Adversarial vs behavioural-based defensive AI with continual and joint learning: automated evaluation of robustness to deception, poisoning and concept drift

Alexandre Dey[2], Marc Velay[1], Jean-Philippe Fauvelle[1], Sylvain Navers[1]

[1] Airbus Defence & Space Cybersecurity, 1 boulevard Jean Moulin 78990 Élancourt
[2] Airbus Defence & Space Cybersecurity, 3 rue Louis Braille 35136 Saint-Jacques-de-la-Lande

jean-philippe.fauvelle@fr.airbus.com

Abstract. Recent advancements in AI have brought new capabilities to behavioural analysis (UEBA) for cyber-security consisting in the detection of hostile action based on the unusual nature of events observed on the Information System. In our previous work (presented at C&ESAR 2018 and FIC 2019), we have associated deep neural networks auto-encoders for anomaly detection and graph-based events correlation to address major limitations in UEBA systems. This resulted in reduced false positive and false negative rates, improved alert explainability, while maintaining real-time performances and scalability. However, we did not address the natural evolution of behaviours through time, also known as concept drift. To maintain effective detection capabilities, an anomaly-based detection system must be continually trained, which opens a door to an adversary that can conduct the so-called "frog-boiling" attack by progressively distilling unnoticed attack traces inside the behavioural models until the complete attack is considered normal. In this paper, we present a solution to effectively mitigate this attack by improving the detection process and efficiently leveraging human expertise. We also present preliminary work on adversarial AI conducting deception attack, which, in term, will be used to help assess and improve the defense system. These defensive and offensive AI implement joint learning, in a step that is necessary in assessing, validating and certifying AI-based defensive solutions.

Keywords: Artificial Intelligence, User and Entity Behavior Analytics (UEBA), Reinforcement learning, Continual learning, Joint learning, Event correlation, Adversarial AI, Deception and Poisoning of AI, Validation and Assessement of AI, Defensive and Offensive Cyber Warfare, Lutte Informatique Défensive (LID), Lutte Informatique Offensive (LIO).

1 Introduction

Les outils actuels de surveillance de la sécurité des systèmes d'information (SI) reposent sur des règles statiques de détection conçues par capitalisation et maintenues postérieurement à l'observation de nouvelles cyber-attaques. L'évolution et la complexité croissante des menaces surchargent les différents niveaux d'expertise opérant des systèmes de détection. Cette expertise rare devrait être focalisée sur des tâches à haute valeur ajoutée plutôt que dispersée sur des tâches répétitives chronophages.

De nombreux travaux récents consacrés à la détection d'intrusion dans un SI, reposent sur l'analyse du comportement des utilisateurs et des entités (UEBA en anglais). Le procédé UEBA le plus souvent mis en œuvre repose sur des algorithmes d'apprentissage machine pour construire des modèles de comportements normaux puis inférer, pour chaque évènement observé, un score d'incongruité qui quantifie la déviance du comportement observé relativement au modèle de normalité. Cette approche est avantageuse : elle rend possible la détection d'attaques non encore observées et permet une adaptation graduelle aux évolutions du SI protégé.

Cependant, les solutions mettant en œuvre cette approche UEBA concèdent certaines limitations, principalement un nombre élevé de faux positifs, une sursimplification des problèmes entraînant des faux négatifs, un passage à l'échelle difficile dans certains cas, un effet « boîte noire » inhérent au caractère faiblement explicable des alertes, et enfin une adaptabilité insuffisante à l'évolution naturelle des comportements au cours du temps ou « dérive conceptuelle ».

Dans nos précédents travaux [1] nous avons présenté des avancées associant un procédé UEBA avec une phase de corrélation d'évènements en graphes, permettant de nous affranchir des limites susmentionnées, à l'exception de la dérive conceptuelle. Cette dérive conceptuelle peut être contournée par la mise en œuvre d'un mode d'apprentissage continu. Cependant, un système de détection basé sur les anomalies et l'apprentissage continu se trouve exposé à deux menaces.

La première découle du besoin d'adapter en continu les modèles comportementaux. Un attaquant pourrait tromper un système de détection apprenant en continu [2, 3, 4] en disséminant progressivement les traces de son attaque jusqu'à ce que cette dernière soit considérée comme normale par l'outil de détection [5, 6]. Ce type d'attaque par empoisonnement est appelé attaque de la « grenouille ébouillantée » [7].

La seconde menace, commune à tout système de détection d'anomalies, consiste à leurrer la défense en présentant une activité discrète lors de la phase offensive. Un attaquant connaissant le type de défense mis en œuvre sur le SI visé pourrait imiter un comportement normal afin de leurrer les défenses. C'est notamment l'approche pratiquée par les APT (*Advanced Persistant Threat*) pour créer des attaques complexes. Ce type d'approche est appelé « attaque adversaire » et peut être automatisée via un apprentissage par renforcement [27, 28].

En prolongement de nos travaux exposés lors de la Conférence C&ESAR 2018 [21] puis du FIC 2019, nous présentons dans cet article, en premier lieu, un système défensif de détection de cyber-attaques conçu pour résister aux deux menaces susmentionnées, puis en second lieu, nos premiers travaux concernant une IA adversaire menant des attaques contre une IA défensive pour en valider l'efficacité.

2 Principes généraux

Procédé UEBA et dérive conceptuelle — Un procédé UEBA comprend généralement une phase d'apprentissage construisant un modèle des comportements habituels (**Fig. 1 : pastille 12**) des constituants d'un SI, et une phase d'inférence comparant chaque comportement au modèle pour en mesurer le caractère inhabituel et potentiellement hostile. Le plus souvent, la phase d'apprentissage est menée une fois pour toute, puis le modèle demeure statique : un comportement pourra alors être perçu normal ou hostile, à tort ou à raison selon l'obsolescence du modèle. Ce phénomène de dérive conceptuelle peut être adressé en instaurant un apprentissage continu, mais ceci expose alors le système aux attaques par empoisonnement [33], où un attaquant mène des actions d'apparence normale qui tendent progressivement vers l'attaque complète, jusqu'à ce que cette dernière soit intégrée dans les modèles comportementaux et ne puisse plus être détectée.

Leurrage d'une IA — Le leurrage d'un système de détection d'anomalies consiste à dissimuler le caractère hostile d'un ensemble d'actions en mimant un comportement normal. L'approche la plus commune consiste à privilégier des actions considérées comme des signaux faibles et entre lesquels il est difficile d'établir une corrélation.

Ethical hacking — Le *ethical hacking* est une pratique repandue dans le domaine de la cyber-sécurité, opposant deux équipes d'experts, l'une nommée *blue-team*, en charge de défendre un SI, et l'autre nommée *red-team*, en charge de l'attaquer.

Cyber Kill Chain — Ce concept, largement accepté par des acteurs gouvernementaux et industriels depuis 2011 [36], divise une cyber-attaque en sept phases successives dont chacune comprend ses propres enjeux et contraintes et peut être mitigée par des défenses spécifiques.

IA adversaire / offensive — Par généralisation et automatisation du *ethical hacking*, une IA adversaire également appelée IA offensive est opposée à une IA défensive, dont elle tente de cerner et contourner les limites [22], ce principe étant applicable à des systèmes logiques ou physiques [29].

Il existe principalement trois approches, la 1ère étant l'apprentissage en parallèle des modèles offensifs et défensifs [32], chacun ayant accès à des informations sur son opposant, qui lui permettent d'améliorer sa propre approche. Leurs optimisations dépendent de la valeur de l'erreur de classification du modèle défensif : ce dernier est entrainé à minimiser l'erreur, tandis que l'adversaire a pour objectif de la maximiser.

Une 2nde approche consiste à opposer le modèle adversaire à une défense statique dont les classifications sont connues [27] [28] : le problème est résolu en apprenant par renforcement, en fonction de récompenses dont les valeurs sont liées à la réussite ou à l'échec de chaque action.

En 3ème approche, le modèle adversaire est opposé à un environnement complexe simulé : l'apprentissage ne dépend pas d'un retour direct du système défensif mais d'une mesure de l'effet des actions de l'adversaire sur l'environnement.

3 Approche retenue

3.1 Contraintes UEBA

Notre recherche d'un procédé UEBA est principalement guidée par ces contraintes :

- Minimiser le nombre de faux positifs par rapport aux solutions existantes ;
- Maximiser le caractère explicable des alertes remontées à l'analyste, c'est-à-dire fournir à ce dernier les informations lui permettant de comprendre les différentes étapes de l'attaque et ce qui les relie certainement ou probablement ;
- Minimiser le temps et l'expertise nécessaires au déploiement et au fonctionnement du procédé ;
- Maximiser la robustesse du procédé aux attaques par empoisonnement, afin de contrevenir aux actions d'un attaquant. À titre d'exemple, [18] propose une variante robuste de l'algorithme PCA (*Principal Component Analysis*).

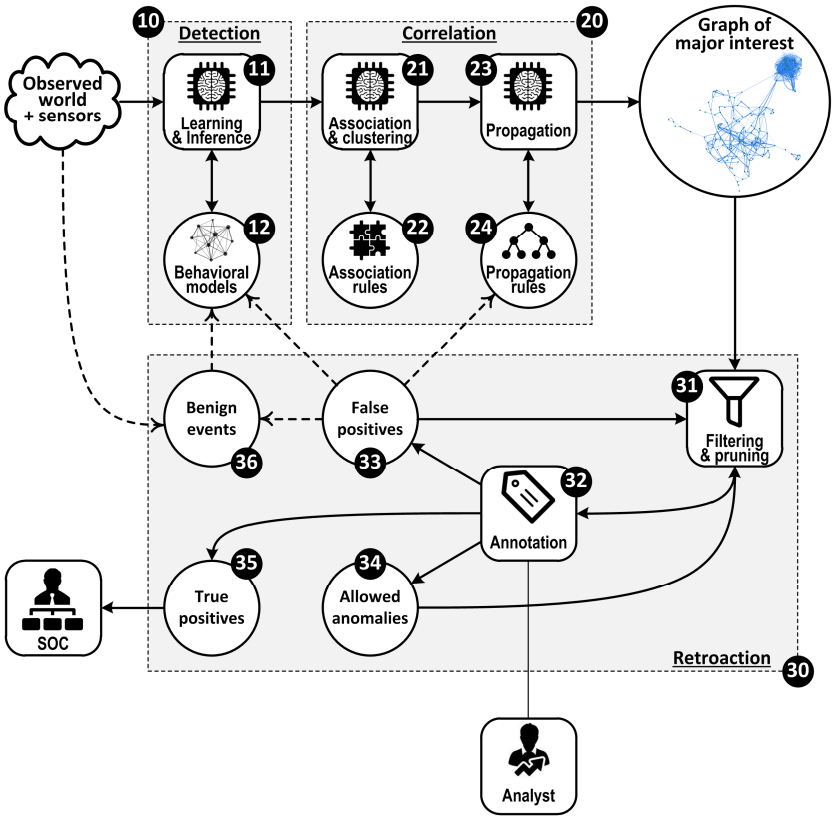
Annotation du niveau d'anomalie — Collecter et maintenir une base de données des attaques visant spécifiquement un système à surveiller est particulièrement difficile. La méthode qui semble la plus viable pour servir cet objectif est la détection d'anomalies comportementales. Dans cette voie, les deux exigences majeures sont l'adaptation aux différents types de variables (ex. : nombres, catégories, séries temporelles) qui existent dans les événements de sécurité {**EX1**} et la production d'alertes pourvues d'un caractère explicable {**EX2**}.

Les réseaux de neurones auto-encodeurs sont utilisables pour générer des plongements (*embedding*) [34] pour un grand nombre de formats de données, ce qui répond à la 1ère exigence **{EX1}**. À titre d'exemple, des architectures CNN (*Convolutional Neural Network*) sont spécialisées dans le traitement d'images et des architectures LSTM (*Long Short Term Memory*) dans le traitement d'informations séquentielles. Les plongements résultants existent dans un espace vectoriel où les distances usuelles (ex. : distance Euclidienne) sont pertinentes, propriété exploitée dans les étapes suivantes (ex. : regroupement d'événements).

Les auto-encodeurs (**Fig. 1 : pastille 11**) sont entrainés à reproduire les données d'entrée après une phase de compression. Les anomalies deviennent détectables en raison d'une forte erreur de reconstruction des données par l'auto-encodeur. Cette erreur de reconstruction est particulièrement élevée pour les variables responsables de l'anomalie (ex. : partie correspondant à l'exploitation dans une requête web malveillante). L'algorithme répond donc également à la 2nde exigence **{EX2}**.

De plus, la capacité de passage à l'échelle des réseaux de neurones rend ce procédé intéressant pour du calcul au fil de l'eau. Ces calculs peuvent également être réalisés sur des architectures matérielles spécialisées (ex. : GPU, TPU, FPGA, ASIC). Cependant, l'entraînement d'un auto-encodeur nécessite un large corpus de données.

Fig. 1. — Synoptique du procédé de défense comprenant trois groupes fonctionnels : la détection (pastille 10), la corrélation (pastille 20), la rétroaction (pastille 30).



Identification d'attaques — Un attaquant conduit son opération en plusieurs étapes, produisant une multitude d'évènements. Une anomalie isolée ne peut être considérée à elle seule comme une attaque, et l'ajout d'un score d'incongruité à chaque évènement capté n'est pas suffisant pour détecter une attaque ni a fortiori la reconstruire.

Notre approche pour détecter et reconstruire de manière explicable une opération hostile s'appuie sur la corrélation d'évènements représentés en graphe, cette technique étant communément considérée comme la plus performante selon l'état de l'art [12].

Construction d'un graphe — Chaque évènement capté sur un SI est porteur à minima des informations relatives à l'acteur (ex. : processus, machines), à l'action (ex. : lecture, écriture, appel système), au moment (horodatage), à la localisation (ex. : site, équipement, application) et parfois aux ressources sollicitées (ex. : fichier). À partir de ces éléments, nous construisons des graphes temporels acycliques orientés, constitués de nœuds (ex. : acteurs, ressources) reliés par des arêtes unidirectionnelles (actions), cette représentation étant appropriée pour modéliser un flux d'information.

Identification et regroupement des structures fréquentes — Du fait de la diversité des capteurs et de la complexité des SI, un unique phénomène peut déclencher une cascade d'évènements concomitants (ex. : démarrage d'une application suivi du chargement de ses dépendances), ou des évènements répétés similaires (ex. : chargement d'un fichier bloc par bloc). Cette abondance d'évènements augmente la taille des graphes et par conséquent la complexité algorithmique spatiale et temporelle des procédés qui les manipulent. Nous adressons cette problématique en regroupant les évènements (**Fig. 1 : pastille 21**), soit par mise en œuvre de règles d'association (**pastille 22**), dans le cas des évènements concomitants, soit par *clustering* réalisé par un réseau de neurones auto-encodeurs, dans le cas des évènements répétés similaires.

Propagation du niveau d'anomalie — Le recours aux réseaux de neurones auto-encodeurs nous procure une quantification du niveau d'anomalie de chacune des variables des événements. Cependant, certaines actions sont plus susceptibles que d'autres de caractériser des anomalies sur un type de variable (ex. : un navigateur et un interpréteur de commandes communiquent parfois vers des adresses IP inconnues, mais le navigateur est plus souvent impliqué). Lors de la création des modèles de propagation du niveau d'anomalie (**Fig. 1 : pastille 24**), nous favorisons les transferts rares et pénalisons les plus fréquents, afin de privilégier les liens incongrus.

Mise à jour des modèles de comportements — Notre méthode repose sur différents modèles construits par des algorithmes d'apprentissage à partir des données collectées sur le SI. Ces algorithmes sont efficaces pour détecter des anomalies mais non pour les interpréter, cette tâche étant dévolue à un analyste, pour lequel il convient de simplifier les représentations et d'abstraire le fonctionnement sous-jacent du procédé.

Présentation à l'analyste — En présence d'un graphe dit d'intérêt majeur, constitué d'événements liés, et dont le niveau d'anomalie est élevé[1], nous élaguons (**Fig. 1 : pastille 31**) les chemins les moins intéressants i.e. dont le niveau d'anomalie ne fait que décroitre. Ce graphe, qui correspond à une alerte présumée, est alors comparé aux anciennes alertes statuées faux positifs regroupées dans une base (**pastille 33**). En présence de similitudes notables, le graphe d'intérêt majeur est abandonné et aucune alerte n'est émise. Dans le cas contraire, le graphe est présenté à l'analyste en même temps que les logs liés aux évènements pour faciliter les investigations de l'analyste.

Sélection des évènements — L'analyste porte la responsabilité d'annoter (Fig. 1 **: pastille 32**) chaque alerte (vrai ou faux positif, action dangereuse mais autorisée). En présence d'un vrai positif (**pastille 35**) une alerte est envoyée au SOC (*Security Operations Center*). En présence d'un faux positif, les évènements sont insérés dans une base des faux positifs (**pastille 33**) destinée au filtrage. En présence d'un comportement dangereux mais autorisé (ex. : actions d'un administrateur), ce dernier est répertorié dans une base des anomalies autorisées (**pastille 34**). Lorsque des faux positifs similaires se multiplient, ils contribuent à l'apprentissage. En outre, notre procédé

[1] Les modalités décidant qu'un graphe est d'intérêt majeur ne peuvent pas être divulguées.

comprend une base de connaissances des événements bénins passés (**pastille 36**) i.e. des données normales destinées à l'entrainement. Cette base fait office de mémoire épisodique (i.e. rejouer des évènements passés marquants pour éviter leur oubli trop rapide) ou permet de reconstruire les modèles en cas de nécessité. Elle est alimentée par échantillonnage aléatoire sur le flux continu des évènements normaux (i.e. ayant un faible score d'anomalie). Elle est également enrichie par les faux positifs que l'on souhaite réintroduire dans les modèles. Une fois mis au point le procédé de défense, nous étudierons la possibilité d'annoter automatiquement les alertes, sans analyste.

3.2 Validation automatisée

Interaction avec l'environnement — L'IA adversaire utilisée pour valider l'IA de défense mime l'approche courante d'un test d'intrusion, hors étape d'ingénierie sociale. L'IA adversaire est positionnée comme un acteur externe qui accéderait aux ressources internes du SI ciblé puis tenterait d'identifier les vulnérabilités.

Afin de conserver le caractère générique du procédé, l'interfaçage avec l'environnement est réalisé via un terminal et des *sockets* réseau. Pour manipuler cet environnement, l'IA adversaire sélectionne soit le programme et ses options, soit le protocole et le contenu à transmettre via le réseau. L'espace des combinaisons d'actions est trop vaste à explorer. Pour en réduire la profondeur, nous recourons à un dictionnaire de commandes de bas niveau. Cette faible granularité permet à l'IA adversaire de générer une importante diversité de combinaisons d'actions dont les signatures s'éloignent de celles produites par les outils populaires de cyber-sécurité.

Pour chacune des sept phases constituant la *cyber kill chain,* un agent adversaire spécialisé pour ladite phase est entrainé à partir d'un dictionnaire spécifique. L'orchestration des agents demeure pour le moment séquentielle, ce qui permet à un agent d'utiliser les informations déjà produites par d'autres agents. Nous réfléchissons à la mise en œuvre simultanée et collaborative de l'ensemble des agents.

Scénarios de validation — Pour valider le fonctionnement des défenses face à diverses attaques, nous avons conçu trois scénarios présentés ci-après, dans lesquels les agents adversaires sont entrainés conformément à la 3eme approche présentée au §2 :

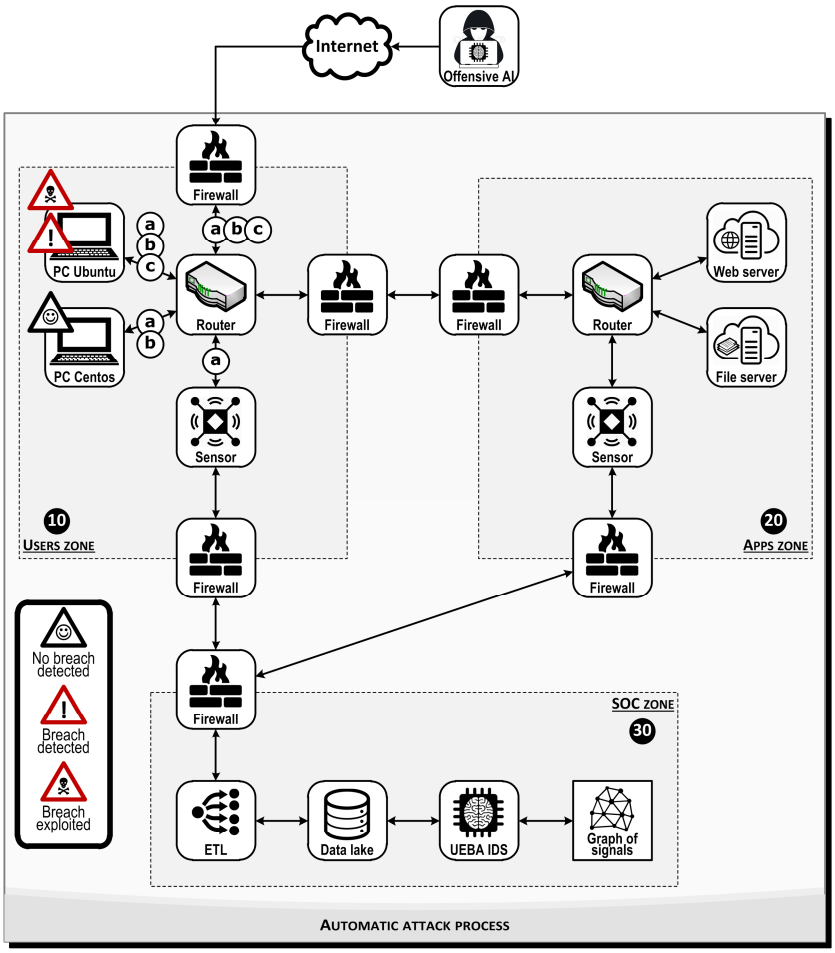
- Le 1er scénario permet de valider la résistance à une attaque classique. Il prend place dans un SI simulé (**Fig. 2**) comprenant plusieurs DMZ (zones démilitarisées). La zone « utilisateurs » (**Fig. 2 : pastille 10**) est constituée d'ordinateurs bureautiques, la zone « applicatifs » (**pastille 20**) de serveurs hébergeant les applications de l'entreprise, et la zone « SOC » (**pastille 30**) de l'infrastructure nécessaire à un SOC. Ces trois zones sont équipées de sondes réseau collectant des informations sur les communications et les transmettant aux agrégateurs de la zone SOC. Dans un contexte d'apprentissage machine, ce SI générique est nommé « environnement ». Notre système défensif est installé dans la zone SOC, mais n'est pas activé. En effet, l'intérêt de ce scénario est de déterminer si des éléments internes sont exposés par erreur et si des vulnérabilités courantes

sont présentes. Ce scénario est l'équivalent d'un test d'intrusion classique, superficiel, mais nécessaire pour rapidement découvrir des failles évidentes.

— Le 2nd scénario a pour objectif de valider le bon fonctionnement du système de détection d'attaque. Notre système défensif, préalablement entrainé sur une activité normale générée synthétiquement, mais n'apprenant plus, est actif et analyse le comportement des utilisateurs et entités du SI. Dans ce contexte, l'IA adversaire essaye de leurrer les défenses afin de ne pas être détectée pendant sa découverte et son exploitation du SI.

— Le 3eme scénario entend valider la résistance à l'empoisonnement de la mémoire de notre système défensif, actif et apprenant en continu. L'IA adversaire essaye de corrompre graduellement la représentation d'un comportement normal.

Fig. 2. — Théâtre et déroulement du premier scénario offensif (version initiale).



Dans ces trois scénarios, l'IA adversaire explore différentes stratégies. L'exploration est une phase d'adaptation par renforcement où l'agent détermine le

moyen le plus efficace d'attaquer le SI. Une méthode heuristique est utilisée pour amener l'IA adversaire à analyser la plus grande surface possible de l'environnement, en termes de cibles et de techniques offensives. L'efficacité des agents adversaires est mesurée via le taux de couverture (surface d'attaque explorée rapportée à la surface existante) et l'efficacité (quantité d'actions pour réaliser une attaque).

La **Fig. 2** illustre le théatre (environnement plus IA) et le déroulement du 1er scénario susmentioné. L'IA adversaire débute son opération à l'extérieur du SI, puis obtient un accès via une interface dédiée aux employés travaillant à distance, enfin explore la zone « utilisateurs » (**pastille 10**). L'approche suit les phases de la *cyber kill chain* : découverte du réseau puis identification et exploitation de vulnérabilités. Le flux (**pastille a**) illustre la découverte, lorsque l'IA identifie les cibles potentielles. Dans l'exemple, deux ordinateurs bureautiques et une sonde sont découverts. Lors de la seconde phase (**pastille b**), l'agent adversaire analyse les deux postes utilisateurs pour déceler des vulnérabilités courantes, par exemple des services dont la version correspondrait à une entrée dans la base CVE [35]. Enfin, le flux (**pastille c**) illustre la mise en oeuvre d'un exploit sur la cible affectée d'une vulnérabilité identifiée. Suite à ce processus, l'IA signale la présence d'un problème à l'équipe défensive.

Évaluation de l'environnement — L'apprentissage des agents adversaires nécessite une quantification de la réponse de l'environnement aux actions réalisées. De plus, la planification des prochaines étapes de l'opération nécessite l'extraction d'informations dans un format que les algorithmes peuvent exploiter. Ces deux tâches sont réalisées par analyse des réponses aux requêtes. Chaque requête possède un format spécifique de sortie dont nous pouvons tirer parti pour extraire des informations, par exemple les adresses IP et les ports ouverts des cibles, les versions des logiciels ou encore la confirmation de lancement d'un terminal.

Ces premières informations brutes permettent aux modèles de sélectionner l'action suivante la plus adaptée. Elles permettent également d'évaluer le fonctionnement des agents sur plusieurs critères. Ces critères sont définis en fonction de l'objectif d'apprentissage désiré pour la phase.

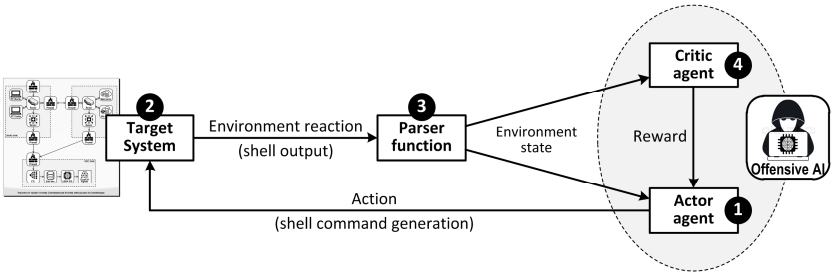
À titre exemple pour la phase d'identification des vulnérabilités, le taux de couverture de l'analyse et sa discrétion sont les deux critères majeurs. Le taux de couverture est obtenu en comparant les informations découvertes par rapport aux informations connues relatives à une infrastructure maîtrisée.

L'évaluation résulte en un score que l'algorithme d'apprentissage par renforcement cherche à maximiser. Un certain nombre d'algorithmes fonctionnent sur un score normalisé entre 0 et 1, mais d'autres ne sont pas limités par une valeur maximale.

Apprentissage par renforcement — L'algorithme d'apprentissage par renforcement de tâches complexes A3C (*Asynchronous Actor-Critic Agent*) permet d'explorer un espace d'actions continu ou discret. Sa structure reprend l'approche des GAN [32] en combinant deux structures disjointes. L'agent acteur choisit une action suivant une politique dépendante de l'environnement. L'agent critique analyse l'effet sur

l'environnement des actions entreprises et calcule un score. L'acteur met à jour sa politique d'action pour améliorer ce score.

Fig. 3. — Synoptique du procédé d'apprentissage machine des agents adversaires.

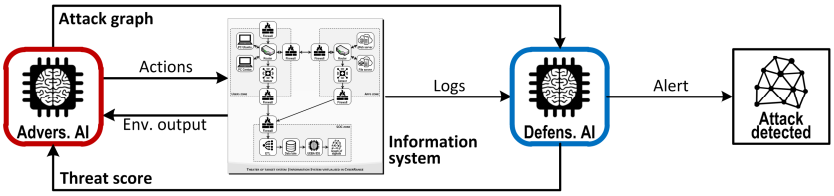


L'apprentissage est un cycle d'essais, d'analyse et d'adaptation. Un agent « acteur » (**Fig. 3 : pastille 1**) construit une commande à partir du dictionnaire à disposition, puis l'exécute dans son interface vers le SI ciblé (**pastille 2**). Le message en sortie de cette commande est analysé (**pastille 3**) et les informations en sont extraites automatiquement. Ces informations constituent l'état actuel de l'environnement. L'agent « critique » (**pastille 4**) utilise les données analysées précédemment pour évaluer l'acteur. En prenant en compte le nouvel état de l'environnement et son score, l'acteur modifie sa politique de génération de commandes pour l'améliorer, puis le cycle recommence pour continuer l'opération.

Des répliques du modèle, dans des environnements identiques mais séparés, peuvent être exécutées simultanément. Chaque acteur réalise une suite d'actions jusqu'à atteindre une condition d'arrêt. L'apprentissage peut être réalisé par lots (*batch*), en combinant des séquences d'actions. L'algorithme combine les différentes expériences des modèles afin d'en tirer des leçons plus générales. Cette approche assure un passage à l'échelle horizontal des agents, tout en évitant un sur-apprentissage.

Apprentissage joint — L'apprentissage joint (*joint learning*) consiste à asservir autour d'un SI l'entraînement des deux IA, défensive et adversaire : l'adversaire exécute des actions sur le SI, produisant des évènements qui sont collectés puis alimentent l'IA défensive ; cette dernière reconstruit des graphes d'actions, dotés d'un score d'incongruité, en corrélant des événements. Au cours de l'apprentissage, les deux IA se transmettent leurs sorties respectives : l'adversaire transmet au défenseur son graphe d'actions (*attack graph*) tandis que l'IA défensive transmet son score d'incongruité (*threat score*) à l'adversaire (**Fig. 4**).

Fig. 4. — Synoptique d'un procédé d'apprentissage joint.



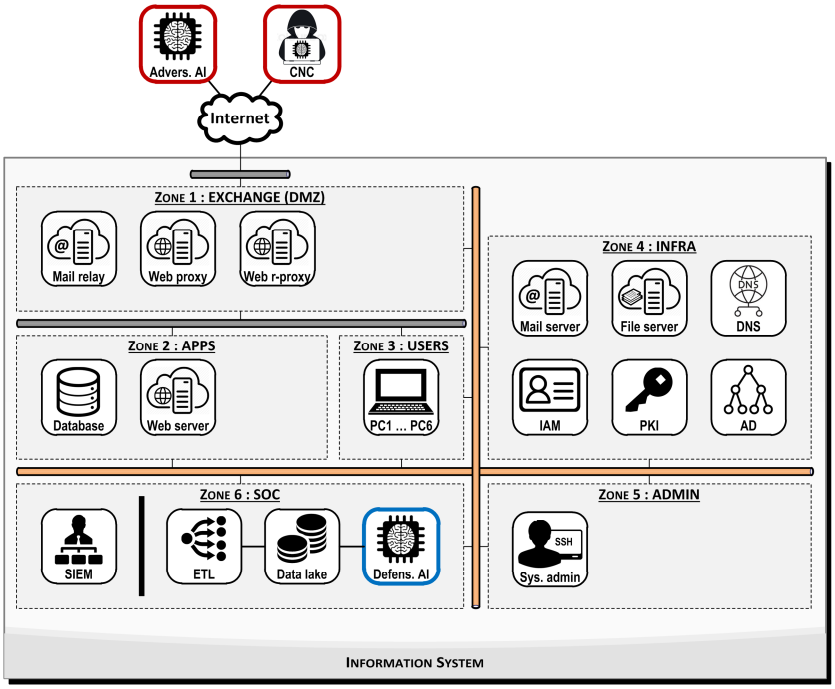
La finalité d'un tel procédé est de réduire l'erreur de reconstruction des graphes et minimiser le score d'incongruité. En théorie, cette approche améliore le fonctionnement de la défense et permet à l'adversaire de développer une stratégie de leurrage dont l'effet se mesure par la dégradation du score de détection au fil de l'attaque.

4 Expérimentation

4.1 Théâtre

En raison de limitations constatées sur le 1er scénario et son théâtre présentés au §3.2, nous avons fait évoluer ces derniers. Le théâtre accueillant nos procédés de défense et d'attaque, découpé en six zones constituées d'éléments virtualisés sur une plateforme CyberRange[2], est représentatif d'un SI assez commun, moyennement protégé, mis en œuvre par une entreprise de taille moyenne (**Fig. 5**).

Fig. 5. — Théâtre virtualisé accueillant les IA adversaire et défensive de l'expérimentation.



Nous employons un logiciel d'émulation d'entrées/sorties clavier et souris qui simule des utilisateurs inter-opérant avec le SI, afin de générer un trafic de vie massif et réaliste que nous utilisons pour l'entrainement de l'IA défensive.

[2] Solution de modélisation et de simulation avancées, conçue par Airbus CyberSecurity.

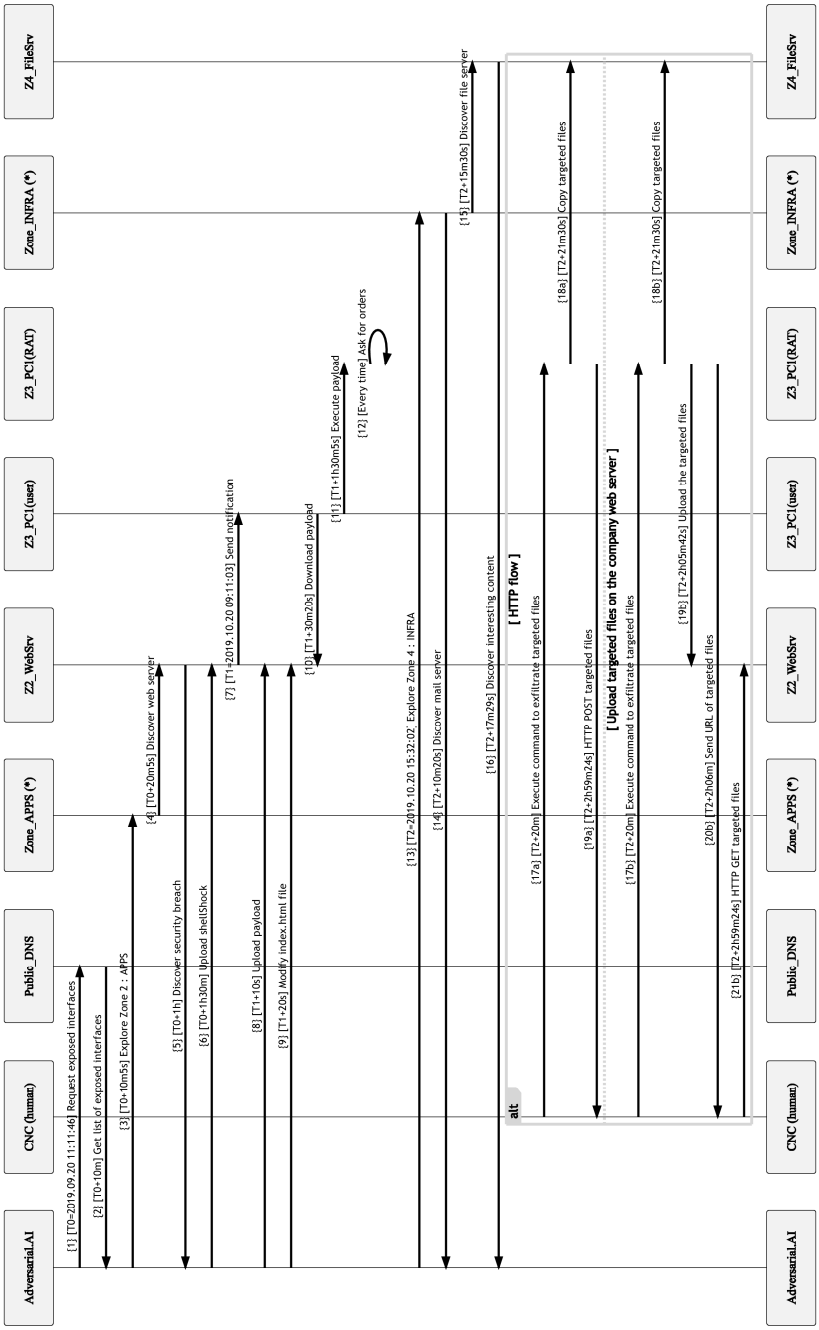
4.2 Scénario d'attaque

Afin d'évaluer les capacités de détection de l'IA défensive confrontée à l'IA adversaire, et de s'assurer qu'un faux négatif serait imputable à l'efficacité de l'adversaire plus qu'à un manquement du défenseur, notre scénario d'attaque se conforme au modèle de la Cyber Kill Chain (CKC). Dans un premier temps, l'attaque est jouée manuellement via des outils standards, puis elle est rejouée en mêlant actions manuelles et actions automatisées par l'IA adversaire (**Table 1** ; **Fig. 6**).

Table 1. — Étapes du scénario d'attaque.

Étape de la CKC	**Étape manuelle**	**Étape automatisée par l'IA adversaire**
Reconnaissance	Découverte du site de vente en ligne (VEL) exposé sur internet, et identification des vulnérabilités ; les utilisateurs externes ont possibilité de mettre en vente des produits et d'y associer une image ; le type des fichiers téléversés (*upload*) sur le site n'est pas vérifié ; le serveur est vulnérable à une attaque reposant sur la vulnérabilité *shellshock* (**Fig. 6 : {1 à 5}**).	
Armement	Création de la charge utile (*payload*) prenant le contrôle d'une page réservée aux modérateurs du site, laquelle est visitée systématiquement par les modérateurs pour valider la mise en vente des produits.	
Livraison	Téléversement de la charge utile, laquelle traverse le *reverse-proxy* car elle ne correspond à aucune signature connue. En conséquence, le site envoie un courriel à un modérateur, lui demandant de valider la mise en vente (**Fig. 6 : {6, 7}**).	
Exploitation	Exécution de la charge grâce à *shellshock* ; une vulnérabilité du navigateur (CVE-2018-8495) du modérateur est exploitée ; la machine du modérateur est infectée (**Fig. 6 : {8 à 11}**).	
Installation	Téléchargement et installation d'un RAT (*Remote Access Tool*) pour réaliser les actions ultérieures de l'attaquant (**Fig. 6 : {12}**).	Téléchargement et installation d'un binaire contenant l'IA d'attaque (**Fig. 6 : {11}**).
Commande & Contrôle	Cartographie du réseau interne avec l'outil standard *nmap* ; découverte du site web interne, d'un serveur de base de données et des dossiers partagés (**Fig. 6 : {13 à 15}**).	Cartographie automatisée par l'IA ; découverte du site web interne, d'un serveur de base de données et des dossiers partagés (**Fig. 6 : {13 à 15}**).
Actions sur objectifs	Exfiltration des données convoitées situées dans les dossiers partagés vers le serveur de contrôle (**Fig. 6 : {17a, 18a, 19a}**).	Téléversement sur la page infectée des données convoitées ; récupération depuis l'extérieur, via une connexion d'apparence anodine vers le site de VEL (**Fig. 6 : {17b à 21b}**).

Fig. 6. — Diagramme de séquence auto-généré par l'IA adversaire déroulant son attaque. (*) Une flèche unique vers une zone résume un lot d'actions explorant ladite zone.



4.3 Stratégie adversaire

Notre IA adversaire exécute une centaine d'actions dans un environnement connu et maitrisé. Afin de préparer une attaque à haute vélocité, chaque action est scorée en fonction des découvertes qui en résultent, et, en particulier, l'attente (choix de ne rien faire) est pénalisée. L'exploration est favorisée en privilégiant les séquences courtes et une forte variation de cibles. L'analyse en détail des cibles est accomplie par des séquences spécifiques d'actions : les informations résultantes sont favorisées au détriment de celles obtenues aisément. Finalement, une stratégie visant à maximiser la discrétion est plus complexe à élaborer. Elle se base sur des hypothèses heuristiques visant à augmenter la difficulté d'être détectée, par exemple en récompensant des méthodes passives d'exploration reposant sur une fréquence d'action réduite.

Le mécanisme actuel de récompense favorise au contraire une stratégie d'analyse davantage focalisée sur la précision des résultats. Pour toutes les cibles, l'agent adversaire identifie l'état des ports réseau (ouvert, fermé, filtré) et les services à l'écoute. En dépit du nombre réduit de types d'actions possibles et de vulnérabilités identifiables, la stratégie de l'IA adversaire est modulable par le mécanisme des récompenses. L'apprentissage mis en œuvre permet d'atteindre un score nettement supérieur à celui d'un agent aléatoire, ce qui conforte la pertinence de l'approche.

Dans le cadre de l'apprentissage joint, l'IA adversaire a accès au score d'anomalie calculé par l'IA de défense. En incluant la minimisation de ce score dans ses objectifs d'apprentissage, l'IA d'attaque choisit les actions qui semblent les moins anormales lors de sa phase exploratoire. Une fois les capacités de l'IA adversaire étendues à d'autre étapes de la CKC, il sera envisageable de duper le système de détection en exploitant ses angles morts, permettant à terme de corriger ceux-ci.

4.4 Volet défensif

Nous avons implémenté partiellement l'écosystème défensif présenté en Fig. 1 en priorisant la facilité de configuration du procédé de détection. À ce jour, l'analyste humain reste en charge de déterminer les types de données intéressants (ex. : chemin d'un exécutable, succès ou non d'une tentative de connexion) et les indications qui en facilitent le prétraitement (ex. : schéma récurrent dans les noms de fichiers temporaires, noms d'applications générant beaucoup d'événements sans réel intérêt pour la détection). À terme, cette étape de configuration sera accélérée par recours à des techniques de visualisation et de fouille de données plus avancées.

Les résultats obtenus par notre système de reconstruction des graphes des scénarios d'attaques sont conformes à nos attentes. Le regroupement des structures fréquentes est rapide (50.000 événements/s sur un processeur standard) et réduit drastiquement (deux ordres de grandeur) la dimension du graphe sans pour autant perdre d'information, préservant ainsi la possibilité d'une fouille manuelle des données lors de l'annotation ultérieure par l'analyste. Notre système permet donc l'utilisation de techniques poussées d'analyses de chacune des connexions du graphe, qui seraient trop gourmandes en temps de calcul sur le graphe original. Cette analyse en détail permet d'éliminer les chemins rares à l'échelle du système, lequel enregistre des cen-

taines de milliers d'évènements par seconde, mais néanmoins régulièrement observés à l'échelle humaine. Une fois ce nettoyage effectué, il ne demeure que quelques sous-graphes correspondant aux alertes, qui sont alors présentés à l'analyste sans saturer sa capacité de travail.

Les premiers résultats de notre système de rétroaction sont positifs, notamment en ce qui concerne sa capacité d'apprentissage incrémental des réseaux de neurone auto-encodeurs et sa vitesse d'entrainement du procédé de reconstruction. Néanmoins, notre système de rétroaction n'étant pas finalisé, ne peut donc être évalué exhaustivement. Ceci fera l'objet de futurs travaux.

5 Conclusion

Notre procédé est conçu pour analyser d'importants volumes d'événements de sécurité, au fil de l'eau, en profondeur et sans règles prédéfinies, en évaluant le caractère inhabituel de groupes d'évènements corrélés, afin de produire une alerte explicable prenant la forme d'un graphe reconstruisant un scénario d'attaque.

Un tel procédé défensif typé UEBA étant naturellement sensible au leurrage, nous avons conçu un écosystème dans lequel un procédé d'IA adversaire s'oppose à un procédé d'IA défensive, dans une approche dérivée de l'apprentissage joint.

Le système d'attaque identifie rapidement les vulnérabilités de la défense facilitant ainsi leur correction. La création de cet écosystème est d'autant plus complexe que nous développons conjointement les deux systèmes IA.

L'objectif poursuivi à moyen terme est triple : d'une part, permettre à chaque IA de contribuer à améliorer l'autre, d'autre part, de pallier au moins en partie la difficulté de collecter des données d'attaque représentatives de notre modèle de menace, et enfin, de vérifier et qualifier le fonctionnement de notre système défensif afin d'étendre notre méthodologie à d'autres solutions ou produits.

En particulier, ce dernier apport consistera à identifier les faiblesses d'un système défensif, car si le succès d'une attaque adversaire démontre logiquement une lacune de défense, a contrario l'échec d'une attaque ne prouve rien puisque il peut être imputé aux qualités du défenseur autant qu'à l'insuffisance de l'adversaire.

Nos travaux futurs nous amèneront à nous pencher sur l'épineuse problématique de la qualification et la certification de l'IA adversaire si cette dernière doit elle-même être utilisée pour qualifier voire certifier des procédés défensifs.

6 Références

1. J.-P. Fauvelle, A. Dey et S. Navers, « Protection of an information system by artificial intelligence: a three-phase approach based on behaviour analysis to detect a hostile scenario », CoRR, 2018.
2. G. I. Parisi, R. Kemker, J. L. Part, C. Kanan et S. Wermter, « Continual Lifelong Learning with Neural Networks: A Review » arXiv:1802.07569 [cs, q-bio, stat], Feb 2018.
3. D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap et G. Wayne, « Experience Replay for Continual Learning » arXiv:1811.11682 [cs, stat], Nov 2018.

4. D. G. Nagy et G. Orbán, « Episodic memory for continual model learning » arXiv:1712.01169 [cs, stat], Dec 2017.
5. Corona, G. Giacinto et F. Roli, « Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues » Information Sciences, vol. 239, pp. 201-225, Aug 2013.
6. M. Kloft et P. Laskov, « Online Anomaly Detection under Adversarial Impact » p. 8.
7. E. Chan-Tin, D. Feldman, N. Hopper et Y. Kim, «The Frog-Boiling Attack: Limitations of Anomaly Detection for Secure Network Coordinate Systems,» Security & Privacy in Communication Networks, vol. 19, Y. Chen, T. D. Dimitriou et J. Zhou, Éds., Springer Berlin Heidelberg, 2009, pp. 448-458.
8. F. Valeur, « Real-Time Intrusion Detection Alert Correlation » 2006.
9. R. Sadoddin et A. A. Ghorbani, « An Incremental Frequent Structure Mining Framework for Real-time Alert Correlation » Computers \& Security, vol. 28, pp. 153-173, 2009.
10. K. Julisch, « Clustering Intrusion Detection Alarms to Support Root Cause Analysis » ACM Transactions on Information and System Security, vol. 6, 2003.
11. Valdes et K. Skinner, « Probabilistic Alert Correlation » chez Proceedings of the Fourth International Symposium on the Recent Advances in Intrusion Detection (RAID2001), 2001.
12. J. Navarro, A. Deruyver et P. Parrend, « A Systematic Survey on Multi-step Attack Detection » Computers & Security, vol. 76, 2018.
13. G. Brogi et V. Viet Triem Tong, « TerminAPTor: Highlighting Advanced Persistent Threats through Information Flow Tracking » 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2016.
14. N. Hossain, S. M. Milajerdi, J. Wang, B. Eshete, R. Gjomemo, R. Sekar, S. D. Stoller et V. N. Venkatakrishnan, « SLEUTH: Real-time Attack Scenario Reconstruction from COTS Audit Data » p. 19.
15. S. M. Milajerdi, R. Gjomemo, B. Eshete, R. Sekar et V. N. Venkatakrishnan, « HOLMES: Real-time APT Detection through Correlation of Suspicious Information Flows » arXiv:1810.01594 [cs], 2018.
16. K. Pei, Z. Gu, B. Saltaformaggio, S. Ma, F. Wang, Z. Zhang, L. Si, X. Zhang et D. Xu, « HERCULE: attack story reconstruction via community discovery on correlated log graph » chez Proceedings of the 32nd Annual Conference on Computer Security Applications - ACSAC 16, 2016.
17. J. Navarro, V. Legrand, A. Deruyver et P. Parrend, « OMMA: open architecture for Operator-guided Monitoring of Multi-step Attacks » EURASIP Journal on Information Security, vol. 2018, Dec 2018.
18. C. Zhou et R. C. Paffenroth, « Anomaly Detection with Robust Deep Autoencoders » Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 665-674, 2017.
19. B. I. P. Rubinstein, B. Nelson, L. Huang, A. D. Joseph, S.-h. Lau, S. Rao, N. Taft et J. D. Tygar, « ANTIDOTE: understanding and defending against poisoning of anomaly detectors » chez Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference - IMC 09, 2009.
20. K. M. C. Tan, K. S. Killourhy et R. A. Maxion, « Undermining an Anomaly-Based Intrusion Detection System Using Common Exploits » chez Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID2002), 2002.
21. J.-P. Fauvelle, A. Dey, S. Navers, « Protection d'un système d'information par une intelligence artificielle : une approche en 3 phases basée sur l'analyse des comportements pour détecter un scénario hostile », European Cyber Week - CESAR Conference - Artificial Intelligence & Cybersecurity, 2018.
22. Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, « Explaining and Harnessing Adversarial Examples » ICLR, 2015.
23. Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, Ananthram Swami, « Practical Black-Box Attacks against Machine Learning », Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, pages 506-519.
24. A. Kurakin, I. Goodfellow, S. Bengio, « Adversarial Machine Learning at Scale », ICLR 2017.
25. Fabian Yamaguchi, Felix 'FX' Lindner, Konrad Rieck, « Vulnerability Extrapolation: Assisted Discovery of Vulnerabilities using Machine Learning », WOOT 2011.

26. Edward Raff, Jon Barker, Jared Sylvester, Robert Brandon, Bryan Catanzaro, Charles Nicholas, « Malware Detection by Eating a Whole EXE », Published in 2018 AAAI Workshop on AI for Cyber Security, 2018.
27. Bojan Kolosnjaji , Ambra Demontis , Battista Biggio , Davide Maiorca , Giorgio Giacinto , Claudia Eckert, Fabio Roli, « Adversarial Malware Binaries: Evading Deep Learning for Malware Detection in Executables », 26th European Signal Processing Conference (EUSIPCO), 2018.
28. Hyrum S. Anderson, Anant Kharkar, Bobby Filar, David Evans, Phil Roth, « Learning to Evade Static PE Machine Learning Malware Models via Reinforcement Learning », arXiv preprint arXiv:1801.08917, 2018.
29. Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, Dawn Song, « Robust Physical-World Attacks on Deep Learning Models », arXiv preprint arXiv:1707.0894, 2017.
30. Yujie Ji, Xinyang Zhang, Shouling Ji, Xiapu Luo, Ting Wang, « Model-Reuse Attacks on Deep Learning Systems », ACM SIGSAC Conference, 2018.
31. Madhu Shashanka ; Min-Yi Shen ; Jisheng Wang, « User and entity behavior analytics for enterprise security », IEEE International Conference on Big Data, 2016.
32. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, « Generative Adversarial Networks », Advances in Neural Information Processing Systems 27, 2014.
33. G. Bateson, A. Montuori « Mind and Nature : A Necessary Unity (Advances in Systems Theory, Complexity, and the Human Sciences) », Hampton Press, 2002.
34. Léonard Todjihounde, « Calcul différentiel », Éditions Cépaduès, 2009.
35. CVE, https://www.cvedetails.com/, on 27/06/2019.
36. Eric M. Hutchins, Michael J. Cloppert, Rohan M.Amin, « Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains », Lockheed Martin Corporation, 2011.