Malicious human behaviour in information system security: Contribution to a Threat Model for Event Detection Algorithms

Olivier de Casanove, Florence Sèdes

Malicious human behaviour in information system security: Contribution to a Threat Model for Event Detection Algorithms

Olivier de Casanove[1] and Florence Sèdes[1]

Université Toulouse III - Paul Sabatier
118 route de Narbonnes 31062 Toulouse CEDEX 9
Institut de Recherche en Informatique de Toulouse (IRIT)
{olivier.decasanove, florence.sedes}@irit.fr

Abstract. Among the issues our community has to fix, information system security is a global concern both for the security of data and algorithms. The security of algorithms is dependent of the reliability of the input data. This reliability is questioned, especially when the data is generated by humans (or bot operated by humans), like in online social networks. Event detection algorithms are an example of technology using this type of data, but the question of the security is not systematically considered in this literature. We propose in this paper a first contribution to a threat model to overcome this problem. This threat model is composed of a description of the subject we are modelling, assumptions made, potential threats and defense strategies.

Keywords: Threat model · Adversarial Learning · Online Social Network · Event Detection

1 Introduction

Machine learning algorithms map an input data to an output data. The reliability of the output is determined, among other factors, by the reliability of the input. A small perturbation in the input can result in a misleading output [22]. This perturbation may be due to either data gathering or malicious behaviour. When the input data are generated by human, perturbations due to malicious behaviour cannot be disregarded. Data from online social networks is an example of data generated by human or prone to malicious perturbation.

Online Social Networks (OSN) allow users to exchange short messages and media. From these data published in real time, information can be extracted on events. Atefeh and Khreich [4], in their review on event detection on OSN, stated that "in general, events can be defined as real-world occurrences that unfold over space and time". To detect these events, event detection algorithms can be used. They take as an input the Twitter stream and give as an output clusters of messages, where each cluster defines an event. There are many applications, ranging from earthquake detection to musical event detection [4]. The literature

on the subject focuses on how to improve the performances of the detection and the question of the detection's security is neglected. Yet, when the input data is made up of messages crafted by unknown users, this subject becomes a concern. If the implicit hypothesis: "Input data are cleaned from malicious messages crafted in order to disrupt the event detection" may hold in specific contexts, it does not in others. For example, when detecting events related to cybersecurity, the adversaries want their attacks to stay undetected. In this field, we easily find papers which do not take into account a potential threat to their detection [13] [18]. When this hypothesis is false, we find ourselves in an adversarial learning context and event detection is under many threats.

This paper is a first contribution to a threat model for event detection algorithms on Twitter, but the same threat model could be used for other OSN. According to OWASP (Open Web Application Security Project) [16], a threat model is "a structured representation of all the information that affects the security of an application. In essence, it is a view of the application and its environment through the lens of security. [...] A threat model typically includes description of the subject to be modelled, assumptions that can be checked or challenged in the future as the threat landscape changes, potential threats to the system, actions that can be taken to mitigate each threat, a way of validating the model and threat, and verification of success of actions taken."

In the next section, we briefly discuss the related work. We will use this definition of a threat model to structure the rest of our paper. In section 3, we describe the subject we model. In section 4 we provide the assumptions on which our model is based. In section 5 we describe the threats. In section 6 we discuss about the future work and we conclude. We conclude in section 7.

2 Related work

Adversarial learning is a recent field, yet there is already a good literature on it [22]. Current threat models for machine learning focus on three aspects: attack direction (do the attack happen during the learning phase or the classification phase ?), security violation (which kind of security concept the attack violate, traditionally confidentiality, integrity, availability) and attack specification (is it a targeted attack or not ?) We will see in subsection 5.2 why this threat model is not useful for us, as it is. Adversarial learning specifically applied to OSN have been studied in different ways. The first one focuses on text processing applications [2], which is what event detection algorithms are. The second one is more specific, it is about evading spam detection [9]. The third one consists in listing the adversaries and threats can be faced in OSN. For example, Sabottke et al. [19] proposed an event detection algorithm with a list of actors willing to disrupt their algorithm. We used this list as a base to construct the list of profiles in subsection 5.1. Finally, the subject of fake news is out of scope because they impact the users. They are not meant to disrupt the operation of a machine learning algorithm.

3 Modelling Event Detection Algorithms

Atefeh and Khreich [4] as well as Hasan et. al. [10] reviewed the literature to list the techniques used to detect events on social media. Regardless of the technique used, we can formalise an event detection algorithm by a function. We define the function F, the function which associates to a set of messages 0 if the messages are not related and a unique value if the messages are related, where $\mathbb{T}$ is the set of all the tweets possible.

Definition 1 (Event Detection function).

$$\begin{aligned} F\colon \mathbb{T}^n &\to E \\ (t_1,...,t_n) &\mapsto \begin{cases} F(t_1,...,t_n), & \textit{if } t_1,...,t_n \textit{ is an event} \\ 0, & \textit{else} \end{cases} \end{aligned}$$

An event, for the rest of this paper, will be defined as a set of messages related one to another and which are in the same spatial or time window.

Definition 2 (Event).

$$\begin{aligned} &\forall (t_1,...,t_n) \in \mathbb{T}^n, \quad \textit{and} \quad \forall t_{k\geq 1} \in (t_1,...,t_n), \\ &t_k \textit{ is spatially or timely close to } t_{k-1} \quad | \quad F(t_1,...,t_n) \neq 0 \end{aligned}$$

An attacker can create fake messages thanks to techniques such as Markov Chain or Neural Network. When executed, these algorithms will produce a new fake message contained in a set of messages the algorithm is able to generate. Therefore we can represent the fake message by a random variable.

Definition 3 (Fake message random variable).

$$\begin{aligned} X\colon \Omega &\to \mathbb{T} \\ \omega &\mapsto X(\omega) \end{aligned}$$

We define a false positive event (FP event), in the adversarial context, as a set of messages mainly composed of messages created by an attacker and recognized as an event.

Definition 4 (False Positive Event).

$$\forall (X_n)_{n\geq 1} \sim X, \quad (X_1,...,X_n,t_1,...,t_m) \in \mathbb{T}^{n+m}, \quad F(X_1,...,X_n,t_1,...,t_m) \neq 0, \\ F(t_1,...,t_m) = 0$$

In opposition, we define a true positive event (TP event), in the adversarial context, as a set of messages mainly composed of legitimate messages and recognized as en event.

Definition 5 (True Positive Event).

$$\forall (X_n)_{n\geq 1} \sim X, \quad (X_0,...,X_n,t_0,...,t_m) \in \mathbb{T}^{n+m}, \quad F(X_0,...,X_n,t_0,...,t_m) \neq 0, \\ \exists t_{i,j,...} \in (t_0,...,t_m), \quad F(t_{i,j,...}) \neq 0$$

4 Assumptions

As previously said in section 1, a threat model needs assumptions. We identify two assumptions for this threat model to make sense.

Assumption 1 *Input data from Twitter, and more generally social networks, contain messages written by malicious users with the objective to deceive algorithms taking this data as input.*

We know that extracted data from Twitter contain spams and other malicious messages like phishing, for example. Those messages have an influence on the quality of the detection of our algorithms. Working in an adversarial context means taking the idea one step further and supposing that malicious users craft messages just to disrupt event detection algorithms.

Assumption 2 *Attackers have access to the ground truth used to develop event detection algorithms.*

The datasets used to compare event detection algorithms are public, papers describing how event detection works are also easily accessible; therefore it is safe to assume that the attackers have access to our ground truth. It also means that the system is a "grey box" for the attackers, they have partial knowledge of how it works. Security by obscurity is not an option here.

Assumption 3 *The benefit of disrupting the detection for the attacker is equal to the cost for the defender to see its detection disrupted.*

We make this assumption to model the adversarial context as a zero-sum game. A zero-sum game, in game theory, is a situation where the benefit of a player (i.e. the attacker) is exactly equal to the cost of the other player (i.e. the defender). Interesting properties could be derived from zero-sum game, we will use them in a future work to validate the model. This is a common assumption in adversarial problems [21][25] and in information system security in general [24][23].

5 Threats to the system

In our context, a threat is defined as the combination of a malicious actor (the attacker) and a means to disrupt the event detection (the attack). We will detail both the attacker and the attack in the next two subsections.

5.1 Attacker profiles

In a previous paper, we reviewed other contributions [7] and identified three profiles in the literature: troll, spammers and adversaries. These profiles are generic and we need to adapt their definition to our context. This gives us the following attacker classification:

- Troll: their objective is to create or make disappear subjects and therefore, events. They target both humans and automatic tools to analyse the news. Their actions decrease the proportion of useful information in the Twitter stream.
- Spammer: they make a lot of messages serving their own interests. They can use buzzwords, keywords or tag people to improve the efficiency of the spamming activity. They do not target our algorithm directly but their activity creates a lot of noise in the Twitter stream.
- Adversary: they seek to create FP events and make TP events disappear. Their means are diverse, but we can suppose that they have at least partial knowledge of the technology behind event detection since they are directly targeting it.

The definitions of the attackers are centred around the impact he could have. These profiles could be refined with two additional criteria: 1) is the attacker ignorant or knowledgeable of the system ? And 2) is the attacker constrained or free of any constraints ? Indeed the attacker could have multiple types of constraints, economic or political, for example. Now that we discussed about the profiles of the attackers, let's continue with the type of attacks they can use.

5.2 Attacks

In information system security the CIA model (Confidentiality, Integrity and Availability) is often used [20]. However, this model does not suit our needs well in our adversarial learning context. For example, it does not make sense to defend the confidentiality of the detected events when all of the message composing it are public. We propose instead to use the reliability and validity. These two concepts are often used to describe tests.

Reliability The reliability of a test is its ability to stay consistent. In other words, a same input should always give the same output. In our adversarial context, the reliability becomes 1) the ability of the event detection algorithm to detect a same event, both when the input data are not corrupted and when a malicious actor is tampering with the input messages and 2) the ability to detect an event, with no more and no less messages in it, when a malicious actor is tampering our data. To measure the impact on consistency, we need to first run our algorithm on a dataset without fake messages and label the messages associated to an event. We run again our algorithm, this time with the fake messages in the dataset. To compare the difference between the two executions, we use the *BCubed precision* to observe how many irrelevant messages are clustered together and *BCubed recall* to see how many messages are missing in events. Finally, we can combine these two metrics in the *BCubed F1-Score* [3]. The reliability can be defined with the following formula:

$$\text{reliability} = \frac{1}{n}\sum_{i=0}^{n} BCubed\_F1\_Score(event_i)$$

Validity The validity of a test is its ability to detect what it pretends to detect. In our case, is the event detection algorithm detecting events and not just give a random output ? The objective of the event detection algorithm is therefore to maximise TP events and minimise FP events. The validity can be redefined by the following formula:

$$\text{validity} = \frac{TP}{TP + FP}$$

Attacks classification After studying event detection algorithms, we identify eight attacks. These attacks are described in terms of their impact on the reliability and the validity of the detection. They are classified in three categories: event creation, event dispersion and event modification. These categories gather the attacks which use the same means, but they don't always have the same goal. We summarise the attacks in table 1.

Event creation The attacker triggers an event detection, which increases the FP events and therefore impact the validity. The attacker uses a tool to procedurally generate fake tweets. Those messages are then injected in the Twitter stream. We identify three attacks in this category:

- Craft: fake tweets are created. Those messages are close enough for the event detection algorithm to consider them as related but does not necessarily make sense for a human. Those messages trigger a detection.
- Message expansion: real tweets, not related to any event, in association with malicious tweets trigger an event. This attack also impacts the reliability since a legitimate message, not related to any event, become related to an event.
- Replay: A TP event is replayed, entirely or partially, at a time where the event doesn't make sense.

Event dispersion The objective is to inject enough malicious messages during a small lapse of time so the legitimate tweets appear too far from one another in the Twitter stream. Three attacks exist in this category:

- Fragmentation: an event is split in two or more subgroups of tweets, resulting in detection of multiple events when they are the same. One TP events become many TP events under attack; therefore the reliability is impacted.
- Cancellation: an event doesn't trigger a detection when it should. The tweets are so split by the malicious messages that they aren't recognised as an event anymore. This attack decreases the number of TP events and transforms a TP event in nothing, therefore both the validity and the reliability are impacted.
- Deterioration: the number of tweets in an event decreases when under attack. This is a mix case between fragmentation and cancellation. The first or last messages are too far to be associated with the event, but they are still enough messages to trigger a detection. This is an inconsistency under attack; therefore it impacts reliability.

Name	Description	Impact on Reliability	Impact on validity
	Event Creation		
Craft	A collection of fake tweets triggered an event	NO	YES
Message Expansion	A collection of fake and real tweets, which wouldn't have triggered an event otherwise, triggered an event	YES	YES
Replay	A true event is replayed a second time by the attacker	NO	YES
	Event Dispersion		
Fragmentation	An event triggered multiple detection due to spam activities	YES	NO
Cancellation	An event doesn't triggered detection due to spam activities	YES	YES
Deterioration	The number of tweets related to an event is less than expected due to spam activities	YES	NO
	Event Modification		
Drift	The attacker change the event keywords or event	YES	NO
Merge	Messages from an event start to aggregate to another event	YES	NO

Table 1. Attacks against event detection algorithms

Event modification The attacker generates malicious tweets which seem related to one another for the event detection algorithm. As for event creation, the messages are generated procedurally.

- Drift: the attacker creates malicious tweets which aggregate on a TP event. The objective is to change the event keywords or subject. It creates an inconsistency; therefore the reliability is impacted.
- Merge: the attacker changes the event keywords or subject so another event messages start to aggregate on the first event. For this attack to be successful, the attacker needs to know the subject of two different events. It is safe to assume that if the attacker knows this, then both events already have been detected by our algorithm. Therefore it only creates an inconsistency on the number of messages aggregated to each event, and not in the number of TP events detected. The reliability is impacted.

6 Defense strategies

The defender can protect the detection by adding filters and two different levels. The first one is at tweet level, where we filter the tweets which seem malicious. The second level is at cluster level, where we try to distinguish TP events from

FP events. We define the filter function h, the function which associates, to each set of messages recognised as an event, 0 if the set of messages does not satisfy the constraints or a unique value otherwise.

Definition 6 (Filter Function).

$$h\colon E \setminus \{0\} \to E$$
$$F(t_0, ..., t_n) \mapsto \begin{cases} F(t_0, ..., t_n), & \textit{if } (t_0, ..., t_n) \textit{ satisfies the constraints} \\ 0, & \textit{else} \end{cases}$$

With this new element in mind we redefined TP and FP event as follows:

Definition 7 (True Positive Event).

$$\forall (X_n)_{n\geq 0} \sim X, \forall (X_0, ..., X_n, t_0, ..., t_m) \in \mathbb{T}^{n+m}, \quad \exists t_{i,j,...} \in (t_0, ..., t_m),$$
$$F(t_i, t_j, ...) = h \circ F(X_0, ..., X_n, t_0, ..., t_m) \neq 0$$

Definition 8 (False Positive Event).

$$\forall (X_n)_{n\geq 0} \sim X, \quad m \geq 0, \quad (X_0, ..., X_n, t_0, ..., t_m) \in \mathbb{T}^{n+m},$$
$$h \circ F(X_0, ..., X_n, t_0, ..., t_m) \neq F(t_0, ..., t_m) = 0,$$

We will now discuss what the defense strategies are. Table 2 summarises which defense strategies mitigate which attacks.

6.1 Filtering messages

The objective of a spam filter is to distinguish fake users, spams and spammers from legitimate tweets and users [1]. A spam filter can be made on the content of the tweets, the characteristics of the tweets, the users behind the tweets or the relationships in the OSN of the users behind the tweets [1]. All these solutions are machine learning solutions; therefore we introduce a new level of adversarial learning. However, the problem of adversarial learning for spam detection has already been discussed by [5] [6] [8] [12]. Generating fake messages that can fool the spam filter increase the cost of the attack. Therefore, this strategy is effective against every attack which needs to create fake tweets. It is especially effective against dispersion attacks since those attacks are based on flooding and flooding are easily detected by spam filters. Finally, spam detection based on user features is effective against replay attacks because it means that the accounts replaying the events should avoid spam detection; therefore it increases the cost of the attack.

6.2 Filtering clusters

TP and FP events have different characteristics. We can set threshold for these characteristics to differentiate TP from FP events. These threshold are used as filters to discard FP events. We identified four metrics in the literature on which we can filter events:

- Word entropy: The entropy of a cluster was introduced by [17]. The formula (1) is used where X is a random variable and $P(X_i)$ is the probability to draw a specific word out of all the words of the cluster. A cluster with a very low word entropy is probably composed of very similar crafted messages.
- User diversity: The formula (1) is applied but instead of applying it to words, we apply it to users in the cluster. We have X a random variable and $P(X_i)$ the probability to draw a specific user out of all the users of a cluster. *User diversity* in a cluster was introduced by [14]. This metric is particularly interesting because accounts are the most difficult thing to fake as an attacker. *User diversity* is one of the rare defense measures against event replay. The attacker can replay the exact same tweets but not the exact same author.
- Least Common Subsequence (LCS): Hasan et al. in [11] use a filtering method based on the LCS at word-level. The idea, based on empirical evidence they found, is that cluster of newsworthy event will have a superior LCS than not newsworthy event. In their paper, the authors fixed an LCS threshold under which every event is discarded. It may help to identify drifted and merged events since the first and last messages are likely to be very different.
- Named entity recognition: This technique is introduced by [15] as a way to pre-select tweets with significant improvement in the final result. The argument behind this constraint is that a tweet without a named entity does not provide any information and is therefore useless.
- Event size: Intuitively a cluster of fewer than 3 tweets cannot be considered as an event. However, finding an exact event size threshold separating meaningful events from similar but not related messages are impossible. Event size should be considered as an hyperparameter of our model to help us drop FP events.

Some of the filter proposed are easy to bypass. For example, attackers can automatically add a random named entity in their fake tweets. We should keep in mind that, for the attacker, every attack is a trade-off between the costs and benefits of the attacks. Therefore, every defense strategy increasing the cost of the attack is worthwhile.

$$H(X) = -\sum_{i}^{n} P(X_i) log_b P(X_i) \tag{1}$$

6.3 Other strategy

Defragmentation is a process where events are reviewed to check if two detected events are in fact only one. Some event detection algorithms are prone to fragmentation [4]. Our context adds another interest to defragmentation: the resilience to the event splitting attack. We found one utilisation of defragmentation in [11].

Name	Post-process security operations
Event Fragmentation	Spam filters, Defragmentation
Event Cancellation	Spam filters
Event Deterioration	Spam filters
Tweets Expansion	Spam filters, Cluster filters
Event Crafting	Spam filters, Cluster filters
Event Replay	Spammer filters, User diversity filter
Event Drifting	Spam filters, LCS filter
Event Merging	Spam filters, LCS filter

Table 2. Defense strategies

7 Conclusion and future work

In this paper we proposed a first contribution to a threat model for event detection. We define the situation we are modelling, assumptions that were made, the attackers' profile, possible attacks and defense strategies. We seek to expand this threat model in order to validate it and test the assumptions. This work is dedicated to help future event detection algorithms to be more resilient against adversarial attacks and therefore, develop a technology more suited for real-life applications. This threat model is especially useful when the event detection algorithms detect events related to any subject where an adversary can be found.

References

1. Abkenar, S.B., Kashani, M.H., Akbari, M., Mahdipour, E.: Twitter Spam Detection: A Systematic Review. arXiv:2011.14754 [cs] (Dec 2020), arXiv: 2011.14754 version: 2
2. Alsmadi, I., Ahmad, K., Nazzal, M., Alam, F., Al-Fuqaha, A., Khreishah, A., Algosaibi, A.: Adversarial Attacks and Defenses for Social Network Text Processing Applications: Techniques, Challenges and Future Research Directions. arXiv:2110.13980 [cs] (Oct 2021), http://arxiv.org/abs/2110.13980, arXiv: 2110.13980
3. Amigó, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. Information Retrieval **12**(4), 461–486 (Aug 2009). https://doi.org/10.1007/s10791-008-9066-8
4. Atefeh, F., Khreich, W.: A Survey of Techniques for Event Detection in Twitter. Computational Intelligence **31**(1), 132–164 (2015). https://doi.org/https://doi.org/10.1111/coin.12017
5. Biggio, B., Fumera, G., Roli, F.: Design of robust classifiers for adversarial environments. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics. pp. 977–982 (Oct 2011). https://doi.org/10.1109/ICSMC.2011.6083796, iSSN: 1062-922X
6. Brückner, M., Kanzow, C., Scheffer, T.: Static prediction games for adversarial learning problems. The Journal of Machine Learning Research **13**(1), 2617–2654 (2012)

7. de Casanove, O., Sèdes, F.: Apprentissage adverse et algorithmes de détection d'évènements : une première typologie. Rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI 2022) (May 2022), https://hal.archives-ouvertes.fr/hal-03668829, poster
8. Chan, P.P.K., Yang, C., Yeung, D.S., Ng, W.W.Y.: Spam filtering for short messages in adversarial environment. Neurocomputing **155**, 167–176 (May 2015). https://doi.org/10.1016/j.neucom.2014.12.034
9. Duddu, V.: A survey of adversarial machine learning in cyber warfare. Defence Science Journal **68**(4), 356 (2018)
10. Hasan, M., Orgun, M.A., Schwitter, R.: A survey on real-time event detection from the Twitter data stream. Journal of Information Science **44**(4), 443–463 (Aug 2018). https://doi.org/10.1177/0165551517698564
11. Hasan, M., Orgun, M.A., Schwitter, R.: Real-time event detection from the Twitter data stream using the TwitterNews+ Framework. Information Processing & Management **56**(3), 1146–1165 (May 2019). https://doi.org/10.1016/j.ipm.2018.03.001
12. Imam, N.H., Vassilakis, V.G.: A Survey of Attacks Against Twitter Spam Detectors in an Adversarial Environment. Robotics **8**(3), 50 (Sep 2019). https://doi.org/10.3390/robotics8030050
13. Khandpur, R.P., Ji, T., Jan, S., Wang, G., Lu, C.T., Ramakrishnan, N.: Crowdsourcing cybersecurity: Cyber attack detection using social media. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. pp. 1049–1057 (2017)
14. Kumar, S., Liu, H., Mehta, S., Subramaniam, L.V.: From Tweets to Events: Exploring a Scalable Solution for Twitter Streams. arXiv:1405.1392 [cs] (May 2014), arXiv: 1405.1392
15. McMinn, A.J., Jose, J.M.: Real-Time Entity-Based Event Detection for Twitter. In: Mothe, J., Savoy, J., Kamps, J., Pinel-Sauvagnat, K., Jones, G., San Juan, E., Capellato, L., Ferro, N. (eds.) Experimental IR Meets Multilinguality, Multimodality, and Interaction. pp. 65–77. Lecture Notes in Computer Science, Springer International Publishing, Cham (2015)
16. OWASP: Threat modeling. https://owasp.org/www-community/Threat_Modeling (2022)
17. Petrović, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to twitter. In: Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics. pp. 181–189 (2010)
18. Ritter, A., Wright, E., Casey, W., Mitchell, T.: Weakly supervised extraction of computer security events from twitter. In: Proceedings of the 24th International Conference on World Wide Web. p. 896–905. WWW '15, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2015). https://doi.org/10.1145/2736277.2741083
19. Sabottke, C., Suciu, O., Dumitras, T.: Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In: 24th USENIX Security Symposium (USENIX Security 15). pp. 1041–1056. USENIX Association, Washington, D.C. (Aug 2015), https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/sabottke
20. Samonas, S., Coss, D.: The cia strikes back: Redefining confidentiality, integrity and availability in security. Journal of Information System Security **10**(3) (2014)

21. Vamvoudakis, K.G., Hespanha, J.P., Sinopoli, B., Mo, Y.: Adversarial detection as a zero-sum game. In: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC). pp. 7133–7138 (2012). https://doi.org/10.1109/CDC.2012.6426383
22. Wang, X., Li, J., Kuang, X., Tan, Y.a., Li, J.: The security of machine learning in an adversarial setting: A survey. Journal of Parallel and Distributed Computing **130**, 12–23 (Aug 2019). https://doi.org/10.1016/j.jpdc.2019.03.003, https://www.sciencedirect.com/science/article/pii/S0743731518309183
23. Wu, C., Li, X., Pan, W., Liu, J., Wu, L.: Zero-sum game-based optimal secure control under actuator attacks. IEEE Transactions on Automatic Control **66**(8), 3773–3780 (2021). https://doi.org/10.1109/TAC.2020.3029342
24. Zhou, R., Lin, J., Liu, L., Ye, M., Wei, S.: Analysis of SDN Attack and Defense Strategy Based on Zero-Sum Game. In: Ren, J., Hussain, A., Zhao, H., Huang, K., Zheng, J., Cai, J., Chen, R., Xiao, Y. (eds.) Advances in Brain Inspired Cognitive Systems. pp. 479–485. Lecture Notes in Computer Science, Springer International Publishing, Cham (2020)
25. Zhou, Y., Kantarcioglu, M., Xi, B.: A Game Theoretic Perspective on Adversarial Machine Learning and Related Cybersecurity Applications. In: Game Theory and Machine Learning for Cyber Security, chap. 13, pp. 231–269. John Wiley & Sons, Ltd (2021), https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119723950.ch13