2021 13th International Conference on Cyber Conflict
Going Viral
T. Jančárková, L. Lindström, G. Visky, P. Zotz (Eds.)
2021 © NATO CCDCOE Publications, Tallinn

Towards an AI-powered Player in Cyber Defence Exercises

Roland Meier
Department of Information Technology
and Electrical Engineering
ETH Zürich
Zürich, Switzerland
meierrol@ethz.ch

Artūrs Lavrenovs
NATO CCDCOE
Tallinn, Estonia
arturs.lavrenovs@ccdcoe.org

Kimmo Heinäaro
NATO CCDCOE
Tallinn, Estonia
kimmo.heinaaro@mil.fi

Luca Gambazzi
Science and Technology
armasuisse
Thun, Switzerland
luca.gambazzi@armasuisse.ch

Vincent Lenders
Science and Technology
armasuisse
Thun, Switzerland
vincent.lenders@armasuisse.ch

Abstract: Cyber attacks are becoming increasingly frequent, sophisticated, and stealthy. This makes it harder for cyber defence teams to keep up, forcing them to automate their defence capabilities in order to improve their reactivity and efficiency. Therefore, we propose a fully automated cyber defence framework that no longer needs support from humans to detect and mitigate attacks within a complex infrastructure. We design our framework based on a real-world case – Locked Shields – the world's largest cyber defence exercise. In this exercise, teams have to defend their networked infrastructure against attacks, while maintaining operational services for their users. Our framework architecture connects various cyber sensors with network, device, application, and user actuators through an artificial intelligence (AI)-powered automated team in order to dynamically secure the cyber environment. To the best of our knowledge, our framework is the first attempt towards a fully automated cyber defence team that aims at protecting complex environments from sophisticated attacks.

Keywords: *artificial intelligence, automation, Locked Shields, cyber defence, security*

1. INTRODUCTION

Attackers and defenders of cyber systems often engage in an arms race on many levels: attacks become more sophisticated, happen at a faster pace, use greater stealth, and show less evidence. Defenders often lag behind attackers because of the underlying asymmetry between them needing to defend a heterogeneous infrastructure against all possible attack vectors and the attackers only needing to find a single or limited number of vulnerabilities to exploit.

It is widely accepted that a human workforce alone cannot keep up with the workload in a typical cyber environment. As a consequence, many tools have been developed to support human defenders: From pattern detection tools [1], [2] to sophisticated anomaly detection using machine learning [3]–[5]. However, while these tools significantly reduce the effort required by humans to detect anomalies, they do not automate the entire defence process and still require significant human labour to mitigate the impact of attacks and keep services running.

In this paper, we develop a novel framework for fully automated cyber defence. Our framework is designed for Locked Shields (LS) [6], [7], the world's largest live-fire cyber defence exercise but is expected to be applicable to the defence of many cyber infrastructures. In LS, human teams from different nations have to defend their networked infrastructure against a series of attacks while maintaining operational services for several days. In contrast to classical defence teams in LS, which may consist of large numbers of human players, our framework consists of an AI-powered system without any human intervention once the exercise has started.

To the best of our knowledge, our framework is the first attempt towards a fully automated cyber defence team. In particular, our contributions in this paper are:

- A description of the situation during LS for attackers and defenders (Section 2);
- A framework architecture for an automated team to participate in LS (Section 3);
- System descriptions of the main components of this framework including sensors and situational awareness (Section 4), actuators (Section 5), AI engine (Section 6) and control logic (Section 7); and
- A case study highlighting how our framework is able to mitigate the impact of attacks in LS (Section 8).

Related Work: A cyber exercise such as LS can be considered as a game with complex and continuously changing rules. Past developments in AI have resulted in computers outperforming human players in games such as chess and go. Game algorithms have traditionally been designed for a single game with predefined rules. However, the authors of [8] also apply AI for learning to play chess without a priori knowing the rules. AI adapting to the rules of several games to achieve a general game system has also been studied in [9]. The problem we consider in this work is different as the attacker does not have to follow a predefined set of moves or rules.

The concept of cyber defence exercises and their importance in training experts and testing defence processes has been studied in [10]. The use of AI in defending IT systems has been considered beneficial for years. Already in 2011, Tyugu described that defending cyberspace cannot be handled by humans due to the speed of processes and the amount of data involved [11]. Using machine learning to detect malicious traffic in industrial networks has been studied in [12]. Bogatinov et al. predict that autonomous AI agents defending IT systems will gradually become better than humans [13]. On the other hand, there is a constant arms race between defenders and attackers also in cyberspace. If AI can be used to improve cyber defence, it can also be used to improve attacks and malware as is studied in [14] and [15].

2. BACKGROUND ON LOCKED SHIELDS

Below, we summarise key aspects of LS: the roles of the different teams, and the scoring system.

A. Teams and Organisation

The two most important teams concerning this paper are the defenders (blue team) and the attackers (red team), which we explain in more detail below. Table I provides an overview of all teams.

TABLE I: TEAMS IN LS.

Team name	Role	Explanation
Blue Team (BT)	Defender	Needs to defend its infrastructure against attacks from the RT while maintaining availability.
Red Team (RT)	Attacker	Executes attacks against the infrastructure of each participating Blue Team.
Green Team (GT)	Infrastructure operator	Creates and maintains the technical exercise infrastructure.
Yellow Team (YT)	Monitoring	Provides situational awareness during the exercise.
User Simulation Team (UST)	Benign users	Legitimate system users with poor cyber hygiene conducting activities on BT systems.
White Team (WT)	Organiser	Responsible for non-technical aspects of the exercise.

Blue Teams (BTs) are the main training audience. There are more than 20 BTs and each of them acts independently within its dedicated infrastructure (the Gamenet). The main tasks of each BT are to harden the provided Gamenet to be available and resilient before the attacks start and to defend them against ongoing attacks.

The **Red Team** (RT) conducts attacks to accomplish predefined objectives (e.g. gain control of a device). The RT is not the training audience, and so its activities are highly regulated to maximise fairness. The RT has three sub-teams: Web, Client-side (CS), and Network and special systems (NET + SS). SS consists of all ICS and cyber-physical systems. The RT knows vulnerabilities in advance and uses a large toolset to attack. It includes a command and control (C&C) infrastructure, custom, and known malware. In addition, the RT takes advantage of user(s) to run commands (e.g. download malware) on Gamenet machines.

BT communication channels with other teams: Besides defending their systems, the BT is required to maintain predefined communication channels as listed in Table II.

TABLE II: COMMUNICATION CHANNELS.

With	Channel	Purpose
GT	Web ticketing, chat	Request manual reverting of SS, receive or request information about Gamenet status
YT	Web interface	Provide periodical or on-demand reports, threat reports, key events, situation reports, adversary assessments
UST	Web ticketing	Read, address and respond to UST tickets
WT	Voice or video call	Verify voice or video call functionality in the Gamenet, voice or video reporting

B. Scoring

LS has a complex scoring system that changes yearly but the most important high-level objectives from a BT perspective remain constant: *(i)* prevent the RT from reaching its objectives; *(ii)* keep systems available; and *(iii)* interact with the UST and YT.

RT objectives: The RT follows a list of objectives it wants to achieve in each phase of the gameplay. After each phase, the scoring is updated based on the achieved objectives in each Gamenet.

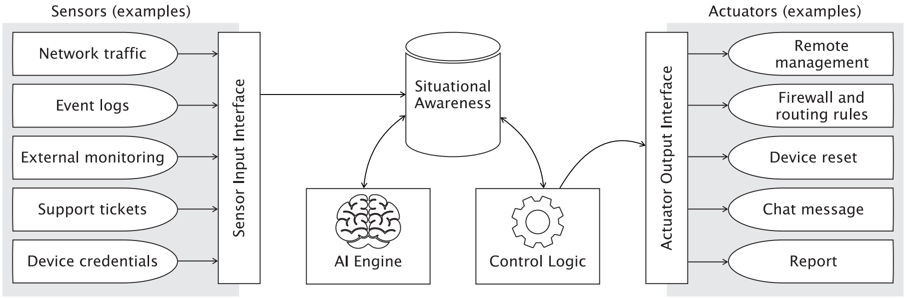
Availability: All systems in the Gamenet need to be available as specified in the rules. Mistakes by the BT (e.g. over-aggressive firewall rules) or attacks from the RT (e.g. compromised services) reduce the availability score.

Interactions with UST and YT: The UST continuously accesses systems and services in the Gamenet. If a service does not work as expected, it creates a support ticket and the BT needs to solve the issue. Otherwise, it will incur a scoring penalty. In addition, the YT periodically requests reports about failed and successful attacks from BTs.

3. ABT FRAMEWORK OVERVIEW

We now provide an overview of our framework to implement an automated BT (ABT). Our framework is designed for LS, but it could easily be adapted to other environments. In our framework, the skilled human players in a traditional BT are replaced with an architecture as shown in Figure 1.

FIGURE 1: ARCHITECTURE OVERVIEW



There are five types of building blocks:

- **Sensors** provide measurements and data;
- **Actuators** trigger actions;
- The **situational awareness database** contains all the sensor data, the AI state and external data (e.g. logs, AI models and malware signatures);
- The **AI engine** continuously (re-)learns models (e.g. to detect anomalies);
- The **control logic** determines actions for actuators to execute based on the available information in the situational awareness database.

4. SENSORS AND SITUATIONAL AWARENESS

We now describe how the ABT maintains situational awareness by collecting data and building models based on these data. We group the sensors in five categories: application-level, device-level, network-level, user-interaction, and organisational inputs.

A. Application Sensors

Application state and configuration: This includes for example the contents of the database, the registry or the filesystem. This allows the ABT to identify misconfigurations, vulnerable components, weak credentials, unauthorised modifications, and to restore a compromised application. The application state can be retrieved at the beginning of the exercises (e.g. for static configurations) or periodically (e.g. for dynamic state).

B. Device Sensors

System information and logs: For example, process creations, filesystem activities, patch level, privileged activities, machine fingerprinting, interaction with remote services, and domain controllers. Besides logs that are collected by default, the ABT may log additional events and forward them to the situational awareness database.

Remotely executed command outputs: The ABT evaluates and collects the outputs of commands executed by the actuators on Gamenet devices.

Active monitoring: the ABT deploys sensors to perform on-demand device monitoring or active probing. This could not be possible on some devices (e.g. SS) because of resource constraints or because of their proprietary architecture. Monitoring agents on workstations can allow logging mouse and keyboard inputs interacting with proprietary GUI applications.

C. Network Sensors

Network traffic: By default, one virtual machine (VM) receives a real-time stream of packets captured in the Gamenet. The traffic is captured at particular devices in the network (switches and routers). Therefore, it only contains packets crossing these devices and can contain the same packet multiple times. To capture additional traffic, the ABT configures additional sensors, for example by recording traffic at hosts.

Network configuration: To discover the network topology and configuration, the ABT relies on network scanning, management and diagnostic tools such as nmap, snmp, or traceroute in addition to the organisational inputs (see below). These sensors allow it to determine the actual behaviour of devices and compare this with the specifications.

D. User-Interaction Sensors

Support tickets: The UST creates support tickets that indicate problems in the Gamenet (usually related to availability and functionality). This input is important for situational awareness because functionality issues are especially hard to detect automatically.

E. Organisational Inputs

System and credential list: The ABT receives a list of all systems in the Gamenet and credentials to access them with administrative privileges. The ABT leverages available remote access services (e.g. SSH, telnet, RDP, and web interfaces) for situational awareness.

Scoring system and external monitoring: The primary components of the scoring are objectives accomplished by RT and the availability state of all systems. These inputs are essential for situational awareness to understand how successful or unsuccessful actions and attack mitigations have been in near real-time.

F. Situational Awareness Database

The situational awareness database contains all the information that is relevant for the ABT to succeed. It is a central entity that includes all collected sensor data, any learned models from the AI engine, actuator output as issued by the control logic, as well as relevant information from outside the context of LS, such as known software vulnerabilities or indicators of compromise. The collected and processed data is stored persistently and made available to the AI engine and the control logic. The ABT represents the stored data in a global knowledge graph in order to reason about the current situation including the Gamenet and the previous interactions with all the other teams.

The data sources are heterogeneous and the data must first be pre-processed and then fused in order to allow queries over data coming from multiple sources and formats. For example, a query to the database might correlate data from device and network sensors in order to identify compromised services that leave attacker traces in both data sources. Furthermore, it is possible to correlate system failures identified from support tickets with application state, so that the control logic can derive the services that may need a restart or a recovery.

5. ACTUATORS

We will now describe the actuators at the application, device and network level, as well as actuators for interacting with humans from other teams.

A. Application-level Actuators

Web application firewall configuration: To block the common attack vectors (e.g. malicious file uploads), the control logic deploys policies to inspect requests and to block those which – according to the AI models – are malicious.

Application configuration and patch: Change the configuration of an application, patching vulnerable libraries, or to update credentials.

B. Device-level Actuators

Remote management: This actuator initially allows the ABT to log in to any device with administrative privileges. If possible, the ABT installs tools to enable effective remote management and infrastructure as a code (IaaC). Having multiple options for remote access improves the ABT's chances of regaining access after a system was compromised and allows consistent management of operations on the Gamenet.

Restoring a device's initial state: The ABT can revert a device to its original state (resulting in a scoring penalty). VMs can be reverted automatically by the ABT's actuators, while some special systems require manual intervention by the GT.

C. Network-level Actuators

Device configuration and rules: To control network traffic, the ABT uses actuators to dynamically adapt the configuration and the rules of firewalls and routers. Initially, the devices are set up as simply as possible: firewalls let everything pass and routers merely guarantee connectivity. The ABT configures these devices to limit authorised traffic, and – if possible – replaces the software running on them. The actuator can also apply a new rule to block traffic from or for compromised devices. To gain higher control over the network traffic, the actuator may also change the routing behaviour of the devices; for example, to configure the network such that all traffic passes a programmable controller.

D. User-Interaction Actuators

Chat and email messages: The ABT has to address and respond to messages sent by other teams. This actuator thus generates human-readable chat and email messages based on the current situation.

Incident reports: The ABT also needs to provide reports about successful and failed attacks to the GT. This actuator reports about those incidents to the GT.

6. AI ENGINE

The AI engine's task is to learn models from data and infer facts from historical events in the Gamenet. It gets its data from the situational awareness database, performs machine learning techniques, and feeds the learned models and facts back to the database. Below, we present a categorisation of AI approaches and then sketch how models are trained.

A. AI Categorisation

We characterise each AI-enabled defence according to three dimensions:

1. Level: How sophisticated is the AI? Example: Narrow AI
2. Tasks: What does the AI do? Example: Classify samples
3. Type: Which AI technique is used? Example: unsupervised learning

The four levels of AI:

- **Level 0 – Reactive narrow AI:** Level-0 AI is very basic in the sense that it only reacts to current inputs. This restricts it to simple decisions (e.g. if a network packet has destination port 22, drop the packet). This level of automation is the same as many existing tools (e.g. [1], [2], [16], [17]). For example, popular intrusion detection systems, such as Snort [2], check network traffic for known signatures, or antivirus software checks executables against a database of known malware. The contribution of Level 0 should not be underestimated in our context. Using external sources (rules, IOC, patterns, etc.), it is possible to identify traces of attacks with few computational resources.
- **Level 1 – Limited-memory narrow AI / Weak AI:** Level-1 AI is what is typically understood as machine learning today (e.g. Siri, Watson, self-driving cars). Many proposed machine learning solutions [18–26] already solve tasks that are important for the ABT. This kind of AI is equipped with data storage and learning capabilities that enable the ABT to use historical data to make informed decisions.
- **Level 2 – General AI / Strong AI / Deep AI:** Level-2 AI mimics human-level intelligence but it does not exist yet. We, therefore, do not consider it in this work.

- **Level 3 – Super AI:** Level-3 AI is considered self-aware AI that surpasses human intelligence. It is not entirely clear if this kind of AI can be achieved and is also out of the scope of our work.

The five tasks for AI we consider useful are:

- **Identification/classification:** tell me what the thing is, such as detecting command and control flows (e.g. [18]); Intrusion/anomaly detection (e.g. [19]–[21]);
- **Categorisation:** group similar things together such as log clustering for detecting security issues (e.g. [22]);
- **Assessment:** tell me whether I should care about this thing, for example, to prioritise security events (e.g. [23]);
- **Recommendation:** tell me what to do about this thing in order to, for example, recommend defence actions (e.g. [24]);
- **Prediction:** tell me if this thing, for example, predicts attacks or vulnerabilities (e.g. [25], [26]).

Three types of AI exist:

- **Supervised:** learning from labelled training data. Supervised approaches can use the situational awareness data from the current and/or past iterations of LS for training;
- **Unsupervised:** detecting previously undetected patterns in unlabelled data, using, for example, clustering techniques;
- **Reinforcement:** AI takes decisions and receives feedback which it uses for future decisions.

B. AI Models for Improved Situational Awareness

Below, we sketch models and applications of the AI engine to improve the situational awareness of the ABT.

Anomaly detection: The AI engine learns the behaviour of applications, devices, and network traffic from sensor data in order to build models of normal behaviour and detect anomalies or intrusions (e.g. an application suddenly deleting many files). For unsupervised learning, the models are learned based on the actual data while supervised learning techniques require labelled datasets from previous exercises.

Data fusion: The situational awareness database contains information from many different sources and the AI engine fuses this information to extract new insights. For example, a successful defacement attack resulting in support tickets, event log entries, and network connections that seem unsuspicious at first is inferred as an attack by means of time correlation.

Prioritise defence actions: Often, the ABT has multiple options to restore a system after a successful attack (e.g. patching or reverting the device) but the options have different costs and side-effects (e.g. scoring penalty, system availability). The AI engine therefore models and suggests the best strategy given the information about the current situation, an objective function (e.g. the scoring rules), or potential previous observations (reinforcement learning).

Predict future attacks: Since the attacks are similar in each iteration of LS, the AI engine learns using data from previous iterations to predict which attacks will happen. Furthermore, predictive models are learned to infer if an observed attack is likely to be performed against other devices.

Interact with humans: The ABT is required to interact with humans in other teams. For this, a chatbot based on AI and natural language processing techniques (e.g. [27]–[30]) is used. The chatbot is able to ask humans about, for example, the classification of the problem and the target IP address. This information is inserted into the situational awareness database. If the chatbot is not able to understand or resolve the problem, the chatbot can ask the user for more information or try to please her with generic statements to optimise scoring points.

Summary: In Table III, we summarise the tasks of the AI engine at the different stages of the exercise categorised according to their function.

TABLE III: TASKS FOR AI MODELS

	Task				
Satge	**Identification**	**Categorisation**	**Assessment**	**Recommendation**	**Prediction**
Initial hardening	Detect misconfigurations between similar clients	Find groups of similar devices	Identify potentially vulnerable devices	Generate secure configurations	Predict events that could indicate a compromise
Monitoring & Response	Detect malicious applications/ commands/ network traffic	Detect malicious patterns in log files	Prioritise security events	Select defence / restore action	Predict future attacks
Reporting	Understand support tickets	Link support tickets and monitoring alerts	Prioritise tickets	Formulate response to ticket	Predict impact on the scoring
Recovery	Find devices that need to be recovered	Find a similar system as a template for the recovery	Determine whether a system needs to be recovered	Select recovery strategy	Predict impact on the scoring

7. CONTROL LOGIC

We will now describe the tasks managed by the control logic, first in the initialisation phase, and then in the gameplay phase. The control logic makes decisions about whether and which actuators to actuate in which way depending on the information available in the situational awareness database.

A. Defence Techniques and Goals

The goal is to defend against complex cyber attacks by hardening the system and maintaining/restoring service availability in order not to lose scoring points. Table IV provides an overview of the employed defence techniques of the control logic for the different stages before and during the exercise.

TABLE IV: DEFENCE TECHNIQUES

Satge	WEB	CS	NET + SS
Initial hardening	Set up WAF. Replace applications with secure clones. Security evaluation.	Set up a centralised management tool. Deploy antivirus, firewall, execution policies, monitoring agents.	Deploy firewall rules for all networks. Deploy network IDS/IPS. Secure interfaces (e.g. of the ICS devices).
	Identify vulnerabilities, patching software and fix misconfigurations. Change credentials for OS accounts, services and application accounts. Remove unrequired accounts and services. Identify and remove RT implants.		
Monitoring & Response	If exploited functionality in the web application can be identified after or mid-attack, disable access to it and attempt fixing.	Process creation and file access have to be monitored to identify and block suspicious activities. Monitoring tools must also be closely guarded.	Locate and remove rogue network connections and rogue hosts. If SS is misbehaving, attempt to identify the cause and deploy rules to minimise unauthorised access.
	Anomaly detection. If a suspected attack, data exfiltration or C&C channels or processes can be identified – block the relevant network traffic and terminate processes.		
Reporting	AReport successful and failed attacks, provide adversary assessment.		
Recovery	Self-recovery generally possible by preparing database and source code (e.g. scripts) dumps in advance.	Self-recovery generally not possible, the automated reverting interface is available to the BT.	Self-recovery is not possible and GT has to take action.

B. Initial Hardening

Since the Gamenet initially comes in a relatively unprotected state, the first task of the control logic is to harden the devices, networks and applications based on templates of secure configurations prepared by humans.

General: The control logic replaces all credentials and disables unused services on all machines. If required, it sends the new credentials to the UST. Operating systems and software updates are installed. Wherever possible additional security software is installed (e.g. antivirus solution, monitoring agents).

Workstations: CS consists primarily of Windows workstations and domain controllers and some Linux or macOS workstations [31], [32]. Workstations are managed in a centralised manner by utilising group policies. Applying group execution policies allows us to whitelist known benign applications while blocking unknown services. Applying group application firewall policies further allows us to restrict the communications of malicious software that has bypassed execution policies. Initial policies are defined by learning workstation behaviour at the AI engine before the attack stage.

ICS devices: Defence involves protecting VMs running engineering workstations and special software combined with physical components. ICSs require initial configuration to mitigate basic vulnerabilities. Depending on the services required from the device, the ABT disables web interfaces or protects them with strong passwords.

Network: The ABT knows which services are running and need to be available at each of the devices from the organisational inputs, and it can derive firewall rules which only allow expected traffic. This is especially important in ICS protocols with no encryption or authentication. If the existing firewalls in the Gamenet are not enough to perform this action (e.g. because not all traffic crosses them), the control logic attempts to change the routing behaviour to forward the traffic through a central controller or a gateway which sees all traffic and can control it (e.g. by means of modifying or blocking the traffic).

Web applications: Since most of the web traffic is encrypted, it is difficult to filter on the network or transport layer and a Web Application Firewall (WAF) becomes the primary defence tool. While a WAF might only protect against basic attack vectors by default, the AI engine creates a behaviour profile based on data from the monitoring sensors and UST interactions in order to enable more aggressive filtering while maintaining functionality. In addition, the AI engine may also reveal vulnerabilities to the control logic from analysis of the web application source code.

Files: While full-disc imaging or filesystem copying is not possible per exercise limitations, calculating hashes of the file system tree on all machines is beneficial for situational awareness. When standard OS file and software hashes are available, pre-planted backdoors and non-default configurations can be detected at this stage already. For some applications (primarily web), it is possible to create snapshots by copying files and creating database dumps.

C. Actions during Gameplay

During the LS exercise, the control logic attempts to identify attacks and mitigate their impact.

Revert and reboot: Quick recovery to a working state can often maintain the availability score. In simple cases, a reboot restores the functionality of a system. If an application can be restored from snapshots created by the ABT in the initial hardening phase, this is preferred as it does not incur a scoring penalty. If this fails, the penalised in-game revert of VMs is executed (restored state could include vulnerabilities). More complex systems might require a pre-defined order of reverting and rebooting multiple targets. SS might require separate interfacing with the GT to request manual reverting of physical devices.

Block malicious traffic: In addition to the static firewall rules, the control logic also relies on learned AI models to identify traffic that is associated with malicious activities and blocks it.

Identify senders of malicious traffic: If the sender or the receiver of malicious traffic is a device under the ABT's control, the control logic takes further actions to patch the device or block it completely.

Block malicious processes: The execution policies might not stop all malicious processes. Based on each new process' properties and event logs (or events for existing ones), the control logic uses learned AI models to determine if the process should be terminated.

Block malicious application requests: Application (most commonly web) requests are intercepted for analysis by the AI engine. Those requests identified as malicious are blocked before being processed.

D. Human Interaction

Report successful and failed attacks: The control logic gathers facts from the situational awareness database regarding all knowledge it has about attacks detected, both successful and failed. A chatbot then processes the data and fills in a reporting template adding polite human conversation as necessary.

Respond to tickets: If the control logic has successfully resolved the reported problem or it has made a promising attempt, the chatbot closes the ticket with a response to contact again if the problem persists.

8. CASE STUDY

We will now discuss a case study that shows how the different components of the ABT framework work together to defend against an attack.

During the initial hardening phase, the control logic deploys on all clients an additional trusted certificate authority (CA) and reconfigures clients to forward client traffic to a proxy, enabling both plain and encrypted traffic analysis. The control logic configures sysmon on Windows CS enabling detailed system monitoring. Finally, network traffic is analysed by the AI engine by extracting features as suggested in [33] from the situational awareness database and classifying C&C channels using data from previous LS exercises and random forest classifiers according to Känzig et al. [18].

Now, let us consider a typical case in the early stages of the exercise: the RT fools a user (UST or trough GT) (i) to download a malicious payload; and (ii) to execute it on his own client (CS). As soon as the payload is executed, it (iii) contacts the C&C server and the RT now has a foothold in the Gamenet. This payload could create persistence in order to be used in later phases of the exercise to perform destructive actions by the RT.

When the user downloads and executes the payload, there are three events that are captured in the situational awareness database: (a) from the proxy, the requests to download a malicious payload which is in this case an executable file (e.g. script, binary, library); (b) sysmon reports the execution of an unknown payload; (c) the payload makes contact with its C&C server, which is detected as an anomaly in the traffic analysis.

The correlation of these three events, interpreted as anomalous and suspicious, triggers the following responses at actuators by the control logic:

Malicious processes and files are identified and neutralised: the malicious process on the compromised CS is killed, the payload sent to the situational awareness analysed, inserted in the blacklist of the proxy (e.g., hash, pattern recognition), and finally deleted from the CS.

CS hardening is improved: CS software restriction policies are updated (e.g. payload hash, execution path) in order to avoid further execution of this payload on Gamenet devices.

Network rules updated: identified C&C IPs are blocked on the network firewalls and further activity to or from these IPs is considered potentially malicious.

Report generation: The control logic generates a human-readable report summarising the events related to the compromised hosts from the situational awareness database and sends it via email.

9. CONCLUSION AND OUTLOOK

We have described a novel framework that connects sensors with network, device, application and user actuators in order to automate the defence of complex cyber infrastructures against sophisticated attacks. By connecting these components together through an AI-powered computing system, we can perform automated mitigating actions to quickly and efficiently combat various attacks upon detection. Our framework is thus well suited to protect complex infrastructures which need to maintain service availability against multistage attacks. We have highlighted how our framework functions in the context of Locked Shields, but the proposed framework is applicable to other infrastructures that shall maintain high service availability while under attack. We acknowledge that this paper presents only the framework and its implementation is outside the scope of the paper. Therefore, as future work, we plan to implement our architecture in order to compete in upcoming cyber defence exercises and to compare its performance with human teams. We hope that our work will also inspire other researchers in implementing similar automated cyber defence teams so that these systems could eventually compete and learn from each other.

ACKNOWLEDGEMENTS

We thank the Swiss Blue Team for sharing their data and expertise with us and the anonymous reviewers for their valuable feedback.

REFERENCES

[1] “Zeek.” https://zeek.org/

[2] “Snort.” https://www.snort.org/

[3] H. Alqahtani, I. H. Sarker, A. Kalim, S. Md. Minhaz Hossain, S. Ikhlaq, and S. Hossain, “Cyber Intrusion Detection Using Machine Learning Classification Techniques,” in *Computing Science, Communication and Security*, Singapore, Jul. 2020, doi: 10.1007/978-981-15-6648-6_10.

[4] A. L. Buczak and E. Guven, “A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection,” *IEEE Commun. Surv. Tutor.*, vol. 18, no. 2, 2016, doi: 10.1109/COMST.2015.2494502.

[5] Q. Liu, P. Li, W. Zhao, W. Cai, S. Yu, and V. C. M. Leung, “A Survey on Security Threats and Defensive Techniques of Machine Learning: A Data Driven View,” *IEEE Access*, vol. 6, 2018, doi: 10.1109/ACCESS.2018.2805680.

[6] “Locked Shields.” https://ccdcoe.org/exercises/locked-shields/

[7] K. Kasak, “Lessons learned from Locked Shields 2013 exercise,” in *2nd ENISA Int. Conf. on Cyb. Coop. and Ex.*, Ath., Greece, 2013. [Online]. Available: https://www.enisa.europa.eu/events/2nd-enisa-conference/presentations/kaur-kasak-nato-ccdcoe-lessons-learned-from-the.pdf

[8] O. E. David, N. S. Netanyahu, and L. Wolf, “DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess,” in *ICANN 2016*, 2016, doi: 10.1007/978-3-319-44781-0_11.

[9] D. Silver *et al.*, “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play,” *Science*, vol. 362, no. 6419, 2018, doi: 10.1126/science.aar6404.

[10] E. Seker and H. H. Ozbenli, “The Concept of Cyber Defence Exercises (CDX): Planning, Execution, Evaluation,” presented at Int. Conf. on Cyb. Sec. and Prot. of Dig. Serv. (Cyber Security), 2018, doi: 10.1109/CyberSecPODS.2018.8560673.

[11] E. Tyugu, “Artificial intelligence in cyber defense,” presented at the 3rd Int. Conf. on Cyb. Conf., 2011, Tallinn. [Online]. Available: https://www.ccdcoe.org/uploads/2018/10/ArtificialIntelligenceInCyberDefense-Tyugu.pdf

[12] G. Bernieri, M. Conti, and F. Turrin, “Evaluation of Machine Learning Algorithms for Anomaly Detection in Industrial Networks,” presented at IEEE Int. Symp. on Meas. Net., 2019, doi: 10.1109/IWMN.2019.8805036.

[13] D. S. Bogatinov, M. Bogdanoski, and S. Angelevski, “AI-Based Cyber Defense for More Secure Cyberspace,” in *Handbook of Research on Civil Society and National Security in the Era of Cyber Warfare*, IGI Global 2016, doi: 10.4018/978-1-4666-8793-6.ch011.

[14] N. Kaloudi and J. Li, “The AI-Based Cyber Threat Landscape: A Survey,” *ACM Comput. Surv.*, vol. 53, no. 1, 2020, doi: 10.1145/3372823.

[15] I. Chomiak-Orsa, A. Rot, and B. Blaicke, “Artificial Intelligence in Cybersecurity: The Use of AI Along the Cyber Kill Chain,” in *Comput. Collect. Intel.*, 2019, doi: 10.1007/978-3-030-28374-2_35.

[16] “Suricata.” https://suricata-ids.org/

[17] “Anti-Virus Software.” https://cs.stanford.edu/people/eroberts/cs201/projects/2000-01/viruses/anti-virus.html

[18] N. Känzig, R. Meier, L. Gambazzi, V. Lenders, and L. Vanbever, “Machine Learning-based Detection of C&C Channels with a Focus on the Locked Shields Cyber Defense Exercise,” presented at CyCon 2019, doi: 10.23919/CYCON.2019.8756814.

[19] M. Ahmed, A. Naser Mahmood, and J. Hu, “A survey of network anomaly detection techniques,” *J. Netw. Comput. Appl.*, vol. 60, 2016, doi: 10.1016/j.jnca.2015.11.016.

[20] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, “Anomaly-based network intrusion detection: Techniques, systems and challenges,” *Comput. Secur.*, vol. 28, no. 1, 2009, doi: 10.1016/j.cose.2008.08.003.

[21] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, “A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection,” in *SIAM Int. Conf. on DM*, 0 vols., SIAM, 2003.

[22] M. Landauer, F. Skopik, M. Wurzenberger, and A. Rauber, “System Log Clustering Approaches for Cyber Security Applications: A Survey,” *Comput. Secur.*, vol. 92, 2020, doi: 10.1016/j.cose.2020.101739.

[23] R. Alexander, “Reducing Threats by Using Bayesian Networks to Prioritize and Combine Defense in Depth Security Measures,” *J. Inf. Secur.*, vol. 11, no. 3, Art. no. 3, 2020, doi: 10.4236/jis.2020.113008.

[24] K. B. Lyons, “A Recommender System in the Cyber Defense Domain.” MA diss., Dept. of Elect. and Comput. Eng., Grad. Sch. of Eng. and Man. Air Force Inst. of Tech., OH USA, 2014. [Online]. Available: https://scholar.afit.edu/etd/612

[25] X. Fang, M. Xu, S. Xu, and P. Zhao, “A deep learning framework for predicting cyber attacks rates,” *EURASIP J. Inf. Secur.*, vol. 2019, no. 1, 2019, doi: 10.1186/s13635-019-0090-6.

[26] Y. Shin and L. Williams, “An empirical model to predict security vulnerabilities using code complexity metrics,” *ACM-IEEE Int. Symp. on Empirical Software Engineering and Measurement*, 2008, doi: 10.1145/1414004.1414065.

[27] S. A. Abdul-Kader and J. C. Woods, “Survey on Chatbot Design Techniques in Speech Conversation Systems,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 7, Art. no. 7, 2015. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2015.060712

[28] E. Handoyo, M. Arfan, Y. A. A. Soetrisno, M. Somantri, A. Sofwan, and E. W. Sinuraya, "Ticketing Chatbot Service using Serverless NLP Technology," presented at the ICITACEE 2018, doi: 10.1109/ICITACEE.2018.8576921.

[29] F. E. Office, "AI Chatbot to Realize Sophistication of Customer Contact Points," *FUJITSU Sci Tech J*, vol. 54, no. 3, 2018.

[30] A. M. Rahman, A. A. Mamun, and A. Islam, "Programming challenges of chatbot: Current and future prospective," presented at IEEE R10-HTC, 2017, doi: 10.1109/R10-HTC.2017.8288910.

[31] NATO CCD COE, "LockedShields 2013 After Action Report," https://ccdcoe.org/uploads/2018/10/LockedShields13_AAR.pdf (retrieved 05.01.2021)

[32] NATO CCD COE, "LockedShields 2012 After Action Report," https://ccdcoe.org/uploads/2018/10/LockedShields12_AAR.pdf (retrieved 05.01.2021)

[33] "Zeek Flowmeter." https://github.com/zeek-flowmeter/zeek-flowmeter