Evaluation through deployment of the Multi-agent System For Advanced Persistent Threat Detection framework in a Cyber Range environment

Georgi Nikolov[1 [0000−0002−9020−8408]], Thibault Debatty[1 [0000−0003−2373−566X]], and Wim Mees[1 [0000−0002−0696−8093]]

Royal Military Academy, Avenue de la Renaissance 30, 1000 Brussels, Belgium
contact@cylab.be
https://cylab.be

Abstract. In recent years, cyber attacks have evolved and multiplied exponentially, largely because of the ease with which malicious actors can obtain the tools needed, and also through the organisation and management of state funded groups. These threats have become even more sophisticated, often mimicking normal network behavior, ultimately causing enormous infrastructural damage. To combat these new sophisticated attacks, we designed the Multi-agent System For Advanced Persistent Threat (APT) Detection framework, or MASFAD for short. It is not evident to gauge the capabilities of such tools without deployment in the field, but it is often not as simple as adding them to an existing Security Operations Center (SOC). As an intermediate step, between the design and real-world deployment, we opted for testing our framework in our cyber range, closely simulating the SOC environment. The results of the cyber range evaluation showed a clear benefit in regards to the ease of deployment, accessibility and repeatability, greatly enhancing the development and testing process. This paper will give an overview of our thought-process in the selection of the best technologies to attain that goal and the insights we gained through-out the deployment and simulation.
We will discuss the integration of our framework in a cyber range, to test its performance and evaluate its detection capabilities in a virtual environment. The description of the framework will be followed by an overview of the cyber range set-up and configuration, ending with a description of the evaluation metrics and the obtained results.

Keywords: anomaly-based analysis · command & control channel · advanced persistent threat · evaluation metrics · cyber range · GHOSTS · ELK

1 Introduction

The threats targeting corporate and operational networks evolve at an extraordinary speed. Each month, information regarding new attacks perpetrated against private citizens, companies and government institutions is made public. In the early years, these attacks were mostly perpetrated by lone-wolf hackers, trying to test their limits and knowledge, but lately a new threat has emerged – state-sponsored organizations that target key government or operational networks. Such cases are of major significance, as cyber attacks on countries can cause major harm to multiple sectors- from the financial, to the defense and private sector. Through the use of various techniques, hackers have managed to penetrate a wide variety of high-value targets. One example of which, is the recent attack on the United Nations [7], where, through the use of stolen credentials, malicious actors got access to sensitive information and remained undetected in the system for more than 4 months. Another, even more striking example, is the attack perpetrated against the Belgian Defense [2], where a newly discovered zero-day vulnerability [1], in the widely used open Java library "log4j", was used to penetrate the government institution and render it non-operational for a significant amount of time, on top of the possibility of sensitive information being exfiltrated. It has become obvious that new types of detection techniques are needed in the detection of this type of sophisticated attacks, ones less focused on signature-based detection, and more aimed towards the use of anomaly-based detection and behavioral analysis. In recent years, multiple reviews of malware detection approaches have been conducted, and the majority of them agree that signature-based detection does not perform well against zero-day vulnerabilities [19] [9], where no signature of the attacks exist, to be compared against. New heuristics-, behavior- and anomaly-based techniques, such as our MASFAD framework [15], offer greater detection capabilities in the detection of previously unknown threats.

Evaluating an APT detection framework can be a difficult endeavor- an ample amount of data is needed, combined with simulated attacks, based on existing real world malware samples. The initial evaluation of our framework [16] proved the efficacy of the anomaly- and behavior-based approach. The next step in the evaluation is to deploy the framework in a cyber range [11], a sophisticated tool for the training of cyber defense situation awareness (CDSA) and the simulation of complex networks. Our key question was to determine the benefits of the cyber range for cyber research and development.

In this paper we will introduce the Multi-agent Ranking Framework (MARK) with the goal to detect Advanced Persistent Threats (APT) in short, the cyber range we used for the deployment of the framework, followed by the deployment specification and configuration. Finally a short description of the retrieved results will be given and we will discuss the benefits of testing our framework in such an environment.

2 The Multi-Agent Ranking Framework

Contrary to currently available intrusion detection system (IDS) solutions, the proposed framework focuses on the use of domain knowledge of the behavior of attacks and adaptability, in regards to emergent threats. The system implements analysis algorithms, which will be used to analyze different aspects of the data, and produce evidence that will be aggregated together with a "suspiciousness" score, to be reviewed by a domain expert. The analyst in turn can use his domain knowledge and context information, injected by the platform into the data, to decide what can be considered a threat and what is not.

The analysis algorithms are encapsulated in different detection modules, or as we call them "agents", which are designed with autonomy and specific behavior in mind. New agents can be integrated in a plug-and-play fashion, acting as a black box, where raw data can be passed to them and relevant evidence will be produced after the analysis. Different agents can display vastly different behavior, focusing on different aspects and treating the data in a multitude of ways.

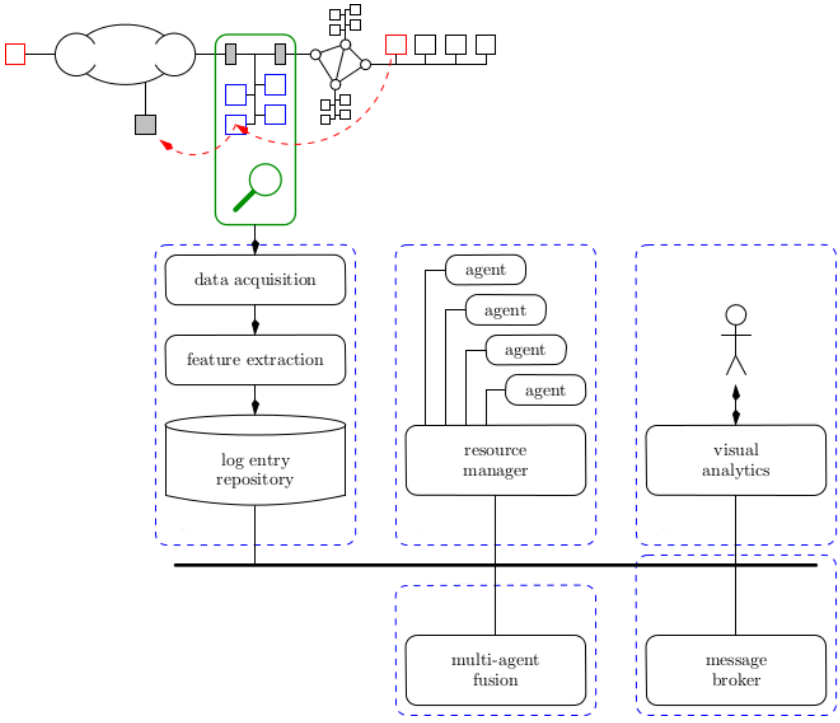

Fig. 1. The MARk Framework overview, representation of the flow of data through the framework

As shown in Fig. 1, the framework collects proxy logs, parses the data, extracting features of interest, and stores them for further analysis. The data is then passed to the various detection agents, which might produce evidences, if abnormal activity has been detected, and the various produced outputs are finally aggregated, as shown in Fig. 2. The agents use fuzzy sets and fuzzy expert system rules to determine the possibility of malicious intent through the assignment to each single event a degree of uncertainty. Using the Ordered Weighted Averaging (OWA) operator, introduced by Yager [22], and the Weighted Ordered Weighted Averaging (WOWA) operator [20], combined with Machine Learning algorithms to augment the multi-criteria decision system [10], the aggregation agents generate an ordered list of rankings. The goal is to push possible false positives lower on the ranked list and to elevate the true positives to the top with the highest score.

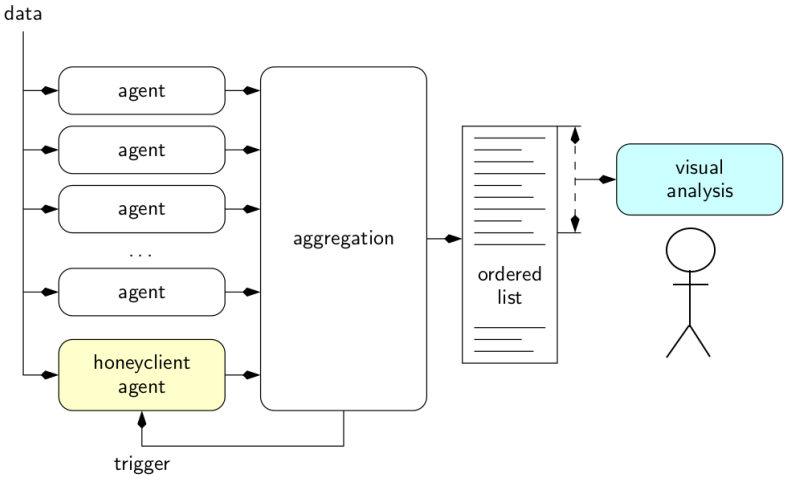

Fig. 2. Agent Aggregation flow, from raw data, through analysis and aggregation, to generation of final order list and visualization

As mentioned previously, each agent is designed to apply a specific analysis algorithm, tailored for a given APT characteristic. Some examples of these algorithms are frequency analysis, distribution of IP and domain names, analysis of time/amount of connections, geo-spacial analysis, etc. As shown in Fig. 3, the agents are interconnected through-out the Activation cascade, based on the concept of a "detection cascade", where the detector agents work independently or together and serve as an initial filter for the data. The modular nature of the agents offers great flexibility in how they are used and linked together inside the activation cascade. To correctly identify complex patterns, it is possible to combine the analysis of multiple agents, through chain activation, which leads

to a more in-depth view of the data and a more sophisticated analysis. A specific agent can be initiated multiple times with different parameters or part of different analysis chains, leading to a finer detection and the capacity to easily tailor the detection framework to look for new and emergent behavioral patterns.

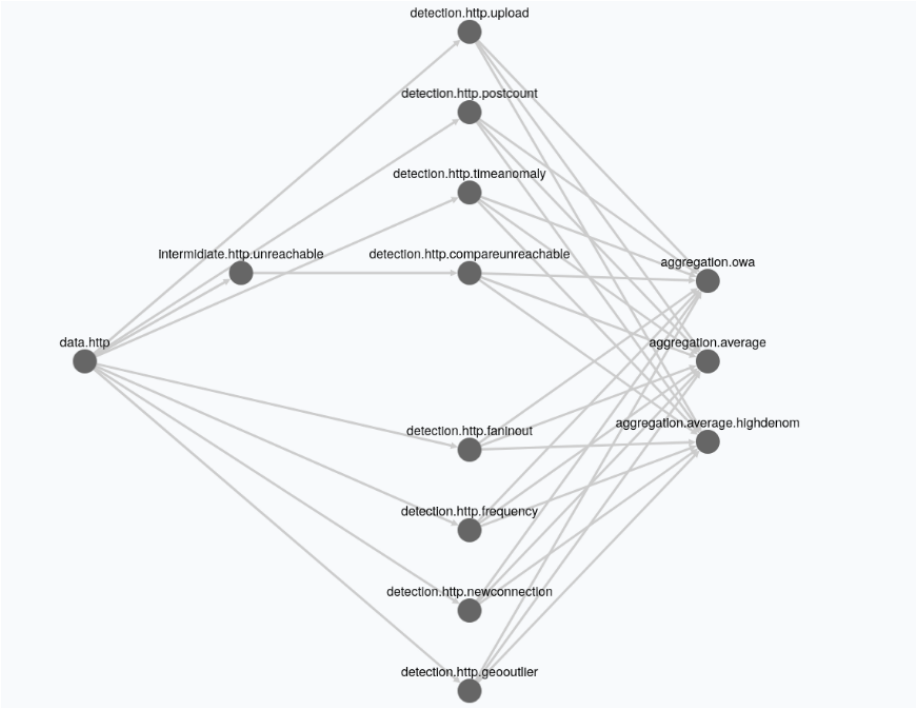

Fig. 3. Detector Agent cascade example, visual representation of the activation cascade of detection agents through injection of Raw Data

A major focal point of the framework is the handling of large amount of data and how best to present the collected information, to better understand and identify any possible abnormal behavior. In recent years, considerable effort has been put into research for better visualization of information [12] and the techniques to use [14]. Using "detection through visualization", the end-user analyst has more tools at his disposal for detecting abnormal behaviour and activity. The MARk framework offers an "all-in-one" package for detection, visualization and analysis that is easy to deploy, extend and use. Examples of the data visualization will be presented in section 4.3.

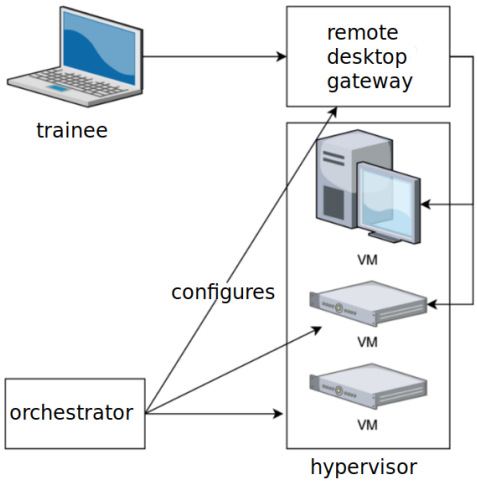

Fig. 4. Basic Cyber Range architecture, consisting of an orchestrator, a hypervisor and a remote desktop gateway

3 Cyber Range

With the rapid rise in quantity and sophistication of cyber attacks, new methods must be used to better prepare ourselves against such threats. Case studies on the use of cyber ranges [23] show the benefits of using them for awareness training, research and education. The possibility of simulating large and complex networks enhances the verisimilitude of the cyber range scenarios and offer greater quality of training and knowledge, which one could gain from them. Two aspects that are of vital importance for us are the capabilities to simulate activity inside a network and the collection and analysis of data. Through attack simulation, the different phases of the cyber kill chain can be modeled and tested in a safe environment and their impact can be modeled. Another type of activity, which is needed, is the user activity simulation, as it makes the simulated environment more realistic. Finally, the capability of the cyber range to collect data from network traffic, memory dumps, tools used and much more, offers a high level of data analysis to determine the exact behavior of an attack, or the performance of a detection tool.

The cyber range used for our research purposes [11] offers great flexibility and scalability in how we want to implement a scenario. As shown in Fig. 4, the cyber range has three major components, which define how a scenario is run:

- The **hypervisor** is responsible for instantiating and running the Virtual Machines (VMs), which will serve as stand-ins for real world machines, be that the attackers, victims or SOC infrastructure.
- The **orchestrator** is the heart of the cyber range, it is responsible for analysing the scenario definition and from there provision the VMs (deploy

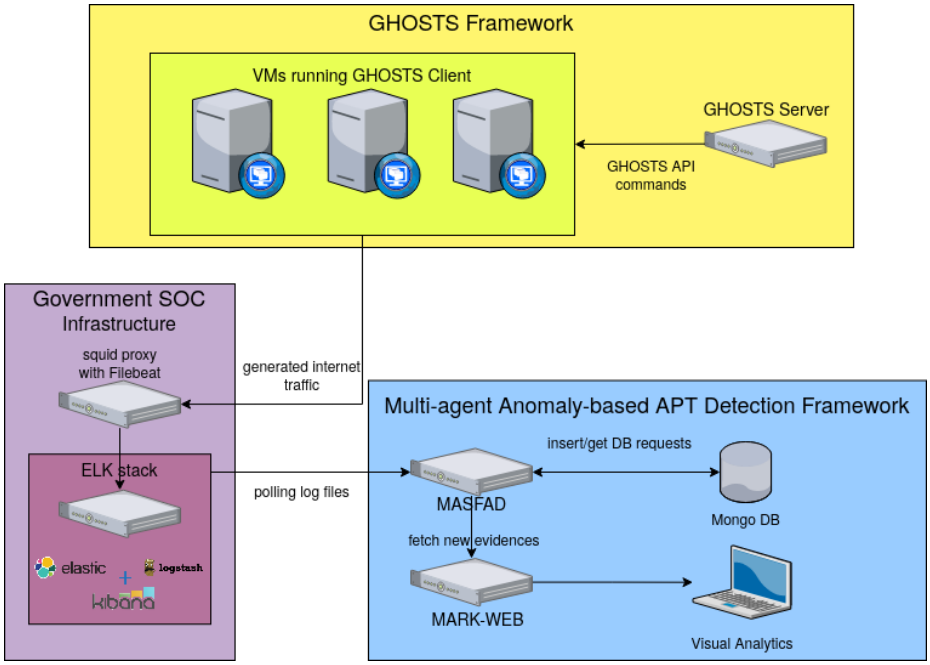

Fig. 5. Deployment Architecture

images, configure VMs, install extra needed software), configure the virtual networks, create and configure any accounts needed for access to the VMs.
- The **scenario** definition takes the form of a json or xml text file, detailing within all the specifications needed for simulation and deployment.
- The **remote desktop gateway** offers the possibility to connect to the cyber range and view the VMs through a web browser.

The graphical representation of our scenario file is shown in Fig. 5. As we can observe, the scenario is composed of three building blocks- the GHOSTS framework, the simulation of a government SOC infrastructure and the MASFAD framework. The configurations of the different machines used in the scenario are as follows:

GHOSTS/ELK/MARk Framework servers

- Operating System: Ubuntu 18.04 server distribution
- vCPUs: 8
- Virtual Memory: 16gb
- Used Programs: Docker

GHOSTS client

- Operating System: Windows 10 Enterprise
- vCPUs: 4

- Virtual Memory: 8gb
- Used Programs: Mozilla Firefox, Google Chrome, Microsoft Edge

The rest of this section goes in more detail on the different components used in the scenario and their purpose.

3.1 GHOSTS framework

To effectively run our scenario, a dataset of log files is needed to simulate traffic from inside the network to outside sources. This is accomplished by utilising the GHOSTS framework [6], a tool designed specifically with the purpose of creating and simulating accurate and highly realistic environment for cyber warfare exercises, by establishing and using behaviorally accurate autonomous non-player characters (NPCs) [21]. The framework uses a centralised server, which has managerial and monitoring functions. The server orchestrates the NPCs through the use of an application programming interface (API) and monitors their functionality. The clients serve as the NPCs, each with a predefined set of realistic behaviors, ranging from console commands to network connections, through the use of web browsers.

3.2 The Elasticsearch Logstash Kibana (ELK) Stack

The management and analysis of data inside the network, be that proxy logs, DNS, or machine log files, is a daunting task, largely because of the sheer volume of it. A solution for this, is the use of log analytics tools in the SOC, solutions which offer the possibility to aggregate, analyze and visualize data from different sources. One such tool is the ELK Stack [5], widely used because of its open source nature, scalability and efficiency. There have been studies on the performance of the ELK Stack [18], which show reasonably high performance and usability. In the presented scenario, the Filebeat module is used for the collection of proxy logs from the network, which are then collected and parsed via Logstash and stored via Elasticsearch. The ELK Stack offers an API, which is used to poll new entries for analysis. A major advantage of using the ELK Stack is the simplicity with which a new detection system can be added to the SOC and its findings stored and visualized.

3.3 The Multi-agent System For APT Detection Framework

In-depth explanation of the framework was given in section 2, here we will explain how our tool has been configured and deployed in the cyber range. The three main components of the framework are the Multi-agent System For APT Detection (MASFAD), the Mongo DB database, used for storing the raw data and generated evidences, and the MARk-Web interface, responsible for monitoring the framework and visualization of the produced evidences. At regular intervals, the MASFAD system polls the ELK Stack for new data entries, if any are available, they are retrieved and stored in the Mongo DB. The resource

manager will act as a triggering mechanism for the detector agents, when new analysable data is available. Any and all produced evidences will be prepared for evidence aggregation and stored in the database by the resource manager.

4 Evaluation

The preliminary evaluation of the scenario deployment will be presented in this section. First we will discuss the set-up used for the evaluation, how the data is generated and what type of malicious APTs were introduced in the scenarios. Afterwards an overview of the methodology and metrics used during the evaluation will be discussed. Finally the results obtained from the various detector and aggregator agents will be shown via screenshots of the output generated by these agents. Each agent generates information, important for any analyst to determine how the agent produced the output it did:

- the triggering label for the agent, denotes what type of data will trigger the analysis by the agent
- interval used by the agent for activation
- the parameters used by the agent during the analysis
- the timestamp of when the ranking of the evidences produced, was generated
- the ranking of evidences produced. Each element in the ranking can be inspected separately, or a *.csv file of the ranking can be generated and downloaded

The aggregation agents presented in the results collect evidences produced by detector agents and, depending on the logic defined in how to compute the aggregation, generate a final score, which designates the "suspiciousness" score of the particular client-server connection. Issues encountered during the deployment of the scenario will also be discussed and, when applicable, possible solutions will be offered.

4.1 Evaluation Set-Up

We evaluate the performance of the MARk framework, as deployed in the cyber range, by running a simple network of 5 automated clients. Each client generates a typical network activity through the use of the GHOSTS framework. This is accomplished through the use of a so-called "timeline" script, written in json, which is responsible for simulating the behavior of a real computer user. The script can be modified to access websites, open specific programs or run command line commands. In regards to simulating network activity, the configuration of the timeline script can be quite in-depth- we can specify the exact timing of the execution, define a flow of commands that need to be run by the GHOSTS client and the result. We have parameterized a set of activities for each of the clients consisting of:

- a simple connection to the "google.com" domain, simulating search activity and clicking links, retrieved from the search.

- a simple periodic connection to a list of domains. A website will be chosen at random and the client will connect to it using its browser.
- a set of one-time connections to suspicious or unreachable domains.
- a set of one-time connections to a domain, followed by commands to download a file from the website and open it (this can be an image, zip-file or other)

Alongside the typical network behavior, simulated by the GHOSTS clients, we are also injecting APTs, modeled on existing attacks. The injected attacks range from basic periodic attacks to high complexity real world APTs such as the Trojan Nap APT [17], the Regin APT [8] and Careto APT [13]. Some examples of possible APTs with the following characteristics:

- connection to the APT server is established by only 2 of the configured clients
- connection to the APT server during an interval with a set frequency
- connection to an unlisted APT server
- connection to the APT server will often return "TCP_MISS", as the server is unreachable
- connection to the APT server happens at random moments and can be considered an outlier, as no other connections happen before or after it

4.2 Evaluation Metrics

To evaluate correctly the performance of the detection framework and the usefulness of the Cyber Range for our research, there are certain criteria which must be attained. The metrics, which we selected to answer our key question, evaluate the performance of the detection framework, while running on the Cyber Range, the ease of deployment, the scalability of the solution, the accessibility and interactions with the tool and the repeatability of deployment. Our goal was to measure these metrics and create a global overview, which afterwards we can score. In the following subsections we will discuss each of these metrics and how they were implemented.

Performance The scenario consists of two types of VMs, which need to be configured and deployed. The Ubuntu Server VMs are responsible for running the simulation infrastructure (GHOSTS/ELK) and to do so, have specific requirements. The VMs are configured during deployment to share the same subnetwork, together with a proxy VM, which serves as the connection to the Internet. The Windows VMs are responsible for running the GHOSTS client code, simulating user activity inside the previously created subnetwork and generating network traffic. From the start, we knew that the Linux servers would demand more power, as they would be responsible for large amounts of computations and storage during the simulation.

Some problems were encountered with the configuration of the ELK stack, as the amount of data generated, as shown in Fig. 6, quickly surmounted the default

	Entries	Size (MB)
Data records	5371048	1355
Evidence	1284895	1196

Fig. 6. Amount of Raw Data generated by the GHOSTS framework and ingested by MARk, alongside amount of evidence produced by the detection framework

configuration. The specific error occurred during execution of specific requests, triggering a circuit breaker fail-safe as to prevent possible memory errors. This was quickly resolved by increasing the heap size used by the ELK stack to 4 gigabytes, as by default it was set to 256 megabytes.

During the execution of the simulation, close eye was kept on the performance of the MARk framework through its integrated web interface. As shown in Fig. 7, the Memory usage and Load on the CPUs during the scenario was in the acceptable limits. It is important also to note that a large number of jobs were running simultaneously, or waiting to be executed, which can slow down the detection. One solution, which was implemented, is limiting the time-frame in which certain agents are going to be triggered. The detector agents' behavior can easily be adapted through changes to their configuration files, configuring them to run at different intervals.

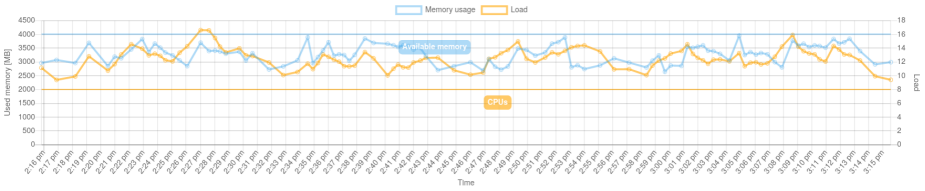

Fig. 7. MARk server performance status as displayed during the execution of the simulation

Another performance issue encountered was a non-significant latency through the web interface, usually when fetching large amounts of data. One solution was to correctly index the database, used for storing the data, and limit the amount of entries fetched per request.

The performance issues encountered were in most cases not directly linked to the Cyber Range itself, but more specifically configuration problems, which needed to be resolved inside the different frameworks. Because of the environment used for the testing and evaluation and the way the scenario is deployed, as described further in 4.2, the different issues could be resolved quickly and the scenario could be restarted with minimal waste of time. The general perfor-

mance of the Cyber Range was exceptional and the performance of the different components of the scenario was quite good after the corrections were made.

Ease of deployment Each of the frameworks used was instantiated on the VMs through the use of Docker [4]. The different components were running from their separate container, with specific ports available for accessing the relevant APIs. This way no major overhead of installation has to be accounted for. If any issues arise, a container can be stopped, adapted and then re-launched with new configuration. To facilitate the initial deployment and configurations of the different elements of each framework, "docker-compose" files were used, which are responsible for the definitions of the connections between the elements and any virtual volumes needed for data storage.

Some issues were encountered with the deployment of the GHOSTS clients, as they are running on Windows VMs and their complex nature does not lend themselves to be easily containerized. The clients can be deployed in variable numbers, though the use of the Cyber Range hypervisor, but they needed manual configuration. A solution is been worked on, for future deployment of the simulation scenario, in the form of orchestration scripts, which can help automate the deployment and configuration of the GHOSTS clients.

For this specific run of the simulation scenario, the ease of deployment was acceptable, taking into account the time spent on deploying the various Windows VMs. We are certain that through the use of our GHOSTS orchestration scripts, we can exceed the current level of ease of deployment.

Scalability For such experiments, it is important to test how the scenario handles a variable quantity of VMs. In our case, this is regarding the number of GHOSTS clients that need to be deployed, as they have an important role in the scenario. We initially ran the scenario with a low amount of clients, as the individual configuration of each Windows machine took a significant amount of time. We aim to test the scalability of the scenario in the future through the use of orchestration scripts, which will help us initialize and manage large amounts of GHOSTS clients in a short amount of time.

It is important to be able to deploy a variable amount of machines to test a complex network topology. The simulation exercise did not score well on scalability, even when the Cyber Range offers us the possibility to deploy a variable number of Windows machines; The problem arises with the extra overhead from configuring each individual client and connecting it to the GHOSTS server. This will be remedied by the integration of GHOSTS orchestration scripts in the near future.

Accessibility and Interactability The cyber range and the machines, which were running the scenario, can be accessed via the web portal. Another solution is to use the virtual private network (VPN), which was set-up for easy access to the cyber range, and use secure shell protocols to connect to each machine.

The results, generated by the different agents, are presented in a clear and informative way, as shown in 4.3. Further in the research and development, more visualization techniques will be used to greatly improve the ergonomics and accessibility of the data, through the use of dashboard layout and astuteness of the data presented.

Repeatability Through the use of the scenario in conjunction with the Docker environment, the machines for testing the performance of the detector framework can be redeployed multiple times with different set-ups. This saves time and effort, when evaluating the capabilities in different environments. The Docker containers can be modified between runs and new ones can be added with ease. A major caveat is the organisation of the Windows VMs, which can be resolved through the use of orchestration scripts for the configuration and launch of the GHOSTS client side of the simulation. Regarding the repeatability of the simulation scenario, the technologies offered by the Cyber Range score quite high and offer a powerful solution for concurrent executions for better evaluation.

4.3 Results

During the analysis, multiple agents discovered the APT server and an aggregation score has been computed for the connection between the malicious server and two clients with IPs "172.20.116.162" and "172.20.116.160". Enough detector agents were triggered and detected abnormal behavior, that the aggregator agent was launched and computed a "suspiciousness" score to the connections.

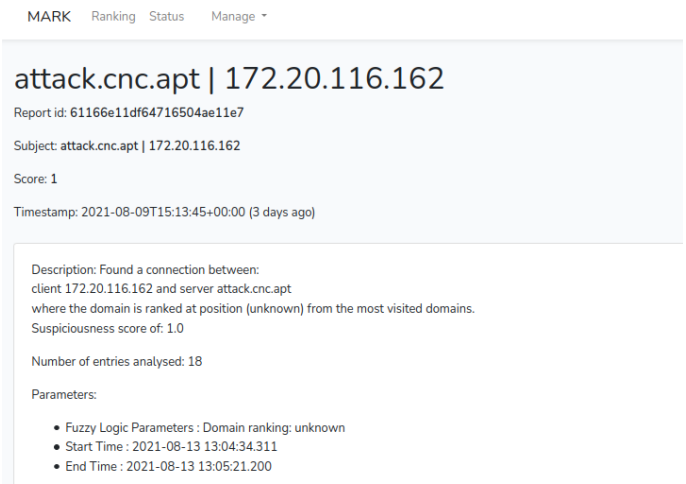

Fig. 8. KnownDomain detector agent output for connection between client 172.20.116.162 and server "attack.cnc.apt"

One of the most simple, but quite useful, detector agents implemented, is the "KnownDomain" detector agent. It uses the Amazon Alexa Top 1 million domains to determine if the domain, connected to by a client, is a popular domain or not. The absence of the domain does not mean categorically that the server is malicious, but it can serve as a starting point for further analysis by triggering another set of agents, as previously shown in Fig. 3. As the domains used by the injected APTs are not part of the list, the detector agent will flag them and produce the corresponding evidence. This will cascade in other more specialised detector agents having a look in the specific domain and producing their own evidence, based on their analysis. The output of the detector agent is shown in Fig. 8.

A more complex example of a detection agent, is the Frequency detector agent. This detector is responsible for creating the frequency spectrum for a given connection between a client and a server, analysing the spectrum and determining if a specific frequency of connections can be observed between the client and the server, as shown in Fig. 9.

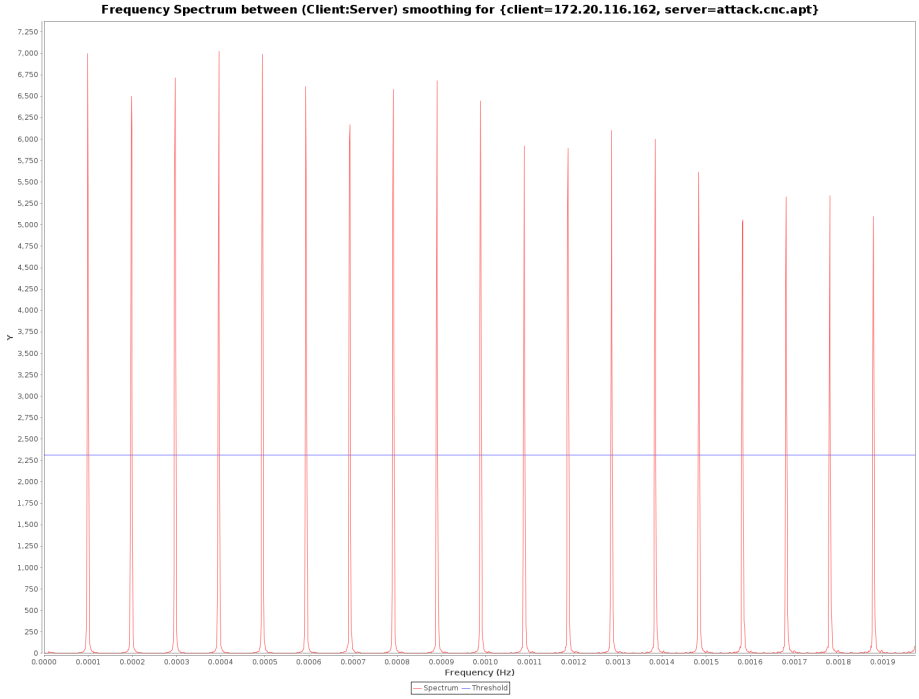

Fig. 9. Frequency spectrum for connections between client "172.20.116.162" and server "attack.cnc.apt"

The ranked list generated by the Frequency detector agent can be viewed in Fig. 10. As shown in the example, the score produced by each agent is a value between 0 and 1, called the "suspiciousness" score. This is accomplished through the use of fuzzy logic and fuzzy membership functions, where the agent will score on a grading scale the results retrieved by the analysis and attribute the final score. Each detection agent, as well as the aggregator agents, use fuzzy logic to create the ordered list, where higher scored evidences will be positioned on top for better visibility. As we can observe in Fig. 10, the two suspicious domains have been placed first and second. This, of course, is not definitive proof of malicious activity, as there are a lot of benign domains, such as those of google and facebook, which by their nature, establish a periodic connection between a client and a server. The high score produced by the Frequency agent, in combination with all other scores produced, will give us a better insight about this specific connection.

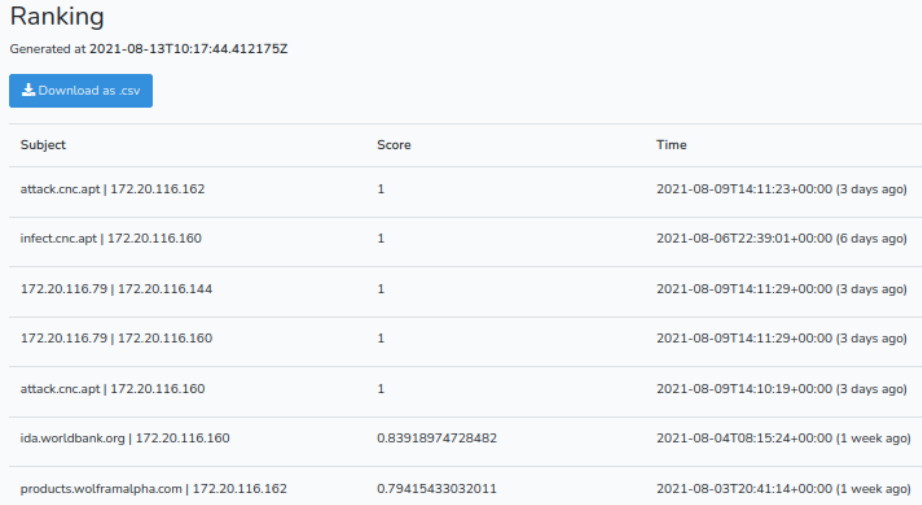

Fig. 10. Example of the Frequency detector agent ranked list output

The evidences produced by the detector agents are aggregated by two Aggregation agents: the Ordered Weighted Average and the Weighted Ordered Weighted Average aggregation agents.

Each evidence aggregated by the OWA aggregation agent will be given a weight, a value that represents how significant it is for the aggregation. For the WOWA aggregator, not only do the evidences receive a weight, but also the detector agents which have produced them. This helps to filter the reliability of the agents and attribute higher weights to the more precise detector agents.

The evidence produced by the aggregation agents is represented in multiple ways, through the use of the D3.js library [3], aiming to aid the domain expert in

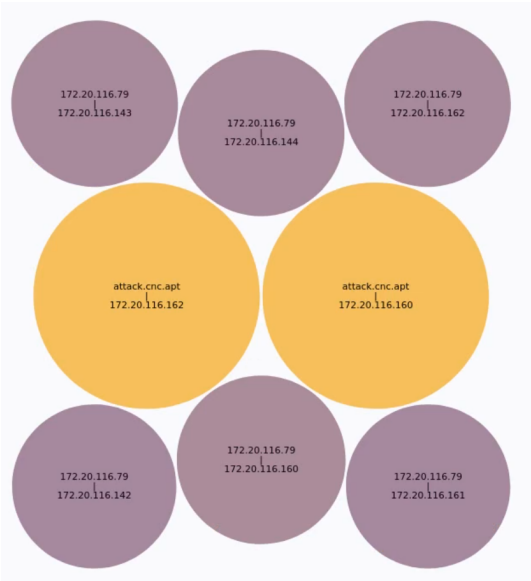

Fig. 11. Bubble Graph Visualization of the OWA aggregator

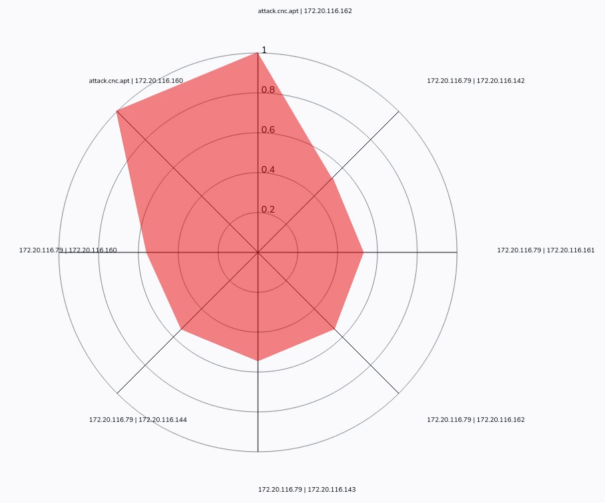

Fig. 12. Radar Chart Visualization of the OWA aggregator

easily investigating and gaining insight into what has happened in the network. Studies in the field of data visualization [14] show that the use of the D3 components offer efficient techniques for detection of APTs, through the analysis of log files.

Examples of the representations are shown in Fig. 11 and Fig. 12, the Bubble Graph and Radar chart respectively. We can observe in those figures, that through the aggregation of the evidences produced by the various detection agents, connections between the two clients ("172.20.116.162", "172.20.116.160"), and the malicious server, were given the highest score.

5 Conclusion

The complexity and severity of recent cyber attacks, shows that newer and more powerful tools are needed to successfully identify them. Further, to be able to develop such tools, there is a high need for an equally powerful environment, such as a cyber range, where these tools can be deployed and their performance tested. Such a platform offers the possibility of attack simulation and, more importantly, security research for detection and prevention. Through the use of the scenario orchestrator, a complex scenario, with interconnected machines in a network, can be successfully created and managed. As we have discussed in section 4.2, deploying the detection framework has shown that the research and development benefit greatly. The use of a dedicated repeatable scenario, which we can adapt and run multiple times with different configurations, offer great flexibility in testing different aspects of the detection framework. The ease of deployment is a major factor in this, as only an initial preparation of specific dedicated VMs is needed, instead of constant re-installation and configuration of multiple machines, when new tests need to be launched. During our evaluation we did notice that the use of Windows VMs for the GHOSTS client was a drawback, as they need to be manually connected to the GHOSTS server. For a low amount of machines, this is not a problem, but if we want to simulate real networks with hundreds of interconnected workstations, the poor scalability of the GHOSTS clients is a hindrance. This can be amended through the use of orchestration scripts, which we are in the process of developing, to automate the connection and configuration of the Windows VMs. Further, we aim to work on implementing more ways to visualize the generated data, greatly increasing the intractability and accessibility of the detection agent inside the cyber range. Lastly, the scenario performed quite well, with the limited machines deployed in the cyber range. When the scalability issues are resolved, it will be important to observe if the performance of the cyber range and the different scenario components behave differently than expected.

The initial results of our scenario were promising, simulating large amounts of network traffic in a short amount of time, while injecting samples of known APTs, and obtaining conclusive detection results from our detection framework. The data generated by the GHOSTS framework did closely resemble average network activity within a small corporate or government network. Down the line, it would be interesting to simulate other types of data, such as file manipulation, command execution and other actions, which can generate important forensic endpoint data.The integration of specialised data source agents shows that the MASFAD framework can easily be integrated in existing SOC solutions, such as

the ELK stack, or work in a stand-alone capacity. A possible avenue for testing would also be to incorporate the MASFAD framework in "Red Team Blue Team" exercises, which we can run on the cyber range. The detection framework can serve as a possible tool for the Blue Team to detect incoming attacks or persistent threats. A cyber range is an ideal platform to host such types of exercises, as the capabilities of the cyber range are highly suited for the needs of cyber resilience and security education, in a low-risk environment.

References

1. Apache log4j security vulnerabilities. https://logging.apache.org/log4j/2.x/security.html, accessed: 2022-02-01
2. Belgian defence ministry network partially down following cyber attack. https://www.brusselstimes.com/belgium/198521/belgian-defence-ministry-network-partially-down-following-cyber-attack, accessed: 2022-01-28
3. D3.js, data-driven documents. https://d3js.org/, accessed: 2022-02-07
4. Docker: Empowering app development for developers. https://www.docker.com/, accessed: 2022-02-07
5. The elk stck. https://www.elastic.co/what-is/elk-stack, accessed: 2022-02-01
6. Ghosts npc automation. https://github.com/cmu-sei/GHOSTS, accessed: 2022-02-01
7. Un computer networks breached by hackers earlier this year. https://www.bloomberg.com/news/articles/2021-09-09/united-nations-computers-breached-by-hackers-earlier-this-year, accessed: 2022-01-28
8. Agency, C..I.S.: Regin malware (2014), https://us-cert.cisa.gov/ncas/alerts/TA14-329A
9. Aslan, Ö.A., Samet, R.: A comprehensive review on malware detection approaches. IEEE Access **8**, 6249–6271 (2020)
10. Croix, A., Debatty, T., Mees, W.: Training a multi-criteria decision system and application to the detection of php webshells. In: 2019 International Conference on Military Communications and Information Systems (ICMCIS). pp. 1–8. IEEE (2019)
11. Debatty, T., Mees, W.: Building a cyber range for training cyberdefense situation awareness. In: 2019 International Conference on Military Communications and Information Systems (ICMCIS). pp. 1–6. IEEE (2019)
12. Erbacher, R.F.: Intrusion behavior detection through visualization. In: SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483). vol. 3, pp. 2507–2513. IEEE (2003)
13. Lab, K.: Unveiling "Careto"-The Masked APT (2014), https://d2538mqrb7brka.cloudfront.net/wp-content/uploads/sites/43/2018/03/20133638/unveilingthemask_v1.0.pdf
14. Lee, J., Jeon, J., Lee, C., Lee, J., Cho, J., Lee, K.: A study on efficient log visualization using d3 component against apt: How to visualize security logs efficiently? In: 2016 International Conference on Platform Technology and Service (PlatCon). pp. 1–6. IEEE (2016)
15. Mees, W., Debatty, T.: Multi-agent system for apt detection. In: 2014 IEEE International Symposium on Software Reliability Engineering Workshops. pp. 401–406. IEEE (2014)

16. Nikolov, G., Debatty, T., Mees, W.: Evaluation of a multi-agent anomaly-based advanced persistent threat detection framework (2020)
17. Parkour, M.: Trojan Nap aka kelihos/hlux - feb. 2013 status update (2013), http://www.deependresearch.org/2013/02/trojan-nap-aka-kelihoshlux-feb-2013.html
18. Son, S.J., Kwon, Y.: Performance of elk stack and commercial system in security log analysis. In: 2017 IEEE 13th Malaysia International Conference on Communications (MICC). pp. 187–190. IEEE (2017)
19. Souri, A., Hosseini, R.: A state-of-the-art survey of malware detection approaches using data mining techniques. Human-centric Computing and Information Sciences **8**(1), 1–22 (2018)
20. Torra, V.: The wowa operator: a review. In: Recent developments in the ordered weighted averaging operators: theory and practice, pp. 17–28. Springer (2011)
21. Updyke, D.D., Dobson, G.B., Podnar, T.G., Osterritter, L.J., Earl, B.L., Cerini, A.D.: Ghosts in the machine: A framework for cyber-warfare exercise npc simulation. Tech. rep., CARNEGIE-MELLON UNIV PITTSBURGH PA (2018)
22. Yager, R.R.: On ordered weighted averaging aggregation operators in multicriteria decisionmaking. IEEE Transactions on systems, Man, and Cybernetics **18**(1), 183–190 (1988)
23. Yamin, M.M., Katt, B., Gkioulos, V.: Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. Computers & Security **88**, 101636 (2020)