IA évolutionniste pour la détection de comportements déviants dans les SI

Olivier Gesny, Adrien Quemat, Tudy Gourmelen, Pierre-Marie Satre, Bertrand Virfollet, Julien Roussel, Robin de Saint, Nicolas Sourbier, Maxime Durand, Maxime Pelcat, et al.

▶ **To cite this version:**

Olivier Gesny, Adrien Quemat, Tudy Gourmelen, Pierre-Marie Satre, Bertrand Virfollet, et al.. IA évolutionniste pour la détection de comportements déviants dans les SI. [Rapport de recherche] SILICOM; INSA RENNES; IETR/INSA Rennes. 2021. hal-03374007

HAL Id: hal-03374007
https://hal.science/hal-03374007

Submitted on 11 Oct 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

IA évolutionniste pour la détection de comportements déviants dans les SI*

Olivier Gesny[1], Adrien Quemat[1], Tudy Gourmelen[1], Pierre-Marie Satre[1], Bertrand Virfollet[1], Julien Roussel[1], Robin de Saint Jores[1], Nicolas Sourbier[2], Maxime Durand[2], Maxime Pelcat[2], and Pierre Delesques[1]

[1] Silicom, France
{ogesny,aquemat,tgourmelen,pmsatre,bvirfollet,jroussel,rdesaintjores,pdelesques}@silicom.fr
https://www.silicom.fr
[2] INSA Rennes, France {nicolas.sourbier,maxime.durand,maxime.pelcat}@insa.fr
https://www.insa-rennes.fr

Abstract. De nombreuses méthodes IA supervisées ou non supervisées sont utilisées pour la détection d'anomalies et d'attaques. Pour apporter davantage de contexte et d'interprétabilité à l'analyste SOC, cet article introduit une nouvelle approche d'apprentissage continu de la normalité et de détection de comportement anormal (nouveau ou déviant) avec la production d'une synthèse facilement interprétable par un analyste SOC. Cette approche s'appuie sur des stratégies IA évolutionnistes, des fonctions d'attentions auto paramétrables et des motivations intrinsèques de développement de connaissances comportementales. Dans un esprit d'accroissement de la confiance des analystes envers les algorithmes IA, ce modèle démontre de bonnes capacités de découverte de règles comportementales interprétables et de détection de comportements anormaux dans des logs applicatifs générés par un nouveau simulateur de Système d'Information (SI).

Keywords: Investigation numérique SI· Détection d'anomalies comportementales · Simulateur de SI · Apprentissage continu · Explicabilité · Interprétabilité · Motivation intrinsèque d'accroissement de connaissances causales · Logs applicatifs

1 Introduction

L'IA s'est dotée depuis quelques années d'une variété grandissante d'algorithmes évolutionnistes tirant parti de techniques Reinforcement Learning (RL) [21], Deep Learning (DL), Intrinsic Motivation, Attention functions (AF), Genetic Algorithm (GA) ou Genetic Programming (GP) exploitées avec succès dans de nombreux cas d'usage (recherche de meilleures stratégies d'actions, classification, optimisation). Ces algorithmes, qui bâtissent un processus d'émergence de règles de décisions pour une tâche donnée, constituent la base de notre travail. Nous sommes partis d'un modèle IA RL [18] lui-même dérivé du Tangled Program Graphs (TPG) [15] et l'avons modifié afin de lui conférer des capacités d'apprentissage de règles interprétables par l'humain et de décisions explicables. En premier lieu, nous rappellerons dans cet article les algorithmes qui ont inspiré notre travail. Puis, nous évoquerons le modèle IA que nous avons développé. Enfin, nous nous intéresserons à son application pour détecter au plus tôt des comportements déviants (potentiellement malveillants) pour un serveur Web grâce aux logs applicatifs récoltés dans un Système d'Information (SI).

2 Motivations et travaux relatifs

De nombreux outils utilisent des méthodes par signatures [5] mais les règles de détection d'anormalité sont, soit trop génériques (engendrant trop de faux positifs ou de faux négatifs), soit trop restrictives et ne permettent pas la détection de nouveautés. D'autres outils proposent des algorithmes ayant appris

* Supported by organizations Silicom and INSA Rennes.

des heuristiques de détection grâce à des méthodes Deep Learning (DL) [2][3][13][16] ou du Reinforcement Learning (RL) [19][20]. D'autres enfin [7][8][11][1][22][24] sont basés sur de l'auto apprentissage continu de la normalité et la détection de tout écart à cette normalité. Cependant, les anomalies présentées aux analystes des Security Operation Center (SOC), par les IA de ces outils, ne sont pas ou peu interprétables. Nous estimons qu'un modèle s'appuyant sur des techniques IA, évolutionnistes [12] (exploitant des motivations intrinsèques [9][10][15]), ainsi que des fonctions d'attention dont le rôle est de donner naissance à des agents qui vont chacun surveiller un comportement plus ou moins complexe, est à même d'acquérir les capacités suivantes dans un environnement SI constitué de logs applicatifs : émergence et mise à jour continue de comportements normaux ; détection et caractérisation de déviances comportementales.

3 Description du modèle

3.1 Introduction

Pour concevoir le nouveau modèle IA (appelé SIVA dans la suite de l'article), nous nous sommes appuyés sur un modèle existant [18] qui construit, par sélection et combinaison d'équipes coopératives les plus intéressantes, un graphe de décisions formalisant la meilleure stratégie d'actions au regard d'un objectif donné. Ce modèle, en dépit des bons résultats pour un objectif de classification de malwares, souffre de plusieurs défauts pour une détection d'anomalies dans des logs SI :

- Défaut de contexte/interprétabilité : à l'instar des systèmes exploitant le Deep Learning (DL) ou Deep Reinforcement Learning, émergent des décisions difficilement interprétables compte-tenu de la quantité d'informations dans chaque chemin de décision du réseau profond appris.
- Défaut d'explicabilité quant à la prise de décision : pas d'indicateur chiffré permettant de transmettre un indice de confiance quant à la prise de décision
- Faiblesse des propriétés à observer : les règles (ou attentions) sont extrêmement simplistes et pour certaines trop "vagues/larges/lâches" pour éviter des faux positifs.
- Trop peu d'évolutivité après chaque décision : en effet, le modèle créait des agents après chaque action mais les mutations effectuées sur le clone étaient trop aléatoires.
- Nécessité de logs labellisés.

Pour palier ces défauts, nous avons imaginé une approche visant à exploiter les techniques IA évolutionnistes non supervisées et à leur adjoindre des fonctions d'attention modélisant des comportements. Le but étant d'apprendre des connaissances comportementales normales sur un SI donné avant de lancer le modèle appris pour repérer les comportements anormaux (et potentiellement malveillants) dans ce même SI.

Principe général d'apprentissage de SIVA

La figure suivante illustre la boucle d'apprentissage de SIVA (Fig. 1).

Modifications apportées au modèle initial

SIVA dérive de l'algorithme CBWAR [18] auquel les capacités suivantes ont été ajoutées :

- Fonctions d'attention permettant la mémorisation de statistiques pour des relations complexes (Ce fonctionnement est inspiré du fameux neurone "Jenifer Aniston" [6]) : chaque fonction d'attention va surveiller une IP ou des valeurs de clefs particulières comme les ports, la taille, les adresses MAC... Elles permettent de caractériser en mémoire après apprentissage ce qu'est un comportement normal, notamment dans des logs applicatifs, et de signaler les anomalies ou les nouveautés.
- Prise de décision rapide et instinctive ou plus lente et réfléchie via la notion de Système I/II [14][17].
- Mémorisation de causalités et exploitation de ces causalités en priorité.
- Apprentissage continu avec ajout de nouveaux agents via clonage des agents les plus efficaces en direct après chaque action.
- Récompenses tenant compte d'éléments non supervisés : les motivations intrinsèques favorisent les agents découvrant de nouvelles observations pertinentes.

Grâce à cet algorithme, émergent un graphe d'agents enchevêtrés et une base d'agents mémorisant des connaissances comportementales corrélées à une décision.

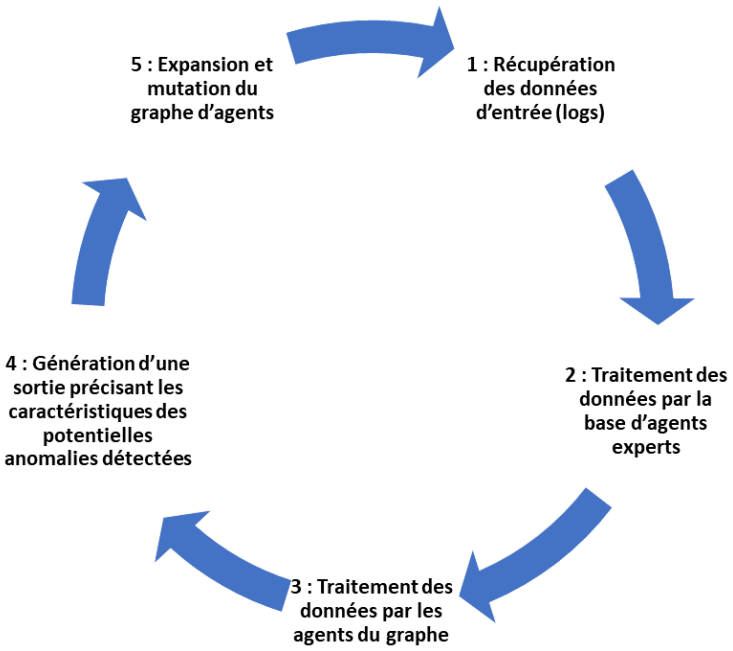

Fig. 1. Principe général d'apprentissage de SIVA.

3.2 Description générale du module métier

Le module métier se charge simplement de retourner au module IA l'environnement observable à un instant donné. Celui-ci est appelé stimulus.

4 Expérimentation

4.1 Simulateur de Système d'Information

Motivations

Lors de nos premières expérimentations utilisant l'IA au service de la cybersécurité, nous avons été confrontés à :

- L'obsolescence des datasets de cybersécurité mis à disposition par la communauté, la non maîtrise de leur topologie et des services déployés.
- La lenteur de déploiement des simulations lorsque les cyberrange [4] exploitent la technologie VM.
- La difficulté à constituer des SI multi réseaux lorsque les cyberrange exploitent la technologie docker.

Afin de pallier ces défauts, nous avons donc développé Sisimulator : une plateforme simulant un environnement numérique grâce à la technologie "container" lxd. L'objectif est d'imiter dans un espace clos un environnement numérique complexe ainsi que les interactions avec les différents services proposés.

Description du déploiement

Le déploiement de l'environnement se déroule en 3 étapes représentées par 3 fichiers de configurations json successifs. Tout d'abord le premier fichier indique à la plateforme les machines et réseaux à créer. Il est possible de spécifier les plages d'IP des réseaux ainsi que les OS des machines. Par la suite la simulation

```
{
  "@timestamp" : "<timestamp>",
  "host" : "<IP de la machine dont sont issus les logs>",
  "message" : "<Message applicatif>",
  "@version" : "1"
}

{
    "host":"10.122.1.15",
    "@version":"1",
    "@timestamp":"2021-07-23T18:50:45.364Z",
    "message":"<142>Jul 23 18:50:43 dvwa iptables Jul 23 18:50:42 dvwa [iptables] IN=eth1
    ↪  OUT=  MAC=00:16:3e:ae:88:e5:00:16:3e:0d:50:68:08:00 SRC=10.122.1.40
    ↪  DST=10.122.1.15 LEN=60 TOS=00 PREC=0x00 TTL=64 ID=12696 DF PROTO=TCP SPT=54266
    ↪  DPT=80 SEQ=4116312003 ACK=0 WINDOW=64240 SYN URGP=0 MARK=0 "
}
```

Listing 1: Format du fichier de logs (environnement observable) suivi d'un exemple de stimulus observable par l'IA

configure l'ensemble des machines grâce à un second fichier qui lui spécifie les modifications à apporter. Enfin, le dernier fichier spécifie les utilisateurs qui exécuteront des scripts de vie.

Pour le moment la simulation bénéfice de 4 services configurables : web (apache2), mail (cyrus), ldap (openldap), dns (bind9). Il est également possible de configurer des machines qui exécuteront des scripts de vie afin de simuler des utilisateurs qui interagiront avec les différents serveurs à disposition.

Simulation de vie

Afin de générer des logs, nous avions besoin de simuler du trafic de vie correspondant à différents types d'actions utilisateur. La simulation débute par la connexion d'un utilisateur. Cela permet de communiquer avec le service LDAP si celui-ci est présent dans sa base de données. La simulation propose l'installation d'un serveur web non sécurisé, de ce fait les scripts de simulation disposent du framework Selenium pour interagir avec les services webs disponibles. Il est possible de les contacter via leur nom de domaine, ce qui entraine une communication avec le serveur DNS.

Centralisation et format des logs

Dans l'expérimentation, les logs applicatifs (ldap, service web) et machines sont centralisés par le simulateur via logstash et dupliqués dans un fichier comprenant les logs au format json.

4.2 Méthode d'apprentissage initial et continu

La mise en oeuvre du modèle se déroule en 2 phases :

- Apprentissage de la normalité comportementale : pour un challenge consitué de N logs, l'algorithme autorise une génération d'agents à s'exécuter. La meilleure équipe d'agents (celle qui a développé la meilleure stratégie de mutation pour engranger de la connaissance) est conservée et testée sur le challenge suivant. Si cette équipe est meilleure que la meilleure équipe précédente alors, celle-ci fait l'objet d'un clone et d'une mutation alors que les moins bonnes équipes d'agents sont supprimées. Cette phase de mutation se termine lorsque plus aucune connaissance n'apparaît dans la base de connaissances sur plusieurs générations.
- Poursuite de l'apprentissage continu et détection de nouveauté ou de déviance comportementale : cette phase est déclenchée après l'apprentissage.

4.3 Expérimentation et résultats

Afin que l'IA puisse disposer de logs aplicatifs, nous avons configuré un système d'information composé de 2 sous-réseaux liés par un routeur.

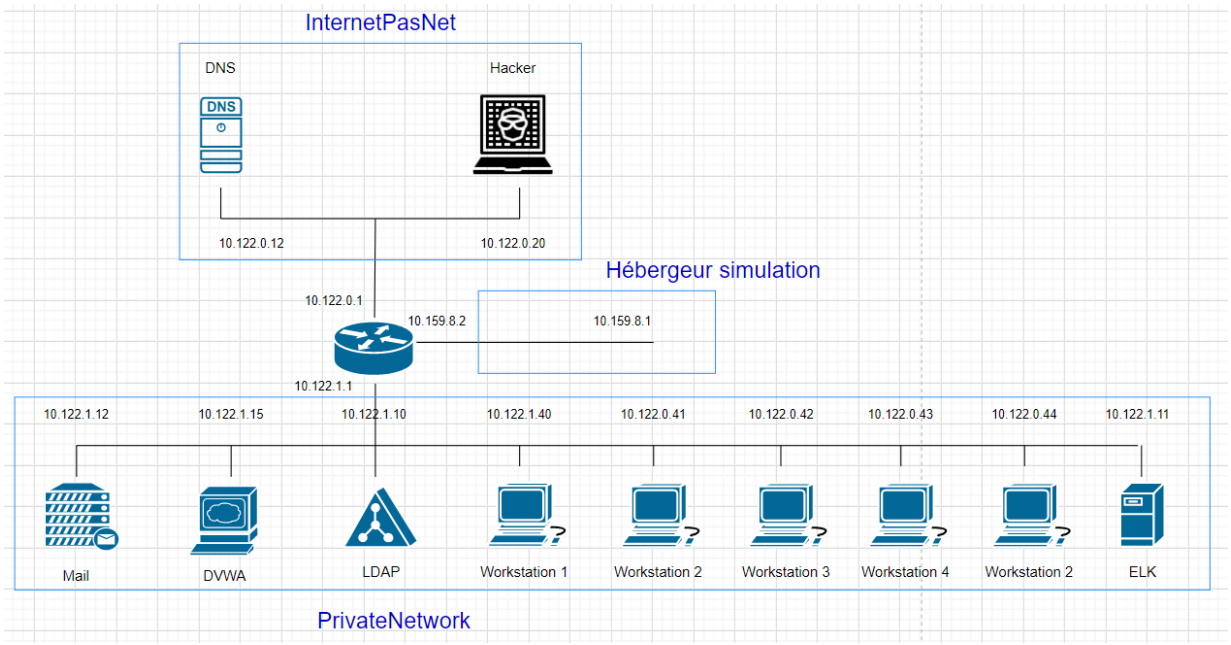

Fig. 2. Topologie du réseau simulé.

Le sous-réseau InternetPasNet simule très sommairement le comportement d'internet avec un serveur DNS et un utilisateur hacker dont le rôle est de pénétrer le sous-réseau privateNetwork. PrivateNetwork simule un système d'information d'une petite entreprise. Il est composé d'un serveur LDAP dans lequel est enregistré l'ensemble des comptes utilisateurs de l'entreprise. Un serveur de mail est lié au serveur LDAP. Un serveur ELK centralisera les logs des différentes machines du système d'information. Pour finir un serveur web faillible permet aux différents utilisateurs d'interagir avec et à l'attaquant de bénéficier d'un point d'entrée. Des scripts sont configurés sur toutes les machines pour simuler un trafic de vie réaliste.

Nous avons implémenté une attaque Web (par injection de la commande '8.8.8.8; "ncat" +my_ip+ my_port + "-e /bin/bash"') qui est noyée au milieu de trafic sain précédemment explicité. Celle-ci utilise le serveur web DVWA [23] comme point d'entrée dans le SI grâce à la mise en place d'un reverse shell permettant une exfiltration de données vers une "infrastructure de Commande et Contrôle (C&C)".

L'expérimentation se déroule en 2 phases :

- Apprentissage de la normalité comportementale sur SI non attaqué : l'apprentissage se déroule sur plusieurs heures et est fonction de la quantité de logs applicatifs.
- Poursuite de l'apprentissage de la normalité comportementale et détection de comportement normal : l'algorithme est configuré en mode détection et, lorsqu'il détecte un nouveau comportement, l'indique par le biais d'un fichier comportant les caractéristiques déviantes du comportement.

Sur une attaque par injection de commandes, l'algorithme parvient à détecter un ensemble d'anomalies visibles dans la courbe suivante qui montre l'arrivée de 6 comportements nouveaux dans la base d'agents experts.

Par ailleurs, l'explication de la différence par rapport au comportement normal habituellement constaté dans les logs applicatifs d'un serveur Web est traduite dans un fichier json. Celui-ci montre une nouvelle

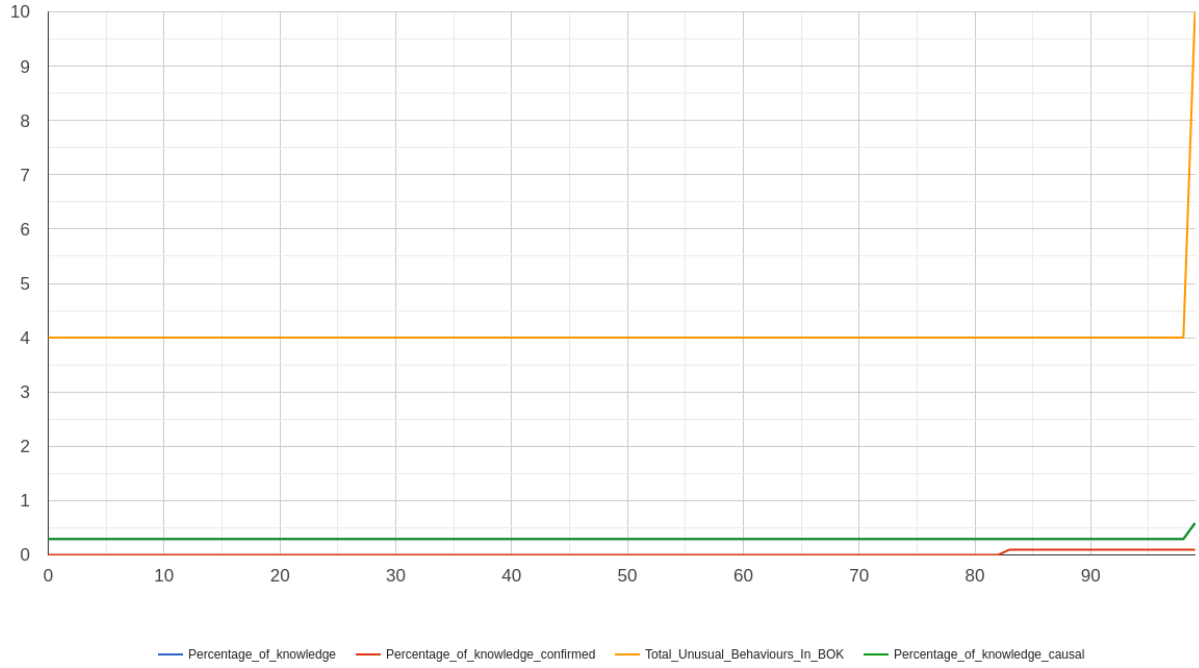

Fig. 3. Détection d'un comportement anormal.

connexion du serveur victime (OUT=eth1 dans le fichier) vers un serveur distant sur le port 80 (DPT=80 dans le fichier). Ce comportement anormal émergeant est symptomatique d'une exfiltration de données.

4.4 Discussion et interprétation

La combinaison de plusieurs algorithmes permet de faire émerger des agents IA évolutionnistes surveillant des comportements interprétables et certains à 100% d'une nouveauté.

Toute anormalité comportementale est transmise à l'humain via un rapport interprétable. Cette capacité a été constatée sur des logs applicatifs rsyslog.

5 Conclusion et perspectives

Grâce à la combinaison de plusieurs algorithmes, le nouveau modèle SIVA est capable d'apprendre des comportements usuels dans des logs ; une fois l'apprentissage réalisé, il détecte avec certitude tout comportement déviant et en rend compte sous une forme interprétable par un analyste.

Il y a une réelle nécessité d'autoriser un retour utilisateur à un moment donné pour qualifier la nature malveillante d'une anormalité. Cet inconvénient est en cours de résolution par l'emploi de la technique Active Learning qui pemettra de rendre un verdict d'analyste sur tout comportement considéré comme déviant ou étranger.

Les perspectives d'intégration de SIVA dans des SOC pour de l'analyse temps réel avec remédiation automatique ou post-mortem constituent notre ambition à court terme. La prochaine étape est de doter SIVA de la capacité de prendre en compte un verdict de l'analyste pour toute anomalie, afin de renforcer les connaissances des agents à postériori en tirant parti de l'expertise de l'analyste.

A court terme l'objectif est également de rendre le simulateur de SI open source afin d'en faire profiter l'ensemble de la communauté.

```
"@timestamp": "2021-07-30T08:21:14.072Z",
"New element": {
    "vector<string>": [
        "OUT",
        "",
        "eth1"
    ]
},
"OpCode": "keyValLog",
"alertType": "new value",
"id": 253814,
"log": "{\"@version\":\"1\",\"@timestamp\":\"2021-07-28T15:34:01.965Z\",
\"host\":\"10.122.1.15\",\"message\":\"<142>Jul 28 15:34:01 dvwa iptables Jul 28 15:34:00
↪ dvwa [iptables] IN= OUT=eth1 MAC=00:16:3e:b5:40:21:00:16:3e:66:f2:d9:08:00
↪ SRC=10.122.1.41 DST=10.122.1.15 LEN=60 TOS=00 PREC=0x00 TTL=64 ID=32870 DF PROTO=TCP
↪ SPT=54132 DPT=80 SEQ=1187789549 ACK=0 WINDOW=64240 SYN URGP=0 MARK=0 \"}",
"parameters": {
    "map<string, vector<string>>": {
        "OUT": [
            "",
            "eth1"
        ]
    }
}
```


Listing 2: Extrait du fichier interprétable montrant un comportement anormal

References

1. Tarem Ahmed, Boris Oreshkin, and Mark Coates. Machine learning approaches to network anomaly detection. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.238&rep=rep1&type=pdf, 04 2007.
2. Vectra AI. Choosing an optimal algorithm for AI in cybersecurity. https://www.vectra.ai/blogpost/choosing-an-optimal-algorithm-for-ai-in-cybersecurity, 2021-06-7.
3. Vectra AI. Vectra AI cognito. https://www.vectra.ai/products/cognito-platform, 2021-06-7.
4. AIRBUS. Cyberrange airbus. https://airbus-cyber-security.com/fr/produits-services/prevenir/cyberrange/, 2021-06-7.
5. Mouhammd Al-Kasassbeh. Network intrusion detection with wiener filter-based agent. *World Appl. Sci. J*, 13(11):2372–2384, 2011.
6. Vadim Axelrod, Camille Rozier, Tal Seidel Malkinson, Katia Lehongre, Claude Adam, Virginie Lambrecq, Vincent Navarro, and Lionel Naccache. Face-selective neurons in the vicinity of the human fusiform face area. https://n.neurology.org/content/92/4/197, 2019.
7. Mark Crosbie, Gene Spafford, et al. Applying genetic programming to intrusion detection. In *Working Notes for the AAAI Symposium on Genetic Programming*, pages 1–8. Cambridge, MA: MIT Press, 1995.
8. Darktrace. Darktrace Cyber AI Analyst. https://www.darktrace.com/fr/plateforme-de-cyber-ia/, 2021-06-7.
9. Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. Go-Explore: a New Approach for Hard-Exploration Problems. arXiv : 1901.10995, 2021.
10. Maxime Gasse, Damien Grasset, Guillaume Gaudron, and Pierre-Yves Oudeyer. Causal Reinforcement Learning using Observational and Interventional Data. arXiv : 2106.14421, 2021.
11. Richard Heady, George Luger, Arthur Maccabe, and Mark Servilla. The architecture of a network level intrusion detection system. Technical report, Los Alamos National Lab., NM (United States); New Mexico Univ., Albuquerque
12. Mohammad Sazzadul Hoque, Md Mukit, Md Bikas, Abu Naser, et al. An implementation of intrusion detection system using genetic algorithm. *arXiv preprint arXiv:1204.1336*, 2012.

13. Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. *Eai Endorsed Transactions on Security and Safety*, 3(9):e2, 2016.
14. Daniel Kahneman. *Thinking, fast and slow.* Farrar, Straus and Giroux, New York, 2011.
15. Heywood M. Kelly S. Emergent Tangled Graph Representations for Atari Game Playing Agents. *cDermott J., Castelli M., Sekanina L., Haasdijk E., García-Sánchez P. (eds) Genetic Programming. EuroGP 2017. Lecture Notes in Computer Science*, 10196, 2017.
16. Manuel Lopez-Martin, Belen Carro, and Antonio Sanchez-Esguevillas. Application of deep reinforcement learning to intrusion detection for supervised problems. *Expert Systems with Applications*, 141:112963, 2020.
17. Houdé Olivier. Le raisonnement, que sais-je ? https://www.cairn.info/le-raisonnement–9782130595250.htm, 2014.
18. Julien Roussel Olivier Gesny, Pierre-Marie Satre. CBWAR: Classification de Binaires Windows via Apprentissage par Renforcement. https://www.cesar-conference.org/wp-content/uploads/2018/11/articles/C&ESAR_2018_J2-16_O-GENTRY_CBWAR.pdf, 2018.
19. Safa Otoum, Burak Kantarci, and Hussein Mouftah. Empowering reinforcement learning on big sensed data for intrusion detection. In *Icc 2019-2019 IEEE international conference on communications (ICC)*, pages 1–7. IEEE, 2019.
20. Arturo Servin and Daniel Kudenko. Multi-agent reinforcement learning for intrusion detection. In *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, pages 211–223. Springer, 2005.
21. R.S. Sutton and A.G. Barto. Reinforcement Learning, second edition: An Introduction. https://books.google.fr/books?id=uWV0DwAAQBAJ, 2018.
22. Jouni Viinikka, Herve Debar, Ludovic Me, Anssi Lehikoinen, and Mika Tarvainen. Processing intrusion detection alert aggregates with time series modeling. *Information Fusion*, 10(4):312–324, 2009.
23. Robin Wood. Damn vulnerable web application. https://dvwa.co.uk/, 2020-09.
24. Stefano Zanero and Sergio M Savaresi. Unsupervised learning techniques for an intrusion detection system. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 412–419, 2004.