iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VII **Month of publication:** July 2021

DOI: https://doi.org/10.22214/ijraset.2021.36539

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

Distributed-Decentralized Intelligent Agents for Offensive Cyber Security: An Introduction to Offensive Cyber Threat Intelligence

Nayan Rande[1], Kanaad Rampurkar[2], Nikhil Rokade[3]
[1, 2, 3]*AISSSMS IOIT & OffSecIntel*

Abstract: ***In order to begin to design a large Offensive Cyber-Threat-Intelligence, we need a distributed-decentralised Intelligent Software framework, which can scale on demand and run with flexibility while providing a room for further improvement both on architectural level as well as strategical level for planning advanced attacks methodologies, will help us conduct secure transaction maintaining CIANA. In this paper, we try to present some of our investigations on Agent-based Modelling of Cyber-Space and Simulation of Cyber-Warfare over distributed system methodology in contributing to these designs. Using this as motivation we try to build system architecture for effective open intelligence for effective offensive cyber threat intelligence (CTI).***

I. INTRODUCTION

Problems in cyber security domain are too complex to handle & approach in their natural form [1]. Due to this highly complex behaviour, it becomes practically impossible to define all of the required parameters to solve the problems at large by defining the attack vectors for various attacks. But the hide & seek approach in game theory [2] can help us build this attack vectors at large and automatically build the parameters required to successfully attack a systems. An Abstract Game of Cyber Security [1] has shown various basic strategies that can be built around the hide and seek approach to successfully search and backtrack attacker (while being a Victim to attacker) or search and exploit a vulnerable victim (while being an attacker).

This results helped us build ground for an AI And ML based system over Distributed Systems which will learn about various tactics to attack a system and create a series of methodologies to successfully "stop an ongoing attack by an attacker and secure a system", as well as attack an attacker to breach his system and use an offensive approach to obfuscate him based upon the basic analogy of Hide and Seek [1] approach on its own, and planning of attack strategies developed by continuous training of the intelligence.

There exists lots of IDS and IPS, and they have evolved a lot in last two decades with more and more exploratory investigations, and research. Cyber Threat intelligence has evolved to an extent where we can detect intrusions effectively with new emerging Intelligence models, but intelligent mitigation and exploit development is still understudied.

The Goal of our research is to present with an architecture and framework design necessary for the development and integration of large Agent based Intelligence scenarios to achieve an Open Offensive Cyber Intelligence. This paper presents the introduction to this architecture with an example of modelling and simulation of DDoS using Agent-based modelling [3] over Distributed Infrastructure.

The rest of the paper is structured as follows. Section 2 gives the background context of and need of this framework design. In Section 3, we try to outline the architectural design for implementing a distributed system framework for supporting integration of open cyber intelligence research. Section 4 presents the methodologies while testing this design implementation, and how this can be further extended to future development of Open-Source Offensive Cyber Threat Intelligence. And finally we conclude our findings and under- standings of this domain.

II. BACKGROUND

Over last two decades Game Theory and AI had expanded in various domains such as medical to Astronomical Sciences, but a very small amount of research has been conducted in Cyber Security domain. We do not claim that intelligent In- trusion Detection Systems (IDS), Intrusion Prevention System (IPS) and some other forms of AI such as deep fake image detection and creation doesn't exist, but eventually, apart from a solid grounds for automated exfiltration of attack traffic and normal traffic, a very less research has been done on automatically exploiting systems at large using AI.

Apart from this, the cryptographic techniques those were impossible to decipher 2-3 decades ago can be cracked in minutes with the computational power at hands today [4], [5]. What we use to see as a very secure algorithms such as AES or RSA are considered breakable today with the advancements in HPC and "Quantum Computing" [6], [7]. And moreover, this same analogy can be used to see the future of Cyber Security. What we see as very secure algorithm, will find itself weak in securing the integrity, confidentiality, non-repudiation, and availability in upcoming 10-20 years. The way complex systems evolves, we also need to evolve the techniques to design algorithms. With the methodology and architecture we present here, we are trying to evolve attack vectors using the agent based modelling [3], reinforcement learning, and deep learning approaches.

A. Hide-and-Seek

In Game Theory, various strategies are designed to be im- plemented for the game of Hide-and-Seek. Various approaches are given to design the strategies for Hiding and Seeking are presented, and continuous efforts are put to optimise these strategies. These strategies are previously been used and tested in cybersecurity [1] based upon the exploit strategy presented in this methodology. We are using this as a starting point for our research work, which again, we understand that, can be extended further in a limitless ways.

B. Cyber Security Intelligence

Cyber Security Intelligence, much commonly known as Cy- ber Threat Intelligence [8], [9], [10], [11] is widely researched area among Cyber Security Researchers which primarily in- clude IDS and IPS.

Machine learning in cyberspace has the potential for both attacking and defending. If trained and devised well, it can breakthrough many defence protocols. Currently machine learning in cybersecurity is deployed to provide robust pro- tection against threats in cyberspace. It can learn, adapt and improvise with time in order to minimize the impacts of at- tacks. Supervised and unsupervised learning have the potential for developing IDS, Malware detection system, understanding and preventing cyber-physical attacks and protecting privacy. Reinforcement learning (RL) is a learning model that is used in machine learning (ML), where machines learn through experience, and gain skills without human intervention. In this technique an agent interacts with an environment by taking actions in order to maximize the total rewards.

Developing an entire system without having prior data on the processes becomes pretty costly to develop. To overcome this many industries tend to adapt simulations to obtain data as per their requirements and trained the model specifically. Agents based simulations have been extensively researched in the last two decades. Integrating reinforcement learning with game theory the performance can be fined tuned for optimization. The system can be trained on adversarial and real-time data where-in it will exploit its own weakness and improvise. This methodology is essential in cybersecurity where cyber - attacks are increasingly sophisticated, rapid and ubiquitous.

Traditionally, cybersecurity methods such as firewalls, an- tivirus software or intrusion detection are normally passive unilateral and lagging behind the dynamic attack. Cyberspace involves various components on which it works, Thus a reliable security system needs to understand the interaction of these components specifically the security policies applied to these components which can have a impact on the decisions taken by other components.

C. Agents-Based Modelling and Simulation of Cyber-Warfare

In agents based modelling, two or more teams of antag- onistic agents covertly work for planning and competing to realise joint/common intensions. Till date, many mathematical models have been proposed to realise these strategies. What we intend to do is to provide a common ground to integrate all these strategies [12] and modern intelligence techniques such as Deep Reinforcement Learning (DRL) [13], for added flexibility to incorporate emerging mathematical strategies, hence providing effective offensive Cyber Threat Intelligence Give Reference to Agent Based modelling in recent research. *Kotenko* [13-14] [12], [3], [14], has given a methodology to design and implement an Agent-based-simulation between malefactors. In initial assessment of the problem of cyber security, subject domain ontology should be created. We will use this to categorise our exploit-DB and to create abstraction between different attack vectors. Then, we need to deter- mine the agents' team structure, which can be performed autonomously with the use of sequential decision making with RL or DL. In the third step, agents' interaction-and coordination mechanisms should be determined which also includes roles' privileges and scenarios for exchange of role. Election algorithms in distributed system can be extended and used in conjunction to DL as well as automation scripts system configuration. Specification of agents' plans as a hierarchy of stochastic formal grammars essentially provides us with a certain expected predictive knowledge of the domain, but informal grammars present more general observations. Roles are then assigned to agents and their process-daemons. Plans are assigned along with necessary parameters.

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

III. MODULE DESIGNING

The core architecture is divided into three project level module, which can then be integrated to single entity to achieve final results and testing. For Isolated environment we use two stages of isolation, Sandboxing/Virtualisation & Con- tainerisation of processes. Each individual module package is responsible for performing specific set of task functions with function specific script, classes, interfaces, etc. However, while deploying the processes the module components can be reused/extended independent of the package they are part of.

A. Cyber-Warfare

Cyber-Warfare in itself is a combination of multi-module system. It can be broadly classified as: Simulator 2)Attacker 3)Victim
All of these again is combination of multiple toolset, running in coordination with agent based modelling.
Isolated Environment provide us with the ability to min- imize the risk of Attacker or Defenders (Hider/Seeker) ac- cidentally attacking and interfering with real world systems, applications or machines, eventually stopping us from do- ing some illegal stuff unknowingly. This also provide us a controlled environment to conduct all testing ethically and stop us from accidentally break things up. In agent based modelling, we start with minimum of two antagonistic teams who covertly plan against each other and act on them to achieve the common goal. Here we start by two Teams each representing a collection/set of system nodes:

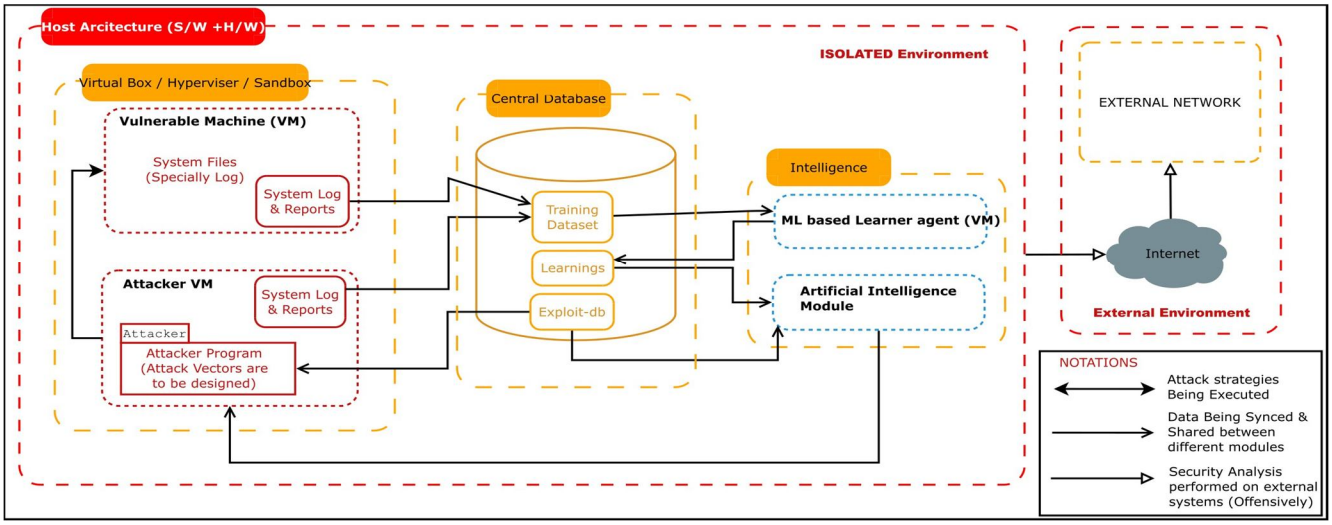


Fig. 1. System Architecture

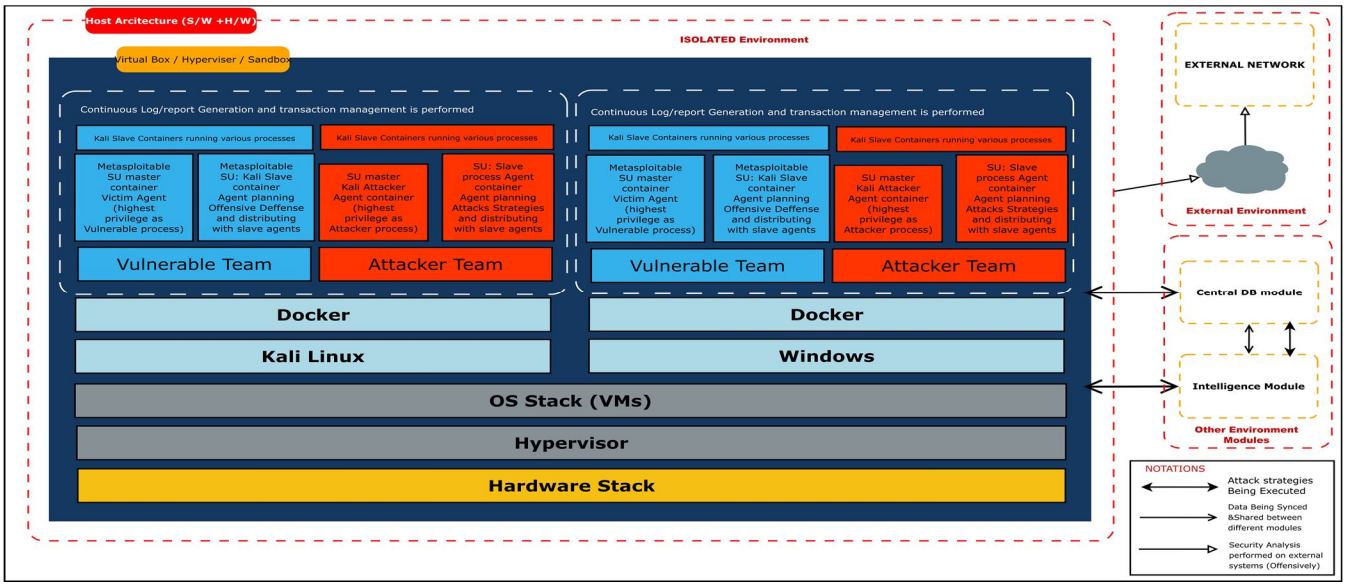


Fig. 2. CyberWarfare module System Architecture

1) *Attacker* – A team of malefactors' agents, which gen- erates distributed coordinated attacks. This include an underlying system as an Attacker VM.
2) *Vicitm* – A set of agents, deliberately created as vulner-able nodes executing various defence strategies running threads for intrusion detection, protection, adversary intensions, actions predictions, and incident response. Information fusion is partially done in this module and rest in Intelligence module node.

The simulator that is being used here, for simulating the game of Cyber Security [1], is combination of many tools. Actual open-source AI, and ML based simulators [15] are being used as needed, with the help of Virtualisation and Containerisation concepts using Hypervisor, Kubernetes, and Dockers.

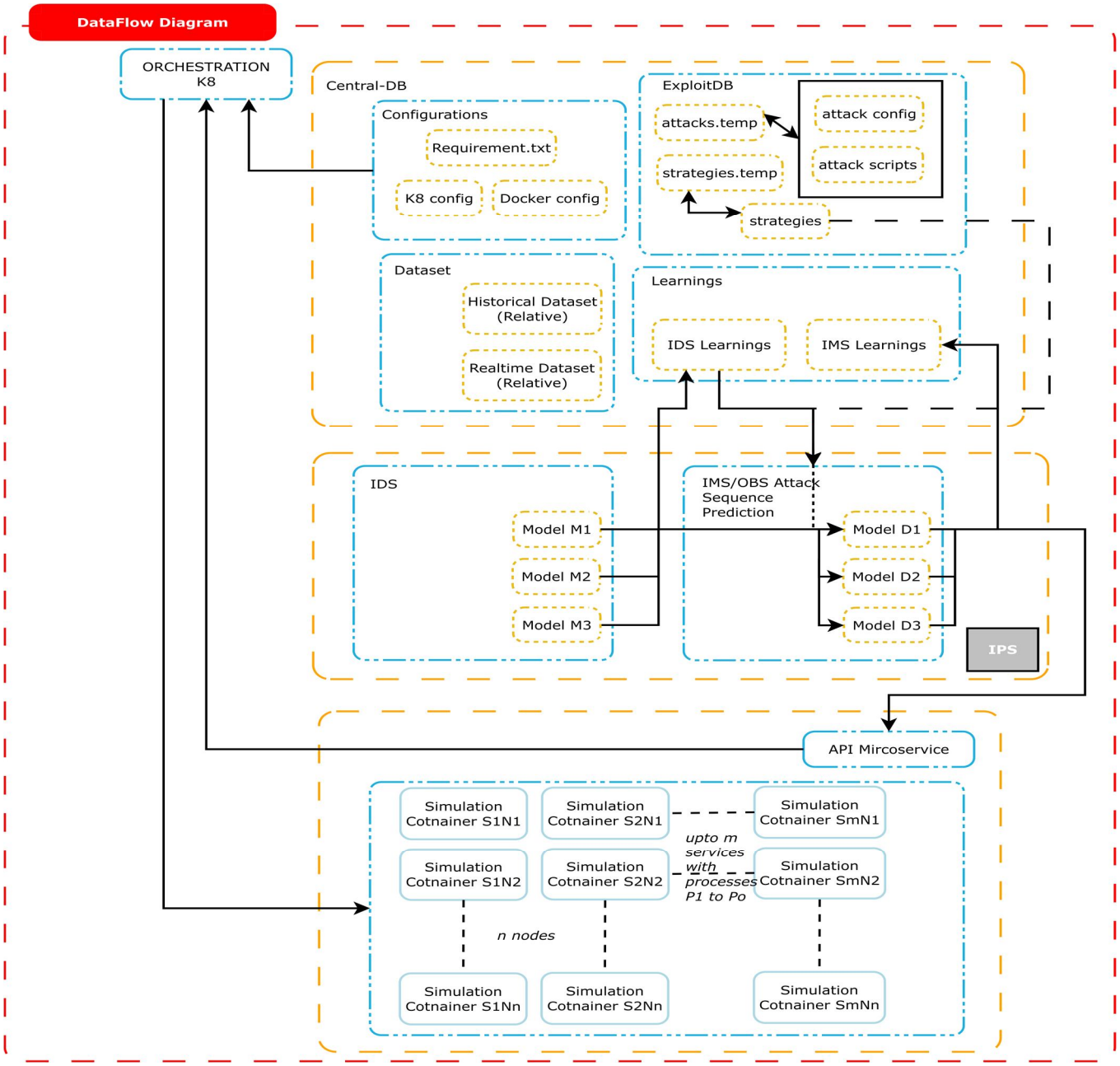


Fig. 3. System Data Flow Architecture

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

B. CentralDB

This is where all the data, code, repos, test results, reports, datasets, and learning from the Intelligence are stored. Central-DB module, we have got three sub-modules.

For Training Dataset module, reports generated from various modules are stored and feed as input. Mainly, reports such as scan reports and test reports, log files from Attacker and Administrator are given as I/P and stored for further process- ing. Learnings module contain the useful learning generated from ML module using the training dataset. This learning are various parameters and attack vectors that can be used to take decision regarding the attack mode. Exploit-DB contains two packages:

1) Strategies
2) Attacks

The strategies and Attack will be updated using the new explorations done by the AI as well as the update cycle of overall project. The new strategies are stored temporarily in *Strategies.temp* while in case of attacks, *Attacks.temp* is used.

Once the successful testing of the strategies and attacks is complete, they are moved to garbage or as to permanent storage depending upon the success of the tests.

Then we have CentralRepo module which contains all the source code for various modules and required. Various branches can be tested and deployed in parallel, and forks can also be used to test various functionalities designed as per new research.

Configurations module that contains all necessary config- urations, requirements for installing and deploying agents as needed for training, testing, or real life scenario. DB(Database) contains all the necessary files required for additional systems features and functioning, including Backups. All transactions and maintenance scripts are stored in here, which can be then be accessed as per system /process requirement.

C. Central Intelligence

We classify cyber threat Intelligence in three broad cat- egories – IDS (Intrusion detection system), IPS (Intrusion Prevention System), and IMS&OBS (Intrusion Mitigation & Offensive Backtracking System). This broad classification helps abstraction of different functionalities at modular level to achieve different functionalities. Central Intelligence module is designed and developed in such a way that it can be can be extended and implemented as a component/element of other modules to perform independent tasks. This provide other modules services to run desired child processes with elevated privileges. The input data to the Intelligence module is broadly classified as 'real time generated' & 'historical' dataset. For our convenience, we define these terms relative to each other. All sub-modules, at the package level, are divided into top level classification of various algorithms.

1) *IDS (Intrusion Detection System):* We trained our IDS initially with Static Historical Dataset with a DL based model. This helps us visualise the attack vectors and there obvious target notions. Later, we used DRL based 'actor critic' models to train IDS on real time generated data. Different models are used to achieve performance, efficiency in function specific requirement. As the data being generated at various stages in time after deploying a specific module is relative, IDS is trained on same data more than once, every time with different training model. The results of all models are ensemble to get optimum results. This helps us balance load based on required performance, and accuracy ensuring efficiency and reliability of system.
2) *IM&OBS (Intrusion Mitigation & Offensive Backtrack- ing System):* For a complex problem of cyber security value- based methods, more specifically DQN, shows very high time complexity, resulting inefficiency or even impracticality [13]. On the other hand, policy gradient methods however try to solve this within feasible time frame, tend to show large fluc- tuations in gradient estimations for probability distribution. It has been shown that actor-critic methods, such as DDPG[16], D4PG[17], A3C[18], UNREAL[19] help overcome these is- sues by Combining value-based and policy-gradient methods [13]. Model-based actor-critic methods can then be integrated and used in conjunction to the model-free methods to improve sample efficiency [20], [13].

IV. METHODOLOGY

While implementing this framework, we started with De- vOps concepts to shorten up development cycle. This will help establish secure code management while having reliable build cycle so that we will be able to provide scalability and flexibility as per demand.

The module implementation of the framework architecture serves specific features and functionalities, which will help us simulate attack and defence strategies, and tend to be tested and deployed on runtime due to design of this system frame- work. Few of the build and implementation stages include:

A. Strategies for Game

While using game theory in conjunction with DRL, we find an extensive mathematical research foundation as a basis to implement grounds for Cyber Security. In our design approach we primarily focusses on two different set of theories as mathematical foundation, one Hide-and-Seek [1], [21]; and other one will be, *Nash Equilibrium* [22] and *Stackelberg Equilibrium* and in general Equilibriums [23]. In the game of *hide-and-seek*, we consider two agents as Attacker and Victim/Administrator with which can have either of two roles as Hider, H (a defensive approach) or Seeker, S (an Offensive approach), provided an specific game instance. The Hider can operate with two strategies as Stochastic or Bias; alternatively, Seeker, S can operate with two strategies as Random or Exploit [1]. This provides with a total of 16 $(2^n \times 2^n$, *where* $n = 2)$ modes of operation. This mathematical basis can be extended to more complex problems with the inclusion of DRL. Nash and Stackelberg Equilibrium provides a basis for optimisation in winning chances in multiplayer game [24]. We use this theory as basis for our DRL based policies for decision making.

B. Simulation

This phase is the most important phase as it helps us generate more Data for studying system behaviour and further research. This phase is always having dependency over other stages to function and operate properly. It pulls code and necessary scripts from CentralDB module and other remote locations, on receiving request from Central Intelligence, and helps execute the simulation in fully isolated, and partially isolated environments, as required by the developer.

C. Decision Making

Decision making phase is mostly achieved by Central Intelligence and partially by strategies that are in execution phase (on runtime), as well as the administrator who may execute critical system call such as terminate all process or halt, or execute some other process with specified parameters. Various parameters are used to define system state, which helps in configuring system, and manging process trees. These parameters also helps in analysing the system behaviour, and decision making based upon that behaviour. While in action, continuous data generation is maintained to ensure the real time analysis of complete system behaviour.

D. Package Management

In order to install and manage process dependencies, man- age agents, and containers, there exists a need of package management stage which will eventually perform this func- tionalities on behalf of administrator, automatically, and some- what partially, manually. (Include dependency management and algorithms)

E. Communication and Transactions

This deals with all internal as well as external commu- nication and transactions required to establish a secure data exchange. A separate transaction management system for managing various files such as logs, reports, pcaps, etc., is im- plemented with ELK stack which helps building a centralised log management and transaction monitoring. The communi- cation protocols and related scripts, defining rules for various privileged and unprivileged communications over the network is also the part of this stage. Apart from above mentioned tasks, this stage is also responsible to keep transactions in sync, while ensuring the real-time data availability to the critical modules.

F. Code Improvement

Every time changes to code are made, and need to be tested, it can be tested using container based process isolation. We tried to speed up this process using DevOps, by using various open-source frameworks. This include, but not imited to, Git (also git based systems such as GitHub), Jenkins, Docker, Kubernetes, ELK Stack, Ansible, etc.

V. CONCLUSION & FUTURE WORK DIRECTIONS

Intelligence as whole has evolved shown an exponential growth in recent years. IDS and IPS systems and other security measures are deployed and implemented at various points to ensure the Cyber Security of cyber specimen. More and more DRL based models had been researched and had emerged as one of the most successful methods of designing and developing close to human or superhuman AI agents. Human malefactors tend to evolve their attack vectors and strategies continuously. Lack of knowledge to this specific attack vector and strategy type lead to the disastrous zero day attack resulting in high time value to safely mitigate the attack and minimising the ramifications of such an attack, e.g. Solarwinds attack in which hackers had first accessed Solarwinds on September 4, 2019; which was undetected for months [25], [26].

Zero day attack causes a lot of damage to the entity owning the system node and other entities depending on the services on the node directly as well as indirectly. Adversarial Intel- ligence such as IDS, IPS using RL, ML can help detect and prevent large and sophisticated attack, but IDS and IPS alone cannot secure a system. Mitigating a specific vulnerability type by exploiting it in advance still remains with an insignificant amount of research. The systems can be offensively tested for any kind of vulnerability, hence securing a system and network along with IDS, and IPS in place. Sequential Decision making in IDS, IPS, and IMSOBS can address complexity and dynamics of the problem of Cybersecurity at large [13], whilst there is still a gap in automatic code generation for cyber security problems as whole [27], [28], [29].

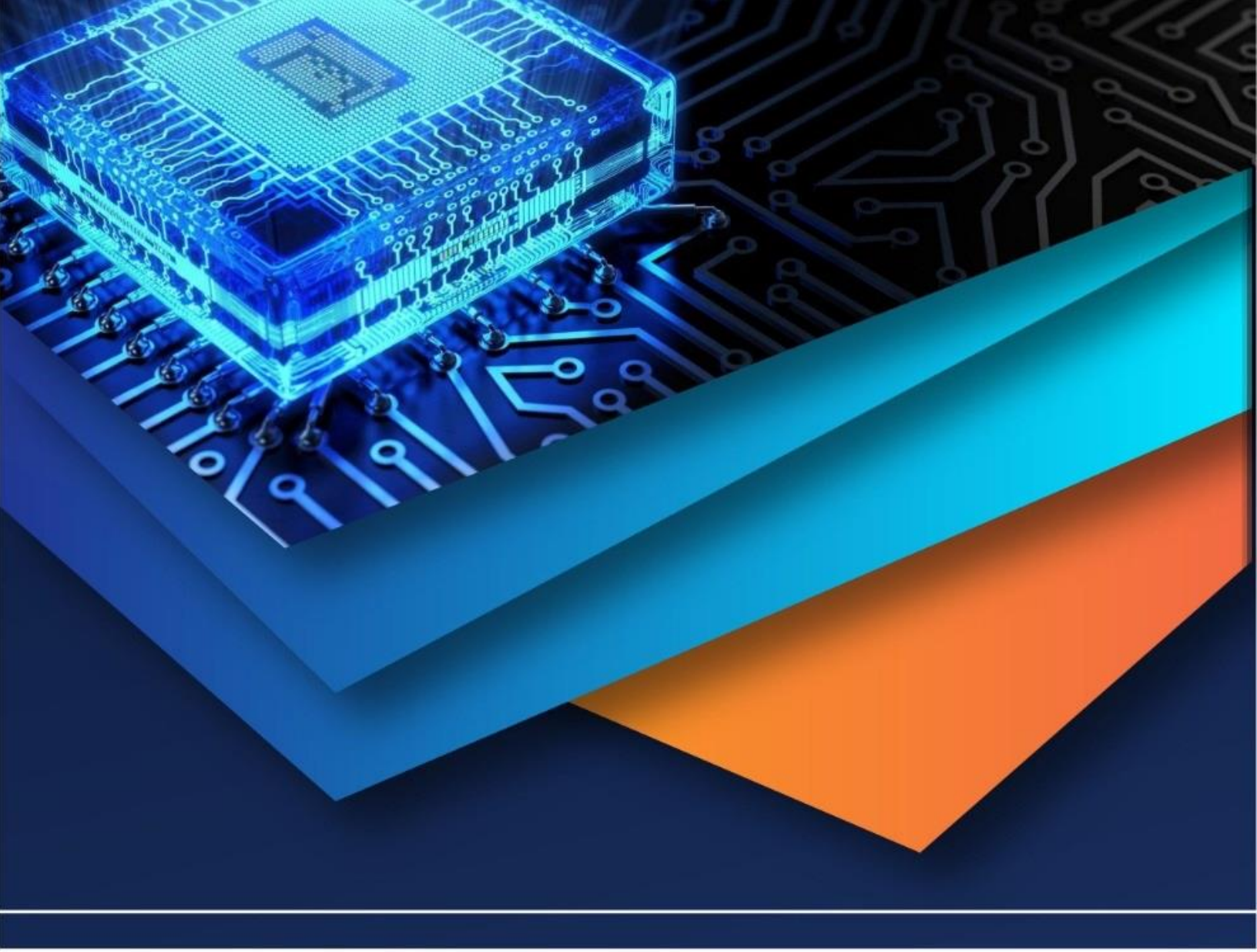
We identified the need of architecture needed for an Open Offensive Cyber Threat Intelligence. In this paper, we present a formal paradigm required for modelling and developing such a system. This paper presents us with an architectural overview of the framework that we will be further using to evolve Open Offensive Cyber Threat Intelligence. We understand that it requires further revision to implement this system securely by following security practices. Security audits should be per- formed to ensure the overall framework security. The system that we designed is only tested for DoS/DDoS based attack simulation, but can easily be scaled and extended to further attack vectors just by adding new strategies, and configurations for simulations.

The further development of this system includes transform- ing it into Open Intelligence, and incorporate various mathe- matical models researched so far in Agents Based Modelling by enlarging the capabilities of this system beyond DoS/DDoS.

REFERENCES

[1] M. Chapman, G. Tyson, P. McBurney, M. Luck, and S. Parsons, "Playing hide-and-seek: an abstract game for cyber security," in Proceedings of the 1st International Workshop on Agents and CyberSecurity, pp. 1–8, 2014.
[2] N. F. Haq, A. R. Onik, M. A. K. Hridoy, M. Rafni, F. M. Shah, and D. M. Farid, "Application of machine learning approaches in intrusion detection system: a survey," IJARAI-International Journal of Advanced Research in Artificial Intelligence, vol. 4, no. 3, pp. 9–18, 2015.
[3] I. Kotenko, "Agent-based modeling and simulation of cyber-warfare between malefactors and security agents in internet," in 19th European Simulation Multiconference "Simulation in wider Europe, 2005.
[4] D. E. R. Denning, Cryptography and data security, vol. 112. Addison- Wesley Reading, 1982.
[5] P. Isasi and J. C. Hernandez, "Introduction to the applications of evolutionary computation in computer security and cryptography," 2004.
[6] J.-P. Aumasson, "The impact of quantum computing on cryptography," Computer Fraud & Security, vol. 2017, no. 6, pp. 8–11, 2017.
[7] M. Roetteler and K. M. Svore, "Quantum computing: Codebreaking and beyond," IEEE Security & Privacy, vol. 16, no. 5, pp. 22–36, 2018.
[8] V. Mavroeidis and S. Bromander, "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," in 2017 European Intelligence and Security Informatics Conference (EISIC), pp. 91–98, IEEE, 2017.
[9] M. S. Abu, S. R. Selamat, A. Ariffin, and R. Yusof, "Cyber threat intelligence–issue and challenges," Indonesian Journal of Electrical Engineering and Computer Science, vol. 10, no. 1, pp. 371–379, 2018.
[10] A. Dehghantanha, M. Conti, T. Dargahi, et al., Cyber threat intelligence. Springer, 2018.
[11] J. Robertson, A. Diab, E. Marin, E. Nunes, V. Paliath, J. Shakarian, and P. Shakarian, Darkweb cyber threat intelligence mining. Cambridge University Press, 2017.
[12] V. I. Gorodetsky, I. V. Kotenko, and J. B. Michael, "Multi-agent model- ing and simulation of distributed denial of service attacks on computer networks," in Proceedings of the Third International Conference on Navy and Shipbuilding Nowadays, St. Petersburg, Russia, 2003.
[13] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," arXiv preprint arXiv:1906.05799, 2019.
[14] I. Kotenko, "Teamwork of hackers-agents: Modeling and simulation of coordinated distributed attacks on computer networks," in International Central and Eastern European Conference on Multi-Agent Systems, pp. 464–474, Springer, 2003.
[15] C.-V. Pal, F. Leon, M. Paprzycki, and M. Ganzha, "A review of platforms for the development of agent systems," arXiv preprint arXiv:2007.08961, 2020.
[16] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971, 2015.
[17] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. Tb, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," arXiv preprint arXiv:1804.08617, 2018.
[18] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep rein- forcement learning," in International conference on machine learning, pp. 1928–1937, PMLR, 2016.
[19] M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Sil- ver, and K. Kavukcuoglu, "Reinforcement learning with unsupervised auxiliary tasks," arXiv preprint arXiv:1611.05397, 2016.
[20] A. Dargazany, "Model-based actor-critic: Gan+ drl (actor-critic)=¿ agi," arXiv preprint arXiv:2004.04574, 2020.
[21] A. Tandon and K. Karlapalem, "Medusa: Towards simulating a multi- agent hide-and-seek game.," in IJCAI, pp. 5871–5873, 2018.
[22] C. A. Holt and A. E. Roth, "The nash equilibrium: A perspective," Proceedings of the National Academy of Sciences, vol. 101, no. 12, pp. 3999–4002, 2004.
[23] M. Br ückner and T. Scheffer, "Stackelberg games for adversarial predic- tion problems," in Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 547–555, 2011.
[24] M. Li, J. Qin, N. M. Freris, and D. W. Ho, "Multiplayer stackelberg-nash game for nonlinear system via value iteration-based integral reinforce- ment learning," IEEE Transactions on Neural Networks and Learning Systems, 2020.
[25] P. Datta, "Hannibal at the gates: Cyberwarfare & the solarwinds sunburst hack," Journal of Information Technology Teaching Cases, p. 2043886921993126, 2021.

[26] M. Willett, "Lessons of the solarwinds hack," Survival, vol. 63, no. 2, pp. 7–26, 2021.
[27] F. J. Budinsky, M. A. Finnie, J. M. Vlissides, and P. S. Yu, "Automatic code generation from design patterns," IBM systems Journal, vol. 35, no. 2, pp. 151–171, 1996.
[28] M. M. R. Mozumdar, F. Gregoretti, L. Lavagno, L. Vanzago, and S. Olivieri, "A framework for modeling, simulation and automatic code generation of sensor network application," in 2008 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp. 515–522, IEEE, 2008.
[29] D. M éry and N. K. Singh, "Automatic code generation from event-b models," in Proceedings of the second symposium on information and communication technology, pp. 179–188, 2011.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY