Rupture de protocole avec garanties de sécurité pour les systèmes de contrôle-commande

Nicolas Kox

Superviseurs : Jean-Louis Roch, Gilles Berger-Sabbatel, Vincent Danjean, Bernard Tourancheau

Equipes : Drakkar, MOAIS

Projet ARAMIS

June, 15 2015

Table des Matières

Cadre de la thèse

Le projet ARAMIS vise à développer un module pour analyser et sécuriser les communications au sein des système de contrôle-commande des grands réseaux et infrastructures : eau, gaz, électricité, nucléaire...

Dans un contexte géopolitique tendu, ces réseaux sont désormais la cible d'attaques de plus en plus sophistiquées :

Stuxnet fut le premier exemple de virus développé par des états, et visait à détruire des central d'enrichissement d'uranium.

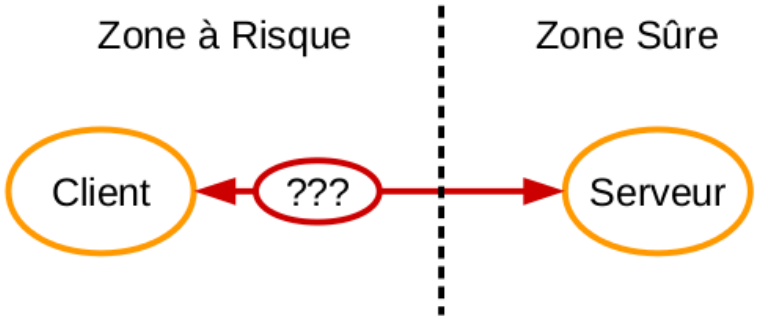
D'autre programmes ont suivi : Duqu, Flame, Shamoon, Mahdi...

Ces virus visaient des sites statégiques, en quête d'informations ou à des fins de destruction

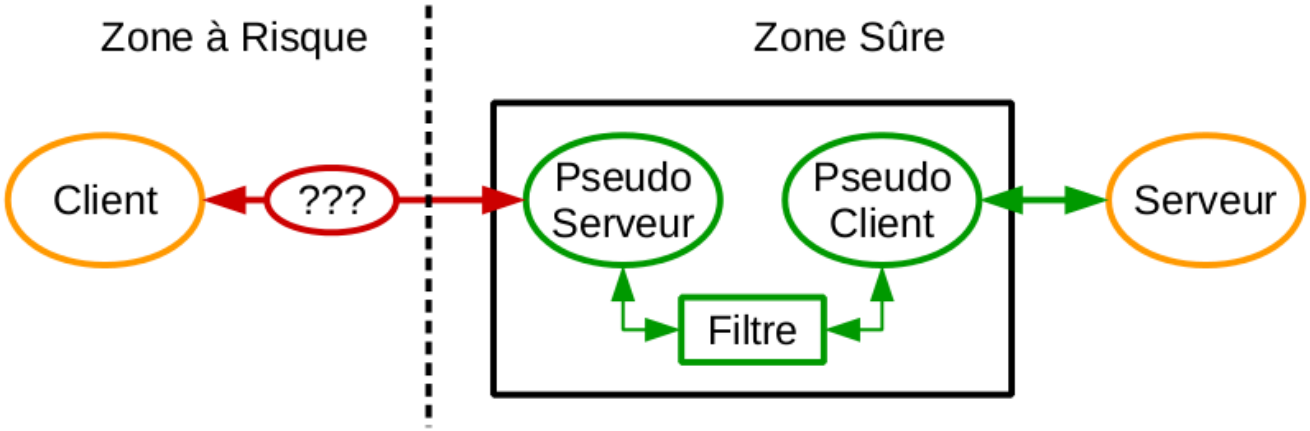
Parmi les acteurs de cette cyber-guerre, on retrouve un bon nombre de pays : USA, Iran, Israël, Royaume Uni, Chine, Russie, Corée du Nord, France...

Rupture de Protocole

En temps normal, un client établit une session avec un un serveur :



Dans notre cas, on veut intercepter cette session et créer deux sessions distinctes :



Le but est double :

- Empêcher les attaques liées à un protocole
- Inspecter le contenu des communications

Couches de protocole

On agit sur plusieurs couches de protocole du modèle OSI :

- Couche Réseau et Transport : TCP/IP
- Couche Session : SSH, TLS
- Couche applicative : Modbus, OPC-UA, FTP, SFTP

Pour chaque protocole, une paire client-serveur doit être développée selon le principe suivant :

- Le serveur interagit avec une zone sensible et doit donc résister à des violations de protocole, ainsi qu'à diverses attaques
- Le client interagit avec une zone sûre, mais vulnérable, et doit donc offrir une interface saine et *loyale*

Bien entendu, c'est règles s'appliquent en pratique tant au serveur qu'au client

Sécurisation des communications industrielles

Au niveau des protocoles industriels, on peut distinguer plusieurs points à traiter :

- Analyse lexicale : vérifier que les commandes sont bien écrites
- Analyse syntaxique : vérifier que l'enchaînement des messages est cohérent
- Transcription : réécriture des messages dans un langage interne, en vue d'une analyse sémantique

On se limite ici aux spécifications du protocole :
l'interprétation de ces commandes fait l'objet d'une autre thèse

Sécurisation du système

Mise en oeuvre des contre-mesures de sécurité connues :

- Protections contre l'exploitation des dépassements de tampon : DEP, ASLR
- Definition stricte des rôles et des autorisations d'accès : mise en oeuvre du Role-Based Access Control
- Utilisation des options de protections du compilateur : Stack-Smashing Protector

L'ensemble de ces fonctionnalités sont apportées par le patch *grsecurity*

Portage sur le Matériel utilisé

Le matériel qui sera utilisé est un processeur ARMv7 couplé à un FPGA.

Des pilotes et des spécifications du matériel sont disponibles pour configurer le noyau Linux

Pour la cross-compilation on utilise une toolchain basée sur gcc, bintuils et la Musl libc comme alternative à la Glibc

Pour les programmes de bases, on se limite au strict minimum.

L'utilisation de Busybox est pour le moment envisagée pour sa flexibilité et sa légèreté :

- Propose une version simplifiée des programmes d'un système Linux
- Les bibliothèques sont communes à tous les programmes
- La configuration est simplifiée par une interface de type *menuconfig*

Développements et Résultats

Travaux effectués

State-of-the-Art

- Noyaux et systèmes : étude des systèmes existant et analyse en terme de sécurité
- Réseaux et communications : etude des attaques existantes, contre-mesures, communications sécurisées (TLS, SSH...)
- Protocoles industriels : Modbus, OPC-UA

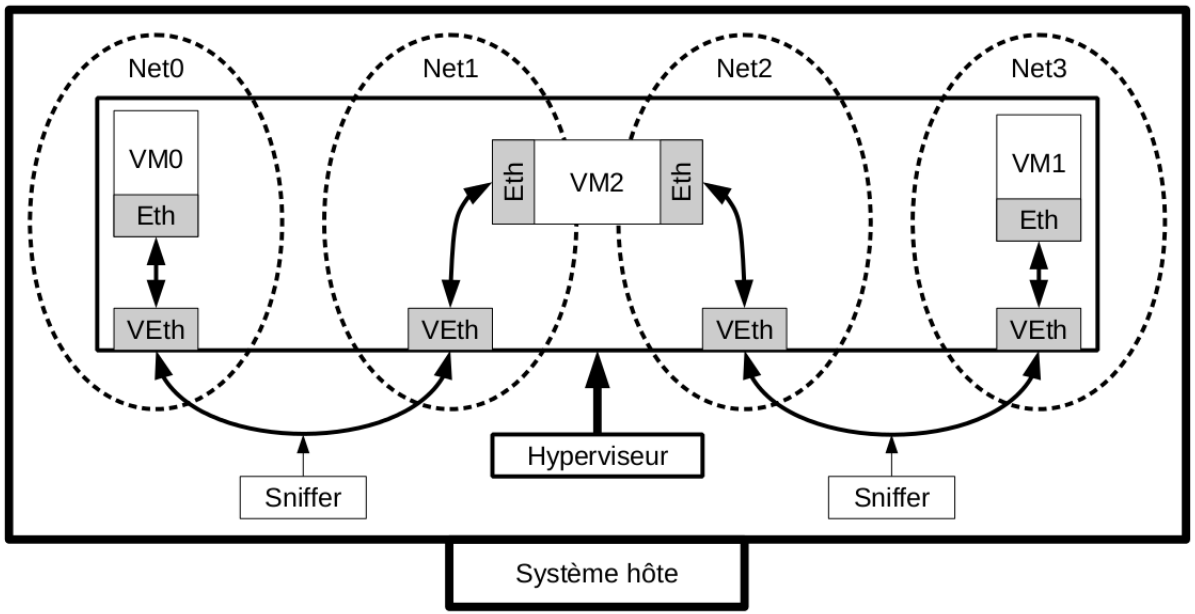
Ce travail a notamment abouti à un rapport concernant le choix du système dans le cadre du projet ARAMIS

Premier Prototype

Un pemier prototype déstiné à filtrer des communications Modbus a été développé.

Cela inclut plusieurs aspects techniques :

- Mise en place d'un réseau industriel virtuel, analyse de paquets de part et d'autre du prototype
- Solution dite de *Proxy Transparent*
- Analyse des trames Modbus, et verification des enchaînements requête-réponse



Travaux futurs

Travaux Futurs

Plusieurs axes de travail restent à étudier à présent :

- Mise en place d'un SYN Proxy pour prévenir les attques de type SYN Flood
- Etude de l'efficacité des solutions en terme de flux réseau
- Extension du prototype à d'autres protocoles : OPC-UA, SSH, TLS, SFTP
- Multiplexage du serveur : étude des différentes solutions (Polling, Threading ou Forking) en terme de sécurité et d'efficacité

Questions?