Introduction à la sécurité – Cours 1

Catalin Dima

Objectifs du cours

- Notions de base : vulnérabilité, menaces, attaque et défense,...
- Services de sécurité : confidentialité, authenticité, intégrité, disponibilité,...
- Politiques de sécurité.
- Modèles de sécurité : contrôle d'accès, flux d'information, modèles de protocôles de sécurité.
- Sécurité dans Java : APIs de sécurité.

Ressources

- Cours en ligne
  http ://www.univ-paris12.fr/lacl/dima/securite
- Matt Bishop, Computer Security, Art and Science.
- Articles, Internet,...

Evaluation

- Examen
- Projet

Pourquoi un domaine de la sécurité

- Aspects sociaux et sociologiques.
- Attaques à grande échelle contre des sites Internet.
- Ménaces contre les ordinateurs personnels.
- Interconnexion entre les attaques à grande échelle et les attaques contre les ordinateurs personnels.

Pluridisciplinarité :

- Aspects théoriques : cryptographie, logiques pour la vérification, modèles probabilistes, évaluation des performances, calculabilité et complexité,...
- Aspects pratiques : systèmes d'exploitation, réseaux, techniques de programmation, bases de données,...

Types de services de sécurité

- *Confidentialité*, : l'assurance que ses actions/données/communications ne soient pas examinées par des personnes/parties non-authorisées.
- *Authenticité* : l'assurance que l'accès/la conversation se passe avec une ou plusieurs personnes/parties légitimes.
- *Integrité* : l'assurance que ses propres données ne soient pas modifiées ou corrompues sans son accord, ou que ses communications ne soient pas pas modifiés après avoir été envoyées.
- *Disponibilité* : abilité d'utiliser une ressource à son gré, tant qu'on veut l'utiliser de manière correcte.
- *Non-repudiation* : l'impossibilité de nier ou désavouer une action/transaction/message dont on est la source.

Menaces

- *Révélation* : (angl. *disclosure*) accès non-authorisé à l'information.
- *De-ception* : acceptation de mauvaises données.
- *Interruption* ou prévention des opérations correctes.
- *Usurpation* : contrôle non-authorisé d'une partie d'un système.

Exemples :

1. *Fouiner* : (angl. *snooping*) interception non-authorisée d'information ; sous-classe : *wiretapping*.
2. *Alteration* : modification non-authorisée de l'information ; sous-classe : *man-in-the-middle attack*.
3. *Imposture* : (angl. *masquerading* ou *spoofing*), se faire passer pour un autre.
4. *Répudiation de l'origine*, *déni de réception*, *delai de service*.
5. *Déni de service*.

Types d'attaque

- *Intrusion* dans des systèmes informatiques, par vol, casse ou récupération non-authorisée de mots de passe ou par usurpation d'identité (même identité IP).
- *Accès* non-autorisé des utilisateurs d'un système informatique aux informations ou ressources auxquelles, normalement, ils n'ont pas l'autorisation.
- *Exploitation des failles de sécurité* (erreurs de conception/programmation/configuration des services), aboutissant à divers degrés de *prise de contrôle* d'une machine.
- *Fuites d'information* en cours de transport à travers des canaux de communication.
- *Deni de service* (angl. DoS *Denial of service*) : attaque contre une ressource (d'habitude des serveurs) en la rendant inutilisable aux utilisateurs légitimes.
- *Inginerie sociale* : exploiter les faiblesses de la nature humaine.
- *Vol* de matériaux sensibles
- *Scavenging*.

Moyens d'attaque

- *Virus* : s'ajoute à un programme et est exécuté avec le programme,
  - Prend le contrôle de l'ordinateur et se propage.
- *Ver* : programme qui s'exécute indépendamment d'autres programmes.
  - But – se réproduir et ainsi ralentir le système par saturation d'une ressource.
- *Cheval de Troie* : code malveillant qui traverse, caché dans un programme, des barrières de communication (pare-feu, IDS, etc.)
  - Permet la mainmise sur les ressources de l'hôte.
- *Bombe logique* : cheval de Troie dont l'action est déclenchée par temporisation ou par certaines actions des utilisateurs du système.
- *Porte arrière* ou *trappe* (angl. *backdoor* ou *trapdoor*) : mechanisme de connexion contournant les voies normales de connexion/accès à un système informatique.
- *Canaux couverts* (angl. *covert channels*) : canaux de communication supposant permettre l'échange d'informations authorisées, mais en fait permettant la fuite d'informations sensibles ou de donner des instructions malveillantes (à une machine sous l'emprise de l'attaquant) sans se faire soupçonner par les "chiens de garde" du système.

Phases d'une attaque

- *Reconnaissance* – apprendre le plus possible sur la future cible de l'attaque :
  - "Ingénierie sociale", effraction, fouille de poubelles...
  - Recherche dans les documents publiques de ou sur la cible : google, usenet, bases de données "whois", DNS...
- *Balayage* (scrutation, scanning) – recherche de points/portes d'entrée mal protégés.
  - Recherche de modems connectés sur une ligne téléphonique (en attente de connexions), recherche de lignes téléphoniques internes.
  - Construction des plans de réseaux cibles : `ethereal`, mais aussi `traceroute, ping`.
  - Détermination des ports ouverts en utilisant des scanneurs : `ethereal`, `nmap`.

    Ménagerie de scanneurs de vulnerabilités : Nessus, eEye, CyberCop...
  - Détermination de vulnerabilités de scripts CGI.
  - Tromper les systèmes de detection d'intrusion au niveau réseau : fragmentation du traffic.

Phases d'une attaque : Accès

- Gagner l'accès au niveau du système d'exploitation :
  - *Script Kiddie* : lancer des attaques conçus par d'autres gens...
  - Techniques de type *Buffer Overflow*.
  - Attaques contre les mots de passe : deviner des mots de passe par défaut, scripts de connexion, "casser" les mots de passe cryptés, récupérer les mots de passe cryptés (après avoir pénétré le système).
- Gagner l'accès/prendre le contrôle d'un réseau :
  - Duper les switchs Ethernet pour les faire envoyer leur traffic par la machine de l'attaquant.
  - Sniffer les *connexions sécurisées* en dupant les clients lors de la connexion.
  - "Déguiser" son adresse IP (angl. *IP adress spoofing*), en prenant une autre identité.
  - Vol de sessions, en combinaison avec des techniques de reniflage.

Phases d'une attaque : Manutention d'accès

- *Chevaux de Troie* : programmes à apparence inoffensive, voire utile, mais qui cachent des capabilités malicieuses.
- *Trappes*, ou *portes arrière* (backdoors) : code permettant à l'attaquant de se connecter sur une machine en utilisant une "entrée alternative", invisible pour un utilisateur banal.
- *RootKits* : modification apportées au système informatique permettant l'obtention de droits de super-utilisateur

Phases d'une attaque : couvrir les traces

- Altérer les journaux d'événements (angl. *event log*) ou du système.
- Création de fichiers cachés ou difficiles à trouver.
- RootKits de niveau noyau.
- Canaux couverts : encapsulation de protocôles, utilisation de protocôles pour d'autre fin que leur mission (ex. ICMP).

Politiques de sécurité

- Une **politique de sécurité** est une déclaration de ce qui est et ce qui n'est pas permis de faire dans, avec ou en intéragissant avec un système.
  - Soit P l'ensemble d'états *franchissables* (ds. lesquels le système peut se trouver durant son travail).
  - Une politique de sécurité Π définit (ou est définie par) un ensemble d'états Q dans lequel le système est autorisé à résider
    - Les états de $P \setminus Q$ sont les états "non-sûrs".
- Un **mécanisme de sécurité** est une méthode, un outil ou une procedure permettant d'implémenter (i.e. renforcer, angl. *enforcing*) une politique de sécurité.
  - Un mécanisme de sécurité est un moyen d'assurer que le système n'entrera jamais dans un état de $P \setminus Q$.
- Le système est sûr si, en démarrant d'un état sûr, il n'entrera jamais dans un état non-sûr.

Politiques de sécurité

- Buts d'une politique de sécurité : prévention, détection et/ou récupération.
- *Hypothèses* d'une politique de sécurité : *axiomes* choisies par les concepteurs, supposées assurer le bon fonctionnement du système.
  - Étape très fragile dans la conception d'un système !
  - Implique une décomposition des états du système en états *sûrs* et *non-sûrs*.
- Conception :
  - Analyse comparative coûts-bénéfices.
  - Assurance, validation.

Quelques principes de sécurité

- *Principe du plus petit privilège* : un sujet (programme, utilisateur...) devrait se voir assigné seulement les privilèges qu'il aurait besoin pour accomplir sa tâche.
- *Principe de la configuration sûre par défaut* : un sujet ne devrait pas avoir accès à un objet sans qu'on lui donne cet accès de manière explicite.
- *Principe de la conception ouverte* : la sécurité d'un mécanisme ne devrait pas dépendre du sécret entourant sa conception ou son implémentation.
- *Principe de la séparation des privilèges* : un système ne devrait pas permettre l'accès à des objets sensibles sur la base d'une seule condition.
  - Variantes : *séparation des tâches/fonctions*.
- *Principe du plus petit mécanisme commun* : on ne devrait pas utiliser des mécanismes différents pour l'accès à la même ressource.
- *Principe de l'économie de mécanismes* : un mécanisme de sécurité devrait être aussi simple que possible.
- *Principe de l'acceptabilité psychologique* : un mécanisme de sécurité ne devrait pas rendre l'accès à la ressource trop difficile par rapport au cas où le mécanisme n'existerait pas.

Quelques remarques pratiques et légales

- Analyse *coût-bénéfices* : des données ou des activités d'une moindre importance *et n'ayant pas de relation avec des activités sensibles* ne devraient pas faire l'objet des mécanismes lourds de sécurité.
- Analyse des *risques* : faire le travail des attaquants pendant l'installation du système pour évaluer ses vulnérabilités et décider des solutions à apporter.
- Analyse *juridique* :
  - Est-ce qu'un certain nombre de solutions de sécurité sont permises d'installation dans le système par les lois ou règlements ?
  - Est il permis d'importer/exporter certaines solutions/outils/modules de sécurité ?
  - Dans quel cadre est-il permis d'utiliser certaines outils d'analyse de sécurité ?
  - Quels sont les moyens légaux de repression des fraudes et quelle est leur implication dans les coûts de fonctionnement de la companie ?
- Le *facteur humain* : problèmes organisationnelles, problèmes de relations inter-humaines,...

Conclusions pour l'introduction

- Importance d'une bonne pratique du développement des logiciels.
- Importance de la maîtrise des concepts de génie logiciel.
- Importance d'une méthodologie de certification de logiciels.
- Importance de la maîtrise des techniques de sécurisation des systèmes et des réseaux.
- Importance de l'expérience avec les techniques de détection d'attaques.