MERLINS – Moving Target Defense Enhanced with Deep-RL for NFV In-Depth Security

Wissem Soussi*§, Maria Christopoulou‡, Gürkan Gür§, Burkhard Stiller*
* University of Zürich UZH, Switzerland [soussi, stiller]@ifi.uzh.ch
§ Zurich University of Applied Sciences, Switzerland [sous, gueu]@zhaw.ch
‡National Center for Scientific Research Demokritos (NCSRD),
Greece maria.christopoulou@iit.demokritos.gr

***Abstract*—Moving to a multi-cloud environment and service-based architecture, 5G and future 6G networks require additional defensive mechanisms to protect virtualized network resources. This paper presents MERLINS, a novel architecture generating optimal Moving Target Defense (MTD) policies for proactive and reactive security of network slices. By formally modeling telecommunication networks compliant with Network Function Virtualization (NFV) into a multi-objective Markov Decision Process (MOMDP), MERLINS uses deep Reinforcement Learning (deep-RL) to optimize the MTD strategy that considers security, network performance, and service level requirements. Practical experiments on a 5G testbed showcase the feasibility as well as restrictions of MTD operations and the effectiveness in mitigating malware infections. It is observed that multi-objective RL (MORL) algorithms outperform state-of-the-art deep-RL algorithms that scalarize the reward vector of the MOMDP. This improvement by a factor of two leads to a better MTD policy than the baseline static counterpart used for the evaluation.**

***Index Terms*—Moving Target Defense, 5G and Beyond 5G, NFV Security Management, Deep Reinforcement Learning.**

I. Introduction

5G telecommunication networks introduced the concept of network slicing [1], allowing the creation of isolated virtual networks upon a single infrastructure with independent data, control, and management planes. Each network slice handles these three planes differently, considering their different service level requirements (*e.g.*, communication latency, bandwidth, and power consumption) [2]. Concurrently, Communication Service Providers (CSPs) are deploying Multi-access Edge Computing (MEC) [3], increasing the size and complexity of the infrastructure with intermediate data centers, cloud platforms, and edge nodes at different access points. Nonetheless, network slicing is still possible with Software-Defined Networking (SDN) and Network Function Virtualization (NFV) in this complex setting [4]. These technologies improve the scalability and flexibility of network services by dynamic instantiation, deletion, and chaining of virtual resources and Virtual Network Functions (VNF). However, this environment's heterogeneity and large-scale nature also increases the attack surface of communication systems, i.e., the set of network elements (hardware and software) that can be accessed by authorized or unauthorized entities to exploit them for cyber-attacks. Protecting a larger attack surface is more challenging as attacks have a greater range of targets and can propagate and affect multiple network slices. This situation is expected to become more critical with Beyond 5G or 6G networks, where Quality-of-Service (QoS) requirements are even more stringent, and services are more diverse [5]. In-depth defense composed of multiple layers of security must be in place to secure such softwarized networks.

Moving Target Defense (MTD) [6] is a promising paradigm that broadens automated network management to include security. It leverages the network's flexibility and heterogeneity by shifting virtual resources in time and space. This creates a proactive security mechanism that raises the difficulty level for attackers to conduct reconnaissance and attack planning, as the gathered intelligence quickly becomes obsolete. In a broader perspective, MTD also provides a reactive security mechanism that can use network modifications to counteract attacks or restore infected resources.

In network orchestration and management with an MTD security mechanism [7], the strategic placement and movement of network resources aim security improvement, performance optimization and resource cost efficiency. These objectives do not overlap, and conflicts might arise when performing MTD operations, favoring one goal to the detriment of the other. For instance, moving a VNF from a remote Virtual Infrastructure Manager (VIM) to an edge node's VIM for communication optimization may be a poor choice security-wise, since an attacker can easily predict that action. A purely random placement, instead, improves security by reducing its predictability but can hinder the network's performance and QoS of the moved service.

Consequently, this paper proposes ***MTD Enhanced with deep-RL for NFV in-depth Security (MERLINS)***, a security framework that provides an added security layer on the orchestration of network resources in a dynamic MEC-enabled telecommunications network. To this end, MERLINS monitors a 5G Telco Cloud network and performs risk and threat assessment to elaborate a proactive MTD strategy. Such strategy is learned via deep-RL and tackles a multi-objective optimization problem, defined with a Multi-Objective Markov Decision Process (MOMDP), where the goal is to find the optimal balanced strategy to maximize security (*i.e.,* minimize threats), to minimize its operational cost, and to alleviate the impact on QoS and service availability. This work shows that MORL effectively outperforms State-of-the-Art (SotA) deep-RL methods that scalarize the rewards of different objectives into one value, improving the MTD policy over an expert-knowledge-based MTD static policy defined for our eval-

DOI: 10.1109/NFV-SDN59219.2023.10329594

uation. Moreover, MERLINS is effective against Advanced Persistent Threats (APTs), which are threats characterized by undetected malware infections and unauthorized intrusions for a long duration.

The contributions of this work are summarized as follows: **1)** the design, architecture, and development of an autonomous and adaptive MTD approach for the in-depth defense of NFV Telco Cloud networks; **2)** an effective proactive security framework against undetected APTs; **3)** the evaluation of MORL algorithms against SotA deep-RL for the MTD policy optimization problem in a 5G testbed; and **4)** an open-source implementation of MERLINS main components[1,2,3].

II. Related Work

Aydeger et al. [8] make use of MTD with SDN and NFV to thwart Crossfire Distributed Denial-of-Service (DDoS) attacks. The framework redirects traffic to virtual shadow networks employed to deceive the attackers. Their study reveals the efficacy of SDN-based MTD mechanisms to disrupt attackers' reconnaissance attacks with negligible costs in resource consumption and acceptable network overhead. Similarly, Rawski et al. [9] explored the use of SDN and NFV to implement MTD techniques allowing to change the logical network topology and dynamically attach security VNFs (such as probes or firewalls) to respond to detected threats. Both studies focus on reacting to detected attacks, leaving the proactive security potential of MTD unexplored. While the mitigation methods used there entail manipulating the traffic or adding security VNFs like firewalls, MERLINS also introduces a different method that includes moving the targeted service itself, for instance, by migrating it to a secure cloud node.

As a more recent development, the usage of RL for cybersecurity is gaining widespread attention in academia and industry. Chai et al. [10] presented DQ-MOTAG (MOving Target defense mechanism AGainst Internet DDoS attacks), a novel MTD framework extending a previous MTD solution against DDoS attacks proposed in [11] and optimizing it with deep-RL. In a different vein, Sengupta et al. [12] used the attack graph of a cloud network to model a general-sum Markov Game. The model formulates a Stackelberg equilibrium problem, whose solution is shown to provide an optimal strategy for the placement of security resources to protect cloud systems. Yoon et al. [13] used a multi-agent deep-RL to train an MTD shuffling strategy of the IP addresses layout of in-vehicle networks with SDN capabilities. Microsoft has open-sourced the research toolkit CyberBattleSim [14], allowing users to model an enterprise network by specifying its nodes, servers, running services, and the used communication protocols. The toolkit then trains a deep-RL agent against predefined threat models. However, the toolkit does not allow modeling environments such as 5G Telco Cloud networks based on NFV. Moreover, measurements on how the QoS of applications can be affected by the MTD operations require a realistic testbed and cannot be provided by the proposed simulated environments. To the best of the authors' knowledge, no previous study has proposed to model telecommunication environments as an MOMDP to optimize MTD strategies with MORL, as an alternative to legacy SotA deep-RL algorithms.

[1]https://github.com/wsoussi/MOTDEC
[2]https://github.com/wsoussi/TopoFuzzer
[3]https://github.com/wsoussi/OptSFC

III. MERLINS Framework Design

As depicted in Fig. 1, MERLINS framework design evolves around the NFV architecture (the gray-scale part on the left-hand side). The colored security modules compose the actual framework and consist of the monitoring module, the risk assessment (RI.AS.) module, the MOMDP modeling module, the MTD controller (MOTDEC), the network topology fuzzer (TopoFuzzer) [15], and the deep-RL based optimizer for MTD policies (OptSFC). The blue and violet modules form the initial data-gathering and analysis phase needed to assess the network state in real-time. While the violet modules focus on the threat and risk analysis of the network , the blue ones gather network monitoring data in a relational database management system (RDBMS) and aggregate them to a real-time MOMDP model of the network state. The detection system module, namely Solidshield Systemic (Systemic) [16], is used for the evaluation of the framework against tampering attacks and sends its detection alerts to the monitoring module, which aggregates them to its RDBMS and MOMDP model. Systemic is not extensively covered in this work and is considered only to showcase the integration of additional anomaly detection systems to MERLINS in a reactive MTD scenario (detailed in Section V-A). The main focus of this paper is given to its novel contributions: *(1)* MOTDEC, enforcing MTD operations implemented in the NFV Telco Cloud (Section III-A); *(2)* the monitoring process, including the RI.AS. and MOMDP modules (Section III-B); and *(3)* the OptSFC and its optimization of MTD strategies via deep-RL (Section III-C).

A. MOTDEC and MTD Operations

MOTDEC is responsible for executing the various MTD actions that OptSFC decides to operate. This study classifies MTD actions into two broad categories:

- **Soft MTD actions**: These are SDN-based shuffle operations performed on network interfaces and traffic flows to disrupt network topology fingerprinting on both the network's internal and external/public views. In the internal view, MOTDEC could prevent an attacker inside the network slice from easily exploring and further penetrating it. In the external/public view, the resource is meant to be always accessible by external devices with a public interface, but MOTDEC can provide a different public IP address to suspicious end-users or UEs, allowing further targeted traffic analysis. For this purpose, MOTDEC integrates an SDN controller and uses another MERLINS sub-module, namely the TopoFuzzer, for a seamless live handover of communication to the new instance.
- **Hard MTD actions**: These are operations directly performed on NFV assets used in the infrastructure, whether allocated by the operator for the provision and management

DOI: 10.1109/NFV-SDN59219.2023.10329594

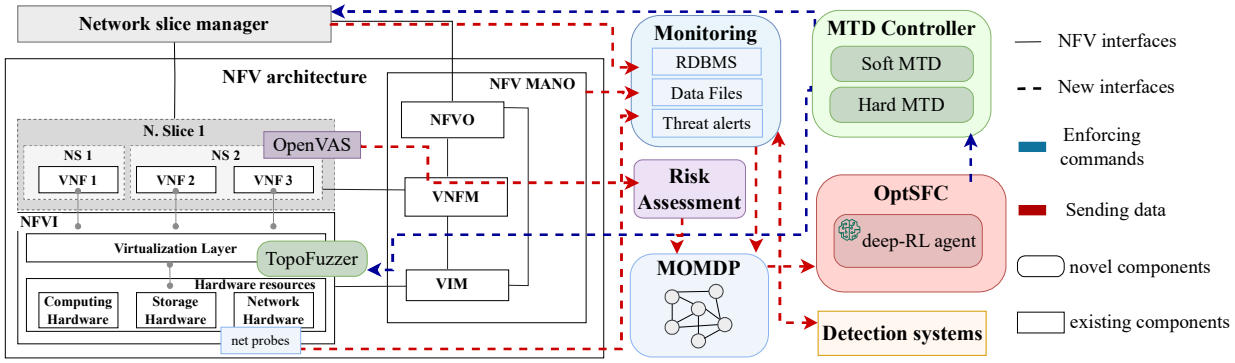


Fig. 1: MERLINS Framework Architecture

of services or NFV assets allocated by the client, following specific service level agreements (SLAs). Such operations are *(1) the live migration* of NSes or VNFs to a different location and *(2) the live re-instantiation/restart* of the same artifacts using verified system images.

As Hard MTD actions are impactful in terms of resource costs and possibly QoS overhead while providing considerable security gains, the remainder of this paper focuses on evaluating the deployment and optimization of such operations. MOTDEC is interfaced with the Katana network slice manager [17] for the management and orchestration of the VNFs at the NFV orchestrator (NFVO) level, required in Hard MTD actions. Hard MTD actions mitigate threats of intruders in the virtual units aiming to eavesdrop and acquire sensitive data, block the application running on the unit (resulting in a DoS attack), encrypt the unit with ransomware, create a command and control (C&C) botnet, and use it as a vector for other chained attacks, or undetected malware originating APTs.

When migrating, MOTDEC can also move the protected resource from a VIM to another one with a different cloud execution environment, e.g., from an OpenStack VIM to a VMware VIM. This action changes the attack surface of the running resource and reduces the threats due to the newly discovered vulnerabilities of a specific system. Another aspect of this specific implementation is that MTD restart actions have as little overhead over the communication performance as the soft MTD actions since the only instant where communications get interrupted is the reconfiguration of the network links using TopoFuzzer. For the migration, the same argument is valid, although there is an additional overhead if the new cloud infrastructure is significantly distant or has worse network performance. For the same reasons, the migration can improve communication performance if the new VIM is physically closer or has better resources. This is another motivation to explore efficient cognitive systems that would strike a trade-off between performance optimization and MTD security efficacy.

B. Monitoring, RI.AS., and MOMDP

The monitoring module is responsible for the data collection regarding the infrastructure's configuration and the runtime metrics of network slices. The module periodically retrieves the catalog of network resources from the network slice manager and NFV MANO, and updates the entities in the RDBMS. The NFV MANO and the network slice manager also deliver information about the life cycle of the network resources (namely the network slices, network services, and VNFs), like their minimum resource requirements, their location (*i.e.*, the hosting VIM), and their runtime states. Low-level data is collected from MMT [18], a network monitoring tool with probes installed in both the core cloud and the edge nodes of the NFV infrastructure (NFVI). This measurement data consists of resource and networking metrics used to monitor the resource consumption and the network state at runtime. The data gathering process is periodic and performed every five seconds, which is the smallest cycle period of MMT probes' metrics collection. The monitoring module can further receive threat alerts from a detection system, enhancing the MOMDP model and supporting the decision-making process.

1) Threat and Risk Assessment Module (RI.AS.): Continuous threat analysis and risk assessment (RI.AS.) are performed to enhance the MOMDP for proactive MTD decisions. Using the open-source vulnerability scanner OpenVAS [19], the RI.AS. module identifies running services using the Common Platform Enumeration (CPE) and performs active and passive vulnerability scans using maintained public and private vulnerability databases such as the Common Vulnerability Enumerations (CVEs) and Network Vulnerability Tests (NVTs) databases. The former allows for finding possible vulnerabilities based on the CPE of the services running in the targeted host. These scans are passive and prompt but may contain false positives. NVTs instead are based on active and more precise scans using local security checks of a range of operating systems and solutions from different vendors (*e.g.*, Intel, Cisco, and Oracle products). The RI.AS. module schedules the scans for all VNFs in one or multiple network slices periodically and every time a VNF is re-instantiated. The module then groups the CVE details of detected vulnerabilities into three general types of threats: 1) APTs, identified by vulnerabilities that allow adversaries to execute code remotely or infect the target with malware; 2) Data Leak Threats, identifying vulnerabilities that allow gaining sensitive information: *e.g.,* SQL and XSS injection, directory traversals, and local file inclusion; 3) DoS Threats, grouping CVEs based on Buffer Overflow vulnerabilities and NVTs finding network-based DoS vulnerabilities.

The RI.AS module quantifies the risks of a VNF for each

DOI: 10.1109/NFV-SDN59219.2023.10329594

TABLE I: Regression results on the CPU and RAM price from 66 percent of cloud market in 2022 Q3.

	Dependent variable:
	Price ($/hour)
α_1 (CPU_core)	0.031^{***} (0.001)
α_2 (RAM_GB)	0.004^{***} (0.0002)
β (constant)	-0.082^{***} (0.018)
Observations	72
R^2	0.994
Adjusted R^2	0.994
Residual Std. Error	0.127 (df = 69)
F Statistic	$5{,}706.468^{***}$ (df = 2; 69)
Note:	*p<0.1; **p<0.05; ***p<0.01

of these three major threat groups in terms of attack success probability (ASP). The ASP value is calculated by considering: **1.** the number of ports open to vulnerable applications, **2.** the maximum *exploitability_score* among detected vulnerabilities, and **3.** the maximum *base_score*. Both scores are obtained from the Common Vulnerability Score System (CVSS) [20] of the National Vulnerability Database (NVD) [21] maintained by the U.S. National Institute of Standards and Technology (NIST). The ASP values are then incremented with time, representing the advantage of attackers when the target is static. When an MTD action is performed on a VNF, the ASP values of the VNF's threats are reset to their original value (i.e., removing the time advantage and only reconsidering the vulnerabilities of the VNF). Finally, similar to the QoS SLAs, OptSFC allows the definition of security SLAs (SSLAs), generalized to a value vnf_impact indicating the criticality of an attack on the VNF. The security risk is then defined for each VNF as the detected threat with the highest impact multiplied by the VNF's criticality value, *i.e.:* $sec\_risk = \max_{threat\ t}(ASP_t \times cvss\_score_t) \times vnf\_impact$.

2) MTD Action Cost Measurement: To measure the cost of MTD actions in terms of resource consumption, empirical measurement of the cost of virtual resources is done to find the coefficients between CPU cost, RAM cost, and storage cost, based on the linear model of $resource_{cost} = \beta + \alpha_1 \times cpu + \alpha_2 \times ram_{gb} + \alpha_3 \times storage_{gb}$. For simplicity, we use the cloud providers' convention of $\$/hour$ as the measurement unit. The publicly available prices of over 70 VM offers are collected, ranging from VM instances of 1 CPU and 0.5GB RAM to instances with 128 CPUs and 864GB RAM, from four major cloud providers: AWS, Azure, Google Cloud, and OVH (covering at least 66 percent of the worldwide market in Q3 2022, estimated by Synergy Research Group [22]). We did not distinguish high-tier hardware from low-tier ones, and as the various prices have different coefficients, a closed-form polynomial system is not solvable. We use linear regression to find the approximate coefficients with a low P-value and a strong correlation between different VM offers. The strong correlation of the variables in the linear regression is represented by the low p-values shown in Table I (also indicated with the * symbols as noted at the bottom of the table). This demonstrates the statistical relevance of the coefficients found per unit costs of CPU and RAM. From such coefficients, we derive the costs of 0.03147 $/h per CPU-core, 0.004244 $/h per GB of RAM, and a constant β of -0.082. As storage is a cloud service provided separately from the VM instance, its cost is separately measured from the average of 37 hot storage prices (*i.e.*, SSD-based fast, regularly accessed, and high data-volume storage with no further distinction on high-tier SSD). Prices are given by the same four cloud providers, giving a final mean price of 0.000066 $/h per GB of storage.

3) The Network's MOMDP Model: For the decision making task, the MOMDP model represents the network as a tuple $(S, A, P, \overline{R}, \gamma)$, where:

- S is the set of all possible states of the Telco Cloud. In practice, at each time t, a state S_t is defined by the status of the resources to be protected, e.g., the 5G core, the UPF, the virtual Evolved Packet Core (vEPC), the gNBs, and the functional VNFs. The status of a resource is defined by its runtime condition (*i.e.,* running, idle, voluntarily stopped, or accidentally stopped), the resource consumption (*i.e.,* CPU, RAM, and storage), its network metrics (I/O frequency, bandwidth, latency, and packet losses), and anomaly detection system alerts (*i.e.,* under attack, suspicious activity).
- A is the set of actions a_i, $i < N_A$, that the RL agent can take. In this MOMDP, the actions are the hard MTD operations available for each network function and the neutral action of "doing nothing".
- P is the transition probability matrix representing the probability that an action a_i changes a specific state s to the state s′, *i.e.*, $\forall a \in A, Pass' = P[S_t + 1 = s' | S_t = s, A_t = a]$. This matrix is updated during training and represents the uncertainty of the RL agent in reaching its goal with a specific action. For instance, the RL agent can decide to perform the action a_i to mitigate an ongoing attack, knowing this action could not succeed with a probability of $1 - Pass'$.
- R is the reward function that defines the reward obtained at time $t+1$ when performing an action a_i from a state *s* at time *t*, i.e., $R_{as} = E[R_{t+1} | S_t = s, A_t = a]$. As the MOMDP has multiple objectives, the reward system of MERLINS is a vector $\overline{R}$ of reward functions, each defined for a specific optimization objective.
- γ is the discount factor used to define the importance of the immediate reward compared to future rewards. As MERLINS performs a single continuous task, the deep-RL model cannot be trained on a batch of episodes, but rather, on the continuity of the management task. Accordingly, γ is set close to 1 to have a more relevant consideration of future rewards during the task continuation, rather than a greedier decision-strategy focused on immediate rewards.

The reward vector $\overline{R}$ groups the collected data/features based on three optimization objectives:

4) Improvement of the risk assessment for proactive security: this is done with the methods and data of the RI.AS. module previously described in Section III-B1.

5) Reduction of the MTD operational cost: As hard MTD actions need to use additional resources to re-instantiate or to migrate a VNF, the operational cost of MTD actions is defined

DOI: 10.1109/NFV-SDN59219.2023.10329594

as $mtd_{cost} = resources_{cost} \times deployment\_time$. The method described in Section III-B2 is used to measure $resources_{cost}$, while $deployment\_time$ is the measured mean value for every VNF running in the network.

6) Improvement of the network performance (and reduction of MTD network overhead): The network performance has to be considered in the OptSFC decision policy. To measure it, OptSFC makes use of the network metrics integrated in the MOMDP model and collected by the MMT probe for every protected VNF, namely: the number of UEs connected to the VNF, connection latency (derived from the RTT values of packets), connection throughput, packet loss rate (derived from packet retransmission requests) and the number of packets flowing in and out. From these monitored values, we derive the mean packet loss rate increase and the mean latency increase caused by the MTD action and thereby define its QoS overhead as $mtd\_QoS\_overhead = (1 + p\_loss\_rate\_increase) \times latency\_increase$. Moreover, if an MTD action causes a violation of a VNF's SLA, the overhead is increased by a penalty factor, which is a hyperparameter of the ML-training.

The MOMDP is implemented in Python, using OpenAI Gym [23], and serves as the observation environment of the deep-RL agent for the training and inference phases.

C. OptSFC and the Deep-RL Agent

By observing the MOMDP model, the deep-RL agent proposes an MTD action to MOTDEC periodically (proactive security case) or immediately after an alert from a detection system (reactive event-driven case). In both cases, the proposed MTD action has to be validated by the network slice manager and the NFV MANO for possible conflicts with other management and orchestration actions. The MERLINS architecture is independent of the deep-RL algorithm used to train the decision-making model. Hence, different deep-RL methods are tested and benchmarked to find the optimal solution to the defined optimization problem. In RL, optimization is essentially based on how the model's decisions affect the environment. This means that the evaluation of the RL model cannot be performed with legacy ML metrics such as false-positive rate, true-negative rate, and the derivative precision, accuracy and recall metrics, which require ground-truth data unavailable for RL's decision-making problems. Comparing the performance of RL models with other models or decision-making systems is the most used and effective way to evaluate their performance.

The deep-RL algorithms benchmarked in this work come mainly from two distinct families of RL: 1) legacy single-objective deep-RL and 2) multi-objective deep-RL algorithms. From the legacy single-objective deep-RL class, four algorithms are evaluated: a deep Q-learning network (DQN) [24], an advantage actor-critic (A2C, synchronous version of A3C [25]), a proximal policy optimization (PPO) algorithm [26], and a variation of PPO with action masking (MaskablePPO) proposed in [27] for scenarios with variable action space states. From the multi-objective deep-RL class, two algorithms are used: Envelope MORL [28] and the expected utility policy gradient (EUPG) [29]. The deep-RL agents are implemented with Stable-Baselines3 [30] and MORL-Baselines [31].

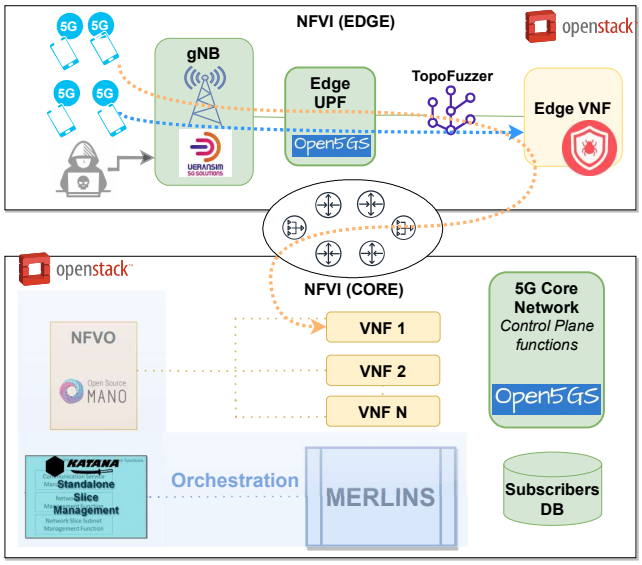


Fig. 2: 5G Testbed Deployment

IV. 5G Testbed Configuration

Fig. 2 illustrates the topology of the 5G testbed where we evaluated the MERLINS framework. It comprises two cloud environments, operated with two OpenStack instances; one for the Edge domain, collectively termed "Edge NFVI", and another for the Core domain, the "Core NFVI". A distributed User Plane Function (UPF) is deployed with UPFs collocated in the Edge domain. The Edge NFVI includes Radio Access elements, *i.e.*, 5G UEs, gNBs, and Edge Cloud elements, *i.e.*, the UPF, a generic VNF for service provision, and the TopoFuzzer. The Core NFVI hosts the control plane of the 5G Core Network, the subscriber database, and generic VNFs for service provision. Two different network slices are defined, distinguished in the figure by the color of the arrows representing their traffic on the data path: a public one managed by the CSP (with an orange arrow), containing the 5G core VNFs, and one private slice (with a blue arrow) containing a generic edge VNF. The Core NFVI also hosts the MERLINS framework, the Katana network slice manager, and the open-source NFV MANO OSM [32], managing and orchestrating the life-cycle of virtualized network resources. The 5G Core is implemented with Open5GS [33], an open-source 3GPP Release-16 compliant 5G core. The Radio Access Network (RAN) and mobile UEs are implemented by UERANSIM [34], an open-source UE and gNB simulator.

V. Results

A. MOTDEC - Reactive scenario

This experiment showcases MERLINS' reactive security using the MOTDEC module in a test case mitigating a malware infection originating binary tampering attacks on the targeted VNF application. The OptSFC module is interfaced with

DOI: 10.1109/NFV-SDN59219.2023.10329594

SolidShield Systemic, a tampering detection system performing binary integrity checks at run-time. A Proof-of-Concept C&C malware, for which remote control is emulated with a REST API interface, is installed in an edge VNF of the 5G testbed. As Systemic detects the attack, OptSFC receives the attack alert, which ingrains a static security policy designed to perform specific MTD operations for specific detected threats — in this case, a tampering (binary corruption) attack.

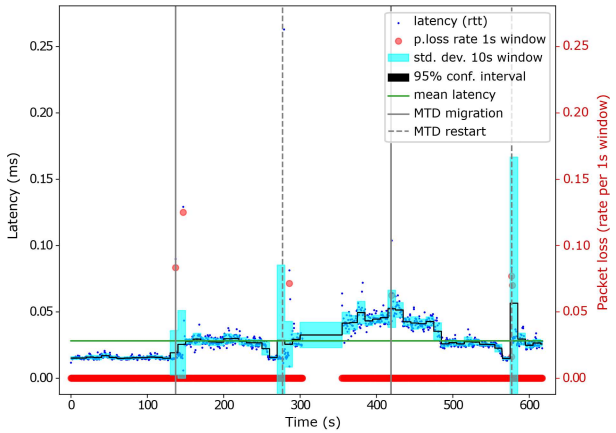
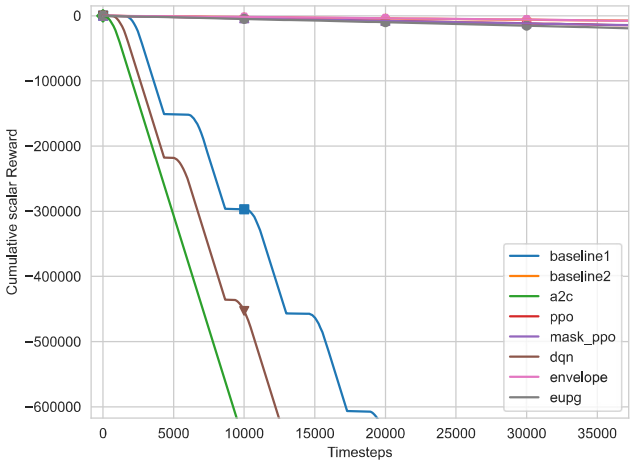


Fig. 3: Latency and packet loss rate during Hard MTD actions on a VNF with 10 HTTP/2 connected UEs

In this scenario, the focus is on evaluating the feasibility and potential overhead of MTD migrations and re-instantiations on the VNF's QoS. To this end, a test is conducted on 10 UEs simultaneously communicating with the target VNF during the MTD operations. The QoS overhead is evaluated based on the latency and packet loss rate of the communications during the MTD actions. Results show that the highest MTD network overhead occurs with HTTP/2 and TCP connections (see Figure 3), with an average 7% increase in the packet loss rate in a one-second timeframe for re-instantiations and 33% increase for migrations (showed with the red scale on the right side and the red points in the figure). This overhead occurs only in a one-second time frame during the TopoFuzzer connection handover, with an average service downtime of 330ms. Assuming a strict availability requirement of 99.999%, this downtime translates to a limit of 17 MTD migrations or 82 MTD re-instantiations per service per week. With a considerably smaller downtime of 10ms for HTTP/3 handovers, the limit rises to 1252 MTD migrations and 5902 MTD re-instantiations per week. Ultimately, Hard MTD actions on the attacked VNF using a new uninfected instance of the service, coupled with the seamless handover of client connections using TopoFuzzer, moves UE connections and the VNF application neutralizing the tampering attack's effects.

B. OptSFC - Proactive scenario

In this scenario, the goal is to prevent attacks by reducing the ASP via proactive MTD operations. A comparative study is done using, during deployment, online training of the deep-RL algorithms for one million timesteps each. Depending on



(a) Continuous management performance of the deep-RL models

RL model	$\bar{x}$ rew/step	RL model	$\bar{x}$ rew/step
Baseline1	-0.44298	*Baseline2*	-0.2123
DQN	-0.647	A2C	-29
PPO-MLP	-0.39	MaskPPO-MLP	-0.39
EnvelopeMORL	**-0.2114**	EUPG	**-0.1932**

(b) Average ($\bar{x}$) reward per step

Fig. 4: Cumulative reward and rew/step

the model's training speed, each model's training phase took from one to three hours. The models' performance is compared with a random MTD policy (*baseline1*) and a static MTD policy (*baseline2*). *Baseline1* is a simple policy that randomly selects an MTD action from the possible ones, *i.e.*, actions that target an existing VNF (the MOMDP's size is fixed so it has been oversized to contain a hypothetical maximum number of 100 simultaneous VNFs, while the actual number of VNFs running in the 5G testbed is four). In contrast, *Baseline2* follows an MTD strategy for the running VNFs based on the security priority level attributed to each of them by a security administrator with the *vnf_impact* value described in Section III-B1. Next, it decides on the MTD action (*i.e.*, migration or re-instantiation) based on the SLA's availability values as introduced in Section V-A. Moreover, while the MOMDP model allows non-conflicting MTD actions to move different VNFs simultaneously, *baseline2* runs only one MTD action at a time to contain the framework's overhead in terms of resource and networking.

The results from the benchmark clearly show a better optimization of the MOMDP return on rewards when using the MORL algorithms described in Section III-C, compared to single-objective models trained on the scalarization of the reward vector $\overline{R}$. In fact, only the Envelope MORL and EUPG algorithms learned a slightly better policy than *baseline2*. A2C appears to get stuck in a local minimum during training and its learned policy avoids performing any MTD action, as to circumvent the penalties of selecting an invalid action such as a non-existing VNF or a VNF that is still undergoing a prior MTD action. DQN policy learned that selecting the same MTD action is better than doing nothing as it resets the ASP values of the given VNF's threats. However, since other VNFs are left

DOI: 10.1109/NFV-SDN59219.2023.10329594

with their threats increasing the ASP with time, the cumulative reward of the proactive MTD management drops exponentially as visualized in Figure 4, where results of models with very similar performances are overlapping. For instance, both PPO and Maskable PPO models learned to re-instantiate the VNFs defined as most critical, leading to an average reward per step of -0.39, greatly improving over *baseline1* and improving the cumulative reward to a linear reduction. One of the reasons is that once the availability constraint of 99.999% is reached, the VNF cannot be moved. *Baseline2* still performs better as it considers the SLA availability constraint and follows a more evenly distributed selection of VNFs, reaching this constraint less frequently. Envelope MORL's policy allows to have similar performances, moving critical VNFs more often and also moving all other VNFs evenly and with less frequency. Finally, we identify the improvement of the EUPG model over Envelope MORL by its tendency to select re-instantiations over migrations, as the former has less network overhead than the latter.

VI. Conclusion

This paper presents MERLINS, a network security architecture that integrates a smart security layer to the NFV architecture using MTD. The MTD strategy and decision-making is optimized for proactive and reactive security using static and deep-RL optimized policies. The results show that MORL algorithms such as EUPG outperformed legacy deep-RL such as DQN and PPO by at least two-fold. Moreover, the experiments in the testbed show that MERLINS effectively neutralizes APTs and can operate efficiently under strict SLA requirements. Future work in this direction is planned for the improvement of the ML model, while making its decisions more explainable when MERLINS is used in a large-scale multi-tenant Telco Cloud, a critical infrastructure where AI/ML decisions should be humanly explainable for technical, economic, and legal reasons.

References

[1] P. R. et al., "Network slicing to enable scalability and flexibility in 5G mobile networks," *IEEE Communications Magazine*, vol. 55, 2017.

[2] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[3] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1657–1681, 2017.

[4] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN — key technology enablers for 5G networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, 2017.

[5] P. Porambage, G. Gür, D. P. Moya Osorio, M. Livanage, and M. Yliantila, "6G security challenges and potential solutions," in *2021 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*, 2021, pp. 622–627.

[6] R. Zhuang, S. A. DeLoach, and X. Ou, "Towards a theory of moving target defense," in *Proceedings of the First ACM Workshop on Moving Target Defense*, ser. MTD '14, 2014, p. 31–40.

[7] W. Soussi, M. Christopoulou, G. Xilouris, and G. Gür, "Moving target defense as a proactive defense element for beyond 5G," *IEEE Communications Standards Magazine*, vol. 5, no. 3, pp. 72–79, 2021.

[8] A. Aydeger, N. Saputro, and K. Akkaya, "A moving target defense and network forensics framework for ISP networks using SDN and NFV," *Future Generation Computer Systems*, vol. 94, 2019.

[9] M. R. et al., "MMTD: MANO-based moving target defense for corporate networks," in *2020 World Conference on Computing and Communication Technologies (WCCCT)*, 2020.

[10] X. Chai, Y. Wang, C. Yan, Y. Zhao, W. Chen, and X. Wang, "DQ-MOTAG: Deep reinforcement learning-based moving target defense against DDoS attacks," in *2020 IEEE Fifth International Conference on Data Science in Cyberspace (DSC)*, 2020, pp. 375–379.

[11] Q. Jia, K. Sun, and A. Stavrou, "MOTAG: Moving target defense against internet denial of service attacks," in *2013 22nd International Conference on Computer Communication and Networks*, 2013, pp. 1–9.

[12] S. Sengupta, A. Chowdhary, D. Huang, and S. Kambhampati, "General sum markov games for strategic detection of advanced persistent threats using moving target defense in cloud networks," in *Decision and Game Theory for Security*. Springer International Publishing, 2019.

[13] Y. S. et al., "Moving target defense for in-vehicle software-defined networking: IP shuffling in network slicing with multiagent deep reinforcement learning," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II*. SPIE, 2020.

[14] M. D. R. Team., "Cyberbattlesim," https://github.com/microsoft/cyberbattlesim, 2021, created by Christian Seifert et al.

[15] W. Soussi, M. Christopoulou, T. Anagnostopoulos, G. Gür, and B. Stiller, "Topofuzzer — a network topology fuzzer for moving target defense in the telco cloud," in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Miami, Florida, USA, 2023.

[16] "Solidshield Systemic," https://www.solidshield.com/dox-preview/systemic/, accessed: April 27, 2023.

[17] Themis Anagnostopoulos, "Katana network slice manager." [Online]. Available: https://github.com/medianetlab/katana-slice_manager

[18] Montimage, "MMT - Montimage Monitoring Toolbox," https://www.montimage.com/products#MMT-DPI, accessed: Apr. 27, 2023.

[19] "Greenbone OpenVAS – open vulnerability assessment system," https://www.openvas.org/, accessed: April 27, 2023.

[20] "National vulnerability database (NVD) - vulnerability metrics," https://nvd.nist.gov/vuln-metrics/cvss, accessed: 2022-02-11.

[21] "NIST National Vulnerability Database (NVD)," https://nvd.nist.gov.

[22] "Q3 Cloud Spending Up Over $11 Billion from 2021 Despite Major Headwinds," Synergy Research Group, Article, Oct. 2022.

[23] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "OpenAI Gym," *CoRR*, vol. abs/1606.01540, 2016. [Online]. Available: http://arxiv.org/abs/1606.01540

[24] V. Mnih et al., "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[25] ——, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 2016.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[27] C.-Y. Tang, C.-H. Liu, W.-K. Chen, and S. D. You, "Implementing action mask in proximal policy optimization (PPO) algorithm," *ICT Express*, vol. 6, no. 3, pp. 200–203, 2020.

[28] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," in *Advances in Neural Information Prsyocessing Systems 32*, 2019.

[29] D. M. Roijers, D. Steckelmacher, and A. Nowé, "Multi-objective reinforcement learning for the expected utility of the return," in *Proceedings of the Adaptive and Learning Agents workshop at FAIM*, vol. 2018, 2018.

[30] A. R. et al., "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[31] F. Felten and L. N. Alegre, "MORL-baselines: Multi-objective reinforcement learning algorithms implementations," https://github.com/LucasAlegre/morl-baselines, 2022.

[32] ETSI, "Osm Release ELEVEN release notes," https://osm-download.etsi.org/ftp/osm-11.0-eleven/OSM_Release_ELEVEN_Release_Notes.pdf, accessed: 2022-02-11.

[33] Sukchan Lee, "Open5gs." [Online]. Available: https://open5gs.org/

[34] Ali Güngör, "Ueransim." [Online]. Available: https://github.com/aligungr/UERANSIM

DOI: 10.1109/NFV-SDN59219.2023.10329594