {
    printk ("<1> Injected\n"); return 0;
}
```

- On le compile sans faire l'édition de lien :

```
cc -O2 -c stealth.c
```

- L'injection se réalise par

```
ld -r moduleOrig.o codeAInjecter.o -o moduleInfecte.o
mv moduleInfecte.o moduleOrig.o
```


Camouflage : patch des Loadable Kernel Modules

Backdoors/Rootkits

- Il suffit ensuite de faire en sorte que le module invoque le code injecté
  - On renomme les symboles dans la table de symboles
  - « init-module » devient « init-new »
  - « inje-module » devient « init-module »
  - Le code injecter doit invoquer l'ancien « init-module »
    - Il suffit d'ajouter la ligne « init-new() » dans le code de « inje-module »

Camouflage : patch des Loadable Kernel Modules

- Le code ajouter ou le module ajouter peut
  - Intercepter la frappe clavier
    - Interception des appels put_queue, receive_buf, tty_read, sys_read
  - Cacher un fichier
    - rappel sous linux, /proc est un système de fichier
    - Action sur le VFS ou directement sur les FS
  - Cacher un PID
    - Manipuler la liste des processus (double liste)
    - Enlever le processus de la liste de processus en attente
    - Mettre le PID du processus à 0
  - Cacher une connexion
  - Exécuter un programme en root

Backdoors/Rootkits

Rootkit :

- Les rootkits sont un assemblage de programmes.
  - Des outils d'attaque (sur un service)
  - Des outils de camouflage (LKM)
  - Des outils de portes dérobées (backdoor)
- Quelques RootKit connus :
  - Knark
    - Installe un module sysmod.o et intercepte les appels fork, read, execve, kill, ioctl, settimeofday, clone
    - Fournit un ensemble d'attaques connus
  - Adore
    - Installe un module et intercepte les appels fork, write, close, clone, kill, mkdir, clone, getdents
    - Il fournit un utilitaire « ava » pour cacher un fichier, une tâche ou une connexion

Backdoors/Rootkits

Plan de cours

Introduction

Attaques génériques de services

Attaques spécifiques de services

Modules de noyaux et trappe

Backdoors / Rootkits

Outils de protection

Audit et check-list

- systrace
- TCPwrapper
- Audit de code (include php)
- limiter services
- PKI

Protection du système : LIBSAFE

Outils de protection

- Il s'agit d'une librairie de fonctions
- Elle offre une protection de base contre les « buffer overflow »
- Elle est compatible avec des exécutable pré-compilé
- Elle s'utilise de façon transparente
- L'overhead reste faible
- Elle remplace les fonctions vulnérables au « bof »
  - Strcpy, strcat ➔ overflow sur le buffer dest
  - getwd, gets ➔ overflow sur le buffer
  - [v]scanf, [v]sprintf, realpath ➔ overflow sur le buffer dest

Protection du système : LIBSAFE

Outils de protection

- Elle intercepte tous les appels à ces méthodes
  - Elle garantie que les opérations restent dans les limites prévues
- Méthode 1 (ignoré par les programmes suid)
  - Utilisation d'un ld récent

```
LD_PRELOAD = /lib/libsafe.so.2
export LD_PRELOAD
```

  - Exécution du programme
- Méthode 2 (pour les programmes suid)
  - Edition de « /etc/ld.so.preload »
- En cas d'attaque, les processus sont tués (SIGKILL)

```
Dec 29 17:18:42 eos libsafe[15704]: Detected an attempt to
        write across stack boundary.
Dec 29 17:18:42 eos libsafe[15704]: Terminating
        /home/legond/bin/test/bof
Dec 29 17:18:43 eos libsafe[15704]: scanf()
```

Protection du système : STACKGUARD

Outils de protection

- Protection contre les « bof »
  - Approche par compilation
  - Ne requiert aucun changement dans le code source
- On patche le compilateur pour qu'il encapsule les données manipulées
  - On insère des marqueurs appelé « canary »
- « Terminator canary »
  - Ils sont insérés en fin de données (Ex: les chaînes)
  - On utilise un marqueur de fin de chaîne mutiple

Protection du système : STACKGUARD

Outils de protection

- « Random canary »
  - On insère des marqueurs à des endroits stratégiques (ex: avant l'adresse de retour)
  - La valeur des marqueurs est générée par random à chaque exécution
  - On les vérifie les valeurs régulièrement
- « Random XOR canary »
  - Ce sont des marqueurs « random canary »
  - Les valeurs sont un xor entre un random et une donnée (signature)

Protection du système : STACKGUARD

Outils de protection

- Algorithme pour chaque appel de fonction :
  - On détermine l'emplacement du canary sur la pile
  - On alloue l'espace sur la pile
  - On initialise le canary
  - On vérifie la valeur avant le retour à l'appelant
  - On désalloue le canary
  - Si la valeur est incorrecte
    - on trace (comme libsafe) et on stoppe
  - Si la valeur est correct, on revient
- Impact léger sur la mémoire et les performances
  - attention aux appels récursifs tout de même

Protection du système : STACKSHIELD

Outils de protection

- Protection contre les « bof »
  - Approche identique à STACKGUARD
- Autre méthode : Création d'une autre pile
  - Lors de l'appel l'adresse de retour est stockée dans cette nouvelle pile
  - Lors du retour on copie l'adresse de retour avant d'effectuer le saut
- StackShield inclus une protection contre la corruption des pointeurs de fonctions
- StackShield ne détecte pas les « bof », il revient toujours à l'appelant !
- STACKSHIELD et STACKGUARD ne sont pas incontournables
  - http://www.phrack.org/show.php?p=56&a=5

Protection du système : Formatguard

Outils de protection

- Protection contre les attaques par « format string »
  - Approche identique à STACKGUARD, STACKSHIELD
  - Approche par compilation
  - Ne requiert aucun changement dans le code source
- Principes
  - Interdire le code de formatage « %n »
  - Interdire le printf dynamique
  - Compter le nombre d'arguments (pas de varargs infini)
  - Filtrer les chaînes (en particulier le signe '%')

Protection du système : Formatguard

Outils de protection

- Compatibilité
  - La version sécuritaire de varargs n'est pas compatible avec l'existant
  - L'interdiction de « %n » peut bloquer certains programmes
  - L'interdiction de printf dynamique peut aussi bloquer des programmes (GNU Intl library)
- Sécurité
  - Attaque par nombre d'arguments < à ceux attendu
    - Il existe des fonctions std qui utilisent cette technique (même dans le glibc)
  - L'utilisateur de pointeur de fonction sur printf et autres interdit la protection FormatGuard
  - Les appels directs a « vsprintf » et consœurs avec une liste dynamique (varargs)

Protection système : Noyaux renforcés

Outils de protection

- Hardening Kernels
  - Il s'agit de patches noyaux
  - Il suffit ensuite de recompiler le noyau
- Ensemble des protections mise en place au niveau du noyau
  - Protection contre les buffer overflow
  - Protection du Système de fichier (FS)
  - Renforcement des moyens d'audit
  - Protection d'exécution
  - Protection réseau
  - Protections diverses

Protection système : Noyaux renforcés

Outils de protection

- Protection contre les buffer overflow
  - Rendre la pile non exécutable
    - Ne protège pas contre le débordement de tas
    - Ne protège pas contre les appels systèmes (libc)
    - Autoriser l'exécution « officielle » de code dans la pile (sauts par trampolines)
  - Empêcher le changement de droit sur les pages
    - NoX → X et R →RW
  - Rendre aléatoire les adresses de programmes mmap
    - Adresse ELF dynamique, la pile d'exécution
    - Interdire les adresses fixes et les droits en exécution
  - Mémoire noyau en lecture seule et désactivation des modules

Protection système : Noyaux renforcés

Outils de protection

- Protection du FS
  - Gestion des ACL
    - Permet de spécifier des ACL complexes sur les FS
    - Un fichier peut être X, R, W, append, hidden
  - Restriction d'accès
    - Consultation seulement les processus que l'on possède
    - Ne pas accéder à dmesg, aux symboles et modules noyaux
    - « /proc » seulement pour root ou un groupe particulier
    - Liens symboliques interdit sur un répertoire +t / un autre uid
    - Empêche un processus de suivre un lien « interdit »
  - Tout programme doit avoir les descripteurs 0,1,2

Protection système : Noyaux renforcés

Outils de protection

- Protection de FS
  - Renforcement du chroot
    - Signaux limités, mount/chmod/mknod/ptrace interdit
    - Interdire les doubles chroot, restriction sur les priorités
- Audit du noyau ➔ attention à la charge engendrée
  - Journalisation des processus pour un seul groupe
    - Eviter le DOS, tous les services doivent être dans le même groupe. ☹
  - Journalisation des appels execve
    - Journalisation des processus normaux et mis en cage
  - Journalisation des appels chdir, u/mount
  - Journalisation IPC, signaux, fork échoués
  - Journalisation du set*uid (restreinte au setuid root)
  - Journalisation de la MAJ de l'horloge

Protection système : Noyaux renforcés

Outils de protection

- Protection d'exécution
  - Les limitations en ressources sont aussi vérifiées lors d'un execve
  - PID au hasard
  - Restrictions d'accès sur les pages mémoires (umask). Par défaut, c'est 777 sous linux.
  - Limitation d'accès de root aux consoles
    - Interdire l'accès root sur les consoles physiques, séries, pseudo-console
  - Limitation du nombre de processus pour un GID
    - Nombre total et création par seconde. Anti-«fork bomb»
  - Interdire l'exécution d'un programme hors de répertoires de confiance
    - Inutile sur les interpréteurs de script (ex: Perl)
    - On peut en limiter les effets de l'interdiction
  - Protection glibc (ignorer le LD_PRELOAD) et programme ld
  - Limiter les appels de « ptrace » à root (et peuvent être journaliser)

Protection système : Noyaux renforcés

Outils de protection

- Protection réseau
  - Rendre aléatoire les numéros IP
  - Altérer les réponses PING
    - éviter la détection des empruntes de la pile IP
  - Rendre aléatoire le TTL entre un min et un max
  - Limiter l'ouverture de certains type de socket
  - Interdire l'ouverture de socket à certains groupes
    - Toutes les sockets, les clientes, les serveurs
  - Rendre aléatoire les ID des appels RPC

Protection système : Noyaux renforcés

Outils de protection

- Protection réseau
  - Rendre aléatoire les numéros IP
  - Altérer les réponses PING
    - éviter la détection des empruntes de la pile IP
  - Rendre aléatoire le TTL entre un min et un max
  - Limiter l'ouverture de certains type de socket
  - Interdire l'ouverture de socket à certains groupes
    - Toutes, clientes, serveurs
- Divers
  - Limite le changement des touches à root
  - Activer/désactiver la configuration dynamique des options de sécurité
  - Changer le noms des fichiers core-dump
    - Ex: coredump-nomprocessus.PID

Protection du système : LIDS / LSM

Outils de protection

- LIDS =« Linux Intrusion Detection System »

  http://www.lids.org
- Patch noyaux permet un contrôle d'accès aux ressources
  - Nom: Mandatory Access Control (MAC)
  - Définition des droits d'accès aux ressources
  - Une ressource peut être interdit même à root
    - Mémoire, Accès E/S, accès aux « devices », fichiers, réseau
- Utilise le framework LSM pour le noyau

  http://lsm.immunix.org/

Protection du système : patch OpenWall

Outils de protection

- Nommé owl (« OpenWall Linux »)

  http://www.openwall.com/

- Collection de quelques patch pour le noyau linux incluant les options de sécurité
  - Non exécution sur la pile
  - Pipes et Liens limités sur /tmp
  - Accès limité sur /proc
  - Fd 0,1,2 toujours ouverts
  - Protection contre les « Fork bomb »
  - Protection IPC
    - purge des blocs de mémoire partagés non utilisés

Protection du système : patch grSecurity / PAX

Outils de protection

- grSecurity
  - http://www.grsecurity.net/
  - A l'origine un portage de OpenWall pour linux
- Collection de quelques patch pour le noyau linux incluant les options de sécurité
  - Intègre les patches OpenWall
  - Intègre ses propres patches
  - Patch PAX (protection mémoire)
    - séparation entre les zones exécutables et les zones en écriture
    - http://pax.grsecurity.net/

Protection du système : la virtualisation

Outils de protection

- Machine virtuelle
  - C'est une couche d'interception et d'indirection entre une application et un OS
  - Découple la machine physique et la vision de la machine par l'application
- Une autre solution de protection est la virtualisation
  - Création de machines virtuels encapsulées sur une machine physique
  - But : isolation des différents services
    - Prison virtuelle: Si on parvient a corrompre un service, on reste enfermé
  - En renaissance actuellement
    - Exploité dans les années 1960 par les mainframes
    - Devenu obsolète par la démocratisation de l'informatique
    - Devenu d'actualité dans les années 1990 pour exploiter les mutli-processeurs
    - Devenu d'actualité dans les années 2000 pour exploiter l'isolation et le multi-core

Protection du système : la virtualisation

Outils de protection

- Pourquoi ? Usage moyen de serveurs !!
  - Mémoire
    - 45% de la RAM non utilisée 99.9% du temps
    - 25% de la RAM jamais utilisé
  - CPU
    - 85% des ressources CPU non utilisée 99.9% du temps
    - 81% des resources CPU jamais exploité en concurrence
  - Disque
    - 68% de l'espace disque jamais occupé
- ➔On peut se permettre de virtualiser

Protection du système : la prison chroot

Outils de protection

- La première étape vers la virtualisation est chroot
  - On l'appel la prison chroot (« chroot jail »)
  - Il s'agit de limiter la vision FS de l'application
  - Il faut alors reconstituer un environnement minimaliste pour l'application
  - Il faut aussi penser a abandonner les privilèges utilisateurs
- Très utile pour des services fortement exposés
  - Ex: BIND, FTP publics
- Utilisation pour un shell limité
  - http://olivier.sessink.nl/jailkit/

Protection du système : la prison chroot

Outils de protection

- Ex: Installation d'un service BIND
- Il faut préparer l'environnement
  - Création de la racine de la prison (ex: /chrootjail)
  - Création des répertoires utiles (base) : /chrootjail/xxx
    - « xxx » ➔ etc, var, dev et bin, lib, usr
  - Création d'un /chrootjail/etc/passwd SANS compte root
  - Création des répertoires nécessaires au service
  - Copie des fichiers de configuration dans notre /chrootjail/etc
    - named.conf (pour BIND), localtime,
  - Création des devices nécessaires (mknod)
    - /chrootjail/dev/zero, /chrootjail/dev/null

Protection du système : la prison chroot

Outils de protection

- Problème du log
  - On log en général par syslogd grâce à /dev/log
  - Il est possible de créer une socket /chrootjail/dev/log
  - Et de relancer syslogd avec l'option –a /chrootjail/dev/log
- Pour l'accès au répertoire, utilisation de « mount –bind »
- Il faut ensuite déterminer les ressources nécessaires

```
# ldd /usr/sbin/named
        linux-gate.so.1 =>  (0xffffe000)
        libcrypto.so.0.9.7 => /usr/lib/libcrypto.so.0.9.7
(0x40027000)
        libldap.so.2 => /usr/lib/libldap.so.2 (0x40126000)
        liblber.so.2 => /usr/lib/liblber.so.2 (0x40158000)
        libresolv.so.2 => /lib/libresolv.so.2 (0x40164000)
        libnsl.so.1 => /lib/libnsl.so.1 (0x40175000)
        libpthread.so.0 => /lib/tls/libpthread.so.0 (0x40188000)
        libc.so.6 => /lib/tls/libc.so.6 (0x40199000)
        libdl.so.2 => /lib/libdl.so.2 (0x402b8000)
        libsasl2.so.2 => /usr/lib/libsasl2.so.2 (0x402bb000)
        libssl.so.0.9.7 => /usr/lib/libssl.so.0.9.7 (0x402d1000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

- Copier les librairies utilisées

Protection du système : la prison chroot

Outils de protection

- Utiliser strace, onjdump nom de service pour déterminer les fichiers (devices) utilisés

```
# strace /usr/sbin/named
   execve("/usr/sbin/named", ["/usr/sbin/named", "-h"], [/* 69 vars */]) = 0
   uname({sys="Linux", node="scylla.lip6.fr", ...}) = 0
   brk(0)                                  = 0x81a7000
   old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1,
   0) = 0x40015000
   open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or
   directory)
   open("/etc/ld.so.cache", O_RDONLY)      = 3
   fstat64(3, {st_mode=S_IFREG|0644, st_size=69439, ...}) = 0
   old_mmap(NULL, 69439, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40016000
   close(3)                                = 0
   …
   open("/usr/share/locale/fr/libdns.cat", O_RDONLY) = -1 ENOENT (No such file
   or directory)
   open("/usr/share/locale/fr/libdns.cat", O_RDONLY) = -1 ENOENT (No such file
   or directory)
   open("/usr/share/locale/fr/LC_MESSAGES/libdns.cat", O_RDONLY) = -1 ENOENT
   (No such file or directory)
   open("/usr/share/locale/fr/libdns.cat", O_RDONLY) = -1 ENOENT (No such file
   or directory)
   open("/usr/share/locale/fr/LC_MESSAGES/libdns.cat", O_RDONLY) = -1 ENOENT
   (No such file or directory)
```

- En particulier pour /usr/share/local/fr
- Ne pas oublier le chown et chmod sur les fichiers!

Protection du système : virtualisation

Outils de protection

- Il reste possible de s'échapper d'une prison chroot
- Il existe des solutions plus poussées d'isolation
  - UML ➔ User Mode Linux, noyau en espace utilisateur
  - C'est un patch noyau ➔ Complexe à mettre en place
  - http://user-mode-linux.sourceforge.net/
  - Utilisé au départ pour la formation et le test
  - Contrôle important sur l'accès aux ressources physiques
  - Il permet de charger une image de système
    - Faire une image disque du système (utilitaire dd)
    - Monter l'image pour (mount –loop)
    - Ou exécuter le noyaux UML avec les bon paramètre
- Solution peu performante en terme de performance

Protection du système : virtualisation

Outils de protection

- Vmware
  - C'est un émulateur complet de machine avec BIOS
  - Le système n'a pas du tout conscient que le système est virtualisé
  - Encore moins performant que UML
  - Nécessite des ressources importantes
  - Isolation très forte, mais les ressources sont bloquées
- Vserver
  - C'est une évolution du principe de chroot (prison)
  - On tente de pousser plus loin l'isolation sans machine virtuelle
  - On créer des contextes (proche des noyaux renforcés)
  - C'est beaucoup plus légé que l'émulation pure
  - Il offre moins d'isolement mais une meilleur gestion des ressources
  - Avantages et Inconvénients : partage interne de ressources

Protection du système : virtualisation

- Xen
  - C'est un manageur de systèmes virtuels
  - Il se place entre le matériel et le système
    - Offre une couche d'abstraction et d'isolement
  - Les systèmes virtualisés ont conscience de la sous-couche Xen
    - Il nécessite une adaptation
  - Xen est un hyperviseur, contrôlable
  - Chaque système tourne dans un domaine configurable

Espace utilisateur	Espace utilisateur	Espace utilisateur	Espace utilisateur	Logiciels de contrôle Xen
Plan 9	**FreeBSD**	**NetBSD**	**Linux**	*Xeno-Linux*
Pilotes Xen	*Pilotes Xen*	*Pilotes Xen*	*Pilotes Xen*	*Pilotes Xen*
Xen				
Matériel : processeur, mémoire, stockage, réseau, etc.				

Plan de cours

Audit : processus accounting

Audit et check-list

- Le premier outil psacct
  - C'est un rpm qui permet d'activer les traces du système dans le noyaux linux
  - Pour activer la trace, il suffit d'exécuter
    - accton fichier_de_log
  - Les fichiers sont binaires comme wtmp et utmp
  - Ils grossissent très vite si l'activité est importante
  - Il faut des outils pour parser les logs
- Les logs ne sont pas facilement transmissible par le réseau
  - Il doit possible de faire un tunnel (sécurisé) vers une autre machine
  - Le tunnel lira ses données à partir du fichier
  - C'est une solution à mettre en place
- Un exemple

```
# ./acct_watch
   program      uid/gid      cpu          start time                flags
   ls           root/root    0.02u 0.00s  [09:46:34 Thu 1997-04-10] ttydev:3/3 SU
   mail-queue   mail/mail    0.01u 0.01s  [09:47:16 Thu 1997-04-10] NOTTY
   mail-smtpd   maild/maild  0.00u 0.02s  [09:47:16 Thu 1997-04-10] SU, NOTTY
   rcs          mbp/mbp      0.02u 0.01s  [09:47:59 Thu 1997-04-10] NOTTY
   id           pdb/pdb      0.01u 0.01s  [09:49:44 Thu 1997-04-10] ttydev:3/4
```


Audit : Contrôle d'intégrité

Audit et check-list

- Tripwire
  - Permet de contrôle l'intégrité
  - Fichier de configuration simple
  - Les md5 sont a stocker sur des supports surs
- « rpm –V –a » vérifie les fichiers par rapport à la base rpm
  - Elle peut être corrompue mais c'est un bon début
  - Affiche de 9 bits de status
    - '.' ➔ c'est ok
    - '5' (somme md5 hs), 'S' (taille du fichier hs)
      'l' (pb de lien symbolique), 't' (erreur horodatage)
      'm' (pb de mode/droit), 'u'/'g' (pb de proriété)
- Plus sur, avoir une référence externe

  # rpm -Vvp ftp://mirror.site/dir/RedHat/RPMS/fileutils-3.16-10.i386.rpm

  S.5....T /bin/ls
- Root kit detection
- Sleuthkit

Audit : processus accounting

- Forensics Analysis
- Pot de miel – Honeyd
- Cgiscan / phpscan