Formalisation et implantation de politiques de sécurité de réseaux.

Nicolas Stouls *(Nicolas.Stouls@imag.fr)* et Vianney Darmaillacq *(Vianney.Darmaillacq@imag.fr)*
Laboratoire LSR IMAG, Grenoble France

Motivations :

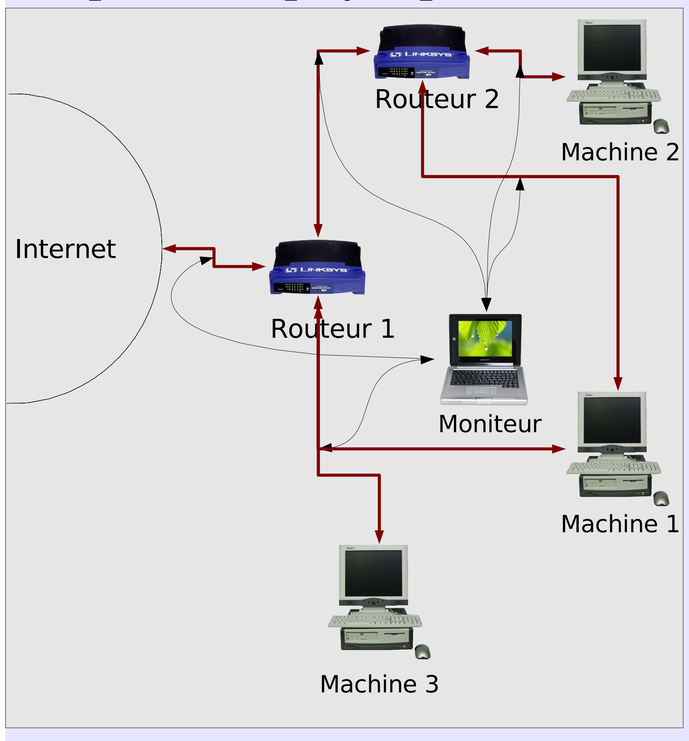
- Description progressive *(Raffinement)* d'une **politique de sécurité** *(Règles définissant les droits de chacun dans un réseau)*
- Mise en relation de la politique **abstraite** avec les configurations du système *(Réseau)*
  - Choix de la *relation de conformité* *(Inclusion, équivalence, ...)*
- Travail motivé par les besoins de l'ACI sécurité POTESTAT

(POTESTAT : POlitiques de sécurité: TEST et Analyse par le Test de systèmes en réseau ouvert)

Résultat attendu :

Programme exécutable qui surveille un réseau :
- Produit une alerte pour tout message du réseau qui n'est pas conforme à la politique
- Peut être utilisé sans connaissance des méthodes formelles
- Nécessite l'entrée des différentes configurations du réseau *(fichiers de configuration des serveurs, ...)*
- Différents niveaux d'abstraction, dont le plus haut est la politique de sécurité *(pour certification)*

Disposition physique du réseau :



Observation du trafic réseau

Message observé sur le réseau :

Machine 1, port 12305 ; Machine 2, port 22

Traduction du message reçu *(en termes de machines et ports)* en l'ensemble des utilisateurs pouvant accéder à la machine cliente et des services de la machine serveur. *(Utilisation de la configuration du système pour la traduction)*

Message traduit :

{(Jean Marie, Accès distant machine 2), (Alice, Accès distant machine 2)}

Test de conformité par rapport à la politique

Politique de sécurité :

```
{
 (Jean Marie, Mail-IMAG),
 (Jean Marie, Accès distant machine 2),
 (Jean Marie, NFS Machine 2),
 (Alice, Mail Machine 1),
 (Alice, NFS Machine 2)
}
```

(Choix de représentation : un ensemble contenant tous les accès autorisés sous la forme de couples Utilisateur-Service. Les autres accès sont interdits)

Résultat obtenu :

Alertes sûres = Ø

Alertes possibles = {(Alice, Accès distant machine 2)}

Description du modèle :

Notions introduites à chaque étape de raffinement :

AccessPolicy

message : couple ***(Utilisateur,Service)***
Politique et **Alerte** : listes de couples ***(Utilisateur,Service)***
*Principe : **Message** donne l'utilisateur et le service présents dans la communication*
Invariant 1.1 : $Alerte = \{message\} - Politique$

NetworkPolicy

Début de description de la configuration du réseau : ajout des ***Machines***, ***Comptes*** et ***Hébergé***.
Comptes : Association de chaque ***Machine*** à ses ***Utilisateurs***.
Hébergé : Association de chaque ***Machine*** aux ***Services*** qu'elle héberge.
Messages, **AlertesPossibles** et **AlertesSures** : listes de couples ***(Utilisateur,Service)***.
*Principe : Calcul de **Messages** à partir des informations reçues (Machines source et destination)*
Invariant 2.1 : $message \in Messages$
Invariant 2.2 : $AlertesPossibles = Messages - Politique$
Invariant 2.3 : $(Messages \cap Politique = \emptyset) \Rightarrow (AlertesSures = Messages)$
Invariant 2.4 : $(Messages \cap Politique \neq \emptyset) \Rightarrow (AlertesSures = \emptyset)$
Invariant 2.5 : $(AlertesSures \subseteq AlertesPossibles) \wedge (Alerte \subseteq AlertesPossibles)$

PacketPolicy

Description complète de la configuration des machines : ajout de ***Ports*** et ***RunOn***.
RunOn : Liste des ***Services*** tournant sur une ***Machine***, *sur un* ***Port*** *donné.*
*Principe : Calcul plus fin de **Messages** (informations reçues : Machines et ports source et destination)*
*Invariants 2.x conservés car on ne fait que préciser comment construire **Messages**.*

CheckPolicy

Représentation implantable des données.
Invariants 2.x et 4.x sont conservés car les données restent équivalentes.

Traduction automatique en C/C++/ADA + *Compilation*

CheckPolicy

Programme exécutable *garantissant les invariants.*

Méthode de récupération des configurations sous Fedora Core (Linux) :

Politique : *liste fournie par l'ingénieur réseau.*
Utilisateur : *fichier /etc/passwd*
Service : *fichier /etc/init.d*

Machines : *liste fournie par l'ingénieur réseau*
Comptes : *fichier /etc/passwd*
Hébergé : *fichier /etc/init.d*

RunOn : *fichier /etc/services*
Ports : *fichier /etc/services*

Résultats pratiques :

- Tout le modèle a été réalisé
- 626 obligations de preuves triviales
- 511 obligations de preuves non triviales
- 86 obligations de preuve non encore prouvées

Légende :
Raffinement
Preuve de raffinement
Preuve de cohérence

Bilan

- Surveillance de la conformité d'un réseau à une politique de haut niveau
- Conservation des propriétés de sécurité par raffinement
- Garantie de la pertinence des alertes et de la cohérence de la configuration
- Modèle formel aidant à la certification du produit

Perspectives

- Génération et analyse des fichiers de configuration
- Autres implantations plus optimisées
- Surveillance intelligente (stateful, inspection de paquets)
- Surveillance multi-niveau (routeur, serveur, station)