REPUBLIQUE DU CAMEROUN
Paix-Travail-Patrie

UNIVERSITE DE YAOUNDE I

CENTRE DE RECHERCHE ET DE FORMATION DOCTORALE EN SCIENCES, TECHNOLOGIE ET GEOSCIENCES

UNITE DE RECHERCHE ET DE FORMATION DOCTORALE EN MATHEMATIQUES, INFORMATIQUE, BIOINFORMATIQUE ET APPLICATIONS

REPUBLIC OF CAMEROON
Peace-Work-Fatherland

THE UNIVERSITY OF YAOUNDE I

POSTGRADUATE SCHOOL OF SCIENCE, TECHNOLOGY & GEOSCIENCES

RESEARCH & TRAINING UNIT FOR DOCTORATE IN MATHEMATICS, COMPUTER SCIENCES AND APPLICATIONS

DEPARTEMENT D'INFORMATIQUE
DEPARTMENT OF COMPUTER SCIENCES
Equipe SI & GL

APPROCHE DE COLLECTE DE PREUVES DANS UNE INVESTIGATION NUMERIQUE

THESE

Soutenue le 04 mars 2015 en vue de l'obtention du Doctorat/PhD en Informatique, par :

MBOUPDA MOYO Achille Mathurin

Matricule : 00U343
D.E.A . en Informatique

Devant le jury composé de :

Président: TCHUENTE Maurice, Professeur, Université de Yaoundé I

Rapporteur: ATSA ETOUNDI Roger, Maître de Conférences, Université de Yaoundé I

Membres:

- FOUDA NDJODO Marcel, Maître de Conférences, Université de Yaoundé I
- NDOUNDAM René, Maître de Conférences, Université de Yaoundé I
- TAYOU Clémentin, Maître de Conférences, Université de Dschang
- KOLYANG, Maître de Conférences, Université de Maroua

Dédicace

A ma famille : Chrystelle, Keryan, Emrys, Daina et Laurence LOUDJOP

Remerciements

Je remercie le seigneur Dieu tout puissant de m'avoir permis de présenter cette thèse. Je voudrais exprimer ma reconnaissance et ma gratitude :

- à tous mes enseignants de l'école maternelle, primaire, secondaire jusqu'à l'université ;

- au Professeur Atsa Etoundi Roger qui a cru en moi et m'a donné la possibilité de me reconstituer. Sa disponibilité, ses conseils, sa bienveillance ainsi que la confiance qu'il m'a accordée ont été précieux. Puisse ce travail être à la hauteur de ses attentes. Plus qu'un directeur, il a été pour moi, un guide, un aîné sur qui je pouvais compter à tout moment ;

- au Professeur Tchuente Maurice pour tous ses conseils et ses encouragements ;

- au Professeur Fouda Djodo Dominique qui a éveillé en moi l'esprit de la recherche et de la détermination ;

- au Département d'Informatique et à tous ses enseignants ainsi qu'à la Faculté des Sciences et à l'Université de Yaoundé I pour m'avoir accueilli après l'obtention de mon diplôme de Licence à l'Université de DSCHANG, et pour m'avoir formé jusqu'à ce jour.

- à la Gendarmerie Nationale et en particulier au Service Central des Recherches Judiciaires où j'ai non seulement eu de nombreux cas d'études mais également, le laboratoire et les outils forensics dont j'avais besoin pour mes expérimentations ;

- au Colonel Amadou Bahagobiri, Chargé d'Etudes à la Direction Centrale de la Coordination et Chef du Service Central des Recherches Judiciaires qui a su créer l'environnement favorable qui pouvait me permettre de poursuivre mes recherches ;

- à mes camarades de laboratoire notamment Nkoulou Onanena Georges et Nkondock Mi Bahanag Nicolas pour les discussions et l'ambiance qui ont régné entre nous tout au long de cette thèse ;

- à toute ma famille pour le soutien moral et financier qu'elle m'a apportée ;

- à mes amis Tagne Fute Elie, Tchabo Modeste, Fah Kue Moïse, Madjo Chrispin et l'Association "Ensemble" ;

- à mes collègues du Service Central des Recherches Judiciaires à la Gendarmerie Nationale, pour leurs inlassables encouragements ;

- à tous ceux que j'ai oublié et qui ont contribué de près ou de loin à la réussite de ce travail.

Table des matières

Liste des figures

Liste des tableaux

Résumé

La cybersécurité est devenue un enjeu majeur en matière de défense et de sécurité nationales. Le développement des Technologies de l'Information et de la Communication a créé un nouvel univers numérique dont la maîtrise est gage du développement socio-économique des Etats. Ce nouvel univers est en soi un espace de souveraineté des Etats qui doit être sécurisé et défendu au même titre que les espaces terrestre, aérien et maritime. Par ailleurs, les méthodes de protection doivent suivre le rythme de l'évolution de l'environnement des cyberattaques. Avec l'émergence d'attaques complexes, une nouvelle approche est nécessaire en matière de sécurité. Il ne suffit plus seulement de renforcer le dispositif sécuritaire de son système d'information mais, un Etat doit être capable d'identifier et d'appréhender les auteurs d'une attaque afin que ceux-ci répondent de leurs actes devant une juridiction nationale ou internationale.

Les techniques de détection des intrusions se sont muées en véritables dispositifs de prévention contre les attaques sur les systèmes d'information. Adaptés conséquemment, ils peuvent être capables d'aider à l'identification des auteurs de certains actes criminels. Malheureusement, les techniques mis en place jusqu'à ce jour ne se limitent qu'aux études comportementales des utilisateurs et approches par scénario. Les ressources qui sont les éléments fondamentaux des systèmes d'information sont mises à l'écart alors que leur exploitation est capitale dans la recherche des traces et indices qui constituent des preuves numériques à charge ou à décharge devant les juridictions. Afin que les Etats et les organisations continuent à entretenir l'atmosphère de confiance qui doit les caractériser, des modèles d'investigation numériques "digital forensic models" ont été élaborés. Malheureusement, ces modèles ont été conçus suivant la logique qu'ils ne peuvent se déployer qu'après qu'un incident se soit produit, alors que le modèle multidimensionnel que nous proposons est à la fois réactif, actif et proactif. Au regard des modèles précédents, tous les indices et preuves qui auraient pu soutenir le processus criminel en amont se trouvent écartés.

Cette thèse apporte une contribution à la détection des intrusions par une approche basée sur le comportement des ressources dans un système d'information. Ensuite, elle concours à la sécurisation des SI à travers la réalisation d'un modèle d'investigation numérique de large spectre capable de rechercher les indices et preuves numériques ainsi que les auteurs en vue de leur présentation devant une juridiction compétente. Après avoir présenté un état de l'art sur la cybercriminalité et l'investigation numérique, un cadre formel de l'investigation numérique après une intrusion est défini. Ceci passe d'abord par la conception d'un système d'information dans lequel les ressources sont les piliers, ensuite par la mise sur pied d'une méthode baptisée " resource behavior technique" qui s'appuie sur le comportement et le flux de travail entre les ressources pour détecter une attaque. Une technique d'investigation numérique multidimensionnelle qui permet d'élucider les attaques indépendamment d'une plateforme particulière est proposée. A cet égard, de nouveaux algorithmes qui améliorent les performances de l'outil de détection des intrusions de référence appelé $Snort$ ont été définis. Cette thèse se termine par des expérimentations permettant de valider l'approche proposée.

Mots clés : Détection des intrusions, investigation numérique, cybersécurité.

Abstract

Cyber security has become a major stake as concerns national defense and security. The development of Communication and Information Technologies has brought about a new digital universe which may be the pledge of the socio-economic development of States if it is mastered. This new universe in fact constitutes the sovereignty of the States which must be protected and defended as well as terrestrial, air and maritime spaces. In addition, the protection methods must match the rhythm of the evolution of the cyber-attack environment. With the emergence of complex attacks, a new approach is necessary as regards security. It is no more enough for a state to reinforce its security device system of information, but it should also be able to identify and apprehend the authors of an attack and prosecute them before a national or international jurisdiction.

The intrusion detection techniques were changed into true prevention devices against attacks on information systems. Consequently adapted, they proved to be able to contribute to the identification of the authors of certain criminal acts. Unfortunately, the techniques set up so far are limited to the behavioral studies of the users and scenario approaches. The resources which are the fundamental elements of the information systems are ruled out whereas their exploitation is essential in investigating the tracks and indications which make up the digital evidence which could be favorable or not before jurisdictions. For the States and the organizations to keep maintaining the atmosphere of confidence which must characterize them, some digital forensic models were worked out. Unfortunately, these models were designed with regards to the logic that they can come into play only after an incident has occurred, whereas the multidimensional model that we propose is at the same time reactive, active and proactive. In relation to previous models, all the indications and evidence which could have supported the criminal process upstream get ruled out.

This thesis contributes to the intrusion detections through an approach based on the behavior of the resources in an information system. Then, it partakes in securing Information Systems through the realization of a digital model of investigation of a broad spectrum able to seek the digital indications and evidence as well as the authors for their prosecution before a jurisdiction. After reviewing the literature on cyber criminality and digital investigation, a formal framework of digital investigation after an intrusion is defined. This passes initially through designing an information system in which resources are the pillars, then through the setting-up of a method named "resource behavior technique" which relies on the behavior of the resources to detect an attack. A technique of multidimensional digital investigation which helps to shed light on the attacks independently of a particular platform is proposed. In this respect, new algorithms which improve the performances of the detection tool for the intrusions of reference called $Snort$ were defined. This thesis ends in experiments making it possible to validate the suggested approach.

Key words : Intrusion detections, digital forensics, Cyber security.

Introduction Générale

Contexte

Un système d'information (SI) est un ensemble cohérent de ressources qui assemble, stocke, traite et fourni des informations pertinentes à une organisation de manière à ce que l'information soit disponible et utile à ceux qui souhaitent l'utiliser à temps opportun pour une prise de décision managériale [1]. Il s'agit en effet de la combinaison d'une technologie, des informations, d'un espace de travail et des Hommes organisés pour accomplir les différents objectifs d'une organisation. En d'autres termes, le SI est donc constitué du matériel, des logiciels, des bases de données, des procédures métiers et des Hommes. Dès lors, des logiciels applicatifs sont utilisés pour automatiser certaines tâches spécifiques des utilisateurs.

De nos jours, on distingue plusieurs types de systèmes d'information : SI de gestion, SI transactionnels, SI décisionnels, SI exécutifs, SI basés sur le travail collaboratif et les systèmes experts. Malgré cette multitude de SI, la constante récurrente demeure l'utilisation des données ou des informations pour une gestion stratégique de l'organisation. La donnée ou l'information est donc la ressource centrale et critique sans laquelle non seulement aucune décision ne saurait être prise, mais également aucune gestion stratégique ne saurait être faite. Cette donnée est la cible de plusieurs concurrents multiformes. Cette convoitise a amené les responsables de la sécurité des systèmes d'information (RSSI) des organisations à mettre en place des mesures permettant de les sécuriser.

Les SI font désormais partie intégrante du fonctionnement des administrations publiques étatiques et du mode de vie des citoyens. Les services qu'ils assurent sont devenus tout aussi indispensables que l'approvisionnement en eau potable ou en électricité. L'explosion mondiale du réseau internet a considérablement modifié la donne et conféré aux systèmes d'information une dimension incontournable au développement même de l'économie et de la société. Les SI sont devenus une nouvelle forme d'économie et de valeur de souverainneté. Les pionniers dans le domaine de la gestion des systèmes d'information seront ceux qui auront maitrisé la sécurité de leurs systèmes d'information dans tous ses contours.

SI comme outil de developpement

La guerre conventionnelle telle que connue au vingtième siècle est pratiquement en voie de disparition pour certains pays developpés. Les missions des armées ont changé, les aptitudes du fantassin d'aujourd'hui ne sont plus celles du soldat de la première ou de la deuxième guerre mondiale. Les équipements militaires (chars, avions, artillerie lourdes, navires et même fusils d'assaut) sont à présent bourrés des fonctions électroniques et connectés aux satellites ou au réseau internet. Les drônes (robots volants) qui effectuent des missions de bombardement sur des cibles au sol (cas des drônes américains en Afghanistan) viennent rappeler qu'il s'agit d'une réalité tangible et palpable. Il est possible que dans les années futures, les Hommes n'aient plus besoin de se déplacer physiquement pour aller en guerre. Les soldats-robots [14] (en préparation dans les laboratoires de l'armée américaine) iraient combattre en territoires ennemis, en lieu et place des Hommes. Au démeurant, les SI sont mis à contribution dans des domaines très variés tels que la surveillance électronique à travers les bracelets électroniques, l'enseignement à distance, le télétravail, la télémédécine, les formations en ligne ouvertes à tous (MOOC).

C'est ainsi que la maîtrise et le contrôle de son système d'information ainsi que de son cyber espace sont devenus un enjeu capital pour les Etats et les organisations. Ce défi demeure très difficile à relever du fait de l'évolution rapide des nouvelles technologies dont les vulnérabilités constituent des avantages concrets et exploitables par des esprits malveillants. Il est ainsi donné de constater la création d'un langage propre au référentiel cybernétique résultant de la transportation des menaces classiques vers ce nouveau monde. Il ne s'agit plus seulement des infractions classiques mais aussi des atteintes graves à la sûreté des Etats. A présent, les Etats s'affrontent à tout instant par réseaux informatiques interposés. Le réseau internet est devenu un lieu de confrontation militaire majeur. De nouveaux champs de bataille se sont créés avec comme cibles les systèmes d'information des organisations gouvernementales, des institutions, des petites et moyennes entreprises, les organisations privées et les particuliers. Les cyber-combattants sont les groupements de pirates informatiques, les organisations terroristes, les escrocs de tous genres mais aussi les membres des forces armées et les organisations gouvernementales. L'on assiste ainsi à la naissance d'une guerre non conventionnelle et essentiellement asymétrique face à laquelle les propriétaires des systèmes d'information (PSI) doivent prendre des mesures pour garantir leur souveraineté. Ces mesures consistent en la définition d'une politique globale de sécurité et de riposte qui permettrait d'assurer la continuité de service.

SI comme cible des attaques

Les technologies de l'information et de la communication sont devenues des cibles de la malveillance (vol d'ordinateurs ou de données, prise en otage de ressources informatiques, etc.) ou des moyens pour commettre des actes illicites (chantage, détournement, blanchiment d'argent, etc.) ou encore le lieu de sauvegarde des informations ayant servi ou qui serviront à la réalisation des actes prohibés. La dématérialisation des services et des transactions, les outils de mise en relation et de communication, la capacité

d'agir à distance et sous de fausses identités ou des identités usurpées, de passer par un grand nombre d'intermédiaires techniques (serveurs, fournisseurs d'accès, etc.) et de pays différents, facilitent des formes d'organisations, d'échanges et d'activités criminelles très profitables au regard de l'investissement nécessaire et du risque encouru.

Au-delà de leur intérêt pour les individus, les infrastructures informatiques et de communications électroniques sont aussi d'importantes ressources stratégiques pour les organisations ou les États. Elles sont la cible de cyber-attaques sur leur disponibilité, leur intégrité ou leur confidentialité. Elles peuvent également être utilisées pour manipuler l'opinion (endoctrination, diffusion de rumeurs, etc.), pour l'espionnage, pour la surveillance et le contrôle social, ou encore pour déstabiliser une économie, voire un Etat. Le réseau Internet est certes un fabuleux outil de communication, mais il est devenu aussi un instrument de pouvoir et une arme de guerre [2] tout comme les armes de destruction massive.

Par ailleurs, la sécurisation des systèmes d'information est subordonnée à la connaissance de l'ensemble des menaces auxquelles ils sont exposés. Les formes les plus récurrentes sont celles qui sont facilitées par les technologies de l'information et de la communication (TIC) et celles dont les TIC sont la cible. En effet, la criminalité classique étend son emprise par une large gamme de forfaits commis à travers le réseau Internet : escroqueries, fraudes, extorsions, abus, espionnages, vandalismes, conflits, harcèlements, etc. A ces termes, on peut désormais accoler le préfix cyber. La cybercriminalité recouvre ainsi toute activité illégale, irrégulière ou immorale réalisée à travers le cyberespace. Par extension, elle intègre toute forme de malveillance électronique effectuée au moyen de l'informatique et des télécommunications. Cette nouvelle forme de criminalité, dont l'ampleur est considérable mais encore mal chiffrée, appelle la société et les gouvernements à réagir.

En outre, la cybercriminalité se développe dans un contexte global de guerre économique permanente, de recherche de profit immédiat, de crise financière internationale, d'injustice sociale, de risques écologiques, sans vision à long terme ni parfois de gouvernance dans de nombreux pays. Sur Internet, tout s'achète et tout se vend, y compris la découverte de vulnérabilités des systèmes informatiques, les outils d'attaques informatiques, les données personnelles, les identités et identifiants, etc. Enfin, la loi qui s'impose sur Internet est celle des acteurs les plus forts.

Au démeurant, plusieurs cas d'actualités montrent à suffisance l'importance des SI et comment ils peuvent être utilisés si les contrôles suffisants ne sont pas faits lors de leur mise en oeuvre et de leur exploitation. L'affaire PRISM dévoilée par M. Edward Snowden qui deffraye encore la chronique a crée des tensions entre les USA, l'Europe et la Russie. Les écoutes telephoniques de monsieur SARKOZI Nicolas, alors qu'il était encore Président de la République française, sont des illustrations de l'utilisation de la technologie aux fins de destabilisation. C'est pour cela que certains pays bien avisés tels que les USA investissent des milliards de dollards à l'effet de maîtriser et sécuriser leurs SI, mais aussi de contrôler les SI des autres pays. L'Agence Nationale de la Sécurité américaine (NSA) à qui est dévolue cette tâche, en est une illustration forte.

Cet aperçu permet de comprendre l'étendue et la complexité du problème de la lutte contre la cybercriminalité. Ce combat dépend de la volonté des pouvoirs publics

animée par des contributions scientifiques, et devrait se fonder sur une approche globale, au service d'une vision partagée de la sécurité publique, pour une protection efficace des personnes et leurs biens, des nations et des valeurs.

Problématique

Le concept de sécurité des systèmes d'information recouvre un ensemble de méthodes, techniques et outils destinés à protéger les ressources d'un système d'information afin d'assurer la disponibilité des services. Elle consiste en la mise en place d'une politique de sécurité ; au déploiement des outils pour identifier, contrôler les différents types d'intrusions et prévenir les attaques ; ainsi qu'en la définition des bonnes pratiques pour garantir l'intégrité des données en transit et la sécurisation des accès Internet. Malheureusement, dans la conception des SI, les objectifs sont tellement axés sur l'accomplissement de leurs fonctionnalités de telle sorte que les aspects liés à la sécurité sont négligés. Les moyens techniques, organisationnels, juridiques et humains nécessaires à mettre en place pour conserver, rétablir et garantir la sécurité de l'information et du SI restent faibles. Ainsi, le système final présente des failles qui seront exploitées par des personnes malveillantes.

Le développement de l'informatique s'est accompagné des problèmes de sécurité. À l'origine, les virus se propageaient lentement, par l'échange de supports informatiques. Avec l'apparition des premiers réseaux TCP/IP, les problèmes de sécurité se sont diversifiés et ont conduit au développement de nouvelles techniques de sécurité. Très tôt dans le développement d'Internet, les vulnérabilités sur les systèmes d'exploitation ont permis à des attaquants de se déplacer virtuellement de système en système. Dans le contexte militaire de déploiement des réseaux TCP/ IP, la détection des actions malveillantes est rapidement devenue une nécessité. Les mesures de prévention se sont révélées insuffisantes et ont amené la création de systèmes de détection d'intrusions (IDS). Ces systèmes ont été développés dans le but de détecter des fonctionnements anormaux des systèmes d'information et des réseaux, indiquant que des actions non conformes à la politique de sécurité sont menées par un ou plusieurs utilisateurs. Deux familles de techniques d'analyse ont été élaborées pour effectuer cette détection. La première famille de techniques d'analyse postule que l'on peut différencier le comportement d'un attaquant du comportement habituel du système d'information surveillé : c'est l'approche comportementale. La deuxième famille exploite la connaissance accumulée sur les vulnérabilités et les manières de pénétrer les systèmes d'information : c'est l'approche par scénarios. De nos jours, le problème de la détection des intrusions dans un SI reste d'actualité du fait de l'inefficacité des méthodes de détection utilisées. Cette inefficacité est corroborée par le fait que dans le cas des systèmes informatiques qui constituent l'essentiel d'un SI, le comportement de la ressource attaquée n'est pas pris en compte par aucune des deux approches précédement citée pourtant, ce sont les ressources qui traitent les données dans un SI. Les outils de détection des intrusions permettent la collecte de nombreuses informations liées aux attaques. Malheureusement, aucun de ses outils ne permet la collecte de ces informations de manière à ce qu'elles soient utilisées comme preuves numériques devant une juridiction. L'adaptation des outils de détection des intrusions aux fins judi-

ciaires en préservant leur fonctionnalité première demeure un problème ouvert.

Il s'avère que la garantie de pouvoir utiliser des techniques d'investigation numérique pour appréhender les auteurs d'une agression ou d'une tentative d'agression d'un SI constitue également un moyen efficace de sécurisation de celui-ci. Dans le but d'élucider de nombreux cas d'agressions sur ou au moyen des SI, des travaux ont été menés partant de la restauration des données effacées [3] jusqu'à la mise sur pied des modèles d'investigations numériques. Malheureusement, tous ses travaux se sont limités à la définition des processus d'enquête numérique qui ne débutent qu'après qu'un incident se soit produit. Pendant que certains chercheurs ont évoqué la possibilité d'une investigation numérique en " live "c'est-à-dire pendant que l'attaque a lieu [4, 5], d'autres ont mentionné la nécessité d'une collection des éléments de preuves de manière préventive [6, 7]. En réaction à ces modèles d'investigations numériques, les cybercriminels développent à présent des techniques dites " anti-forensics " [**?**, 8], c'est à dire anti-investigation. Par contre, les modèles d'investigation numérique existants sont hiérarchisés ou décomposés en processus sans qu'aucune description détaillée ne soit faite ni sur les composants de chaque processus, ni sur la technologie, encore moins sur la localisation des ressources nécessaires au fonctionnement de ces composants.

La légalité et la recevabilité des informations constituant des preuves numériques devant une juridiction reste encore un débat selon le processus utilisé pour leur collecte [9, 10]. En effet, lorsque les experts sont amenés à expertiser un objet comportant des éléments d'information numérique, ils se posent de nombreuses questions, notamment sur le processus de collecte de ces informations, sur leur conservation, sur les vérifications et sur les précautions à prendre [11, 12]. Alors que la fugacité des données informatiques et numériques multiplie les difficultés de recueil et de conservation de la preuve, la nature immatérielle de ces données entraîne un effacement problématique des frontières. Cette nature immatérielle oblige aussi à reconsidérer les notions de souveraineté des PSI, face à l'extraterritorialité des données qui ne sont pas toujours constituées dans le lieu de commission de l'infraction car, la cybercriminalité s'épanouit dans des réseaux numériques complexes et pose des questions juridiques nouvelles pour les pénalistes [9]. Cependant, l'aspect légal de l'investigation numérique ne fait pas parti de ces travaux.

L'efficacité d'une investigation numérique est un facteur déterminant pour l'emergence d'une entreprise et des investisseurs dont les activités sont basées sur l'utilisation de l'internet. Dans la plupart des délits actuels, une partie des preuves sont aujourd'hui des données numériques. A cet effet, lorsqu'une infraction impliquant un support informatique est suspectée, une intervention rapide et méthodique doit être engagée afin de délimiter et préserver la scène du cybercrime. Ces procédures sont indispensables, avant toute analyse, pour garantir la maîtrise et la recevabilité des informations collectées et traitées en cas de poursuites judiciaires.

Cependant, les officiers de police judiciaire ou tout autre intervenant dans la chaine judiciaire font face à une masse de données importantes à analyser et provenant d'outils numériques de plus en plus divers et parfois non physiquement accessible. Les enquêteurs peinent à collecter, analyser et préserver ces données convenablement. Il est donc nécessaire de déployer une démarche d'investigation qui permettra aux enquêteurs de travailler en toute sécurité. Le focus de ces actions se situe dans le domaine de l'investi-

gation numérique.

Le challenge dans la gestion des SI consiste en la définition d'une procédure d'investigation numérique visant à apporter les eléments matériels des infractions perpétrées dans les SI ainsi qu'à en déterminer les auteurs, en les accompagnant avec des preuves numériques irréfutables. L'investigation intègre la gestion des attaques et la détection des intrusions qui se résument à la triade proaction, action et réaction. Chacune de ces opérations se fonde sur la collecte des preuves qui pourront établir les faits incriminés dans le cadre d'une action en justice.

L'objectif principal de cette thèse est résumé ainsi :

"*Apporter une réponse à l'insécurité dans les systèmes d'information en réalisant un modèle d'investigation numérique dans la phase policière d'une enquête judiciaire*.

Ce faisant, ce travail montre non seulement en quoi les technologies de l'information et de la communication favorisent les dérives délictueuses et criminelles, mais aussi, nous participons à la lutte contre ces dérives qui s'inscrivent dans des espaces mondialisés et le plus souvent virtuels. De ce fait, la solution proposée vient compléter les moyens actuels de sécurisation des SI par détection des intrusions puis, d'investigation des crimes conventionels auxquels ont recours les justices et les officiers de police judiciaire pour poursuivre les cybercriminels.

Contribution

Cette thèse contribue à la résolution du problème sus évoqué en présentant un modèle d'investigation en milieu numérique qui permet d'obtenir des preuves numériques inaltérables, irréfutables et opposables. Ce faisant, les détails sur la collecte, l'analyse et la préservation des données sont apportés dans le but de garantir l'originalité et l'incontestabilité des preuves collectées.

Ce travail propose dans la lutte contre la cybercriminalité, une approche multidimensionnelle de l'investigation en milieu numérique. Il s'agit d'une investigation numérique préventive (Proactive Digital Forensics : ProDF), en situation de flagrance (Active Digital Forensics : ActDF) et après qu'une attaque ait eu lieu (Reactive Digital Forensics : ReaDF). Le déploiement de la solution proposée s'appuie sur la détection des intrusions dans un environnement hétérogène. L'approche proposée prendra son fondement sur le workflow mining et le business process.

Au terme de ce travail, les contributions suivantes seront obtenues :

1. La définition d'un cadre formel du système d'information basée sur la technique du "resource behaviors" qui prend en compte l'ensemble des ressources opérationnelles. Les attaques sur un système d'information se traduisent par un comportement inattendu de certaines ressources qui le constitue. La mise en place d'une politique de sécurité permet de comparer à tout instant l'état courant de chaque ressource à son état normal dans l'optique de détecter une attaque.

2. Une modélisation du processus de l'investigation numérique dans son entièreté, sous le prisme des aspects proactif, actif et réactif. Il s'ensuit une description de chaque composant y compris les ressources dont il a besoin. Le modèle obtenu permet la collecte des informations numériques qui constituent les preuves numériques disculpantes,indicatives ou corroborative, en vue d'être présentées devant une juridiction.
3. La création d'un prototype à travers lequel les informations obtenues par les systèmes de détection/prévention des intrusions sont transformées en preuves numériques. Ceci conduit au raffinement des règles de détection des intrusions et à l'extension de $Snort$, outil de détection des intrusions d'expérimentation.

Plan du document

Après la présentation des objectifs de nos travaux, cette section introduit le plan de la thèse. Ce document est organisé en trois chapitres qui se résument ainsi qu'il suit :

Chapitre 1 : Cybercriminalité et investigation numérique

Dans ce chapitre, un état de l'art est fait sur la cybercriminalité. Il s'agit en effet d'une forme de criminalité qui connaît actuellement la croissance la plus forte, stimulée par le progrès des technologies de l'information et de la communication [13]. Malheureusement, les auteurs des actes malveillants sur les SI bénéficient encore de certaines conditions favorables à la réalisation de leurs activités. Il s'agit en l'occurence de la dématérialisation des transactions, la mise en réseau des ressources, la disponibilité d'outils d'exploitation des failles et vulnérabilités et de l'aterritorialité de l'internet [14].

Nous mettons en exergue certains cas de cybercriminalité, catégorisons les cybercriminels en fonction des types d'infractions puis, étudions leurs modes opératoires avant de nous interesser aux différentes problématiques. L'analyse de ces dernières nous conduit à l'étude des méthodes actuelles de riposte à la cybercriminalité. Nous procédons ainsi à l'état de l'art sur l'une des approches de lutte contre la cybercriminalité qu'est l'investigation numérique et constatons que les travaux actuels ont permis d'obtenir plusieurs parades contre les actes cybercriminels perpétrés dans le cyberespace [4, 13, 19]. Cependant, malgré l'abondance de ces travaux, les auteurs de certains actes malveillants sont restés inconnus ou alors, les preuves n'ont pas pû être collectées à l'effet d'incriminer un tiers. En effet, la recherche et la collecte de la preuve au cours d'une investigation numérique présente encore de nombreuses difficultés. Ces dernières sont d'autant plus grandes lorsque les auteurs ont utilisé des logiciels ou matériels qui facilitent l'anonymisation ou encore des techniques antiforensics ou anti-IDS. Le Cloud-forensic en particulier reste très difficile à effectuer alors que l'informatique dématérialisé (cloud-computing) est en pleine évolution [20].

Tout comme pour la criminalité traditionnelle, les techniques d'investigation en matière de lutte contre la cybercriminalité existent mais ne demeurent pas assez efficaces. A cet égard, il n'est pas évident de déterminer les auteurs des actes malveillants perpétrés dans un SI. De plus l'on ne saurait déclencher des actions pénales, si oui contre inconnus,

sans aucune garantie sur l'aboutissement de l'enquête. A la fin de ce chapitre, nous nous focalisons sur une approche de cybersécurité par détection des intrusions étant donné que les outils de détection des intrusions sont très utilisés dans les SI actuels et, peuvent constituer un réservoir potentiel de preuves numériques [24, 29].

Chapitre 2 : Cadre formel de la détection des intrusions

Parmi les problèmes de sécurité qui minent les systèmes d'information, la question de la détection des intrusions est le défi majeur qui doit être discuté en priorité parce que, toutes les intrusions commencent par une attaque et celle-ci se compose de diverses activités malicieuses. De nombreux travaux existent dans le domaine de la détection des intrusions [21, 22, 23, 24]. Ces travaux s'articulent autour de l'approche par scénario et de l'approche comportementale. Cependant, dans la deuxième approche, le comportement des ressources, élément essentiel d'un SI, n'est pas pris en compte en vue de la détection d'une attaque. Plus encore, la qualité du flux d'informations échangées entre diverses ressources d'un SI n'a quant à elle jamais été étudiée pourtant, ce workflow est une source abondante d'informations permettant de déceler la présence d'une attaque.

Par une approche basée sur le workflow mining, nous prenons en compte dans ce chapitre, les travaux de [25] pour définir un cadre formel du système d'information dans lequel la ressource est le pilier essentiel. Ainsi, la détection d'une attaque est traduite par un comportement inattendu des ressources en violation de la politique de sécurité mise sur pied. Ainsi, les fichiers logs associés à ces ressources peuvent être exploités pour servir de preuves dans une investigation numérique. Le modèle obtenu apporte une valeur ajoutée pour la détection des intrusions et la collecte des preuves dans un système d'information.

Chapitre 3 : Approche méthodologique de collecte de preuves numériques

Une fois qu'une intrusion est détectée, il est nécessaire de prendre des mesures pour collecter les preuves utiles non seulement afin d'apporter les éléments matériels de la constitution de l'infraction, mais aussi pour établir la responsabilité des suspects. Cela n'est possible que par une investigation numérique.

De nombreux travaux ont été conduits dans le domaine de l'investigation numérique [3, 4, 6]. Les modèles d'investigations qui en résultent sont basés sur un ensemble d'activités qui doivent être accomplies pour obtenir les preuves numériques nécessaires dans le cadre des poursuites judiciaires. Au regard des travaux déjà effectués dans ce domaine, deux processus d'investigations ont été mis en relief en fonction des cas de figure. Il s'agit des processus en mode actif et réactif. Malheureusement, aucun des modèles d'investigations susmentionnés n'envisage non seulement la possibilité d'un processus en mode proactif mais également l'exécution simultanée de tous ses processus malgré qu'ils soient liés les uns aux autres dans le cadre de la gestion d'un incident dans une organisation. Ceci se justifie par le fait que les modèles jusqu'alors proposés n'ont pas été construits sur la base d'une étude et d'une conception préalable en utilisant les outils appropriés pour leur optimisation.

Dans ce chapitre, le problème de l'investigation numérique est vu comme un en-

semble de processus métiers qui doivent être accomplis pour permettre la collecte des preuves numériques et déterminer les auteurs d'une aggression sur un SI. Il en découle des procédures d'investigations composées de primitives d'investigations, elles mêmes constituées des actions à réaliser pour élucider une attaque. Prenant en compte les limites des modèles existants, nous nous appuyons sur les travaux de [4, 6, 7] pour mettre en œuvre un modèle d'investigation numérique multidimensionnel à trois composantes : proactive, active et réactive. De ce fait, l'effectivité d'une investigation numérique proactive qui vient soutenir selon le cas d'étude, la recherche et la collecte de la preuve numérique dans une investigation active ou réactive, est alors établie.

Chapitre 4 : Etude de cas - collecte de preuves numériques par détection des intrusions

Les techniques de détection des intrusions sont apparues pour inspecter les activités entrantes ou sortantes d'un réseau et identifier les cas suspects qui indiqueraient une attaque susceptible de compromettre un système d'information. Malheureusement, bien que les systèmes de détection des intrusions soient une source fournie d'informations sur les attaques, les travaux jusqu'à lors effectués n'ont pas établi que ces informations pouvaient être utilisées comme preuve numérique devant une juridiction [26, 27]. En effet, la difficulté principale est de pouvoir offrir aux IDS la capacité de produire des informations forensics sans changer leur mission première qui est la détection des intrusions.

Dans ce chapitre, la version libre de l'outil de détection des intrusions $Snort$ est prise comme logiciel de référence. Nous nous interessons aux extrants des IDS et leur relation avec les preuves numériques. Nous modifions l'architecture fonctionnelle de $Snort$ de telle enseigne qu'en plus de ses données binaires produites habituellement en sortie après la détection des attaques, $Snort$ fourni un fichier journal qui est pris en entrée par la composante d'investigation Proactive ProDF. L'exécution des algorithmes de cette composante conduit à l'identification, la collecte, la préservation, l'analyse et la documentation des preuves inhérentes aux intrusions détectées. A l'issue, nous procédons à des tests de détection des intrusions et de collecte de preuves numériques. Ces tests établissent que les potentialités de $Snort$ n'ont pas été altérées malgré l'intégration en son sein de la composante d'investigation numérique ProDF.

Conclusion générale

Cette thèse s'achève par un bilan des travaux de recherche. Nous dressons ensuite un ensemble de perspectives pouvant être développées, ouvrant ainsi de nouvelles voies et propositions dans le but d'améliorer ce travail.

CHAPITRE 1

CYBERCRIMINALITÉ ET INVESTIGATION NUMÉRIQUE

1.1 Introduction

De nos jours, toute organisation possédant des ordinateurs et des équipements réseaux inter-connectés fait face à un flux permanent de reconnaissances et d'attaques. Les cybercriminels, à travers de milliers d'ordinateurs dont ils ont pris le contrôle, exécutent des scripts et des programmes spécifiques pour analyser en permanence toutes les adresses IP accessibles. Leur objectif est de cartographier les périmètres internet, d'inventorier les parcs informatiques et de collecter des informations sur les failles inhérentes aux logiciels et à la configuration des terminaux. Malgré une analyse de risque permettant aux défenseurs des systèmes d'information d'identifier les scénarii d'attaques et par conséquent de mettre en place les parades nécessaires à la protection des informations, les attaquants démontrent chaque jour qu'ils progressent sur un terrain facile à apprivoiser.

Cependant, maîtriser la sécurité d'un SI implique que chaque incident face l'objet d'une investigation scrupuleuse afin de minimiser les pertes induites, de constater les infractions, d'en rassembler les preuves, d'en rechercher les auteurs et complices en vue de leur présentation devant une juridiction [28]. Les infractions sur ou à travers les SI ont longtemps existé mais, les organisations ont pris un sérieux retard dans la recherche des solutions d'investigation. L'investigation numérique est une activité particulièrement nouvelle étant donné qu'il s'agit d'un acte judiciaire dont les acteurs qui sont les officiers de police judiciaire, les magistrats, les avocats et les huissiers de justice ne reçoivent pas de formations appropriées pour le cas particulier du Cameroun. De nombreux modèles d'investigation existent. Malheureusement, ils sont tous confinés à la recherche de la preuve numérique en mode réactif, c'est à dire qu'elles ne prennent effet qu'après qu'un incident ce soit produit. Plus encore, il n'existe pas un cadre formel ou un standard qui régi le déroulement d'une investigation numérique.

A la suite de la partie introductive de ce chapitre, nous donnons dans la deuxième partie un panorama de la cybercriminalité en ressortant les problématiques. Dans la troisième partie, nous faisons l'état de l'art de l'investigation numérique ainsi que les défis

à y relever. La quatrième partie quant à elle s'appesentie sur l'approche de protection par les systèmes de détection des intrusions (IDS). Elle présente en outre les travaux effectués pour rendre les IDS suffisamment contributifs dans une investigation numérique, ainsi que les limites desdits travaux. Ce chapitre s'achève par une synthèse à la cinquième partie.

1.2 Cybercriminalité

Dans cette partie, l'expérience acquise dans la conduite des enquêtes policières traditionnelles guidera l'analyse des activités cybercriminelles. Ainsi, une similitude sera faite entre le crime et le cybercrime, la scène de crime et le cyberespace, la victime et la cyber- victime, l'arme du crime et un ordinateur ou Internet, le mode opératoire et la cyberattaque.

L'Organisation de Coopération et de Développement Economiques (OCDE) a défini en 1983 l'infraction informatique comme étant tout comportement illégal, immoral ou non autorisé qui implique la transmission et/ou le traitement automatique de données [9, 14]. Un crime informatique est une infraction pour laquelle un système informatique est l'objet du délit et/ou le moyen de le réaliser. C'est un crime lié aux technologies du numérique qui fait partie de la criminalité en col blanc. Le cybercrime est une forme de crimes informatiques faisant appel aux technologies de l'internet pour leur réalisation. Il s'agit de tous les délits réalisés dans le cyberespace (figure 1.1).

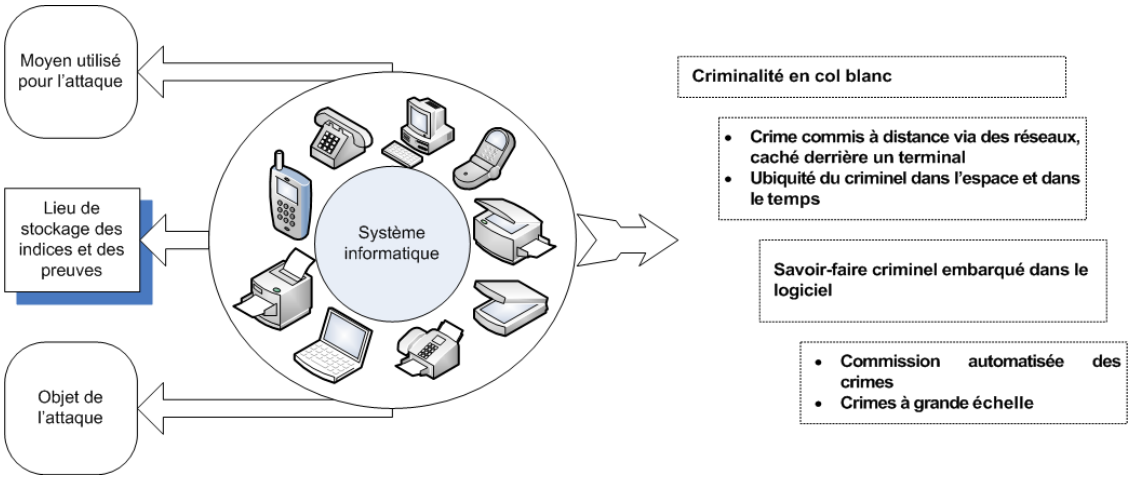


Figure 1.1 – Caractéristiques du crime informatique

Les actes de cybercriminalité qui se composent entre autres de l'usurpation d'identité, le leurre de systèmes, l'accès indu, l'exploitation frauduleuse des ressources, l'infection, la détérioration, la destruction, la modification, la divulgation, le déni de service, le vol, le chantage, etc. mettent en évidence les limites des approches sécuritaires actuelles. Quelles que soient les motivations des acteurs de la criminalité informatique, celle-ci engendre toujours des conséquences économiques non négligeables et constitue dans sa

dimension de cybercriminalité un fléau grandissant, transfrontalier et complexe.

La cybercriminalité constitue le prolongement naturel de l'expression de la criminalité classique. De nos jours, les activités criminelles s'effectuent à travers le cyberespace, par d'autres moyens que ceux habituellement mis en œuvre, et de manière complémentaire à la criminalité classique [28]. Non seulement internet offre des conditions exceptionnelles pour de nouvelles entreprises et activités illicites, il autorise également la réalisation de fraudes ou délits habituels via l'outil informatique. Une exploitation efficace des nouvelles technologies permet aux criminels de prendre des risques négligeables pour commettre des délits par lesquels ils réalisent des bénéfices considérables.

D'après les statistiques du service central des recherches judiciaires à la Gendarmerie Nationale, au Cameroun, les pratiques des cybercriminels sont récurrentes dans l'escroquerie et l'abus de confiance sur les réseaux, la contrefaçon appelée vulgairement « piraterie », les intrusions ou « hacking » et la fraude sur les cartes de crédit. Les victimes se comptent parmi les chefs de mission diplomatique du Cameroun à l'étranger, les grandes chancelleries camerounaises et étrangères, les représentants des Organisations Internationales du Cameroun, les hommes d'affaires étrangers résidant au Cameroun, les hauts commis de l'Etat, les personnalités ressources de la République, les sociétés publiques, parapubliques et privées, les hauts responsables des forces de sécurité et de défense.

La sécurité intérieure d'un pays est aujourd'hui confrontée à des formes d'expression de menaces criminelles liées à l'existence des technologies de l'information. Les technologies de l'internet sont au coeur de la guerre de l'information dont les enjeux sont d'ordre économique, politique, sociale et diplomatique. Internet permet non seulement la manipulation de l'information mais, il est aussi un outil privilégié pour répondre à des rumeurs ou toute forme d'intoxication et de campagne de déstabilisation. De même, sont facilitées les activités d'espionnage et de renseignement, puisqu'il est devenu aisé d'intercepter des informations transférées sur ce reseau [29].

1.2.1 Profil du cybercriminel

Les cybercriminels agissent suivant un profil bien déterminé. Selon des types d'infractions, les études [33, 35] ont mené à l'identification de 06 catégories de cybercriminels.

Agresseurs : les deux profils d'agresseurs les plus souvent identifiés sont :

- hacker ou passionné : individu curieux, qui cherche à se faire plaisir. Pirate par jeu ou par défi, il ne nuit pas intentionnellement et possède souvent un code d'honneur et de conduite. En général il n'a pas conscience de la mesure de ses actes. L'agresseur passionné n'est pas très expérimenté.
- cracker ou casseur : plus dangereux que le hacker, cherche à nuire et montrer qu'il est le plus fort. Souvent mal dans sa peau et dans son environnement, il peut causer de nombreux dégâts en cherchant à se venger d'une société (ou d'individus) qui l'a rejeté ou qu'il déteste. Il veut prouver sa supériorité et fait partie des clubs où il peut échanger des informations avec ses semblables.

Fraudeurs : le fraudeur bénéficiant souvent d'une complicité volontaire ou non chez ses victimes, il cherche à gagner de l'argent par tous les moyens. Son profil

est proche de celui du malfaiteur traditionnel. Parfois lié au grand banditisme organisé, il peut attaquer une banque, falsifier des cartes de crédit ou se placer sur des réseaux de transferts de fonds et, si c'est un particulier, il peut vouloir falsifier sa facture d'électricité ou de téléphone.

Employés malveillants (le fraudeur interne) : possédant de bonnes compétences sur le plan technique, il est souvent informaticien et sans antécédents judiciaires. Il peut penser que ses qualités ne sont pas reconnues, qu'il n'est pas apprécié à sa juste valeur. Il veut se venger de son employeur et chercher à lui nuire en lui faisant perdre de l'argent. Il peut répondre à un besoin matériel personnel qui induit des conduites de dépendances (jeux, sexe, etc.). Pour parvenir à ses fins, il possède les moyens, qu'il connaît parfaitement, et qui ont été mis à sa disposition par son entreprise.

Militants : motivés par une idéologie ou la religion, ils disposent de compétences techniques très variables. Leurs objectifs peuvent être limités à la diffusion massive de messages, comme ils peuvent s'étendre à des nuisances effectives sur les systèmes d'information des organismes en opposition avec leur idéologie.

Espions : ils participent à la guerre économique. Ils travaillent pour un Etat ou pour un concurrent. Ils sont patients et motivés. Ils savent garder le secret de leur réussite pour ne pas éveiller les soupçons et continuer leur travail dans l'ombre. Ils agissent souvent depuis l'intérieur de l'organisme, soit en ayant trouvé un moyen d'y pénétrer, soit en soudoyant une personne ayant accès aux biens. Ils ont pour but de voler des informations ou de détruire des données stratégiques (vitales) pour l'organisme. Dans tous les cas, les espions ont un excellent niveau de maîtrise de soi, ainsi qu'une grande capacité d'adaptation aux environnements.

Terroristes : souvent appelés les cyber-terroristes, moins courants, les terroristes sont aidés dans leur tâche par l'interconnexion et l'ouverture croissante des réseaux : très motivés, ils veulent faire peur et faire parler d'eux. Les actions se veulent spectaculaires, influentes, destructrices, meurtrières. Ce profil est pris de plus en plus au sérieux par les Etats depuis l'attentat du 11 septembre 2001. Ils considèrent qu'une cyber-attaque perpétrée par un terroriste pourrait gravement nuire aux infrastructures économiques et critiques d'un Etat devenu très dépendant de ses systèmes d'informations vitaux.

Au Cameroun, les profils des cybercriminels correspondent le plus souvent à celui des agresseurs, fraudeurs et employés malveillants.

1.2.1.1 Type du cybercriminel

Parvenir à distinguer la motivation du cyberdélinquant, ainsi que son niveau de technicité, permet, d'évaluer la gravité d'une attaque, de mieux la contrer et de prévoir quelles approches permettraient de collecter le maximum de preuves. Sécuriser un système d'information nécessite de connaître contre qui l'on doit se protéger. De nos jours, on observe deux grands types de cyberdélinquants tels qu'illustrés dans la figure 1.2. Il s'agit des professionnels dont les activités sont directement rémunératrices et les amateurs généralement animés par un fort besoin de reconnaissance sociale [14, 35].

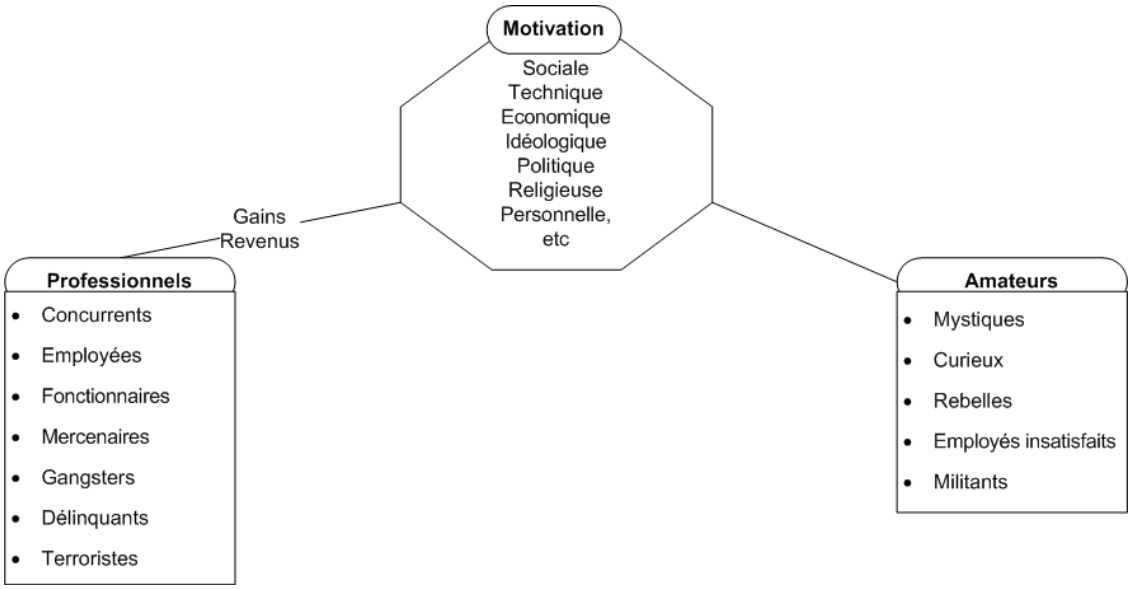


Figure 1.2 – Deux grandes familles de cyberdélinquants

Les professionnels sont généralement :
— des concurrents directs de l'organisation visée ;
— des fonctionnaires au service de leur Etat ;
— des mercenaires (pouvant agir aussi bien pour le compte d'institutions privées que publiques) ;
— des truands de toutes sortes.

Parmi les amateurs, se reconnaissent :
— les techniciens, successeurs des premiers passionnés, ces hackers des premiers âges dont la motivation essentielle était le désir de maîtriser toujours mieux les technologies ;
— les curieux ;
— les immatures : souvent appelés "script-kiddies" ou "kiddiots". Ils sont fréquemment sur le devant de la scène après avoir été attrapés. Le fait qu'ils soient pris par les forces de l'ordre ne signifie pas nécessairement qu'ils sont les seuls cyberdélinquants ;
— les psychopathes ;
— les militants, mus par idéologie ou religion, qui sont d'ailleurs souvent à cheval entre amateurisme et professionnalisme.

Au regard des statistiques du service central des recherches judiciaires de la Gendarmerie Nationale, il apparaît qu'au Cameroun, les cybercriminels se comptent en majorité parmis les étudiants globalement établis près des zones universitaires dans les régions administratives du centre, Sud-ouest et Nord-ouest.

Les motivations fondamentales des cybercriminels sont relatives à des composantes d'ordre social, technique, politique, financière ou étatique [32, 33].

La motivation sociale trouve ses racines dans le besoin de reconnaissance de l'indi-

vidu par ses pairs, lié généralement à une structure de bande. Il veut prouver sa valeur au groupe en se référant aux critères culturels internes. Il s'agit, d'un phénomène analogue à celui des "taggers", et qui est relié à une vision très primaire des rapports sociaux. On le retrouve fréquemment chez les immatures pour lesquels le "hacking" apporte un sentiment de supériorité et de contrôle d'institutions qu'ils estiment subir dans leur quotidien.

La motivation technique reste rare. Elle a pour objet premier la recherche des limites de la technologie, afin d'en mettre en lumière les limites et les faiblesses et d'en mieux comprendre les atouts.

La motivation politique consiste à créer un événement propre à alerter les médias, pour les focaliser sur un problème grave en espérant provoquer une prise de conscience collective qui amènera sa résolution. Il est à noter alors que la frontière avec le terrorisme peut être ténue, au moins d'un point de vue conceptuel. On doit également souligner que bon nombre de personnes dissimulent leur motivation sociale derrière un objectif politique.

La motivation financière peut s'avérer très forte et sous-tend beaucoup d'actions illicites. L'appât du gain permet à des criminels en col blanc de s'exprimer via le réseau internet (voleurs, escrocs, concurrents déloyaux, etc.).

Enfin, on peut distinguer une motivation gouvernementale. Qu'il s'agisse de guerre de l'information ou d'espionnage, elle concerne des services administratifs agissant pour le compte de puissances étatiques.

Les délinquants ont su s'adapter aux nouvelles technologies pour faire fructifier leurs activités traditionnelles. On peut légitimement s'inquiéter en voyant à quel point ils peuvent être créatifs lorsqu'il s'agit d'inventer de nouveaux usages pour ces technologies.

1.2.2 Mode opératoire des cybercriminels

1.2.2.1 Phases d'une attaque

La figure 1.3 présente les différentes phases de déroulement d'une attaque. La première phase de collecte d'informations et de recherche de vulnérabilité d'un système cible a pour objet de récolter le maximum d'informations sur le système ciblé afin de les exploiter. Cela consiste à connaître les mécanismes et niveaux de sécurité en vigueur concernant l'identification, l'authentification, le contrôle d'accès, la cryptographie, la surveillance et à identifier les failles techniques, organisationnelles, humaines de l'environnement [32]. L'attaquant tirera partie le plus souvent de la naïveté ou de la crédulité des utilisateurs pour leur soutirer des informations facilitant la création d'une attaque (notion de social engineering).

De plus, le fraudeur s'emploiera à détecter et exploiter les failles de sécurité connues mais non encore réparées (non patchées) et à utiliser les outils disponibles (notions de bibliothèques d'attaques ou de boites à outils d'attaques) pour s'introduire dans les sys-

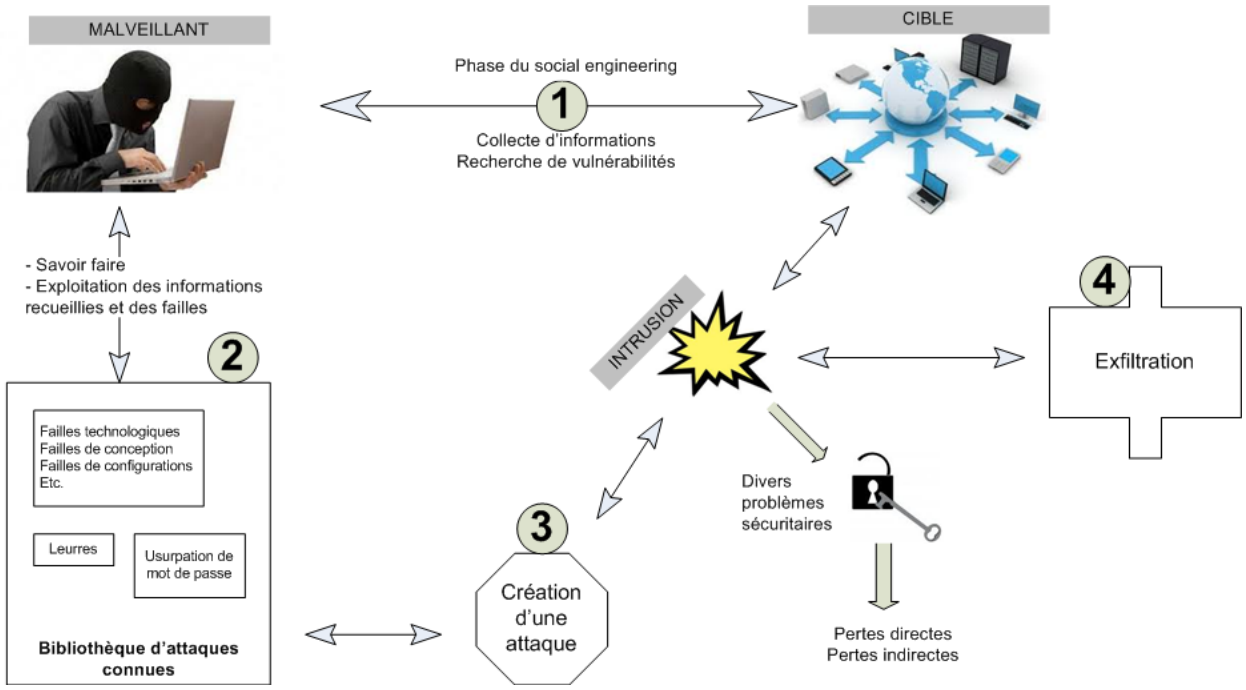


Figure 1.3 – Phases caractéristiques du déroulement d'une attaque

tèmes. La phase d'exfiltration a pour objectifs principaux de faire en sorte que l'attaque ne soit pas détectée et que l'attaquant ne laisse pas de trace pouvant servir à son identification. Pour contribuer à cela, il s'emploiera à rester anonyme, à utiliser des alias (pseudonymes), à usurper l'identité numérique d'utilisateurs, ou encore à brouiller les pistes en passant par plusieurs systèmes intermédiaires ou relais.

1.2.2.2 Attaques sur les réseaux

La mise en réseau des ordinateurs les expose à des menaces qui peuvent tout simplement alarmer un système d'information ou le dévaster. Ceci est essentiellement dû au fait que le réseau véhicule des informations qui échappent au contrôle physique de l'utilisateur. Les attaques les plus récurentes sont :

Injection de commandes SQL : Les attaques par injection de commandes SQL sont des attaques visant les sites web s'appuyant sur des bases de données relationnelles. Dans ce type de sites, des paramètres sont passés à la base de données sous forme d'une requête SQL. Ainsi, si le concepteur n'effectue aucun contrôle sur les paramètres passés dans la requête SQL, il est possible à un pirate de modifier la requête afin d'accéder à l'ensemble de la base de données, voire à en modifier le contenu.

Déni de service (Dos) : Une attaque par déni de service (denial of service attack) est une attaque informatique ayant pour but de rendre indisponible un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. Il peut s'agir de l'inondation d'un réseau afin d'empêcher son fonctionnement ; la perturbation des connexions entre deux machines, empêchant l'accès à un service particulier ; l'obstruction d'accès à

un service à une personne en particulier. L'attaque par déni de service peut ainsi bloquer un serveur de fichiers, rendre impossible l'accès à un serveur web ou empêcher la distribution de courriel dans une entreprise.

Débordement de tampon : Le fonctionnement général d'une attaque par débordement de tampon consiste à faire crasher un programme en écrivant dans un tampon mémoire plus de données qu'il ne peut en contenir, dans le but d'écraser des parties du code de l'application et d'injecter des données utiles pour exploiter le crash de l'application. Cela permet donc en résumé d'exécuter du code arbitraire sur la machine où tourne l'application vulnérable.

Elévation de privilèges : L'élévation de privilège résulte de l'octroi à un intrus (ou une application malveillante) d'autorisations supérieures à celles initialement accordées généralement en exploitant des bugs du système d'exploitation.

Exécution de code arbitraire : Il s'agit d'une vulnérabilité dans un système qui peut être utilisé par un pirate pour exécuter un script ou un programme quelconque sur un ordinateur ciblé.

1.2.3 Problématique

La réalité de l'insécurité des technologies de traitement de l'information trouve ses origines dans les caractéristiques des technologies du monde virtuel. La dématérialisation des acteurs, les accès à distance, un relatif anonymat, les problèmes de conception, de mise en oeuvre, de gestion, de contrôle des systèmes d'information, associée aux pannes, dysfonctionnements, erreurs, incompétences, incohérences ou encore aux catastrophes naturelles, confèrent de facto un certain niveau d'insécurité aux infrastructures informatiques [14, 31, 32, 33]. Par conséquent, les cybercriminels n'hésitent pas à tirer profit de nombreuses possibilités qu'offre internet pour réaliser des actes malveillants (figure 1.4).

1.2.3.1 Monde virtuel et dématérialisation

La dématérialisation des transactions, les facilités de communication associées aux solutions de chiffrement, de stéganographie et d'anonymat, autorisent des liaisons entre criminels de différents pays sans contact physique, de manière flexible et sécurisée en toute impunité. Ainsi, ils peuvent s'organiser en équipes, planifier des actions illicites et les réaliser par le biais des nouvelles technologies. La couverture internationale du réseau internet permet aux criminels d'agir au niveau mondial, à grande échelle et très rapidement.

1.2.3.2 Mise en réseau des ressources

La généralisation de la mise en réseau des ressources informatiques et informationnelles, font qu'elles deviennent des cibles attrayantes pour la réalisation de crimes

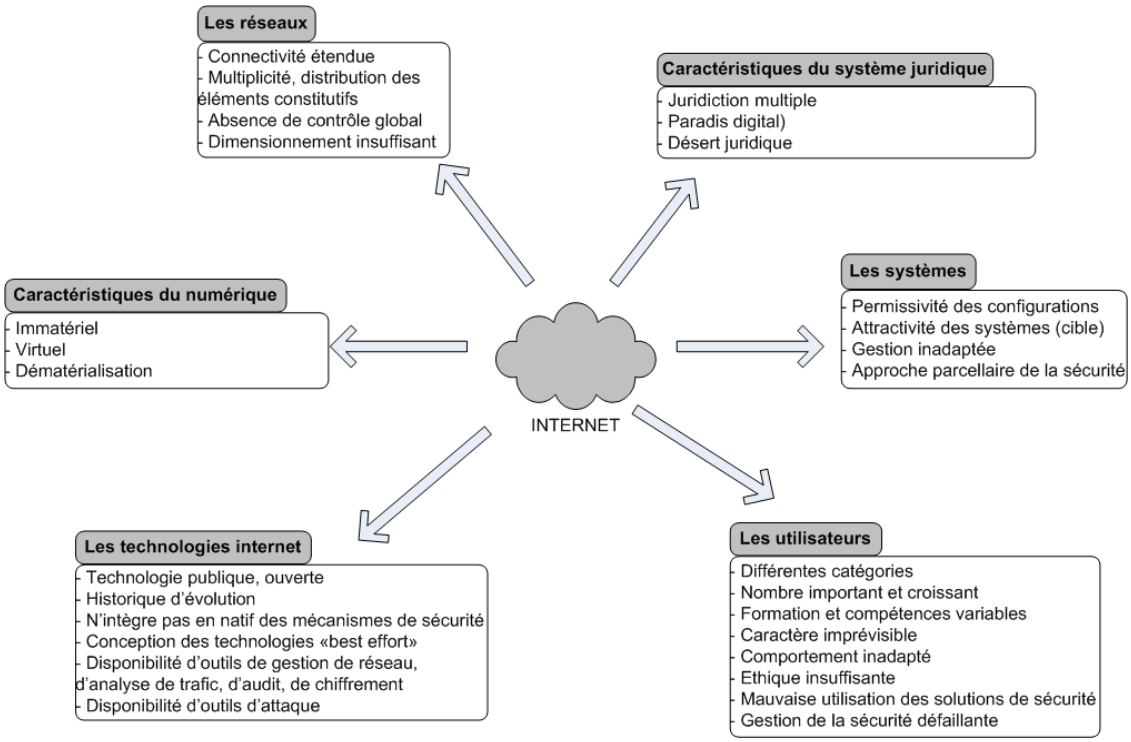


Figure 1.4 – Principales caractéristiques du réseau internet exploitées à des fins criminelles

économiques via les nouvelles technologies. Les différentes formes d'attaques informatiques existantes ont pour dénominateur commun qu'elles font courir relativement peu de risques à leur auteur et, possèdent des conséquences négatives et dommages potentiels bien supérieurs aux ressources nécessaires pour les réaliser. L'usurpation d'identité électronique, les possibilités d'anonymat ou la prise de contrôle d'ordinateurs par exemple, facilitent la réalisation d'actions illégales sans prise de risque excessive. C'est le cas particulier des attaques par déni de service distribué (Ddos).

1.2.3.3 Disponibilité d'outils et existence de failles

La disponibilité d'outils d'exploitation des failles et vulnérabilité des systèmes, de bibliothèques d'attaques et de logiciels qui capitalisent le savoir-faire criminel dans un programme, facilite la réalisation des attaques informatiques. Cette disponibilité associée à la dématérialisation des actions, favorise le comportement malveillant des informaticiens qui possèdent la fibre criminelle et des criminels qui possèdent des aptitudes en informatique. Le cyberespace facilite pour certains le passage à l'illégalité sans parfois de prise de conscience réelle de la dimension criminelle des actes perpétrés.

1.2.3.4 Vulnérabilité et défaillance

La criminalité tire parti des vulnérabilités et défaillances organisationnelles et techniques de l'internet, de l'absence d'un cadre juridique harmonisé entre les Etats et d'un manque de coordination efficace des polices. Il peut s'agir de criminalité classique (commission de délits classiques avec de nouvelles technologies : blanchiment d'argent, chantage, extorsion, etc.) ou générer de nouveaux types de délits à partir des technologies du numérique : intrusion dans des systèmes, vol de temps processeur, vol de code source, de bases de données, etc. Dans tous les cas, ils s'effectuent dans des conditions exceptionnelles d'optimalité (risques minimaux, couverture importante, profitabilité maximale).

1.2.3.5 Difficulté à identifier l'auteur d'un délit

La figure 1.5 identifie différents types de problèmes induits par la malveillance tels que la destruction physique ou le vol de matériel qui empêche l'accès aux systèmes et données, puis l'infection des ressources et la compromission des processus de décision ou de communication par des attaques de déni de service (ou suite à de l'espionnage ou à l'intrusion dans des systèmes), l'appropriation illégale ou la manipulation d'informations (endoctrination, guerre de l'information). Elle met également en évidence les principales caractéristiques du cybercrime qui rendent difficiles l'identification des malveillants.

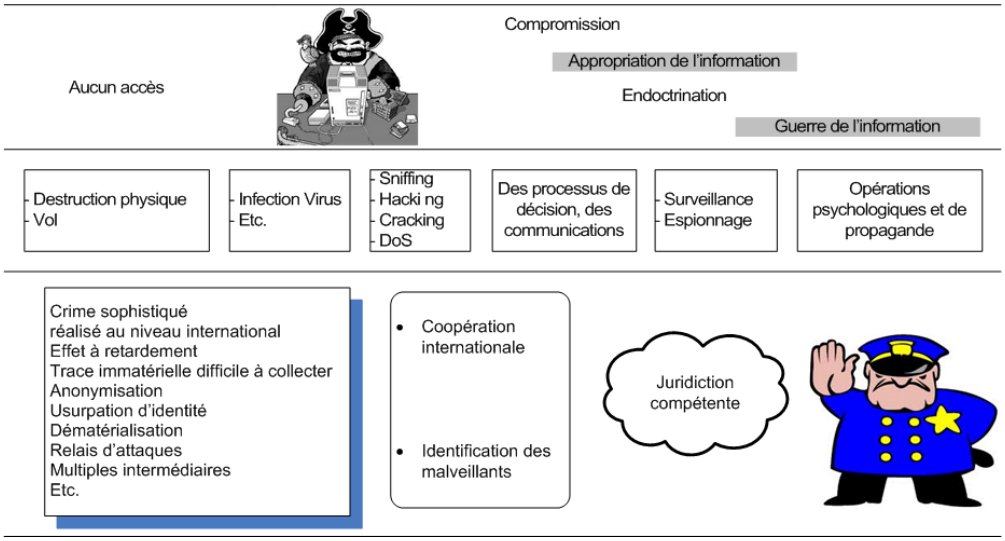


Figure 1.5 – Difficulté à identifier un malveillant

Par ailleurs, dans la plupart des Etats, il existe un décalage significatif entre les aptitudes des criminels à effectuer des crimes de haute technologie et les moyens mis à disposition des forces de justice et police pour les poursuivre. Le niveau d'adoption des nouvelles technologies par les instances de justice et de police aux niveaux national et international reste faible et très disparate d'un pays à l'autre [34]. Ce sont généralement les moyens courants d'investigation des crimes traditionnels auxquels ont recours les forces

de justice et police pour poursuivre les cybercriminels qui permettent de les identifier et de les arrêter.

1.2.3.6 Aterritorialité et paradis numérique

Le monde criminel tire parti de l'aterritorialité de l'internet, de l'inexistence dans certains Etats de lois réprimant le crime informatique et des juridictions multiples dont relève l'internet. De manière analogue aux paradis fiscaux, les paradis numériques permettent aux criminels d'héberger des serveurs, diffuser des contenus illicites ou réaliser des actions illicites en toute impunité. Le fait de pouvoir localiser des serveurs dans des Etats faibles constituent des refuges à des opérations transnationales.

Le manque de régulation internationale et de contrôle, l'inefficacité de la coopération internationale en matière d'investigation et de poursuites judiciaires font qu'internet offre une couche d'isolation protectrice aux criminels. A l'heure actuelle, aucune réponse correcte tant sur le plan juridique que technique est apportée pour maîtriser les différents délits favorisés par l'internet tels que [35] :

— l'industrie parallèle et très organisée de la copie à la chaîne de logiciels, de films, de musique, etc., qui a pris dans le cyberespace une dimension sans précédent ;
— les atteintes au copyright, droits d'auteur, la violation du secret professionnel, de l'intimité numérique ou de la propriété intellectuelle ;
— les atteintes à la propriété, l'appropriation illégale de la propriété d'autrui, l'endommagement, la destruction de la propriété d'autrui ou l'immixtion dans la propriété d'autrui (notion de violation de domicile virtuel) ;
— la dissémination de contenus illégaux ;
— les attaques concurrentielles, l'espionnage industriel, l'atteinte aux droits des marques, la diffusion de fausses informations, le déni de service commandités par des concurrents.

1.2.3.7 Réglementation

1.2.3.7.1 Réglémentation nationale : Les Etats possèdent des responsabilités importantes pour la réalisation d'une sécurité numérique. Ceci ne peut s'avérer que par la définition d'un cadre légal approprié, c'est-à-dire unifié et applicable, pour la promotion d'une culture de la sécurité qui respecte l'intimité numérique des individus, tout en renforçant la lutte contre la cybercriminalité.

Au Cameroun, la lutte contre la cybercriminalité a conduit à création d'un texte de loi relatif à la cybersécurité et la cybercriminalité. Il s'agit de la loi N°2010/012 du 21 décembre 2010. D'autres lois toutes aussi importantes dans ce domaine ont vu le jour notamment celle N°2010/013 du 21 décembre 2010 régissant les communications électroniques. Toutes ces lois mettent en évidence les mesures répressives à travers les sanctions pénales, des peines privatives de liberté et des amendes importantes pour les infractions relatives à la cybercriminalité.

Cependant, cette lutte est butée à plusieurs obstacles : malgré l'urgence de la réaction contre la cybercriminalité, la loi n'est pas encore effective étant donné que son décret d'application est encore attendu. Plus encore, les officiers de police judiciaire n'ont en général pas le profil, la technique et la technologie, et ne recoivent pas la formation adéquate leur permettant d'investiguer et d'appliquer la loi. Jusqu'à présent, les efforts sont essentiellements orientés vers la sensibilisation.

1.2.3.7.2 Réglémentation internationale : Les moyens de lutte contre le fléau grandissant et transfrontalier qu'est la cybercriminalité passe par la mise à disposition d'un cadre légal harmonisé et applicable au niveau international, et des moyens d'une véritable coopération internationale des instances de justice et police.

Beaucoup de pays africains à l'heure actuelle ont en cours des projets de loi sur la cybercriminalité. Il en existe dans des regroupements sous régionaux tel que la Communauté Economique des Etats de l'Afrique de l'Ouest (CEDEAO) [36]. Cependant, l'Union africaine à travers les chefs d'Etat et de gouvernement des pays membres ont adopté en 2013 un projet de convention sur la cybersécurité, visant à protéger les pays africains de cyberattaques sur les institutions et la population contre la cybercriminalité.

La première réglementation internationale, contribuant à appréhender la dimension internationale de la cybercriminalité est la Convention sur la cybercriminalité, adoptée sous l'égide du Conseil de l'Europe et entrée en vigueur en juillet 2004 [37]. Cette convention a été récement ratifiée par les américains. En effet, le Conseil de l'Europe aide à protéger les sociétés contre les menaces de la cybercriminalité par le biais de cette Convention et son Protocole additionnel relatif à l'incrimination d'actes de nature raciste et xénophobe commis par le biais de systèmes informatiques.

1.3 Investigation numérique

Le developpement de l'investigation numérique comme une profession et une discipline scientifique nait de la volonté des administrations de barrer la route aux crimes facilités par les nouvelles technologies. Entre les années 1980 et 1990, les agences américaines chargées de l'application de la loi ont conjugué leurs efforts pour developper cette nouvelle approche d'investigation. Plusieurs expressions sont utilisées pour désigner l'investigation numérique. Il s'agit entre autres de : computer forensics, digital forensics, inforensique, legal information and technology, informatique juridique.

On désigne par investigation numérique l'application des techniques et des protocoles d'investigation numérique respectant les procédures forensics et destinée à apporter des preuves numériques à la demande d'une institution de type judiciaire par réquisition, ordonnance ou jugement. Ces techniques et protocoles sont destinés à la collecte, l'identification, la description, la préservation, l'extraction, l'authentification, l'analyse, l'interprétation et l'explication de l'information numérique en vue de les produire dans le cadre d'une action en justice. Ces techniques sont mises en œuvre quand une affaire comporte

des questions relatives à l'usage d'un ordinateur, du cyberespace et de tout autre support d'information, ainsi qu'à l'examen et à l'authentification des données en faisant appel aux techniques d'analyse du fonctionnement des ordinateurs ou à la connaissance des structures de données. L'investigation en milieu numérique prend en compte non seulement les techniques et méthodes à utiliser mais également la coordination des activités que les investigateurs doivent mener. La stratégie de sécurité adoptée tant pour la gestion des indices et preuves numériques que pour la collaboration entre les enquêteurs est un préalable pour la poursuite judiciaire. L'investigation ou enquête numérique est une activité particulièrement rescente. Elle requiert une très bonne maîtrise de l'univers numérique en général (matériels, systèmes, logiciels, bases de données, réseaux, sécurité, nouvelles technologies, etc.) et des contraintes juridiques particulières qui le régisse.

La figure 1.6 et le tableau 1.1 resument l'existant en matière d'investigation jusqu'en 2009. L'on peut noter que les modèles jusqu'à lors proposés étaient essentiellement réactifs.

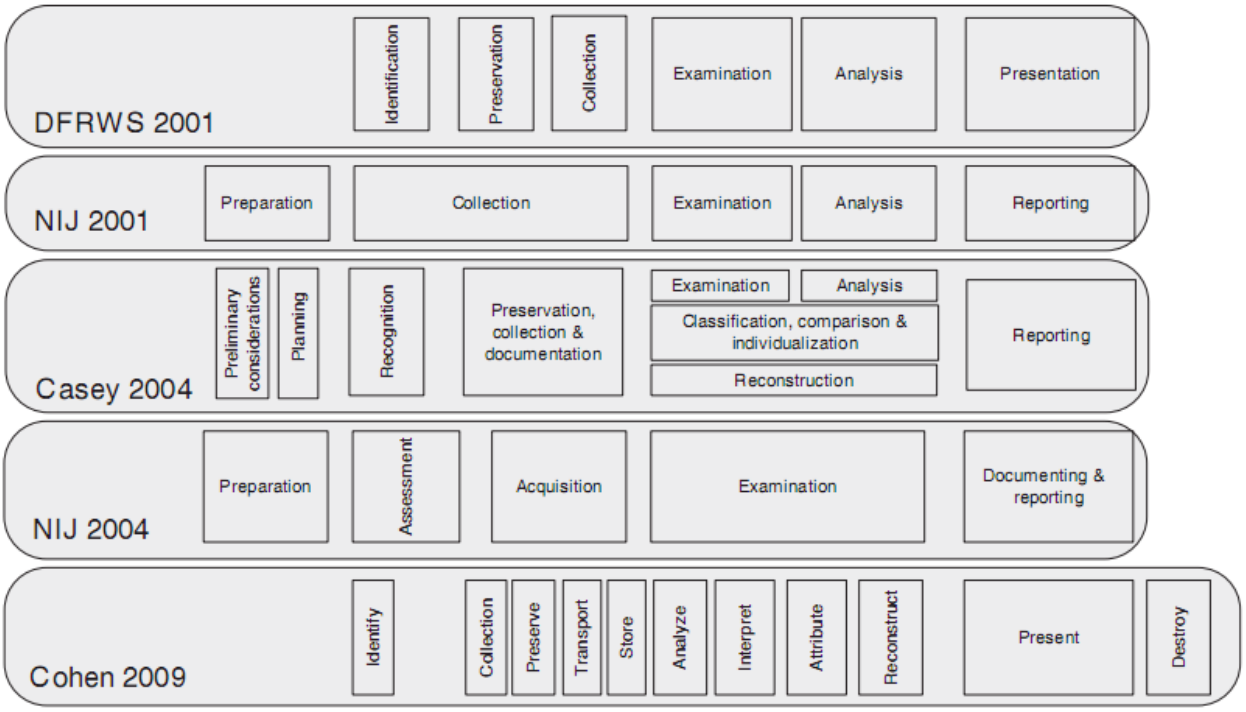


Figure 1.6 – Comparaison sur la terminologie employée dans les modèles d'investigation numérique

Dans le domaine de l'investigation numérique, quelques auteurs seulement ont proposés une investigation proactive. Alors que certains l'ont explicitement évoqué, ce processus se sous entend chez les autres. Cependant, les idées convergent régulièrement vers la nécessité d'élaborer un tel processus.

Dans [55], Rowlingson déclare qu'au sein de plusieurs organisations, l'équipe de gestion des incidents réalise déjà les activités de collecte de preuves. Mais, il ajoute que le besoin de collecter ces preuves et de les préserver dans une approche systématiquement

N°	Auteurs ; Références ; Genre	Identification	Nombre de phases (Ph)
1	G. Palmer ; [38] ; Conférence	A road map for digital forensics research-report from the first Digital Forensics Research Workshop (DFRWS)	6 Ph
2	R. Mark, C. Clint, G. Gregg ; [39] ; Journal	An Examination of Digital Forensic Models	9 Ph
3	Carrier, Spafford ; [40] ; Journal	Getting physical with the digital investigation process	17 Ph organisées en 5 Ph principales
4	Stephenson ; [41] ; Journal	A comprehensive approach to digital incident investigation	9 Ph
5	Baryamureeba, Tushabe ; [42] ; Journal	The Enhanced Digital Investigation process	5 Ph avec des sous Ph
6	Ciardhuain ; [43] ; Journal	An Extended model of Cybercrime investigations	13 Ph
7	Carrier, Spafford ; [44] ; Conférence	An Event-Based Digital Forensic investigation Framework	5 Ph avec des sous Ph
8	Harrison ; [45] ; Journal	The digital detective : An introduction to digital forensics	7 Ph
9	Beebe, Clark ; [46] ; Journal	A hierarchical, objectives-based framework for the digital investigations process	6 Ph
10	Kohn, Eloff, Olivier ; [47] ; Conférence	Framework for a digital forensic investigation	3 Ph
11	Kent, Chevalier, Grance, Dang ; [48] ; Journal	Guide to Integrating Forensic Techniques into Incident Response	4 Ph
12	Rogers, Goldman, Mislan, Wedge, Debrota ; [49] ; Conférence	The Computer Forensic Field Triage Process Model	6 Ph avec des sous Ph
13	Ieong ; [50] ; Journal	FORZA - Digital Forensics Investigation Framework Incorporation Legal Issues	8 Ph
14	Freiling, Schwittay ; [51] ; Conférence	A Common Process Model for Incident Response and Computer forensics	3 Ph avec des sous Ph
15	Khatir, Hejazi, Sneiders ; [52] ; Séminaire	Two-Dimensional Evidence Reliability Amplification Process Model	5 Ph avec des sous Ph
16	S. Yong-Dal ; [53] ; Conférence	New Digital Forensics Investigation Procedure Model	10 Ph avec des sous Ph
17	Billard ; [54] ; Livre	An Extended Model for E-Discovery Operations	10 Ph
18	grobler, Louwrens, Solms ; [6] ; Conférence	A multi-component View of Digital Forensics	3 Ph avec des sous Ph
19	Soltan Alharbi, Jens Weber-Jahnke, Issa Traore ; [4] ; Conférence	The Proactive and Reactive Digital Forensics Investigation Process : A Systematic Literature Review	2 Ph avec des sous Ph

Tableau 1.1 – Différentes approches d'investigations numériques

proactive demeure un challenge.

Dans [**?**], Garfinkel propose que pour élucider les attaques antiforensics, les organisation doivent auparavent déterminer les informations à collecter et à préserver en respectant la chaine de traçabilité.

Dans [6], Grobler et al. notent que l'investigation proactive est bloquée par l'absence des définitions de mécanismes et de procédures standardisés. Ils pensent qu'il faut automatiser et activer des outils de collecte de preuves de manière à limiter l'intervention humaine pour assurer l'intégrité des preuves receuillies. Ces auteurs ont ainsi proposé un modèle d'investigation ayant trois composants : Proactive, active et réactive. Cependant, la composante proactive dont ils font allussion renvoit à la prédisposition d'une organisation à mener efficacement une investigation. Malheureusement, non seulement la composante proactive n'est pas définie, mais encore, le niveau d'abstraction appliqué pour la réalisation de ce modèle ne permet pas de le rendre opérationnel de manière à en créer des outils automatisés. Par ailleurs, dans ce modèle, il est impossible de dissocier les tâches spécifiques des enquêteurs de ceux des responsables de la sécurité au sein d'une organisation.

Dans [26], Garfinkel fait l'état de l'art des modèles d'investigation existants. Dans son résumé, il déclare qu'il serait inopportun de se limiter aux pistes d'audits et aux fichiers logs internes dans le cadre d'une investigation numérique. Il ajoute que les modèles d'investigation numérique efficaces seront ceux qui pourront donner naissance aux outils et aux techniques capables de collecter et de présever les preuves numériques de manière proactive.

En somme, bien que la majorité des modèles existants soient réactifs, les auteurs ont toujours mentionné le besoin d'une méthodologie capable de collecter et de préserver les preuves numériques de manière proactive.

1.3.1 Notion de preuve numérique

Terme adapté de l'anglais " digital evidence ", l'expression " preuve numérique " représente toute information numérique pouvant être utilisée comme preuve dans une affaire de type judiciaire [32]. La collecte de l'information numérique peut provenir de l'exploitation de supports d'information, de l'enregistrement et de l'analyse de trafic de réseaux (informatiques, téléphoniques, etc.) ou de l'examen de copies numériques (copies-images, copies de fichiers, etc).

Les indices et traces recherchées peuvent être : des images ; des documents bureautiques (lettres, documents, feuilles de calcul, etc.) ; les adresses électroniques ; les courriers électroniques ou SMS envoyés et reçus (si cela a été explicitement autorisé par le Magistrat) ; les sites Web visités ; des mots de passe mémorisés ; les cookies (informations personnelles d'accès à un site internet donné) ; les logiciels installés ; les dates d'activité du PC (dates de création et de dernière modification d'un fichier, etc.) ; les numéros appelés ou reçus. Les informations présentes sur un support numérique, peuvent être visibles,

mais aussi être délibérément cachées ou détruites (certaines pouvant être restaurées).

1.3.1.1 Exigence de la preuve numérique en justice

La preuve n'existe pas en elle-même de manière absolue. Il s'agit de tout ce qui est utilisé pour établir la vérité sur un fait ou un état de fait particulier. Généralement, les preuves doivent satisfaire les exigences de forme et de fond. En ce qui concerne la forme, la preuve doit être conforme à certaines règles légales qui sont appliquées par un juge [36, 58] suivant la juridiction considérée. Quant au fond, la preuve doit être comprise et suffisamment convaincante pour la cour, lorsqu'il y a un juge ou un jury pour juger les faits.

Devant une cour de justice, une preuve peut être une entité matérielle, un témoignage, un rapport d'expertise ou tout ce qui en découle [59, 60]. Par conséquent, pour être acceptée par un tribunal, il doit y avoir une chaîne de tracabilité ou de continuité de la preuve et la méthode d'investigation utilisée doit être transparente, c'est-à-dire susceptible d'être librement évaluée par un tiers. Quoiqu'il en soit, devant une cour de justice, les plaignants doivent démontrer qu'un système d'information a été attaqué, qu'on y a accédé, que cet accès était non autorisé au moment où il a eu lieu et que le prévenu le savait. D'autres sources peuvent être utilisées pour soutenir certaines preuves. Elles peuvent être des extraits du Journal du pare-feu, du Journal d'Accès au Serveur Web, SMTP/IMAP (emails), des Serveurs FTP, du journal du Serveur de Proxy, des logs de connexions SSH (accès à distance), des Routeurs et des Switch, des Serveurs de Chat (DNS), des systèmes informatiques de la victime et de l'attaquant.

La preuve numérique est une modalité particulière d'établissement de la vérité qui consiste à avoir recours à des moyens numériques variés qui vont de l'étude des contenus dans la mémoire d'un disque dur aux messages électroniques en passant par l'enregistrement numérique. Cependant, le caractère immatériel, volatile et évolutif de l'information numérique ont longtemps été des obstacles à son élection de preuve. Dans la procédure pénale française, la preuve numérique est soumise à deux series d'exigences juridiques :

— sa collecte doit être loyale et proportionnée. La collecte de la preuve numérique necessite une certaine publicité des moyens mis en œuvre pour éviter de porter atteinte aux libertés individuelles ou collectives des personnes ;
— lorsqu'elle a pour objet un acte, sa recevabiité est conditionnée à des exigences d'intégrité, d'imputabilité et de fiabilité.

Pour donc être recevable, la preuve numérique doit avoir passé avec succès une série de test relatif à l'authenticité, l'intégrité, la traçabilité, l'exploitabilité, la pérénité et l'interpretation.

Selon la législation américaine, pour être recevable, la preuve numérique doit être :

— authentique : elle doit être originale et se raporter au crime pour lequel l'investigation est en cours ;
— fiable : elle doit avoir été collectée en utilisant des procédures fiables qui en cas de besoin, peuvent être exécutées de nouveau par un tiers et produire le même résultat ;
— complète : elle peut être utilisée à charge ou à décharge ;

— crédible : elle doit être convaincante et présentée de façon à ce qu'elle ait un sens ;
— admissible : elle doit avoir été collectée en utilisant des procedures prévues par la réglémentation en vigueur en matière d'investigation.

1.3.1.2 Chaine de traçabilité et flux de preuve

La chaine de traçabilité est la ligne de conduite que l'enquêteur a suivi pour l'obtention de la preuve numérique [37, 61]. En effet, l'enquêteur peut être requis devant une juridiction pour témoigner que la preuve présentée devant la cour est la même que celle qui a été receuilie au cour de l'enquête. Il lui reviendra dans ce cas, la charge de démontrer que la preuve présentée n'a pas été altérée depuis sa collecte sur la scène de crime jusqu'à sa présentation devant la cour. Pour le faire, l'enquêteur doit consigner dans un document pré-établi par une institution judiciaire, manuscrit ou numérique, tous les actes qu'il a posé pour la collecte des indices et preuves en vue de la manifestation de la vérité. L'illustration d'un tel document est faite à la figure 1.7 pour la réalisation d'une copie image d'un disqur dur suspect.

cmdLabs
Continuity of Possession Form

Case Number:	2010-05-27-00X		Client/Case Name:	Digifinger Intrusion
Evidence Type:	hard drive		Evidence Number:	0023
Details:	Mac storage <network share>			
Date of Transfer	Transferred From	Transferred To	Location of Transfer	Action Taken by Recipient
5/27/10	Sam Spade	Philip Marlowe	Digifinger HQ Linthicum MD	Collected evidence for examination

Figure 1.7 – Exemple de chaine de traçabilité

Le but principal de toute investigation est de suivre les traces laissées par un criminel pendant la commission d'une infraction et d'établir une corrélation (figure 1.8) entre l'auteur, la victime et la scène de crime. Le transfert de preuve dans les dimensions numériques et physiques aide les enquêteurs à établir des connexions entre la victime, l'attaquant et la scène de crime.

1.3.1.3 Nature de la preuve numérique

La preuve apportée est à sens unique : "l'absence de preuve n'est pas la preuve de l'absence". L'investigation numérique peut en effet déboucher sur le constat de l'absence des informations recherchées. Compte tenu des techniques d'enregistrement des données numériques, ces informations peuvent pourtant avoir été présentes sur le support analysé et avoir été recouvertes ensuite par d'autres [62].

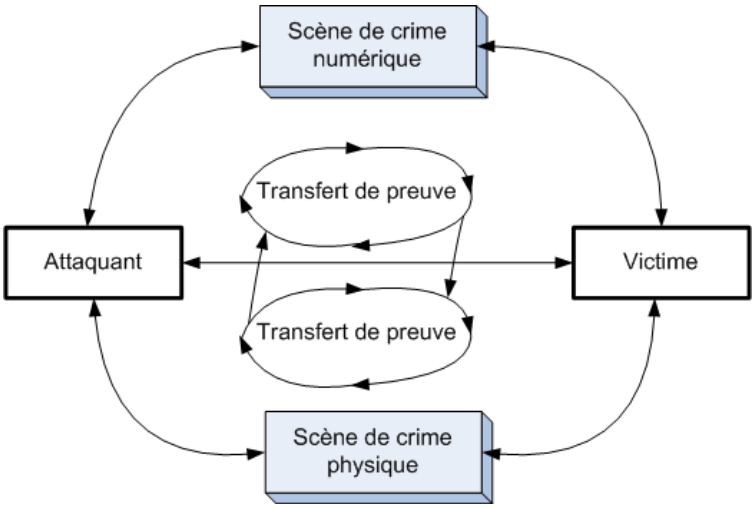


Figure 1.8 – Correlation auteur, victime et scène de crime

Autrement dit, si l'investigation numérique ne trouve pas l'information recherchée, cette preuve ne vaut qu'en l'état du support et au moment de son analyse. Il ne peut en aucun cas en être déduit que ce support n'a jamais comporté l'information en cause. Les services compétents de la police judiciaire produisent généralement :

une preuve disculpante qui élimine la possibilité qu'un suspect ait participé à une affaire ;

une preuve indicative qui indique qu'il y a eu crime mais, qui ne donne pas necessairement les informations sur l'auteur de l'infraction ;

une preuve corroborative qui consiste en la réunion de faisceaux d'indices qui corroborent les témoignages afin de convaincre le juge de l'identité de l'auteur du crime et de la manière dont celui-ci a été commis.

Les preuves et traces laissées sur les réseaux numériques n'obéissent à aucune hiérarchie et sont même parfois générées à l'insu de l'internaute (1), même si l'utilisateur dépose lui aussi des preuves lors de ses connexions à un réseau ouvert (2) en se servant d'une adresse IP (Internet Protocol)(3).

1.3.1.4 Preuves générées à notre insu

— les Cookies : Ce sont des éléments indispensables à la navigation. Il est parfois impossible d'accéder à un site internet si l'option " accepter les cookies " n'est pas activée. Ils sont si inévitables qu'ils ont fini par apparaître comme des utilitaires innocents. Très récemment, il a été mis à jour que les sites gouvernementaux américains, tels que la CIA, la NSA, avaient pisté les internautes connectés à leurs sites internet, en utilisant des cookies permanents [63, 64]. Les cookies sont des fichiers-texte envoyés sur le disque dur de l'internaute par le serveur auquel il se connecte et permettent, par exemple, de l'identifier lors d'une nouvelle visite. Ces cookies sont considérés comme des fichiers espions, leur objet est bel et bien de collecter des informations sur le comportement en ligne. Les cookies peuvent enregistrer l'adresse

IP de l'ordinateur qui, si elle est fixe, donne l'origine géographique de la personne, le système d'exploitation, le nom donnée à la machine à voir, l'heure de la connexion et sa durée, les pages visitées, les mots de passe et les login utilisés sur le site, etc. et ce à chaque nouvelle visite. En effet, les cookies collectent des informations nominatives dès lors que l'on est enregistré comme utilisateur habituel ou qu'on a donné un numéro de carte bancaire à la fin d'une transaction.

— Les traces laissées par le serveur : Tous les serveurs d'une entreprise connectés à l'internet conservent des traces des connexions des employés à travers les différents fichiers de journalisation. On peut conserver un certain nombre d'informations telles que le nom de l'hôte de destination, les octets envoyés, le journal de connexion, les ports de destination et bien entendu, les adresses IP des sites consultés.

1.3.1.5 Traces envoyées de notre plein gré

C'est dans le cadre du commerce en ligne que ce type de preuve est le plus important.

En effet, le consommateur est en situation de faiblesse par rapport à la conservation de la preuve. Par exemple, le cyber-commerçant en ligne doit indiquer le jour de conclusion du contrat électronique et laisser ce document à la disposition du consommateur [64]. Le moment de conclusion du contrat constitue le point de départ du délai de livraison. Le consommateur a deux, voire trois, possibilités de conservation de la preuve. Il ne peut qu'imprimer le document contractuel en ligne, l'accusé de réception envoyé par courriel, ou faire des captures d'écran, c'est-à-dire sauvegarder des images numériques. Néanmoins, le commerçant est tenu d'envoyer un accusé de réception sous la forme d'un courrier électronique. Cependant, face aux difficultés rencontrées par le consommateur avec les commerçants en ligne, il serait nécessaire de constituer une véritable preuve numérique. En réalité, le consommateur doit se reposer sur des preuves plus traditionnelles : impression d'écran, relevé bancaire, à défaut de pouvoir utiliser une véritable signature électronique. Toutefois, une architecture basée sur les signatures de groupe [65] permet d'asseoir un climat de confiance mais aussi de garantir la traçabilité et par conséquent la collection rapide des preuves numériques.

1.3.1.6 Adresse IP

L'adresse IP est le point de départ névralgique pour la recherche de l'auteur d'une infraction car c'est une preuve qui authentifie fortement l'identité d'un intrus ou d'une machine qui tente de s'introduire. En effet, l'adresse Internet Protocol est un numéro qui identifie de manière certaine un ordinateur. C'est une ressource rare de l'Internet standardisée et contrôlée par l'Internet Corporation for Assigned Names and Numbers (ICANN - en réalité plutôt l'IANA au sein de l'ICANN) [14]. Elle est donc une ressource essentielle en ce sens, qu'étant centralisée, il est aisé de savoir à qui a été attribuée telle adresse IP. Or, le statut de l'adresse IP est le même que pour toute preuve électronique : c'est bien a priori une preuve admissible devant un Tribunal. Pour autant, rapporter l'existence

d'une adresse IP laissée comme trace sur un serveur attaqué, nécessitera une sauvegarde du journal des connexions (logs) qui généralement enregistre cette information, l'établissement d'un rapport sous une forme familière à destination d'un juge, une explication complémentaire donnée par une tierce partie ou mieux un expert. Cette procédure vise à convaincre le juge de ce que l'exploitation que le plaideur fait des information devant le Tribunal est juste.

1.3.1.7 Exploitation des traces

L'identité de l'auteur est parfois difficile à établir. Il est nécessaire d'effectuer des recoupements, si possible avec des indices externes au média analysé. En effet, les indices découverts sur un PC peuvent avoir été produits par un tiers qui aurait alors pris son contrôle à l'insu de son propriétaire (Par exemple par un accès à distance via Internet). On dit alors que le PC a été compromis. Il est assez difficile de prouver qu'un ordinateur n'a pas été compromis, car un intrus avisé peut avoir effacé les traces de son intrusion. La nécessité du recoupement des indices est un nouveau défi technologique car il faut établir les liens reliant l'indice aux différents médias utilisés.

Les informations mémorisées peuvent être incompréhensibles car chiffrées (disque dur des ordinateurs portables, fichiers encryptés proposés par les dernières versions de Windows (XP, Vista, 7) ou des systèmes Linux [66, 67]. La cryptanalyse est une science et une pratique plus ou moins difficile selon le système de chiffrement utilisé. Parfois banal (mais désormais dans de rares cas), le décryptage s'avère généralement impossible, ou au moins très difficile, les algorithmes de génération de clé et de chiffrement actuels étant mieux élaborés.

Plusieurs logiciels, dont EnCase, WinHex, Forensic Tool Kit (FTK), Digital Forensics Framework (DFF), SMART, The Coroner's Tool Kit (CTK), The Sleuth Kit (TSK), Safeback, Snapback et matériels dédiés, dont Forensic Recovery of Evidence Device (FRED), gamme Logicube, ainsi que X-Ways, CelleBrite (pour les téléphones portables) ont été développés pour répondre à la forte croissance des besoins (justice, police et gendarmerie, services secrets, armée, laboratoires privés, etc.) [67, 68].

Ces outils sont plus ou moins performants selon le cas d'utilisation mais, malheureusement, ils ne sont utilisés que lors d'une investigation numérique en mode réactif. La mise sur pied des outils d'investigation proactif reste un challenge.

1.3.2 Problématique

A l'heure actuelle, les chiffres des sondages annuels du CSI ou les statistiques du CERT [33] prouvent que la cybercriminalité est mal maîtrisée. Ainsi, on constate que les mesures de sécurité mises en place par les institutions tendent à protéger un environnement donné dans un contexte particulier, mais ne peuvent aucunement empêcher la conduite d'activités criminelles via l'internet. Les raisons de cette situation sont notam-

ment liées :

- aux caractéristiques du cybercrime (capacité à être automatisé, savoir-faire embarqué dans le logiciel, réalisation à distance) ;
- à la possibilité offerte au cybercriminel d'usurper facilement et sans risque excessif, l'identité d'utilisateurs légitimes, ruinant par là même la capacité de la justice à identifier les auteurs réels d'une infraction ;
- à la détermination des compétences pour réaliser une enquête ;
- à la pénurie de ressources humaines et matérielles au sein des services chargés de la répression des crimes et délits informatiques ;
- au caractère transnational de la cybercriminalité qui nécessite des recours fréquents à la coopération et à l'entraide judiciaire internationale. Cette dernière implique des contraintes de temps non compatibles avec la rapidité d'exécution des agressions et les besoins de reprise immédiate des systèmes informatiques concernés par les cyberattaques ;
- à la difficulté de qualifier les faits au regard de certaines législations pénales ;
- à la nature mal définie et à la volatilité de la plupart des preuves informatiques.

Pour toutes ces causes, le système judiciaire dans le contexte de l'internet, n'est pas efficace. De plus, de même qu'il existe des paradis fiscaux, il existe des paradis légaux. Ce n'est pas nécessairement par absence de lois, que le crime informatique est peu ou mal réprimé. Un certain nombre de crimes et de délits informatiques sont déjà qualifiés par le biais des législations pénales existantes dans certains pays.

1.3.2.1 Cas particulier de l'anonymisation

Il existe aujourd'hui des matériels et des logiciels qui facilitent l'anonymisation. C'est le cas par exemple d'un proxy anonymiseur qui est un service permettant de naviguer sur le web anonymement. En général, c'est un serveur proxy qui masque les données personnelles techniques (adresse IP, système d'exploitation, détails techniques concernant le navigateur, traces numériques, etc.) aux sites visités.

L'utilitaire le plus utilisé aujourd'hui est « Tor » [28]. Cet utilitaire est un logiciel libre et un réseau ouvert permettant de se défendre contre une forme de surveillance de réseau qui menace les libertés individuelles et l'intimité, les activités commerciales, les mises en relations, ainsi que la sécurité de l'État. Cette surveillance est connue sous le nom d'analyse de trafic. « Tor » protège en faisant rebondir les communications à l'intérieur d'un réseau distribué de relais, maintenus par des volontaires partout dans le monde. Il empêche qu'une tierce personne qui observe une connexion internet puisse prendre connaissance des sites visités. Il empêche également aux sites visités de connaître la position géographique du visiteur. « Tor » fonctionne avec beaucoup d'applications existantes, comme les navigateurs web, les clients de messagerie instantanée, les connexions à distance et d'autres applications basées sur le protocole TCP. Des centaines de milliers de personnes à travers le monde utilisent Tor pour une multitude de raisons : les journalistes et les blogueurs, les défenseurs des Droits de l'Homme, les agents d'application des lois, les soldats, les entreprises, les citoyens de gouvernements répressifs, ou les simples citoyens.

1.3.2.2 Techniques antiforensics

L'antiforensic est un ensemble d'outils et de techniques qui inhibe l'action des outils forensics, des modèles d'investigation et limite l'action des enquêteurs [8]. Le but de l'antiforensic est de :

- empêcher la détection des attaques ;
- pertuber ou interrompre la collecte des preuves ;
- augmenter significativement le temps mis par les investigateurs pour mener à terme leurs actions ;
- jeter le discredit sur les rapports d'investigation ou sur les témoignages y afférents ;
- contraindre un outil forensic à se magnifester ou à se revéler ;
- corrompre un outil d'investigation en l'utilisant pour attaquer une organisation.

1.3.2.3 Techniques anti-IDS

Les IDS constituent une mesure sécuritaire qui rend la vie difficile aux pirates. Ainsi ces derniers ont compris la nécessité de trouver des moyens pour outrepasser les mécanismes de sécurité assurés par les IDS afin d'attaquer sans se faire remarquer [**?**, 8]. En effet, il existe des techniques anti-IDS qui rendent possible le contournement des IDS. Ces techniques se présentent sous trois catégories :

Attaque par déni de service (saturation) :

Elle permet de rendre l'IDS inopérant en le saturant. Le but de ce type d'attaque est de saturer les ressources utilisées par l'IDS afin de le désactiver. Ainsi l'IDS ne peut plus remplir sa fonctionnalité et le pirate peut lancer ses attaques contre le réseau ciblé sans se faire remarquer.

Pour se faire les pirates utilisent des techniques d'attaque par flood qui consistent à surcharger le NIDS pour qu'il ne puisse pas fonctionner correctement et qu'il ne détecte pas l'attaque principale, ou par la noyade sous les faux-positifs dont le principe est de provoquer de nombreuses remontées d'alertes. En parallèle, une attaque réelle contre le réseau est lancée, et l'administrateur occupé à analyser les alertes, ne s'en rendra pas compte immédiatement.

Attaque par insertion :

Il s'agit d'une insertion de trafic afin de déjouer l'IDS en lui faisant croire à un trafic légitime. Le principe de ces techniques est d'injecter une attaque parmi beaucoup d'informations sans incidences. Les signes de l'attaque n'apparaissent donc pas à l'IDS mais quand les données atteignent la cible, seule l'information malintentionnée est acceptée par le système. Cela est très proche du principe des canaux cachés.

Il existe plusieurs techniques d'insertion ou d'injection qui permettent d'outrepasser les IDS. Il s'agit entre autres des attaques par "Segmentation TCP", "Fragmentation IP" et "Insertion de faux paquets".

Attaque par évasion :

Cette technique est l'inverse de l'attaque par insertion. Le principe est de faire passer des données superflues qui sont ignorées par l'IDS mais prises en compte par les

systèmes ciblés.

Cette technique exploite les faiblesses dans les mécanismes utilisés par les IDS, par exemple le mécanisme de recherche des motifs pour réussir à outrepasser l'IDS. Les techniques d'attaque par évasion les plus courantes sont : l'évasion HTTP et le Shellcode polymorphique.

Bien évidement, avant de commencer à lancer des attaques anti-IDS, il faut détecter au préalable l'existence d'un IDS sur le réseau ciblé. Pour ceci les pirates utilisent certaines techniques qui révèlent l'existence d'un IDS en observant certains comportements sur le réseau ciblé, notamment l'existence d'une interface en mode promiscuité, la mesure du temps de latence et l'exploitation des réponses actives des IPS.

Bien que répandus dans les organisations aujourd'hui, les systèmes de détection d'intrusions ne représentent qu'un maillon d'une politique de sécurité. En effet, même s'ils permettent la détection et parfois l'arrêt des intrusions, ils restent néanmoins vulnérables eux aussi face aux attaques externes.

1.3.2.4 Investigation dans le Cloud

De nombreux outils sont disponibles pour des investigations numériques usuelles. Il s'agit entre autres de : SANS SIFT, ProDiscover Basic, Volatility, The Sleuth Kit (+Autopsy), FTK Imager, CAINE, Oxygen Forensic Suite 2013 Standard, Digital Forensic Framework, HELIX3 Free, ENCASE [56, 57]. Tous ces outils sont inefficaces pour une investigation en matière de Cloud computing. En effet, l'investigation dans le Cloud présente de nombreuses difficultés :

— perte du contrôle des données ;
— pas d'accès physique aux infrastructures ;
— problème de multiples juridictions ;
— absence d'outils d'investigation de large spectre adaptés aux environnements distribués et virtualisés ;
— pas de collaboration des « provider » ;
— difficulté de produire des preuves numériques admissibles devant une juridiction.

1.3.2.5 Police judiciaire

La police judiciaire regroupe toute les activités qui visent à rechercher et à arrêter les auteurs des infractions pénales. Au Cameroun, le Procureur de la République est le chef de la police judiciaire et les officiers de police judiciaire sont ses auxiliaires. Le code de procédure pénal détaille clairement la qualité d'officier de police judiciaire. Ce dernier en tant que cheville ouvrière du service de la police judiciaire, a la charge de la recherche de la preuve ainsi que des auteurs d'une infraction. Pour cela, il est doté de certains privilèges suivant qu'il pose ses actes d'enquête en préliminaire ou en flagrance.

Lors de la phase policière (enquête préliminaire ou enquête de flagrance), les offi-

ciers de police judiciaire disposent d'un certain nombre de prérogatives leur permettant de rechercher les preuves nécessaires pour établir la culpabilité ou l'innocence des suspects [9]. En effet, ils ont la possibilité de faire des contrôles d'identité, de recueillir des témoignages, des aveux ou encore des preuves matérielles. Des auditions peuvent avoir lieu s'ils estiment que cela est nécessaire et des procès-verbaux devront dans ce cas, être dressés. Des perquisitions et saisies peuvent être effectuées sous le stricte respect de la loi.

Toutefois, lorsque la situation l'impose, une enquête judiciaire peut se tenir en commission rogatoire. Quand la délégation vise une autorité judiciaire ou de police étrangère, il s'agit alors d'une commission rogatoire internationale.

Dans le cadre de nos travaux, la flangrance désignera les attaques en cours sur un système d'information pour lesquelles les officicers de police judiciaire ont été saisis.

1.3.2.6 Perquisition numérique

La perquisition est une mesure d'enquête qui consiste à rechercher des éléments de preuve d'une infraction, au domicile d'une personne ou dans tous lieux où ils peuvent se trouver. Ainsi, si les éléments de preuve d'une infraction se trouvent dans l'univers numérique, la perquisition numérique consiste alors en la recherche d'éléments de preuve sur internet. Cette recherche d'éléments de preuve peut nécessiter la perquisition de l'outil informatique ayant servi à réaliser l'infraction ou à se connecter sur internet [31, 32]. Ce type ou cette forme de perquisition peut être qualifiée de perquisition numérique ou de cyberperquisition, intervenant à distance via le réseau internet. La cyberperquisition est opérée à partir du poste de l'officier de police judiciaire ou du juge d'instruction vers le système où l'infraction a été commise ou vient de se commettre. Elle permet à celui-ci de fouiller toutes les données présentes sur un ordinateur, de lire les frappes clavier, activer le micro, la webcam, regarder les fichiers. Autrement dit, les enquêteurs pourront voir et enregistrer en temps réel, à distance, les données informatiques telles qu'elles s'affichent sur un ordinateur, même lorsque les données ne sont pas stockées sur le disque dur (lecture d'un CD-Rom, saisie de texte en live sur Internet, etc.). De nos jours, des mouchards sont installés à distance ou physiquement sur un ordinateur incriminé pour servir de logiciel d'écoute. L'opération est semblable au système actuel d'interceptions de communications téléphoniques.

La perquisition numérique suppose alors trois éléments majeurs :
— l'accès à des données informatiques,
— la captation des données informatiques ;
— la conservation ou le stockage des données captées pouvant servir de preuve.

Internet véhicule des informations, ces informations peuvent servir à commettre une infraction ou avoir été destinées à la commettre, elles pourraient aussi être le produit de cette infraction ou représenter les avantages tirés de sa commission. C'est alors que la perquisition d'un système informatique, sa saisie ou la saisie des données ou informations qu'il contient devient nécessaire.

1.3.2.7 Saisies et scellés numériques

Les algorithmes de chiffrement permettent de réaliser les scellés numériques. Le crime informatique est un crime sophistiqué, réalisé le plus souvent au niveau international avec parfois un effet à retardement. Les traces laissées dans les systèmes sont immatérielles et difficiles à collecter et à sauvegarder. Il s'agit d'informations numériques mémorisées sur toute sorte de supports : mémoire, périphériques de stockage, disque dur, disquettes, clé USB, divers composants électroniques, etc [33]. Il se pose alors la question de leurs saisies lors d'une perquisition informatique. Les questions ci-dessous énoncent la problématique de la saisie des preuves numériques :

— comment identifier les données pertinentes ?
— comment les localiser ?
— comment les sauvegarder ?
— comment constituer une preuve recevable auprès d'un tribunal ?
— comment récupérer des fichiers effacés ?
— comment prouver l'origine d'un message ?
— comment remonter jusqu'à l'identité d'une personne sur la base uniquement d'une trace numérique du fait qu'il est difficile d'établir une correspondance certaine entre une information numérique et son auteur (dématérialisation) et de l'usurpation d'identité fréquente ?
— quelle est la valeur d'une trace numérique en tant que preuve contribuant à établir la vérité auprès d'un tribunal (notion de preuve numérique) sachant que les supports de mémorisation sur lesquelles des traces ont été recueillies sont faillibles (les notions de date et d'heure sont variables d'un système informatique à l'autre et aisément modifiables).

Les traces informatiques sont d'autant plus difficiles à obtenir lorsqu'elles sont laissées dans des systèmes ressortissants de différents pays. Leur obtention relève de l'efficacité de l'entraide judiciaire internationale et de la rapidité d'intervention. Leur exploitation efficace pour identifier des individus dépend de la durée de traitement de la requête d'obtention qui peut mettre un certain temps et qui rend alors toute identification impossible.

1.4 Détection des intrusions et preuves numériques

Plusieurs solutions de sécurité matérielles et logicielles existents pour protéger les SI. Il s'agit des logiciels de parefeux et des routeurs, les antivirus, les agents d'authentification, les VPN et les IDS. Ces solutions n'offrent une protection que dans la mesure où l'ensemble des communications vers l'extérieur passe systématiquement par leur intermédiaire et qu'ils sont correctement configurés. Malheureusement, les hackers peuvent arriver à passer à travers les règles de sécurité des dispositifs sus-cités, notamment en exploitant les éventuelles failles des systèmes protégés [14, 29]. Parmis les solutions existantes en matière de sécurité des SI, les IDS apparaissent comme la solution de sécurité réseau la plus utilisée. En effet, il s'agit d'un mécanisme destiné à repérer les activités normales ou suspectes sur la cible analysée (réseau ou hôte) pour avoir une connaissance sur

les tentatives des intrusions réussies ou échouées. Cependant, les fonctionalités des IDS se limitent à détecter les intrusions en envoyant des alertes conséquentes à l'administrateur. Ces dispositifs ne sont pas capables de préserver les preuves inhérentes à ces intrusions. De ce fait, il est non seulement difficile d'apprécier les domages causés sur un SI par une attaque mais aussi, de rassembler les informations utiles sur cette attaque afin d'identifier et d'interpeller les suspects. Toutefois, bien que les limites subsistent, beaucoup de travaux ont été effectués dans le but de permettre aux IDS de produire des informations utiles pouvant être utilisées comme preuves numériques dans une procédure judiciaire.

Dans [15], Brian Cusack et all. ont évalué la performance d'un NIDS dans un réseau filaire suivant différentes charges de travail (flux de données à examiner par un IDS) pour établir la capacité des NIDS à fournir des preuves numériques. Après avoir effectué des analyses d'attaques de type injection SQL et cross site scripting détectées par les NIDS $Snort$, Suricata et Bro-IDS, ces auteurs ont établi que la capacité de détection des intrusions décroissait rapidement lorsque la charge de travail augmentait.

Dans [16], Jorge Herrerias et all. ont prouvé que les fichiers logs constituent une véritable source d'information pour une investigation. Cependant, ces fichiers journalisent les activités provenant de plusieurs applications. Il en résulte un volume de données très important qui rend difficile l'exploitation des fichiers logs au cours d'une investigation. De ce fait, ces auteurs ont proposé une méthode de collecte, de filtrage, de normalisation et de corrélation des fichiers logs provenant des applications diverses. Bien que ces travaux minimisent les efforts à fournir par un enquêteur pour rechercher les preuves dans les logs, l'approche proposée ne décrit pas le processus de translation des logs utilisés dans leur module de normalisation. Par ailleurs, la définition des pré et post condition est subordonnée à une bonne connaissance du langage LAMBDA utilisé pour la spécification.

Fahmid Imtiaz dans [17], s'est intéressé à l'utilisation légale des fichiers logs générés par un IDS dans une procédure judiciaire. En étudiant les législations australiennes, américaines et anglaises, l'auteur a conclu que la législation liée à l'utilisation des logs issus des IDS comme preuve dans une enquête en est un handicap sérieux.

Barhate et all dans [18], ont proposé une automatisation de l'investigation numérique associée au système de détection des intrusions. Selon cette approche, une fois que l'IDS détecte une intrusion, un message d'alerte est envoyé à l'administrateur, suivit de l'appel d'un outil d'investigation pour capturer l'état du système. Cette capture pourra alors en temps opportun être présentée comme preuve devant une juridiction pour prouver les dommages subits. Cependant, les auteurs n'indiquent pas dans leurs travaux comment les rapports générés par l'analyse de l'image du système à l'instant donné, peuvent être utilisés comme preuve dans une investigation numérique. Plus encore, les travaux ainsi décrits s'appuient uniquement sur les IDS basés sur l'approche par scénarios et de ce fait, ne se limitent qu'à la détection des attaques connues.

Bien que beaucoup de travaux aient été menés dans ce domaine, le problème de la détection de l'intrusion ainsi que celui de l'utilisation de ses extrants comme preuves numériques reste un champ de recherche ouvert en informatique si l'on en juge par le grand nombre d'ateliers et de conférences organisés dans le monde.

1.4.1 Méthodes de détection des intrusions

Pour bien manipuler un système de détection des intrusions, il est important de comprendre son fonctionnement interne et les méthodes que ce dernier utilise pour détecter les tentatives d'intrusions dans le cas d'un IDS et pour les bloquer dans le cas d'un IPS.

Approche par scénario (misuse detection) : Cette technique nécessite une connaissance préalable des techniques d'attaques utilisées par les pirates pour pouvoir en déduire des scénarios typiques. Elle est dite sans état "stateless", puisqu'elle ne tient pas compte des activités passées sur le système et s'appuie, comme pour les antivirus, sur une base de signatures d'attaques pour pouvoir repérer des tentatives d'intrusions et remonter des alertes, notamment en se basant sur un ensemble de caractères permettant d'identifier une activité intrusive ou en analysant la conformité protocolaire des paquets. Ainsi cette technique présente un inconvénient principal qui se traduit par le niveau de précision des signatures d'attaques. De plus, le niveau de détection des attaques dépend impérativement de la manière dont elles sont gérées, des mises à jour de la base des signatures et du nombre des règles présentes pour gérer les alertes. En général, cette approche se base principalement sur des techniques de recherche de motifs statiques ou dynamiques, ainsi que sur une analyse protocolaire pour vérifier la conformité (par rapport aux RFC) des flux qui circulent sur le réseau (figure 1.9).

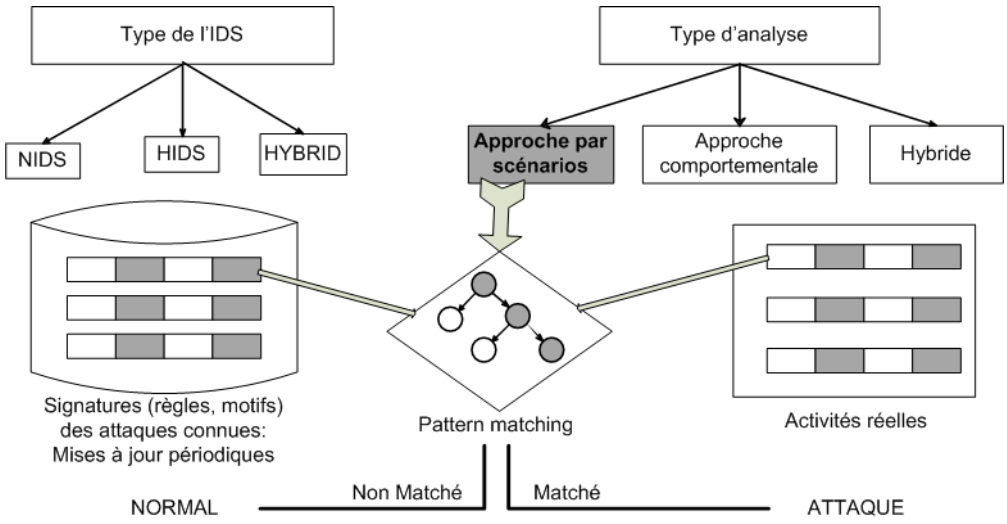


Figure 1.9 – Approche par scénario

Les techniques utilisées dans le cadre de l'approche par scénario sont les suivantes [21, 29] :

— *signature-based technique* ; une technique qui permet de détecter les intrusions en observant des événements et en identifiant les motifs qui correspondent aux signatures d'attaques connues. Une signature d'attaque définit les événements essentiels nécessaires pour effectuer l'attaque, et l'ordre dans lequel ils doivent être effectués ;

- *rule-based systems* ; une approche qui vise à détecter les intrusions en codant les scénarios d'intrusion comme un ensemble de règles. Ces règles reflètent une séquence partiellement ordonné des actions qui composent un scénario d'intrusion, certaines règles pouvant s'appliquer à plus d'un scénario d'intrusion ;
- *state transition analysis* ; une méthode consistant à modéliser les pénétrations comme étant une série de changements d'états, partant d'un état initial sécurisé à un état cible compromis . ;
- *data mining based techniques* ; dans cette approche, chaque instance dans un ensemble de données est étiquetée comme «normale» ou «intrusive». Un algorithme de détection est ensuite exécuté sur les données étiquetées pour permettre la détection des intrusions ;
- *genetic algorithms* ; les algorithmes génétiques utilisent la notion de sélection naturelle et l'appliquent à une population de solutions potentielles à un problème difficile (dont on ne sait pas trouver la solution optimale) pour trouver une solution approchée dans un temps raisonnable. La population de départ est l'ensemble des règles de détection basiques. A travers l'algorithme génétique, on génère d'autres règles qui couvrent au mieux les cas des flux anormaux.

Approche comportementale (anomaly detection) : Cette approche s'appuie principalement sur l'analyse du comportement passé des utilisateurs, des services ou des applications. Elle permet d'établir une certaine corrélation entre les différents événements sur un système et ainsi détecter des anomalies qui mènent à la détection de tentatives intrusives sur ce dernier. En effet, cette approche définie un comportement normal du système et considère toute déviation par rapport à ce comportement comme étant suspecte. De plus il est nécessaire de modéliser des comportements normaux pour le système à protéger en définissant un profil système qualifié comme profil normal. Par ailleurs, cette approche permet de détecter des attaques non connues, contrairement à l'approche par scénario présenté ci-dessus, puisqu'elle utilise des techniques heuristiques, probabilistes ou statistiques pour la détection des tentatives intrusives (figure 1.10). Cependant les IDS, qui adoptent cette approche, sont dotés de fonctionnalités d'adaptation et d'apprentissage afin de définir un comportement normal du système. Du coup la mise en place de cette approche nécessite plus de temps pour arriver à couvrir la totalité des systèmes à protéger.

L'approche comportementale comprend les techniques suivantes [22, 29] :

- *statistical methods* ; Le modèle statistique permet de déterminer, au vu de n observations $x_1, \ldots, x_n$ faites sur une variable x, si la valeur x_{n+1} de la $(n+1)me$ observation est normale ou non. Ces méthodes surveillent les utilisateurs ou le comportement du système en évaluant l'état de certaines variables dans le temps (par exemple, durée de connexion et déconnexion de chaque session). Les modèles de base gardent les valeurs moyennes de ces variables pour vérifier de temps à autre si les seuils sont dépassés ;
- *profiling methods* ; les profils de comportement normaux sont conçus pour différents types de trafic réseau, utilisateurs, programmes, etc, et tout éloignement de ces profils est considéré comme des intrusions ;

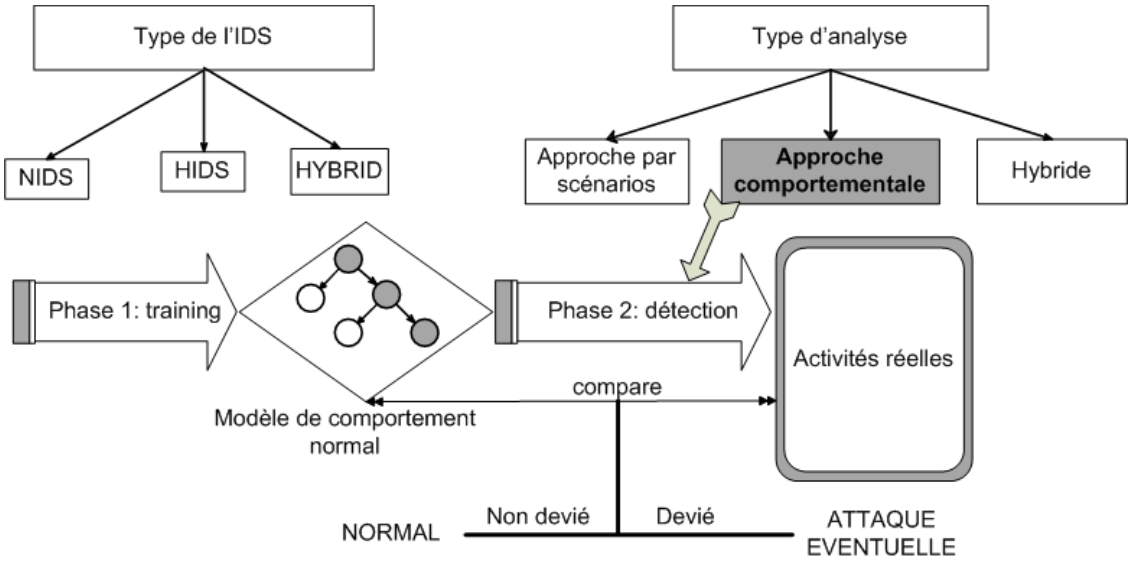


Figure 1.10 – Approche comportementale

— *model based approaches* ; différents types de modèles sont utilisés pour caractériser le comportement normal du système surveillé. Dans les approches basées sur des modèles, des anomalies sont détectées comme étant des déviations pour le modèle que représente le comportement normal ;
— *immunology* ; l'approche immunologique tente de calquer le comportement du système immunologique pour faire la différence entre ce qui est normal (le soi) et ce qui ne l'est pas (le non-soi). Cette approche vise à rechercher ce qui est mauvais en connaissant ce qui est bon.

En général, les IDS mélangent les différentes techniques de détection par scénario ou par comportement en proposant des combinaisons entre la recherche des motifs, l'analyse protocolaire et la détection d'anomalies.

1.4.1.1 Les systèmes de détection et prévention des intrusions (IDS/IPS)

La diversité des attaques utilisées par les pirates et des failles exploitées par ces attaques sur les différents niveaux des systèmes d'informations justifie l'existence de plusieurs types des IDS/IPS. En effet, pour assurer le bon fonctionnement de ces mécanismes, il faut savoir évaluer les risques encourus par un système d'information et classifier ces risques afin de déterminer le type de système de détection/prévention d'intrusion à mettre en place.

Ainsi plusieurs types d'IDS/IPS existent pour répondre à des problématiques précises selon le contexte d'utilisation de ces derniers et les fonctions qu'ils doivent remplir [24]. Les types d'IDS et leurs principales caractéristiques sont illustrés ainsi qu'il suit :

Les systèmes de détection d'intrusions réseau (NIDS) : Ce type d'IDS écoute tout le trafic réseau, analyse de manière passive les flux et détecte les intrusions en temps réel afin de générer des alertes. Ce type est le plus utilisé des IDS.

Les systèmes de détection d'intrusions de type hôte (HIDS) : Un HIDS est plus lié à une machine qu'à son réseau. Il permet d'analyser non seulement le trafic réseau, mais aussi l'activité se passant sur la machine. Le but de ce type d'IDS est d'assurer l'intégrité des données d'un système et d'analyser le flux de données relatif à une machine ainsi que ses journaux. Toutefois, ce type d'IDS présente une limitation qui se traduit par la nécessité d'avoir un système sain pour vérifier l'intégrité des données et donc un HIDS devient inefficace dans le cas ou il est implémenté sur un système déjà infecté.

Les systèmes de détection d'intrusions hybrides : Ce type d'IDS est utilisé dans un environnement décentralisé. Il centralise les informations en provenance de plusieurs emplacements (senseurs) sur le réseau. Le but de ce type d'IDS est d'avoir une vision globale sur les composants constituants un système d'information en permettant une supervision centralisée en matière d'alertes d'intrusions remontées par les NIDS et les HIDS présents sur l'architecture du réseau supervisé.

Les systèmes de prévention d'intrusions Kernel (KIDS /KIPS) : Ce type d'IDS fait partie de la famille des HIDS. Cependant, sa fonctionnalité principale est de détecter les intrusions au niveau noyau. L'utilisation d'un KIPS s'avère parfois nécessaire selon le niveau de sécurité à apporter pour une machine et permet de reconnaître, non seulement les caractéristiques des failles présentes sur un système, mais aussi d'interdire l'OS d'exécuter des appels systèmes qui peuvent s'avérer dangereux ou qui visent la compromission du système. Toutefois un ralentissement dans le fonctionnement peut être observé sur un système protégé par un KIPS. En effet, ce dernier analyse les appels systèmes et bloque tout accès suspect au système. Ainsi les KIPS sont des solutions rarement utilisées sur des serveurs souvent sollicités. Un exemple de KIPS est SecureIIS qui constitue une surcouche des serveurs web IIS de Microsoft.

1.4.2 Problématique

Bien que les informations collectées à travers les IDS et les autres fichiers logs d'un SI puisse être utilisées comme preuves numériques, le doute persiste sur l'intégrité, la fiabilité, et l'état complet de telles preuves étant donné que ces fichiers logs peuvent être altérés par un pirate.

Les défis à relever pour la collecte de la preuve dans une investigation numérique sont nombreux.

Préservation de preuves : les fonctionnalités des IDS se limitent à la détection / prévention des intrusions mais, ces outils sont incapables de préserver les preuves dans leur formes originales ;

Intégrité : les IDS actuels ne sont pas capables de préserver l'intégrité des informations qui sont nécessaires pour la recherche des preuves numériques ;

Etat complet des preuves : il est difficile de maintenir les preuves dans leur état complet étant donné que les fichiers logs qui les contiennent peuvent être corrompus ;

Détection de nouvelles attaques : les modèles actuels s'appuient sur l'approche par scénario pour la détection des intrusions. Par conséquent, ils sont limités pour la détection de nouvelles attaques.

1.5 Synthèse

Nous nous sommes attelés dans ce chapitre à faire un état de l'art sur la cybercriminalité et l'investigation numérique en ressortant les diverses problématiques. Par la suite, nous nous sommes appesentis sur les approches de collecte de preuve numériques potentielles produite par les systèmes de détection des intrusions étant donné que ces outils sont très utilisés dans les SI actuels comme dispositif de sécurité. Il apparaît clairement qu'au delà des menaces et vulnérabilités qui pèsent sur les SI, il existe de nos jours une activité commerciale dont l'objectif visé est l'effondrement des SI. Il s'agit de la cybercriminalité qui est l'une des formes de criminalité qui connaît actuellement la croissance la plus forte. Malheureusement, les auteurs des actes malveillants sur les SI bénéficient encore de certaines conditions favorables à la réalisation de leurs activités.

Les travaux actuels dans le cadre de la lutte contre la cybercriminalité permettent d'ores et déjà de déterminer les auteurs des actes criminels perpétrés dans le cyberespace. Cependant, malgré l'abondance de ces travaux, les auteurs de certains actes malveillants sont restés inconnus ou alors, les preuves n'ont pas pû être collectées à l'effet d'incriminer un tiers. En effet, la recherche et la collecte de la preuve au cours d'une investigation numérique présente encore de nombreuses difficultés. Ces dernières sont d'autant plus grandes lorsque les auteurs ont utilisé des logiciels ou matériels qui facilitent l'anonymisation, des techniques antiforensics ou anti-IDS.

Tout comme pour la criminalité traditionnelle, les techniques d'investigation en matière de lutte contre la cybercriminalité existent mais demeurent très peu efficaces. A cet égard, il devient très difficile de déterminer les auteurs des actes malveillants perpétrés dans un SI. De plus l'on ne saurait déclencher des actions pénales, si oui contre inconnus et généralement, moins de 10% de telles enquêtes aboutissent au regard des statistiques du Service Central des Recherches Judiciaires à la Gendarmerie Nationale. Face à cette réalité cybernétique, il est nécessaire de mettre sur pied un cadre formel d'investigation ayant un large spectre. Ce faisant, les étapes antérieures et postérieures doivent être prises en considération afin qu'un modèle complet soit réalisé.

CHAPITRE 2

CADRE FORMEL DE LA DÉTECTION DES INTRUSIONS

2.1 Introduction

Le rôle joué par le système d'information dans l'évolution de la société a conduit à une nouvelle forme d'économie dont les leaders seront ceux qui pourront maîtriser leur système du point de vue sécuritaire. Etant donnée la qualité des données gérées par le système d'information en vue de la prise de décision, la question sécuritaire devient de plus en plus cruciale. Parmi les défis auxquels font face les systèmes d'information, la question de la détection des attaques est le défi majeur qui doit être discuté en priorité parce que toutes les intrusions commencent par une attaque et, celle-ci se compose de diverses activités malicieuses.

A travers l'étude des processus métiers et des workflow dont les ressources sont les piliers, nous présentons dans ce chapitre une modélisation des SI en vue de la détection d'une attaque et de la collecte des preuves. La technique utilisée, baptisée "resource behaviors technique" permet de décomposer le SI en sous éléments constitués de ressources, d'états, de processus métiers et de workflows. Dans ce SI, une politique de sécurité y compris sa mise à jour est définie. Ainsi, tout évènement qui conduit à la violation ou à la transgression de la politique de sécurité constitue une attaque.

Après la partie introductive, nous donnons dans la deuxième partie de ce chapitre, les préceptes de la modélisation d'un système d'information. Dans la troisième partie, nous définissons le cadre formel de la modélisation en nous appesentissant sur la démarche de détection des intrusions et de la collecte de preuves. Ce chapitre s'achève par une sysnthèse à la quatrième partie.

2.2 Modélisation du système d'information

2.2.1 Justification

Une intrusion est une violation de la sécurité logique d'un système informatique [23]. Le mécanisme d'audit de sécurité, consistant en l'enregistrement de tout ou partie des actions effectuées sur le système pour en faire une analyse a posteriori, permet de détecter des intrusions. Les systèmes d'information sont devenus la cible des criminels étant donné qu'ils jouent un rôle vital au sein d'une organisation. La mise en place des méthodes efficaces pour les proteger est devenue une necessité. La sécurité d'un SI est compromise lorsqu'une attaque est réussie.

L'exemple le plus illustratif est l'attaque contre le système d'information estonien en 2007 par des pirates russes [70]. Pour y parvenir, les criminels ont utilisé un système appelé « Botnet » pour envoyer des spams à plusieurs services importants (banques, gouvernement, etc.) pour saturer le réseau estonien. Les infrastructures de ce pays étant essenciellement gérées par des outils techniques et technologiques, toutes les activités se sont vues interrompues.

Un autre exemple est l'action d'un virus appelé « Stuxnet » créé pour infecter le programme nucléaire iranien [71]. Ce virus a réussi sa mission en 2010 en compromettant le processeur utilisé pour contrôler la vitesse de rotation des centrifugeuses de la station nucléaire. En effet, une clé USB infectée et utilisée par une personne autorisée dans un système nucléaire protégé avait corrompu tout le système. Le virus « Stuxnet » s'était répandu dans plusieurs ordinateurs et avait compromis la majorité des postes de travail trois mois plus tard. Les processeurs ont ainsi été poussés par un ordre malicieux à cacher aux administrateurs les vraies informations sur la vitesse d'exécution des centrifugeuses. Finalement, en peu de temps, toutes les centrifugeuses étaient détruites et le programme nucléaire arrêté pour deux ans. Dans ce dernier cas, il apparait qu'une bombe n'a pas été fabriquée pour attaquer le programme nucléaire iranien mais un virus nommé « Stuxnet » a agi plus efficacement. Il existe de nombreux exemples similaires qui montrent et justifient l'importance de la gestion de la sécurité dans un SI.

2.2.2 Préliminaires

La technique du Workflow mining a été utilisée pour modéliser un SI. Il s'agit d'une technique de modélisation basée sur les logs du workflow qui contiennent des informations sur les processus workflow tel qu'ils sont réellement en train d'être exécuté dans le système d'information. Le cycle de vie du workflow tient sur quatre phases [72, 73] :

(1) la conception qui permet la construction du modèle du workflow sur la base des informations disponibles et des objectifs à atteindre avec les activités identifiées et les contraintes y relatives ;

(2) la configuration qui traite des limites et des particularités du système de gestion

du workflow qui en résulte, pour atteindre certains objectifs de l'organisation ;

(3) la mise en œuvre qui traite de l'intégration de nouvelles fonctionnalités dans le workflow utilisé, selon les nouvelles exigences qui ont été identifiées sur la base de la stratégie de l'organisation. Dans cette phase, le système d'information est arrimé à la vision de l'organisation ;

(4) le diagnostic qui consiste à analyser les données obtenues des exemples d'exécution du workflow. Cette analyse peut conduire à reprendre la conception du workflow ou à donner des informations pour la conception d'un nouveau workflow qui complète le cycle de vie du précédent.

Pour effectuer les tâches de mise en œuvre et de diagnostic du workflow, le Workflow mining est utilisé comme guide [25, 73]. L'objectif du Workflow mining est d'inverser les processus et de collecter les données d'exécution pour soutenir la conception et l'analyse du workflow. L'information collectée pendant la phase d'exécution d'un workflow est utilisée pour tirer un modèle qui explique les évènements enregistrés dans le système d'information. La modélisation d'une instance de workflow prend généralement en compte l'idée de conception mais, elle n'exprime pas exactement ce qui sera fait en pratique. Le constat est le même dans le domaine de la sécurité puisque la politique sécuritaire est définie en fonction de l'idée de conception du responsable du système de sécurité et non ce qui est réellement fait dans le monde réel. Pour ce faire, les modèles sont souvent normatifs en ce sens qu'ils énoncent ce qui devrait être fait et non ce qui est réellement fait dans le processus de workflow. Ce type de modèle est par conséquent subjectif. Pour qu'un modèle soit objectif, son concepteur doit prendre en compte les données qui sont produites par l'exécution réelle des cas de workflow. Les données obtenues sont donc liées aux activités, aux contraintes de temps et aux ressources responsables de leur production. Le Workflow mining peut donc être très contributif dans le problème de détection de l'intrusion puisqu'il permet d'établir ce qui y est réellement fait en construisant un modèle correspondant à la situation réelle ou à l'état du système d'information. Ce modèle peut donc être comparé au modèle perçu aussi connu sous le nom de "modèle à priori" en vue de mettre en exergue l'écart entre les deux modèles. Le "modèle à priori" est généralement la perception de la politique sécuritaire du système d'information qui est défini par un ensemble d'actions à réaliser par des utilisateurs précis sur la base de ressources données. Le "modèle à priori" est conçu sur la base d'informations obtenues pendant la phase d'exécution de toutes les activités dans le système d'information. Ces données connexes sont stockées dans les fichiers logs du système d'information. Ces fichiers logs conservent donc les traces de tous les évènements qui arrivent dans un système d'information pour une analyse future et une prise de décision.

Une surveillance étroite des évènements qui ont lieu pendant l'exécution permet aussi de détecter les différences entre la conception telle que faite pendant la phase de conception de la politique sécuritaire et l'exécution enregistrée pendant la gestion de l'intrusion. La technologie du workflow mining tend de nos jours vers plus de flexibilité opérationnelle pour prendre en compte la gestion des exceptions dans le workflow. La plupart des exceptions peuvent être vues comme des intrusions dans le système. Par conséquent, les utilisateurs du système d'information peuvent "à priori" s'écarter de la politique sécuritaire du système d'information. Les données étant enregistrées dans les

fichiers logs du système, ledit écart peut être suivi en vue de l'éliminer de la procédure de détection d'intrusion. Il ne fait aucun doute que les techniques de Workflow mining peuvent être utilisées pour créer un feedback et, adapter le modèle de workflow aux changements de circonstances afin de détecter les imperfections pendant et après la phase d'exécution. L'utilisation du Workflow mining dans ce travail n'est pas destinée à montrer comment les modèles de workflow sont construits à partir des évènements des fichiers logs du système, mais de détecter des intrusions à partir des fichiers logs sur la base d'une politique de sécurité d'un système d'information prédéfini.

2.2.2.1 Entités du système d'information

Toutes les composantes d'un système d'information peuvent être porteuses de vulnérabilités. Le système d'information d'une organisation peut être présenté comme un ensemble d'entités sur lesquelles reposent des éléments essentiels (fonctions et informations) et sur lesquelles peuvent porter des menaces. Dans ce chapitre, l'intérêt est porté sur les menaces dont la cause est malveillante.

La typologie proposée dans la méthode d'expression des besoins et identification des objectifs de sécurité (EBIOS) identifie six grands types d'entités [74] :
— les logiciels ;
— les matériels ;
— les réseaux ;
— les sites ;
— les organisations ;
— les personnels.

Cette typologie permet notamment de distinguer le cadre organisationnel, l'environnement physique et humain dans lequel le système d'information et informatique sont exploités.

Ainsi, on distinguera par exemple :
— l'organisation du siège de l'organisme ;
— son personnel ;
— ses sous-traitants ;
— son bâtiment, sa salle informatique et ses bureaux ;
— ses réseaux d'énergie et de télécommunications ;
— ses serveurs, postes de travail et supports de sauvegarde ;
— son réseau informatique interne et ses accès Internet ;
— son parc applicatif.

Les principaux centres d'intérêt seront essentiellement les logiciels, matériels et réseaux qui, possèdent des vulnérabilités exploitables dans le cadre des attaques cybernétiques.

2.2.2.2 Sémantique dénotationnelle

La notion algébrique de l'ordre partiel complet sera utilisée dans la suite, pour définir la sémantique dénotationnelle [75] de la modélisation des processus métiers ainsi que des workflows dans un système d'information.

Définition 2.1. *(Ensemble partiellement ordonné)*

Soit C un ensemble. Un ordre partiel $\subseteq$ défini sur C est un sous ensemble de $C\ X\ C$ qui satisfait les conditions suivantes pour tout c_1, c_2 et c_3 dans C.

(1) $c \subseteq c$ (reflexivité)

(2) si $c_1 \subseteq c_2$ et $c_2 \subseteq c_3$ alors $c_1 \subseteq c_3$ (transitivité)

(3) si $c_1 \subseteq c_2$ et $c_2 \subseteq c_1$ alors $c_1 = c_2$ (antisymetrie)

Ce domaine mathématique ne s'applique pas seulement aux ensembles arbitraires mais également aux fonctions car il est possible de construire des ensembles de fonctions partiellement ordonnées. Un ensemble partiellement ordonné de fonctions de type $C_1 \rightarrow C_2$ peut dériver de l'ordre partiel complet établi sur C_1 et C_2.

2.2.2.3 Modèle de description de l'environnement

Des travaux qui ont été menés dans la modélisation des activités des organisations, l'on constate que l'impact de l'environnement d'exécution est régulièrement négligé. C'est l'un des aspects qui fragilise la sécurité des SI. Par exemple, dans les pays en voie de développement, en allant d'un hôpital à l'autre, il sera donné de constater que la façon de pratiquer la chirurgie varie même si les chirurgiens sont des produits de la même école. Ceci est dû au fait que les hôpitaux ne sont pas équipés de la même manière. C'est donc pour cela que dans la modélisation d'une activité, l'environnement dans lequel elle sera menée doit être pris en considération.

Un environnement est considéré comme étant est un ensemble de métrologies dont la valeur peut changer [25]. Ces métrologies sont des primitives d'observation booléennes dénotées par des observateurs. La valeur associée de chaque observateur dépend de l'état de l'environnement à ce moment-là.

Formellement un environnement E est défini comme un triplet $\langle \Theta, S, val \rangle$ où :
- Θ est un ensemble non vide d'observateurs ;
- S est un ensemble non vide d'états ;
- $val : \Theta \rightarrow (S \rightarrow Bool)$ est une fonction qui décrit le comportement des observateurs.

Dans la suite, $val(o)(s)$ sera dénoté par $s(o)$ dans lequel s denote un état, o un observateur, $s(o)$ est la valeur de l'observateur o dans l'état s. Etant donné un état s, l'ensemble des observateurs dont la valeur est vraie, définit la caractéristique de s. Cet ensemble est $S_c = \{o \in \Theta, s(o) = true\}$.

Etant donné deux états $s1$ et $s2$ de l'ensemble des états S de l'environnement E, l'ensemble des observateurs dont les valeurs associées ne sont pas les mêmes est défini à

partir des caractéristiques des deux états. Cet ensemble est appelé gap entre $s1$ et $s2$ et est dénoté $s1 \bullet s2 = (s1_c - s2_c) \cup (s2_c - s1_c)$.

Etant donné un environnement E, les observateurs dans Θ définissent l'alphabet qui permet d'analyser les évènements qui arrivent à E. Le langage définit à partir de cet alphabet est dénoté par l'ensemble des conditions ou formules C. Une condition $c \in C$ est une affirmation sur les observateurs et est définie comme une formule du premier ordre. Les éléments essentiels de C sont donc tous les observateurs de Θ. Les éléments de C sont constitués comme suit :

$$\begin{cases} \text{if} & \text{o} \in \Theta, \text{then } o \in C \\ \text{if} & \text{o} \in \Theta, \text{then } \neg o \in C \\ \text{if} & \text{o}_1, o_2 \in \Theta, \text{then } o_1 \wedge o_2, o_1 \vee o_2, o_1 \Rightarrow o_2 \in C \end{cases}$$

Une condition c peut être décomposée en un ensemble d'observateurs $+c$ dont la valeur est vraie et un ensemble d'observateurs $-c$ dont la veleur est fausse. Les deux ensembles n'ont rien en commun, c'est à dire $+c \cap -c = \emptyset$.

Etant donnée une condition $c \in C$ et un état s, c est satisfait dans l'état s si le résultat de son évaluation est positif, c'est à dire, $s(c) = true$.

2.2.2.4 Etat d'un environnement

Un état est une image instantanée d'un environnement à un moment précis. A partir de cette image, des faits sont observés. Certains de ces faits ou caractéristiques d'un état sont vrais ou faux à ce moment précis. Ces faits sont présentés comme des formules équivalentes au calcul des prédicats. De façon rigoureuse, ces faits et relations seront désignés comme des attributs d'un état. Ainsi, si F est considéré comme un ensemble de formules, et s comme un état, alors s est un élément de F c'est à dire $s \in F$.

En général, soit S un ensemble d'états, selon la définition d'un état, $(S, \subseteq)$ est un ensemble partiellement ordonné. Dans le processus de modélisation, l'intérêt est porté sur le fait que S ait un état minimal $\perp_S$, connu comme l'état initial d'une entreprise ou d'un workflow à partir duquel l'exécution peut commencer. Cet état initial est donc contenu dans tous les états de S c'est à dire que pour tout $s \in S$, $\perp_S \subseteq s$. Par ailleurs, S dispose d'une borne supérieure $\bigcup_S$ connue comme l'état où les objectifs de la structure sont atteints. Ainsi, l'ensemble des états d'un processus est dressé selon le concept de l'ordre partiel complet (Complete Partial Order ou CPO)[25].

2.2.2.5 Modèle de description de tâche

Une tâche est une activité atomique qui ne peut pas être divisée en plus petites activités. La réalisation ou l'exécution d'une tâche transforme l'état de l'environnement en un autre état. Une tâche est donc une action à l'intérieur de l'état d'un environnement. Avant qu'une tâche ne soit exécutée, l'état de l'environnement doit satisfaire une condition spécifique appelée pré-condition et, quand cette exécution est terminée, une autre condition

appelée post-condition. Une tâche est formellement définie par un triplet $< nt, pre, post >$ où nt représente le nom de la tâche, pre sa pré-condition, et $post$ sa post-condition.

L'action d'une tâche à l'intérieur d'un environnement est de transformer un état en un nouvel état. Lorsque $< t, pre, post >$ est une tâche, s un état donné où la pré-condition pre est satisfaite c'est à dire $s(pre) = true$, l'action de t dans l'état s est le nouvel état t_s qui satisfait la post-condition $post$ c'est à dire $t_s(post) = true$. En général, l'action d'une tâche t dans l'état s est caractérisée par les observateurs de s dont la valeur a été modifiée.

Définition 2.2. *(Action d'une tâche)*

Soient $E = \langle \Theta, S, val \rangle$ un environnement, s un état donné et t une tâche dont la pré-condition est satisfaite dans s, alors l'action de t dans s est dénotée par t_s, spécifiée par $t_s = \{o : \Theta, s(o) \neq t(s)(o)\}$.

Quand il n'y aura aucune ambigüité, une tâche sera représentée par son nom t. $pre(t)$ et $post(t)$ dénoteront respectivement sa pré-condition et sa post-condition. Sur la base de la post-condition d'une tâche t et l'état s où $s(post(t)) = true$, l'on conjecture que $t_s = +post(t) \cup -post(t)$.

Définition 2.3. *(Tâches conflictuelles)*

L'action des tâches dans un environnement peut être conflictuelle puisque beaucoup de tâches peuvent modifier les mêmes observateurs en même temps. Pour cela, t_1 et t_2 sont des tâches conflictuelles dans l'état s, et l'on note $overlap(t_1, t_2, s)$, si et seulement si : $\begin{cases} \text{s(pre(t}_1\text{))} = s(pre(t_2)) = true \\ +\text{post(t}_1\text{)} \cap -post(t_2) \neq 0 \\ +\text{post(t}_2\text{)} \cap -post(t_1) \neq 0 \end{cases}$

Définition 2.4. *(Décalage)*

Soient SoT un ensemble non vide de tâches et s un état donné, un décalage noté Shf est un couple $Shf = < s, SoT >$ composé de l'état s et la somme des tâches non conflictuelles SoT dans s.

Formellement, soit $Shf = < s, SoT >$ un décalage, les propriétés suivantes sont satisfaites : $\begin{cases} \text{SoT} \neq \emptyset \quad (1) ; \\ \forall t \in SoT, s(pre(t)) = true \quad (2) ; \\ \forall t, t' \in SoT, t \neq t' \Rightarrow overlap(t, t', s) = false \quad (3) \end{cases}$

Soit $Sht = < s, SoT >$ un décalage, les actions simultanées de SoT dans s, notées SoT_s, sont prises en compte par l'ensemble des observateurs dont les valeurs sont modifiées dans s, c'est à dire : $\text{SoT}_s = \bigcup \left\{ \begin{array}{l} \text{o} \in \Theta : \\ \text{o} \in -post(t_i) \end{array} \right\}, \text{t}_i \in SoT$

Définition 2.5. *(Chaîne)*

Une chaîne est une ligne d'exécution de tâches selon leurs actions dans les états et leurs conditions de déclenchement. La chaine notée $P = \prod_{i=1}^{n} Sht_i$, est spécifiée comme une séquence finie de décalages où n représente la longueur de la séquence.

Soit P une ligne de longueur $n > 1$, $sh_k = \langle\ s_k, st_k\ \rangle$, $sh_{k+1} = \langle\ s_{k+1}, st_{k+1}\ \rangle$ dénotent respectivement les décalages d'amplitude k et $k+1$, l'état s_{k+l} est l'état résultant de l'exécution de l'ensemble des tâches st_k c'est à dire $s_{k+1} = st_k(s_k)$. Quand il n'y aura aucune ambiguïté, le décalage d'amplitude k de la ligne P sera noté $P(k)$.

Soient $Sht_k = \langle\ s_k, SoT_k\ \rangle$ et $Sht_{k+1} = \langle\ s_{k+1}, SoT_{k+1}\ \rangle$ deux décalages où $Sht_k = SoT_k(s_k)$, la différence entre les états s_k et s_{k+1} est notée $\overline{s_k + s_{k+1}}$ et est définie comme suit :

$\overline{s_k + s_{k+1}} = SoT_k(s_k)$

Lemme 2.1.

Soient ch un chemin d'exécution et $t \in SoT(ch(k))$ avec $k \leq length(ch)$ alors il existera toujours m tel que $m > k$ et $(post(t), S(ch(m))) = false$.

Preuve :

Soit ch un chemin d'exécution. $ch = (T, S)^*$ où S est un état et T est un ensemble de tâches actives, concurrentes et non conflictuelles.
Par ailleurs, $(T, S)^* \rightarrow (T(S))^*$. C'est à dire $(T1, S1) \rightarrow T2(S2)$.....$(Tn-1, Sn-1) \rightarrow Tn(Sn)$ avec $S_{i+1} = S_i \cup act(T, S_i)$.
Si $length(ch) = m$ alors, $ch(m) = (T_m, S_m)$, où S_m est l'état de satisfaction du chemin d'exécution ch. Ceci implique qu'aucune autre tâche ne peut s'exécuter. C'est à dire $T_m = \emptyset$ et par conséquent pour tout $ch_k = (T_k, S_m)$ tel que $m > k$, si $t \in T_k$ alors $(post(t), S_m) = false$. □

Lemme 2.2.

Soit ch un chemin d'exécution, alors $SoT(ch(length(ch))) = \emptyset$

Preuve :

(Par l'absurde)
Soit un ch un chemin d'exécution tel que $ch(m) = (T_m, S_m)$ et $length(ch) = m$. Supposons que ch est une ligne d'exécution et que $T_m \neq \emptyset$.
Alors, $T_m \neq \emptyset$ implique qu'il existe $t \in T_m$ telle que $(pre(t), S_m) = true$.
Ceci implique qu'il existe S_{m+1} tel que $S_{m+1} = S_m \cup act(T, S_m)$.
Ceci implique que $length(ch) \geq m + 1$. Ce qui est faux par hypothèse. Conclusion, $T_m = \emptyset$.

□

Définition 2.6. *(Ordonnancement d'états)*

Soient P une ligne de longueur $n > 1$, Sht_k = $\langle$ s_k, SoT_k $\rangle$ et Sht_{k+1} = $\langle$ s_{k+1}, SoT_{k+1} $\rangle$ deux décallages consécutifs dans P avec $k < n$ alors $s_k \subseteq s_{k+1}$ specifie le fait que l'ensemble des observateurs modifiés dans s_k après les actions de SoT sont contenus dans l'ensemble des observateurs de s_{k+1} avec les mêmes valeurs.

Lemme 2.3.

Soient P un chemin d'exécution, S l'ensemble des états de P, alors $(S, \subseteq)$ est un CPO où la borne supérieure est le dernier état de P et la borne inférieure est le premier état de P.

Preuve :

Selon la définition de l'état d'un environnement, $(S, \subseteq)$ est un ensemble partiellement ordonné.

Soit ch un chemin d'exécution. $ch = (T, S)^*$ où S est un état et T est un ensemble de tâches actives, concurrentes et non conflictuelles.

Par ailleurs, $(T, S)^* \rightarrow (T(S))^*$. C'est à dire $(T1, S1) \rightarrow T2(S2)$.....$(Tn-1, Sn-1) \rightarrow Tn(Sn)$ avec $S_{i+1} = S_i \cup act(T, S_i)$.

Etant donné que $(S, \subseteq)$ est un ensemble partiellement ordonné, il existe un état minimal S_1 pour lequel correspond un chemin d'exécution minimal $ch_1 = (T, S_1)$. ch_1 qui est le premier état de P est donc la borne inférieure de l'ensemble des états de P.

De même, il existe un état maximal S_m pour lequel correspond un chemin d'exécution maximal $ch_m = (T, S_m)$. ch_m qui est le dernier état de P est donc la borne supérieure de l'ensemble des états de P. □

Rendu à cette étape de la modélisation, il est important de s'assurer que l'exécution d'une tâche t s'arrêtera à un certain moment. Pour ce faire, l'ensemble des observateurs qui devraient être modifiés par t doivent être partiellement ou totalement contenus dans les observateurs qui constituent sa pré-condition c'est à dire :

$$(-pre(t) \cup -pre(t)) \cap (-post(t) \cup +post(t)) \neq \emptyset.$$

Partant de la définition de la ligne d'exécution des tâches, l'on spécifie la relation à l'intérieur de l'ensemble T des tâches sur la base de l'ensemble S des états. Cette relation est notée $\trianglelefteq$.

Définition 2.7. *(Ordonnancement des tâches)*

Soient T un ensemble de tâches, t_1 et t_2 deux tâches de T, $t_1 \trianglelefteq t_2$ si et seulement si pour toute chaine CH telle que si n_{t_1} et n_{t_2} dénotent respectivement l'amplitude maximale de t_1 et de t_2 dans CH, alors $n_{t_1} \leq n_{t_2}$.

Cette relation a les propriétés suivantes :

1. *reflexivité* : $t \trianglelefteq t$ ceci signifie que la tâche t appartient à la chaîne CH ;
2. *antisymmétrie* : si $t_1 \trianglelefteq t_2$ et $t_2 \trianglelefteq t_1$ dans la chaîne D alors $t_1 = t_2$. Par convention, il existera toujours une ligne de chaque tâche à elle-même ;
3. *transitivité* : si dans la chaîne CH, $t_1 \trianglelefteq t_2$ et $t_2 \trianglelefteq t_3$ alors $t_1 \trianglelefteq t_3$.

Lemme 2.4.

L'ensemble des tâches T associé à la relation précédemment définie $\trianglelefteq$, c'est à dire $(T, \trianglelefteq)$, forme un ordre partiel complet.

Preuve :

L'ordre partiel complet est établi par construction dans la définition 2.7 ci-dessus. □

2.2.2.6 Palette

Définition 2.8. *(Palette)*

Soient E un environnement et S un ensemble d'états distincts que E pourrait atteindre selon les actions des tâches T, alors une palette P est un couple $< E, S \rightarrow S >$. Dans la suite, l'ensemble des fonctions $S \rightarrow S$ sera noté T, l'ensemble des tâches de la palette. Quand il n'y aura pas d'ambigüité, $E(P)$ et $T(P)$ dénoteront l'environnement et l'ensemble des tâches de la palette P respectivement.

Les actions de l'ensemble des tâches T de la palette P dans l'environnement E consistent à changer au moins une fois la valeur de chaque observateur de Θ dans E. A cet effet, les actions consécutives d'un ensemble de tâches non vide dans un environnement ne pourraient pas modifier tous les observateurs dans cet environnement. L'ensemble des observateurs dont la valeur ne change pas pendant l'exécution d'un quelconque ensemble de tâches non vide sera abstrait de tous les états possibles de l'environnement, c'est-à-dire : $\forall o \in \Theta \begin{cases} \exists\, t_1 \in T, \text{o} \in -post(t_1) \text{ ou} \\ \exists\, t_2 \in T, \text{o} \in +post(t_2) \end{cases}$

Etant donnée une palette P, en fonction des changements d'environnement à l'intérieur des organisations et les différentes exécutions de tâches qui peuvent avoir lieu, différentes manières d'exécuter les tâches doivent être appréhendées. Il sera utilisé dans la suite SP_p pour spécifier l'ensemble des lignes d'exécution qui peuvent être obtenues d'une palette P.

Lemme 2.5.

Soient P une palette, $s \in S(P)$ un état donné de l'environnement $E(P)$ de P. Il existe toujours un chemin d'exécution $p \in SP_p$ tel que $s \in S(p)$, où $S(p)$ denote l'ensemble des états du chemin p.

Preuve :

Soit $P =< E, T >$ une palette. $s \in S_P$ implique que $S_P = S_{P-1} \cup act(T, S_{P-1})$.
Ceci implique qu'il existe une chaine d'exécution ch_P telle que $ch_P = (T, S_P)$. □

Lemme 2.6.

Soient $P =< E, T >$ une palette et $t \in T$. Il existe un chemin d'exécution $ch \in SP_p$ où SP_p denote un ensemble de lignes d'exécution possibles de T, $ch(n)$ = $\langle\ s_n$, $SoT_n\ \rangle$ tel que $t \in SoT_n$.

Preuve :

Découle du Lemme 2.5 ci-dessus. □

2.2.2.7 Modèle du Business Process

Un business process est un ensemble d'activités ou tâches destinées à produire un résultat spécifique pour les clients. Il implique une attention particulière sur la façon dont le travail est fait dans une organisation en vue de produire un service particulier [76]. Un processus est donc un ordre spécifique d'activités de travail dans un temps et un espace, avec un début, une fin, des inputs et outputs clairement définis. L'output est la raison pour laquelle l'organisation fait ce travail et se définit en termes de profit que le processus rapporte à l'organisation dans son ensemble.

Définition 2.9. *(Un service)*

Un service est la caractéristique d'un business process et se définit comme une composition d'un ensemble de critères qui caractérisent ce qui est produit au sein d'une organisation, où chaque critère est représenté par un observateur.

Le modèle d'un business process se définit comme un couple $< P, G >$ où P est une palette et G le service à rendre. Selon la définition de la palette, le classement des tâches est explicitement appréhendé par leurs pré-conditions et les états de l'environnement dans lequel leur exécution est faite. Cette approche réduit le nombre de modèle à utiliser pour appréhender les différentes façons de classer des tâches.

Lemme 2.7.

(1) Il existera toujours un état S_{lub} tel que lorsqu'il est atteint, les autres états ne peuvent pas être atteints. Cet état est appelé état borne supérieure du business process.

(2) Il existera toujours un état S_{ini}, point de départ de l'exécution du business process. Cet état est appelé borne inférieure du business process. Pour chaque service associé à un business process, un ensemble de qualités de service est défini pour tenir compte du travail quotidien et de la pression concurrencielle.

Preuve :

Soit $BP =< P, G >$ un business process avec $P =< E, S \to S >$ et $T : S \to S >$.
D'après le Lemme 2.4, $(T, \trianglelefteq)$ est un ordre partiel complet et toute chaine d'exécution $ch(BP)$ a une amplitude maximale ou minimale. □

2.2.2.8 Modèle de la Qualité de Service (QoS)

La qualité du service notée QoS représente les performances du service qui déterminent le niveau de satisfaction projeté par le bénéficiaire du service. Le niveau de satisfaction est défini par un ensemble de propriétés, critères, caractéristiques et performances des services rendus aux clients. Dans la littérature, il n'y a pas encore un consensus autour de la définition de l'ensemble de critères communs pour évaluer la qualité du service rendu dans les organisations. Les critères d'évaluation sont définis en fonction des objectifs et des spécificités de chaque entreprise. Un modèle abstrait qui donne la sémantique de la qualité de service est défini dans cette partie.

Définition 2.10.

Soient Cr un ensemble de critères considérés dans l'évaluation de la qualité de service, Val l'ensemble des valeurs qui peuvent être attribuées à ces critères, et f une fonction définie par $f : C \to Val$. La QoS est définie par (C, Val, f).

Etant données deux QoS q_1 et q_2 telles que $q_1 = (Cr_1, Val_1, f_1)$ et $q_2 = (Cr_2, Val_2, f_2)$, q_1 et q_2 sont compatibles et l'on note $q_1 \triangle q_2$ si et seulement si $C_1 = C_2$ et $Val_1 = Val_2$. Lorsque q_1 et q_2 sont compatibles, on dit que q_1 est meilleure que q_2 et l'on note $q_1 \subseteq q_2$ si et seulement si $\forall c \in C_1$, $f_1(c) \leq f_2(c)$. Il sera utilisé dans la suite $(\Phi, \subseteq)$ pour dénoter l'ordre partiel complet des qualités de service compatibles.

Définition 2.11. *(Business Process bien défini)*

Soit, $BP =< P, G >$ un business process, BP est bien défini si et seulement si tous les observateurs qui constituent son objectif (service) sont contenus dans l'ensemble des observateurs de l'environnement E c'est à dire $-G \cup +G \subseteq \Theta$ (E).

Définition 2.12. *(Business Process bien formé)*

Soit $BP =< P, G >$ un business process, BP est dit bien formé si et seulement si chaque chaine d'exécution SCH atteint la borne supérieure S_{lub} qui satisfait le service G c'est à dire

$$\begin{cases} \forall ch \in SCH n_{ch} \in N, S_{lub} \in S \\ \mathrm{n}_{ch} = length(ch) \\ \mathrm{S}_{lub} \\ \mathrm{S}_{lub}(G) = true \end{cases}$$

Plus formellement, soit SCH l'ensemble non vide des différentes chaînes qui peuvent être obtenues d'un business process BP, et $CH \in SCH$ avec la longueur n_{CH} tel que l'état n^{th}_{CH} de CH satisfait G, c'est-à-dire $S_{lub}(G) = true$.

Définition 2.13. *(Absence d'impasse)*

Soit BP un business process, BP est déclaré sans impasse si et seulement si il garantit que chaque chaîne d'exécution arrivera à sa borne supérieure atteignant l'objectif du business process BP.

Théorème 2.1.

Soit BP un business process tel que BP est bien défini et bien formé, BP est donc sans impasse.

Preuve :

Soit $BP =< P, G >$ un business process bien défini et bien formé. Par définition :

(1) tous les observateurs qui constituent sont objectif sont contenus dans l'ensemble des observateurs de son environnement.

(2) chaque chaine d'exécution de BP atteint sa borne supérieure en assurant son service.

De (1) et (2), il en découle que toutes les chaines d'exécution arriveront à leurs bornes supérieures en atteignant l'objectif du business process BP. Ainsi, BP est déclaré sans impasse. □

Toutes les lignes d'exécution d'un business process commencent par le même état noté S_{ini}. Il peut facilement être démontré que l'ensemble des états S_{BP} associés à la relation de classement $\subseteq$ telle que définie précédemment est un ordre partiel complet.

2.2.2.9 Modèle d'agent générique

Il y a beaucoup d'agents qui participent à la réalisation des tâches dans un système d'information. Un système d'information qui s'occupe de la réalisation des tâches est un système hybride qui inclut des composantes matérielles avec des logiciels attachés,

les acteurs humains interagissant avec les matériels et les logiciels. Les agents réalisent des tâches en vue d'entreprendre certaines missions qui, à leur tour, ajoutent une autre dimension à la qualité de service [76]. Chaque agent a une capacité qui est caractérisée par (Sk, Tks, mch) où Sk est l'ensemble des compétences, Tks l'ensemble des tâches et mch une fonction qui donne pour chaque compétence $cp \in Sk$ l'ensemble de tâches $mch(cp) \in Tks$ qui peuvent être réalisées sur la base de cp avec $mch(cp) \neq \emptyset$. Quand il n'y aura aucune ambiguïté, la structure (Sk, Tks, mch) sera représentée par Sk. En fonction de l'organisation mise en place, l'ensemble des tâches réalisées par un agent est enregistré dans un journal.

Un journal est décrit par l'ensemble des tâches et l'ensemble des intervalles de temps dans lesquels ils sont réalisés. Soit $Pds = (TI, \subseteq, \cap, \triangle)$ un ensemble d'intervalles de temps tel que $(TI, \subseteq)$ est un ensemble partiellement ordonné avec ∂ l'intervalle de temps le plus petit, $\cap$ et $\triangle$ les deux fonctions définies comme suit $\cap : TI$ x $TI \rightarrow TI$ et $\triangle : TI$ x $TI \rightarrow Boolean$. Si p_1 et p_2 sont deux intervalles de temps, p_1 et p_2 chevauchent si et seulement si il existe un intervalle de temps p_3 tel que : $p_1 \cap p_2 = t_3 \Rightarrow \begin{cases} p_3 \triangle p_1 \wedge p_3 \triangle p_2 \\ p_3 \subseteq p_1 \wedge p_3 \subseteq p_2 \end{cases}$

où $\cap$ et $\triangle$ définissent respectivement la relation d'intersection et de chevauchement. Quand il n'y aura aucune ambiguïté, l'ensemble des intervalles de temps sera représenté par Pds. En se basant sur les concepts de tâches et d'intervalles de temps, le concept de journal est modélisé par $< Tks, Pds, g >$ où Tks est l'ensemble des tâches, Pds l'ensemble des intervalles de temps associés, et g une fonction définie par

$g : Tks \rightarrow Pds$ tel que $\forall\, t_1, t_2 \in Tks, t_1 \neq t_2 \Rightarrow \neg(g(t_1) \triangle g(t_2))$.

Définition 2.14. *(Modèle de ressource humaine)*

Un modèle de ressource Rm est défini par $< Id, Sk, Ex, Dy, f >$ où Id est son identifiant, Sk son ensemble de capacités, Ex l'ensemble des expériences associées, Dy le journal associé et f une fonction qui définit pour chaque capacité $sk \in Sk$ l'expérience associée $f(sk) \in Ex$.

2.2.2.10 Entreprise

Une entreprise est un système s'occupant de la fourniture de services sur la base d'une certaine qualité de service [76, 77]. Ce système est articulé autour des business process qui sont réalisés, des agents chargés de la réalisation des tâches associées, et du workflow qui en résulte. Un système d'information SI est modélisé par $(Io, BPs, Emps, WFs)$ où Io est son identification, BPs l'ensemble des business process qui peuvent être mis en œuvre, $Emps$ son ensemble d'agents qui participent à la réalisation des tâches, WFs son ensemble de workflow.

Un workflow est formellement défini par $(Ts, Es, Ps, h, g, q, Q)^+$ où Ts est l'ensemble des tâches non conflictuelles, Es l'ensemble des agents s'occupant de la réalisation de Ts dans les intervalles de temps Ps pour obtenir la qualité de service Q, h la fonction

$Ts \rightarrow Ps$ qui définit pour chaque tâche t, son intervalle de temps $h(t)$ dans lequel elle est réalisée, g une fonction $Es \rightarrow Ts$ qui donne pour chaque agent ag sa tâche associée $g(ag)$, et q une fonction $Es \times Ts \rightarrow Q$ qui donne pour un agent donné ag et sa tâche associée tk, la qualité de service $g(ag, tk)$ obtenue après l'exécution.

Selon les agents utilisés ou qui entrent en jeu dans une entreprise donnée et leur disponibilité ainsi que les services requis par les clients, les agents intervenant dans différents workflow associés à un business process ne seront pas nécessairement les mêmes. A cet effet, selon leurs capacités, la qualité des services rendus pourrait être différente. Les critères d'évaluation de la qualité de service seront donc parfois associés aux valeurs minimales quand les tâches seront réalisées par un agent avec peu d'expérience. Par contre, ces valeurs seront maximales quand un agent ayant une expérience maximale aura été impliqué dans la réalisation des tâches.

2.3 Cadre formel

Dans cette section, le cadre proposé pour le problème de détection des intrusions dans un système d'information est baptisé « Resource behaviors technique ». L'approche utilisée prend ses fondements sur la théorie du workflow présentée dans la section précédente. La construction du modèle commence par une description formelle générique de plusieurs ressources fondamentales du SI pouvant être sujettes à des attaques. Cette description est suivie de la définition du modèle normatif associé à la politique de sécurité du système d'information. Le modèle normatif est suivi de la définition du modèle descriptif associé au fonctionnement courant des différentes ressources du système d'information. La section s'achève par la définition d'une approche établissant la détection de l'intrusion sur la base de l'audit des instances d'exécution du workflow dans le système d'information. L'audit compare les données liées aux modèles normatif et descriptif.

2.3.1 Modèle du système d'information

Une ressource dans un système d'information est soit un acteur humain, soit des équipements, soit une fonction (programme) ou une donnée. Les ressources utilisent d'autres ressources pour réaliser des activités spécifiques à l'intérieur de certaines ressources. Les activités sont les différentes tâches ou fonctions qui peuvent être réalisées par des ressources à l'intérieur d'un système d'information. Chaque ressource est associée à un ensemble d'activités qu'elle pourrait réaliser, et un ensemble d'activités auxquelles elle peut être appliquée. En plus, étant donnée une ressource, un ensemble de traces d'activités auxquelles elle peut être appliquée ainsi qu'un ensemble de traces d'activités qu'elle réalise sont spécifiés. De la même manière, une ressource pourrait contenir des sous-ressources. Chaque sous-ressource dans une ressource donnée est considérée comme une ressource. L'ensemble des traces définit en réalité ce qu'on appelle les journaux des évènements utilisés dans le Workflow mining [25].

Définition 2.15. *(Ressource matérielle)*

Formellement, une ressource notée R est modélisée par
$< Id, Log\_in, Log\_out, Task\_in, Task\_out, SubRes >$ où :

- Id est l'identifiant d'une ressource ;
- Log_in représente le journal des activités entrantes ;
- Log_out représente le journal des activités sortantes ;
- $Task\_in$ représente l'ensemble des activités entrantes possibles ;
- $Task\_out$ représente l'ensemble des activités sortantes possibles ;
- $SubRes$ dénote l'ensemble des sous-ressources.

Quand il n'y aura pas d'ambiguïté, pour une ressource R donnée, les expressions suivantes seront équivalentes :

- $R(Id) = Id$;
- $R(Log\_in) = Log\_in$;
- $R(Log\_out) = Log\_out$;
- $R(Task\_in) = Task\_in$;
- $R(Task\_out) = Task\_out$;
- $R(SubRes) = SubRes$.

Etant donnée une ressource dans un système d'information, ses sous-ressources sont aussi des ressources de ce système d'information. Dans cette approche de modélisation, il existe deux types de ressources : les ressources intermédiaires et les ressources terminales. Une ressource r est dite intermédiaire lorsqu'elle dispose d'une ou de plusieurs sous-ressources. Autrement dit, r est appelée ressource terminale.

Les ressources sont n'importe quel agent qui exécute ou qui est capable de participer à l'accomplissement d'une activité dans le système d'information. Etant donnée qu'une activité elle-même est capable de lancer l'exécution d'une autre activité, il en est de même pour les ressources du système d'information. Ainsi, pour une tâche t à réaliser, beaucoup d'agents sont considérés. A cet effet, les différentes tâches qu'une ressource peut réaliser ou provoquer sont basées sur l'environnement du système d'information.

Définition 2.16. *(Evènement)*

Un évènement noté Ev, est caractérisé par
$< Id, Task, T\_In, T\_Out, Resp\_Res, Used\_Res, Task\_Act >$ où :

- Id est l'identifiant de l'évènement ;
- $Task$ représente l'activité de l'évènement ;
- T_in est le moment où débute l'évènement ;
- T_out est le moment où fini l'évènement ;
- $Resp\_Res$ est la ressource responsable de l'évènement ;
- $Used\_Res$ est la ressource utilisée ;
- $Task\_act$ représente l'action de la tâche.

L'exécution d'une activité donnée (tâche) est basée sur un état spécifique du système d'information. Un état S est formellement modélisé par $S = \bigcup_{r \in R}(S_r)$

où $S_r = Log\_in(r) \cup Log\_out(r)$.

Après avoir défini l'environnement, l'état et le modèle de ressource, l'on peut à présent étudier le système d'information.

Définition 2.17. *(Système d'information)*

Sur la base de tous les éléments précédemment définis, le système d'information est représenté par :

$SI = (R, S, BPs, WFs, BPs^{WFs}, Id\_Res^{Log\_in}, Id\_Res^{Log\_out}, Id\_Res^{Tr}, Id\_Res^{Task\_in-set}, Id\_Res^{Task\_out-set})$

où :

— R : est l'ensemble des ressources ;
— S : est l'ensemble des états ;
— BPs : est l'ensemble des business process qui peuvent être supportés dans SI ;
— WFs : est l'ensemble des workflow qui peuvent être mis en œuvre dans SI ;
— $BPs^{WFs} = BPs \underset{m}{\rightarrow} WFs$: est l'ensemble des fonctions qui donne pour chaque business process l'ensemble de workflow équivalent ;
— $Id\_Res^{Log\_in} = Id\_Res \underset{m}{\rightarrow} Log\_in$: est l'ensemble des fonctions qui définit pour une ressource donnée, les évènements pour lesquels elle participe ;
— $Id\_Res^{Log\_out} = Id\_Res \underset{m}{\rightarrow} Log\_out$: est l'ensemble des fonctions qui définit pour une ressource donnée, les évènements qu'elle initie ;
— $Id\_Res^{Tr} = Id\_Res \underset{m}{\rightarrow} Tr$: est l'ensemble des fonctions qui définit pour une ressource donnée, les conditions de sa mise en activité ;
— $Id\_Res^{Task\_in-set} = Id\_Res \underset{m}{\rightarrow} Task\_in - set$: est l'ensemble des fonctions qui définit pour une ressource donnée, l'ensemble des tâches pour lesquelles elle est impliquée ;
— $Id\_Res^{Task\_out-set} = Id\_Res \underset{m}{\rightarrow} Task\_out - set$: est l'ensemble des fonctions qui définit pour une ressource donnée, l'ensemble des tâches qu'elle peut réaliser.

Le modèle de système d'information défini ne prend pas en compte la façon dont différentes activités seront réalisées en fonction de l'opérabilité de diverses ressources. Ceci l'expose aux opérations malicieuses. Pour s'en préserver, il est nécessaire de définir une politique de sécurité qui indiquera comment les tâches et les ressources seront mises ensemble pour réaliser les multiples objectifs du système d'information.

2.3.2 Modèle de la politique de sécurité

La politique de sécurité à l'intérieur d'un système d'information est d'autant plus importante qu'elle indique la façon dont chaque ressource sera utilisée. Cette politique est un ensemble de règles et d'exigences qui doivent être respectées par toute entité connue

et autorisée à l'intérieur d'une organisation. La politique de sécurité est basée sur quatre principaux concepts :

— les ressources : ce concept est utilisé en vue de lister toutes les ressources de l'organisation dont l'utilisation peut violer les règles de sécurité ;
— les tâches : ce concept permet d'identifier plusieurs activités, fonctions ou tâches qui peuvent être exécutées de n'importe quelle façon dans l'organisation sur la base de ressources identifiées ;
— le temps : quand une activité donnée est donc exécutée au sein de l'organisation, la trace de cette exécution doit être gardée pour une éventuelle investigation. Pour cela, on pourrait chercher à savoir à quel moment l'exécution d'une tâche a eu lieu. La réponse à cette question importante est basée sur la définition des moments de début et de fin de l'exécution. Le concept de période qui est lié au concept de temps est donc crucial ;
— le déclencheur d'action : dans la gestion du système d'information, certaines ressources maintiennent les autres prêtes à être utilisées, c'est-à-dire que si la ressource maintenue est par exemple un programme installé dans la machine, il peut être exécuté à tout moment par une autre ressource. Ceci signifie que le programme a l'environnement requis pour son exécution. Pour contrôler l'exécution d'une telle ressource ou activité, on a besoin d'une condition favorable à l'exécution qui en résultera. Ceci signifie aussi que même si la pré-condition d'une activité est satisfaite, si la condition de déclenchement n'est pas satisfaite, alors cette activité ne pourra pas être exécutée.

Sur la base des concepts définis, la politique de sécurité SP est modélisée comme suit :

$$SP = (R, T_S, T_R, P_d, (R_{out}^{T_S}, R_{in}^{T_S}), (R_{out}^{f_{out}}, R_{in}^{f_{in}}), T_S^{T_R})$$

où

$f_x = R_x^{g_x}$ avec $g_x = P_d^{T_S}$

Les différentes fonctions définissent la relation entre les quatre concepts en vue de mettre un accent sur les différents évènements qui ont eu lieu dans le système d'information en violation de sa politique de sécurité. Sur la base de la politique de sécurité, le modèle normatif du système d'information peut maintenant être exprimé.

2.3.3 Démarche de détection des intrusions et de collecte de preuves

2.3.3.1 Modèle normatif du système d'information

En se basant sur l'application de la politique de sécurité dans le système d'information cible, et en l'absence de toute action externe qui viole cette politique, le modèle de système d'information en exécution dans une organisation dans ce cadre espace temps est le modèle normatif. Ce modèle noté SI_{nor}, est formellement spécifié comme suit

$$SI_{nor} = (SI; Runs, SP)$$

où :
— SI est le système d'information cible ;
— $Runs$ est l'ensemble des workflow qui ont été exécutés dans SI en appliquant la politique de sécurité précédemment définie ;
— SP est la politique de sécurité qui y est associée.

Le modèle normatif est loin de correspondre à la réalité du système d'information. Si l'on considère le grand nombre d'attaques auxquelles font face différents systèmes de sécurité dans le monde, le modèle normatif du système d'information n'est qu'une vue de l'esprit puisque chaque fois, les règles de sécurité sont violées du fait d'actions internes ou externes. Le nombre d'attaques déclarées ne traduit pas la situation réelle puisque dans beaucoup d'organisations, les gestionnaires ne révèlent pas les attaques au public. Ceci est parfois dû au fait qu'ils ne veulent pas frustrer leurs clients, ce qui aurait pour conséquence la perte de leur fidélité. Parfois, ils ne savent même pas que leur système d'information a fait l'objet d'une attaque. Compte tenu des observations faites sur le modèle normatif, la vraie situation du système d'information est représentée par le modèle descriptif.

Définition 2.18.

Soit SI_{nor} un système d'information normatif,

$log_{y=in|out}(r, SI_{nor})$ = $\{x \mid x$ est un évènement de SI et x cadre avec la politique de sécurité $\}$.

2.3.3.2 Modèle descriptif du système d'information

Ce modèle décrit le système d'information en utilisant l'exécution des activités réelles dans le système. Soit SI_{des} le modèle descriptif défini comme suit :

$SI_{des} = \langle SI, Runs \rangle$

où :
— SI est le système d'information ;
— $Runs$ est l'ensemble des événements dont les conditions de déclenchement sont satisfaites en vue de l'exécution des tâches.

2.3.3.3 Modèle de détection des Intrusions

Soient SI_{des} et SI_{nor} respectivement les modèles descriptif et normatif du système d'information SI. SI_{des} est dit conforme au système d'information normatif noté $SI_{nor} \models SI_{des}$ si et seulement si pour chaque ressource r les contraintes suivantes sont satisfaites :

$log\_y(r, SI_{des}) \subseteq log\_y(r, SI_{nor}) \wedge log\_y(SI_{des}) \subseteq log\_y(SI_{nor})$

où $log\_y(r, SI_x)$ dénote le journal associé à la ressource r dans le système d'information (normatif ou descriptif), avec $y = in \mid out$ et $x = des \mid nor$. Par ailleurs,

$log_y(SI_x$représente l'ensemble des évènements du journal du système d'information (normatif ou descriptif).

De la même manière, le modèle descriptif SI_{des} ne sera pas conforme au modèle normatif SI_{nor}, Ceci est dénoté par $SI_{nor} \nvDash SI_{des}$, si et seulement si il existe une ressource r dans le système d'information descriptif SI_{des} avec une activité qui a été exécutée en violant les règles de la politique de sécurité dans une période de temps $Period$ définie par $[T\_in, T\_out]$.

Lemme 2.8.

(1) Soient $p1$ et $p2$ deux périodes ;

$p1 \models p2$ si $p1[T\_in] < p2[T\_in]$ et $p1[T\_out] > p2[T\_out]$.

(2) Une activité tk a été exécutée en violation de la politique de sécurité si et seulement si une des contraintes suivantes est remplie :

$$\exists k \in E\_SI_{des}/ \begin{cases} \text{Task (log\_x(r, SI}_{des})[k]) = tk \wedge tk \notin task\_x(r), \\ x = out \mid in \\ \text{P\_x(task (log\_x(r,SI}_{des})[k])) = pk \wedge pk \nvDash P\_x(r), \\ x = out \mid in \\ \text{trigger((Task, SI}_{des})[k]) = tgk \wedge tgk \notin Tg(Task) \end{cases}$$

où :

- $Task(log\_x(r, SI_{des})[k])$ est l'activité associée à la ressource r qui était consignée dans le log_x à travers le système d'information descriptif SI_{des} rapporté par l'évènement k ;
- $task\_x(r)$ est l'ensemble des activités qu'une ressource r peut réaliser ou, être impliquée dans leurs réalisation ;
- $P\_x(task(log\_x(r, SI_{des})[k]))$ est la période de temps pendant laquelle l'activité est réalisée ;
- $P\_x(r)$ est la période de temps durant laquelle une ressource r peut être mise en activité ;
- $trigger((Task, SI_{des})[k])$ dénote le déclencheur de la tâche associée à l'évènement k dans le système d'information descriptif ;
- $Tg(Task)$ est l'ensemble des déclencheurs associés à toutes les tâches dans SI_{des}.

Preuve :

(Par l'absurde)

Supposons que $p1 \models p2$ avec $p1[T\_in] > p2[T\_in]$ ou $p1[T\_out] < p2[T\_out]$.

Ceci implique qu'il existe $tk_1 \in Task$ tel que tk_1 s'exécute dans $p2[T\_in]$ et ne s'exécute pas dans $p1[T\_in]$; ou bien, il existe $tk_2 \in Task$ tel que tk_2 s'exécute dans $p2[T\_out]$ et ne s'exécute pas dans $p1[T\_out]$. Ceci est impossible puisque $p1 \models p2$. □

Définition 2.19.

Soient SI un système d'information, r une ressource, log_in et log_out ses fichiers journaux (ou fichiers logs). La détection de l'intrusion notée $ID(SI)$ est identifiée comme suit :

$$ID(SI) = \bigcup_{i=1}^{m}(log\_in(r_i, SI_{des}) \setminus \text{log\_in}(r_i, SI_{nor}) \vee log\_out(r_i, SI_{des}) \setminus \text{log\_out}(r_i, SI_{nor}))$$

Théorème 2.2.

Soit SI un système d'information, si un évènement Ev de SI n'est pas conforme avec sa politique de sécurité que l'on note ($Ev \nVdash SP(SI)$), alors $Ev \in ID(SI)$.

Preuve :

Soient $SP = (R, T_S, T_R, P_d, (R_{out}^{T_S}, R_{in}^{T_S}), (R_{out}^{f_{out}}, R_{in}^{f_{in}}), T_S^{T_R})$ où
$f_x = R_x^{g_x}$ avec $g_x = P_d^{T_S}$, avec $x = in \mid out$, la politique de sécurité et Ev un évènement du système d'information SI.
Si $Ev \nVdash$ avec $SP(SI)$
$\Longrightarrow (R_{out}^{T_S}, R_{in}^{T_S}) \vee (R_{out}^{f_{out}}, R_{in}^{f_{in}}) \vee T_S^{T_R} \neq \emptyset$
$\Longrightarrow \exists r \in (Resp\_Res \vee Used\_Res)$ de Ev tel que $log\_y(r, SI_{des}) \nsubseteq log\_y(r, SI_{nor}) \vee log\_y(SI_{des}) \nsubseteq log\_y(SI_{nor})$
$\Longrightarrow$ pour ce $r \in (Resp\_Res \vee Used\_Res)$ de Ev, on a $(log\_in(r_i, SI_{des}) \setminus log\_in(r_i, SI_{nor}) \vee log\_out(r_i, SI_{des}) \setminus log\_out(r_i, SI_{nor})) \neq \emptyset$
$\Longrightarrow Ev \in ID(SI)$ □

Définition 2.20.

Soient Ev un évènement et SP la politique de sécurité de SI. Ev est dit conforme à SP si et seulement si $Ev = \langle Task, T\_in, T\_out, Resp\_res, Used\_res, Task\_out \rangle$ avec la condition suivante :

- Task ne peut être réalisée que si et seulement si sa condition de déclenchement est remplie
- Task ne peut être réalisée que pendant une période de temps T_in et T_out
- Resp_res et Used_res ne peuvent être utilisés par Task que pendant une période de temps T_in et T_out
- L'action d'une tâche sur Resp_Ress ou Used_ress appartient à Task_act

Définition 2.21. *(Fonction de transition du système d'information)*

C'est une fonction qui permet aux évènements de transformer un modèle normatif en un modèle descriptif équivalent selon la politique de sécurité. Cette fonction est définie comme suit :

$f : Event$ X $SI \rightarrow SI$ / Si $Ev \nvDash SP(SI)$ alors $Ev \in ID(SI)$ sinon $Ev \notin ID(SI)$

Définition 2.22. *(Collecte de preuves)*

Notons $Proof(SI)$, l'ensemble des preuves d'intrusions dans un système d'information SI.

$\forall Ev \in ID(SI)$, $Proof(SI) = \bigcup_{i=1}^{m}(log\_in(r_i, SI_{des}) \vee log\_out(r_i, SI_{des})$

$avec$r$_i \in Resp_Res \vee Used_Res$

où $Resp\_Res$ et $Used\_Res$ sont respectivement l'ensemble des ressources responsables et l'ensemble des ressources utilisées de l'évènement Ev.

Théorème 2.3.

En cas d'intrusion dans un système d'information, il est possible de déterminer les activités qui ont été exécutées en violation de la politique de sécurité. Les fichiers logs des ressources impliquées dans l'exécution de ces activités constituent les preuves.

Soit SI un système d'information. Le système d'information descriptif n'est pas conforme au système d'information normatif et l'on note $SI_{nor} \nvDash SI_{des}$ si et seulement si $Proof(SI) \neq \emptyset$

Preuve :

Soit SI, un système d'information.
$SI_{nor} \nvDash SI_{des}$
$\Longleftrightarrow ID(SI) \neq \emptyset$
$\Longleftrightarrow \bigcup_{i=1}^{m}(log\_in(r_i, SI_{des}) \setminus log\_in(r_i, SI_{nor}) \vee log\_out(r_i, SI_{des}) \setminus log\_out(r_i, SI_{nor}))$

$\neq \emptyset$

$\Longleftrightarrow \exists r^0 \in R$ tel que $\bigcup_{i=1}^{m}(log\_in(r_i, SI_{des}) \setminus log\_in(r_i, SI_{nor}) \vee log\_out(r_i, SI_{des}) \setminus log\_out(r_i, SI_{nor}))$ = $\bigcup_{i=1}^{\alpha}(log\_in(r_i^0, SI_{des}) \vee log\_out(r_i^0, SI_{des})$

$avec 1 < \alpha < m$

$\Longleftrightarrow \bigcup_{i=1}^{\alpha}(log\_in(r_i^0, SI_{des}) \vee log\_out(r_i^0, SI_{des}) = Proof(SI)$

$\Longleftrightarrow Proof(SI) \neq \emptyset$ □

2.3.4 Comparaison des modèles et discussion

De nombreuses approches de modélisation ont été développées afin de s'attaquer au problème de la détection des intrusions [78, 79, 80, 81]. Malgré la popularité de ces techniques, les solutions proposées sont loin d'atteindre l'objectif de la sécurisation des SI. En effet, ces solutions ne tiennent pas compte des différents types et des abstractions des ressources qui agissent dans un système d'information. Elles se focalisent uniquement sur le comportement des utilisateurs en faisant abstraction du fait que ces utilisateurs interagissent avec les ressources. Par conséquent, les solutions associées ne peuvent pas gérer efficacement la politique de sécurité du système d'information afin d'identifier ou empêcher l'intrusion. Il est possible de détecter ou d'empêcher efficacement les intrusions si les activités de chaque ressource sont bien fixées. Ainsi, cela permet de définir un modèle de sécurité normatif du système d'information qui doit être régulièrement comparé au modèle de sécurité descriptif. L'écart observé après cette comparaison est appelé intrusion qui doit être détecté ou empêché.

2.4 Synthèse

Ce chapitre offre un cadre formel qui modélise la détection des intrusions dans un système d'information ainsi que la collecte des preuves de ces intrusions. L'approche Workflow mining a été utilisée pour prendre en compte l'évolution du système en analysant ses ressources, son workflow, et ses journaux pour détecter les intrusions. Nous avons présenté les concepts permettant le fonctionnement des SI et avons abouti à la réalisation d'un modèle de détection des intrusions et de collecte de preuves. Une comparaison avec les techniques de détection existantes a été faite et il en ressort que le modèle mis sur pied a l'avantage de contrôler le workflow entre les différentes ressources d'un SI.

Le modèle obtenu dans ce chapitre établit qu'il est possible de détecter une intrusion et de collecter des preuves en s'appuyant uniquement sur le workflow entre les ressources du SI. Cependant, après la détection des intrusions, la collecte de leurs preuves n'est en réalité qu'une étape de l'investigation numérique. Afin que les informations receuillis lors des intrusions dans un SI puissent servir de preuves véritables, il est essentiel que leur collecte soit encadrée par les exigences légales de traçabilité et de continuité de la preuve numérique. C'est la raison pour laquelle dans les deux chapitres suivants, nous mettons sur pied un cadre formel d'investigation capable de donner du crédit aux informations issues de l'outil qui sera implémenté sur la base du "resource behaviors technique" en vue de la détection des attaques sur les SI.

CHAPITRE 3

APPROCHE MÉTHODOLOGIQUE DE COLLECTE DE PREUVES NUMÉRIQUES

3.1 Introduction

Une fois qu'une intrusion est détectée, il est nécessaire de prendre des mesures pour collecter les preuves utiles non seulement afin d'apporter les éléments matériels de la constitution de l'infraction mais aussi pour établir la responsabilité des suspects. Cela n'est possible que par une investigation numérique.

Dans ce chapitre, le problème de l'investigation numérique est vu comme un ensemble de processus métiers qui doivent être accomplis pour permettre la collecte des preuves numériques et déterminer les auteurs d'une aggression sur un SI. Il en découle des procédures d'investigations composées de primitives d'investigations, elles mêmes constituées des actions à réaliser pour élucider une attaque. Prenant en compte les limites des modèles existants, nous nous appuyons sur les travaux de [4, 6, 7] pour mettre en œuvre un modèle d'investigation numérique multidimensionnel à trois composantes : proactive, active et réactive. De ce fait, l'effectivité d'une investigation numérique proactive qui vient soutenir selon le cas d'étude, la recherche et la collecte de la preuve numérique dans une investigation active ou réactive, est alors établie.

Après la partie introductive, nous présentons dans la deuxième partie un modèle multidimensionnel de l'investigation numérique. Une comparaison contributive entre ce modèle et les modèles existants est faite dans la troisième partie. Une synthèse clôture ce chapitre à la quatrième partie.

3.2 Modèle multidimensionnel de l'investigation numérique

De nos jours, l'utilisation des systèmes informatiques soit comme l'objet de crime, soit comme un instrument utilisé pour commettre un crime ou un lieu de stockage de preuves liées à un crime a conduit à l'apparition des cyber-enquêteurs. Quand une infrac-

tion implique les technologies de l'information et de la communication, une intervention rapide et méthodique doit être menée pour limiter et protéger le lieu du crime. Ces procédures sont indispensables, avant n'importe quelle analyse, pour assurer le contrôle et l'acceptabilité des preuves rassemblées et traitées en cas de poursuite judiciaire. Le développement de plusieurs modèles d'investigations par des pays et des organismes est la conséquence évidente de l'expansion du crime numérique. Certains s'intéressent à l'aspect technologique de l'investigation tant dis que d'autres se focalisent sur l'analyse de l'exploitation des données pour les utiliser devant un tribunal [87]. Très peu des modèles existants tiennent compte d'un procédé numérique proactif ou actif de recherche de la preuve.

Avant le déroulement de chaque phase d'investigations, les unités chargées des investigations peuvent être mises en mouvement par une plainte de la victime, une alerte émanant de la structure attaquée ou encore, un constat effectué lors des cyber-patrouilles. Il s'agit généralement du point de départ d'une investigation. A cet effet, les autorités doivent obtenir une autorisation de la victime, évaluer et confirmer l'attaque. Après avoir obtenu un mandat d'investigation du procureur de la république de céans, les enquêteurs formulent une stratégie et mettent sur pied un plan d'investigations. Il s'ensuit une coordination des différentes ressources en vue de l'application des procédures forensics.

Ces différentes phases doivent être effectuées avec minutie et peuvent prendre énormément de temps, alors que les enquêteurs, lors des perquisitions, ou de gardes à vue, n'en disposent pas beaucoup.

La méthodologie d'investigation (encore appelée méthodologie forensic) proposée a trois composantes (figure 3.1). Chaque composante est constituée de procédures d'investigations elles-mêmes constituées de plusieurs processus. Chaque processus renferme des primitives et les primitives se décomposent en actions forensics.

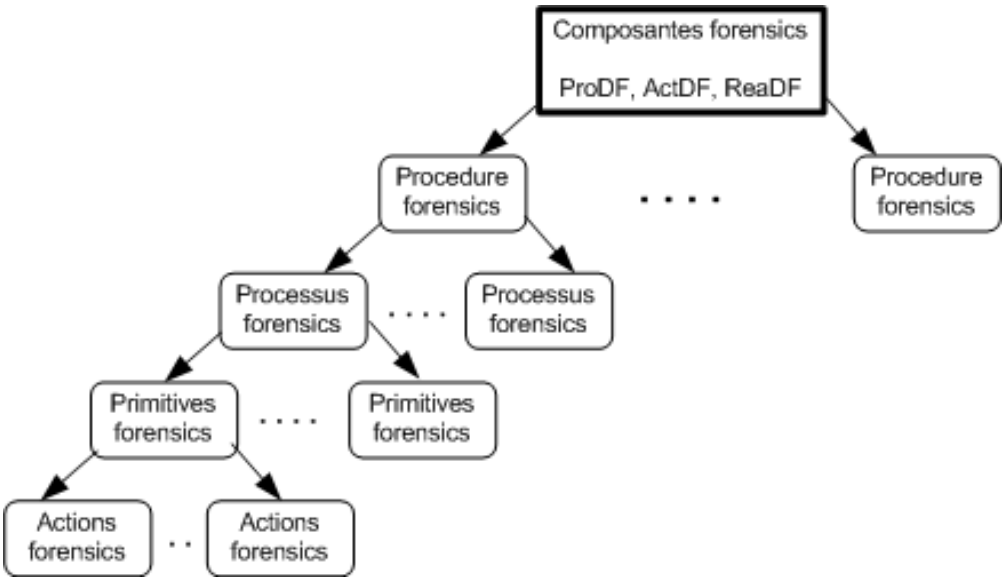


Figure 3.1 – Décomposition de la méthodologie forensic

3.2.1 Composante forensic Réactive (ReaDF)

La phase réactive de l'investigation numérique baptisée Réactive Digital forensic (ReaDF) se concentre sur la recherche traditionnelle qui aura lieu après qu'un incident ait été détecté et confirmé. Ceci inclut l'investigation physique, l'identification, la préservation, la collecte, l'analyse, et la production du rapport final. Les objectifs de l'investigation en mode réactif sont :

— étudier une attaque ;
— collecter des preuves ;
— déterminer les causes de l'attaque ;
— déterminer les auteurs de l'attaque ;
— confondre les auteurs de l'attaque.

Pour atteindre ces objectifs, l'organisation méthodique des opérations d'investigation conduit à l'obtention des procédures et primitives ci-dessous.

3.2.1.1 Procédures forensics

3.2.1.1.1 Procédure forensic « Physical investigation » La scène du crime englobe tout lieu sur lequel un cybercrime a été perpétré et dont l'examen nécessite l'intervention d'un groupe d'investigateurs. Il s'agit non seulement de l'endroit géographique où l'acte de cybercriminalité a été commis mais également tout élément qui y est lié : supports ou ordinateurs utilisés, butin et traces emportés, inforoutes, etc. La scène de crime se caractérise donc souvent par une multiplicité d'emplacements géographiques et une multiplicité de traces et d'objets.

La phase d'investigations physiques englobe les premiers actes que les enquêteurs posent dès leur arrivée sur la scène du crime. En effet, elle vise à identifier tout le matériel électronique ainsi que les individus qui ont permis ou participé à la commission du cybercrime. Ainsi, tous les objets et matériels présents sur les lieux doivent être identifiés et analysés. Les individus quant à eux doivent être identifiés et auditionnés. Toutes les preuves matérielles doivent être assemblées et mises sous scellées. Une documentation clairement élaborée doit permettre de décrire les actes qui ont été posés et de tirer les premières conclusions. La matérialisation de la procédure d'investigations physiques est traduite par l'algorithme suivant :

Algorithme 1 Physical investigation (ReaDF)

ENTRÉES: scène de crime
SORTIES: preuves matérielles

1: initialiser ;
2: protéger la scène du crime ;
3: repérer les traces et indices potentiels ;
4: auditionner les témoins présents ;
5: acquérir, identifier, analyser et rassembler les preuves matérielles ;
6: établir les croquis et plans ;
7: préserver les preuves ;

3.2.1.1.2 Procédure forensic « Digital investigation » La phase d'investigation numérique consiste à rechercher sur la scène de crime les traces et indices numériques qui caractérisent l'acte cybercriminel. Ceci passe par une identification, préservation, collecte et analyse de toutes les informations numériques dont le spectre s'étend aussi bien sur les données que sur les applications. Cette étape détermine le succès d'une investigation. Cette phase englobe l'acquisition et l'analyse de la preuve immatérielle. La matérialisation de la procédure d'investigation numérique est traduite par l'algorithme suivant :

Algorithme 2 Digital investigation (ReaDF)

ENTRÉES: scène de crime
SORTIES: traces et indices numériques
1: initialiser ;
2: identifier les éléments découverts ;
3: préserver ;
4: collecter ;
5: analyser ;

3.2.1.1.2.1 Processus forensic « Identification » L'identification consiste à localiser tous les éléments en particulier ceux présents dans les médias rencontrés, puis de réduire encore cet ensemble à des artefacts d'intérêts. La sous procédure d'identification se décline par l'algorithme ci-dessous :

Algorithme 3 Identification (ReaDF)

ENTRÉES: scène de crime
SORTIES: activités et dossiers d'identification des individus
1: initialiser ;
2: identifier les équipements électroniques ;
3: identifier les composantes volatiles ;
4: déployer une méthode d'acquisition ;
5: identifier les individus présents et ceux déjà partis ;
6: obtenir les détails des activités de ces personnes ;
7: classifier les individus (victimes, suspects, témoins) ;
8: repérer et noter les emplacements de chacun d'eux ;

3.2.1.1.2.2 Processus forensic «Preservation » La préservation consiste à isoler et à sécuriser les preuves physiques et numériques. La sous procédure de préservation se décline par l'algorithme ci-dessous :

3.2.1.1.2.3 Processus forensic « Collection » A travers ce processus, les enquêteurs prélèvent les traces et indices présents sur la scène de crime physique. Ensuite, ils prélèvent les preuves numériques en dupliquant le contenu des supports numériques.

Algorithme 4 Preservation (ReaDF)

ENTRÉES: scène de crime
SORTIES: scène de crime
1: initialiser ;
2: établir une barrière de sécurité ;
3: réaliser des croquis, plan, photographies et films ;
4: renforcer les sources d'énergie pour maintenir les entités volatiles actives ;

Ils utilisent également les logiciels appropriés pour restaurer les données cachées, supprimées, tronquées ou corrompues y compris les métadonnées et autres fichiers logs. La sous procédure de collecte se décline par l'algorithme ci-dessous :

Algorithme 5 Collection (ReaDF)

ENTRÉES: scène de crime
SORTIES: ensemble de preuves numériques
1: initialiser ;
2: collecter les preuves volatiles ;
3: collecter les preuves sur les supports de stockage ;
4: collecter les preuves sur les réseaux ;

Collecte de preuves volatiles : Les éléments volatiles sont les régistres, des caches, les tables de routage, les caches arp, les tables de processus, les fichiers systèmes temporaires, etc. La plupart des preuves se trouvant dans des équipements mobiles seront de nature volatile. La collecte des preuves volatiles pose un problème étant donné que l'état du périphérique et le contenu des mémoires peuvent être modifiés. Le choix entre collecter les données volatiles sur la scène de crime ou le faire plus tard dans un laboratoire forensic dépend de la situation en cours et de l'état d'alimentation en énergie électrique des équipements. Si l'équipement n'est pas alimenté à travers un accumulateur de charge, la totalité des informations sera très vite perdue. S'il n'y a pas d'autres alternatives pour maintenir la charge énergétique des équipements, alors le contenu des mémoires doit être imagé en utilisant les outils adéquats le plus rapidement possible. La combinaison de plusieurs outils d'investigations est nécessaire pour l'obtention de meilleurs résultats [3, 88]. A ce stade, l'enquêteur doit aussi vérifier la présence de logiciels malveillants installés par le suspect. La primitive de collecte de preuves volatiles se définit par l'algorithme ci-dessous :

collecte de preuves non volatiles : Il s'agit des informations qui se trouvent sur les supports de stockage tels que les disques durs, les cartes MMC, les clés USB etc. Les données issues des postes de travail qui sont synchronisés avec ces périphériques doivent être collectées [3]. L'intégrité et l'authenticité des preuves collectées doivent être assurées par des mécanismes de hachage, de protection en écriture, etc. Bien entendu, les câbles d'alimentation, les adaptateurs ainsi que tout autre accessoire doivent être récupérés. Les enquêteurs prendrons soin de rechercher les autres preuves de nature non électroniques telles que les mots de passe écrits, les manuels d'utilisation de matériels et de logiciels ou tout autre document y relatif, les documents

Algorithme 6 Collecte de preuves volatiles (ReaDF)

ENTRÉES: scène de crime
SORTIES: ensemble de preuves numériques volatiles
1: initialiser ;
2: vérifier la présence de logiciels malveillants ;
3: **si** les équipement peuvent être transportés en gardant leur énergie **alors**
4: déplacer les équipements dans un laboratoire approprié ;
5: extraire les informations volatiles avec les outils adaptés ;
6: **sinon**
7: extraire rapidement les informations des mémoires vives par les outils adaptés ;
8: **finsi**
9: réaliser l'ensemble des pièces à conviction ;

imprimés, etc. La primitive de collecte de preuves sur les supports de stockage est définie par l'algorithme ci-dessous :

Algorithme 7 Collecte de preuves non volatiles (ReaDF)

ENTRÉES: scène de crime
SORTIES: ensemble de preuves numériques non volatiles
1: initialiser ;
2: extraire les informations sur tous supports amovibles ;
3: assurer l'intégrité et l'authenticité des informations extraites ;
4: rechercher toute autre information non digitale ;
5: réaliser l'ensemble des pièces à conviction ;

3.2.1.1.3 Procédure forensic « analysis » A travers ce processus, les enquêteurs cherchent les preuves qui se rattachent aux crimes suspectés. Ils établissent une signification pour tous les éléments recueillis, établissent les relations entre les données et dressent des conclusions partielles. Cela peut être l'analyse du système de fichiers, l'examen du contenu d'un fichier, l'analyse d'un fichier de journalisation, l'analyse statistique ou n'importe quel autre type d'examens. Enfin, l'examinateur doit interpréter les résultats de cette analyse. La qualité de l'analyse dépend beaucoup de la formation de l'examinateur, de son expertise technique et de son niveau d'expérience.

La sous procédure d'analyse de preuves rassemblées est définie par l'algorithme ci-dessous :

3.2.1.1.4 Procédure forensic « Reconstitution » La reconstitution inclut l'empaquetage, le déplacement et le stockage. La procédure légale doit être suivie et documentée pour assurer que les preuves numériques collectées ne sont pas altérées ou détruites. Toutes les sources de preuves potentielles doivent être identifiées et étiquetées convenablement avant leur empaquetage. Les équipements saisis ne doivent pas être exposés à la pression, à l'humidité, à la chaleur, aux radiations électromagnétiques, à la poussière et à

Algorithme 8 Analysis (ReaDF)

ENTRÉES: scène de crime
SORTIES: document d'analyse
1: initialiser ;
2: revoir le plan de recherche ;
3: revoir la pertinence des outils utilisés et les outils d'expertise disponibles ;
4: développer les hypothèses ;
5: analyser les preuves recueillis ;
6: évaluer les hypothèses ;
7: obtenir et valider les résultats ;
8: documenter l'analyse ;

la moisissure. Le guide du NIST [89] souligne la nécessité d'utiliser des procédures convenables de transport et de stockage pour garantir le respect de la chaîne de traçabilité. A partir de cet instant, l'équipe d'investigation peut combiner les hypothèses et procéder à une reconstitution historique des faits. La procédure de reconstitution des faits est définie par l'algorithme ci-dessous :

Algorithme 9 Reconstitution (ReaDF)

ENTRÉES: scène de crime
SORTIES: documentation sur les faits
1: initialiser ;
2: établir une relation entre les évènements en se référant aux phases d'investigations physiques et numériques ;
3: établir une documentation ;

3.2.1.1.5 Procedure forensic « Present findings » En fonction de la nature du crime, les enquêteurs peuvent être amenés à présenter toutes les preuves numériques ou physiques devant une juridiction. A cet effet, ils résument et donnent des explications sur les hypothèses et les conclusions partielles tirées dans les processus précédents. La procédure de présentation des découvertes est définie par l'algorithme ci-dessous :

Algorithme 10 Present findings (ReaDF)

ENTRÉES: ensemble des pièces à conviction
SORTIES: preuves mis sous scellés
1: initialiser ;
2: catégoriser les pièces à conviction ;
3: réaliser les scellés physiques ;
4: réaliser les scellés numériques ;
5: justifier les hypothèses énoncées ;
6: élaborer les conclusions partielles ;

3.2.1.1.6 Procedure forensic « Incident closure » La procédure de clôture est définie par l'algorithme ci-dessous :

Algorithme 11 Incident closure (ReaDF)

ENTRÉES: scène de crime, documentation des étapes précédentes
SORTIES: rapport d'enquête
1: initialiser ;
2: donner l'historique des faits ;
3: décliner l'identité des auteurs ;
4: établir les activités de chaque auteur ;
5: établir les relations entre les preuves matérielles et les auteurs ;
6: décliner l'identité des victimes ;
7: décliner l'identité des témoins ;
8: rédiger la documentation de l'enquête ;

La composante ReaDF se résume en :

Algorithme 12 Reactive Digital Forensic

ENTRÉES: scène de crime
SORTIES: rapport d'enquête
1: initialiser ;
2: Physical investigation ;
3: Digital investigation ;
4: Reconstitution ;
5: Present findings ;
6: Documentation ;

3.2.1.2 Représentations

Le métamodèle UML fournit une panoplie d'outils permettant de représenter l'ensemble des éléments du monde objet (classes, objets, etc.) ainsi que les liens qui les relient. Toutefois, étant donnée qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues. Une vue est constituée d'un ou plusieurs diagrammes. Dans la suite, l'intérêt sera porté sur une vue statique (Diagramme de cas d'utilisation) qui représentera le système physiquement, et une vue dynamique (Diagramme d'activités) qui montrera le fonctionnement du système [90].

Les acteurs représentés dans la figure 3.2 de cette phase sont :
— « First responder » : il s'agit des intervenants matériels ou logiciels qui découvrent l'attaque en premier et sonnent l'alerte. Les enquêteurs se réfèrent à eux pour obtenir les premiers éléments sur l'incident ;
— « Investigator » c'est l'officier de police judiciaire en charge des investigations. L'investigateur est celui qui conduit les enquêtes numériques. Il s'agit d'une personne rompue à la tâche. Une connaissance approfondie des systèmes informatiques n'est pas pour lui une condition suffisante. Il lui faut aussi de bonnes connaissances de

la procédure pénale et de la législation en vigueur. En somme, un investigateur devrait avoir des connaissances sur les systèmes d'exploitation, les langages de programmation, la sécurité informatique et le hacking éthique, la loi sur les crimes informatiques, les affaires civiles et pénales, le management des systèmes d'information et des politiques de sécurité, les retombées sociopolitiques et l'impact sociopsychologique des ordinateurs sur la vie privée des individus.

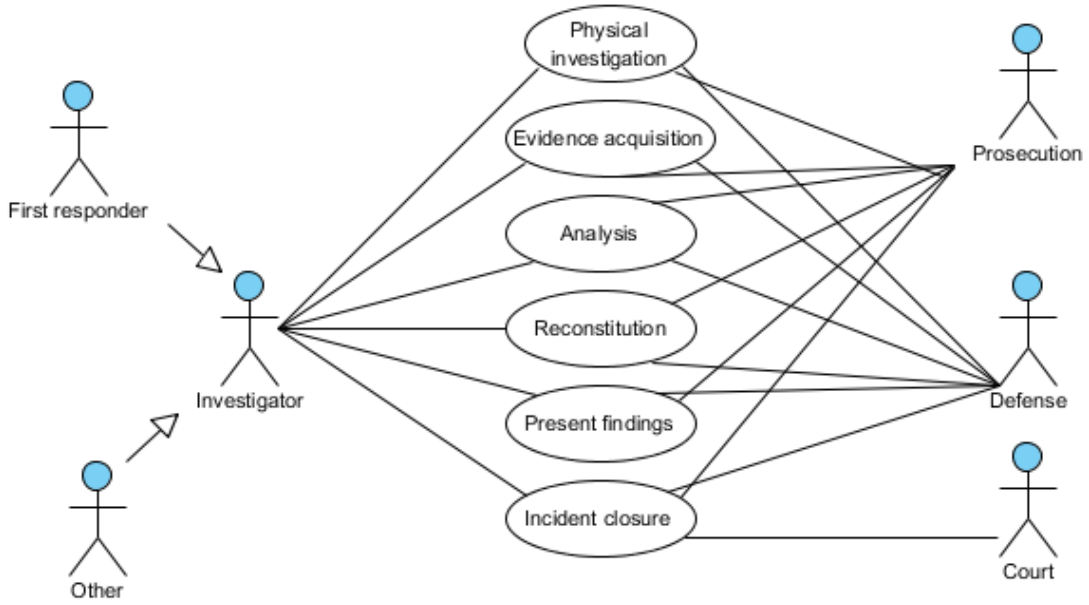


Figure 3.2 – Diagramme de cas d'utilisation (ReaDF)

— « Other » : ce sont tous les autres intervenants y compris les administrateurs systèmes et réseaux. Ces derniers apportent également leur soutien aux enquêteurs sans directement y être impliqué. Il peut aussi s'agir d'un CERT qui dans ses fonctions de cyberpatrouille, découvre qu'une organisation a été attaquée et dénonce auprès des autorités judiciaires. Ils ne posent pas les actes d'enquêtes.
— « Prosecutor » C'est le Procureur de la République qui a la responsabilité du déclenchement ou non des poursuites judiciaires. Il peut réaliser en cas de besoin, toutes les actions de l'officier de police judiciaire.
— « Defense » c'est la partie contradictoire. Elle peut procéder à tout moment à une contre expertise en posant les mêmes actes que les officiers de police judiciaire.
— « Court » c'est l'instance de jugement. Elle prononce ses sanctions en prenant en compte les rapports d'investigations produits éventuellement par toutes les parties prenantes au procès après la clôture de l'incident.

Le diagramme d'activités représenté dans la figure 3.3 suivante illustre et consolide la description du diagramme de cas d'utilisation précédent. Les doubles flèches de transition entre certaines actions indiquent que certaines actions peuvent être répétées.

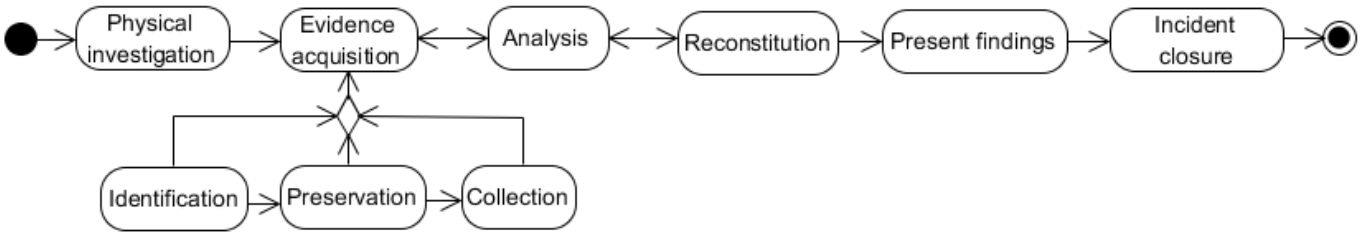


Figure 3.3 – Diagramme d'activités (ReaDF)

3.2.2 Composante forensic Active (ActDF)

L'investigation numérique active est une investigation qui se tient lorsqu'une attaque est en cours. En effet, certaines attaques s'étalent dans le temps. De petites attaques non observées par les organisations sont préalablement perpétrées avant qu'une attaque d'envergure ne soit découverte. Cette phase vise à rassembler l'ensemble des preuves disséminées dans le temps, correspondants chacune aux différentes attaques. Les objectifs de l'investigation active sont :

— réduire l'effet et l'impact d'un incident en cours ;
— rassembler les preuves numériques recevables (y compris les preuves volatiles) en employant les outils et technologies appropriés ;
— fournir les intrants suffisamment riches pour une investigation numérique réactive.

3.2.2.1 Procédures forensics

Les procédures qui suivent correspondent aux étapes qui soutiennent la phase d'investigations actives.

3.2.2.1.1 Procédure forensic « Evidence acquisition » Les mêmes techniques qu'à l'étape 2 de la phase d'investigations réactives s'appliquent. Cependant, des outils spécifiques sont utilisés pour profiler l'attaquant, rassembler la preuve volatile ou déterminer la source de l'attaque. Ces applications peuvent également servir à la collecte supplémentaire de preuves. Par la suite, il est nécessaire de sécuriser et authentifier toutes les données extraites. Pour s'assurer que la chaîne de traçabilité et de recueil de la preuve a été maintenue, toutes les actions effectuées doivent être documentées.

3.2.2.1.2 Procédure forensic « Analysis » Les mêmes techniques qu'à l'étape 3 de la phase d'investigations réactives s'appliquent. Les preuves primaires acquises doivent être analysées afin de déterminer si elles contiennent les renseignements suffisants pour la reconstruction de l'incident ou sont en conformité avec les hypothèses initiales. Elles sont d'une grande importance pour documenter progressivement toutes les activités qui sont effectuées. Il s'agit des différentes tâches réalisées, des actions qui sont entreprises dans

le but de s'assurer que les preuves sont complètement acquises. La régularité et la fiabilité des résultats doivent être assurées. La scène de crime doit être totalement examiné en employant les techniques ou les outils existants pour l'extraction des preuves. Cette pratique est indiquée lorsqu'il s'agit des systèmes de fichiers de chiffrage par exemple, où les clefs de chiffrage doivent être rassemblées et, parfois, une image disque dur doit être réalisée avant l'extinction des postes de travail.

3.2.2.1.3 Procédure forensic « Reconstitution » Les résultats de l'analyse sont exploités et l'on procède à une brève reconstitution de l'incident. L'objectif principal de cette phase est d'établir si les preuves suffisantes ont été réunies afin de mettre un terme à ActDF. Cependant, ActDF peut être répété si la quantité de preuves requise n'est pas encore atteinte.

La procédure de reconstitution des faits est définie par l'algorithme ci-dessous :

Algorithme 13 Reconstitution (ActDF)

ENTRÉES: scène de crime
SORTIES: documentation sur les faits
1: initialiser ;
2: **répéter**
3: établir une relation entre les évènements en se référant à la phase d'acquisition ;
4: établir une documentation ;
5: **jusqu'à** obtenir un ensemble consistant de preuves ;

3.2.2.1.4 Procedure forensic « Incident closure » Lorsque les preuves suffisantes ont été rassemblées, une documentation doit être établie pour permettre une future investigation réactive. En effet, aussitôt que la phase ActDF s'achève, le composant ReaDF doit être en mesure de continuer son analyse et reconstruire l'incident en se basant sur toute les autres preuves (numériques et physiques éventuelles) qui serviront à l'élaboration des conclusions de l'enquête.

La procédure de Clôture est définie par l'algorithme ci-dessous :

Algorithme 14 Incident closure (ActDF)

ENTRÉES: scène de crime
SORTIES: documentation
1: initialiser ;
2: reconstituer les faits ;
3: rédiger la documentation de l'enquête ;
4: activer la composante ReaDF ;

La composante ActDF se résume en :

Algorithme 15 Active Digital Forensic

ENTRÉES: scène de crime
SORTIES: rapport d'enquête
1: initialiser ;
2: Evidence acquisition ;
3: Analysis ;
4: Reconstitution ;
5: Incident closure ;

3.2.2.2 Représentations

Dans cette phase, les acteurs sont : « First responder », « Other » ou « Investigators ». L'entité « First responder » ou l'entité « Other » saisi les « Investigators » qui dès leur arrivée sur les lieux posent les actes d'enquêtes. L'intervention peut se faire à distance. Tout dépend de la politique de collaboration mise en place entre l'organisation victime et les autorités judiciaires. Etant donné qu'il s'agit d'une attaque qui est en train de se commettre, les échanges entre les protagonistes doivent être rapides et brèves. La figure 3.4 détaille les actions à effectuer. Le diagramme d'activités représenté dans la figure 3.5 ci-

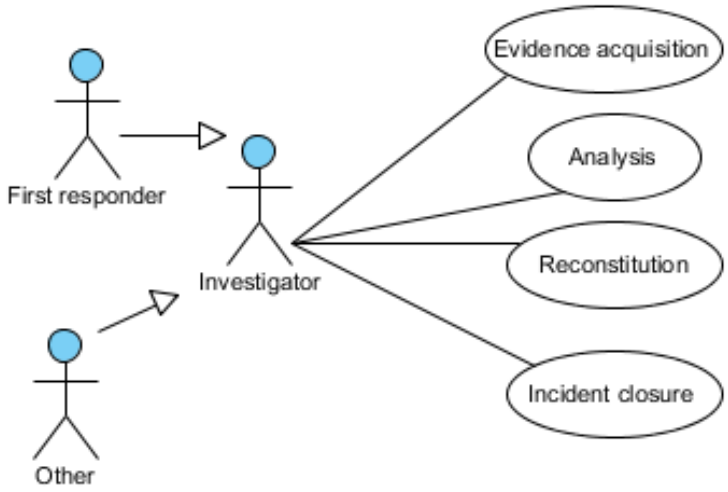


Figure 3.4 – Diagramme de cas d'utilisation (ActDF)

dessous illustre la description du diagramme de cas d'utilisation.

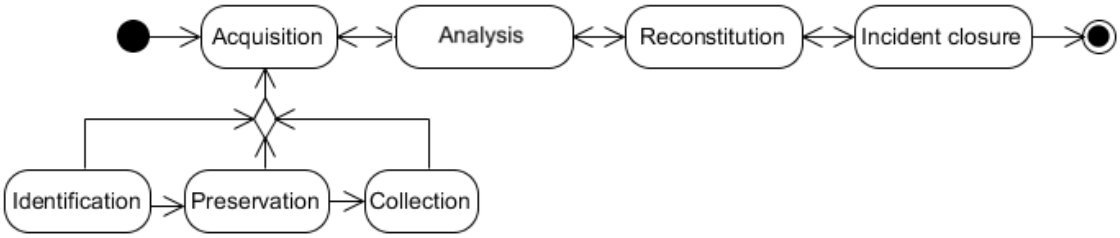


Figure 3.5 – Diagramme d'activités (ActDF)

3.2.3 Composante forensic Proactive (ProDF)

La phase de l'investigation proactive est une réorganisation, une description de processus, de procédures et de technologies dans le but de générer, rassembler, préserver et gérer des preuves numériques exploitables en vue d'assurer une investigation numérique efficiente. Cette prédisposition à l'investigation ne doit aucunement influer sur la qualité de service. Elle doit en outre assurer que les preuves numériques seront collectées en respectant une chaîne de traçabilité judiciaire [85, 89]. Bien que les conditions d'admissibilité de preuves devant les juridictions dépendent des Etats, le succès d'une investigation dépend de la crédibilité des preuves numériques recueillies.

La phase d'investigations proactives permet de manière proactive d'identifier, collecter, regrouper les évènements, préserver et analyser les preuves associées à un incident potentiel. En outre, une documentation automatisée est produite pour une investigation future conduite par les composantes ActDF et ReaDF.

La composante d'investigation proactive diffère des systèmes ordinaires de détection des intrusions (IDS) en ce qu'elle assure l'intégrité des preuves recueillies et les préserve en respectant une chaîne de traçabilité judiciaire. En plus, l'analyse de la preuve sera faite afin d'être admise par une juridiction pour permettre la poursuite des suspects.

3.2.3.1 Procédures forensics

Les procédures qui constituent la composante proactive sont définies comme suit :

3.2.3.1.1 Procédure forensic « Identification » Le processus d'identification vise à identifier les informations suivant leur volatilité. Ensuite, tous les scénarios et comportement anormaux sont catalogués.

3.2.3.1.2 Procédure forensic « Collection » Il s'agit dans cette procédure d'automatiser la collecte des données associées aux comportements et scénarios anormaux identifiés.

Algorithme 16 Identification (ProDF)

ENTRÉES: Un ensemble de trafics ou de fichiers logs
SORTIES: Toutes les informations concernant un incident

initialiser ;
répéter
  si l'organisation dispose d'un système de détection des intrusions **alors**
    sélectionner une alerte ;
    extraire toutes les informations pertinentes qui caractérisent cette alerte ;
  sinon
    détecter les comportements et scénarios anormaux ;
    extraire toutes les informations pertinentes qui caractérisent les comportements et scénarios anormaux ;
  finsi
  créer ou mettre à jour le fichier temporaire d'identification ;
jusqu'à la fin journalier des alertes ;

Algorithme 17 Collection (ProDF)

ENTRÉES: fichier temporaire d'identification
SORTIES: fichier temporaire de collecte

initialiser ;
pour chaque alerte du fichier temporaire d'identification **faire**
  extraire les informations pertinentes ;
  créer un identifiant ;
  regrouper les informations par identifiant ;
  créer ou mettre à jour le fichier temporaire de collecte ;
fin pour

3.2.3.1.3 Procédure forensic « Preservation » Les preuves collectées sont protégées par des techniques de hachages.

Algorithme 18 Preservation (ProDF)

ENTRÉES: fichier temporaire de collecte
SORTIES: fichier temporaire de collecte
1: initialiser ;
2: protéger le fichier temporaire de collecte ;
3: sauvegarder le fichier temporaire de collecte ;

3.2.3.1.4 Procédure forensic « Analyse » Il s'agit d'une analyse directe et automatisée des preuves recueillies en utilisant des techniques de data mining pour soutenir et construire les hypothèses initiales des incidents.

Algorithme 19 Analysis (ProDF)

ENTRÉES: fichier temporaire de collecte, fichier temporaire d'identification
SORTIES: fichier temporaire d'analyse
1: initialiser ;
2: identifier la nature de l'attaque ;
3: catégoriser l'attaque ;
4: indiquer les composants du système qui ont été affectés ;

Algorithme 20 Documentation (ProDF)

ENTRÉES: fichier temporaire d'analyse
SORTIES: Fichier de preuves numérique
1: initialiser ;
2: classer le fichier d'analyse ;
3: créer ou mettre à jour le fichier de preuves numériques ;

3.2.3.1.5 Procédure forensic « Documentation »

3.2.3.2 Représentations

Le seul acteur dans cette phase est le « First responder ». En effet, il s'agit d'un outil logiciel ou matériel capable d'alerter les autorités judiciaires en cas d'attaques en fonction des directives de l'organisation au sein duquel il se déploie. C'est un outil précieux pour les enquêteurs puisqu'il garde la trace des attaques sur le système d'information pendant toute sa durée de fonctionnement. Il exécute les procédures indiquées dans la figure 3.6.

Le diagramme d'activités représenté dans la figure 3.7 indique les corrélations entre les différentes procédures forensics.

Algorithme 21 Proactive Digital Forensic

ENTRÉES:
SORTIES:
1: initialiser ;
2: exécuter identification ; ;
3: exécuter collection ;
4: exécuter preservation ;
5: exécuter analysis ;
6: exécuter documentation ;
7: activer ActDF ou ReaDF ;

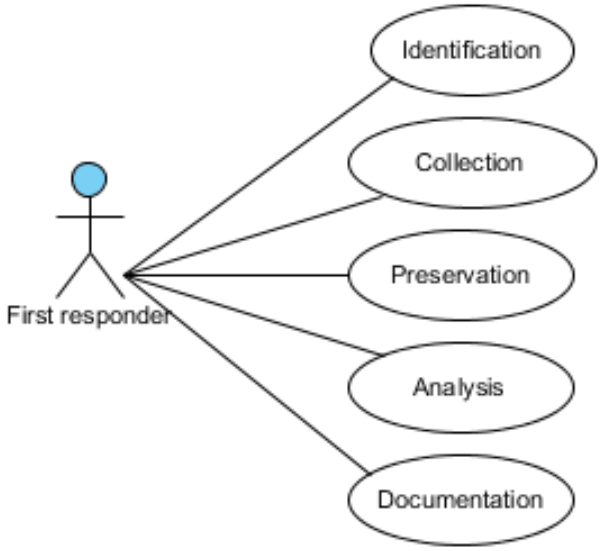


Figure 3.6 – Diagramme de cas d'utilisation (ProDF)

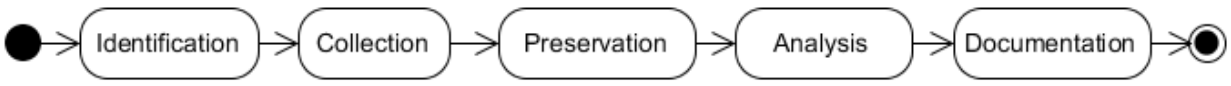


Figure 3.7 – Diagramme d'activités (ProDF)

La figure 3.8 est une vue schématique de l'approche multidimensionnelle de l'investigation numérique ci-dessus décrite.

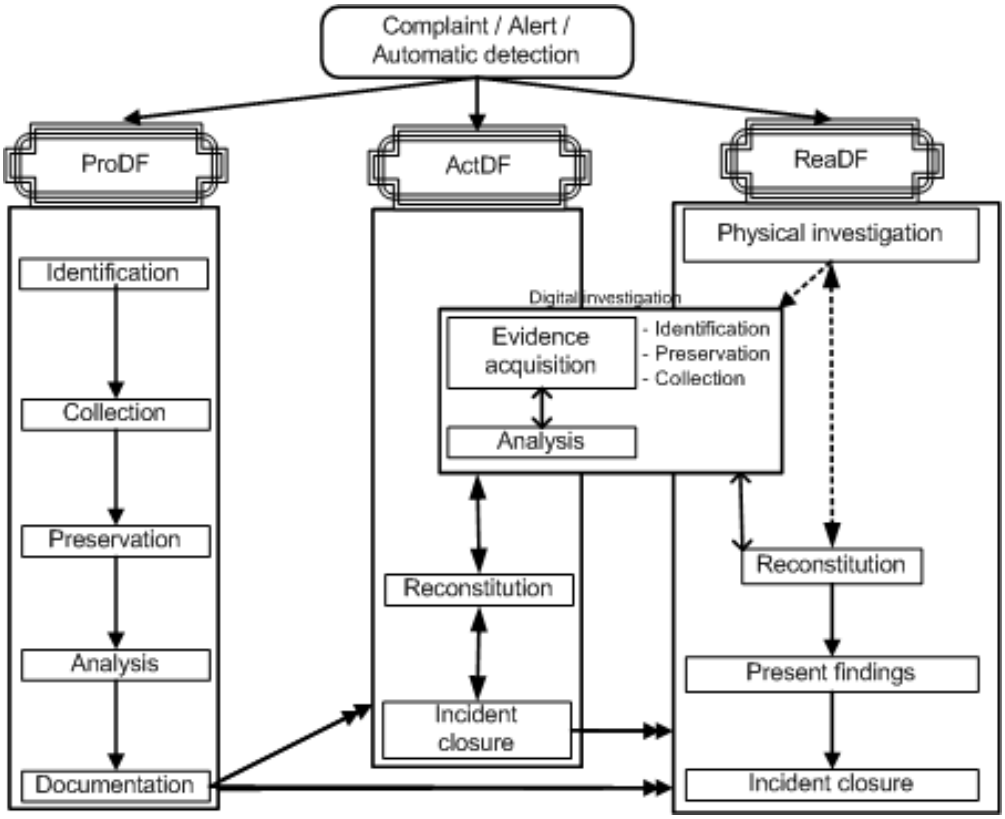


Figure 3.8 – Le modèle multidimensionnel

3.3 Comparaison et discussion

3.3.1 Interactions entre les composants

Les trois composantes sont interdépendantes (figure 3.9). Une plainte au sujet d'un incident, d'une alerte ou une détection individuelle/automatique d'un incident est le point de départ d'une investigation. Un pare-feu est installé par l'organisation pour acquérir des données dès que certaines alertes inhérentes à des comportements suspects sont détectées dans la phase de l'investigation proactive. Si l'attaque est confirmée, la composante ActDF est immédiatement activée. Lorsqu'une quantité suffisante de preuves a été engrangée, un terme est mis à la composante ActDF et la composante ReaDF est déclenchée.

Les investigations Proactives, Actives et Réactives sont globalement soumises à la plainte des victimes et aux autorisations des autorités judiciaires. Une investigation Proactive devient Active lorsque la victime fait appel aux autorités judiciaires pendant

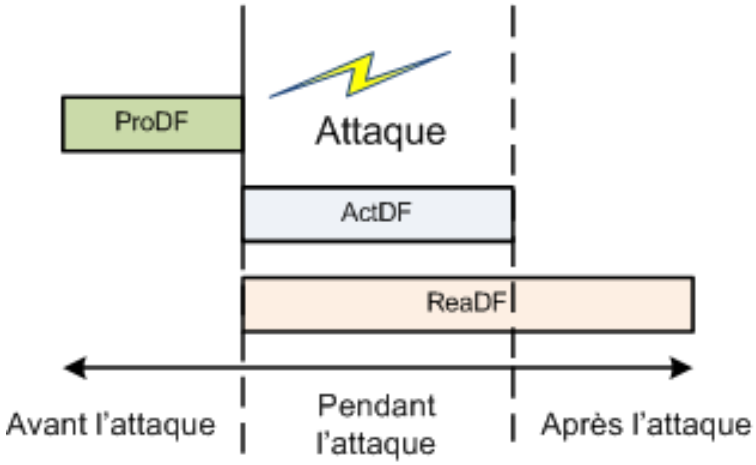


Figure 3.9 – Exécution du modèle dans l'espace et dans le temps

que l'attaque se commet. La figure 3.10 montre l'exécution des phases ProDF, ActDF et ReaDF dans l'espace et dans le temps.

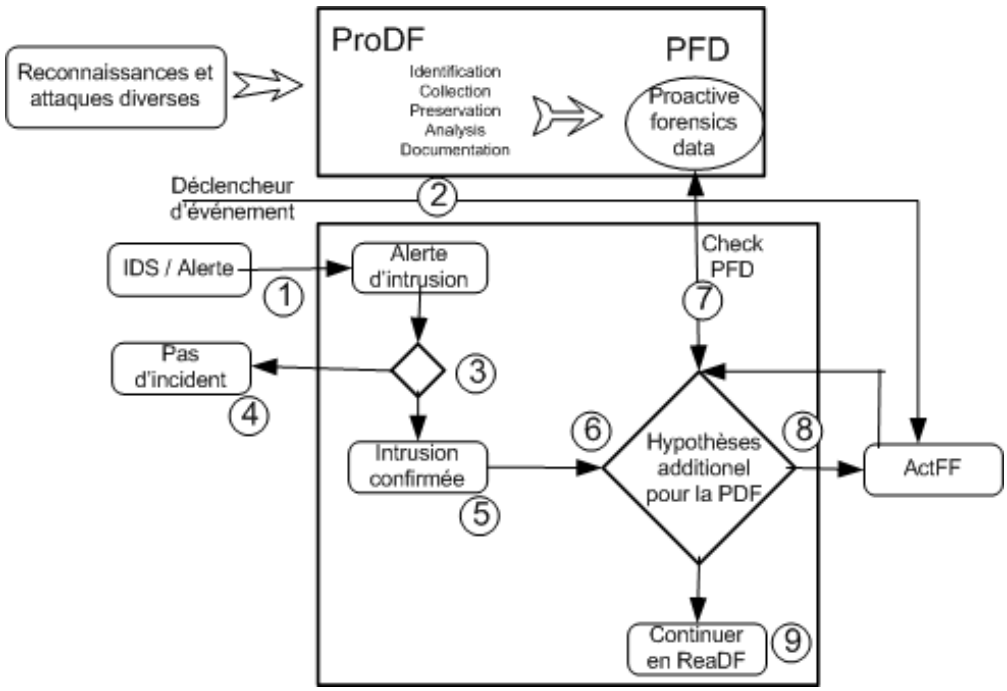


Figure 3.10 – Interaction entre les composants ProDF, ActDF et ReaDF

Dans la figure 3.10, l'alerte (1) issue de la détection d'une intrusion constitue le point de départ de l'investigation. L'organisation victime peut en coaction avec les autorités judiciaires actionner un déclencheur d'événements (2) qui provoquera la collecte des données en flagrance. La troisième étape (3) est une évaluation de la situation. Cette évaluation peut permettre l'interruption (4) ou la poursuite (5) de l'investigation. Lorsque l'incident est confirmé, il est important de déterminer s'il existe assez de preuves permet-

tant d'élucider l'attaque (6). Pour le faire, l'investigateur consulte non seulement les informations obtenues en flagrance mais également ceux obtenues dans le PFD (Proactive Forensic Data) (7). Il s'agit d'une masse de données issues de l'exécution de la composante $ProDF$. S'il n'y a pas de preuves suffisantes pour élucider l'attaque ou encore s'il est nécessaire d'obtenir de nouvelles preuves en flagrance, la composante ActDF (8) reste activée. Dans le cas contraire, c'est la composante ReaDF (9) qui prend le relais.

3.3.2 Discussion

CP Grobler et Al. en 2010 [6] ont fait l'état de l'art sur les modèles d'investigations numériques existants. De cette étude, ces auteurs ont proposé un modèle forensic. Cependant, leur modèle ne couvre pas une investigation numérique dans son entièreté. En effet, ce modèle ne prend pas en compte la possibilité d'une investigation numérique proactive qui intègre les processus d'identification, de collecte, de préservation et d'analyse de données pertinentes permettant d'anticiper sur une attaque. Dans la même perspective, Soltan Alharbi et Al en 2011 [4] ont proposé une approche d'investigation numérique active et réactive. Mais dans cette approche, la phase d'investigation active est confondue à une investigation conduite sur un système en fonctionnement. Ceci conduit à une incomplétude des actions à réaliser afin de mener efficacement une investigation numérique de grande envergure. Compte tenu des différentes activités qui doivent être effectuées dans le cadre d'une investigation numérique légale, le tableau 3.1 ci-dessous fait une comparaison entre les modèles existants et celui qui a été réalisé. Dans ce tableau, le symbole ✓ précise le fait que le modèle intègre le processus associée.

Le principal problème dans le domaine de l'investigation numérique est la définition d'un procédé complet d'investigation en cybercriminalité. Le modèle proposé qui peut être validé par son application dans diverses situations de cybercriminalité, donne une base pour atteindre cet objectif.

La composante proactive développée dans le modèle d'investigation proposé permet d'investiguer même pour des cas d'attaques dont les méthodes anti-forensics ont été utilisées. Elle assure également la collecte de preuves liées à une attaque pendant le fonctionnement courant d'un système d'information. Par ailleurs, l'automatisation de ce processus forensic assure une intervention humaine minimale garantissant ainsi l'intégrité des informations collectées.

Modèle forensic ; Date ; référence	Proactive investigation						Active investigation					Reactive investigation								
	Alert	Identification	Collection	Preservation	Analysis	Documentation	Complaint / Alert / Individual detection	Evidence acquisition	Analysis	Reconstitution	Incident closure	Complaint / Alert / Individual detection	Physical Investigation	Evidence acquisition	Analysis	Reconstitution	Present findings	Dissemination of results	Incident closure	Final report
A road map for digital forensics research-report from the first Digital Forensics Research Workshop (DFRWS) ; 2001 ; [38]														✓	✓	✓	✓	✓	✓	✓
An Examination of Digital Forensic Models ; 2002 ; [39]														✓	✓	✓	✓	✓	✓	✓
Getting physical with the digital investigation process ; 2003 ; [40]														✓	✓	✓	✓	✓	✓	✓
A comprehensive approach to digital incident investigation ; 2003 ; [41]															✓	✓	✓	✓	✓	✓
The Enhanced Digital Investigation Process ; 2004 ; [42]														✓	✓	✓	✓	✓	✓	✓
The Extended Model of Cybercrime Investigations ; 2004 ; [43]														✓	✓	✓	✓	✓	✓	✓
An Event-Based Digital Forensic Investigation Framework ; 2004 ; [44]														✓	✓	✓	✓	✓	✓	✓
The digital detective : An introduction to digital forensics ; 2004 ; [45]														✓	✓	✓	✓	✓	✓	✓
A hierarchical, objectives-based framework for the digital investigations process ; 2005 ; [46]														✓	✓	✓	✓	✓	✓	✓
Framework for a digital forensic investigation ; 2006 ; [47]														✓	✓	✓	✓	✓	✓	✓
Guide to Integrating Forensic Techniques into Incident Response ; 2006 ; [48]														✓	✓	✓	✓	✓	✓	✓

Tableau 3.1 – Comparaison des modèles (1)

	Proactive investigation						Active investigation					Reactive investigation							
The Computer Forensics Field Triage Process Model ; 2006 ; [49]														✓	✓	✓	✓	✓	✓	
FORZA - Digital Forensics Investigation Framework Incorporation Legal Issues ; 2006 ; [50]														✓	✓	✓	✓	✓	✓	✓
A Common Process Model for Incident Response and Computer Forensics ; 2008 ; [51]														✓	✓	✓	✓	✓	✓	✓
Two-Dimensional evidence Reliability Amplification Process Model ; 2008 ; [52]														✓	✓	✓	✓	✓	✓	✓
New Digital Forensics Investigation Procedure Model ; 2008 ; [53]														✓	✓	✓	✓	✓	✓	✓
An Extended Model for E-Discovery Operations ; 2009 ; [54]														✓	✓	✓	✓	✓	✓	✓
A Multi-component View of Digital Forensics ; 2010 ; [6]							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
The Proactive and Reactive Digital Forensics Investigation Process : A Systematic Literature Review ; 2011 ; [4]		✓	✓	✓	✓	✓								✓	✓					✓
Multi-perpective cybercrime investigation process modeling ; 2012 ; [91]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Tableau 3.2 – Comparaison des modèles (2)

3.4 Synthèse

Afin que les informations ainsi obtenues qui justifient la présence d'une attaque soient utilisées comme preuve dans le cadre d'une investigation numérique véritable, nous avons mis sur pied un cadre formel d'investigations capable de donner du crédit aux informations issues de l'outil qui sera implémenté sur la base du "resource behavior technique" en vue de la détection des attaques sur les SI. Ce faisant, nous avons réalisé un modèle d'investigation multidimensionnel capable de se déployer de manière proactive, active et réactive. Ce modèle qui a trois composantes $ProDF$, $ActDF$ et $ReaDF$ permet non seulement de palier le problème de la collecte des preuves lorsque les techniques anti-forensics sont utilisées, mais aussi, de couvrir une investigation dans l'espace et dans le temps. Les résultats de la mise en œuvre de ce modèle d'investigation au cours d'une intrusion dans un système d'information sont présentés dans le chapitre suivant.

CHAPITRE 4

ETUDE DE CAS : COLLECTE DE PREUVES NUMÉRIQUES PAR DÉTECTION DES INTRUSIONS

4.1 Introduction

Les techniques de détection des intrusions sont apparues pour inspecter les activités entrantes ou sortantes d'un réseau et identifier les cas suspects qui indiqueraient une attaque susceptible de compromettre un système d'information. Malheureusement, bien que les systèmes de détection des intrusions soient une source fournie d'informations sur les attaques, les travaux jusqu'à lors effectués n'ont pas établi de manière concrète que ces informations pouvaient être utilisées comme preuve numérique devant une juridiction. En effet, la difficulté principale est celle de pouvoir offrir aux IDS la capacité de produire des informations forensics sans changer leur mission première qui est la détection des intrusions.

Dans ce chapitre, nous investigons sur un cas de cybercriminalité en l'occurence, une attaque par déni de service distribué de type *XMAS tree attack*. Pour cela, nous utilisons l'outil de detection des intrusions $Snort$ que nous raffinons pour lui permettre de collecter des informations qui peuvent être utilisées comme preuves numériques dans une procédure judiciaire.

Dans la deuxième partie de ce chapitre, nous présentons les extrants des IDS et donnons les conditionnalités de transformation de ces extrants en preuve numérique en prenant en compte le modèle multidimensionnel étudié au chapitre précédent. Dans la troisième partie, nous déployons un NIDS dans un réseau expérimental et nous apportons une nouvelle implémentation au fichier $snort.conf$ de l'IDS $Snort$. Ce faisant, nous présentons également les résultats des tests effectués. Nous achevons ce chapitre par une synthèse à la quatrième partie.

4.2 Extrants des IDS et preuves numériques

Les objectifs attendus d'un IDS sont les suivants [64] :

— la capacité de réagir promptement pour prévenir ou réduire les dommages substantiels à partir d'une intervention automatique ou manuelle ;
— la capacité à identifier un attaquant ou une activité malveillante qui pourrait causer de sérieux domages à un système d'information ;
— la capacité à découvrir de nouvelles méthodes d'attaque ou, de manière préventive, fournir une protection supplémentaire du système au-delà de celles déjà disponibles.

Pendant l'expérimentation, il a été donné de constater que $Snort$ sauvegarde beaucoup de messages dans le répertoire $/var/log/snort$ sous Linux et \Snort\log sous Windows. Il s'agit d'un répertoire non sécurisé qui peut facilement être compromis. Ces messages contiennent des informations pertinentes sur un incident chaque fois qu'il se produit, en fonction de certaines règles spécifiques indiquées dans le code source de $Snort$. Ces informations sont : Temps/Date, Adresse IP source, Adresse IP de Destination, Durée de vie (TTL), Type de service (TOS), Longueur de l'en-tête du paquet IP, Longueur totale du paquet IP, Type du champ ICMP, Valeur du code ICMP, ID du paquet IP, Numéro de séquence, Type du paquet ICMP.

Malheureusement, les données appréhendées par $Snort$ ne sont pas sauvegardées sous forme de texte clair. Il n'est pas possible de les lire avec un éditeur de texte. De ce fait, l'utilisation de "$TCPDUMP$" a permi d'avoir une interprétation claire des fichiers produits.

Du point de vue de la preuve, ce qui est recherché c'est ce que l'on peut démontrer aux tiers longtemps après l'attaque en question. L'IDS l'offre à travers des enregistrements de divers types. Il s'agit des journaux systèmes, d'audits, d'application et de gestion de réseau. La capture du trafic réseau et les écoutes sur les ports sont d'autres sources potentielles de preuve [97, 101]. Cependant, les données qui en découlent peuvent être transformées sous une forme qui est facile à analyser et à comprendre. Autrement dit, pour être recevable en justice, le rassemblement de preuves potentielles devra suivre une chaîne de traçabilité. Ainsi, les produits de l'IDS seront assez efficaces pour persuader un tiers.

Pour ce faire, les enregistrements produits par un IDS qui visent à fournir des informations pertinentes pour les besoins d'une enquête numérique doivent respecter les spécificités suivantes [97] :

— les enregistrements ne devront pas avoir été compromis avant ou pendant leur collecte comme preuves potentielles, ni pendant l'analyse qui suit leur collecte ;
— dans le cas d'un réseau en temps réel, le dispositif qui héberge l'outil de surveillance devra être positionné tel qu'il peut appréhender tout trafic pertinent, même si certains paquets utilisent d'autres routes ;
— en cas de surveillance en temps réel, l'outil de surveillance doit être capable de supporter le trafic dont il est supposé monitorer ;
— les enregistrements devront suffisamment distinguer les accès légitimes des accès non désirés ; des enregistrements doivent exister sur une période suffisante pour

que la comparaison des activités normales et anormales puisse se faire ;
— les enregistrements doivent aider à identifier les auteurs de manière utile, être complètes pour la période concernée et riches en détails ;
— les enregistrements doivent rassembler des informations pertinentes.

Par conséquent, la contribution que l'IDS peut apporter en cas de procès est de prouver qu'un système d'information a été violé et qu'on y a accédé. Ensuite, l'enquête numérique produira suffisamment de preuves et permettra d'enquêter en vue d'identifier l'auteur des faits récriminés.

L'application des conditions énumérées ci-dessus permet de définir un modèle de détection (figure 4.1). Ce modèle repose essentiellement sur le modèle de base de détection de l'intrusion mais, il a la particularité d'être capable de produire des informations susceptibles de servir de preuve numérique.

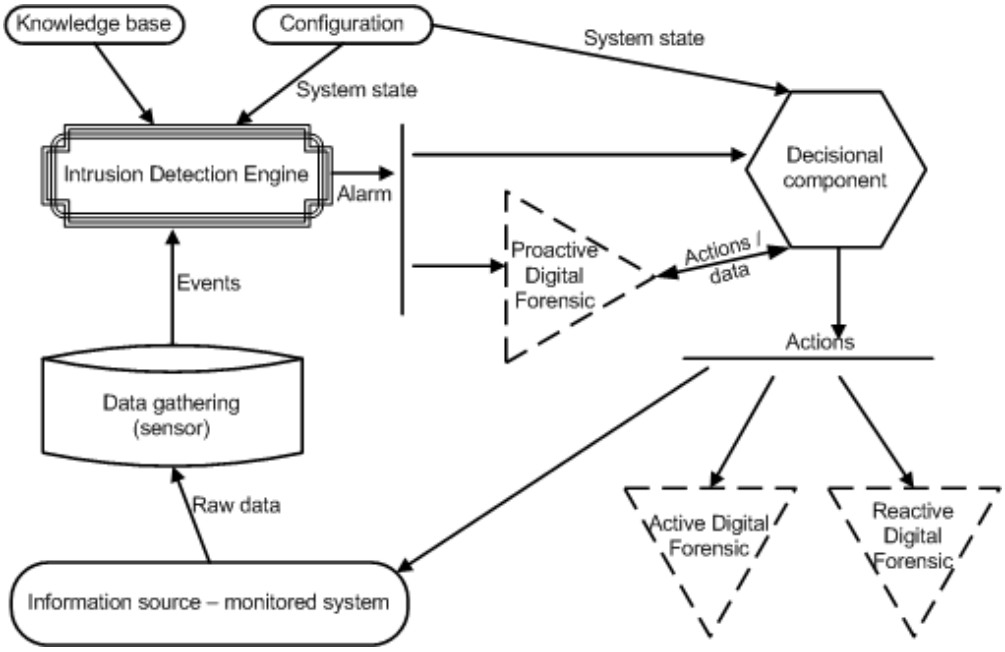


Figure 4.1 – Architecture de base d'un IDS pour les besoins d'enquête numérique

Le modèle proposé dans la figure 4.1 ci-dessus est constitué de 9 composantes qui sont définies comme suit :
— Information source : il s'agit du système monitoré en vue de la détection des intrusions
— Data gathering : c'est un dispositif de collecte en charge de la collecte de données brutes dans le système monitoré ;
— Intrusion detection engine : ce moteur traite les données collectées par les capteurs pour identifier les activités intrusives ;
— Knowledge base : elle contient les informations collectées par les capteurs, mais sous un format prétraité. Il s'agit entre autres de bases de connaissance des attaques et de leurs signatures, de données filtrées et de profils de données. Cette information est généralement fournie par les CERT ou par les experts en réseau et en sécurité ;

— Configuration : elle donne des informations sur l'état actuel du système de détection d'intrusion ;
— Proactive Digital Forensic (ProDF) : elle permet d'assurer une exécution efficiente des enquêtes numériques avec une perturbation minimale des activités. Cette composante permet de s'assurer que des preuves et processus acceptables sont mis en place et disponibles quand il le faut pour une enquête [4, 7, 101]. Chaque fois qu'une alerte est déclenchée, cette composante commence la collecte de toutes les informations liées à l'activité intrusive. Elle sauvegarde activement l'intégrité des informations collectées et les préserve de manière acceptable pour une enquête judiciaire ;
— Decisional component : elle initie une action lorsqu'une intrusion est détectée. Ces réponses peuvent être soit automatiques, soit impliquer une interaction humaine ;
— Reactive Digital Forensic (ReaDF) : elle cible l'enquête numérique traditionnelle qui aura lieu après qu'un incident ait été détecté et confirmé. Ceci implique l'identification, la préservation, la collecte, l'analyse et la production du rapport final. Ce module est activé par la composante décisionnelle pour la poursuite de l'investigation ;
— Active Digital Forensic (ActDF) : elle permet d'identifier, de préserver, de collecter et d'analyser les preuves numériques. Quand l'alerte est déclenchée et que l'incident est confirmé, le dispositif décisionnel active la composante ActDF pendant toute la durée de l'attaque.

4.3 Expérimentations et résultats

4.3.1 Mise en œuvre et déploiement du NIDS dans le réseau

4.3.1.1 Mise en œuvre du NIDS

Dans la phase expérimentale, il a été mis en place une solution NIDS basée sur le logiciel $Snort$ et la console BASE (Basic Analysis and Security Engine) pour analyser et superviser les alertes remontées par un NIDS. Le choix de $Snort$ en particulier se justifie par le fait que cet outil peut être adaptée pour une petite ou grande entreprise (TPE, PME, etc.) mais aussi, la disponibilité de nombreuses règles dans la communauté de $Snort$ et des projets comme $Oinkmaster$ qui gèrent la mise à jour des signatures et fournissent un nombre important de règles optimisées et efficaces pour la détection des intrusions. Par ailleurs, $Snort$ dispose d'un ensemble de plugins qui aident à étendre ses fonctionnalités, notamment $Snortsam$ qui ajoute des fonctionnalités permettant à $Snort$ de jouer le rôle d'un IPS.

4.3.1.2 Déploiement du NIDS dans le réseau

Le laboratoire informatique du Service Central des Recherches Judiciaires de la Gendarmerie Nationale, crée dans le cadre de la lutte contre la cybercriminalité, a servi de

cadre pour diverses expérimentations.

Le dispositif expérimental est constitué d'un routeur et de six postes de travail. Le routeur est connecté à internet et les postes de travail forment un réseau local. En vue de ne détecter que les activités intrusives au réseau local, le système de détection des intrusions a été placé entre le routeur et le réseau local (figure 4.2). Le poste de travail de la victime a les caractéristiques suivantes : Microsoft Windows XP Home with Service Pack 3 ; Intel CPU T2250, 1.6 GHz ; 2 GB de RAM ; adresse IP 192.168.1.101. Les caractéristiques du poste de travail de l'attaquant sont : Microsoft Windows XP Professional with Service Pack 3 ; Intel Core 2 Duo CPU T5200 @ 1.6 GHz ; 2.00 GB de RAM ; adresse IP 41.204.92.97/192.168.1.108. Le routeur d'expérimentation est de marque Cisco Linksys 802.11g/2.4 GHz Wireless-G Broadband Router ; Model No : WRT54G2 V1.

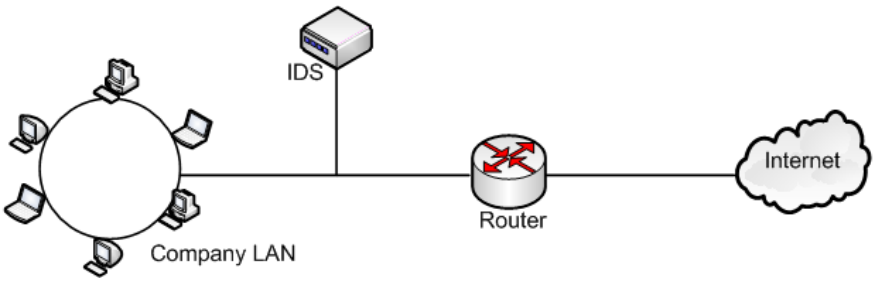


Figure 4.2 – Architecture du réseau expérimental

Ce choix est apparu judicieux dès lors que les attaques externes au réseau local (Troyen, Virus, etc.) constituaient la principale préoccupation.

4.3.1.3 Mise à jour de l'architecture de SNORT

Snort est connu pour être une application puissante. Ce logiciel est libre et peut fonctionner avec les environnements Linux ou Windows. La compréhension du fonctionnement des différentes composantes internes de $Snort$ a permis de l'adapter au réseau d'expérimentation et d'éviter certaines faiblesses connues de $Snort$ (techniques d'évasion de $Snort$) [99]. $Snort$ peut être divisé en cinq grandes composantes dont chacune est indispensable à la détection de l'intrusion (figure 4.3).

— Packet sniffer : c'est un renifleur de paquets qui prélève des paquets sur différents types d'interfaces du réseau et les transmet au préprocesseur ;
— Préprocesseurs : c'est le composant utilisé par $Snort$ pour prétraiter les paquets reçus du Sniffer. Il réorganise les paquets avant de les acheminer au moteur de détection ;
— Detection engine : ce moteur a pour but de détecter si une activité d'intrusion existe dans un paquet. Pour ce faire, il emploie de règles spécifiques. Lorsqu'un paquet est retenu par une règle, des mesures appropriées sont prises, sinon le paquet est abandonné. Ces mesures appropriées peuvent être la journalisation le paquet incriminé ou la génération des alertes ;

— Alarm/log : En fonction des découvertes du moteur de détection à l'intérieur d'un paquet, le paquet peut être utilisé pour journaliser l'activité ou générer une alerte.
— Binary log file/Database : les activités suspectes sont sauvegardées dans un fichier binaire ou dans une base de données.

En vue de rassembler des preuves numériques, de nouveaux algorithmes ont été implémentés dans le fichier $snort.conf$. Par conséquent, ce fichier a été mis en œuvre de telle sorte qu'aussitôt que l'alerte est lancée, les paquets de données incriminés sont simultanément préservés dans la base de données binaires des fichiers d'enregistrement de $Snort$, et convertis pour servir comme input à la composante $ProDF$. La figure 4.3 donne une représentation graphique simplifiée du flux de données de la nouvelle architecture de $Snort$ obtenue.

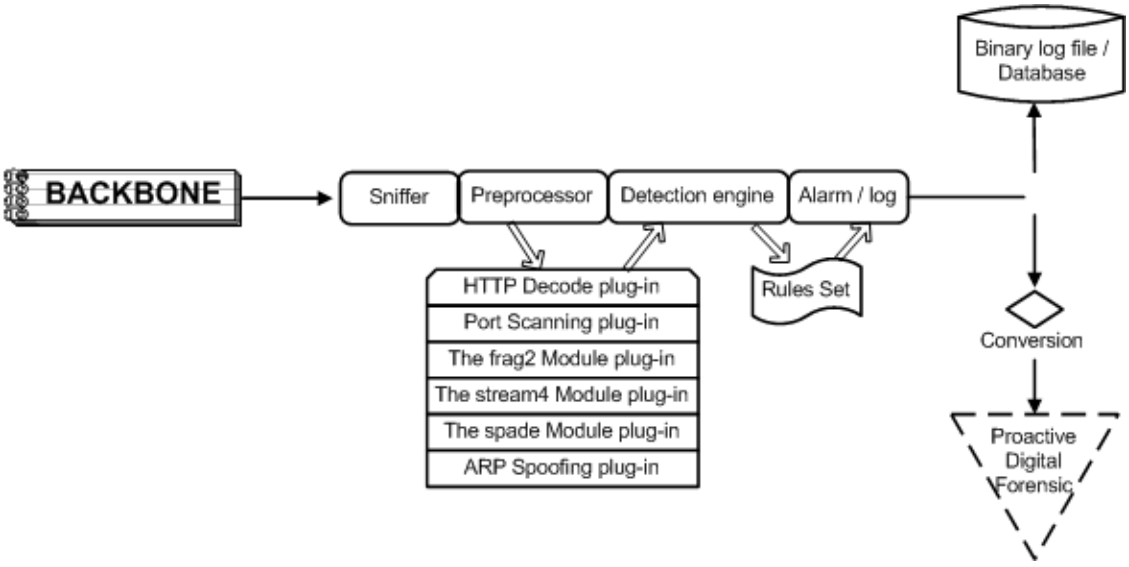


Figure 4.3 – Architecture de $Snort$ pour les besoins d'enquête numérique

4.3.2 Nouvelle implémentation du fichier SNORT.CONF

$Snort$ est un instrument léger mais puissant pour détecter les trafics malicieux dans un réseau donné. Avec un langage de définition de règles flexible et robuste, $Snort$ est capable de détecter presque toutes les menaces qui traversent le réseau. Cependant, les rapports qui en résultent sont peu exploitables. Il enregistre des dizaines ou centaines de milliers d'évènements suspects chaque jour dans un réseau ayant un gros volume de trafic. Dans le but de raffiner $Snort$ en lui ajoutant de nouvelles options, le fichier $snort.conf$ a été modifié pour qu'il puisse produire en sortie le fichier $evidence.ids$ à partir des algorithmes ci-dessous qui constituent une instance d'exécution de la composante $ProDF$:

Algorithme 22 Identification

ENTRÉES: Un ensemble de trafic émis par l'engin de détection de SNORT
SORTIES: fichier temporaire d'identification
1: initialiser ;
2: **répéter**
3: sélectionner une alerte ;
4: extraire toutes les informations pertinentes qui caractérisent cette alerte ;
5: créer ou mettre à jour le fichier temporaire d'identification ;
6: **jusqu'à** la fin journalière des alertes

Algorithme 23 Collection

ENTRÉES: fichier temporaire d'identification
SORTIES: fichier temporaire de collecte
1: initialiser ;
2: **répéter**
3: sélectionner une alerte dans le fichier temporaire d'identification ;
4: extraire l'adresse IP source ;
5: extraire les informations associées à cette adresse IP source ;
6: réaliser un enregistrement référencé par l'adresse IP source ;
7: créer ou mettre à jour le fichier temporaire de collecte ;
8: **jusqu'à** la fin du fichier temporaire d'identification

Algorithme 24 Préservation

ENTRÉES: fichier temporaire de collecte
SORTIES: fichier temporaire de collecte
1: initialiser ;
2: protéger le fichier temporaire de collecte ;
3: sauvegarder le fichier temporaire de collecte ;

Algorithme 25 Analyse

ENTRÉES: fichier temporaire de collecte, fichier temporaire d'identification
SORTIES: fichier temporaire d'analyse

1: initialiser ;
2: identifier la règle qui a permis de déclencher l'alerte ;
3: catégoriser l'attaque ;
4: indiquer la nature de l'attaque ;
5: indiquer les adresses IP source ;
6: indiquer les adresses IP de destination ;
7: indiquer les ports de connexion utilisés ;
8: indiquer l'horodatage ;
9: indiquer les protocoles utilisés ;

Algorithme 26 Documentation

ENTRÉES: fichier temporaire d'analyse
SORTIES: evidence.ids

1: initialiser ;
2: classer le fichier d'analyse par nature d'attaque, IP source, IP destination, protocol, port et horodatage ;
3: créer ou mettre à jour le fichier evidence.ids ;

Algorithme 27 Proactive Digital Forensic (ProDF)

ENTRÉES: evidence.ids
SORTIES: evidence.ids

1: initialiser ;
2: exécuter identification
3: exécuter collection
4: exécuter preservation
5: exécuter analysis
6: exécuter documentation

Le fichier $evidence.ids$ en sortie de la composante forensic $ProDF$ contient les preuves numériques.

4.3.3 Analyse d'un honneypot compromis

Pour compléter l'expérimentation, un honneypot [100] a été déployé dans le réseau pour appâter les pirates en rassemblant les preuves de leurs activités. C'est un système qui est utilisé pour leurrer les attaquants en exposant délibérément des faiblesses connues. Certains services ont été à dessein laissés en fonctionnement sur le Honneypot. Il s'agit entre du Telnet (port 23), du HTTP (port 80) et du FTP (port 21). Le leurre était placé tel que les pirates pouvaient facilement le prendre pour un serveur réel, utilisant une adresse IP très proche de celle du serveur réel. Les attaques enregistrées au terme de cette expérience ont permis de constituer beaucoup de fichiers logs. L'étude de cas

s'est focalisé sur la détection des attaques de type *XMAS tree attack*. Il s'agit d'une attaque Ddos par saturation dont la finalité est l'occupation abusive des ressources du système informatique en l'occurence la mémoire vive (RAM), les processeurs et les routeurs.

Ce type d'attaque consiste à envoyer sur un terminal un grand nombre de paquets de données de type Chrismas tree. Sur un paquet de données Chrismas tree, toutes les options sont activées de manière à ce que n'importe quel protocole puisse être utilisé. Les routeurs prennent par conséquent beaucoup plus de temps pour les traiter par rapport aux autres paquets. Ainsi, les paquets Chrismas tree peuvent préoccuper les routeurs ou les terminaux par leur temps de traitement élevé de telle enseigne qu'ils ne fassent plus d'autres traitements, créant ainsi un déni de service. La réception des paquets Chrismas Tree est inhabituelle et devrait créer la suspicion. Les systèmes de détection des intrusions ainsi que certains pare-feux peuvent détecter ce type d'attaque.

Sur un paquet *XMAS tree*, les drapeaux des champs Fin, Push et Urg sont tous activés. La règle de $Snort$ pour détecter les attaques *XMAS tree* est la suivante :

$alert$ tcp 41.204.92.97 any $->$ 192.168.1.101 any $(flags : SFAR; msg : "Xmastreescan")$

où 41.204.92.97 est l'adresse IP de l'attaquant et 192.168.1.101 est l'adresse IP de la victime. L'attaquant peut déclencher cette attaque avec l'outil $nmap$ à travers la commande :

$nmap - v - sX192.168.1.101$

Le fichier en sortie de $Snort$ traduit avec tcpdump qui a permi d'élucider une attaque de ce type, est présenté dans la figure 4.4 ci-dessous.

```
[cc lang="VHDL"]
snort2.9.0.6: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
[**] [021:02:1] spp_stream5: STEALTH ACTIVITY (nmap XMAS scan) detection [**]
08:15:22.748471 192.168.1.108:33434 -> 192.168.1.101:369
TCP TTL:42 TOS:0x0 ID:55954 IpLen:20 DgmLen:40
**U*P**F Seq: 0x0 Ack: 0x0 Win: 0xC00 TcpLen: 20 UrgPtr: 0x0
```

```
[cc lang="VHDL"]
snort2.9.0.6: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
[**] [033:03:2] spp_stream5: STEALTH ACTIVITY (nmap XMAS scan) detection [**]
09:21:35.873683 192.168.1.108:33434 -> 192.168.1.101:369
TCP TTL:42 TOS:0x0 ID:55954 IpLen:20 DgmLen:40
**U*P**F Seq: 0x0 Ack: 0x0 Win: 0xC00 TcpLen: 20 UrgPtr: 0x0
```

Figure 4.4 – Fichiers en sortie de $Snort$

Ce fichier est stocké dans le répertoire $/var/log/snort$ du poste de travail où s'exécute $Snort$. La configuration de $Snort$ pour l'expérimentation est faite de telle enseigne que ce fichier temporaire est passé en entrée de la composante $ProDF$ avant d'être stocké dans le répertoire habituel.

De ce fait, chaque alerte est immédiatement analysée et toutes les informations qui la caractérisent sont extraites pour constituer alors le fichier temporaire d'identification

(figure 4.5). Il s'ensuit instantanément le processus de collecte. A partir du fichier tem-

```
[cc lang="VHDL"]
tcpdump: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
08:15:22.748471 IP (tos 0×0, ttl 37, id 45231, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.108.33434 > 192.168.1.101.369: Flags [FPU], cksum 0x65ec (correct), seq 30145115847, win 2541, urg 0,
length 0
08:15:22.755673 IP (tos 0×0, ttl 121, id 57432, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.101.369 > 192.168.1.108.33434: Flags [R.], cksum 0x87ff (correct), seq 0, ack 30145115848, win 0, length 0
[/cc]
```

```
[cc lang="VHDL"]
09:21:35.873683 IP (tos 0×0, ttl 27, id 50647, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.108.33434 > 192.168.1.101.369: Flags [FPU], cksum 0xdf4a (correct), seq 2649036211, win 2541, urg 0,
length 0
09:21:35.886578 IP (tos 0×0, ttl 121, id 51626, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.101.369 > 192.168.1.108.33434: Flags [R.], cksum 0xec4d (correct), seq 0, ack 2649036212, win 0, length 0
[/cc]
```

Figure 4.5 – Contenu du fichier temporaire d'identification

poraire d'identification, les informations associées à chaque adresse IP sont extraites et classifiées. Afin de préserver l'intégrité des informations ainsi collectées, un algorithme de hachage MD5 est appliqué au fichier temporaire de collecte. L'analyse des données collectées et préservées peut donc s'effectuer en ressortant toutes les informations utiles pour justifier et établir l'ampleur des dégats d'une attaque. Cette analyse conduit directement à l'obtention du fichier crypté $evidence.ids$ contenant les preuves numériques (figure 4.6). Ce fichier de preuve est définitivement sauvegardé dans un répertoire protégé.

```
Attack    No.    Time            Source              Destination    Prot Port Info
Xmas_scan 224512 08:15:22.748471 192.168.1.108.33434>192.168.1.101  TCP  80   369 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 224512 09:21:35.873683 192.168.1.108.33434>192.168.1.101  TCP  80   369 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 224512 18:10:22.234231 41.205.86.200>192.168.1.101        TCP  22   45125 > ssh [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 225121 19:15:01.324561 77.175.26.168>192.168.1.101        TCP  80   45631 > http [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 225530 19:25:15.325467 79.233.134.80>192.168.1.101        TCP  53   48123 > domain [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 226910 20:05:18.348854 82.83.205.73>192.168.1.101         TCP  443  48032 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 227011 20:29:09.426495 83.163.68.56>192.168.1.101         TCP  111  49569 > sunrpc [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
```

Figure 4.6 – Contenu du fichier evidence.ids

Après application des procédures de la composante $ProDF$, l'on constate dans la figure 4.6 que les preuves collectées de manière proactive sont regroupées par type d'attaque, numéro de paquet, temps, adresse IP source, adresse IP destinataire, protocole, port et détails. La description de ces informations est donnée dans la table 4.1 ci-dessous.

No.	Field or Activity	Context/Notes
1	08 :15 :22.748471	Ceci est l'heure du système au moment de la requête : 8 heures 15 minutes 22 secondes.
2	IP	Il s'agit de tous les réglages liés au protocole IP.
3	tos 0x0	Champ de types de services.
4	ttl 37 qui est la durée de vie	Nombre de sauts que le paquet doit effectuer pour atteindre sa destination. Ceci indique le nombre de routeurs par lesquels les paquets doivent passer, pour éviter de laisser le paquet voguer sur le net continuellement.
5	id 45231	En cas de hijacking (comme pour le cas de l'attaque du milieu), l'attaquant doit être capable de pirater l'ID du paquet et présenter en réponse un paquet avec le même ID mais, contenant des données malicieuses.
6	proto TCP	C'est le type de protocole. Il peut être UDP ou ICMP.
7	length 30	La longueur du paquet TCP.
8	41.204.92.97.33434	C'est l'adresse de l'IP source et 33434 est le port utilisé par le pirate.
9	192.168.1.101.369	C'est l'adresse de l'IP de destination (l'adresse de l'IP du Honneypot) et 369 est le port utilisé.
10	Flags [FPU]	C'est l'indicateur FPU du protocole TCP (Fin, Push ou Urg) pendant la réalisation d'un Chrismas scan. Il peut avoir la valeur[S] pour signifier une réponse ACK venant du Honneypot, ou [R] qui signifie RESET et dans ce cas, la connexion est réinitialisée, ou [F] pour mettre fin au transfert, etc.
11	cksum 0x65ec	Ceci est la valeur du contrôle de l'en-tête du paquet de données TCP (pour vérifier l'intégrité du paquet)
12	seq 3014515847	Le numéro de séquence TCP.
13	win 2541	La quantité de paquets à envoyer avant d'attendre une réponse du serveur.
14	urg	Le degré d'urgence.

Tableau 4.1 – Description du fichier journal contenant les preuves numériques

Comme le montre les illustrations, les fichiers logs regorgent d'une quantité substantielle de données qui pourraient être pertinentes dans une affaire criminelle. Les logs pourraient révéler des informations qui relient l'activité observée à l'utilisateur suspecté, y compris l'adresse IP utilisée et le type de système d'exploitation, de navigateur web, et les applications installées sur le poste de travail de l'attaquant.

4.3.4 Evaluation des propriétés de l'IDS modifié

Dans le cadre de cette expérimentation, une version libre de $Snort$ a été utilisée dans un premier temps. Par la suite, l'architecture de $Snort$ a été modifiée en implémentant la composante $ProDF$ à travers les algorithmes présentés plus haut. Le temps mis par l'IDS pour la capture des paquets est présentée dans les tableaux ci-dessous.

Dans le premier cas (tableau 4.2), les informations utiles à l'enquête étaient logées dans le repertoire de sauvegarde par défaut de $Snort$. Dans le deuxième cas (tableau 4.3), les preuves se trouvent dans le fichier $evidence.ids$. Il s'agit d'un dossier sécurisé contenant des données obtenues en respectant une chaine de traçabilité de préservation et de collecte de la preuve numérique. En observant le nombre de paquets reçus, le nombre d'alertes et le nombre de paquets capturés dans les deux cas, la différence est négligeable. Cela prouve que l'IDS $Snort$ bien que sa structure ait été modifiée pour produire en sortie des preuves numériques recevables, n'a pas vu sa performance se dégrader en tant que outil de détection des intrusions [101].

Numéro de test	Nombre de paquets envoyés dans le réseau	Nombre d'alertes	Ratio (Packet/sec)	Nombre de paquets capturés par $Snort$
1	3445263	76	300	3445112
2	9655422	102	500	9655315
3	2712657	52	200	2712645
4	6845795	151	350	6845710
5	8932698	134	400	8932624

Tableau 4.2 – Exécution de $Snort$ sans la composante d'investigation ProDF

Numéro de test	Nombre de paquets envoyés dans le réseau	Nombre d'alertes	Ratio (Packet/sec)	Nombre de paquets capturés par $Snort$
1	3445263	76	300	3445112
2	9655422	102	500	9655311
3	2712657	52	200	2712645
4	6845795	151	350	6845713
5	8932698	134	400	8932621

Tableau 4.3 – Exécution de $Snort$ avec la composante d'investigation ProDF

4.4 Synthèse

En investigant sur un cas de déni de service distribué de type *Xmas tree attack*, il est apparu dans ce chapitre qu'un IDS pouvait être utilisé comme une source d'informations forensics à condition qu'il soit convenablement implémenté. Nous sommes parvenus à ce but en analysant et en mettant à jour le modèle de base des systèmes de détection des intrusions et le modèle multidimensionnel de l'investigation numérique vu dans le chapitre précédent. Ceci nous a conduit à un modèle théorique combiné pour l'investigation et la détection des intrusions. Par la suite, nous avons réalisé des tests sur le modèle combiné pour montrer comment les fichiers logs de $Snort$ sont exploités aux fins d'enquêtes judiciaires. Les tests bien que réalisés sur une architecture NIDS peuvent également s'effectuer sur des architectures HIDS et Hybrides à condition de modifier conséquemment l'implémentation du fichier $snort.conf$ de $Snort$.

Les tests effectués avec l'IDS $Snort$ montrent bien qu'il est possible de doter les IDS de la possibilité de générer des informations forensics sans que leur performance en tant que IDS ne soit dégradée. Cependant, l'enquête judiciaire ne peut pas reposer uniquement sur les IDS sinon, ceux-ci subiraient des changements significatifs qui pourraient inévitablement les dévier de leurs missions premières.

Conclusion Générale

Bilan

L'objetif principal de cette thèse a été de définir une approche par workflow pour l'identification, l'analyse et la protection des indices et preuves constitués lors de l'intrusion dans un système d'information. De ce fait, un cadre formel de l'investigation numérique a été réalisé pour faciliter la collecte selon une méthodologie précise, des preuves nécessaires à présenter devant les autorités compétentes afin de mener des poursuites judiciaires.

Pour y parvenir, nous avons fait un état de l'art sur la cybercriminalité et les investigations numériques avant de définir une approche méthodologique de collecte de preuves numériques. Pour enrichir la banque de données de ces preuves, nous avons conçu un modèle de détection des intrusions basé sur le comportement des ressources. Puis, nous avons developpé une technique d'investigation numérique multidimensionnelle permettant d'élucider les attaques indépendamment d'une plateforme particulière. Nous avons ainsi proposé de nouveaux algorithmes pour améliorer les performances de l'outil de détection des intrusions de référence appelé $Snort$ pour le rendre capable de collecter des preuves numériques. Plusieurs expérimentations nous ont permi de mettre en œuvre les différentes techniques proposées.

La méthode ainsi élaborée apporte une réponse au problème des attaques basées sur les techniques antiforensics car, la composante ProDF intégrée dans un outil de détection des intrusions, catalogue au fil du temps toutes les traces laissées par l'attaquant dans la phase de reconnaissance. Compte tenu des menaces en constante progression et un niveau d'exposition des structures toujours plus important, il n'est plus question de savoir si une organisation va être un jour attaqué, mais plutôt quand ?. Le cadre formel issu de nos travaux apporte des connaissances pour identifier, traquer et poursuivre judiciairement des cybers criminels.

A travers des cas pratiques, nous nous sommes attelés à expérimenter les techniques d'investigations et d'analyses d'attaques au regard des juridictions américaines et françaises : l'usage de plusieurs outils a permi d'établir la faisabilité de déploiement de l'architecture obtenue. La méthodologie élaborée a conduit à la collecte des preuves numériques

qui mettent en évidence un acte de piratage informatique ou un acte visant à détériorer ou détruire des systèmes d'information sensibles au sein d'une organisation. Les enquêteurs et les acteurs du domaine de la cybersécurité en sont les principaux bénéficiaires. De ce fait, ceux-ci seront en mesure de valider et auditer des schémas d'implémentation des données sur les systèmes d'une organisation, et ainsi prévenir des risques d'attaques. Ils pourront également identifier les traces laissées lors de l'intrusion dans un système d'information par une personne malveillante, afin de collecter correctement les preuves nécessaires pour des poursuites judiciaires.

Toutefois, certaines difficultés ont émaillé nos travaux. Sur le plan législatif, en l'absence d'un cadre juridique de la collecte de la preuve numérique pouvant s'associer au code de procédure pénale camerounais, nous nous sommes referés aux législations françaises et américaines en la matière dont les textes de lois sont plus étoffés. Sur le plan technique, contourner les dispositifs d'anonymisation a rarement été aisé. La solution que nous proposons n'est pas une boule de cristal qui, étant donnée une attaque sur un système d'information, indiquera instantannement l'auteur. Elle consiste à rassembler le maximum d'éléments possibles pouvant conduire à l'identification du ou des terminaux qui ont mené l'attaque. L'enquête d'environnement qui est l'une de procédures fonrensics de cette solution dans sa composante réactive, permet de poser les actes classiques d'enquêtes judiciaires qui conduiront aux auteurs en tant qu'individus.

Perspectives et travaux futurs

Le résultat de notre travail bien qu'encourageant peut donner lieu à des améliorations à différents niveaux. Nous en présentons ci-dessous plusieurs axes d'approfondissement.

Audit des workflows dans un réseau hétérogène : Le prototype que nous avons réalisé pour la détection des intrusions dans un système d'information par audit des workflows a été expérimenté sur un poste de travail muni d'un système d'exploitation Windows. Nous nous proposons de l'améliorer en étendant son utilisation dans un réseau hétérogène. Ce faisant, nous pourrons mettre à contribution un administrateur réseau qui devra être alerté chaque fois qu'une opération malveillante est effectuée sur un poste de travail du réseau.

Outil d'aide aux investigations : le cadre formel de l'investigation numérique proposé dans ce travail se décompose en plusieurs primitives forensics qui doivent être exécutées au cours d'une investigation véritable. Nous envisageons la transformation des procedures forensics concus dans ce travail en un package unique qui pourra être utilisé comme de logiciel libre au sein des administrations en charge de l'application de la loi dans le cadre des investigations numériques.

Investigation à distance : Pour les cas étudiés, nous avons constaté que le modèle d'investigation déployé a permis d'investiguer proprement sur un cas d'attaque par déni de service sur un système physiquement accessible. Ceci n'est pas toujours le cas. En effet, les composants de certains systèmes d'information à investiguer

peuvent se situer à des emplacements dont l'accès physique est difficile. Ainsi, l'informatique dématérialisé peut être mis à contribution pour collecter et préserver les informations collectées dans le cadre d'une investigation délocalisée (remote forensics). Ainsi, un serveur délocalisé pourrait permettre aux enquêteurs associés à partir des pays différents de collaborer et d'améliorer les capacités des outils employés pour augmenter les chances de faire aboutir une enquête.

Investigations contributives : Puisque les investigations se conduisent toujours par équipe, la troisième perspective vise la mise sur pied d'une architecture sécurisée de travail en groupe qui, exploitant les avantages des signatures de groupes (sécurité, fiabilité des échanges et optimisation de la répartition des tâches), facilitera les échanges de données entre les enquêteurs d'une équipe.

Investigation comme service : Une perspective à long terme concerne la définition d'un cadre formel d'investigation adapté à la technologie du cloud computing. En effet, les outils forensics actuellement en exploitation sont non seulement utilisés dans le cadre d'une investigation réactive mais, ils ne permettent pas également la collecte de preuves sur un poste de travail distant. Par ailleurs, pour les enquêteurs du domaine de l'informatique légale, le cloud computing est synonyme de nouveaux défis touchant tant les technologies utilisées que les méthodes d'enquêtes. Il s'agit :

- des considérations extratéritoriales de tracabilité ;
- de la sécurité des ressources "virtualisées" ;
- de la collecte des données et la chaine de conservation ;
- de l'acquisition, de la segrégation et de la préservation d'une preuve dans un environnement infonuagique.

Bibliographie

[1] Anderson Ross, Security Engineering : "A Guide To Building Dependable Distributed Systems", Wiley, ISBN 0-471-38922-6, 2001.

[2] Dufour Arnaud, Ghernaouti-Hélie Solange : "Internet - PUF, Que sais-je ?", Ed Dunod, N° 3073 - ISBN : 2-13-053190-3, 2004.

[3] Brian Carrier Eugene H. Spafford : "Getting Physical with the Digital Investigation Process". International Journal of Digital Evidence, Volume 2, Issue 2, Feb 2003.

[4] Soltan Alharbi, Jens Weber-Jahnke, Issa Traore : "The Proactive and Reactive Digital Forensics Investigation Process : A Systematic Literature Review". International Journal of Security and Its Applications Vol. 5 No. 4, October 2011.

[5] Ieong, R. and H. Leung : "Deriving Cse-specific Live Forensics Investigation Procedures from FORZA". In Symposium on Applied Computing archive, Proceedings of the 2007 ACM symposium on Applied computing, Seoul, Korea : ACM Press New York, USA, 2007.

[6] CP Grobler, CP Louwrens, SH Von Solms : "A multi-component view of Digital Forensics". International Conference on Availability, Reliability and Security (ICARS), IEEE Computer Society, ARES 61, 2010.

[7] Ryan Leigland and Axel W. Krings : " A Formalization of Digital Forensics". International Journal of Digital Evidence (IJDE). Fall 2004, Volume 3, Issue 2, 2004.

[8] Gary C. Kessler : "Anti-Forensics and the Digital Investigator". In Proceedings of the 5th Australian Digital Forensics Conference, 2007.

[9] H. Axlerod and D. Jay : "Crime and Punishment in Cyberspace : Dealing with Law Enforcement and the Courts". In Proceedings of the 27th Annual ACM SIGUCCS Conference on User Services,1999.

[10] B. Grundy : "The Law Enforcement and Forensic Examiner Introduction to Linux : A Beginner's Guide ", Available at http ://www.linux-forensics.com/linuxintro-LEFE-2.0.5.pdf, January 2004.

[11] Digital Forensic Research Workshop : "A Road Map for Digital Forensics", Research 2001, Digital Forensics Research Workshop, Available at http ://www.dfrws.org/dfrws-rm-final.pdf. 6 November 2001.

[12] R. Erbacher et al : "Computer Forensics Education", IEEE Security and Privacy, August 2003.

[13] Giancarlo De Maio : "On The Evolution of Digital Evidence : Novel Approaches for Cyber Investigation", Tesi di Dottorato in Informatica, Dipartimento Di Informatica, Universita Degli studi Di Salerno, "Renato M. Capocelli", anno accademico 2012-2013.

[14] Union internationale des télécommunications : "Guide de la cybersécurité pour les pays en développement". Edition 2007.

[15] Brian Cusack and Muteb Alqahtani : "Acquisition of evidence from network intrusion detection systems". Edith Cowan University, Research Online, Australian Digital Forensics Conference, 2013.

[16] Jorge Herrerias and Roberto Gomez : "A log correlation model to support the evidence search process in a forensic investigation". In Systematic Approaches to Digital Forensic Engineering, 2007. SADFE 2007. Second International Workshop on, pages 31–42. IEEE, 2007.

[17] Fahmid Imtiaz : "Intrusion detection system logs as evidence and legal aspects". Technical report, School of Computer and Information Science Edith Cowan University, 2007.

[18] Komal Barhate and CD Jaidhar : "Automated digital forensic technique with intrusion detection systems". In Advance Computing Conference (IACC), 2013 IEEE 3rd International, pages 185–189. IEEE, 2013.

[19] Kristin M. Finklea, Catherine A. Theohary : "Cybercrime : conceptual issues for congress and U.S. law enforcement". The United State Department of Justice (USDOJ), Congress Research Service (CRS), 7-5700, R42547, January 2013.

[20] Mark Taylor, John Haggerty, David Gresty, David Lamb : "Forensic investigation of cloud computing systems", Elsevier, Network Security, Vol 2011, Issue 3, P 4-10, April 2011.

[21] Heady, R., G. Luger, A. Maccabe, and M. Servi lia : "The Architecture of a Network Level Intrusion Detection System". Technical report, Computer Science Department, University of New Mexico, 1990.

[22] A. Kartit, A. Saidi, F. Bezzazi, M. El marraki, A. Radi : "A new approach to intrusion detection system ". Journal of Theoretical and Applied Information Technology, Vol. 36, No.2, ISSN : 1992-8645, www.jatit.org, E-ISSN : 1817-3195, 29th February 2012.

[23] Lunt, T. : "Detecting Intruders in Computer Systems". In : Proceedings of the 1993 Conference on Auditing and Computer Technology, 1993.

[24] Karthikeyan .K.R and A. Indra : "Intrusion Detection Tools and Techniques - A Survey". International Journal of Computer Theory and Engineering, Vol.2, No.6, December, 2010.

[25] Atsa Etoundi Roger : " ATSERO Method : A Guideline for Business Process and Workflow Modeling Within an Enterprise ". International Journal of Scientific Engineering Research, Dec 2011.

[26] S. L. Garfinkel : "Digital forensics research : The next 10 years". Digital Investigation, vol. 7, pp. S64-S73, 2010.

[27] Peter Sommer : "Intrusion Detection Systems as Evidence". Computer Networks 31, 1999.

[28] Loi n° 2010/012 du 21 décembre 2010 relative a la cybersécurité et a la cybercriminalité au Cameroun

[29] Susan W. Brenner : "Organized Cybercrime ? How Cyberspace May Affect the Structure of Criminal Relationships". North Carolina Journal of Law and Technology, Volume 4, Issue 1, Fall 2002.

[30] Kumar, S. and E. H. Spafford : "A Software Architecture to Support Misuse Intrusion Detection". In Proceedings of the 1sth National Information Security Conference, 1995.

[31] Abdullah Alshalan "Cyber-Crime Fear and Victimization : An Analysis of A National Survey". Department of Sociology, Anthropology, and Social Work, Mississippi State University, September 21, 2005.

[32] David S. Wall : "The Internet as a Conduit for Criminal Activity", Information Technology and the Criminal Justice System, Pattavina, A., ed., Sage Publications, Inc., pp. 77-98, 2010.

[33] H. Axlerod and D. Jay : "Crime and Punishment in Cyberspace : Dealing with Law Enforcement and the Courts". In Proceedings of the 27th Annual ACM SIGUCCS Conference on User Services,1999.

[34] 16 Sinrod, Eric J., Reilly, William P. : "Cyber-Crimes : A Practical Approach to the Application of Federal Computer Crime Laws". Santa Clara Computer & High Tech. L. J. 177 (2000).

[35] Karlovassi, Samos, Greece : "Internet Forensics : Legal and Technical Issues", IEEE Computer Society, Second International Workshop on Digital Forensics and Incident Analysis, ISBN : 0-7695-2941-0, 2007.

[36] Technical Working Group for Electronic Crime Scene Investigation : " Electronic Crime Scene Investigation : A Guide for First Responders". United States Department of Justice, 2001.

[37] Kristin Archick : "Cybercrime : The Council of Europe Convention", CRS Report for Congress, Congressional Research Service, The Library of Congress, Order Code RS21208, April 26, 2002.

[38] G. Palmer : "A road map for digital forensics research-report from the first Digital Forensics Research Workshop (DFRWS)". Utica, New York, 2001.

[39] R. Mark, C. Clint, and G. Gregg : "An Examination of Digital Forensic Models". International Journal of Digital Evidence, vol. 1, pp. 1-12, 2002.

[40] B. Carrier and E. Spafford : "Getting physical with the digital investigation process". International Journal of Digital Evidence, vol. 2, pp. 1-20, 2003.

[41] P. Stephenson : "A comprehensive approach to digital incident investigation". Information Security Technical Report, vol. 8, pp. 42-54, 2003.

[42] Baryamureeba and Florence Tushabe : "The Enhanced Digital Investigation Process Model". Institute of Computer Science, Makerere University P.O.Box 7062, Kampala Uganda www.makerere.ac.ug/ics.

[43] S. O. Ciardhuain : "An extended model of cybercrime investigations". International Journal of Digital Evidence, vol. 3, 2004.

[44] B. Carrier and E. Spafford : "An event-based digital forensic investigation framework". In In Proceeding of the 4th Digital Forensic Research Workshop, pp. 11-13, 2004.

[45] W. Harrison : "The digital detective : An introduction to digital forensics". In Advances in Computers, Vol. 60. vol. 60, pp. 75-119, 2004.

[46] N. L. Beebe and J. G. Clark : "A hierarchical, objectives-based framework for the digital investigations process". Digital Investigation, vol. 2, pp. 147-167, 2005.

[47] M. Kohn, J. Eloff, and M. Olivier : "Framework for a digital forensic investigation". In Proceedings of Information Security South Africa (ISSA) 2006 from Insight to Foresight Conference, 2006.

[48] K. Kent, S. Chevalier, T. Grance, and H. Dang : "Guide to Integrating Forensic Techniques into Incident Response". NIST Special Publication 800-86, 2006.

[49] M. Rogers, J. Goldman, R. Mislan, T. Wedge, and S. Debrota : "The Computer forensics field triage process model". Journal of Digital Forensics, Security and Law, vol. 1, pp. 27-40, 2006.

[50] R. S. C. Ieong : "FORZA - Digital forensics investigation framework that incorporate legal issues". Digital Investigation, vol. 3, pp. 29-36, 2006.

[51] F. Freiling and B. Schwittay : "A common process model for incident response and computer forensics". In 3rd International Conference on IT-Incident Management and IT- Forensic, 2007.

[52] M. Khatir, S. M. Hejazi, and E. Sneiders : "Two-Dimensional Evidence Reliability Amplification Process Model for Digital Forensics". In Digital Forensics and Incident Analysis, WDFIA '08, Third International Annual Workshop, pp. 21-29, 2008.

[53] S. Yong-Dal : "New Digital Forensics Investigation Procedure Model". in Networked Computing and Advanced Information Management, NCM '08. Fourth International Conference, pp. 528-531, 2008.

[54] D. Billard : "An Extended Model for E-Discovery Operations". In Advances in Digital Forensics V. vol. 306, G. Peterson and S. Shenoi, Eds., ed Springer Boston, pp. 277-287, 2009.

[55] R. Rowlingson : "A ten step process for forensic readiness". International Journal of Digital Evidence, vol. 2, pp. 1-28, 2004.

[56] Yong-Dal Shin. "New Model for Cyber Crime Investigation Procedure". Journal of Next Generation Information Technology, Volume 2, Number 2, May 2011.

[57] Sundresan Perumal : "Digital Forensic Model Based On Malaysian Investigation Process". IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.8, August 2009.

[58] Rodney McKemmish : "What is Forensic Computing ?". Australian Institute of Criminology, Trends & issues in crime and criminal justice, ISSN 0817-8542, ISBN 0642241023, June 1999.

[59] E. Casey : " Digital Evidence and Computer Crime ". Elsevier Acedemic Press, 2004.

[60] N. L. Beebe and J. G. Clark : "A hierarchical, objectives-based framework for the digital investigations process". Digital Investigation, vol. 2, pp. 147-167, 2005.

[61] M. Kohn, J. Eloff, and M. Olivier : "Framework for a digital forensic investigation". In Proceedings of Information Security South Africa (ISSA) 2006 from Insight to Foresight Conference, 2006.

[62] Nathan Balon, Ronald Stovall, Thomas Scaria : "Computer Intrusion Forensics". Research Paper, CIS 544, 2004.

[63] Eugene Spafford Diego Zamboni : "Data collection mechanisms for intrusion detection systems". Center for Education and Research in Information Assurance and Security 1315 Recitation Building Purdue University West Lafayette, IN 47907-1315 CERIAS Technical Report 2008.

[64] Christophe Bidan, Guillaume Hiet, Ludovic M, Benjamin Morin, Jacob Zimmermann : "Vers une détection d'intrusion à fiabilité et pertinence prouvables". Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS), October 2003.

[65] Mboupda Moyo Achille, Atsa Etoundi Roger : " Modelling and Verification of Group Signatures Properties ", International Journal of Applied Information Systems (IJAIS) - ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 3- No.9, February 2012 - www.ijais.org.

[66] Inikpi O. Ademu, Dr Chris O. Imafidon, Dr David S. Preston : "A New Approach of Digital Forensic Model for Digital Forensic Investigation". (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No.12, 2011.

[67] P. Stephenson : "Modeling of Post-Incident Root Cause Analysis". International Journal of Digital Evidence, Vol. 2 Issue 2, Fall 2003.

[68] National Institute of Justice : "Computer Forensic Testing Tool Program". Available at http ://www.ojp.usdoj.gov/nij/sciencetech/cftt.htm, 2004.

[69] K. Goldman and E. Mackenzie : "Computer Abuse". Information Technologies and Judicial Affairs, Proceedings of the 28th annual ACM SIGUCCS Conference on User Services, pp. 170-176, 2000.

[70] Stephen Herzog : "Revisiting the Estonian Cyber Attacks : Digital Threats and Multinational Responses". Journal of Strategic Security, Strategic Security in the Cyber Age,Volume 4, Number 2, Article 4, Summer 2011.

[71] Michael B. Kelley : "The Stuxnet Attack On Iran's Nuclear Plant Was 'Far More Dangerous' Than Previously Thought". Military & Defense, Nov 2013. Availlable at http ://www.businessinsider.com/stuxnet-was-far-more-dangerous-than-previous-thought-2013-11ixzz34KDwNjrk

[72] Marusterl, G.Schimm, and AJ.M.M.Weijters : "Workflow Mining : A Survey of issues and Approaches". Data and Knowledge Engineering, Accepted for publication, 2003.

[73] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster : "Workflow Mining : Discovering Process Models from Event Logs". IEEE Transactions on Knowledge and Data Engineering (TKDE), Accepted for publication, 2003.

[74] Hisham M. Haddad and Brunil D. Romero : "Asset Identification for Security Risk Assessment in Web Applications". Int.J. of Software Engineering, IJSE Vol.2 No.3 December 2009.

[75] Atsa E. R., Fouda M. and Abessolo G. : "A Denotational Semantics Me-thodology (DSM) Approach for Business Processes Modelling". International Journal of Computer Applications, Vol 1, N°1, 2010.

[76] Atsa Etoundi Roger, Mboupda Moyo Achille, Nkoulou Onanena Georges, Nkondock Mi Bahanag Nicolas : " A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit ", International Journal of Computer Applications (0975 8887) Volume 81 - No. 1, November 2013.

[77] Bharat S. Dhak, Shrikant Lade : "An Evolutionary Approach to Intrusion Detection System using Genetic Algorithm". International Journal of Emerging Technology and Advanced Engineering. ISSN 2250-2459, ISO 9001 :2008, Certified Journal, Volume 2, Issue 12, December 2012.

[78] A. Ojugo, O. Eboka, E. Okonta, E Yoro, O. Aghware : "Genetic Algorithm Rule-Based Intrusion Detection System (GAIDS)". Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 8, ISSN 2079-8407, Aug 2012.

[79] Yogita B. Bhavsar, Kalyani C.Waghmare : "Intrusion Detection System Using Data Mining Technique : Support Vector Machine". International Journal of Emerging Technology and Advanced Engineering, Website : www.ijetae.com, ISSN 2250-2459, ISO 9001 :2008 Certified Journal, Volume 3, Issue 3, March 2013.

[80] Parekh s.p, Madan b.s, Tugnayat r.m : "Approach for intrusion detection system using data mining". Journal of Data Mining and Knowledge Discovery ISSN : 22296662, and ISSN : 22296670, Volume 3, Issue 2, pp.-83-87, 2012.

[81] Ryan Leigland, Axel W. Krings : "A Formalization of Digital Forensics". International Journal of Digital Evidence, Volume 3, Issue 21, Fall 2004.

[82] Peter Stephenson : "The Application of Intrusion Detection Systems in a Forensic Environment". United States, Department of Justice, Executive Office for United States Attorneys, Washington, DC 20530, Volume 59, Number 6, November 2011.

[83] Computer Emergency Response Team : " CERT/CC Statistics " 1988-2003, Carnegie Mellon University Software Engineering Institute, CERT Coordination Center, 22 January 2004. Available : http ://www.cert.org/stats

[84] D. Dittrich : "The DoS Project's trinoo distributed denial of service attack tool". Available at http ://staff.washington.edu/dittrich/misc/trinoo.analysis.txt

[85] Technical Working Group for Electronic Crime Scene Investigation : "Electronic Crime Scene Investigation : A Guide for First Responders". United States Department of Justice, 2001.

[86] Apple Computer : "Technical Notes". Apple Computer. Available at http ://developer.apple.com/technicalnotes/

[87] Jim Yuill, S. Felix Wu, Fengmin Gong, Ming-Yuh Huang : "Intrusion Detection for an On-Going Attack". International Journal of Computer Science & technology Vol. 2, Issue 4, Oct. - Dec. 2011.

[88] Ankit Agarwal, Megha Gupta, Saurabh Gupta, S.C. Gupta : "Systematic Digital Forensic Investigation Model". International Journal of Computer Science and Security (IJCSS), Volume(5) : Issue(1) : 118-131, 2011.

[89] Karen S., Murugiah S., Amanda C., Angela O. : "Technical Guide to Information Security Testing and Assessment". National Institute of Standards and technology (NIST), US Department of Commerce, Gaithersburg, MD 20899-8930, September 2008.

[90] G. Booch, J. Rumbaugh, and I. Jacobson : "The Unified Modeling Language User Guide". Addison Wesley, 1999.

[91] Atsa Etoundi Roger and Mboupda Moyo Achille : "Multi-perspective Cybercrime Investigation Process Modeling". International Journal of Applied Information Systems 2(8) :14-20, Published by Foundation of Computer Science, New York, USA, June 2012.

[92] Biswanath Mukherjee, Todd L. Heberlein, Karl N. Levitt : "Network intrusion detection". IEEE Network, 8(3) :2641, May/June 1994.

[93] Amen Souissi : "Modélisation centrée sur le procéssus métier pour la génération complète de portail collaboratifs", thèse de Doctorat, Sciences et Technologies, Université Lille 1, décembre 2013.

[94] K.Rajasekhar, B.Sekhar Babu, P.Lakshmi Prasanna, D.R.Lavanya, T.Vamsi Krishna : "An Overview of Intrusion Detection System". InternatIonal Journal of Computer Science & technology Vol. 2, Issue 4, Oct. - Dec. 2011.

[95] Philippe Bogaerts : "HPING tutorial". http ://www.radarhack.com. Version 1.5 24-08-2003

[96] D. Dittrich : "The Stacheldraht Distributed Denial of Service Attack Tool". Availlable at http ://staff.washington.edu/dittrich/misc/stacheldraht.analysis.txt, December 1999.

[97] Bill Allen : "Collecting digital evidence from Intrusion Detection Systems". CGS 5132. Computer Forensic II, Spring 2002.

[98] Vera Marinova Boncheva : "A short survey of Intrusion Detection Systems". Problems of Engineering Cybernetics and Robotics, 58, Soa, 2007.

[99] Rafeeq Ur Rehman : "Intrusion Detection Systems with Snort : Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID" Ed 2003.

[100] Ren, W. and H. Jin : "Honeynet Based Distributed Adaptive Network Forensics and Active Real Time Investigation". In 2005 ACM Symposium on Applied Computing, Santa Fe, New Mexico, USA, 2005.

[101] Mboupda Moyo Achille, Atsa Etoundi Roger : "Obtaining Digital Evidence from Intrusion Detection Systems", International Journal of Computer Applications (0975 8887), Doi 10.5120/13973-1970, Volume 95 - No. 1, June 2014. www.ijca.org.

[102] Nkondock Mi Bahanag Nicolas : "Framework de détection d'intrusions dans un système d'information basé sur l'analyse des workflows", Mémoire de Master II, Université de Yaoundé I, Département d'Informatique, Fevrier 2014.

Annexe

Détection des intrusions par audit des workflows : cas de Windows

L'outil de détection des intrusions tel que décrit dans cette partie permet de détecter certaines opérations qui surviennent dans un ordinateur en violation de la politique de sécurité. Il s'agit principalement de la détection en temps réel, de la modification, de la création,ou de la suppression de fichiers d'une arborescence donnée dans un ordinateur équipé d'un système d'exploitationWindows. Grâce à cet outil, il devient possible de monitorer un système via la détection automatique des évènements qui modifient son environnement. Ces informations sont disponibles et affichées sur l'écran d'un administrateur. Etant donné que le modèle mis en application étudie le comportement des ressources pour détecter une intrusion, des fichiers logs sont associés à ces ressources pour garder une trace des workflows qui découlent de leurs multiples échanges. Dans le prototype réalisé, les tâches prises en compte et relatives aux fichiers sont [102] :

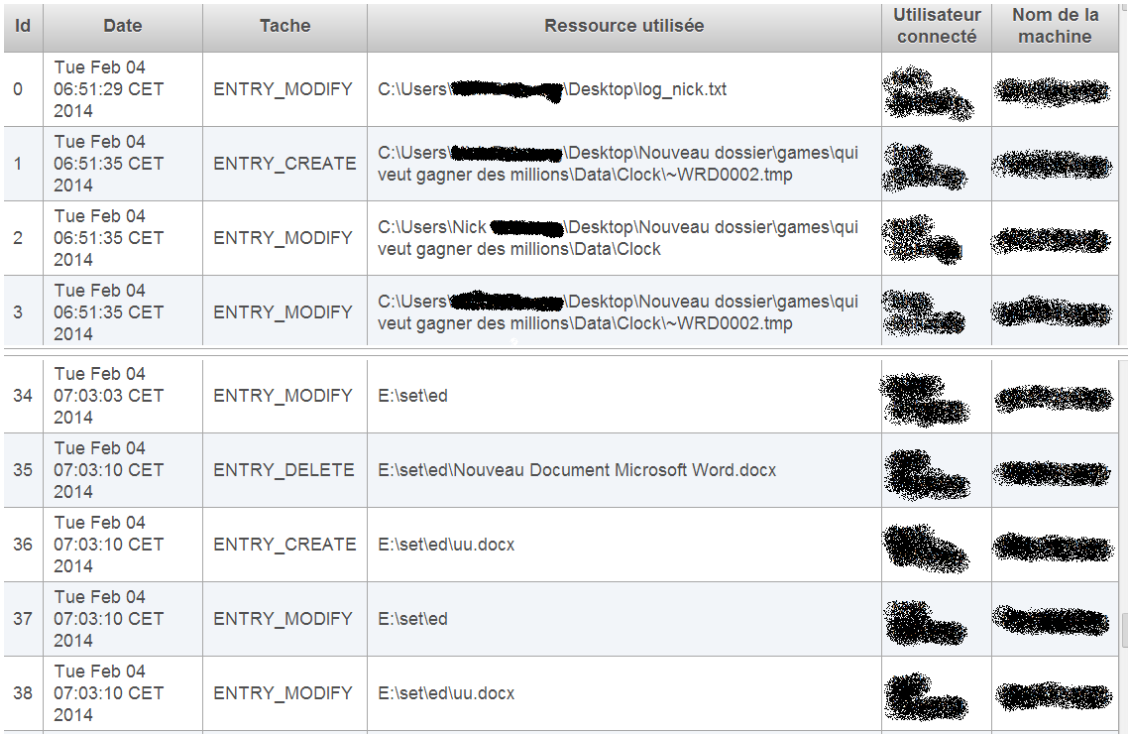
— $ENTRY\_CREATE$ qui indique la création d'un fichier ;
— $ENTRY\_MODIFY$ pour la modification d'un fichier ;
— $ENTRY\_DELETE$ qui indique la suppression d'un fichier.

Les ressources contrôlées dans ce cas sont les différents disques et les simples dossiers de l'arborescence Windows. Ainsi, un sous-dossier est une ressource intermédiaire. Les fichiers quant à eux sont des ressources terminales. Les principaux disques et répertoires particuliers contrôlés sont les suivants : Disque C ; Disque D ; Disque E ; Disque F ; le Bureau Windows ; le répertoire des Documents personnels ; le dossier de téléchargements ; les dossiers multimédia par défaut (musique, images et vidéos).

Il est important de remarquer qu'ici, les disques C et D sont des partitions du disque dur. Il en existe 2 dans le poste de travail d'expérimentation. Les disques E et F sont principalement les racines des arborescences associées aux ports usb. a chacune des ressources ci-dessus listées est associé un fichier journal qui contient un ensemble d'évènements. Dans l'implémentation du modèle, les journaux $logs\_in$ et log_out associés aux ressources sont fusionnés pour contenir les évènements relatifs aux tâches que nous avons définies et concernant non seulement la ressource considérée mais également ses sous ressources. Les évènements (figure 4.7)qu'on y retrouve contiennent les informations suivantes :

— Id qui est l'identifiant de l'évènement dans le log ;
— $Date$ renseigne sur l'heure, le jour, le mois et l'année de l'évènement ;

— $Tche$ précise de quel type de tâche il s'agit : $ENTRY\_CREATE$, $ENTRY\_MODIFY$ ou $ENTRY\_DELETE$;
— $Ressourceutilise$ est un champ qui renseigne sur le fichier utilisé (créé, modifié ou supprimé) ;
— $Utilisateurconnect$ fournit le nom du compte utilisateur ouvert lorsque l'évènement s'est produit ;
— $Nomdelamachine$ est un champ qui est beaucoup plus intéressant lorsque l'outil est déployé en réseau, ce qui constitue une amélioration du prototype.



Id	Date	Tache	Ressource utilisée	Utilisateur connecté	Nom de la machine
0	Tue Feb 04 06:51:29 CET 2014	ENTRY_MODIFY	C:\Users\[redacted]\Desktop\log_nick.txt	[redacted]	[redacted]
1	Tue Feb 04 06:51:35 CET 2014	ENTRY_CREATE	C:\Users\[redacted]\Desktop\Nouveau dossier\games\qui veut gagner des millions\Data\Clock\~WRD0002.tmp	[redacted]	[redacted]
2	Tue Feb 04 06:51:35 CET 2014	ENTRY_MODIFY	C:\Users\Nick[redacted]\Desktop\Nouveau dossier\games\qui veut gagner des millions\Data\Clock	[redacted]	[redacted]
3	Tue Feb 04 06:51:35 CET 2014	ENTRY_MODIFY	C:\Users\[redacted]\Desktop\Nouveau dossier\games\qui veut gagner des millions\Data\Clock\~WRD0002.tmp	[redacted]	[redacted]
34	Tue Feb 04 07:03:03 CET 2014	ENTRY_MODIFY	E:\set\ed	[redacted]	[redacted]
35	Tue Feb 04 07:03:10 CET 2014	ENTRY_DELETE	E:\set\ed\Nouveau Document Microsoft Word.docx	[redacted]	[redacted]
36	Tue Feb 04 07:03:10 CET 2014	ENTRY_CREATE	E:\set\ed\uu.docx	[redacted]	[redacted]
37	Tue Feb 04 07:03:10 CET 2014	ENTRY_MODIFY	E:\set\ed	[redacted]	[redacted]
38	Tue Feb 04 07:03:10 CET 2014	ENTRY_MODIFY	E:\set\ed\uu.docx	[redacted]	[redacted]

Figure 4.7 – Journal des évènements

Présentation Technique

Pour implémenter cette solution, nous avons utilisé $Javaweb$, notamment la version de jdk (java development kit) 1.7. Ce choix est motivé par :
— Sa portabilité : Le code $java$ peut facilement s'intégrer à différents environnements ;
— $Java$ possède une très grande communauté de développeurs qui sont très souvent ouverts ce qui rend la tâche de programmation beaucoup moins complexe ;
— Les primitives $Java$ possèdent également une bonne documentation ;
— Il s'agit d'un langage permettant la réalisation aisée d'interfaces riches.
L'environnement de devéloppement utilisé est $Netbeans$ version 7.3 et le serveur d'application (puisqu'il s'agit d'une application web) utilisé est $Glassfish$ version 3.1.2.2 qui s'intègre facilement à $Netbeans$. $Primefaces$ est le framework utilisé pour gérer les vues qui sont en réalité des interfaces web. Le stockage des données se fait grâce à des fichiers. Tous les évènements sont stockés dans les logs. Ainsi, lorsqu'une tâche s'exécute

en utilisant l'une des ressources contrôlées, le programme Java se charge de fabriquer l'évènement en y ajoutant les autres informations à savoir : la date et l'heure à laquelle cela s'est produit, la sous ressource qui est concernée, le nom de l'utilisateur connecté et enfin le nom de la machine. Ces informations sont concaténées bien que séparées par des ' ; ' et sont ensuite rangées dans une variable de type chaine de caractère. C'est le contenu de cette variable qui est inséré au début du fichier log (figure 4.8).

```
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001e;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001f;Nick Bahanag;NickBahanag
CET 2014;ENTRY_CREATE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001f;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001b;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001c;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001a;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_00001d;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_000019;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Cache\f_000018;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Windows\Prefetch;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\NTUSER.DAT;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\ntuser.dat.LOG1;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Sync Data\SyncData.sqlite3;Nick Bahanag;NickE
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Sync Data\SyncData.sqlite3-journal;Nick Bahan
CET 2014;ENTRY_MODIFY;C:\Windows\System32\config\SOFTWARE;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Windows\System32\config\SOFTWARE.LOG1;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Roaming\NetBeans\7.3\config\Preferences\org\netbeans\modules\uihandler.properties;N
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences~RF2b7537a.TMP;Nick Bahanag;NickBa
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences;Nick Bahanag;NickBahanag
CET 2014;ENTRY_CREATE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\5019.tmp;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences~RF2b7537a.TMP;Nick Bahanag;NickBa
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences;Nick Bahanag;NickBahanag
CET 2014;ENTRY_CREATE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences~RF2b7537a.TMP;Nick Bahanag;NickBa
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\5019.tmp;Nick Bahanag;NickBahanag
CET 2014;ENTRY_CREATE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\5019.tmp;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Shortcuts;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Session Storage;Nick Bahanag;NickBahanag
CET 2014;ENTRY_MODIFY;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Shortcuts-journal;Nick Bahanag;NickBahanag
CET 2014;ENTRY_DELETE;C:\Users\Nick Bahanag\AppData\Local\Google\Chrome\User Data\Default\Preferences~RF2b70451.TMP;Nick Bahanag;NickBa
```

Figure 4.8 – Contenu du fichier log

Par ailleurs, un démon a été développé pour scruter le log et actualiser l'interface web chargée de présenter le contenu du fichier à l'utilisateur.

Quelques difficultés ont émaillé cette expérimentation. En effet, pour un contrôle stricte du comportement des ressources, il faut identifier tous les comportements possibles de chaque ressource en fonction de la plate-forme (Linux, Windows) auxquels l'on devra associer des observateurs. Pour le cas d'espèce, nous nous sommes limités à la création, modification et suppression des fichiers.

Glossaire

cia Central intelligence Agency, Agence Centrale du Renseignement américain. 27

cookies informations personnelles d'accès à un site Internet donné. 24

cpo Complete Partial Order: Le CPO est un ensemble partiellement ordonné qui a un plus petit élément et dont toutes les chaînes ont une borne supérieure. 46

crime informatique infraction pour laquelle un système informatique est l'objet du délit et/ou le moyen de le réaliser. 11

criminalité en col blanc activités financières illégales qui échappent aux lois des différents pays. 11

csi The Computer Security Institute (CSI) est une organisation professionnelle au service des administrateurs réseaux et responsables de la sécurité des systèmes d'information. 29

cyber-patrouilles cyber-patrouilles: actes de patrouille effectués dans le cyber espace par les officiers de police judiciaire. 65

cyberespace ensemble de données numérisées constituant un univers d'information et un milieu de communication, lié à l'interconnexion mondiale des ordinateurs. 3

cyberperquisition encore appelée perquisition numérique, elle consiste en la recherche d'éléments de preuve d'une infraction commise sur internet. Cette recherche d'éléments de preuve peut nécessiter la perquisition de l'outil informatique ayant servi à réaliser l'infraction ou à se connecter sur internet. 33

digital forensic models modèles d'investigation numérique. xv

dns Domain Name System (système de noms de domaine). 25

ebios Expression des Besoins et Identification des Objectifs de Sécurité. 44

eléments matériels événement entrant dans les prévisions d'une incrimination. 6

emergence d'une entreprise il s'agit pour une entreprise d'atteindre ses objectifs stratégiques, de prendre des décisions sur la base des données contenues dans son SI. C'est à dire, un alignement stratégique à ces visions. 5

enquête judiciaire investigations effectuées par la police judiciaire, pour rechercher les auteurs d'une infraction et pour déterminer les conditions dans lesquelles elle a été commise. 6

enquête préliminaire enquête diligentée d'office ou à la demande du Parquet par la Police ou la Gendarmerie avant l'ouverture de toute information et permettant au ministère public d'être éclairé sur le bien-fondé d'une poursuite. 32

ftp File Transfer Protocol (protocole de transfert de fichiers). 25

hids Host intrusion detection system. 39

honneypot Poste de travail configuré avec des services actifs pour leurrer les pirates. 93

iana Internet Assigned Numbers Authority. 28

icann Internet Corporation for Assigned Names and Numbers. 28

ids Intrusion detection system. 4

imap Internet Message Access Protocol. 25

infractions classiques entorses à une loi ou à une réglémentation en vigueur dans un pays, qui ne font pas intervenir l'usage des technologies de l'information et de la communication. 2

ip Internet Protocol. 4, 114

kids Kernel intrusion detection system. 39

mooc Massive Open Online Course: exemple de formation ouverte et à distance en téléenseignement où les participants aux cours, enseignants et élèves, sont dispersés géographiquement et communiquent uniquement par Internet. 2

métadonnées donnée servant à définir ou décrire une autre donnée quel que soit son support (papier ou électronique). 68

nids Network intrusion detection system. 31

nsa National Security Agency: organisme gouvernemental du département de la Défense des États-Unis, responsable du renseignement d'origine électromagnétique, de la sécurité des systèmes d'information et du traitement des données du gouvernement américain. 3, 27

ocde Organisation de Coopération et de Développement Economiques; organisation internationale qui a pour mission de promouvoir les politiques qui amélioreront le bien-être économique et social partout dans le monde. Elle offre aux gouvernements un forum où ils peuvent conjuguer leurs efforts, partager leurs expériences et chercher des solutions à des problèmes communs. 11

phase policière enquête de police judiciaire (police ou gendarmerie), qui s'effectue sous le contrôle du parquet. Elle consiste en l'accomplissement d'actes de procédure destinés à réunir un ensemble d'éléments tendant à établir la véracité ou au contraire l'inconsistance des allégations de la victime (auditions de témoins, perquisitions, gardes à vue, constatations ou prélèvements nécessaires aux examens de police scientifique, etc.). 6

prism Programme qui permet aux autorités américaines, via son Agence Nationale de Sécurité, d'accéder sans restriction à toutes les informations hébergées et traitées par tous les géants du web. Les données ciblées sont les e-mails, les discussions instantanées, les vidéos, les photos, les données stockées, la voix sur IP, le transfert de fichiers, etc.. 3

procédures forensics il s'agit des procédures judiciaires à suivre pour une investigation numérique. Elle dépendent de l'Etat dans lequel l'investigation numérique est menée. 21

psi propriétaire des systèmes d'information. 5

rfc (Request For Comments) ensemble de documents qui font référence auprès de la Communauté Internet et qui décrivent, spécifient, aident à l'implémentation, standardisent et débattent de la majorité des normes, standards, technologies et protocoles liés à Internet et aux réseaux en général. 36

smtp Simple Mail Transfer Protocol. 25

ssh Secure Shell. 25

tcp Transmission Control Protocol. 4

vpn Virtual Private Network, Réseau privé virtuel. 34

Articles publiés dans les Conférences et journaux scientifiques

A) Conférences de recherches scientifiques avec comités de lecture

- **Mboupda Moyo Achille** and Atsa Etoundi Roger: " **A Basic Intrusion Detection Model For Cybercrime Investigation Purposes** ", In proceeding in Conference de Recherche en Informatique (CRI'2013), 09 - 13 Décembre 2013 Yaoundé, Cameroun. http://www.cri.uninet.cm/.

- Atsa Etoundi Roger, Nkoulou Onanena Georges, Nkondock Mi Bahanag Nicolas and **Mboupda Moyo Achille**: "**A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit**", In proceeding in Conference de Recherche en Informatique (CRI'2013), 09 - 13 Décembre 2013 Yaoundé, Cameroun. http://www.cri.uninet.cm/.

B) Publication dans les journaux scientifiques avec comités de lecture

- **Mboupda Moyo Achille**, Atsa Etoundi Roger: " **Modelling and Verification of Group Signatures Properties** ", International Journal of Applied Information Systems (IJAIS) - ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 3- No.9, February 2012 - www.ijais.org.

- Atsa Etoundi Roger, **Mboupda Moyo Achille**: " **MultiPerspective Cybercrime Investigation Process Modeling** ", International Journal of Applied Information Systems (IJAIS) - ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 2- No.2, June 2012 - www.ijais.org.

- Atsa Etoundi Roger, **Mboupda Moyo Achille**, Nkoulou Onanena Georges, Nkondock Mi Bahanag Nicolas : " **A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit** ", International Journal of Computer Applications (0975 8887), Doi 10.5120/13973-1964, Volume 81 - No. 1, November
2013. www.ijca.org.

- **Mboupda Moyo Achille**, Atsa Etoundi Roger: " **Obtaining Digital Evidence from Intrusion Detection Systems** ", International Journal of Computer Applications (0975 8887), Doi 10.5120/13973-1970, Volume 95 - No. 1, June 2014. www.ijca.org.

I

Mboupda Moyo Achille, Atsa Etoundi Roger: " **Modelling and Verification of Group Signatures Properties** ", International Journal of Applied Information Systems (IJAIS) - ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 3-No.9, February 2012 - www.ijais.org.

Indexed with CiteSeer, NASA Hardvard ADS, DOAJ, Google Scholar, DBLP, Scientific Commons, Informatics, ProQuest CSA Technology Research Database.

FEVRIER 2012

International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 3– No.9, February 2012 – www.ijais.org

Modelling and Verification of Group Signatures Properties

MBOUPDA MOYO Achille
University of Yaounde I
Faculty of Sciences
Cameroon

ATSA ETOUNDI Roger
University of Yaounde I
Faculty of Sciences
Cameroon

ABSTRACT
A group signature allows any member of a group to sign anonymously and to act on behalf of the group in such a way that only the associated group manager, in case of any dispute, is able to reveal the identity of the author of a transaction. The functioning of the group signature is based on a set of rules that are required to be met at any time by the associated scheme. The state-of-the-art of the group signature research has shown that several schemes have been proposed but none of them satisfy all the fundamental group signature properties such as liveliness. When some of these properties are satisfied, only informal proofs are available. Hence, one cannot claim that these properties are really supported by the associated schemes. This paper contributes in the definition of a group signature scheme by integrating all the required constraints and making it possible to carry out their formal proof. The specification of the scheme is based on the finite state automata techniques and the temporal logic PLTL of CTL* to formally specify the required properties. Finally, the model-checker SPIN is used to verify the consistency of the resulting scheme.

Keywords
Group Signature Scheme, Temporal Logic, Model Verification.

1. INTRODUCTION
The notion of Group signature was introduced by Chaum and Van Heyst [1]. This notion is a very useful tool in applications where the signer's privacy should be protected and in case of abuse an authority can identify the misbehaving user. However, a well-known group signature problem is that it is difficult for the group manager to identify a malicious user since all signatures are anonymous. The group manager obviously cannot afford to open all the group signatures signed, as by so doing, he would compromise the privacy of every signer. To tackle this problem, in practice many group signature schemes have been proposed. In this purpose, M. Abdalla and *al.* in [19] introduced the concept of unique signatures which is of great value from both theoretical and applied perspectives in this domain. Abe and al. in [20, 21] proposed group signatures with efficient concurrent join. A number of unique group signature schemes without random oracles under a variety of security models that extend the standard security models of ordinary group signatures are defined in [22]. Despite the volume of schemes defined in the literature, the formal proof of the associated group signature properties has not yet been carried out based on any of these schemes. This is due to the fact that a generic model of the group signature scheme has not been specified. The existing schemes have been presented based on specific cases. For this end, they do not consider the general aspect of the group signature problem. In order to carry out the proof of these properties, the paper argues that the scheme should first verify two more constraints. Therefore, a scheme obtained from the initial group signature theory cannot be used to formally prove the main properties of a group signature.

Formal specification of application software is becoming more and more common. Communicating finite state machines are suitable for modeling the basic group signature scheme. This motivated our choice of *Promela* as our specification language and *Spin* for checking large state spaces of the system. Although *Promela*'s expressive power is rather low and provides only a few basic data types and primitive constructor types, it offers the possibility of expressing reachability, safety and liveliness requirements by logical assertions or linear temporal logic expressions. The model is further animated by the Spin simulator and verified by the validator [6, 12] with different degrees of precision depending on the problem size. The properties are checked during the simulation or the verification step and if an error occurs it is possible to run the simulator again and look at each state to find the problem.

Only the fundamental group signature scheme of group signatures presented by J. Camenish and M. Stadler [4] informally satisfies most of the required properties of group signatures. A fully specified and verified group signature scheme could find application in a wide variety of fields where information security is a critical issue [5]; examples of information security sensitive domains include: electronic commerce, electronic bank transfer, video conferencing, etc. In this paper, we contribute to research in the area of group signatures by providing a formal specification using finite state automata and a formal verification of group signatures basic properties presented by Camenish and Stadler in [4], which so far have only been informally verified.

The rest of the paper is organized as follows. Section 2 identifies fundamentals of group signatures. Section 3 describes models for the different operations of group signatures using finite state automata techniques. Section 4 formally verifies some properties of our system and provides results generated by the model-checker Spin. Section 5 deals with the conclusion and highlighting some perspectives for future research.

2. FUNDAMENTALS OF GROUP SIGNATURES

A group signature scheme is a digital signature scheme comprised of the following operations [1, 9, 13, 14]:
SETUP: An algorithm for generating the group public key **y** and a group secret key **S**.
JOIN: A protocol between the group manager and a user that results in the user becoming a new group member.
SIGN: A protocol between a group member and a user whereby a group signature on a user supplied message is computed by the group member.
VERIFY: An algorithm for establishing the validity of a group signature giving a group public key and a signed message.
OPEN: An algorithm that, given a signed message and a group secret key, determines the identity of the signer.
According to [9, 10], a group signature scheme must satisfy the following security properties:
Unforgeability: Only group members are able to sign messages on behalf of the group.
Anonymity: Given a signature, identifying the actual signer is computationally hard for everyone except the group manager.
Traceability: The group manager is able to open a signature and identify the actual signer; moreover, a signer cannot prevent the opening of a valid signature.

2.1 Components and Operations

The main protagonist in group signatures schemes are:
- *the group manager*: it's a trusted personality who is in charge of managing the group.
- *group members*: members of a group who can carry out group operations such as signing messages on behalf of the group. After the initial set up of the group which is achieved by the group manager through the SETUP primitive, group members and group manager will undertake JOIN, SIGN, VERIFY and OPEN operations during their stay in the group.

3. OPERATIONS WITH ASSOCIATED AUTOMATA

The proposed model holds on proactive, active and reactive components. A detailed description of each one follows.

3.1 Operation 1 "SETUP"

The group manager computes the following values [9]:
• an RSA modulus **n** and two public exponents $\mathbf{e_1}$, $\mathbf{e_2} > \mathbf{1}$, such that $\mathbf{e_2}$ is relatively prime to **Φ(n),** whereas **Φ** is the Euler **Φ** function;
• two integer $\mathbf{f_1}, \mathbf{f_2} > \mathbf{1}$, whose $\mathbf{e_1}$ - roots and $\mathbf{e_2}$ - roots cannot be computed without knowing the factorization of **n**;
• a cyclic group $\mathbf{G} = \langle \mathbf{g} \rangle_{\mathbf{n}}$ of order **n** in which computing discrete logarithm is infeasible;
• an element $\mathbf{h} \in \mathbf{G}$ whose discrete logarithm to the base **g** must be unknown.
• his public key $\mathbf{y_R} = \mathbf{h}^{\rho}$ for a randomly chosen value $\rho \in Z_n$;
• the group's public key consist of $\mathbf{y} = (\mathbf{n}, \mathbf{e_1}, \mathbf{e_2}, \mathbf{f_1}, \mathbf{f_2}, \mathbf{G}, \mathbf{g}, \mathbf{h}, \mathbf{y_R})$, whereas **ρ** and the factorization of **n** remain the group manager's secret key (also called group secret key).

- **Associated automaton**

Figure 1 bellow provides the automaton that represents the operation.

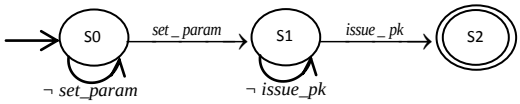


Fig 1: setup automaton: $\mathbf{A_S}$

- Transition elements are boolean variables: ***set_param; issuing_pk***. The symbol (¬) denote the negation.
- Atomic propositions

Here, we denote two atomic propositions.
S1: all components of the group public key have been registered (***set_param*** is set to true).
S2: the group public key has been issued (***issuing_ pk*** is set to true).

- Automaton's formal definition

We obtain the automaton $\mathbf{A_S} = (\mathbf{Q}, \mathbf{E}, \mathbf{T}, \mathbf{Q_0}, \lambda)$ defined as follows:

- the set of states is Q = {S0, S1, S2};
- the set of transition labels is E = {¬set_param; set_param; ¬issuing_pk; issuing_pk};
- the set of transitions is T = {(S0,¬set_param,S0); (S0,set_param,S1); (S1,¬issuing_pk,S1); (S1,issuing_pk, S2)}
- the initial state of automaton is Q_0 = S0
- the application λ which for all states of Q associates the set of elementary properties which is verified in this state is:

$$\lambda = \begin{cases} S0\ a & \varnothing \\ S1\ a & \{S1\} \\ S2\ a & \{S2\} \end{cases}$$

3.2 Operation 2 "JOIN"

Let's call Alice, the person who intends to join the group. Alice. Alice and the group manager both participate in this operation which comprises three steps presented below.

❖ ***first step:***

To become a group member, Alice computes
• $\gamma = r^{e_2}(f_1 y + f_2) \pmod n$ for $\mathbf{r} \in Z_n^*$
• U = E-SKROOTLOG $[\alpha : z = g^{\alpha^{e_1}}]$
• V = E-SKROOTLOG $[\beta : g^{\gamma} = (z^{f_1} g^{f_2})^{\beta^{e_2}}]$
and sends γ, z, U and V to the group manager.

❖ ***second step:***

The group manager verifies whether U and V are correct and sends to Alice the blinded certificate $\gamma = \gamma^{\frac{1}{e_2}} \pmod n$

❖ ***third step:***

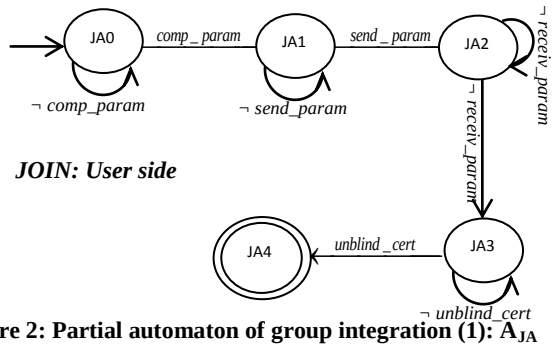
Alice unblinds this certificate and obtains her membership certificate $v = \gamma / r = (f_1 y + f_2)^{\frac{1}{e_2}} \pmod n$

- **Associated automaton**

We will distinguish two automata:
The first one describes the user's (Alice) actions and the second the group manager's actions.

✓ **Alice's case**

This is the automaton that represents the operation.



JOIN: User side

Figure 2: Partial automaton of group integration (1): A_{JA}

- Transition elements are boolean variables: ***comp_param***; ***send_param***; ***receiv_cert***; ***unblind_cert***.
- Atomic propositions

Here, we denote four atomic propositions.
JA1: Alice has computed her member's key (***comp_param*** is set to true).
JA2: Alice has sent a blinded version of her member key to the group manager (***send_param*** is set to true).
JA3: Alice has received a blinded version of her membership certificate from the group manager (***receiv_cert*** is set to true).
JA4: Alice has unblinded her membership certificate (***unblind_cert*** is set to true).

- Automaton's formal definition

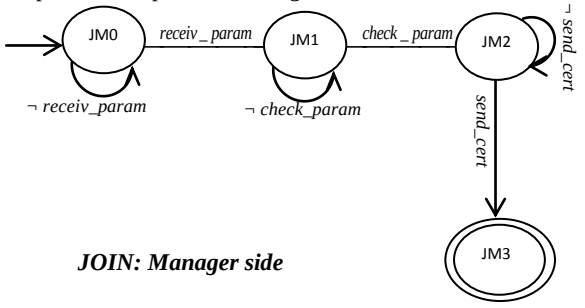
We obtain the automaton A_{JA} = (Q, E, T, Q_0, λ) defined as follows:

- the set of states is Q = {JA0, JA1, JA2, JA3, JA4}
- E = {¬comp_param; comp_param; ¬send_param; send_param; ¬receiv_cert; receiv_cert; ¬unblind cert; unblind_cert;}
- T={(JA0,¬comp_param,JA0); (JA0,comp_param,JA1); (JA1,¬send_param,JA1); (JA1,send_param,JA2); (JA2,¬receiv_cert,JA2); (JA2, receiv_cert,JA3); (JA3,¬unblind_cert,JA3); (JA3,unblind_cert, JA4)}
- the initial state of automaton is Q0 = JA0
- and then,

$$\lambda = \begin{cases} JA0 \text{ a } \varnothing \\ JA1 \text{ a } \{JA1\} \\ JA2 \text{ a } \{JA2\} \\ JA3 \text{ a } \{JA3\} \\ JA4 \text{ a } \{JA4\} \end{cases}$$

✓ **Group manager's case**

The automaton associated with the group manager's set of operations is presented in figure 3 below.



JOIN: Manager side

Fig 3: Partial automaton of group integration (2): A_{JM}

- Transition elements are boolean variables: receiv_param; check_param; send_cert.
- Atomic propositions

JM1: the group manager has received Alice's blinded member key (receiv_param is set to true).
JM2: Alice's signatures of knowledge are correct (check_param is set to true).
JM3: the group manager has sent a blinded version of Alice's membership certificate to her (send_cert is set to true).

- Automaton's formal definition

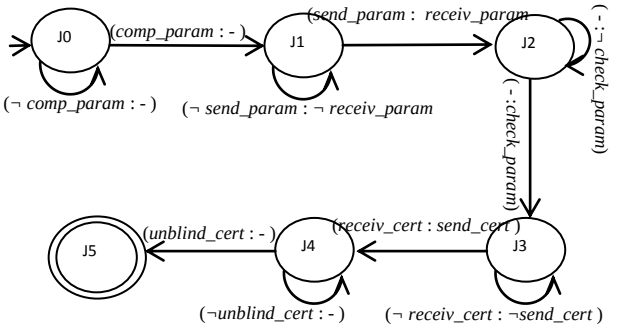
We obtain the automaton AJM = (Q, E, T, Q_0, λ) defined as follows:

- Q = {JM0, JM1, JM2, JM3}
- E={¬receiv_param; receiv_param; ¬check_param; check_param; ¬send_cert; send_cert}
- T={(JM0,¬receiv_param,JM0);(JM0,receiv_param,JM1);(JM1,¬check_param,JM1);(JM1,check_param,JM2);(JM2,¬send_cert,JM2);(JM2,send_cert,JM3)}
- the initial state of automaton is Q_0 = JM0
- and then,

$$\lambda = \begin{cases} JM0 \text{ a } \varnothing \\ JM1 \text{ a } \{JM1\} \\ JM2 \text{ a } \{JM2\} \\ JM3 \text{ a } \{JM3\} \end{cases}$$

3.2.1 Global automaton of "JOIN"

The interaction of the group manager and the user (Alice) in the JOIN primitive compels us to synchronize their automaton [2, 6]. The synchronized product is based on the send / receive message synchronization. In the transition labels, we have separated those which correspond to the send ("send") from those corresponding to the receive ("receive") of the same message.
The synchronization constraint [2] is reduced to the simultaneity of the send and the reception of messages. The set of synchronization is defined as follows:
Sync = {(comp_param, -); (send_param, receiv_param); (-, check_param); (send_cert, receiv_cert); (unblind_cert, -)}, where the symbol "-" means "*do nothing*". After building the cartesian product automaton [2, 3], we extract the synchronized product which is the final automaton of JOIN as shown in figure 4 below.



JOIN

Figure 4: Final automaton JOIN: $A_J = A_{JA} \otimes A_{JM}$

3.3 Scenario 3 "SIGN"

Only the group members intervene in this procedure. To sign a message on behalf of the group, Alice performs the following computations:

- $\not{2}\not{o} = h^r g^y$
- $d = y_R^r$

• V1 = E-SKROOTREP $[(\alpha,\beta): \text{illegible} = h^{\alpha} g^{\beta^{e_1}}](M')$

•V2=E-SKROOTREP $[(\gamma,\delta): \text{illegible} \cdot g^{f_2} = h^{\gamma} g^{\gamma^{e_2}}](M')$

•V3=E-SKREP $[(\varepsilon,\zeta): d = R^{\varepsilon} \wedge \text{illegible} = h^{\varepsilon} g^{\zeta}](M')$

The resulting signature on the message **M** consist of (illegible, d, V1, V2, V3) and it is valid if the three signatures of knowledge V1, V2 and V3 are correct.

- Associated automaton

The associated automaton is shown in figure 5.

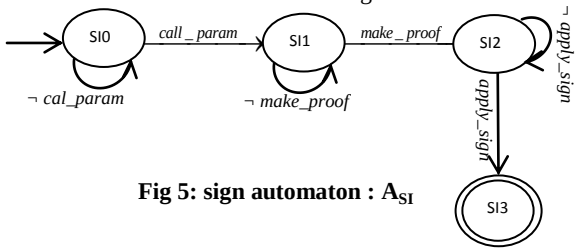


Fig 5: sign automaton : A_{SI}

- Transition elements are boolean variables: cal_param; make_proof; apply_sign.
- Atomic propositions

SI1: Alice has used the group manager's public key to compute the first part of the signature (cal param is set to true).
SI2: Alice has established signatures of knowledge (make_proof is set to true).
SI3: Alice has applied her signature on the massage (apply_sign is set to true).

- Automaton's formal definition

We obtain the automaton A_S = (Q, E, T, Q_0, λ) defined as follows:

- the set of states is Q ={SI0, SI1, SI2,SI3}
- E ={¬cal_param; cal_param; ¬make_proof; make_proof; ¬apply_sign; apply_sign}
- T = {(SI0,¬cal_param,SI0); (SI0,cal_param,SI1); (SI1,¬make_proof,SI1); (SI1,make_proof,SI2); (SI2,¬apply_sign,SI2); (SI2,apply sign,SI3)}
- the initial state of automaton is Q_0 = SI0
- and

$$\lambda = \begin{cases} SI0 \text{ a } \varnothing \\ SI1 \text{ a } \{SI1\} \\ SI2 \text{ a } \{SI2\} \\ SI3 \text{ a } \{SI3\} \end{cases}$$

3.4 Operation 4 "OPEN"

Only the group manager is involved here. Given a signature (illegible, d, v1, V2, V3) on a message M, he computes the following:

• $\hat{z} = \text{illegible} \cdot d^{1/w}$, which corresponds to the signer's membership key **z**.

• SKREP $[w: \text{illegible} = zd^{w} \wedge h = y_R^{w}](M')$, to prove that **z** is indeed encrypted in illegible and **d**.

• extracts in **z** the identity of Alice and reveals it.

- Associated automaton

Here is the associated automaton. (figure 6).

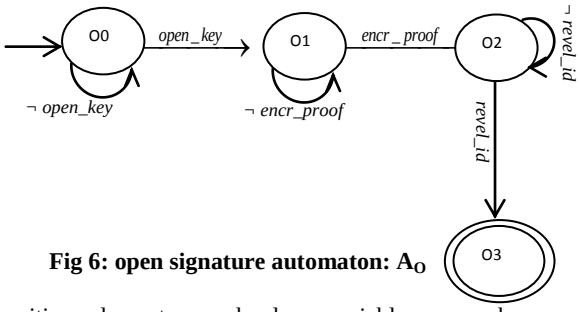


Fig 6: open signature automaton: A_O

- Transition elements are boolean variables: open_key; encr_proof; revel_id.
- Atomic propositions

Here, we denote two atomic propositions.
O1: the group manager has extracted the member key of the signer in the signature (open_key is set to true).
O2: the group manager has proven that the member key is indeed encrypted in the signature (encr_proof is set to true).
O3: the group manager has revealed the identity of the signer (revel_id is set to true).

- Automaton's formal definition

We obtain the automaton A_O = (Q, E, T, Q_0, λ) defined as follows:

- Q ={O0, O1,O2}
- E ={¬open_key; open_key; ¬encr_proof; encr_proof; ¬Revel_id; revel_id}
- T={(O0,¬open_key,O0); (O0,open_key,O1); (O1,¬encr_proof,O1); (O1,encr_proof,O2); (O2,¬revel_id,O2);(O2,revel_id,O3)}
- Q_0 = O0
- and,

$$\lambda = \begin{cases} O0 \text{ a } \varnothing \\ O1 \text{ a } \{O1\} \\ O2 \text{ a } \{O2\} \\ O3 \text{ a } \{O3\} \end{cases}$$

3.5 Operation 5 "VERIFY"

Only the group manager intervenes in this operation. He carries out the following tasks [3]:

• check V1 to convince himself that the signer knows the secret key.

• check V2 to convince himself that the signer knows the membership certificate associated to his secret key.

• check V3 to convince himself that the signature could be opened if it necessary.

- Associated automaton

This is the associated automaton (figure 7).

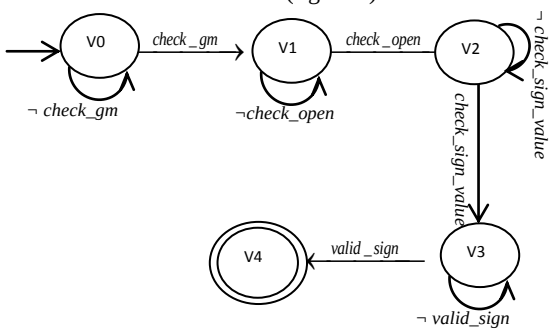


Figure 7: verify automaton: A_V

- Transition elements are boolean variables: check_gm; check_open; check_sign_value; valid_sign.
- Atomic propositions

V1: the group manager is convinced that the first signature of knowledge is correct (check_gm is set to true).
V2: the group manager is convinced that the signature could be opened (check_open is set to true).
V3: the group manager is convinced that the signer is a group member (check_sign value is set to true).
V4: the group manager has validated the signature (valid_sign is set to true).

- Automaton's formal definition

We obtain the automaton A_V = (Q, E, T, Q0, λ) defined as follows:

- Q = {V0, V1, V2, V3,V4}
- E ={¬check_gm; check_gm;¬check_open; check_open;¬check_sign_value; check_sign_value;¬valid_sign;valid_sign}
- T = {(V0,¬check_gm,V0); (V0,check_gm,V1); (V1,¬check_open,V1); (V1,check_open,V2); (V2,¬check_sign_value,(V2,check_sign_value,V3); (V3,¬valid_sign,V3); (V3,valid_sign,V4)}
- the initial state of automaton is Q_0 = V0
- and then,

$$\lambda = \begin{cases} V0\,\text{a}\ \varnothing \\ V1\,\text{a}\ \{V1\} \\ V2\,\text{a}\ \{V2\} \\ V3\,\text{a}\ \{V3\} \end{cases}$$

4. GLOBAL AUTOMATON OF THE SYSTEM

Given that the system is asynchronous, it was split into components that do not interact among themselves. The global automaton will therefore be the cartesian product of automata that represent the various components (SETUP, JOIN, SIGN, VERIFY, OPEN). Hence, a (global) state of the automaton is actually a vector of the various (locals) states of the components.

5. VERIFICATION

The following issues are consigned during verification: preventing major problems such as global deadlock of system, eliminating unspecified receptions and detecting non executable codes at a given moment. Properties verified by the system fall into three major groups presented below. Every property is first of all formalized using LTL (Linear Temporal Logic) of Spin.
The return of Spin is very verbose but rather discreet concerning the truthfulness of a formula. In figure 8 - 12, " never-claim + " in the right diagram indicate that the process of check was launched on the property stated in Formula As typed of the left diagram. Every time that one " - " replace one " + ", it means is that the property was not taken into account, in which case Spin will indicate that there is no error. The met number of errors is indicated to the end of check. If it indicates 0, it means that the property is the true. Otherwise, the shown value will indicate the number of errors. Futhermore, whenever the specication is not satisfied, the ModelChecker will produce a counterexample execution trace that shows why the specification does not hold.

5.1 Reachable properties

A reachable property indicates that a given situation can be reached [3, 6].
They are all atomic propositions that we have identified during the modeling process. To verify them, Spin ensures that states in which these properties are defined are really reachable. The pertinent ones are those which are associated with finals states of automata defined in the previous section.
In SETUP, we have the following proposition:
(S2) *"group public key has been issued"*. This proposition is checked by *assert (issuing pk=1)*.
Where ***assert (boolean variable)*** is a Spin primitive which stops the verification process whenever the Boolean variable is false.
In JOIN, we have the following proposition:
(J5) *"Alice has unblinded her member certificate"*. This proposition is checked by *assert (unblind cert=1)*.
In SIGN, the pertinent proposition is:
(SI3) *"Alice has signed the massage"*. The proposition is checked by *assert (apply sign=1)*.
In VERIFY, we have:
(V4) *"The group manager has validated the signature on the massage"*. This proposition is checked by *assert (valid sign=1)*.
In OPEN, we have:
(O3) *"The group manager has revealed the identity of the signer"*. This proposition is checked by *assert (reveal_id=1)*.

5.2 Safety properties

A safety property means that, under certains conditions, something cannot take place [6]. Examples include *anonymity* and *unforgeability*.

5.2.1 Anonymity

"a group member cannot verify or open a signature on a message". This is formalized in PLTL (Propositional Linear Temporal Logic) as follows:

$$G(member \rightarrow \neg(verify \vee open))$$

The associated automaton generated by Promela in order to run the verification (called *never claim*) is on the left side. (See figure 8)
One can observe in the verification results (on the right side) the large state spaces of the system which had been checked without any error (errors:0).

```
#define member   (_pid > 1 && _pid < 6)
#define verify    (verifie == MESS_VERIFIED)
#define open    (ouvre == MESS_OPENED)
/*
 * Formula As Typed: []  (member -> ! (verify || open))
 * The Never Claim Below Corresponds
 * To The Negated Formula ![]  (member -> ! (verify || open)))
 * (formalizing violations of the original)
*/
never {    /* ![]  (member -> ! (verify || open))) */
T0_init:
if
:: ((((member) && (open)) || ((member) && (verify)))) -> goto accept_all
:: (1) -> goto T0_init
fi;
accept_all:
skip
}
```

```
#ifdef RESULT
(Spin Version 3.5.0)
Bit statespace search for:
never-claim        +
assertion violations+ (if within scope of claim)
acceptance   cycles + (fairness disabled)
invalid endstates- (disabled by never-claim)

State-vector 208 byte, depth reached 1953079, errors: 0
    748 states, stored
3.90609e+06 states, matched
3.90684e+06 transitions (= stored+matched)
4.88195e+06 atomic steps
hash factor: 1399.97 (expected coverage: >= 99.9% on avg.)
(max size 2^20 states)

268.426memory usage (Mbyte)

144.01user 3.11system 3:34.50elapsed 68%CPU (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (13516major+65537minor)pagefaults 0swaps
#endif
```

Figure 8: PROPERTY (a): $G(member \rightarrow \neg(verify \vee open))$

5.2.2 *Unforgeability*

"any person who signs a message had previously been registered before as a group member". His PLTL formalization is:

$G(sign \rightarrow F^{-1} join)$ which is equivalent to $(\neg sign \cup join) \vee G\neg sign)$

The *never claim* associated and the verification results are presented in figure 9.

5.3 Liveliness properties

A liveliness property means that, under certains conditions, something will eventually occur [10]. Examples include *belonging*, *traceability* and *feasibility*.

```
#define member    (_pid > 1 && _pid < 6)
#define join   (memcert == MEMCERT)
#define sign   (signe == MESS_SIGNED)
/*
 * Formula As Typed: ( ! sign U join) || []   ! sign
 * The Never Claim Below Corresponds
 * To The Negated Formula !(( ! sign U join) || []   ! sign)
 * (formalizing violations of the original)
 */
never {    /* !(( ! sign U join) || []   ! sign) */
T0_init:
if
:: (! ((join)) && (sign)) -> goto accept_S4
:: (! ((join)) && (sign)) -> goto accept_all
:: (! ((join))) -> goto T0_init
:: (! ((join)) && (sign)) -> goto accept_S13
fi;
accept_S4:
if
:: (! ((join))) -> goto accept_S4
:: (! ((join)) && (sign)) -> goto accept_all
fi;
accept_S13:
if
:: ((sign)) -> goto accept_all
:: (1) -> goto T0_S13
fi;
T0_S13:
if
:: ((sign)) -> goto accept_all
:: (1) -> goto T0_S13
fi;
accept_all:
skip
}
```

```
(Spin Version 3.5.0)

Bit statespace search for:
never-claim        +
assertion violations+ (if within scope of claim)
acceptance   cycles + (fairness disabled)
invalid endstates- (disabled by never-claim)

State-vector 208 byte, depth reached 349, errors: 0
  120199 states, stored
  229172 states, matched
  349371 transitions (= stored+matched)
  410225  atomic steps
hash factor: 8.72359 (best coverage if >100)
(max size 2^20 states)

Stats on memory usage (in Megabytes):
25.963equivalent memory usage for states (stored*(State-vector + overhead))
0.131 memory used for hash-array (-w20)
0.524 memory used for bit stack
252.000memory used for DFS stack (-m9000000)
256.854total actual memory usage

14.66user 2.13system 1:13.64elapsed 22%CPU (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (162major+62689minor)pagefaults 0swaps
```

Figure 9: PROPERTY (b): $(\neg sign \cup join) \vee G\neg sign)$

5.3.1 *Belonging*

"one cannot be a group member until he runs join" is formalized in PLTL as follows:

$\neg\ member\ W\ join$

Its equivalence is $\neg member \cup join \vee G\neg member$

Here is the associated *never claim* (figure 10).

```
#define member   (_pid > 1 && _pid < 6)
#define join   (memcert == MEMCERT)
/*
 * Formula As Typed: ( ! member U join) || ! member
 * The Never Claim Below Corresponds
 * To The Negated Formula !(( ! member U join) || ! member)
 * (formalizing violations of the original)
 */
never {    /* !(( ! member U join) || ! member) */
accept_init:
T0_init:
if
:: (! ((join)) && (member)) -> goto accept_S3
:: (! ((join)) && (member)) -> goto accept_all
fi;
accept_S3:
T0_S3:
if
:: (! ((join))) -> goto accept_S3
:: (! ((join)) && (member)) -> goto accept_all
fi;
accept_all:
skip
}
```

```
(Spin Version 3.5.0 )
+ Partial Order Reduction

Full statespace search for:
never-claim         +
assertion violations+ (if within scope of claim)
acceptance   cycles + (fairness disabled)
invalid endstates- (disabled by never-claim)

State-vector 104 byte, depth reached 2547, errors: 0
      178 states, stored
     1457 states, matched
     2678 transitions (= stored+matched)
     2724 atomic steps
hash conflicts: 187 (resolved)
(max size 2^20 states)

256.399memory usage (Mbyte)

0.19user 2.00system 0:59.30elapsed 57%CPU (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (156major+62575minor)pagefaults 0swaps
```

Figure 10: PROPERTY (c): $\neg member \cup join \vee G\neg member$

5.3.2 Traceability

"the group manager can sign messages, verify and open signatures on the messages". In PLTL, we have:

$(manager \rightarrow G(sign \vee verify \vee open)$

The associated never claim is presented in figure 11.

5.3.3 Feasibility

"any group member can sign a message on behalf of the group" is translated in PLTL by:

$member \rightarrow \mathbf{F}$ sign

The associated never claim is presented in figure 12.

```
#define member   (_pid > 1 && _pid < 6)
#define join   (memcert == MEMCERT)
#define manager   (_pid == 1)
#define sign   (signe == MESS_SIGNED)
#define verify   ( verifie == MESS_VERIFIED)
#define open   (ouvre == MESS_OPENED)
/*
 * Formula As Typed: manager -> <>  (sign || verify || open)
 * The Never Claim Below Corresponds
 * To The Negated Formula !(manager -> <>  (sign || verify || open))
 * (formalizing violations of the original)
 */
never {    /* !(manager -> <>  (sign || verify || open)) */
accept_init:
T0_init:
if
:: (! ((open)) && ! ((sign)) && ! ((verify)) && (manager)) -> goto accept_S3
fi;
accept_S3:
T0_S3:
if
:: (! ((open)) && ! ((sign)) && ! ((verify))) -> goto accept_S3
fi;
}
```

```
#ifdef RESULT
(Spin Version 3.5.0)
Bit statespace search for:
never-claim         +
assertion violations+ (if within scope of claim)
acceptance   cycles + (fairness disabled)
invalid endstates- (disabled by never-claim)

State-vector 208 byte, depth reached 1995379, errors: 0
     820 states, stored
4.00609e+06 states, matched
4.01684e+06 transitions (= stored+matched)
5.91195e+06 atomic steps
hash factor: 14401.97 (expected coverage: >= 99.9% on avg.)
(max size 2^20 states)

298.321memory usage (Mbyte)

156.01user 3.43system 4:35.50elapsed 77%CPU (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (13516major+65537minor)pagefaults 0swaps
#endif
```

Figure 11: PROPERTY (d): $(manager \rightarrow G(sign \vee verify \vee open)$

```
#define member   (_pid > 1 && _pid < 6)
#define sign   (signe == MESS_SIGNED)
/*
 * Formula As Typed: member -> <>  sign
 * The Never Claim Below Corresponds
 * To The Negated Formula !(member -> <>  sign)
 * (formalizing violations of the original)
 */
never {    /* !(member -> <>  sign) */
accept_init:
T0_init:
if
:: (! ((sign)) && (member)) -> goto accept_S3
fi;
accept_S3:
T0_S3:
if
:: (! ((sign))) -> goto accept_S3
fi;
}
```

```
#ifdef RESULT
(Spin Version 3.5.0 )
Bit statespace search for:
never-claim         +
assertion violations+ (if within scope of claim)
acceptance   cycles + (fairness disabled)
invalid endstates- (disabled by never-claim)

State-vector 104 byte, depth reached 225, errors: 0
     405 states, stored
598609 states, matched
570664 transitions (= stored+matched)
675224 atomic steps
hash factor: 2951.4 (expected coverage: >= 99.9% on avg.)
(max size 2^20 states)

173.240 memory usage (Mbyte)

130.01user 3.51system 2:25.50elapsed 75%CPU (0avgtext+0avgdata 0maxresident)k
0inputs+0outputs (1296major+65596minor)pagefaults 0swaps
#endif
```

Figure 12: PROPERTY (e): $member \rightarrow \mathbf{F}$ **sign**

6. CONCLUSION AND FUTURE WORK

Much work has been done in the group signature field leading to the development of a great number of group signature schemes. However, none of these schemes allows for the carrying out of formal proof of the associated group properties. The defined schemes have been modeled based on specific case studies and do not really take into consideration all the initial outlines properties. In this paper, we have presented a formal model of group signatures specified using finite state automata in order to contribute in the resolution of this difficulty. The proof process has been done incrementally; first we have added and proved two salient properties that are used in verifying the main properties of the group signature defined by D. Chaum and al. The Linear Temporal Logic PLTL which asserts how the behavior of the system evolves over time has been used. Secondly, the SPIN model checker was later used to check whether the abstract model satisfies the properties identified and formalized using PLTL temporal logic. These properties include: reachable properties, safety properties and liveliness properties. By applying each property in turn as well as the model of the system to the SPIN model-checker, we formally verified that the defined model satisfied all the properties listed by D. Chaum. During the verification process, liveliness properties such as "belonging" and "feasibility" impose themselves as conditions of the achievement of the "traceability" required to be one of the fundamental group signature properties. As future work, we plan to implement the defined group signature scheme, and extend the model in order to carry out verification of a group signature scheme in which the group management is shared among its members such that every member is involved in all management transactions.

7. REFERENCES

[1] D. Chaum: "Blind signature systems". In advances in cryptology-CRYPTO, 1983, pages 153. Plenum Press, 1984.

[2] G. Berry, P. Couronne and G. Gonthier: "Synchronous Programming of Reactive Systems: An introduction to ESTEREL". Proc. 1st Franco-Japanese Symp. on programming of Future Generation Computers, 1986, Tokyo. pp.35-56.

[3] W. Thomas: "Automata on Infinite Objects". Handbook on Theoretical Computer Science, J. Van Leeuwen, ed, pp. 133-187, Elsevier Science, 1990.

[4] D. Chaum and E. Van Heyst: "Group signatures". In advances in cryptology-EUROCRYPT 1991, volume 547 of Lecture notes in computer Science, pages 257-265. Springer-Verlag, 1991.

[5] G. J. Holzmann: "Design and Validation of Computers Protocols". Englewood Cli s, N.J. Prentice Hall, 1991.

[6] Z. Manna, A. Pnueli: "The Temporal Logic of Reactive and Concurrent Systems Programs". Springer-Verlag, 1991.

[7] G. J. Holzmann: "Basic Spin Manual". AT&Bell Laboratories, Murray Hill, New Jersey. 1995.

[8] J. Magnier: "Représentation symbolique et Vérification formelle de machines séquentielles". State Thesis University of Montpellier II, France, 1996.

[9] J. Camenisch: "Efficient and generalized group signatures". In advances in cryptology-EUROCRYPT'97, volume 1233 of Lecture notes in computer Science, pages 465-479. Springer-Verlag, 1997.

[10] Jan Camenisch and Markus Stadler: "Efficient Group Signatures Schemes for large Groups". In advances in cryptology-CRYPTO, 1997, volume 1294 of Lecture notes in computer Science, pages 410-424. Springer-Verlag, 1997.

[11] B. Berard, M. Bidoit, F. Laroussinie, A. Petit, P. Schnoebelen: "Vérification de logiciels – Techniques et outils du model-checking". Vuibert, Paris, 1999.

[12] G. Ateniese and G. Tsudik: "Some Open Issues and New Directions in Group Signatures". In cryptography'99, 1999.

[13] Dan Boneh, Xavier BoyenetHovav Shacham: "Short Group Signatures" Advances in Cryptology – CRYPTO 2004, Volume 3152/2004, 227-242

[14] Jan Camenische and Jens Groth: "Group Signatures: Better Efficiency and New Theoretical Aspects" Security in Communication Networks, 2005, Volume 3352/2005, 120-133.

[15] Jens Groth: "Fully Anonymous Group Signatures without Random Oracles" Advances in Cryptology – ASIACRYPT, Volume 4833/2007, 164-180.

[16] Xiaojun Wen, Yuan Tian, Liping Ji1 and Xiamu Niu "A group signature scheme based on quantum teleportation" Physica Scripta; Volume 81; Number 5; May 2010.

[17] S. Dov Gordon and Jonathan Katz and Vinod Vaikuntanathan "A Group Signature Scheme from Lattice Assumptions" Cryptology ePrint Archive, Version: 20110208:170219, last revised 8 Feb 2011.

[18] G. J. Holzmann: "Basic Spin Manual". AT&Bell Laboratories, Murray Hill, New Jersey.

[19] M. Abdalla, D. Catalano, and D. Fiore. "Veritable random functions from identity-based key encapsulation. EUROCRYPT 2009, LNCS vol. 5479, Springer, pp. 554-571, 2009.

[20] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. "Structure-perserving signatures and commitments to group elements". CRYPTO 2010, LNCS vol. 6223, Springer, pp. 209-236, 2010.

[21] M. Abe, J. Groth, K. Haralambiev, and M. Ohkubo. "Optimal structure-preserving signatures in asymmetric bilinear groups". CRYPTO 2011, LNCS vol. 6841, Springer, pp. 649-666, 2011.

[22] Matthew Franklin, Haibin Zhang "Unique Group Signatures", ESORICS 2012.

II

Atsa Etoundi Roger, **Mboupda Moyo Achille**: " **MultiPerspective Cybercrime Investigation Process Modeling** ", International Journal of Applied Information Systems (IJAIS) - ISSN : 2249-0868, Foundation of Computer Science FCS, New York, USA, Volume 2- No.2, June 2012 - www.ijais.org.

Indexed with CiteSeer, NASA Hardvard ADS, DOAJ, Google Scholar, DBLP, Scientific Commons, Informatics, ProQuest CSA Technology Research Database.

JUIN 2012

International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 2– No.2, June 2012 – www.ijais.org

Multi-Perspective Cybercrime Investigation Process Modeling

ATSA ETOUNDI Roger
University of Yaounde I
Faculty of Sciences
Cameroon

MBOUPDA MOYO Achille
University of Yaounde I
Faculty of Sciences
Cameroon

ABSTRACT
Several works have been carried out in the domain of cybercrime investigation. Each of the resulting models is based on a set of activities that should be performed in order to obtain the required evidences that are needed in the court for prosecution. In the literature, three processes have been highlighted for the digital forensic investigation based on a current situation; they include proactive, active and reactive processes. However, none of the defined approaches for investigation has taken into consideration the three perspectives despite the fact that they are linked together in the management of cybercrime within an organization. Moreover, there is no agreement in the definition of different tasks to be performed for each process in the achievement of the associated investigation goal. Each researcher comes with a specific set of activities based on the case under studies. In the same manner, the ordering of activities for a given process is not clearly specified; therefore, in different cases using the same process with the same activities, the associated executions are sometimes very different. There is a lack of standards in the cybercrime investigation processes. As the cybercrime increases in the modern society based on the use and the growth of ICTs (Information and Communication Technologies), there is an urgent need to set up a standard which takes into account the above issues. This paper proposes a multi-perspective cybercrime investigation process modeling that can be considered as a basis for standardization. The proposed model is constructed by extending and unifying the existing approaches.

Keywords
Computer forensics, Cybercrime investigation, Forensic process models, Proactive forensic investigation, Active and Reactive forensic investigation.

1. INTRODUCTION
The use of computer systems todays either as an object of crime, an instrument used to commit a crime or a storehouse of evidence related to a crime has led to the emergence of computer forensics [1]. When an infraction implies an information technology support, a fast and methodical intervention must be committed to bound and protect the crime scene. These procedures are indispensable, before any analysis, to guarantee the control and the admissibility of the evidence collected and treated in case of legal proceedings. The development of many forensic investigation models by countries and organizations is the obvious consequence of the spread of digital crime and its rising rate of improvement all over the world. The technological aspect of investigation is the goal of the ones, and data analysis part for the others [3]. Very few of the existing models take into account a proactive or active digital investigation process. Whenever conducting a computer investigation for potential criminal violations of the law, the legal process only depends on local cyber law. Anyway, the investigation is made after a complaint and the prosecution follows. The increasing rate of computer crimes is a fact. For example: the use of anti-forensic methods as the "Zeus Botnet Crime-ware toolkit" that has the ability to counteract digital forensic investigations with its obfuscation levels. Thus, some specific types of proactive investigation methods or systems are required viewing the volatility and the dynamicity of the information flow in such a toolkit. In this paper, we present a brief overview of previous forensic models and propose a model which explores the different processes involved in the investigation of cybercrime in case of proactive, active and reactive forensic approaches. Furthermore, the model can be used in a proactive way to identify opportunities for the development and deployment of technology to support the work of investigators, and to provide a framework to capture and analyze the requirements for investigative tools, particularly for advanced automated analytical tools. At present, there is a lack of general models specifically directed at cybercrime investigations. The current models focus on part of the investigative process (that concerns gathering, analyzing and presenting evidence) but a fully general model must integrate other important aspects to be comprehensive.
The rest of the paper is organized as follows. Section 2 outlines the previous works. Section 3 describes the proposed model. Section 4 lays out a mapping between the proposed model and the previous ones. Section 4 deals with the conclusion and highlighting some perspectives as future works.

2. PREVIOUS WORKS
Although several models exist within the digital forensic industries, there are essentially limited to reactive digital forensics. The investigation of the crime scene and the evidence are the main targets of most of the existing models; as such, their sphere of action is very tight [4]. Other models have tended to concentrate on the middle part of the process of investigation, i.e. the collection and examination of the evidence. However, the earlier and later stages must be taken into account if a comprehensive model is to be achieved, and in particular if all the relevant information flowing through an investigation is to be identified.
In accordance with the documentation, very few writings have considered a proactive digital forensic investigation process. Some of these writings have mentioned very clearly the proactive process, while the others did not. However, all have

revealed that such a process is essential. In [10], CP *Grobler and al.* proposed a high-level framework that consider three components, Proactive DF (the proactive reorganization and describing of processes, procedures and technologies in order to generate, accumulate, preserve and manage exploitable digital evidence so as to facilitate a successful, cost effective investigation, with minimal interruption of business activities), Active DF or "live forensics" (the skill of an organization to easily identify, collect, and preserve exploitable digital evidence in a live environment so as to enable a prosperous investigation) and Reactive DF or "dead forensics" (the methodical and exploratory practices used for the identification, extraction, preservation, documentation, analysis, and interpretation of digital media stored or encoded for evidence issue).

In [11], *Soltan Alharbi and al.* proposes a digital forensic model with two components : Proactive and Reactive Digital forensic investigation. The proactive component deals with the collection of live evidence in real time while an event or incident is happening. The reactive component is the traditional approach to digital forensic investigation.

A multi-component view of the digital forensic investigation process proposed in [10] is a high-level view of the investigation and, as such, cannot directly be operationalized to create automated tools. Additionally, the process described in this model contains phases, such as service restoration, that lie outside the scope of the investigation. So doing, the investigator roles and those of the incident response team of an organization are completely confused. Furthermore, the proactive digital forensic aspect of the multicomponent view proposed by *Grobler and al.* only focuses on the readiness.

Proactive and Reactive Digital forensic investigation process proposed in [11] do not take into account the eventuality of an attack during the investigations. This model entirely ignores the readiness in its reactive component in the case that an organization is not able to set up proactive or anti-forensic measures. Furthermore, there is a misplacement of the *event triggering function* which involves abusive storage in proactive collection step.

All these breaches lead us to set up a multiperspective model of investigation which automation (future works) would lead to an efficient conditioning of the digital evidence ready to be received in a court of law.

3. THE PROPOSED MODEL

The proposed model holds on proactive, active and reactive components. A detailed description of each one follows.

3.1 Proactive digital forensics (ProDF) component

Proactive Digital Forensics is the proactive reorganization and describing of processes, procedures and technologies in order to generate, accumulate, preserve and manage exploitable digital evidence so as to facilitate a successful, cost effective investigation, with minimal interruption of business activities. That forensics disposition must be establish so as to involve negligible business activity interruption. It will ensure that admissible evidence and sound processes are in place and available when needed for an investigation or as required during the normal flow of business. Requirements for admissible evidence are specific to the country, the jurisdiction and the industry. The success of any investigation is determined by the credibility of the evidence [10]. Therefore, Proactive Digital Forensic Component is the ability to proactively identify, collect, gather an event, preserve and analyze evidence to detect an incident as it eventually occurs. Furthermore, a programmed documentation is produced for a later investigation by the active and reactive components [11]. The evidence contained within this component is proactive evidence that leads to a specific event or incident as it happens [12].

This proactive component differs from common Intrusion Detection Systems (IDS) by safeguarding the integrity of evidence and preserving it in a forensically sound manner, while IDS only detects an intrusion as it happens and be able to respond to it. In addition, the analysis of the evidence will be done so as to be admitted in a court and enable prosecution of the suspect.

Phases under the proactive component are defined as follows:

1. **Alert**: According to the local cyber law, the incidence response team of any organization should display an alert system considering a set of cataloged incident.
2. **Identification**: Identified data in the order of volatility and priority, and related to a specific requirement of an organization. Hostile and outstanding behaviors are also selected and cataloged.
3. **Collection**: Programmed live collection of identified data and unusual behaviors. This can be done by a trigger event that will start live data collection as soon as certain types of incident alerts are detected.
4. **Preservation**: Automated preservation of the evidence related to the suspicious event, via hashing methods.
5. **Analysis**: Automated live analysis of the evidence, which might use forensic techniques such as data mining to support and construct the initial hypothesis of the incident.
6. **Documentation**: Automated report for the proactive component.

3.2 Reactive digital forensics (ReaDF) component

None of existing organizations are fully prepared for incidents. Reactive Digital Forensics as defined by this paper focuses on the traditional DF investigation that will take place after an incident had been detected and confirmed. This includes physical investigation (eventually), identifying, preserving, collecting, analyzing, and generating the final report. Should an incident happen, there should be an acceptable proven DF investigation protocol in place as detailed by ProDF on how to conduct the investigation [10]. Reactive evidence refers to collecting all the static evidence remaining, such as an image of a hard drive. The goals of ReaDF are to [11]:

- Successfully investigate an incident (offences);
- Collecting evidence;
- Determine the root-cause of the incident;
- Identify and link the perpetrator and accomplices to the incident.

As result of various DF methodologies or investigation protocols studied from literature, many authors have proposed the inclusion of the following phases with steps [1-9].

1. **Complaint/Alert/Individual detection**: Detect an incident, activity or offences; report the incident; check Readiness [6]; determine the assessment of worth, incident confirmation; obtain an authorization; obtain search warrant; determine a containment strategy; formulate an investigation plan; coordinate the resources; accelerate the investigation; notification of the investigation.

2. **Physical Investigation**: It is absolutely necessary to take in account the physical crime scene to gather as much evidence as possible so as to ensure successful investigation, even if it is a DF investigation. Steps involve the security of the physical crime scene; survey of crime scene for potential evidence; examine witness, search and collect; documentation (label and seal all evidence); acquire; analyze; identify possible digital evidence; reconstruct the event; make a finding; transport and store the evidence.
3. **Digital Investigation**: The success of the investigation is determined by the essential steps followed during this phase. The steps are:

Evidence acquisition: This step consist of identification and seizure, preservation and collection of evidence; acquirement of the relevant evidence if live evidence is required, activation of the ActDF component; ensure integrity; authentication; transportation of the evidence; storage of the evidence; and documentation of the acquisition process.

Identification: The investigator is mostly interested by the process of identifying all the electronic equipment used by the suspect. Additional imperative sub-procedure will be identifying fragile or volatile evidence. Live acquisition process can be set up to retrieve the file time stamp, registry key, swap files, and memory details. All what is mentioned above falls under the fragile evidence category as most probably will change or will be lost upon the plugging of the power cable. The investigators must try to obtain the maximum information from the various people present in the scene, without violating the jurisdictional laws and corporate policies. Everything surrounding the incident (those who reported it at first, the date and the time) has to be taken in account. It is important to make a list of those present at the scene, those who came after, and those who left, etc., at the same time with the summary of their activities while at the scene. It is very important to put people per group (victims, suspects, bystanders, witnesses, other assisting personnel etc..) and store their location by the time they entered.

Preservation: All the evidence collected has to be located in a safe place as it is important that the evidence is safe from tempering and need to preserve the integrity of the evidence. Thus, cybercrime investigators establish "police line" to protect evidence of damaged cybercrime scene. They set cybercrime evidences collection equipment such as Encase, imaging devices in the victim scenes. Evidences in the victim scenes can also be obtain through photo evidences by digital or video camera. Thus, all the electronic devices at the scene must be photographed along with the power adaptors, cables, cradles and other accessories. What is appearing on the screen should also be documented if the digital or mobile device is unpowered. A record of all visible data must be created, which helps in recreating the scene and reviewing it any time. This is mainly important when the investigator has to do a testimony in a court, which could be several months after the investigation.

Collection: The investigating organization takes possession of the evidence in a form which can be preserved and analyzed in the collection step. So doing, some activity of the investigator can consist of imaging hard disks or seizure of entire computers.

- ***Collect Volatile Evidence***: Most of the evidence concerning mobile devices are volatile, because they are present in Random Access Memory. Because the device state and memory contents can be changed, the collection of volatile evidence presents a serious problem. The choice of collecting evidence at the crime scene or after in a secure forensics workshop will lie on the specific situation, taking in account the ongoing power state. If the device is running out of battery power, the entire information will be lost soon. In that case, adequate power needs to be maintained if possible by using the power adaptor or replacing batteries. In case the battery power is not sure, the memory contents should be copied using suitable tools as fast as possible. To obtain better results, it is suitable to combine tools. A suitable power supply ought to be maintained through recharging or replacing the battery. In case sufficient power is not possible to be provided, the device must be switched off to save battery and the content of the memory. At this stage, the presence of any malicious software installed by the user has to be checked too.
- ***Obtain the evidences of storage media***: This stage deals with collecting evidence from external storage media lying on devices like MMC cards, compact flash (CF) cards, memory sticks, secure digital (SD) cards, USB memory sticks etc. Evidence from computers, which are synchronized with these devices, must be collected. Appropriate forensic tools must be used for collecting evidence to ensure its admissibility in a court of law. The integrity and authenticity of the evidence collected should be ensured through techniques like hashing, write protection etc. It is important to collect all power cables, adaptors, cradle and other accessories. Care should also be taken to look for evidence of non-electronic nature, like written passwords, hardware and software manuals and related documents, computer printouts etc.
- ***Obtain the evidences of network [10][15]***: This consist of capturing, recording, analyzing network audit trails in order to discover the source of security breaches or other information assurance problems. Not all the information captured or recorded will be useful for analysis. Network forensic systems are mainly classified as "Catch it as you can" and "Stop, look and listen" systems. Most network forensic systems are based on audit trails. Network forensic products are sometimes known as Network Forensic Analysis Tools (NFATs). For example, nstreams, slogdump, tcpflow, chaosreader, dhcpdump, etc. This stage involves proper documentation of the crime scene along with photographing, sketching and crime-scene mapping.

Analysis: The investigative team will revisit the investigation plan; review the relevance of tools and expertise available; develop the hypothesis; analyze the evidence; test the hypothesis; make a finding; validate the results of analysis; document the case; and secure the documentation.

4. **Reconstitution**: The physical investigation phase (1) and digital investigation phase (2) findings will be strengthened by the investigation team in this stage. In the case of lack of evidence to support the hypothesis, repeat phase (1) and (2). A well documented report in approving the hypothesis is the main aim of this phase.
5. **Present findings to management or authorities**: Legal jurisdiction location requirements are taken in account by the investigation team while preparing the case; the timeline of the whole case is to be incorporated; the target audience has to be determined; expert witness should be prepared; exhibits have to be prepared; suitable presentation aids should be used; and the chain of custody

and the evidence should be preserved while presenting the case. The protocol must also involve an appeal process.

6. **Dissemination of results of investigations**: It is very important to review the result of the identified case an apply lessons learned from it. All evidence ought to be preserved, returned or disposed, in accordance with the policies.
7. **Incident closure**: A waterfall framework seems to characterize all identified phases for ReaDF with some eventual repetition (whenever needed) between the different phases. ReaDF as discussed meets the need to investigate incidents to reveal the root-cause of an incident and successfully prosecute a perpetrator.
8. **Final report**: Documentation is a permanent loop back needed in all the stages of investigation and required to maintain proper chain of custody. A final report is produced at the end of investigations.

3.3 Active digital forensics (ActDF) component

ActDF is the skill of an organization to easily identify, collect, and preserve exploitable digital evidence in a live environment so as to enable a prosperous investigation. Eventually based on the documentation powered by the Proactive Digital Forensic, the goals for ActDF are [10]:

- Reduce the effect and impact of an ongoing incident;
- Collect relevant live receivable digital evidence (including volatile evidence) on a live system or production environment by using appropriate tools and technologies;
- Provide a meaningful starting point for a reactive investigation within the parameters of the risk control framework of the organization.

We have identified the following phases for Active Digital Forensics from the literature [13-16] and have formulated the following phases independent of any tool or technology.

1. **Complaint / Alert / Individual detection**: Once the investigators are on the scene, they have to rapidly identify volatile or live evidence and determining which of them have to be preserved to successfully investigate the incident or to acquire potential missing evidence for a new or unknown incident; From this point, an investigation plan is supposed to be set up. The management plan of the organization is to be taken into consideration. Therefore, Reactive DF must be activated whenever the risk management policies of the organization did not allow the continuity of the current Active DF evidence acquisition. It important for an organization to have a trigger event because a predefined one is supposed to be activated to release monitoring or other procedures as soon as an incident alert is activated. At the end, a containment strategy must be deployed. This is very important as ActDF deals with ongoing or real time incidents.
2. **Active investigation**:

Evidence acquisition: (Part 3 of ReaDF applies). All the remaining evidence that can be exploited or used as proof of any type of dis-functioning should be collected with the use of appropriate tools. In fact, some specific technologies or applications are required to profile the attacker, gather volatile evidence or to determine the source of the attack. Those applications can also serve to collect additional live evidence lacking from, or required by the exploitable digital evidence map. The next act is to secure and authenticate all the extracted data. This is easily done by hashing immediately after the collecting process to preserve before analysis. To ensure that chain of custody of the evidence acquired was maintained, all performed actions must be documented. There exists several applications capable of collecting digital evidences. A good technology is to be chosen in order to automatically collect such information. Some authors like *Ieong and Leung* [15] hold that human intervention in the collection of evidence should be reduced to the lowest rate. They affirm that all what is necessary is to be done by human intervention be reduced at all course. The static digital evidence is to be recorded without any modifications. In all, according to these authors, data acquisition should follow the order of volatility and priority of digital evidence collection; the other evidence which is not priority can be acquired using traditional evidence collection methods. The extraction of the data obtained can only be done when the original data has not been altered or corrupted. [4].

Analysis: (Part 3 of ReaDF applies). The primary acquires evidences are to be analyzed in order to determine whether it constitutes sufficient information for the reconstruction of the incident or it is in line with the initial hypothesis. It is of great importance to document progressively all the activities that are being carried out. That is the different task realized, actions that are undertaken with the aim assuring that the evidences are acquired completely. The regularity and the reliability of the results are to be assured. The computer on which the operating system is still running should be totally examined by using custom forensic technics or existing system administrator tools for the extraction of evidences. This practice is suitable when we have to do with encrypting file systems for example, where the encryption keys has to be collected and, in some instances, an image of the logical hard drive volume should be realized before turning off the computer.

Reconstitution: Here, results from the analysis should be exploited and one can proceed to a brief reconstruction of the incident. The prime objective of this phase is to establish whether sufficient evidence have been assembled so as to put an end to ActDF. However, ActDF can be repeated if the amount of evidence needed is not yet attained.

Incident closure: If on the other hand sufficient evidence has been accumulated, the investigation team now documented case files with exploitable digital evidence for the reactive investigation team to complete investigation. As soon as the ActDF investigation ends, the Reactive DF component has to continue its analyzes and reconstruct the incident basing on all the other evidences (comprising physical evidence and static exploitable digital evidence) which will serve in drawing conclusions of the investigation. In fact, during ongoing attacks, the ActDF component meets the need to gather live evidence.

3.4 Interactions between suggested components

In this subsection, we discuss the relationship between the different components of digital forensic to demonstrate the dependency between these components.

A complaint about an incident, an alert or an individual/automatic detection of an incident is the starting point of an investigation; a trigger event is known to be set-up by organizations to permanently perform live data acquisitions as soon as certain types of incident alerts are detected in proactive investigation purposes. The established

rules about any suspicious activity or event detection determine whether to run an investigation or not. If an investigation is needed, the investigator must consult the exploitable digital evidence map of the organization delivered at the ProDF component as well as the risk profiles and risk profile case scenarios mentioned in the incident guide books. If there is not sufficient evidence or the need for live evidence, ActDF must start, otherwise the ReaDF component will be activated. Once the investigator is satisfied that sufficient evidence exist, the ActDF component is terminated and the ReaDF component is then activated.

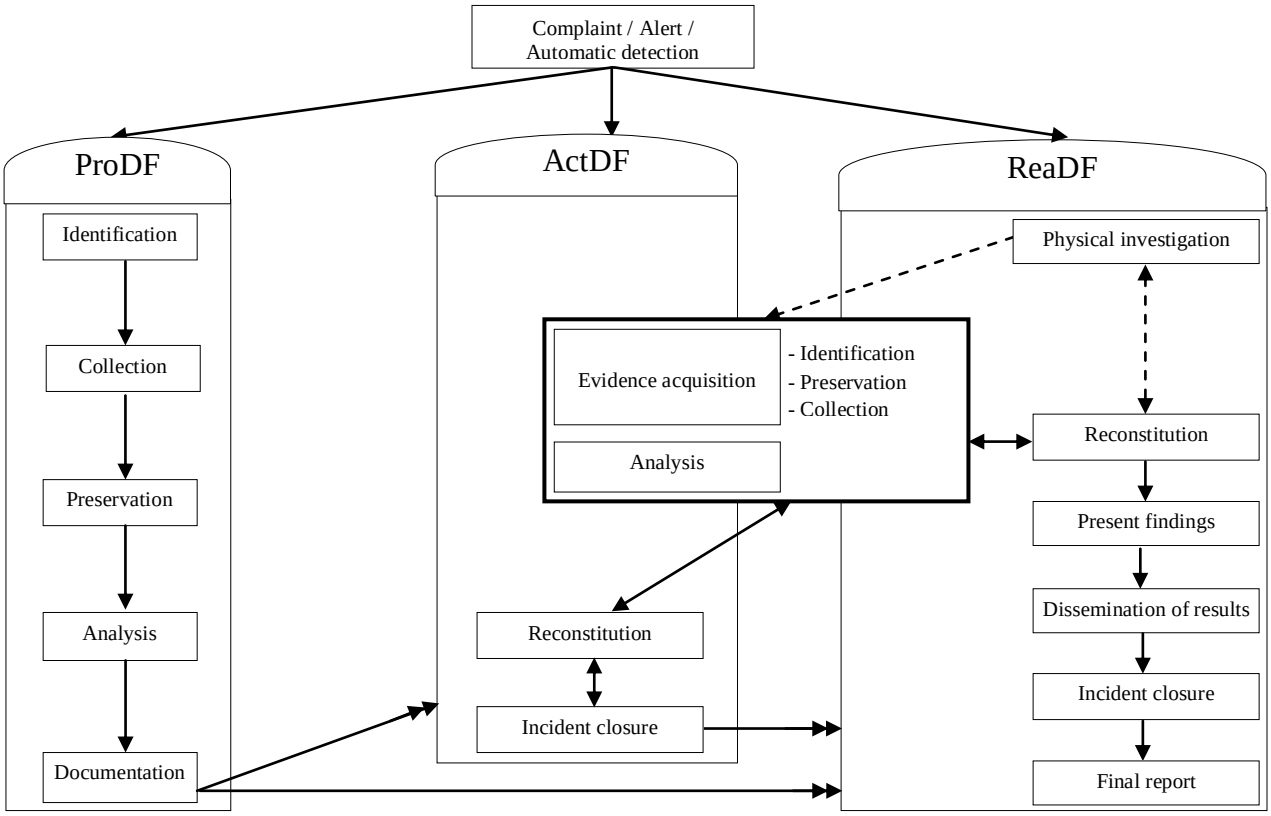


Fig 1: Process for ProDF, ActDF and ReaDF investigation system and interactions

4. MODELS MAPPING

CP Grobler and al. in 2010 [10] have defined a mapping between their model and the existing ones. They argue that their model is more appropriate than the previous models. Their argument was based on the advantages of their model compared to previous existing models. However, their proposed model does not cover the entire digital forensic investigation process. It does not deal with the proactive investigation integrating identification, collection, preservation, and analysis of salient data to prevent attacks. Dealing with this perspective, *Soltan Alharbi and al* in 2011 [11] have proposed a proactive-reactive digital forensic approach. However, this approach does not take into consideration the attacks detection. Furthermore, the ActDF process is quite ignored. This lies to insufficient actions to be undertaken to efficiently address reactive issue of digital

Digital forensic model	Proactive investigation						Active investigation					Reactive investigation								
	Alert	Identification	Collection	Preservation	Analysis	Documentation	Complaint / Alaert / Individual detection	Evidence acquisition	Analysis	Reconstitution	Incident closure	Complaint / Alaert / Individual detection	Physical Investigation	Evidence acquisition	Analysis	Reconstitution	Present findings	Dissemination of results	Incident closure	Final report
A multi-component view of Digital Forensics, 2010							√	√	√	√	√	√	√	√	√	√	√	√	√	√
The Proactive and Reactive Digital Forensics Investigation Process: A Systematic Literature Review, 2011		√	√	√	√	√								√	√					√
Multi-perpective cybercrime investigation process modeling, 2012,	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√	√

Table 1: Models mapping

investigation. Based on the different activities that need to be carried out in the digital forensic investigation process, the following table gives the mapping between the *CP Grobler and al.* model, the *Soltan Alharbi and al.* model and our proposed model. In this defined table, the symbol √ specifies the fact that the approach supports the associated activity. Moreover, the absence of this symbol denotes the fact that the approach does not address a given activity. By comparing the different rows representing each model, the row representing our proposed model support all the activities that should be taken into consideration in dealing with digital investigation within an organization. These activities are regrouped into reactive, active and proactive process and should be conducted within an organization that needs to efficiently face the cyber crime problem. The main problem in this domain is the definition of a complete investigation process of cybercrime. Our model gives the base for this process that needs to be validated by its application in various cybercrime situations within organizations. When this process will be validated, the next step will be the concrete implementation of the associated activities. In this phase, the Service Oriented Architecture (SOA) can be used since each main activity, i.e proactive, active and reactive, can be seen as a service. This is one of our future focuses.

5. CONCLUSION AND SUGGESTIONS FOR FUTURE WORKS

A detailed digital forensic procedure provides important assistance to forensic investigators in gathering evidence admissible in a court of law. From the digital forensic investigation process model proposed, it has been clearly stated that the investigation process will lead into a better prosecution as the very most important stages such as live data acquisition and static data acquisition are being implanted to focus on fragile evidence. Several models have already been established in the digital forensic field. However, none of this models take into consideration the overall investigation process characterize by three main interrelated activities: (a) Proactive digital forensics; (b) Active digital forensics; (c) Reactive digital forensics. In this paper, a multi-perspective cybercrime investigation process model has been defined by considering the three mains activities defined above. The defined model has been compared with the proposed models in [10, 11]. This comparison shows that the proposed digital forensic model takes into consideration the different properties of the three activities which is not the case for the previous models. The digital forensics community needs a structured framework for the rapid development of standard operational procedures that can be peer-reviewed, tested effectively and validated quickly. Digital forensics practitioners can benefit from the iterative structure proposed in this research paper to build forensically sound cases. The proposed model aims to develop a consistent and simplified forensics guides on digital forensic investigations that can be a guideline for standard operational procedure and a model for developing future technologies in digital forensic investigations.

This work does not consider the detection of an attack for which the inherent alert is the condition for beginning a digital investigation; In perspective, investigation are required to be done in order to extend the defined model by taking into consideration the detection of an attack in order to trigger the investigation process. Moreover, some case studies need to be done in order to validate the model.

6. REFERENCES

[1] Yong-Dal Shin. "New Model for Cyber Crime Investigation Procedure". Journal of Next Generation Information Technology, Volume 2, Number 2, May 2011.

[2] Inikpi O. Ademu, Dr Chris O. Imafidon, Dr David S. Preston: "A New Approach of Digital Forensic Model for Digital Forensic Investigation". (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No.12, 2011.

[3] Sundresan Perumal: "Digital Forensic Model Based On Malaysian Investigation Process". IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.8, August 2009.

[4] Séamus Ó Ciardhuáin: "An Extended Model of Cybercrime Investigations". International Journal of Digital Evidence Summer 2004, Volume 3, Issue 1.

[5] Ankit Agarwal, Megha Gupta, Saurabh Gupta \& Prof. (Dr.) S.C. Gupta: "Systematic Digital Forensic Investigation Model". International Journal of Computer Science and Security (IJCSS), Volume(5): Issue(1): 2011, 118-131.

[6] Baryamureeba and Florence Tushabe: "The Enhanced Digital Investigation Process Model". Institute of Computer Science, Makerere University P.O.Box 7062, Kampala Uganda www.makerere.ac.ug/ics.

[7] Brian Carrier Eugene H. Spafford: "Getting Physical with the Digital Investigation Process". International Journal of Digital Evidence Fall 2003, Volume 2, Issue 2.

[8] W. Thomas: Automata on Infinite Objects". Handbook on Theoretical Computer Science, J. Van Leeuwen, ed, pp. 133-187, Elsevier Science, 1990.

[9] G. Berry, P. Couronne and G. Gonthier:} "\texttt{Synchronous Programming of Reactive Systems: An introduction to ESTEREL". Proc. 1st Franco-Japanese Symp. on programming of Future Generation Computers, 1986, Tokyo. pp.35-56.S. Garfinkel, "Anti-forensics: Techniques, detection and countermeasures," in 2nd International Conference on i-Warfare and Security, 2007, p. 77.

[10] CP Grobler, CP Louwrens, SH von Solms "A multi-component view of Digital Forensics}". International Conference on Availability, Reliability and Security, 2010.

[11] Soltan Alharbi, Jens Weber-Jahnke, Issa Traore: "The Proactive and Reactive Digital Forensics Investigation Process: A Systematic Literature Review". International Journal of Security and Its Applications Vol. 5 No. 4, October, 2011.

[12] A. Orebaugh: "Proactive forensics". Journal of digital forensic Practice, vol. 1, p. 37, 2006.

[13] Ieong, R. and H. Leung: "Deriving Cse-specific Live Forensics Investigation Procedures from FORZA". In Symposium on Applied Computing archive Proceedings of the 2007 ACM symposium on Applied computing 2007. Seoul, Korea: ACM Press New York, NY, USA.

[14] Ren, W. and H. Jin: "Honeynet Based Distributed Adaptive Network Forensics and Active Real Time Investigation". In 2005 ACM Symposium on Applied Computing. 2005. Santa Fe, New Mexico, USA.

[15] Foster M, W.J: "Process Forensics: A pilot study on the use of checkpointing technology in computer forensics". International Journal of Digital Evidence, 2004. 3(1).

III

Atsa Etoundi Roger, **Mboupda Moyo Achille**, Nkoulou Onanena Georges, Nkondock Mi Bahanag Nicolas : " **A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit** ", International Journal of Computer Applications (0975 8887), Doi 10.5120/13973-1964, Volume 81 - No. 1, November
2013.
www.ijca.org.

Indexed with EBSCO, Google Scholar, Informatics, ProQuest CSA Technology Research Database, NASA ADS (Harvard Univ.), CiteSeer, Ulrich Web, Scientific Commons (Univ. of St Gallens), University of Karlsruhe, Germany, PennState University

NOVEMBRE 2013

A Formal Framework for Intrusion Detection within an Information System based on Workflow Audit

Atsa Etoundi Roger
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

Mboupda Moyo Achille
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

Nkoulou Onanena Georges
University of Paris-Est Creteil
Faculty of science and technology

Nkondock Mi Bahanag Nicolas
University of Yaounde I, Cameroon
Faculty of science
Department of Computer Science

ABSTRACT

Nowadays information systems are very critical and important as their various users are distributed around the world. The role played by the information system in the evolution of the society has led to a new form of economy, the leaders in this economy are those who will be able to master their information system in terms of security issues. Based on the quality of data managed in the information system for decision making, the security issue becomes more and more crucial. Among the challenges that are faced in the information system, it appears that the intrusion detection problem is the major challenge that needs to be discussed first as all attacks start with the intrusion which precedes various malicious activities. Many works had been done in this domain but the intrusion detection problem is still an open research topic in computer science. In this paper, the described problem is considered as an engineering one. The approach used in this research is based on the workflow theory which allows carrying out an efficient identification in different activities that are able to be performed. The defined approach is focused on a formal and sound description of resources that participate in the execution of identified activities. The result of this paper is the definition of a formal framework for intrusion detection based on workflow execution analysis.

Keywords:

Intrusion detection model, Workflow execution audit, Information system audit, Activity categorization

1. INTRODUCTION

As information systems play increasingly vital roles in modern society, they have become the target of enemies and criminals. Therefore, there is a need to find all possible best ways to protect these information systems. The security of an information system is compromised when an intrusion takes place. Based on the quality of data stored and managed in the information system for decision making within an organization, the security issues become more and more crucial. Attacks in information systems are dangerous as atomic bomb. The illustration example is the attack of the Estonian system in 2007 by Russian hackers. To do that, they used a system called Botnet to diffuse spams to many important services (bank, government,. . .) and then saturated the Estonian network; knowing that Estonia is a country where everything is managed with technology, so nothing was available . Another example is the action of a virus called "Stuxnet" created to affect Iranian nuclear program. It succeeds its task in 2010 by affecting processors used to control the rotation speed of centrifuges present in the nuclear station. "Stuxnet" was introduced in many computers in such a way that 3 months after its diffusion, a very large scope of computers was infected. Thus, an infected usb flash which was used on the protected nuclear system by an authorize person, had corrupted the system. The processors was constraint by a malicious code to hide real information to administrators about centrifuges speed execution. Finally, in a few moment, all the centrifuges where destroyed and nuclear program halted there for 2 years.

In this case, it appears that a bomb was not built to attack Iranian nuclear but a virus called "Stuxnet" acted more efficiently than a bomb. There exists some examples like that which show and justify the importance of security management. Among challenges that are faced in the information system, it appears that the intrusion detection problem [1] is the major challenge that need to be tackled first as all attacks start with the intrusion which precedes various malicious activities in the information system. An intrusion can be defined as any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource [2]. Many works have been done in this domain but the intrusion detection problem is still an open research topic in computer science based on the huge number of workshops and conferences that are organized in this field. In the state of the art of the intrusion detection, many techniques have been defined such as Target Monitoring which uses cryptography algorithm that computes a cryptochecksum for each target file and each modification and changes such as file modification or program logon which would cause changes in the cryptochecksum are reported by the IDS; Stealth Probes [1] which collects and correlates data to try to detect attacks made over long period of time, often referred to as low and slow attacks, but there are mainly two

types of intrusion detection techniques namely the Anomaly based intrusion detection and signature-based intrusion detection.

1.1 The anomaly detection technique

Anomaly detection is based on the normal behavior of an actor within an information system. Therefore, any action that significantly deviates from the normal behavior is considered like intrusion. Anomaly based intrusion detection is useful for detecting attacks like (1) misuse of protocol and service ports, in this case features of the standard protocols can sometimes be violated or modified by an intruder in order to tunnel through a firewall. An installation of backdoor services on well-known standard ports is another common misuse of service ports; (2) Denial of Service (DoS) based on crafted payloads, in this type of attack when a malicious intruder creates an attack using a crafted IP packet, the resulting DoS can occur on the network bandwidth, CPU cycles, memory resources, or application process/programs. The impact of the DoS attack is an anomaly in service quality; (3) Distributed Denial of Service (DDoS) is a form of attack that floods the network with a large volume of traffic. This is due to the fact that sophisticated attack traffic may not be distinguishable from regular network traffic on a peer packet basis and the attack does not manifest a specific signature that can be captured by signature-based mechanisms; (4) Buffer overflow which is the most common vulnerability exploited by attackers. Buffer overflow with shell code execution is the most serious form of this exploit because a successful attack can result in arbitrary program execution on the victim information system. Many exploited fields, such as user passwords for File transfer protocol, are supposedly made of printable ASCII characters based on the standard Request For Comments (RFCs) by the Internet Engineering Task Force (IETF). Excessive non-printable ASCII characters are anomalies of strong suspicion. Furthermore, shell code embedded in these fields are sure signs of malicious intent; (5) Other natural network failures which is based on the failures in routers/switches that can result in changes in traffic pattern observed at certain points of the network. This type of attack can be observed by a sudden dropping in the volume of traffic due to broken connections, or in the form of traffic shift from one link to another due to traffic rerouting as a recovery action. All these changes are noteworthy and can be detected as traffic anomalies. Anomaly detection technique represents a broad spectrum of detection techniques, for example Protocol anomaly detection, Application payload anomaly, and Statistical anomaly based intrusion detection [3, 4]. The anomaly based intrusion detection techniques allow to detect unusual behavior and thus have the ability to detect symptoms of attacks without specific knowledge of details. They also help in producing information that can be used to define signatures for misuse detectors. However, these techniques usually produce a large number of false alarms due to the unpredictable behaviors of users and networks; moreover, they often require extensive training sets of system event records in order to characterize normal behavior patterns.

1.2 The signature-based detection technique

The signature-based detection technique also known as misuse detection technique allows in detecting and catching intrusions in terms of the characteristics of known attacks or system vulnerabilities. Therefore, any action that conforms to the pattern of a known attack or vulnerability is considered as intrusive. This technique refers to techniques that use patterns of known intrusions or weak spots of a system to match and identify intrusions. The sequence of attack actions or activities, the conditions that compromise an information system's security, as well as the damage left behind by intrusions can be represented by a number of general pattern matching models. In order to carry out the detection of intrusion, researchers in this field use rules to describe attack actions [5], state transition diagrams to model general states of the system and access control violations, and Colored Petri nets to represent intrusion signatures as sequences of events on the target system. In the signature based intrusion detection, two techniques are used: (a) "Expression matching" is based on expression matching, which searches an event stream for occurrences of specified patterns or signatures; (b) "State transition analysis" models attacks as a network of states and transitions (matching events) where every observed event is applied to the finite state machine instances which represents each an attack scenario, possibly causing transitions. Any machine that reaches its final state indicates an attack. This approach allows complex intrusion scenarios to be modeled in a simple way, and is capable of detecting slow or distributed attacks, but may have difficulty in expressing elaborated scenarios. Based on these techniques, the signature-based intrusion detection technique allows in detecting attacks without generating an overwhelming number of false alarms as it was the case for the anomaly detection technique, they can also help information system manager in quickly and reliably diagnosing the use of a specified attack tool or technique in order to take corrective measures and track security problems on their information systems. However, the signature-based intrusion detection can only detect known attacks and needs to constantly update the information regarding new attacks. Therefore, newly invented attacks will likely go undetected, leading to unacceptable false negative error rates.

Based on these various techniques, many tools have been developed in order to face the intrusion detection problem depending on the application domain. Despite the popularity of the above techniques and associated tools, there is still enormous work to be carried out in the intrusion detection problem field as there is no solution that handles this problem in an entire manner. As the various techniques defined do not overcome all the difficulties encountered in the intrusion detection problem. In this paper, the intrusion detection problem is considered as an engineering one. The approach used in this research essay is based on the workflow theory which allows carrying out an efficient identification of different activities that are able to be performed within an information system. The proposed approach is focused on a formal and sound description of resources (equipments, programs, and human actor) that participate in the execution of identified activities. The result of this paper is the definition of a formal framework for intrusion detection based on workflow execution audit.

The rest of the paper is organized as follows: in section 2 a survey of a workflow mining is presented. This survey is followed by the section 3 where the definition of keys elements of the workflow theory that are suitable to build the proposed framework in section 4 are given. The section 5 concludes the paper by highlighting some futures works.

2. WORKFLOW MINING

The starting point for workflow mining is based on workflow log containing information about the workflow process as it is actually being executed in the information system. The workflow life cycle consists of four phases [5]: (1) the workflow design which allows the construction of the workflow model based on the information at hand and the goals to be achieved with the identified activities and associated constraints, (2) the workflow configuration which deals

with limitation and particularities of the resulting workflow management system in the achievement of certain objectives within an organization, (3) the workflow enactment which deals with the integration in the used workflow system new functionalities according to new requirement that have been identified based on the strategy of the organization. In this phase, the information system is aligned to the vision of the organization, and (4) the workflow diagnosis in this phase from which data obtained from workflow instances execution are analyzed. This analysis can result to workflow/workflow process reengineering or provide inputs for the design of a new workflow which completes the live cycle of the former workflow.

To carry out the workflow enactment and workflow diagnosis tasks, the workflow mining is used as a guideline. The purpose of workflow mining is therefore to reverse the process and collect data at runtime to support workflow design and analysis. The information collected during the execution phase of a workflow is used to derive a model explaining the events recorded in the information system. The modeling of a workflow instance is usually carried out by considering the perception but does not really express what will exactly be done in practice. This looks to be the same in the security domain as the security policy is defined about the perception that the information system security manager has but not what is really carried out in the real world. For this purpose, models are often normative in the since that they state what should be done rather than what is really done by the workflow process. The type of models seems to be subjective. For a model to be objective, the designer must take into consideration data that are produced by the real execution of workflow instances. The data obtained are therefore linked to activities, time constraints and resources responsible for their generation. The workflow mining can then be very important in the intrusion detection problem as it allows drawing up what is really carried out in the information system by building a model corresponding on the exact situation or the state of the information system in terms of workflow execution, this model can therefore be compared to perceived model also known as an " priori" model in order to highlight the gap between the two models. The "a priori" model is generally based on the perceived security policy of the information which is defined by the set of actions to be performed by identified users based of given resources. The " priori" model is computed based on the information obtained within the execution phase of all activities in the information system. These associated data are stored in the information system logs. These logs therefore keep track of every event that occurs in the information system for future analysis and decision making. Closely monitoring the events taking place at runtime also enables detecting discrepancies between the design constructed in the design phase of the security policy and the actual execution registered in the intrusion handling. Workflow technology is moving into the direction of more operational flexibility to deal with workflow exception handling. Most exceptions can be seen as an intrusion in the intrusion detection problem. As a result, information system users can deviate from the " priori" information security policy design, for this purpose, according to the data kept in information system logs, the related deviation can be monitored. In order to push out the above defined gap for intrusion detection purpose. There is no uncertainty that workflow mining techniques can be used to create a feedback loop to adapt the workflow model to changing circumstances and detect in the mean time imperfections during and after the execution phase. The use of workflow mining in this work is not to show how workflow models are built from workflow event logs but detect intrusion from event logs based on the predefined information system security policy. More materials concerning workflow mining can be found in [6, 7, 8].

3. KEYS ELEMENTS

In this section, salient concepts suitable for the modeling of a workflow mining for intrusion detection problem are defined. The presentation of these concepts is given using the denotational semantics [4]. Concepts are modeled from the environment used by an activity for its execution to human actors in charge of this execution.

3.1 The Environment Description Model

In the modeling of business processes proposed by a great number of researchers, such as Van der Aalst [6] and Jorg Becker and al. [9], one can notice that they have neglected the effect that an environment can produce in the achievement of a given business goal within an organization. For example, in developing countries, moving from one hospital to another, the manner of carrying out surgeries differ even if the surgeons graduated from the same institute. This is due to the fact that hospitals are not equipped in the same manner. ln the modeling of a business process, the environment within which it will be performed has to be taken into consideration.

Without any loss of generality, an environment is assumed to be a set of different metrics whose value may change [10, 11]. These metrics are primitive Boolean observers denoted by Observer. The associated value of each observer depends on the current state of the environment.

Formally, an environment E is defined as a triple $\langle \Theta, S, val \rangle$ where:
- Θ is a non empty set of observers;
- S is a non empty set of states;
- $val : \Theta \rightarrow (S \rightarrow Bool)$ is a function which describes the behavior of observers.

In the rest of this paper, $val(o)(s)$ is denoted by $s(o)$ where s denotes a state and o an observer, $s(o)$ is the value of the observer o in the state s. Given a state s, the set of observers whose value is true defines the characteristic of s and is $S_c = \{o \in \Theta, s(o) = true\}$.

Given two states $s1$ and $s2$ of the set of states S of the environment E, the set of observers whose associated values are not the same is defined from the characteristics of the two states. This set is called gap between $s1$ and $s2$ and is denoted by $s1 \bullet s2 = (s1_c - s2_c) \cup (s2_c - s1_c)$.

Given an environment E, the observers in Θ define the alphabet that permits to reason about events that occur on E. The language defined from this alphabet is denoted by the set of conditions or formulae C. A condition $c \in C$ is an assertion over observers and is defined as a first order formula. The basic elements of C are therefore all the observers of Θ. The elements of C are formed by the following:

$$\begin{cases} \text{if} & \text{o} \in \Theta, \text{then } o \in C \\ \text{if} & \text{o} \in \Theta, \text{then } \neg o \in C \\ \text{if} & \text{o}_1, o_2 \in C, \text{then } o_1 \wedge o_2, o_1 \vee o_2, o_1 \Rightarrow o_2 \in C \end{cases}$$

A condition c can be decomposed into a set of observers $+c$ whose values are evaluated to true and a set of observers $-c$ that are evaluated to false. The two sets do not have any common element i.e. $+c \cap -c = \emptyset$.

Given a condition $c \in C$ and a state s, c is satisfied within the state s if the result of its evaluation is true, i.e. $s(c) = true$.

3.1.1 State of an Environment. A state is a snapshot of an environment within a time. From this snapshot, facts are observed. Some of these facts or features of a state are true or false at this particular time. These facts are represented as some equivalent of predicate calculus formulae. Somewhat loosely, these facts and relations will be refer as attributes of a state. In a rigorous manner, let F be a set of formulae, and s be a state, then s is a subset of F i.e. $s \in F$.

In general, let S be a set of states, according to the definition of a state, $(S, \subseteq)$ is a partial ordered set. This paper doesn't dealing with any kind of set of states, it considers the set of states S having a least state, $\perp_S$, known as initial state of a business process or workflow from which the execution can be started. This initial state is therefore contained in all states of S i.e for all $s \in S$, $\perp_S \subseteq S$. In the meantime, S is required to have a least upper bound $\bigcup_S$ known as a state where the goal of the business process is satisfied. As such, the set of states of a business process is mapped to a CPO concept.

3.2 Task Description Model

A task is an atomic activity that cannot be split into smaller activities. The performance or execution of a task transforms the state of the environment into another state. A task is therefore an action within a state of an environment. Before a task can be executed, the state of the environment should satisfy a specific condition called pre-condition,and when this execution is completed another condition, called post-condition is satisfied. A task is formally defined by a triple $< nt, pre, post >$ where nt denotes the name of the task, pre its pre-condition, and $post$ its post-condition.

The action of a task within an environment is to transform its current state into a new one. When $< t, pre, post >$ is a task, s a given state where the precondition pre is satisfied i.e $s(pre) = true$, the action of t in the state s is the new state $t(s)$ which satisfies the post condition $post$ i.e $t(s)(post) = true$. In general, the action of a task t within the state s is characterized by the observers of s whose value has been modified.

DEFINITION 1. *(Task action)*

Let $E = \langle \Theta, S, val \rangle$ be an environment, s a given state and t a task whose pre condition is satisfied in s, then the action of t in s denoted by t_s and is specified by $t_s = \{o : \Theta, s(o) \neq t(s)(o)\}$.

When there will be no ambiguity, a task will be represented by its name t and $pre(t)$ respectively $post(t)$ will denote respectively its pre and post-condition. Based on the post-condition of a task t, and the state s where $s(post(t)) = true$, the conjecture $t_s = +post(t) \cup -post(t)$ can be done.

DEFINITION 2. *(Conflicting Tasks)*

The action of tasks within an environment can be conflicted as many tasks can modify the same observers at the same time. For this purpose, t_1 and t_2 are conflicting tasks in the state s, which is denoted by $overlap(t_1, t_2, s)$, if and only if:

$$\begin{cases} \text{s(pre}(t_1)) = s(pre(t_2)) = true \\ +\text{post}(t_1) \cap -post(t_2) \neq 0 \\ +\text{post}(t_2) \cap -post(t_1) \neq 0 \end{cases}$$

DEFINITION 3. *(Shift)*

Let SoT be a none empty set of tasks and s a given state, a shift denoted by Shf is a couple $Shf = < s, SoT >$ composed with the state s and the set of non conflicting tasks SoT within s.

Formally, let $Shf = < s, SoT >$ be a shift, the following properties are satisfied:

$$\begin{cases} \text{SoT} \neq \emptyset \quad (1); \\ \forall t \in SoT, s(pre(t)) = true \quad (2); \\ \forall t \in SoT, t \neq t' \Rightarrow overlap(t, t', s) = false \quad (3) \end{cases}$$

Let $Sht = < s, SoT >$ be a shift, the simultaneous actions of SoT in s, denoted by $ts(s)$, is captured by the set of observers whose values are modified within s, that is: $\text{SoT(s)} = \bigcup \left\{ \begin{array}{l} \text{o} \in \Theta: \\ \text{o} \in -post(t_i) \end{array} \right\}, t_i \in SoT$

DEFINITION 4. *(Chain)*

A chain is an execution path of tasks according to their actions in states, their triggering conditions is denoted by $P = \prod_{i=1}^{n} Sht_i$, and is specified as a finite sequence of shifts where n represents the length of the sequence.

Let P be a path of length $n > 1$, and $sh_k = \langle s_k, st_k \rangle$, $shk + 1 = \langle s_{k+1}, st_{k+1} \rangle$ notes respectively the shift in the range k and $k+1$, the state s_{k+l} is the resulting state of the execution of the set of tasks st_k i.e $s_{k+1} = st_k(s_k)$. When there will be no ambiguity, the shift of the range k of the path P will be denoted by $P(k)$.

Let $Sht_k = \langle s_k, SoT_k \rangle$ and $Sht_{k+1} = \langle s_{k+1}, SoT_{k+1} \rangle$ be two shifts where $Sht_k = SoT_k(s_k)$, the difference between the states s_k and s_{k+1} is denoted by $\overline{s_k + s_{k+1}}$ and is defined as follows: $\overline{s_k + s_{k+1}} = SoT_k(s_k)$

LEMMA 5.

Let p be an executin path and $t \in SoT(p(k))$ with $k \leq length(p)$ then there will always exist m such that $m > k$ and $S(p(m))(post(t)) = false$.

LEMMA 6.

Let p be an execution path then $SoT(p(length(p))) = \emptyset$

DEFINITION 7. *(State ordering)*

Let P be a path of length $n > 1$, $Sht_k = \langle s_k, SoT_k \rangle$ and $Sht_{k+1} = \langle s_{k+1}, SoT_{k+1} \rangle$ be two consecutive shifts in P with $k < n$ then $s_k \subseteq s_{k+1}$ specifies the fact that the set of observers modified in s_k after the actions of SoT are contained in the set of observers of s_{k+1} with the same values.

LEMMA 8.

Let P be an execution path, S the set of states of P, then $(S, \subseteq)$ is CPO where the least upper bound state in the last state of P and the least state is the first state of P.

At the level of this modeling, it is important to ensure that the execution of a task t will stop at a certain time. In order to do so, the set of observers that should be modified by t must be contained partially or totally in the observers forming its pre-condition $(-pre(t) \cup -pre(t)) \cap (-post(t) \cup +post(t)) \neq \emptyset$. From the definition of the execution path of tasks, the relation within the set of tasks T based on the set of states S can be specified. This relation is denoted by $\trianglelefteq$.

DEFINITION 9. *(Ordering of Tasks)*

Let T be a set of tasks, and t_1 and t_2 be two tasks of T, $t_1 \trianglelefteq t_2$ if and only if for all chain CH such that if n_{t_1} and n_{t_2} denote respectively the maximum range of t_1 and t_2 in CH, then $n_{t_1} \leq n_{t_2}$.

This relation has the following properties:

(1) *reflexivity*: $t \trianglelefteq t$ this simply means that the task t belongs to the chain CH;
(2) *antisymmetric*: if $t_1 \trianglelefteq t_2$ and $t_2 \trianglelefteq t_1$ in the chain D then $t_1 = t_2$. By convention, there will always exist a path from each task to itself;
(3) *transitivity*: obviously if in the chain CH, $t_1 \trianglelefteq t_2$ and $t_2 \trianglelefteq t_3$ then $t_1 \trianglelefteq t_3$

LEMMA 10.

The set of tasks T associated with the relation previously defined $\trianglelefteq$, i.e. $(T, \trianglelefteq)$, forms a complete partial ordered set.

3.2.1 Palette

DEFINITION 11.

Let E be an environment, and S be a set of different states that E may reach according to the actions of tasks T, then a palette P is a couple $< E, S \rightarrow S >$. In the rest of this paper, the set of functions $S \rightarrow S$ will be denoted by T, the set of tasks of the palette. When there will be no ambiguity, $P(E)$ and $P(T)$ will denote the environment and the set of tasks of the palette P respectively.

The actions of the set of tasks T of the palette P in the environment E are to change at least once the value of each observer of Θ in E. To this end, the consecutive actions of a non empty set of tasks within an environment may not modify all the observers in this environment. The set of observers whose value are not changed during the execution of any given none empty set of tasks will be abstracted from all the possible states of the environment, i.e. $\forall o \in \Theta \begin{cases} \exists\, t_1 \in T, \text{o} \in -post(t_1) \text{ or} \\ \exists\, t_2 \in T, \text{o} \in +post(t_2) \end{cases}$

Given a palette P, according to the environment changes within organizations and the different executions of tasks that can take place, different ways in which tasks can be executed have to be captured. In the rest of the paper, SPP will be use to specify the set of execution paths that can be obtained from a palette P.

LEMMA 12.

Let P be a palette, $s \in S(P)$ a given state of the environment $E(P)$ of P, there will always exist a path $p \in SP_p$ such that $s \in S(P)$, where $S(p)$ denotes the set of states of the path p.

LEMMA 13.

Let $P =< E, T >$ be a palette, and $t \in T$, there will exist an execution path $ch \in SP_p$ where SP_p denotes the set of possible execution paths of T, $ch(n) = \langle s_n, SoT_n \rangle$ such that $t \in SoT_n$.

3.3 Business Process Model

A business process is a collection of activities or tasks designed to produce a specific output for customers. It implies a strong emphasis on how work is done within an organization in order to deliver a particular service. A process is thus a specific order of work activities across time and space, with a beginning, an end, and clearly defined inputs and outputs. The output is the reason the organization does this work and is defined in terms of the benefits this process has for the organization as a whole.

DEFINITION 14. *(A service)*

A service is the characteristic of a business process and is defined as a composition of a set of criteria that characterize what is delivered within an organization, where each criterion is represented by an observer.

The model of a business process is defined as a couple $< P, G >$ where P is a palette and G the service to be achieved. According to the definition of the palette, the ordering of tasks is captured explicitly by their pre conditions and the states of the environment within which their execution is being carried out.

This approach reduces the number of patterns to be used in order to capture various ways tasks can be ordered. This is the main difference between the approach in this paper and other BPM theory papers presented in the literature. In these works, the Workflow Management Coalition [6] has identified four basic control structures for workflows: OR-SPLIT, OR-Join, AND-Split and AND-Join. More control structures have been identified by Van der Aalst in [7]. These control flow structures can be formally captured by the first order formula. A new direction for future works will consist of presenting these concepts using the denotation defined above.

LEMMA 15.

There will always exist a state S_{lub} such that when it is reached, other states cannot be reached. This state is called a least upper bound state of the associated business process.

LEMMA 16.

There will always exist a state S_{ini} from which the execution of the business process starts. This state is called a least state of the associated business process. For each service associated to a given business process, a set of qualities of service is defined to deal with the daily work and the competitive pressure of the network economy.

3.4 QoS Model

The quality of service denoted by QoS represents the performances of the service which determine the level of satisfaction projected for the recipients of the services. The level of satisfaction is defined as a set of properties, criteria, characteristics and performances of the services delivered to the customers. Several works are made in this field, each one defining a specific set of criteria specified in order to measure the QoS. In the literature, there is no consensus yet on the definition of the set of common criteria to evaluate the quality of service delivered in the organizations [12, 13]. The evaluation criteria are defined according to the objectives and specificities of each company. In this work, an abstract model which gives the semantics of the quality of service is defined.

DEFINITION 17.

Let Cr be a set of criteria considered in the evaluation of the quality of service, Val the set of values that can be assigned to these criteria, and f a map defined by $f : C \rightarrow Val$, the QoS is defined by (C, Val, f).

Given two QoS q_1 and q_2 such that $q_1 = (Cr_1, Val_1, f_1)$ and $q_2 = (Cr_2, Val_2, f_2)$, q_1 and q_2 are compatible and denote by $q_1 \triangle q_2$ if and only if $C_1 = C_2$ and $Val_1 = Val_2$. When q_1 and q_2 are compatible, q_1 is better to q_2 and denote by $q_1 \subseteq q_2$ if and only if $\forall c \in C_1$, $f_1(c) \leq f_2(c)$. In the rest of the paper, $(\Phi, \subseteq)$ will be use to denote the partial ordered set of compatible qualities of services.

DEFINITION 18. *(Well Defined Business Process)*

Let, $BP =< P, G >$ be a business process, BP is well defined if and only if all the observers that form its goal (service) are contained in the set of observers of the environment E i.e. $-G \cup +G \subseteq \Theta$ (E).

DEFINITION 19. *(Well Formed Business Process)*

Let $BP =< P, G >$ be a business process, BP is said to be well formed if and only if each execution chain SCH reaches the least upper bound state S_{lub} which satisfies the service G i.e.

$$\begin{cases} \forall ch \in SCH n_{ch} \in N, S_{lub} \in S \\ \mathrm{n}_{ch} = length(ch) \\ \mathrm{S}_{lub} \\ \mathrm{S}_{lub}(G) = true \end{cases}$$

More formally, let SCH be the non empty set of different chains that can be obtained from a business process BP, and $CH \in SCH$ with the length n_{CH} such that the n^{th}_{CH} state S_{lub} of CH satisfies G i.e. $S_{lub} = true$.

DEFINITION 20. *(Deadlock- and Livelock-Free)*

Let BP be a business process, BP is deadlock- and livelock-free if and only if it guarantees that every execution chain reaches its least upper bound state satisfying the goal of the business process BP.

THEOREM 21.

Let BP denote a business process such that BP is well defined and well formed, then BP is deadlock- free and livelock-free.

The proof of this theorem can be found in [10].

All the execution paths of a business process start from the same state denoted by S_{ini}. It can be easily being shown that the set of states S_{BP} associated with the ordering relation $\subseteq$ as defined previously is CPO.

3.5 Agent Generic Model

There are many types of agents participating in the processing of tasks within an information system. An information system dealing with the processing of tasks is a hybrid system including hardware components with embedded software, human actors interacting with the hardware and software. Agents are performing tasks in order to carry out certain missions, which, in its turn, add a dimension to the quality of service. Each agent has a skill which is characterized by (Sk, Tks, mch) where Sk is the set of competencies, Tks the set of tasks and mch a map that gives for each competence $cp \in Sk$ the set of tasks $mch(cp) \in Tks$ that can be processed based on cp with $mch(cp) \neq \emptyset$. When there will be no ambiguity, the structure (Sk, Tks, mch) will be represented by Sk. Based on the organization put in place, the set of tasks performed by an agent is kept in a diary.

A diary is described by the set of tasks and the set of time intervals within which there are processed. Let $Pds = (TI, \subseteq, \cap, \triangle)$ be a set of time intervals such that $(TI, \subseteq)$ is a partial ordered set with ∂ the smallest time interval, $\cap$ and $\triangle$ be two maps defined as follows $\cap : TI \text{ x } TI \rightarrow TI$ and $\triangle : TI \text{ x } TI \rightarrow Boolean$. If p_1 and p_2 are two time intervals, p_1 and p_2 overlapped if and only if there exists a time interval p_3 such that: $\mathrm{p}_1 \cap p_2 = t_3 \Rightarrow \begin{cases} \mathrm{p}_3 \triangle p_1 \wedge p_3 \triangle p_2 \\ \mathrm{p}_3 \subseteq p_1 \wedge p_3 \subseteq p_2 \end{cases}$ where $\cap$ and $\triangle$ define respectively the intersection and the overlapping relationship. When there will be no ambiguity, the set of time intervals will be represented by Pds. Based on the concepts of tasks and time interval, the diary concept is modeled by $< Tks, Pds, g >$ where Tks is the set of tasks, Pds the set of associated time intervals, and g a map defined by $g : Tks \rightarrow Pds$ such that $\forall\, t_1, t_2 \in Tks, t_1 \neq t_2 \Rightarrow \neg(g(t_1) \triangle \mathrm{g}(\mathrm{t}_2))$.

DEFINITION 22. *(Resource model)*

A resource model Rm is defined by $< Id, Sk, Ex, Dy, f >$ where Id is its identification, Sk, its set of skills, Ex, the set of associated experiences, Dy its associated diary and f a map which defines for each skill $sk \in Sk$ its associated experience $f(sk) \in Ex$.

3.6 Enterprise

An enterprise is a system dealing with the service delivery based on a certain quality of service. This system is organized in terms of business processes that are carried out, agents in charge of the processing of the associated tasks, and the resulting workflows. An information system IS is modeled by $(Io, BPs, Emps, WFs)$ where Io is its identification, BPs is the set of its business processes that can be run, $Emps$ its set of agents who participated in the processing of tasks, WFs its set of workflows.

A workflow is formally defined by $(Ts, Es, Ps, h, g, q, Q)^+$ where Ts is the set of none conflicting tasks, Es the set of agents dealing with the processing of Ts within the time intervals Ps to obtain the quality of service Q, h is the map $Ts \rightarrow Ps$ which defines for each task t, its time interval $h(t)$ within which it is processed, g a map $Es \rightarrow Ts$ which gives for each agent ag the associated performing task $g(ag)$, and q a map $Es \times Ts \rightarrow Q$ which gives for a given agent ag and its associated task tk the quality of service $g(ag, tk)$ obtained after the execution.

Based on the agents using or working in a given enterprise and their availability and the services required by customers, agents involved in different workflows associated to a business process will not necessarily be the same. To this end, according to their skills, the quality of service delivered may be different. The criteria for the evaluation of the quality of service will then sometimes be associated with minimum values when tasks will be processed by agent with minimum experience. More-over these values will be maximal when agent with maximum experience has been involved in the processing of tasks. The set of quality of service associated to a given business process will therefore have two specific qualities of service Q_{min} and Q_{max} which have the following properties.

LEMMA 23.

Let $Q_{min} = (C, Val, f_{qmin})$ and $Q_{max} = (C, Val, f_{qmax})$, be minimal and the maximal quality of service of a business process (P, Φ) then $\forall p = (C, V, f_p) \in \Phi$, $c \in C$, $f_{qmin}(c) \leq f_p(c)$, and $\forall q = (C, V, f_q \in \Phi$, $c \in C$, $f_q(c) \leq f_{qmax}(c)$.

4. FORMAL FRAMEWORK

In this section, the proposed framework for the intrusion detection problem within an information system is presented based on the workflow theory presented in the previous section. The construction of the model starts with a generic formal description of various resources that are founded in the information system suitable to be subject to attacks. This description is followed by the definition of the normative model of the security policy applied in the information system. The normative model is followed by the definition of the descriptive model associated to the effective usability of different resources of the information system. The section ends

with the definition of an approach defining the intrusion detection based on the audit of workflow instances execution within the information system. The audit compares data related to the normative and descriptive models.

4.1 Information System Model

As defined in the introduction of this work, a resource within an information system is either a human actor, equipment, a function (program) or a data. Resources are using other resources in order to perform specific activities within some resources. Activities are different tasks or functions that can be carried out by resources within the information system. Each resource is associated with a set of activities that it may perform, and a set of activities to which it can be applied. Moreover, from a given resource, a set of traces of activities to which it can be applied, and the set of traces of activities that it performs are specified. In the same manner, a resource may contain sub-resources. Each sub-resource in a given resource is considered as a resource. The set of traces actually defines the so called event logs used in workflow mining.

DEFINITION 24. *(Resource)*

Formally, a resource, denoted by R in this work, is modeled by $< Id, Log\_in, Log\_out, Task\_in, Task\_out, SubRes >$ where:
- Id is the identifier of the resource;
- Log_in represents the event log of incoming activities;
- Log_out represents the event log of out coming activities;
- $Task\_in$ represents the set of possible in coming activities;
- $Task\_out$ represents the set of possible out coming activities;
- $SubRes$ denotes the set of sub resources.

When there will be no ambiguity for a given resource R, the following expressions will be equivalent:
- $R(Id) = Id$;
- $R(Log\_in) = Log\_in$;
- $R(Log\_out) = Log\_out$;
- $R(Task\_in) = Task\_in$;
- $R(Task\_out) = Task\_out$;
- $R(SubRes) = SubRes$.

Given a resource in an information system, its sub-resources are also resources of that information system. In this modeling approach, two type of resources are considered intermediary resources and terminal resources. A resource r is said to be intermediate when r has one or many sub-resources. Otherwise, r is called a terminal resource.

Resources are any agent that execute or able to participate in the achievement of an activity in the information system. Based on the fact that an activity itself is able to launch the execution of another activity, it is also considered to be the case among resources of the information system. Thus for a task t to be performed, many agents are considered. For this end, the different tasks that a resource can perform or trigger are based on the environment of the information system.

DEFINITION 25. *(Environment)*

The environment is defined by $E_{IS} =< \theta_{IS}, S_{IS}, Bool >$ Where:
- θ_{IS} is the set of observable events ($\theta_{IS} = \bigcup_{i=0}^{\infty}(Ev_i)$);
- S_{IS} is the set of states.

DEFINITION 26. *(Event)*

An Event, denoted by Ev, is modeled by $< Id, Task, T\_In, T\_Out, Resp\_Res, Used\_Res, Task\_Act >$ where:
- Id is Event identifier;
- $Task$ represents activity of the event;
- T_in is the beginning time of the event;
- T_out is the ending time of the event;
- $Resp\_Res$ is the responsible of resource used;
- $Task\_act$ represents the action of the task.

The execution of a given activity (task) is based on a specific state of the information system. A state S is formally modeled by the following $S = \bigcup_{r \in R}(S_r)$ where $S_r = Log\_in(r) \cup Log\_out(r)$.

Having defined the environment, the state and the resource model, one can easily deal with the information system.

DEFINITION 27. *(Information system)*

The information system based on all previously defined elements is represented by:
$IS = (R, S, BPs, WFs, BPs^{WFs}, Id\_Res^{Log\_in}, Id\_Res^{Log\_out}, Id\_Res^{Tr}, Id\_Res^{Task\_in-set}, Id\_Res^{Task\_out-set})$
where:
- R: the set of resources;
- S: the set of states;
- BPs: the set of business processes that can be supported in SI;
- WFs: the set of workflows that can be run in SI;
- $BPs^{WFs} = BPs \underset{m}{\rightarrow} WFs$: the set of function that give for every business process the set of associated workflows;
- $Id\_Res^{Log\_in} = Id\_Res \underset{m}{\rightarrow} Log\_in$: the set of functions define for a given resource its associated supported events;
- $Id\_Res^{Log\_out} = Id\_Res \underset{m}{\rightarrow} Log\_out$: the set of functions that define for a given resource its associated initiated events;
$Id\_Res^{Tr} = Id\_Res \underset{m}{\rightarrow} Tr$: the set of functions that define for a given resource its triggering conditions;
- $Id\_Res^{Task\_in-set} = Id\_Res \underset{m}{\rightarrow} Task\_in - set$: the set of function that define for a given resource the set of tasks that it can handle;
$Id\_Res^{Task\_out-set} = Id\_Res \underset{m}{\rightarrow} Task\_out - set$: the set of function that define for a given resource the set of tasks that it can perform.

The defined model of the information system does not take into consideration the way different activities will be carried out based on the usability of various resources. This lets the resulting model exposed to malicious operations. In order to avoid this, the security policy on how tasks and resources are put together in the achievement of various goals is needed. The requirement is discussed in the following section.

4.2 Security policy

The security policy within an information system is so important that it guides how each resource should be used in the resulting organization. This policy is a set of rules that are required to be followed by any known and authorized entity in the organization. In this work, based in the intrusion detection problem which is considered as an engineering one, the security policy is based on four main concepts i.e the *Resources*, the *Tasks*, the *Time* and the *Trigger*. The concept of resource is used in order to list all the resources of the organization whose usability can violate its security rules. The task concept allows to identify various activities, functions or tasks that can be executed anyhow in the organization based on identified resources. When a given activity is then executed with the organization, the trace of this execution should be kept for fur-

ther investigation. For this purpose, one may like to know the time within which an observed execution has taken place. The answer of this important question is based on the definition of the starting and ending time of this execution. The *Period* concept related to the time concept is therefore very critical in this situation. In the management of the information system, some resources are keeping others ready to be used i.e if the resource kept is for example a program installed within a machine, it can be run at any time by another resource. this means that that this program has the required environment for its execution available. In order to control the execution of such resource or activity, one needs to a triggering condition that will really trigger the resulting execution. This also means that, an activity can have its pre condition satisfied, but if the triggering condition is not satisfied, this activity will not be able to be executed.Based on these defined concepts, the security policy SP is modeled as follows:

$$\text{SP} = \begin{cases} \text{out} = \text{R} \underset{m}{\longrightarrow} Task - set \\ \text{in} = \text{R} \underset{m}{\longrightarrow} Task - set \\ \text{p\_out} = \text{R} \underset{m}{\longrightarrow} R \underset{m}{\longrightarrow} Period \underset{m}{\longrightarrow} Task - set \\ \text{p\_in} = \text{R} \underset{m}{\longrightarrow} R \underset{m}{\longrightarrow} Period \underset{m}{\longrightarrow} Task - set \\ \text{Tg} = \text{Task} \underset{m}{\longrightarrow} Trigger \end{cases}$$

The various maps defined the relationships between the four concepts in order to highlight different events that took place in the information system by violating its security policy. Based on the security policy, the normative model of the information system can now be expressed.

4.3 Normative Information system model

Based on the application of the security policy in the target information system, and without any external action that that violate this policy the model of the information system within the time is as expected by the different managers in the organization. The Normative Model, denoted by IS_{nor}, is formally specified as follows
$IS_{nor} = (IS; Runs, SP)$
where:
- IS is a target information system;
- $Runs$ is the set of workflows that have been executed in IS by applying the security policy previously defined;
- SP is the associated security policy.

The normative model is far to match the reality of the information system. If one considers the higher number of attacks that faced different information systems around the world, the normative model of the information system is a view of spirit as each time, security rules are violated based on internal or external actions. The number of attacks reported is not the real situation since in many organizations, managers do not reveal attacks to the public. This is sometimes due to the fact that they do not want to frustrate theirs customers. As a consequence, it lead to the anger of customers, their weight bill which depends on the faithful relationship with their customers will decrease.In most of the time, they did not even know that their information system has been a source of attacks. Based on this strong remarks of the normative model, the real situation of the information system is represented by the descriptive model

4.4 Descriptive security model

This Model describes the information system using real activities execution within the system. Let IS_{des} be the Descriptive Model defined as follow:
$IS_{des} = \langle IS, Runs \rangle$
where:
- IS is the information system;
- $Runs$ is the set of events satisfied by triggering conditions for task execution.

4.5 Intrusion detection model

Let IS_{des} and IS_{nor} denote respectively the descriptive and the normative model of the information system IS, IS_{des} is said to be conform to the normative information system, noted $IS_{nor} \models IS_{des}$ if and only if for each resource r the following constraint is satisfied:

$log\_y(r, IS_{des}) \subseteq log\_y(r, IS_{nor}) \wedge log\_y(IS_{des}) \subseteq log\_y(IS_{nor})$

where $log\_y(r, IS_x)$ denotes the events log associated to the resource r in the (Normative or Descriptive) information system, with $y = in \mid out$ and $x = des \mid nor$. $log\_y(IS_{des}$ represents the set of events logs of the(Normative or Descriptive) information system.

In the same manner, the descriptive model IS_{des} will not be conformed to the normative model IS_{nor}, this is denoted by $IS_{nor} IS_{des}$, if and only if there exist a resource r in the Descriptive information system IS_{des} with an activity that has been executed by violating the security policy rule(s) in a periode of time $Period$ defined by $[T\_in, T\_out]$.

LEMMA 28.

Let $p1$ and $p2$ be to two Period; $p1 \models p2$ if $p1[T\_in] < p2[T\_in]$ and $p1[T\_out] > p2[T\_out]$.

An activity tk has been executed with the violation of the security policy if and only if one of the following constraints is satisfied:

$$\exists k \in E\_IS_{des} / \begin{cases} \text{Task (log\_x(r, IS}_{des})[k]) = tk \wedge tk \notin task\_x(r), \\ x = out \mid in \\ \text{P\_x(task (log\_x(r,IS}_{des})[k])) = pk \wedge pkP\_x(r), \\ x = out \mid in \\ \text{trigger((Task, IS}_{des})[k]) = tgk \wedge tgk \notin Tg(Task) \end{cases}$$

where:
- $Task(log\_x(r, IS_{des})[k])$ is the activity associated of the resource r that was recorded in the log_x through the Descriptive information system IS_{des} reported by the event k;
- $task\_x(r)$ is the set of activities that the resource r can initiated or support;
- $P\_x(task(log\_x(r, IS_{des})[k]))$ is the period of time in which activity runs;
- $P\_x(r)$ is the period of time that a resource r can run;
- $trigger((Task, IS_{des})[k])$ denote trigger for the Task associated to the event k in the Descriptive information system;
- $Tg(Task)$ is the set of triggers associated with all Task in IS_{des}.

DEFINITION 29.

Let IS denote an information system, r a resource and log_in and log_out it log files. The intrusion detection denoted by ID is defined as follow:
$ID = \bigcup_{i=1}^{m} (log\_in(r_i, IS_{des}) \setminus \text{log\_in}(r_i, IS_{nor}) \cup log\_out(r_i, IS_{des}) \setminus \text{log\_out}(r_i, IS_{nor}))$

THEOREM 30.

Let IS be an information system, if an event e of IS does not match it secure policy then, $e \in ID(IS)$.

DEFINITION 31.

Let IS_{nor} denote a normative information system,
$log_{x=in|out}(r, IS_{nor})$ = $\{x \mid x$ is an event of IS and x match the security policy $\}$.

LEMMA 32.

Let e be a event and SP the security policy of the IS. e is said to match SP if and only if $e = \langle Task, T\_in, T\_out, Resp\_res, Used\_res, Task\_out\rangle$ with the following condition:

$$\begin{cases} \text{Task can run if and only if it triggering condition is satisfied} \\ \text{Task can run if and only during a time period T\_in and T\_out} \\ \text{Resp\_res and Used\_res can be used by Task only during a time period T\_in and T\_out} \\ \text{The action of a task on Resp\_Ress or Used\_ress belong to Task\_act} \end{cases}$$

DEFINITION 33. *(Information system transition function)*

It is a function that allows events to transform a normative model to the associated descriptive model according to the security policy. This function is defined as follows:
$f : Event$ X $IS \rightarrow IS$ / If e match SP then $e \notin ID(IS)$ else $\in ID(IS)$

5. RELATED WORKS

Many modeling approaches have been developed in order to tackle the above defined problem. Some of the resulted models are based on genetic algorithms [14, 15, 16], while others are based on data mining [17, 18, 19]. Despite the popularity of these techniques, the proposed solutions are far from meeting the target as long as the information security is concerned. The above mentioned solutions do not take into consideration different types and abstractions of various resources that are found or act in the information system. Therefore, the associated solutions cannot efficiently handle the information system security policy in order to identify or prevent intrusion. It is possible to efficiently de tect or prevent intrusion when the description of every resource is defined. This allows defining the normative security model of the information system that should next be compared to the descriptive one in order to define the gap. This gap is called intrusion that needs to be detected or prevented. Based on the security policy representation and different resource models, the proposed approach in this paper deals with the definition of the so called gap.

6. CONCLUSION AND PERSPECTIVES

This paper provides a formal framework modeling intrusion detection within an information system. The workflow mining theory to be aware of the system evolution by analyzing its resources workflows events logs to detect intrusions. After presented two main approaches used for intrusion detection, one can defined some concepts manipulated in information systems to be able to set up the model used to detect intrusion especially thanks to events logs. Thus, using the proposed framework to detect attacks and then stoping them is not easy and it involves a deep knowledge in the domains of workflow mining and information systems security. Two challenge for future works are highlighted here. The first consists on the construction of an efficient tool based on the proposed model to handle intrusions within a given information system. The second deals with the definition of a digital forensic framework that could help to investigate after an attack in an information system detected by an intrusion detection engine.

7. REFERENCES

[1] Karthikeyan .K.R and A. Indra: "Intrusion Detection Tools and Techniques - A Survey". International Journal of Computer Theory and Engineering, Vol.2, No.6, December, 2010.

[2] Heady, R., G. Luger, A. Maccabe, and M. Servi lia: "The Architecture of a Network Level Intrusion Detection System". Technical report, Computer Science Department, University of New Mexico, 1990.

[3] Lunt, T.: "Detecting Intruders in Computer Systems". In: Proceedings of the 1993 Conference on Auditing and Computer Technology, 1993.

[4] Kumar, S. and E. H. Spafford: "A Software Architecture to Support Misuse Intrusion Detection". In: Proceedings of the lSth National Information Security Conference, 1995.

[5] A. Kartit, A. Saidi, F. Bezzazi, M. El marraki, A. Radi: "A new approach to intrusion detection system". Journal of Theoretical and Applied Information Technology, 29th February 2012. Vol. 36 No.2, ISSN: 1992-8645, www.jatit.org, E-ISSN: 1817-3195.

[6] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster: "Workflow Mining: Discovering Process Models from Event Logs". IEEE Transactions on Knowledge and Data Engineering (TKDE), Accepted for publication, 2003.

[7] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Marusterl, G.Schimm, and AJ.M.M. Weijters: "Workow Mining: A Survey of issues and Approaches". Data and Knowledge Engineering, Accepted for publication, 2003.

[8] Ilgun, K., R. A. Kemmerer, and P. A. Porras: "State Transition Analysis: A Rule-Based lntrusion Detection Approach". IEEE Transactions on Software Engineering 21(3), 1995.

[9] Jorg Becker, Patrick Delfmann: "Reference modeling: Efficient Information System Design Through Reuse of Information Models". Kindle Edition, Jul 2007.

[10] Atsa Etoundi Roger: "ATSERO Method: A Guideline for Business Process and Workflow Modeling Within an Enterprise ". International Journal of Scientific Engineering Research, Dec 2011.

[11] Bob Glushko: "Process Modeling for Information System Design". Oct 2008.

[12] Sandy Kemsley: "Business Process Modeling". TIBCO Software Inc, global headquarters, 3303 hillview avenue, Palo alto, ca 94304, 2011.

[13] Rafael Accorsi, Thomas Stocker, Gnter Mller: "On the Exploitation of Process Mining for Security Audits: The Process Discovery Case". SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing, Pages 1462-1468, ACM New York, NY, USA 2013.

[14] Mohammad Sazzadul Hoque, Abdul Mukit, Abu Naser Bikas: "An implementation of intrusion detection system using genetic algorithm". International Journal of Network Security and Its Applications (IJNSA), Vol.4, No.2, March 2012.

[15] Bharat S. Dhak, Shrikant Lade: "An Evolutionary Approach to Intrusion Detection System using Genetic Algorithm". International Journal of Emerging Technology and Advanced

Engineering. ISSN 2250-2459, ISO 9001:2008, Certified Journal, Volume 2, Issue 12, December 2012.

[16] A.A. Ojugo, A.O. Eboka, O.E. Okonta, R.E Yoro, F.O. Aghware: "Genetic Algorithm Rule-Based Intrusion Detection System (GAIDS)". Journal of Emerging Trends in Computing and Information Sciences, VOL. 3, NO. 8 Aug, 2012 ISSN 2079-8407.

[17] Qinglei Zhou, Yilin Zhao: "The Design and Implementation of Intrusion Detection System based on Data Mining Technology".Research Journal of Applied Sciences, Engineering and Technology 5(14): 3824-3829, 2013 ISSN: 2040-7459; e-ISSN: 2040-7467, Maxwell Scientific Organization, 2013.

[18] Yogita B. Bhavsar, Kalyani C.Waghmare: "Intrusion Detection System Using Data Mining Technique: Support Vector Machine " International Journal of Emerging Technology and Advanced Engineering, Website: www.ijetae.com, ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 3, Issue 3, March 2013.

[19] Parekh s.p, Madan b.s, Tugnayat r.m: "Approach for intrusion detection system using data mining", Journal of Data Mining and Knowledge Discovery ISSN: 22296662, and ISSN: 22296670, Volume 3, Issue 2, 2012, pp.-83-87.

IV

Mboupda Moyo Achille, Atsa Etoundi Roger: " **Obtaining Digital Evidence from Intrusion Detection Systems** ", International Journal of Computer Applications (0975 8887), Doi 10.5120/13973-1970, Volume 95 - No. 12, June 2014. www.ijca.org.

Indexed with EBSCO, Google Scholar, Informatics, ProQuest CSA Technology Research Database, NASA ADS (Harvard Univ.), CiteSeer, Ulrich Web, Scientific Commons (Univ. of St Gallens), University of Karlsruhe, Germany, PennState University

JUIN 2014

Obtaining Digital Evidence from Intrusion Detection Systems

Mboupda Moyo Achille
University of Yaounde I
Faculty of Sciences
Cameroon

Atsa Etoundi Roger
University of Yaounde I
Faculty of Sciences
Cameroon

ABSTRACT
Intrusion detection techniques have appeared to inspect all of the inbound and outbound network activities, and to identify suspicious patterns that indicate an attack that might compromise an information system. However, related information can be collected so as to supply evidence in criminal and civil legal proceedings. Several works have been carried out in the domain of Intrusion Detection and Prevention System (IDPS) but, none of the resulting models taking into account the possibility to collect intrusion related information in such a way that some of it can be turned in evidence in a proactive digital forensic purpose. In the literature, some authors have mentioned the possibility of re-designing IDPS as sources of evidence but, a formal model has never been proposed. This paper proposes an intrusion detection architecture for digital forensic purposes implemented using SNORT program.

Keywords
Intrusion detection and prevention system, Digital forensic, Cybercrime investigation

1. INTRODUCTION
This Many organizations and public administrations put in place today IDPS in order to prevent attacks on their information system. In fact, once an attacker intends to manipulate data into a computer system, he gathers information about the target computer, probe it for vulnerabilities and attempt to exploit them. After gain unauthorized access into the computer, the attacker escalates from an unprivileged account to privileged account. He hides tracks and instantiate a persistent reentry. Next, he can extend unauthorized access to other areas of the network and pursue goal of it intrusion which can include stealing information or destroying data.

Largely used, the IDPS appears to be a countermeasure which produces satisfactory results. The IDPS simplifies the task of detecting attacks quite before the actual attack by tracing the trails that the attacker leaves while gathering intelligence about a network. They can be passive (in this case they can only give on alert) or active (when

in addition to alerting, they can react against attacks). The IDPS is the method of identifying unauthorized use, misuse and abuse of computer systems by both system insiders and external attackers. Basically, there are three steps in the process of intrusion detection and prevention which can be decline to monitoring and analyzing traffic; identifying abnormal activities; assessing severity and raising alarm [1,2].

When an IDPS detects an intrusion, it will actually log the event, store relevant data or traffic, notify an administrator and in some cases it will intervene. If it is exploited, the consequently stored data and the logs can be valuable forensic information that may be used as evidence in a legal case against the attacker. In fact, forensic computing appears to identify, preserve, analyze and present digital evidence in a manner that is legally acceptable or accepted in a court. Digital forensic is not a single activity, but draws upon many disciplines [3,4]. It involves the application of information technology to the search for digital evidence either by media and electronic device analysis, network intrusion or misuse detection, or data interception.

The IDS/IDPS becomes today a very useful source of information about an attack. However, they are not originally designed to collect and preserve the integrity of the type of information required to conduct law enforcement investigations. In the course of numerous digital forensic operations, it has been establish that IDS are useless to the investigators whereas, they constitute an important source of information. This is due to the fact that the integrity and the authenticity of information that come thereof are not guaranteed [5]. To face this difficulty, it is necessary to put in place a means of data collection (following a chain of custody) which can produce the first aspects of inquiry in case of investigation. Yuill et al [6] state that IDS can collect enough information during an on-going attack to profile or to identify the attacker. Our aim in this paper is to provide a model of IDS, combined to digital forensic primitives which can proactively or actively brings out relevant information about an attack without materially affecting it primary mission.

The rest of the paper is organized as follows: section 2 outlines the previous works; section 3 describes the proposed IDS architecture and gives some experimental results; section 4 deals with the conclusion and highlighting some perspectives as future works.

2. PREVIOUS WORKS
Over the past years, there have been some controversies about the applicability of IDS to the forensic evidence collection process. Two aspects have essentially emerged. The first one views forensic evidence collection and preservation in the case of a computer or network security incident to be inappropriate for an intrusion detection system. Another perspective submits that the IDS are the most likely candidate for collecting forensically pristine evidentiary data in real or near real time. The main idea was to know whether it was possible to use intrusion detection systems to gather forensic evidence in the case of a detected penetration or abuse attempt [7,8]. Several authors have tried to find relevant contribution to this idea. Many authors have mentioned the possibility to

rebuilt IDS in such a way that its output can serve as evidence in a court of law.

[9] address IDS and a view to its forensic applications. They view a forensic application within the framework of intrusion detection and detail the advantages and disadvantages of some IDS. They point that IDS are the places to look for evidence during an investigation process besides Hard drives, Memory, System logs, Email servers, Network traffic.

[6] puts in place a formal descriptive method named Investigative Intrusion-Detection. They show that IDS can collect enough information during an on-going attack to profile, if not to identify, the attacker. The ability of IDS to gather significant information about an attack in progress without materially affecting the primary mission of the intrusion detection system suggests that IDS could be deployed that would provide both detection/response and forensically pristine evidence in the case of a security incident. They focus on attacker activities concerning what he has done, what he can do, what he does, what he knows, what he wants, and what identifies him.

[10] states that although the main aim of IDSs is to detect intrusions to prompt evasive measures, a further aim can be to supply evidence in criminal and civil legal proceedings. However the features that make an IDS product good at providing early warning may render it less useful as an evidence acquisition tool. But, he gives direction and condition to Re-designing IDSs as sources of evidence before concluding that if logs are to be produced from IDS tools, a prosecutor must be prepared to disclose complete details of the tool, and how it was configured and operated.

[7] describes a project which reviews the performance and forensic acceptability of several types of intrusion detection systems in a laboratory environment. He develops a theoretical model and architecture for an intrusion detection system that can also perform forensic tasks. This theoretical model also concerns the case of host based intrusion detection systems.

[11] states that IDS belong to the set of log records along the path. They show that log records contain a substantial amount of content that may be relevant in a criminal case. The log records may reveal identity information that connects the activity to user attributes, including the IP address used and the type of operating system, browser, and applications of the computer user. Logs are timestamp-centric, making them ideal for filling in time line gaps in an investigation. But they also address the fact that log records, like other forms of electronic evidence, can be modified by a third party, but they precise that it would be highly improbable that all the log records along the path of transmission could be altered because each of the devices creating log records would have to be compromised to some degree. This brings out the fact that proofs from IDS are not sufficient enough to accountability; they must be backed by proofs from other sources.

[12] proposed a digital forensic investigation process model including proactive, active and reactive processes. They claimed that this model can be used in a proactive way to identify opportunities for the development and deployment of technology to support the work of investigators, and to provide a framework for the capture and analysis of requirements for investigative tools, particularly for advanced automated analytical tools. The authors implemented a digital forensic model which is divided into three components: The Proactive digital forensics (ProDF) component, the Reactive digital forensics (ReaDF) component and the Active digital forensics (ActDF) component. So doing, they stated that the ProDF component is the ability to proactively identify, collect, gather an event, preserve and analyze evidence to detect an incident as it eventually occurs.

In addition, an automated documentation is generated for a later investigation by the active and reactive components. The evidence that will be gathered in this component is the proactive evidence that relates to a specific event or incident as it occurs. The ProDF as described in [12] can be efficiently associated to IDS to ensure the integrity of evidence and preserve it in a forensically sound manner. Furthermore, the analysis of the evidence will be done in such a way that it can enable prosecution of the suspect and admission to the court of law. Phases under the proactive component fall into Alert, Identification, Collection, Preservation, Analysis and Documentation.

3. INTRUSION DETECTION SYSTEM AND DIGITAL FORENSIC: MODEL BUILDING

3.1 Output of IDS

Depending on the precise IDS, its outcomes can include [10]:
- The skill to react in a promptly manner to prevent or to reduce substantive damage by automatic or manual intervention;

- The skill to identify an attacker or an activity which can cause more serious attack;

- The skill to discover new attack patterns or as a preventive measure, to provide an additional measure of system protection beyond that available from other forms of security measure.

During our experimentation, it appears that SNORT saves many messages under */var/log/snort* direction. These messages contain relevant information about an incident whenever it occurs, depending on some specified rules indicated in Snort source code. Information concerns Time/date, Source IP address, Destination IP address, Time to Live (TTL) value in the IP packet header, the Type Of Service (TOS) value in the IP packet header, length of IP packet header, total length of IP packet, ICMP Type field, ICMP code value, IP packet ID, Sequence number, ICMP packet type [13].

Unfortunately, the repository where SNORT keeps relevant data is not secure. The data integrity can easily be compromised by an attacker. Furthermore, IDS evasion techniques can also be used to compromise data or to make IDS inefficient.

3.2 Requirement of evidence in court

Evidence is used to establish the truth of a particular fact or state of affairs. Generally, evidence has to satisfy tests of admissibility and weight. For admissibility, evidence must conform to certain legal rules which are applied by a judge [14]. For weight, evidence must be understood by, and be sufficiently convincing to the court, whether there is a jury or a judge acting as a trier of fact. Before a court, evidence can be real, testimonial, documentary, expert or derived [9]. Therefore, to be accepted in a court, there should be a clear chain of custody or continuity of evidence and the forensic method used needs to be transparent, that is, freely testable by a third party expert. Anyway, before a court, the prosecutors

need to demonstrate that an information system was involved, it was accessed, such accessing was unauthorized access knew at the time that the access was unauthorized. Nevertheless, a clear chain of custody will be respected if the following basic principles for evaluating the acceptability of evidence as describe by [10] is applied:

- Authentic: the evidence should be specifically linked to the circumstances and persons alleged, and produced by someone who can answer questions about them. Unless a party shows that the evidence is what that party claims it to be, the court will view the evidence as irrelevant.

- Accurate: the evidence should be free from any reasonable doubt about the quality of procedures used to collect the material, analyze the material if that is appropriate and necessary, and finally, to introduce it into court – and produced by someone who can explain what has been done.

- Complete the evidence should be able to tell, within its terms, a complete story of a particular set of circumstances or events.

Among all, other sources can be used to support some given evidence. These can be extract in Firewall Logs, Web Server Access Logs, Simple Mail Transfer Protocol / Internet Message Access Protocol Servers (email), FTP Servers (file transfer protocol), Proxy Server Logs, Secure Shell Servers (remote access), Routers and Switches, Chat Servers, DNS Servers (Domain Name System), Victim and Attacker Systems.

3.3 Bridge between IDS outputs and DF evidence

From an evidential point of view, what one looks for is something one can demonstrate to others long after the event itself is over. IDS provide it through logs of various kinds. These include system, audit, application and network management logs. Other sources of potential evidence are network traffic capture and contemporaneous manual entries [15,16]. However, the derived data can be split into a form in which it is easier to analyze and understand. Otherwise, to be admissible in a court of law, the collection of potential evidence will respects a chain of custody.

Thus, outcomes of IDS will be efficient enough to persuade a third party. So doing, logs issued by IDS which intend to provide relevant information for digital forensic purpose must respect the following specification [10]:

- the logs may not have been compromised during or prior to collection as potential evidence and during post collection analysis;

- in the case of real-time network, monitoring the network location of the device hosting the monitoring tool may be such that it is able to capture all relevant traffic, some of the packets using other routes;

- in the case of real-time monitoring, the monitoring tool may be able to keep up with the stream of traffic with which it is expected to deal;

- the logs may sufficiently distinguish between a legitimate and an unwanted access; the logs may exist over a sufficient period of time for comparisons of normal and abnormal activities to be made;

- the logs may be helpful to identify the perpetrator in any useful way, complete for the relevant period of time, rich in detail;

- the logs will gather relevant information.

Hence, the contribution the IDS can make in case of prosecution is to prove that an information system was involved and was accessed. Then, digital forensic will subsequently bring out sufficient legal evidence and will allow to investigate in order to identify the perpetrator. One should note that current IDS are not fully designed to collect and protect the integrity of all information require to conduct investigations in respect of law enforcement. Basically, there are two broad categories of analysis performed to look for signs of intrusion. The first is misuse detection. It works by looking for known indications of misuse, whereas the basis for allowable activity is specified in the security policy of an organization. The second type of analysis performed is anomaly detection. It works by defining parameters for normal activity for a given set of resources [17,18]. This defined normal activity becomes a baseline against which all activity is measured. Actions falling outside the scope of normal activity are flagged as anomalous for investigation as potential security violations.

The implementation of the conditions listed above allows us to define a model of detection (Fig.1). This model is primarily supported on the basic model of intrusion detection but has the particularity of being able to produce information that can serve as digital evidence.

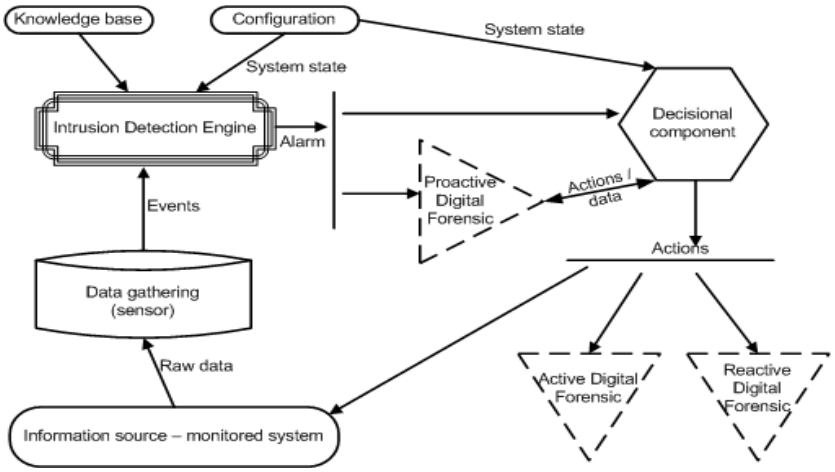


Fig. 1. The basic IDS architecture for digital forensic purpose

The proposed model in figure 1 above fall in 9 components which are described as follow:

- Sensor: it is a data gathering device which is responsible for collecting raw data from a monitored system;

- Intrusion Detection Engine: this engine processes the data collected by sensors to identify intrusive activities;

- Knowledge base: it contains information collected by the sensors, but in preprocessed format such as knowledge base of attacks and their signatures, filtered data, data profiles. This information is usually provided by network and security experts;

- Configuration device: it provides information about the current state of the intrusion detection system;

- Proactive digital forensics (ProDF) component: it allows to ensure successful cost of effective digital investigations with minimal business activity disruption and ensuring that admissible evidence and sound processes are in place and available when needed for an investigation or as required during the normal flow of business [12,19]. Each time an alarm is triggered, this component start the collection of all information related to the intrusive activity. It actively safeguards the integrity of collected information and preserves it in a forensically sound manner;

- Decisional component: it initiates actions when an intrusion is detected. These responses can either be automated or involve human interaction;

- Reactive digital forensics (ReaDF) component: it targets the traditional digital forensic investigation that will take place after an incident had been detected and confirmed. This involves identifying, preserving, collecting, analyzing, and generating the final report. This module is active when an attack could not be detected via the intrusion detection engine;

- Active digital forensics (ActDF) component: it allows to gather (identify, collect, analyze and preserve) receivable digital evidence in a live environment to facilitate a successful investigation. When the alert is enabled, the response component triggers the ActDF component throughout the duration of the attack.

3.4 Experimental setting: the place of the IDS in Network Topology

Our experimental device consists of one router and six workstations. The router is connected to the internet and the workstations are set to the local network. In order to detect only external intrusion activities, the intrusion detection system was placed directly inside the router as shown in figure 2.

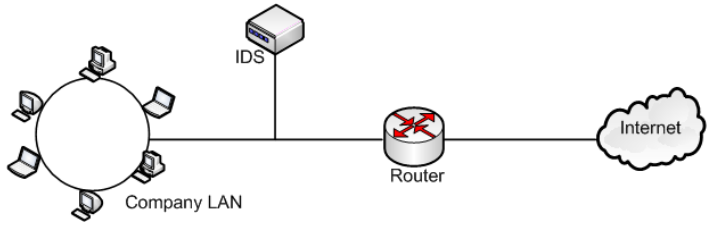


Fig. 2. Experimental network architecture

3.5 Experimental setting: the updated architecture of SNORT

Snort is known to be a powerful application. This software is free and it can run either on Linux or Windows environments. Understanding the functioning of the internal components of Snort helped us to customize it to our network and helped us to avoid some of the common Snort pitfalls. Snort can be divided into five major components that are each critical to intrusion detection (Fig.3). The first is the packet capturing mechanism. Snort relies on an external packet capturing library to sniff packets from the backbone. After packets have been captured in a raw form, they are passed into the packet decoder. The decoder is the first step into Snort's own architecture. The packet decoder translates specific protocol elements into an internal data structure. Once the initial preparatory packet's capture and decode is completed, traffic is handled by the preprocessors. Any number of pluggable preprocessors either examines or manipulates packets before handing them to the next component: the detection engine. The detection engine performs simple tests on a single aspect of each packet to detect intrusions. The last component is the output plug-in, which generates alerts to lay out suspicious activities. In order to collect digital evidence, some codes have been added in the *snort.conf* file. Therefore, the *snort.conf* file have been implemented in such a way that once the alert is triggered, the incriminated packets are simultaneously preserved in the binary database of snort logs files, and converted to serve as input to the Proactive Digital Forensic component. Figure 3 shows a simplified graphical representation of the dataflow.

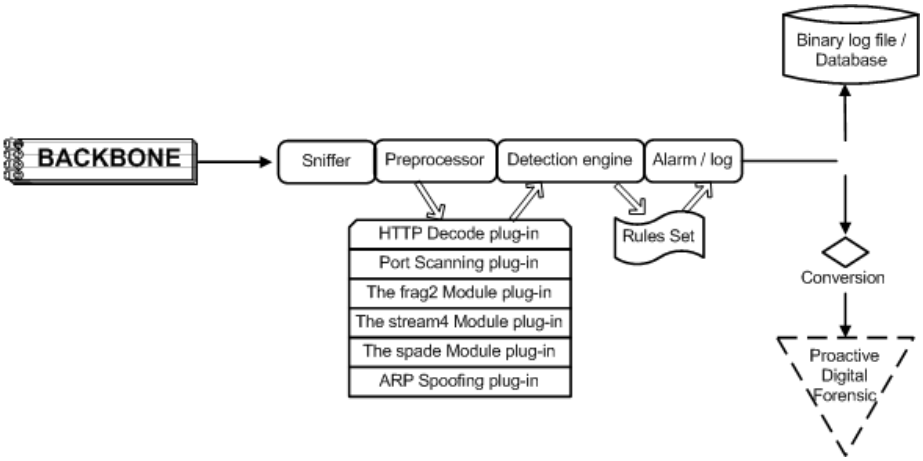


Fig. 3. SNORT architecture for digital forensic purpose

3.6 Experimental setting: the algorithms for a new implementation of *snort.conf* file

Snort is a lightweight but powerful tool for detecting malicious traffic on a given network. With a flexible and robust rules definition language, Snort is capable of detecting nearly any threat that crosses the network. However, reporting is not its strength. It records tens or hundreds of thousands of suspicious events every day on a busy network. Snort has been made valuable by reviewing and acting on the data it produces. So doing, using the following algorithms derived from the Proactive Digital Forensic component [12], the *snort.conf* file have been modified in such a way that it can produce evidence.ids as another output.

Algorithm1 Identification algorithm
*REQUIRE A set of traffic transmitted by the detection engine of Snort /*an Alert file*/;*
ENSURE All information related to an incident;
begin
initialisation;
repeat
select an alert;
extract all relevant information that characterizes this alert;
create or update the temporary identification file;
until (there are no more alerts)
end

Algorithm2 Collection algorithm
REQUIRE temporary identification file
ENSURE temporary collection file
begin
initialisation;
repeat
select an alert in the temporary identification file;
extract source IP address;
retrieve information associated with the source IP address;
make a record referenced by the source IP address;
create or update the temporary collection file;
until (the end of the temporary identification file)
end

Algorithm3 Preservation algorithm
REQUIRE temporary collection file
ENSURE temporary collection file
begin
initialisation;
protect temporary collection file;
save temporary collection file;
end

Algorithm4 Analysis algorithm
REQUIRE temporary collection file, temporary identification file
ENSURE temporary analysis file,
begin
initialisation;
identify the rule that has triggered the alert;
categorize the attack;
indicate the nature of the attack;
specify the IP source address;
specify the IP destination address;
indicate the connection ports used;
indicate the timestamp;
indicate the protocols used;
update the temporary analysis file;
end

Algorithm5 Documentation algorithm
REQUIRE temporary analysis file
ENSURE evidence.ids
begin
initialisation;
sort the temporary analysis file by type of attack, source IP, destination IP, protocol, port, and time stamp;
create or update evidence.ids file;
end

Algorithm6 Proactive Digital Forensic /(ProDF) Documentation algorithm */*
REQUIRE evidence.ids
ENSURE evidence.ids
begin
initialisation;
run identification;
run collection;
run preservation;
run analysis;
run documentation;
update evidence.ids;
end

The *evidence.ids* file that is outputted by ProDF component contains digital evidence, while *Alert file* is the set of derived useful information that constitute a chain of evidence (Time/date, Source IP address, Destination IP address, Time to Live (TTL) value in the IP packet header, the Type Of Service (TOS) value in the IP packet header, length of IP packet header, total length of IP packet, ICMP Type field, ICMP code value, IP packet ID, Sequence number, ICMP packet type).

3.7 Results

To complete our experience, a Honeypot have been deployed in our network to prosecute hackers by gathering evidence of their activities. It is a system used to lure hackers by exposing known vulnerabilities deliberately. The honeypot had among others some services running on it such as Telnet server (port 23), Hyper Text Transfer Protocol (HTTP) server (port 80), File Transfer Protocol (FTP) server (port 21) and others. It was placed somewhere so that the hackers could easily take it for a real server, using an IP address very close to the real server. Attacks recorded at the end of this experience have enabled us to achieve many log files. Figure 4 is a snapshot of one of these log files. The detection of Christmas or XMAS tree attack was the focused case study. A Christmas tree attack sends a large number of Christmas tree packets to an end device. A Christmas tree packet has all the options set so that any protocol can be used. It require much more processing by routers and end devices than other packets.

Large numbers of these packets can use up so much processing power that it ties up these devices effectively making any other task nearly impossible thus denying service to legitimate traffic. Receiving these types of packets is not usual and therefore should be regarded as suspicious. Intrusion detection systems can detect these packets as do some firewalls. An XMAS scan, is a port scan typology with flags set to Fin, Push and Urg at the same packet. The SNORT output file translated with *tcpdump* is shown in Figure 5 below.

```
[cc lang="VHDL"]
tcpdump: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
08:15:22.748471 IP (tos 0×0, ttl 37, id 45231, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.108.33434 > 192.168.1.101.369: Flags [FPU], cksum 0x65ec (correct), seq 30145115847, win 2541, urg 0, length 0
08:15:22.755673 IP (tos 0×0, ttl 121, id 57432, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.101.369 > 192.168.1.108.33434: Flags [R.], cksum 0x87ff (correct), seq 0, ack 30145115848, win 0, length 0
[/cc]
```

```
[cc lang="VHDL"]
09:21:35.873683 IP (tos 0×0, ttl 27, id 50647, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.108.33434 > 192.168.1.101.369: Flags [FPU], cksum 0xdf4a (correct), seq 2649036211, win 2541, urg 0, length 0
09:21:35.886578 IP (tos 0×0, ttl 121, id 51626, offset 0, flags [none], proto TCP (6), length 30)
192.168.1.101.369 > 192.168.1.108.33434: Flags [R.], cksum 0xec4d (correct), seq 0, ack 2649036212, win 0, length 0
[/cc]
```

Fig. 4. SNORT output files

In this figure, the current version of SNORT detects an attack which source IP address is 192.168.1.108. The attacker launches an XMAS attack on the destination IP address 192.168.1.101, where the flags FPU are activated.

```
[cc lang="VHDL"]
snort2.9.0.6: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
[**] [021:02:1] spp_stream5: STEALTH ACTIVITY (nmap XMAS scan) detection [**]
08:15:22.748471 192.168.1.108:33434 -> 192.168.1.101:369
TCP TTL:42 TOS:0x0 ID:55954 IpLen:20 DgmLen:40
**U*P**F Seq: 0x0 Ack: 0x0 Win: 0xC00 TcpLen: 20 UrgPtr: 0x0
```

```
[cc lang="VHDL"]
snort2.9.0.6: listening on wlan0, link-type EN10MB (Ethernet), capture size 65535 bytes
[**] [033:03:2] spp_stream5: STEALTH ACTIVITY (nmap XMAS scan) detection [**]
09:21:35.873683 192.168.1.108:33434 -> 192.168.1.101:369
TCP TTL:42 TOS:0x0 ID:55954 IpLen:20 DgmLen:40
**U*P**F Seq: 0x0 Ack: 0x0 Win: 0xC00 TcpLen: 20 UrgPtr: 0x0
```

Fig. 5. SNORT output files after applying ProDF algorithms

In this figure, the updated version of SNORT, using Proactive Digital Forensic derived algorithms, reveal a STEALTH ACTIVITY (which is XMAS Scan) on a target computer at 192.168.1.101. Immediately, a file of evidence is built and an instance of this file is given in Figure 6 below.

```
Attack    No.    Time          Source              Destination    Prot Port Info
Xmas_scan 224512 08:15:22.748471 192.168.1.108.33434>192.168.1.101  TCP  80   369 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 224512 09:21:35.873683 192.168.1.108.33434>192.168.1.101  TCP  80   369 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 224512 18:10:22.234231 41.205.86.200>192.168.1.101        TCP  22   45125 > ssh [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 225121 19:15:01.324561 77.175.26.168>192.168.1.101        TCP  80   45631 > http [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 225530 19:25:15.325467 79.233.134.80>192.168.1.101        TCP  53   48123 > domain [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 226910 20:05:18.348854 82.83.205.73>192.168.1.101         TCP  443  48032 > https [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
Xmas_scan 227011 20:29:09.426495 83.163.68.56>192.168.1.101         TCP  111  49569 > sunrpc [FIN,PSH,URG] Seq=0 Ack=0
Win=1024 Urg=0 Len=0
```

Fig. 6. SNORT output files containing evidences

The file of evidence is sorted by type of attack, sequence number, time, IP source, IP destination, protocol, port and other relevant information. The process of collecting these information in respect of the chain of custody, unsure the integrity of the data which can be use in case of legal inquiries. The table I below gives a short description of information found in the SNORT output files:

Table 1. Logs file description.

No.	Field or Activity	Context/Notes
1	08:15:22.748471	This is the timestamp of the request, it was made on h 08H 15min 22 sec pm
2	IP	This are all IP (protocol) related settings
3	tos 0x0	Type of service field
4	ttl 37 which is time to live	Number of hops that the packets have to reach its destination. This indicate throw how many routers the packets should pass, this is for not living the packets travel the net for ever
5	id 45231	In a case of hijacking (such as man in the middle attack), the attacker should be able to hack the packet ID and present as a response a packet with the same ID but with malicious data
6	proto TCP	It is the protocol type. It can be some times UDP or ICMP
7	length 30	The length of the TCP packet
8	192.168.1.108.33.434	It is the source IP address and 33434 is the port used by the hacker
9	192.168.1.101.369	It is the destination IP address (The honeypot IP address) and 369 is the port used
10	Flags [FPU]	It is the TCP flag FPU (Fin, Push or Urg) when running an XMAS scan. It could be [S] to mean an ACK reply from the honeypot, or [R] which means RESET and in this case the connection is reseted, or [F] for finishing a transfer, etc.
11	cksum 0x65ec	This is the TCP-header check sum of the packet (for checking packet integrity)
12	seq 3014515847	The TCP sequence number
13	win 2541	The amount that will send before requiring a response from the server
14	urg	The urgency

As the illustration shows, log records contain a substantial amount of content that may be relevant in a criminal case. The log records may reveal identity information that connects the activity to user attributes, including the IP address used and the type of operating system, browser, and applications of the computer user. Logs are timestamp-centric, making them ideal for filling in time line gaps in an investigation.

3.8 Discussion

The aim of this section is to evaluate the behaviors of the modified IDS. As part of this experiment, a free version of Snort has been used. Thereafter, the architecture of Snort has been changed by implementing ProDF component through the algorithms presented above. The duration of the execution of the IDS in both cases is presented in the tables below.

Table 2. Running Snort without ProDF component.

Test number	Number of receive packets	Number of alert	Ratio (Packet/sec)	Number of packets captured by Snort
1	3445263	76	300	3445112
2	9655422	102	500	9655315
3	2712657	52	200	2712645
4	6845795	151	350	6845710
5	8932698	134	400	8932624

Table 3. Running Snort within ProDF component.

Test number	Number of receive packets	Number of alert	Ratio (Packet/sec)	Number of packets captured by Snort
1	3445263	76	300	3445112
2	9655422	100	500	9655311
3	2712657	52	200	2712645
4	6845795	151	350	6845713
5	8932698	136	400	8932621

In the first case, the relevant information to the investigation was housed in the default backup directory of Snort. In the second case, the evidence is found in the evidence.ids file. It is a safe file containing data obtained in accordance with a chain of custody for the preservation and collection of digital evidence. Observing the number of packets received the number of alarms and the number of captured packets in both cases, the gap is negligible. This proves that the IDS Snort although its structure has been modified to output admissible digital evidence, has not seen its performance deteriorate as a tool for detecting intrusions.

4. CONCLUSION AND FUTURE WORKS

In this paper, it has been established that the IDS could be used as input to a digital forensics door. To carry out this study, a detailed research and cataloging of prior formal work in forensics and intrusion detection was performed. Next, the

general impact of forensic evidence management on IDS was presented. After analyzing and updating the basic intrusion detection system model, a combined model for intrusion detection in a forensic environment using the multiperspective cybercrime investigation process model was theorized. The designed architecture for IDS in a forensic environment using SNORT has been experimented and it has been showed how log files can be exploited in a forensic purpose. The results obtained in this paper are limited to a Network Intrusion Detection System (NIDS) environment. Be able to generalize a theory that supports intrusion detection and digital forensics in the same system remains a significant challenge. IDS can help investigators during a digital forensic process, but computing forensic cannot rely solely on the IDS otherwise, these would be subject to acute changing that could undeniably devious them to their primary goals.

5. REFERENCES

[1] Aleksandar Lazarevic, Vipin Kumar, and Jaideep Srivastava.: Intrusion Detection: A Survey, (2005).

[2] Biswanath Mukherjee, L. Todd Heberlein, and Karl N. Levitt.: Network Intrusion Detection. IEEE Network 8, 3, 26–41, (1994).

[3] Rodney McKemmish.: What is Forensic Computing? Australian Institute of Criminology. http://books.google.pt/books?id=NoqGmgEACAAJ, (1999).

[4] Eoghan Casey.: Digital Evidence and Computer Crime, 3rd Edition, Forensic Science, Computers, and the Internet. Academic PressPrint Book, Baltimore, USA, (2011).

[5] George M. Mohay, Alison Anderson, Byron Collie, Rodney D. McKemmish, and Olivier de Vel.: Computer and Intrusion Forensics. Artech House, Inc., Norwood, MA, USA, (2003).

[6] Jim Yuill, Shyhtsun Felix Wu, Fengmin Gong, and Ming-Yuh Huang.: Intrusion Detection for an On-Going Attack.. In Recent Advances in Intrusion Detection (2002-01-03). http://dblp.uni-trier.de/db/conf/raid/raid1999.html#YuillWGH99, (1999).

[7] Peter Stephenson.: The Application of Intrusion Detection Systems in a Forensic Environment. Executive Office for United States Attorneys, Vol. 59. United States, Department of Justice, Washington, DC 20530, (2011).

[8] Golden G. Richard, III and Vassil Roussev.: Next-generation Digital Forensics. Commun. ACM 49, 2 (Feb. 2006), 76–80. DOI:http://dx.doi.org/10.1145/1113034.1113074, (2006).

[9] Thomas Scaria Nathan Balon, Ronald Stovall. : Computer Intrusion Forensics. (2004).

[10] Peter Sommer.: Intrusion detection systems as evidence. Computer Networks 31, 2324, 2477 – 2487. DOI:http://dx.doi.org/10.1016/S1389-1286(99)00113-9, (1999).

[11] Mark L. Krotoski and Jason Passwaters.: Obtaining and Admitting Electronic Evidence. Executive Office for United States Attorneys, Vol. 59. Washington, DC 20530. http://www.justice.gov/usao/eousa/foia reading room/usab5906.pdf, (2011).

[12] Roger Etoundi Atsa and Achille Moyo Mboupda.: Multi-perspective Cybercrime Investigation Process Modeling. International Journal of Applied Information Systems 2, 8 (June 2012), 14–20. Published by Foundation of Computer Science, New York, USA, (2012).

[13] Rafeeq Ur Rehman.: Intrusion Detection Systems With Snort: Advanced IDS Techniques Using Snort, Apache, MySQL, PHP, And ACID. Prentice Hall PTR, Upper Saddle River, N.J. http://isbnplus.org/9780131407336, (2003).

[14] Kristin M. Finklea and Catherine A. Theohary.: Cybercrime: conceptual issues for congress and U.S. law enforcement. United States, Department of Justice, Washington, DC 20530, (2013).

[15] P.Lakshmi Prasanna D.R.Lavanya K.Rajasekhar, B.Sekhar Babu and T.Vamsi Krishna.: An Overview of Intrusion Detection System Strategies and Issues. InternatIonal Journal of Computer Science and technology 2, 4 (December 2011).

[16] Eoghan Casey.: Network traffic as a source of evidence: tool strengths, weaknesses, and future needs. Digital Investigation 1, 1 (2004), 28 – 43. DOI:http://dx.doi.org/10.1016/j.diin.2003.12.002, (2004).

[17] Eugene H. Spafford and Diego Zamboni.: Data Collection Mechanisms for Intrusion Detection Systems. Technical Report. Cerias, Purdue University, 1315 Recitation Building, (2000).

[18] Andrew Case, Andrew Cristina, Lodovico Marziale, Golden G. Richard, and Vassil Roussev.: FACE: Automated digital evidence discovery and correlation. Digital Investigation 5, Supplement, 0 (2008), S65 – S75. DOI:http://dx.doi.org/10.1016/j.diin.2008.05.008 The Proceedings of the Eighth Annual DFRWS Conference, (2008).

[19] Talania Grobler, C. P. Louwrens, and Sebastian H. von Solms. 2010. A Multi-component View of Digital Forensics. In ARES (2010-03-22). IEEE Computer Society, 647–652. http://dblp.uni-trier.de/db/conf/IEEEares/ares2010.html#GroblerLS10