Multi-Agent Case-Based Reasoning: a Network Intrusion Detection System

Jakob Michael Schoenborn^{1,2}, Klaus-Dieter Althoff^{1,2}

¹University of Hildesheim, Samelsonplatz 1, 31141 Hildesheim, Germany ²German Research Center for Artificial Intelligence (DFKI), Trippstadter Str. 122, 67663 Kaiserslautern, Germany

Abstract

We propose a multi-agent case-based reasoning system to detect malicious traffic in a network. We introduce ten topic agents, including nine different attack categories and one agent covering normal, benign traffic. Using the four knowledge containers, we fill our case base with the labeled training data of the commonly used UNSW_NB15 data set, in sum 82332 cases with (mostly numeric) 47 attribute features. We calculate average values for each attribute and search for outliers to identify characteristic attributes for each attack category, increasing weights in the amalgamation function for those attributes. For local similarities, we define polynomial similarity functions with heavily decreasing similarity for differing attribute-values pairs between case and query, depending on the range of the attribute values. *Purpose*. The proposed system is aimed to detect malicious traffic, such as denial of service attacks, to alert the security engineer of a company or even an individual person. The system can either be included into already existing intrusion detection systems, support regular log analysis, or used for forensic analysis.

Findings. We were able to successfully detect Generic and Fuzzer attacks with a high true-positive rate. With additional adjustments, we are confident to successfully detect more attack categories.

Implications and value. Despite only detecting two out of nine attacks, we are confident for this approach to provide an important step into the right direction with possible improvements and opportunities for fruitful synergies and discussions inside the security domain and case-based reasoning community.

Keywords

Case-based Reasoning, Intrusion Detection System, IT-Security, Multi-agent system, SEASALT, myCBR

1. Introduction

Intellectual property needs to be protected from unauthorized access. With the increasing amount of companies going online through industry 4.0 standards for higher globalization, the amount of cybercrimes increases as well. Especially due to the COVID-19 pandemic, most organizations were furthermore forced into digitisation, for example allowing their employees to establish a home office. Properly integrating digitisation into a company environment takes time and effort. Unfortunately, security is not the top priority during this phase. This development offers a wider range of possible targets for cybercriminals to move through a companies' network. It is not uncommon for companies to have multiple thousands of known vulnerability issues - especially for larger IT companies. However, the upper management

LWDA'22: Lernen, Wissen, Daten, Analysen. October 05–07, 2022, Hildesheim, Germany

Schoenb@uni-hildesheim.de (J. M. Schoenborn) Schoenborn

https://github.com/jmschoenborn (J. M. Schoenborn)

D 0000-0001-9669-8148 (J. M. Schoenborn)

^{© 02022} Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

might only provide budget to fix a smaller percentage of those issues, since investment into security does not necessarily directly create visible value. Certainly, there are different scoring systems to rate the severance of a given security issue, such as CVSS¹. Using scores, the most severe security issues can be identified and ranked. However, the given budget might not be enough to fix all of those, leaving the choice which issues to be fixed to the security engineers. On the one hand, it might be a reasonable approach to fix SQL injections (SQLi) and Cross Site Scripting (XSS) issues, as they might cover the most percentage of the critical findings². On the other hand, leaving certain known other vulnerabilities open might be very risky, resulting in an economical shut down of a company. To ease the difficulty in the decision making process, more information about the potential attacker could be helpful.

This contribution is the first step in a series of multiple necessarily required steps to reach the previously described goal of identifying possible attackers, alongside with explainability of the given decision. Based on the current state of the art, case-based reasoning (CBR) seems to be a promising candidate as an addition to current intrusion detection systems: a relatively small casebase can already be sufficient for detecting novel attacks, which are usually just slightly adjusted from known attacks. CBR is a methodology usually cycling through four steps: retrieve, reuse, revise, retain. Knowledge representation is supported by four knowledge containers: vocabulary, similarity measures, adaptation knowledge, case base. Generally, CBR follows the paradigm "Similar problems have similar solutions.", thus, retrieving experience from old situations (*cases*) to solve a new occurring problem. The possibility of modularization by using the SEASALT architecture [1] and initializing CBR agents whenever needed is an advantage to adjust resources accordingly to incoming attacks - the more incoming data and attacks, the more agents can be initialized.

Nevertheless, unfortunately, there is only very limited research inside of the CBR community towards this area, whereas security is a domain, which affects everyone of us - not only companies, but also us as individuals. A recent study of the International Telecommunication Union reports a surge of 800 million users on the internet from 2019 to 2021, which results in 4,9 billion - or 63 % of the world population - total users, with the trend still going upwards [2]. We are confident in CBR to be a valuable addition to existing security mechanisms to protect these users, in addition to persons who do not use the internet but are indirectly dependant of, for example, their bank having proper security mechanisms.

The next section describes related work to position our contribution within other approaches in a similar direction. We present our concept and the used data set in section 3, followed by a description of the practical implementation in section 4. Closing, we evaluate and discuss our results in section 5 and 6.

¹Common Vulnerability Scoring System, see https://nvd.nist.gov/vuln-metrics/cvss, last validation: 03/18/2022 ²see https://owasp.org/www-project-top-ten/, last validation 03/18/2022

2. Related Work

As the meaning of the term *security* spans over various areas, we used a different combination of multiple keywords such as *case-based reasoning*, *security*, *intrusion*, *detection*, *system*, *network*, *malware*, *malicious*, *traffic*, ... to receive a competitive overview on the current state of the art regarding the usage of case-based reasoning in the IT-security domain. In the following, we present the most similar literature that we could identify in comparison to our case, sorted into the categories *Profiling*, *General Intrusion Detection* and *Attack-centric research*, which all also apply to our contribution.

Profiling. Regarding to our long-term goal in matching security issues to potential attacker groups, Kapatenakis et al. started 2014 by '*examining whether a CBR approach can help security and forensic investigators to profile human attackers with regards to their behavioural, demographic and technical characteristics' [3]. This data has been used to formulate cases of an attacker and also cases for attacks. Within their experiments including 87 individuals to shut down targeted systems, they received an average classification rate of 69 %. Han et al. developed a tool called '<i>Web-Hacking Profiling using CBR*' [4]. The mostly South Korean authors investigated attacks with the same signatures, assigned to North Korea. However, there are only very few details about the implementation of the system and not open source - the realization is at least questionable in multiple regards.

General intrusion detection. El Ajjouri et al. suggest a case retrieval implementation for intrusion detection based on multi-agent case-based reasoning [5], using jColibri and begin with an initial set of five intrusion cases. Similar to Han et al., the case structure focuses primarily on the protocol, IP addresses, and packets contents. Here, it should be noted that over time, different actors may have the same IP address³. In this paper, "agents" are rather focusing on different tasks in the overall system (sniffer-, preprocessor-, filter-, decider-, generator-, CBR agent), than focusing on the attacks itself - resulting in a more static distributed problem solving than a multi-agent system. Additionally, the weight for each attribute is the same and each attribute is checked whether the value for the case and the query are equal or not, missing the finesse of CBR. Erbacher and Hutchinson are providing a CBR system for automated cyber security report generation, in addition to reporting hostile actors, which are trying to take actions to hide from established defensive measurements [6]. Wand et al. apply graph theory and case templates to represent vulnerabilities and reason the attack paths using the graph structures [7]. Creating this structure also allows to assess the current security situation based in the network [7].

Attack-centric research. Long et al. focused on the detection of distributed denial of service (DDoS) attacks [8], similar to our DoS agent. The authors are using multi-sensor data, which is included in two different DARPA data sets and their results show that their CBR approach is effective - which could also be integrated into our proposed system. Slightly out of our scope, but still relevant: Abutair and Belghith provide a CBR system to detect at least 96% of phishing emails [9]. Emails, which try to trick the receiver to click a malicious link and thus installing malware on their computers. Phishing emails are a subcategory of social engineering, which Lansley et al. [10] focused on and received similar results to Abutair and Belghith.

³IPv6 will strongly lessen the likelihood for this case to happen.

3. Concept

3.1. Dataset

To choose a data set, Ring et al. provided a survey of 34 network-based intrusion detection data sets [11]. Their general recommendation mentions four possible data sets whereas the CICIDS 2017 and UNSW_NB15 offer a wide range of attack scenarios [11], which fits to our needs. While the former contains more detailed metadata, we chose the latter as CBR does not necessarily need a large amount of data to provide proper classifications.

UNSW_NB15 has been created by N. Moustafa and J. Slay and spans over 47 different attributes, which can be sub-categorized into basic features, connection features, content features, time features, additional generated features, and labeled features. The attack categories are labeled as 1, while normal traffic is labeled as 0 and are described in Table 1 [12].

The dataset is split into training data (82332 packages, thus, in sum 82332 cases) and testing data (175341 packages). For a detailed description of the 47 features, we refer to the original publication by Moustafa and Slay [12]. Table 2 shows an excerpt of the calculated average values of the UNSW_NB15 training data set.

Fuzzers	attacker attempts to discover security loopholes in a network by feeding it with massive inputting of random data to make it crash.					
Analysis	a type of variety intrusions that penetrate the web applications via ports,					
	emails, and web scripts.					
Backdoor	a technique of bypassing a stealthy normal authentication, securing unautho-					
	rized remote access to a device.					
DoS	intrusion which disrupts the computer resources, to be extremely busy in					
	order to prevent the authorized requests from accessing a device.					
Exploit	a sequence of instructions that takes advantage of a vulnerability to be caused					
	by an unintentional behavior on a host or network.					
Generic	technique that establishes against every block-cipher to collision without					
	respect to the configuration of the block-cipher.					
Reconnaissance	can be defined as a probe; an attack that gathers information about a computer					
	network to evade its security controls.					
Shellcode	an attack in which the attacker penetrates a slight piece of code starting from					
	a shell to control the compromised machine.					
Worms	an attack whereby the attacker replicates itself in order to spread on other					
	computers. Often, it uses a computer network to spread itself.					

Table 1

Description of the attack categories based on Moustafa and Slay [12]

3.2. CBR agents and their knowledge containers

Observing network traffic to identify a possible attacker can be a tedious task as thousands of packages per second are rushing through the network. Therefore, in terms of efficiency and scalability, we suggest a multi-agent system with at least one agent per attack category and one agent for normal data traffic. With the given data set, this leaves us with ten agents - with the

Attribute	Overall	Backd.	Fuzzer	Generic	Shellc.	Worm
protocol	tcp(43095)	unas(206)	tcp(3713)	udp(18303)	tcp(139)	tcp(38)
state	FIN(39339)	INT(522)	FIN(3703)	INT(18325)	FIN(192)	FIN(38)
duration	1.01	0,93	2,13	0,07	0,36	1,06
sbytes	7994,44	581,32	5197,3	552,8	542,11	1819,5
dbytes	13235,35	168,13	513,29	1830,8	149,58	68940,28
sttl	180,97	248,1	253,98	251,64	254	254
dttl	95,72	22,67	154,36	7,04	128,67	217,64
sloss	4,76	0,22	3,41	0,22	1,02	1,82
dloss	6,31	0,22	1,18	0,74	0,52	26,78
service	-(47153)	-(572)	-(5527)	dns(18162)	-(378)	http(34)
sload	65447 384	123637 232	108336 848	92705 224	122868 016	69576 520
dload	630569,07	1694,86	3372,86	6044,48	2580,43	141641,49
spkts	18,67	4,39	11,8	2,8	6,07	16,78
dpkts	17,55	0,84	5,8	1,64	3,36	57,32
rate	82403	154631,5	67530,1	195076,07	28347,25	20921,27
swin	133,46	22,31	156,19	7,03	130,2	220,23
dwin	128,29	22,31	156,19	7,03	130,2	220,23
stcpb	1084642816	185336 576	1340419 072	60907 388	1103935 232	1738327 296
dtcpb	1073468 224	195774 848	1299013760	59711 008	1119258 880	1699302 272
smean	139,53	98,9	214,88	65	123,22	186,91
dmean	116,28	9,66	36,5	12,35	22,84	238,28
trans_depth	0,1	0,02	0,03	0,01	0	0,62
response	1595,34	0,39	4,73	191,85	0	24569,3
sjit	636,04	464,01	26552,59	175,94	2299,7	2969,66
djit	535,18	13,14	1072,12	49,35	85,47	343,46
sinpkt	755,39	38,25	378,72	3,1	37,96	53,52
dinpkt	121,71	7,47	402,84	2,56	52,73	66,38
tcprtt	0,06	0,01	0,1	0,01	0,06	0,12
synack	0,03	0,01	0,05	0,01	0,03	0,07
ackdat	0,03	0,01	0,05	0,01	0,04	0,06
ct_srv_src	9,55	10,91	7,79	23,1	2,6	1,69
ct_state_ttl	1,37	1,95	1,41	1,98	1,5	1,14
ct_dst_ltm	5,75	4,14	2,62	15,45	1,35	1,19
ct_src_dport_ltm	4,93	3,9	2,45	15,35	1	1,12
ct_dst_sport_ltm	3,67	3,89	1,7	11,43	1	1
ct_dst_src_ltm	7,46	4,21	3,25	3,92	1,19	1,03
is_ftp_login	0,01	0	0,01	0	0	0
ct_ftp_cmd	0,01	0	0,01	0	0	0
ct_flw_http_mthd	0,13	0,02	0,03	0,01	0	0,62
ct_src_ltm	6,47	5,93	3,42	15,73	1,68	1,98
ct_srv_dst	9,17	10,11	6,92	23,05	1,59	1,5
is_sm_ips	0,02	0	0	0	0	0

Table 2

Excerpt of the UNSW_NB15 training data set based on [12], calculating average values and highlighting the highest values. Either average values for float attributes or the most occurred value for string attributes (such as Protocol, Service, and State (with count in brackets)), sorted by attack category. *Italic* values are not the highest, but still distinct in contrast to other attack categories.

possibility of multithreading, increasing the amount of agents based on the amount of incoming traffic for scalability. The agents can easily be incorporated into multi-agent frameworks such as the SEASALT architecture [1].

We use case-based reasoning agents, each containing four knowledge containers according to M. M. Richter [13]: *vocabulary, similarity measure, adaptation knowledge, casebase.*

In terms of *vocabulary* structure, we use an attribute-value representation, as the measurable data contains 35 attributes in addition to 12 derived attributes (see section 3.1). No set of attributes contains unknown values, thus only complete situations are evaluated. Correlations between certain attributes could not be detected, yet. Certainly, attributes contain correlation to attack categories, which will be covered next in the *similarity measure* container.

Following the weighted Hamming similarity measure

$$sim(q, p) = \sum (g_i \times sim_i(q_i, p_i) \mid 1 \le i \le n)$$
(1)

as Richter and others suggested, we utilize the local-global principle [14, 13, 15]. For local measures sim_i , we inspect the attributes A_i based on their minimum and maximum values and calibrate a symmetrical polynomial function with heavily decreasing similarity for differing attributes based on the variability of an attribute. The narrower the data points of an attribute, the stronger decreasing the similarity function. For the amalgamation function, we set values for the non-negative real weight vector coefficients $g = (g_1, ..., g_n)$, normalized to $\sum g_i = 1$ [13].

For the values of *g*, we calculate the average value of each attribute ranging over the whole data set and also the average value filtered by each attack category. This enables us to identify attributes which seem to hint at a certain attack for given values. For example, *spkts* (='*source packets*') depicts the source-to-destination packet count with the following calculated average values for each attack category, reading: "For an exploit attack, 37,7 packets have been sent on average from the source to the destination":

AVG	Analysis	Backd.	DoS	Exploit	 Normal	Recon	Shellcode	Worm
18,67	3,12	4,39	28,9	37,7	 6,97	6,97	6,07	16,78

While the average on the whole data set is at 18,67 and has lower values for other attacks, Exploit points out with an average value of 37,7 packets from source to destination. This confirms the intuitive expectation of an exploit attack: *to exploit* means in the IT-security context to systematically abuse known security issues of a given system. However, it needs to be tested, which security issues the target might have - resulting into multiple requests and consequently an increased amount of packets running from source-to-destination. Therefore, *spkts* receives a higher weight than other attributes for the exploit agent. The more distinct an attribute-value, the higher the weight.

On a similar notion, this situation also holds true for the denial of service (DoS) attack: with a value of 28,9, it is also distinct enough from other attack categories, which range on average between 2,8 and 16,78. Therefore, we are also able to identify attribute values, which are not the maximum, but still unique to a certain attack category (highlighted *italic* in Table 2) - and use this information to increase the weight of the given attribute for the corresponding agent.

We repeat this process for each agent and each attribute.

Each agent is trained to detect its respective attack, i. e., a DoS agent only contains cases labeled with denial of service attacks. Thus, the *case base* contains the experiences based on the training data set. We store each line of the data set as a case, resulting in 82332 cases. However, there is still room left for improvement regarding the two conflicting goals: having the case base as large as possible for increased competence knowledge while having the case base as small as possible for better efficiency. We discuss this problem and the yet missing *adaptation knowledge* further in section 6.

For each package in the testing data set, each agent 'votes' by submitting its *n* most similar cases to a coordination agent. For now, it will be left open for discussion in section 5 whether *n* should be 1 to submit only the most similar case or to calculate the average similarity of n > 1 cases to reduce the risk of outliers. For our experiments, we choose n = 10 to remove outliers and gain insight whether the similarity of other similar cases is decreasing correctly, as to be expected. The votes with the highest similarities will be reported to the (human) user. After receiving the results, the user may decide which agent is ultimately correct - leaving the responsibility and legal liability to the human user - and might choose to start further actions to stop the attack, such as blocking the source IP address of the potential attacker.

4. Implementation

We implemented the system described in section 3 by using myCBR 3.4 and the programming language JAVA. MyCBR is an open-source similarity-based retrieval tool and software development kit (SDK)⁴ and has been further developed by students of the University of Hildesheim and by the authors, hence the increased version number. MyCBR 3.4 and the whole prototype described in this contribution are available under the LGPL licence at Github⁵.

Figure 1 provides a brief overview. In favor of simplicity in the presentation, we do not list the argument parameters and return types of the used functions. First, the user will be provided with a simple graphical web interface, asking to import either training- or testing data. Either way, the class Stats will provide the knowledge engineer with average statistics (among other functions) on the data set by visibly printing out the statistics to the IDE console or log file especially important for viewing the training data.

In case of importing the training data set, new agents are initialized with the given training data. As described in section 3, each initialized agent learns only cases of its corresponding attack category. Each topic agent, such as analysis agent, backdoor agent, DoS agent, ... is extending the abstract class *Agent*, which forces each agent to implement the methods initProject(), initCaseBase(), addCases(), changeWeights(), startQuery(), print(). Initializing an agent forces the agent to create a project, which begins to create the four knowledge containers, especially defining the similarity measures. For String values, such as protocol, service and state, the Levenshtein similarity function has been used. For all other (numeric) attributes, a symmetrical polynomial function has been established (according to section 3.2). Afterwards, the casebase will be initialized and allocated to the initialized project. Adding the cases (lines of the training data) to the casebase and afterwards exporting the agent to the local storage disk

⁴see http://mycbr-project.org/index.html

⁵see https://github.com/jmschoenborn

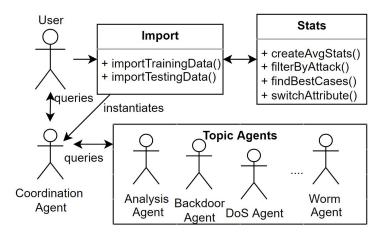


Figure 1: Overview of the implementation, including 10 topic agents.

finalizes the initialization process of the agent. Using the exported files allows us to load the agent for the testing data set. The agents can easily be adjusted to fit for other training data sets as well.

In case of importing the testing data set, already established agents are activated by the coordination agent. Additionally, the user may provide a positive number *a* for the minimum number of different attack categories that should be considered and a positive number $c \ge a$ for the number of cases that should be presented. This allows the user to receive a broader picture of the similarity distribution between multiple attack categories to prevent missing out on ambiguous results. For each line (case) of the testing data set, the agents provide their *n* best matching cases and the coordination agent provides the best *c* cases among *a* attack categories to the user. Figure 2 shows the end results after the voting phase by showing the 4 most similar distinct attack categories along with the 10 best cases overall.

Step 1, Attack: Fuzzers (ID: 48024).
Best 4 distinct attacks and 11 cases
Rank 1.) ID: 44364 (0.9319408901562877): Fuzzers
Rank 2.) ID: 62813 (0.8736727022100131): Generic
Rank 3.) ID: 60018 (0.8478893/229311343): Backdoor
Rank 4.) ID: 44845 (0.83907495193343): Shellcode
---Rank 5.) ID: 44364 (0.9319408901562877): Fuzzers
Rank 6.) ID: 45836 (0.9037475502033318): Fuzzers
Rank 7.) ID: 45562 (0.895840410276636): Fuzzers
Rank 8.) ID: 48788 (0.8908535009905201): Fuzzers
Rank 10.) ID: 45534 (0.8878862179872327): Fuzzers
Rank 10.) ID: 45534 (0.8851765976233792): Fuzzers
Rank 11.) ID: 50816 (0.8851765976233792): Fuzzers

Figure 2: Example result after voting. The first four ranks are **not** the overall four best cases, but instead the four best cases of four distinct attack categories. For the test case with ID 48024, labeled as Fuzzers, the Fuzzers agent provides the best case with 93 % similarity (ID 44364). Additionally, 7 further distinct cases of the Fuzzers agent casebase with at least 88 % similarity are provided. Thus, in terms of a majority votum, the attack has been correctly identified.

5. Evaluation

5.1. Results

Table 3 presents our results of querying the topic agents with the testing data set. We provided the first 1000 cases of an attack category and each activated topic agent voted with their n = 10 most similar cases. Out of this pool of best cases (50 with 5 active agents), the 10 most similar cases have been chosen. Each correct vote will be counted. If there are at least six correct votes, the query is considered as correctly classified. All correct votes will be summarized and a true-positive rate (*TPR*) in the last line provided. Consequently, 100 - TPR depicts the false-positive rate (*FPR*).

Majority Vote x	Backdoor	Fuzzers	Generic	Shellcode	Worm		
0	566	32	2	93	94		
1	254	33	1	150	33		
2	33	16	1	74	3		
3	12	14	1	111	0		
4	20	26	1	113	0		
5	24	18	0	88	0		
6	35	25	1	58	0		
7	33	40	3	75	0		
8	16	56	6	93	0		
9	7	142	21	81	0		
10	0	598	963	64	0		
TPR	9,1 %	86,1 %	99,4 %	37,1 %	0 %		
FPR	90,9 %	13,9 %	0,6 %	62,9 %	100 %		
Results of other approaches (no CBR) using the same dataset							
Wheelus et al. [16]	Wheelus et al. [16] ranging from 69 % to 83 % TPR for all attacks						
Pratomo et al. [17]	69,21 % TPR on average for all attacks						
Mebawondu et al [18]	76,96 % TPR for all attacks						

Table 3

Results. Results are the absolute number x of representative cases to the corresponding attack category. Additionally, other approaches and their results to set our results into context.

5.2. Limitations

During the first test runs we encountered a few challenges with the data set, which resulted into limitations to this prototype version. We are confident to lift these limitations in future work.

1. Redundancy

As to be expected, the training data set contained multiple redundant cases, containting the same attribute-value pairs. If these cases turn out to be the most similar case for a given testing data input query case, the majority vote will easily be flooded by the redundant cases. A relatively quick fix to this challenge would be to simply remove redundant cases and remain with one pivotal case. The occurrence of a large amount of redundant cases might contain context information, which should not easily be discarded. However, a more elegant and efficient way would be a proper introduction of case base maintenance under the aspect of pivotal cases, and coverage and reachability of cases in a casebase as introduced by Smyth & Keane [19].

 \rightarrow For our tests, we focused on the Backdoor, Fuzzers, Generic, Shellcode, and Worm agents which per se do not contain redundant cases.

2. Same case, different attack category

We identified multiple cases with exactly the same attribute-value pairs, but different attack category labels (249-Analysis, 710-DoS, 1413-Reconnaissance, 1416-Exploits, 3421-Fuzzers). During the training phase, and tests within the training data set, this resulted into a 100 % similarity for a given case for multiple different attack categories, which is not a desirable result. However, during the testing phase, this problem actually did not seem to occur, thus **no limitation** here. However, for the sake of completeness, it should be mentioned.

3. Resources and iterations

Due to insufficient resources, the machine which executed the testing data set (CPU i7-4790K @ 4 GHz, 16GB RAM) could not provide enough resources to process all test cases, especially when reaching 1000+ cases. Therefore, we filtered the testing data set by attack category and moved through the first 1000 test cases per attack category. According to the trend of the results, this number is still representative for the remaining test data cases. The table below shows the attack category and the corresponding number of testing data cases:

BackdoorFuzzersGenericShellcodeWorm174618184400001113130

6. Discussion and future work

We presented a novel approach of using case-based reasoning for supporting intrusion detection systems by using the UNSW_NB15 data set for training and testing purposes. We established a multi-agent CBR system prototype with at least one topic agent for each type of attack category, training these agents with the given training data set and modeling the similarity measures based on the identification of distinct attribute-value pairs, characterizing given attack categories.

Despite the limitations described in section 5.2, the results in Table 3 show very different results. For Fuzzers and Generic, we receive very positive results by correctly identifying 86,1% and 99,4% of the testing data - better, than other (non-CBR) approaches. However, other agents do not achieve any competitive results, especially the Worm agent with 0% correct majority votes. This is most likely due to only containing 44 cases in the training data set whereas the Generic agent contains 18871 cases. Yet, based on the data, Worms contain by far the most distinct and characteristic values, which leaves us optimistic to receive better results after further fine-tuning the local similarity measures. The backdoor agent also only contains 583 cases in its casebase and additionally contains only two distinct attributes. Other attributes can

wrongly be attributed to other attack categories, increasing the risk of false-positives. Despite not actively evaluated here, the same problem arises for Analysis and Exploit: both attack categories share very similar characteristics.

However, there is still a lot of room for improvement as mentioned above. Introducing a proper casebase maintenance system for removing redundant cases without reducing the competence of the system by taking coverage and reachability into account, further adjustment of the localand global similarities, and learning further cases using additional datasets may improve the overall performance of the system. Additionally, we have not integrated adaptation knowledge yet, which still has to be identified but also provides a promising increase of performance. These steps remain for future work and we hope to spark some interest in using CBR inside of the IT security domain.

References

- K. Bach, Knowledge Acquisition for Case-Based Reasoning Systems, Ph.D. thesis, University of Hildesheim, 2013. URL: http://www.dr.hut-verlag.de/978-3-8439-1357-7.html.
- [2] I. T. Union, Measuring digital development facts and figures 2021, ITUPublications, Geneva (2021). URL: https://www.itu.int/en/ITUD/Statistics/Documents/facts/FactsFigures2021. pdf, last validation: 04/30/2022.
- [3] S. Kapetanakis, A. Filippoupolitis, G. Loukas, T. Saad Al Murayziq, Profiling cyber attacks using case-based reasoning, in: 19th UK Workshop on Case-Based Reasoning, 2014, pp. 39–48.
- [4] M. L. Han, H. C. Han, A. R. Kang, B. I. Kwak, A. Mohaisen, H. K. Kim, Whap: Web-hacking profiling using case-based reasoning, in: 2016 IEEE Conference on Communications and Network Security (CNS), 2016, pp. 344–345.
- [5] H. M. Mohssine El Ajjouri, Siham Benhadou, Case retrieval implementation for intrusion detection architecture based on multi agent systems and case based reasoning technique, in: International Journal of Scientific & Engineering Research, volume 10, 2019, pp. 1184–1189. ISSN 2229-5518.
- [6] R. F. Erbacher, S. E. Hutchinson, Extending case-based reasoning to network alert reporting, in: 2012 International Conference on Cyber Security, 2012, pp. 187–194.
- [7] Y. Wang, A. Zhu, J. Zhang, A case-based reasoning method for network security situation analysis, in: 2011 International Conference on Control, Automation and Systems Engineering (CASE), 2011, pp. 1–4.
- [8] J. Long, D. Schwartz, S. Stoecklin, Application of case-based reasoning to multi-sensor network intrusion detection, in: Proceedings of the 4th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, CIMMACS'05, World Scientific and Engineering Academy and Society (WSEAS), Stevens Point, Wisconsin, USA, 2005, p. 260–269.
- [9] H. Y. Abutair, A. Belghith, Using case-based reasoning for phishing detection, Procedia Computer Science 109 (2017) 281–288. 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal.

- [10] M. Lansley, N. Polatidis, S. Kapetanakis, K. Amin, G. Samakovitis, M. Petridis, Seen the villains: Detecting social engineering attacks using case-based reasoning and deep learning, in: S. Kapetanakis, H. Borck (Eds.), Workshops Proceedings for the Twenty-seventh International Conference on Case-Based Reasoning co-located with the Twenty-seventh International Conference on Case-Based Reasoning (ICCBR 2019), Otzenhausen, Germany, September 8-12, 2019, volume 2567 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2019, pp. 39–48. URL: http://ceur-ws.org/Vol-2567/paper4.pdf.
- [11] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, A. Hotho, A survey of network-based intrusion detection data sets, Computers & Security 86 (2019) 147–167.
- [12] N. Moustafa, J. Slay, UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set), in: 2015 Military Communications and Information Systems Conference (MilCIS), 2015, pp. 1–6.
- [13] M. M. Richter, The knowledge contained in similarity measures, Invited Talk at the First International Conference on Case-Based Reasoning, ICCBR'95, Sesimbra, Portugal, 1995.
- [14] R. Bergmann, Experience Management: Foundations, Development Methodology, and Internet-Based Applications, volume 2432 of *Lecture Notes in Computer Science*, Springer, 2002. URL: https://doi.org/10.1007/3-540-45759-3.
- [15] S. Wess, Fallbasiertes Problemlösen in wissensbasierten Systemen zur Entscheidungsunterstützung und Diagnostik: Grundlagen, Systeme und Anwendungen (translated: Case-based problem solving in knowledge-based systems for decision support and diagnostic: basics, systems and applications), Ph.D. thesis, University of Kaiserslautern, 1995. Infix-Verlag.
- [16] C. Wheelus, E. Bou-Harb, X. Zhu, Tackling class imbalance in cyber security datasets, in: 2018 IEEE International Conference on Information Reuse and Integration (IRI), 2018, pp. 229–232.
- [17] B. A. Pratomo, P. Burnap, G. Theodorakopoulos, Unsupervised approach for detecting low rate attacks on network traffic with autoencoder, in: 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), 2018, pp. 1–8.
- [18] J. O. Mebawondu, O. D. Alowolodu, J. O. Mebawondu, A. O. Adetunmbi, Network intrusion detection system using supervised learning paradigm, Scientific African 9 (2020) e00497.
- [19] B. Smyth, M. T. Keane, Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes, Morgan Kaufmann, 1995, pp. 377–383. URL: http://ijcai.org/Proceedings/95-1/Papers/050.pdf.