Computing
https://doi.org/10.1007/s00607-020-00893-8

SPECIAL ISSUE ARTICLE

Attacks and failures prediction framework for a collaborative 5G mobile network

Yosra Benslimen[1] · Hichem Sedjelmaci[1] · Ana-Cristina Manenti[1]

Received: 5 June 2020 / Accepted: 17 December 2020

Abstract
Although mobile technologies keep evolving through years, Fault management and cyber-security management in mobile networks are still treated as separated notions with different blocks and different approaches whereas in practice, they are highly correlated. In this paper, we propose a framework that takes into account the correlation between these two management systems. The framework is based on several prediction agents where each agent is composed of a security predictor, a fault predictor and a generic anomaly detection model. A re-enforcement process allows to enhance the reliability of the machine learning training and prediction phases of the different predictors. Besides, each agent can collaborate with its neighborhood for a more resilient network. An application of this framework to 5G architecture is proposed by mapping the components of our framework with network slices. Finally, an experimentation is held over a testbed that we set up on openstack in order to forecast future anomalies related to proxy overload, latency violation in call session network functions and to excessive usage of memory. The training is achieved with ARIMA and deep learning models with promising results.

Keywords Cyber-security · Fault management · Anomaly detection · Mobile networks · Cognitive manegement · Machine learning · Re-enforcement learning

Mathematics Subject Classification 49N30 · 68Q32 · 68T05 · 68T07 · 97C30

✉ Yosra Benslimen
yosra.benslimen@gmail.com

Hichem Sedjelmaci
hichem.sedjelmaci@orange.com

Ana-Cristina Manenti
anacristina.manenti@orange.com

[1] Orange Labs, Chatillon, France

Published online: 23 January 2021

1 Introduction

As mobile network technologies evolve, new services are offered and more sophisticated networks are needed. The increasing number of Internet users leads to a redesign of network architecture, forcing designers to take into account new parameters such as the need of global coverage combined with low latency, as well as a high reliability and security level. Additionally, new networking experiences are added, such as Internet-of-Things (IoT), which promise to offer new services and facilities to people's daily lives. In this demanding environment, 5G technology is emerging, playing a decisive role in the implementation of new visions and promising to deliver solutions. A major innovation introduced by 5G technology [1] is the scalability. 5G architectures take into account the possible need of extending the capabilities of the network, both at the level of user traffic growth and at the level of new services input from providers. Slicing could be the ideal solution for such networks, offering scalability as well as flexibility in managing a giant network. Network Slicing is set to be a prominent feature of 5G to allow connectivity and data processing tailored to specific customers requirements. Mobile communications provided by smart networks will enhance the efficiency and productivity of business processes and will open up opportunities for network operators to address the Business-to-Business segment more effectively.

5G architecture faces a number of security risks and challenges due to network virtualization. In order to successfully fulfill its envisioned goals these issues must be resolved: both conceptually, by clearly defining the functionality and scope of security and privacy features of the architecture, and technically, by utilizing the most suitable solutions in the architecture design. Further, given the wide range of verticals to be involved (e.g., e-health, emergency services, smart grids), a strong isolation of the individual slices is crucial. It is expected that 5G infrastructure slices offered by the telecom operators will replace and augment critical infrastructures previously operated on dedicated resources. Thus, slices must provide a level of availability, performance, and security that is at least equal to the infrastructure that they are supplanting. Specifically, the architecture must guarantee that the slice control and data planes cannot be disrupted by external parties or co-hosted slice elements, and must detect and mitigate attacks which may expose slice data to unauthorized parties. The problem is aggravated by high degree of virtualization and automation that 5G infrastructures are expected to employ. A strong consistency between the various levels of abstraction used is therefore essential. To this end, the architecture requires effective mechanisms for monitoring and managing the infrastructure components - end-to-end - across multiple administrative domains.

Apart from security challenges, Network Function Virtualization (NFV) introduces meaningful challenges concerning fault management including network failures, anomalies on network equipment and functions, degradation of quality of services and also SLA violation. One of the main reasons is that the virtual network functions can be deployed in any place in the infrastructure with dynamic interconnections. Hence, the underlying dependencies of a network service may change several times over the service life cycle which makes the fault management more challenging and fault propagation behavior more complex.

Despite the evolution of mobile technologies and despite the resulted new challenges, cyber-security management and failures management are still defined by different building blocks in the architectures ignoring the fact that these events may be highly correlated. Firstly, in machine learning language, both are considered as an anomaly detection problems with a root cause analysis. Anomaly detection [2] is a family of techniques aiming to detect observations that deviate from the majority of the observed data. Usually, anomalies are related to critical events in real world. For example, a fraudulent credit card transaction can be defined as an anomaly because it means unauthorized charges from an account. A faulty behaviour of a network equipment is an anomaly because it produces deviation as the usual one. An internet intrusion is an anomaly because it means unauthorized access or anomalous network traffic. Second, an attack could be achieved by provoking failures in the network equipment. Third, in order to detect attacks and also failures, shared attributes could be used such as the energy consumption, exhausted energy, overload, etc. For all these reasons, we propose in this paper a cognitive framework that contains several Attacks and Failures Prediction Agents (AFPA). This framework aims to ensure the resiliency and the security of 5G networks with a common building block that allows the agents to communicate and to interact while taking advantage of the common features. Our framework is based on a re-enforcement process that allows the different agents to learn continuously and to collaborate with their neighboring agents in order to be more efficient. Our second contribution aims to map the AFPA framework to a 5G architecture.

The remaining of this paper is structured as follows: Sect. 2 reviews the main work found in the literature in regards of fault and security management in mobile networks. Section 3 details the proposed AFPA framework. Section 4 elaborates the mapping of our AFPA framework and 5G networks. Section 5 presents the testbed and the numerical experiments that aim to evaluate the performance of the our solution. Finally, Sect. 6 elaborates the conclusions and future works.

2 Related works

A couple of research works focus on addressing the issues of cyber security and network failure in wireless and mobile networks [3–7]. In [3], the authors focus to detect the attacks that aim to cause a network failure, while ensuring a high detection and false positive rates against cyber-attacks. They propose and develop a risk management mechanism to assist the cyber security expert in evaluating the security risk and network failure. The main features of risk management mechanism are the probabilities of failures and attacks. The authors in [4] also rely on a probabilistic approach in order to define a system that detects an adversary who is corrupting the communication between IoT devices and the access point by compromising the gateway. They use the uplink packet drop probability of the IoT devices to monitor the behavior of the gateway with which they are associated. The determine the detection rule using the generalized likelihood ratio test, where the attack probabilities are estimated using maximum likelihood estimation. In [5], the authors aim to detect the attacks that target the access points with a purpose to avoid the serious network failures. Their security

and network maintenance algorithm requires a low cost to prevent the occurrence of attacker while avoiding a failure of network access point. The algorithm relies on a multi-agents concept, while the monitoring agents collaborate between each other to ensuring a high level of security and efficient network maintenance. The authors in [6] aim to detect the blackhole attacks and to mitigate their impact on both data and control planes of the OLSR routing protocol where each node depends on itself to identify and isolate malicious nodes. In [7], the authors propose a network and security mechanism to avoid possible security breaches, network congestion or even complete network failure. The mechanism is based on a graph theory to detect forwarding rules that cause forwarding loop, flow violation (caused by attackers) in a distributed controller environment of Software Defined Network. In [8], the authors analysis the security and resilience issues by using the multi-criteria decision-making approach. They defined a set of static factor that are used to analyze the monitored infrastructure and systems, and hence determine their levels of security and resilience issues. According to the authors, the proposed decision-making approach is most adaptable for the real-time applications as compared to the approach based on machine learning algorithms since it requires a low time during the analysis process. In [9], the authors provide a list of security and resilience issues that could occur in the 5G architecture. Specifically the issues occurred in NFV/SDN implementation. In order to overcome these issues related to security and resilience, the authors propose for their future work a trust and reputation systems that aims to evaluate and to quantify the level of security and resilience issues related to each monitored NFV/SDN implementation.

Regarding the network cognitive management, this issue has been widely addressed in literature. In [10], a solution is proposed to forecast failures in 4G radio access networks and more precisely drop calls and accessibility problems by using functional data analysis. In [11], the authors compare different statistical machine learning approaches in or order to predict future failures in LTE networks.The authors in [12] propose a root cause analysis approach to detect and to investigate the noisy neighbor anomalies in virtual network functions.

The automation of network cognitive management has been recognized by several large European Union projects under the 5G PPP in recent years. For instance, in the Phase 1 of the 5G PPP program, the SelfNet project [13] proposes a Software Defined Networking (SDN)-/Network Function Virtualization (NFV)-based network management framework for advanced Self-Organizing Network (SON) capabilities in 5G infrastructures. Similarly, another 5G PPP Phase 1 project CogNet [14] also targets artificial intelligence/machine learning (AI/ML)) based network management solutions and provides a service portfolio including data gathering and fault management. In addition, there are several works in literature that target different facets of cognitive management in softwarized network environments. As a summary of the application of ML/AI techniques in SDN/NFV environments, authors in [15] review the challenges and opportunities of ML/AI in softwarized network environments, with a special focus on data-driven decision making for management and control of SDN/NFV-based infrastructures. To this end, the authors propose to enhance the functional primitives of monitoring, composition and control with ML modules. All these works, among many others, pave the path towards cognitive-aided management of networks, specially in the presence of SDN/NFV technologies.

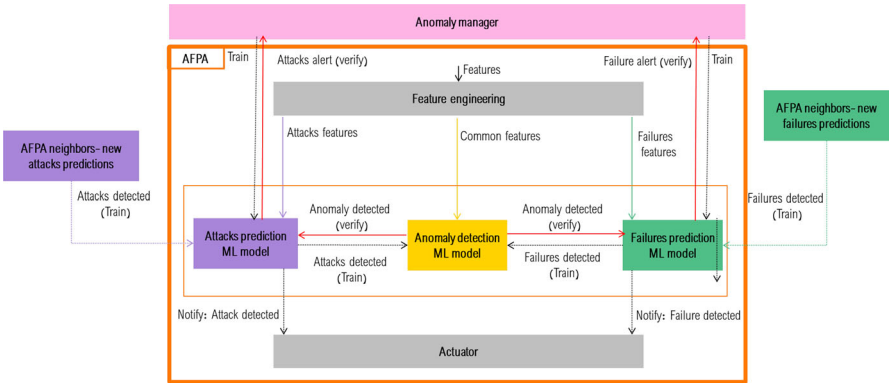


Fig. 1 AFPA framework

The major weakness of these works is the fact that they define solutions in order to predict network attacks and/or network failures separately. In other words, no correlation between network attacks and network failures is considered. The innovation of this paper is that it proposes an accurate cyber-attacks and network failures prediction framework that takes into account the correlation between these two notions. We propose a predictive framework for integrating security management with failures management for a more resilient network. We call it "Attacks and Failures Prediction Agent" (AFPA).

3 Attacks and failures prediction framework

The purpose of this framework is to propose an efficient AI predictive framework to predict the cyber-attacks and future network failures, while ensuring a high level of security and of resiliency in an optimal response time. More specifically, we propose Attacks and Failures Prediction Agent (AFPA) that aims to detect over time the most frequent network attacks and failures in an interactive way and with a re-enforcement process for predictive and preventive network management. Attacks and Failures Prediction Agent could be deployed at each system, host and function to monitor the attacks network and failures network. These agents will communicate between each other and collaborate together for a more efficient analysis. Each AFPA is equipped with three main modules as illustrated in Fig. 1.

3.1 Features engineering module

This first module focuses on monitoring the network behavior. It is responsible for extracting the most distinguishable features over time by analyzing various sources such as log files, probes counters, Intrusion Detection and Prediction (IDS/IPS), firewall, access control, etc…Later, these features are categorized into three families:

- Attacks features: they are the set of features that are exclusively useful for the determination of attacks such as DoS and Botnet, failed login error rate, server error rate, number of messages that are dropped, send and received, false detection rate generated by the Intrusion Detection System/ Intrusion Prevention System (IDS/IPS) and the number of false positive that firewall generates;
- Failures features: they are the set of features that are exclusively useful for the determination of failures such as drop calls percentage, congestion ratio, number of users in a cell, number of uplinks/downlink packets, signal to noise ratio, CPU usage, memory usage, network inbound, network outbound, reference Signal Receive Power, alarms, etc.
- Common features: they are the set of features that are commonly used for attacks predictions and also for failures predictions. Examples of these features are: Energy consumption, exhausted energy, overload, computation and communication overhead, interference, etc.

In order to classify the features into these three families, a first possibility relies on machine learning classification techniques that could be used to automatically feed and to classify the features' vectors. This solution may require a high computation and communication overhead to achieve its purpose. Therefore, a simpler possibility is to use a static approach that relies on cyber security and network experts who will feed over time the most relevant and attractive features in order to maximize the attacks and failures detection and to reduce the false positive rates. A dictionary could be used in this case in order to define the list of features as keys and their families as values. This solution allows to profit from the engineers expertise. Besides, the fact that network features definition does not change frequently makes the use of the dictionary option more advantageous since accessing and manipulating the data are easier.

3.2 Prediction module

This module mainly focuses on predicting the network failures and network attacks based on the outputs of the previous module. It is based on machine learning algorithms for anomaly prediction with a re-enforcement process. as illustrated in Fig. 1, we distinguish three main blocks.

A first block is attacks prediction machine learning (ML) model that is responsible for predicting attacks in the network by using the attacks features. In parallel, a second block is failures prediction ML model that is only dedicated at predicting failures in the network by using failures features. A third block is a general anomaly detection (AD) ML model that learns from the common features. The objective of this latter is to predict if an anomaly occured or will occur. Contrarily to the two previous ML models, this AD model has a weaker prediction mechanism. At the deployment stage, this model has no clue if the detected anomaly is related to failures or to attacks. The model will need support from the attacks prediction and the failures prediction models. For this reason, the anomaly together with its features vector will be shared with the attacks prediction ML model that will verify if the anomaly is an attack or not. The same case is applied in parallel to the failures prediction ML model that will verify if the anomaly corresponds to a failure. The two models will re-enforce the AD model

by sending to it the correct label and a reward that aims to penalize the AD model in case of a wrong prediction and to reward it in case of a correct one. Note that an anomaly can be at the same time related to attacks and also to failures.

Attacks prediction-ML algorithms modules are modeled as $\Psi_i^{Attacks} = \{\gamma_j, \delta, \theta\}$, where i is the number of anomaly managers that communicate directly with the anomaly manager. $\gamma_j = \{\gamma_1, , \gamma_m\}$, which corresponds to a features vector that attacks prediction-ML algorithm use to monitor its target and m is the number of features, that is varied over time since the cyber security and network experts feed over time the features vector with a new and relevant features. A supervised deep neuronal network can be used for the training and attacks detection process such as recurrent neural networks, convolutional neural networks, LSTM, or dense neural networks. We refer the reader to [16] for more details about these algorithms. The action $\delta = \{Normal, Attack1, Attack2, \ldots, AttackJ\}$ corresponds to the output attacks prediction- neuronal network algorithm, where J is the total number of attackers that are detected in the anomaly manager. In our solution, we focus to detect three kind of attacks, Denial of Service (DoS), Botnet and fuzzing threats. The value of payoff $\theta_t(\gamma_j, \delta)$ increases when the attacks prediction-ML algorithm correctly detects the attacks. Otherwise, the value of payoff θ_t decreases. In case of the attacks prediction-ML algorithm persists in providing false detection, it will be considered as an infected module (by the attacker) and hence the cyber security experts will change this module or feed it with a new training data set.

Failures prediction-ML algorithms modules are molded as $\Psi_i^{Failures} = \{\gamma\prime_{j\prime}, \delta\prime, \theta\prime\}$. Here, we can also use deep neuronal networks for failures detection process. $\gamma\prime_{j\prime} = \{\gamma\prime_1, \ldots, \gamma\prime_{m\prime}\}$ corresponds to a set of features that the ML algorithm monitors and uses as data entry for a training and failures detection process. $m\prime$ is the number of failures features, which is updated by network experts, while the experts focus to feed the ML algorithms with a new and relevant failures features. $\delta\prime = \{Normal, Failures1, Failures2, \ldots, FailuresJ\prime\}$ is the action that Failures prediction-ML algorithm provide as an output and $J\prime$ is the total number of failures that varies over time. Among the failures that we attempt to detect, we cite congestion problems in network cells, interference problems, drop calls, overload or noisy neighbors in virtual machines, degradation of a service, packet losses, problems in interfaces or routers…The payoff $\theta\prime(\gamma\prime_{j\prime}, \delta\prime)$ could increase or decrease, depending on the performance of the failures prediction-ML algorithm, i.e., when the ML algorithm continues to provide wrong detections (detect failures as normal or vice versa), it will be considered as a not robust ML model and hence the network experts should update the oldest failures features with a new and relevant features or/and feed the ML algorithm with new training data set. The utility function U_t of an Attacks and failures prediction agent i is computed as shown in Eq 1:

$$U_t = \frac{D_t - (P_t + N_t)}{AF_t} \tag{1}$$

D_t is the number of failures and attacks that are detected correctly by attacks and failures prediction-ML algorithms modules. P_t and N_t are respectively the number of

false positive and false negative provided by the ML algorithms modules. AF_t is the total number of attacks and failures that occured in the network.

At each iteration, the anomaly manager computes U^i_{t-1} and U^i_t of the monitored anomaly manager, compares the results and updates the payoffs θ_t and $\theta\prime_t$. The value of θ_t and $\theta\prime_t$ increases when $U\prime^i_t > U\prime^i_{t-1}$ and $U\prime\prime^i_t > U\prime\prime^i_{t-1}$, where $U\prime^i_t$ and $U\prime\prime^i_t$ are the utility functions of attacks and failures detection, respectively. At the end of each iteration, the anomaly manager requests the attacks and failures prediction-ML algorithms modules of the anomaly manager i to update their actions δ and $\delta\prime$ and the values of features γ_j and $\gamma\prime_{j\prime}$. These new actions and features are chosen as the one that generates a greater utility value between $U\prime^i_t$ and $U\prime^i_{t-1}$ for attacks prediction-ML algorithm and $U\prime\prime^i_t$ and $U\prime\prime^i_{t-1}$ for failures prediction-ML algorithm. The anomaly manager forwards the new values of features (γ_j and $\gamma\prime_{j\prime}$) to anomaly manager i to update its attacks or/and failures features in case actions (δ and $\delta\prime$) do not match the attacks and failures that are detected at anomaly manager level.

As shown in Eq 1, the anomaly manager updates the utility functions of anomaly manager i denoted as $U*_t$ recursively by estimating the optimal values of features $(\gamma_{j_{t+1}}, \gamma\prime_{j\prime_{t+1}})$, payoffs $(\theta_{t+1}, \theta\prime_{t+1})$ and actions $(\delta_{t+1}, \delta\prime_{t+1})$ [17].

$$\begin{aligned} U*_{t+1}(\gamma_{jt}, \gamma\prime_{j\prime_t}, \delta_t, \delta\prime_t) = U_t(\gamma_{jt}, \gamma\prime_{j\prime_t}, \delta_t, \delta\prime_t) + \alpha * [\theta_{t+1} + \theta\prime_{t+1} \\ +\gamma * maxU_t(\gamma_{jt+1}, \gamma\prime_{j\prime_{t+1}}, \delta, \delta\prime) - U_t(\gamma_{jt}, \gamma\prime_{j\prime_t}, \delta_t, \delta\prime_t)] \end{aligned} \tag{2}$$

Here, $\alpha \in]0, 1[$ is the learning rate and $\gamma \in]0, 1[$ is a constant which corresponds to a discount factor.

In order to make the training continuous in time and more efficient, the training data of the attacks prediction and the failures prediction ML models can be enriched by their neighbours. The correctly classified instances could be sent from an attack model to its neighbouring attack model by sending the set of features with their label i.e the absence of an attack or the type of the attack. The same applies for the failures models and their neighbors. The communication between the different models can be achieved by using application programming interfaces (APIs) based on Representational state transfer (REST) or a streaming tools such as kafka. This message will be injected in the training data of the model and used for the next planned training phase. The message will contain data relative to the ID of the host having a problem, the timestamp and the features.

3.3 Actuation module

The output of the previous attacks and failures ML models are to be used by the actuation module. This latter aims to react once an anomaly is detected. It aims to make remedial actions in order to moderate or to correct the impact of the anomaly. This module can be traditional rule-based techniques such as policies or it can be automated by using re-enforcement models or recommendation systems. Examples of actuation are as follows: if a failure is detected related to an overload in the virtual network functions, one actuation could be to migrate the function to another virtual machine or to re-scale it. If a congestion is predicted in the network, a load balancing solution could

be triggered. If an antenna is down, a remedial action could be to handover to another antenna. If a slice is down, a solution could be to migrate to another slice or to change the parameters of a slice. Regarding the reactions that can be triggered toward the detected cyber-attacks, we cite for instance attacks ejections,i.e., remove the infected systems (host and devices) from the network, inform other IDSs/IPSs and firewall about the identities of infected systems and finally update the keys cryptography to prevent the attackers to overhear the relevant data exchanged between IDSs/IPSs for instance.

4 AFPA for resilient 5G network slices

According to [18], a network slice (NS) consists of physical and/or virtual network functions (PNF/VNF) that can belong to access and core network part. Then, this network functions are interconnected by means of network resources, composing a synthetic infrastructure with specific characteristic, both in functionalities and resource capacities. The synthesis of a NS, then, serves a particular functional purpose and once instantiated, it is used to support certain communication services, which ultimately are deployed to support vertical services on top. Each network slice can have its own architecture, provisioning management and security that supports a particular use case. Each slice may be decomposed of one or several sub-slices that may belong to one or different network service providers. Each sub-slice contains physical and virtualized infrastructure.

In order to trigger a fault in the network, the slices and also the sub-slices should be investigated. However, in literature, most approaches aim to detect or to forecast attacks and faulty services at a sub-slice level [19–21]. Although this approach will gain in terms of precision due to its fine-granularity, its drawback is that they suppose that the sub-slices are independent which is not the case. In contrast, other approaches [22] aim to detect attacks and faulty behaviours in the end-to-end slice. This solution has the advantage of giving a general view of the global network slice behaviours. However, it lacks in terms of precision. Besides, sub-slices may be operated by different service providers which may limit the visibility and the tracking of faults and attacks.

In this section, we propose to apply the AFPA framework to 5G slices and sub-slices in order to ensure the resiliency and the security of 5G networks with a common building block that allows them to communicate and to interact and that takes advantage of the common features. The re-enforcement process allows the different agents to learn continuously and to be more efficient. The application of AFPA framework to 5G network is described in Fig. 2.

An anomaly manager is proposed in each sub-slice that allows to continuously control the failures and also the attacks that may occur in the sub-slice. Data are continuously collected from the infrastructure and the virtual layers. The data can be collected from different sources: KPIs, alarms, logs, etc. A feature engineering module (A and B in Fig. 2) allows the collection of data, the monitoring and the categorization of these data into Attacks features, failures features and common features with the type of the anomaly (i.e type of the attack or type of failures).

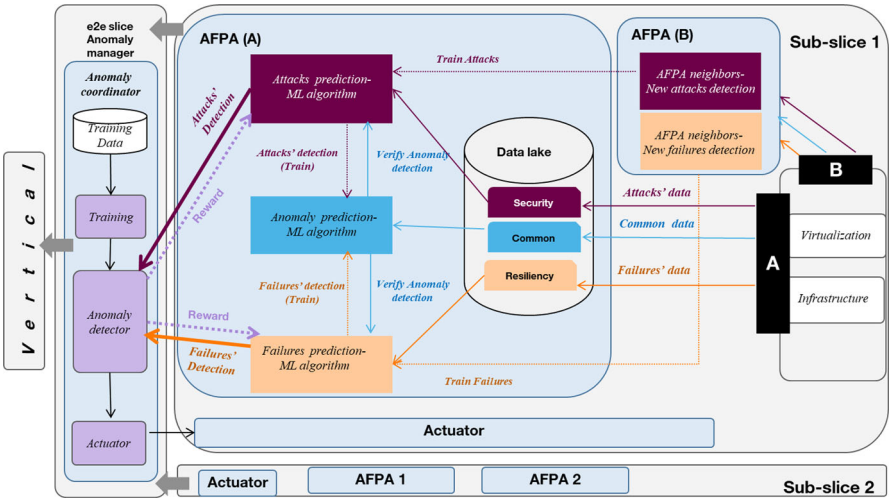


Fig. 2 AFPA for 5G networks

4.1 Deployment and initial training phase

The deployment phase consists of the first instantiating of the AFPAs in a 5G architecture. The training could be achieved offline by collecting data and then injecting the trained model in the framework. It can also be applied online by following these steps:

- The collected data are saved in a data lake. A first partition of this data lake is booked for data related to security attacks. A second partition is booked for common attacks. A third partition is booked for failures data.
- Within the sub-slices, an online training phase allows to learn the different patterns in order to detect or to forecast future failures/threats. Three ML models are to be trained within each AFPA agent, and the trained model are saved in the anomaly manager. The first model is dedicated to attacks prediction by using attacks data base. The second model is dedicated to failures prediction by only using failures data base. The third model is dedicated to the anomaly detection by using common features stored in the common data base.
- The previous step is achieved per sub-slice in order to learn the sub-slice behaviour with a fined view. In parallel, at the e2e slice-level, an anomaly manager is responsible to learn the general pattern of the e2e slice. This anomaly manager has a visibility on all its sub-slices. It has a prediction mechanism that will collect data related to all the sub-slices. A processing step allows to filter security from fault management data. In this stage, the training should be achieved with a strong machine learning model such as deep learning. The general anomaly detectors (GAD) will be stored to be used in the run-time phase.

4.2 Run-time phase

The run-time phase aims at using the trained model for exploitation. Note that even during run-time, the ML models will still continuously train in an interactive way.

In a first step, the data are continuously collected from the infrastructure and also from the virtualized level within a sub-slice by the corresponding features engineering module. In a second step, the collected data are saved in the data lake. Attacks features are saved in the attacks data base. Failure features are stored in failures data base. Common features are stored in common data base. In a third step, each instance will be analyzed by the corresponding sub-slice AFPA in order to predict future failures and attacks. The common anomaly detection ML model will predict if the received instance corresponds to a normal behaviour or to an anomaly. If an anomaly is detected, it will predict if it is an attack and/or a network failures and to which type of attacks or the failures it belongs. Once a prediction is achieved, this model will send the data containing the common features together with its prediction to the attacks ML model and also to the failures ML model. The features could be directly transferred by using a streaming mode for a rapid response time. For instance, the features can be transferred with a JSON format using a Kafka bus. The attacks and failures ML models will verify the prediction of the anomaly detection model. If the prediction is wrong, they will re-enforce the learning process of the anomaly detection model by computing the penalty and sending new training data in order to enhance its performance.

In a parallel way, the attacks and the failures models can be fed by their neighbors. A neighboring attacks predictor in the same sub-slice could aliment the current attack predictor by sending to it events (instances that are verified to be attacks) in order to enrich its training data and to keep it informed of the neighboring problems. The same applies for the failures model.

The attacks ML model and the failures ML models are also re-enforced by the E2E slice. In fact, each event (positive predictions) will be sent to this latter in order to alert it and also to ask for verification. The role of the agent in the E2E slice level is to ensure that the sub-slice AFPA agents continue to be efficient. Hence, By using its trustworthy trained model, the agent at the E2E level will predict the instance and re-enforce the corresponding model by penalizing it and sending to it more reliable training data. The objective of this penalty system is to make the sub-slice AFPA agents more stable and more reliable. The reliability of the models can be measured by using these performance metrics: False positive rate, False negative rate, Precision, Recall, F1-score, Error rate such as mean squared error, absolute squared error, etc. Since the number of transmitted messages could be huge, these messages could be aggregated over a time interval and also repeated messages could be aggregated.

Once a problem is confirmed by the E2E slice agent, a remedial action should be triggered. Hence, the prediction model will send an alert to the actuation mechanism (such as the orchestrator or policy manager [23]) containing the ID of the host in question, the timestamp, the predicted failure and/or attack. Depending to the received notification, the actuator will correct the problem at the e2e slice level and it will ask for corrective actions to be applied by the actuator at sub-slice levels.

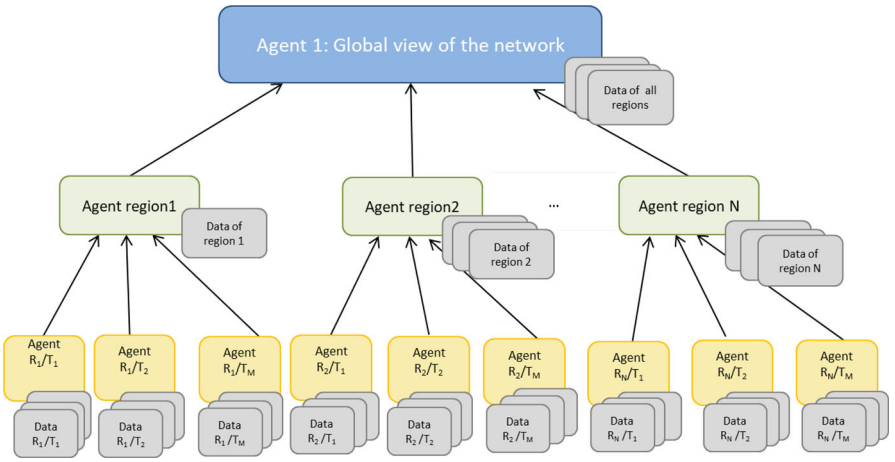


Fig. 3 AFPA for 4G/3G/2G networks

4.3 AFPA for a multi-technology network

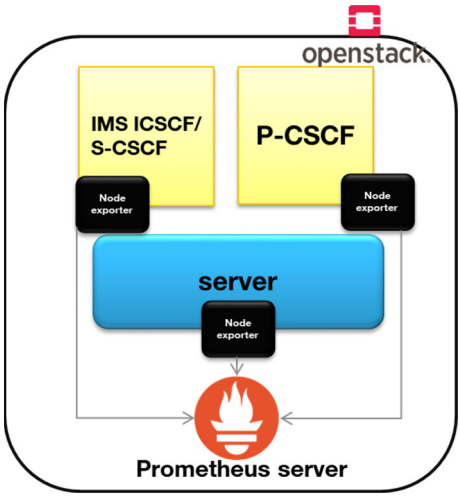
The approach that we propose is still usable for other mobile technologies, other than 5G. In this case, it will be based on a hierarchical structure that is composed of two or several levels. The levels could be defined for instance, by regions, types of equipment, geographical proximity, network functions or by type of services etc. As depicted in Fig. 3, the most bottom-level has the more restricted view. Each level has an AFPA that is re-enforcing the AFPA of the level below. At the same level, the common AD model is re-enforced by the attacks and the failures detectors. These latter are communicating with their neighbors and are re-enforced. Once a problem is detected at the top level, an actuation should be triggered. Figure 3 illustrates an example of applying the AFPA framework in a multi-technology system.

The highest level has a global vision of the network. In this level, The learning is achieved on data that covers all the technologies of all the regions for a large period. The middle level has a more restricted view. The learning in this level is achieved by region in a distributed way. The learning is achieved by several AFPA agents. Each agent covers data of a specific region. The bottom level has the most restricted view. This level focuses on learning from each technology for each region in a distributed way. Each agent receives data of a specific technique for a specific region. The agents aim to detect if yes or no there is an attack and/or a failure given the observation of a specific technology for a specific region. In case of problem, the agent notifies the higher level.

5 Numerical experimentation

In order to evaluate the proposed AFPA agent, we created a testbed that is set-up in an *Openstack* environment as illustrated in Fig. 4. The tesbed contains two virtual

Fig. 4 Testbed in openstack



machines running over the same hypervisor. The two machines are sharing the same memory.

The first machine implements The IMS I-CSCF (Interrogating-Call Session Control Function) and S-CSCF (Serving-Call Session Control Function) functionality. The second machine simulates a Proxy, it uses Sprout (SIP Router) and it implements the P-CSCF functions (Proxy-Call/Session Control Functions). It is the entry point of SIP clients, which is in turn routed SIP requests to the first machine. A Prometheus Server is used to create a training set by collecting several metrics describing the behaviour of each machine.

Our objective is to apply the AFPA agent by defining three machine learning models: (1) An attack prediction model that aims to train from the proxy machine and to forecast future threats causing overload in the proxy by using a uni-variate feature vectors and an ARIMA approach for time series; (2) A failures prediction model aims to train from the first machine and to forecast future latency problems related to call sessions by using multi-variate features and a deep learning model; (3) A general anomaly detection models that aims to train from the shared memory usage in order to forecast future latency in accessing the memory by also using multi-variate features and a deep learning model.

Regarding the data collection, we use *Node Exporter* as an agent in every Virtual machine and physical machine. It allows collecting information from the machine in the form of counters that are sent to *Prometheus Server*. This server creates metrics that describe the utilization of the component in order to create a dataset. 21 metrics are collected and they belong to the following families:

- CPU usage: is a percentage of the time during it the CPU was busy.
- CPU wait percentage: Percentage of time the CPU is idle AND there is at least one I/O request in progress
- Network inbound: is the amount of data arriving to the machine but originating elsewhere and

– Network outbound: is the amount of data originating at the machine to arrive elsewhere. Although memory usage could also be used, in our case, the memory is not shared so the machines do not compete over it.
– Memory usage: The percentage of the used memory
– Load: The normalized (by number of logical cores) average system load over a 1-min period

Hence, three data sets are collected (one per machine). Each dataset is composed of $N = 177000$ instances. Let x be the dataset collected from a machine M and composed of N instances x_i that are collected periodically and $F = 21$ features so that $x_i = \{x_{i1}, .., x_{ij}\}$ where $1 \leq i \leq N$ and $1 \leq j \leq F$.

We start by categorizing the collected features into three families. Since the two machines are sharing the same memory, the attributes related to the memory usage correspond to the common attributes. These latter will be used by the generic anomaly detection model. The remaining of the attributes (CPU utilization, load, network inbound and outbound) of the proxy machine correspond to security features that will be used by the security predictor. The same metrics related to call session machine correspond to failures features.

5.1 ARIMA model for attacks prediction

The first model aims to predict the threats with an overload in the future 10 min by training from a uni-variate feature: the average load in the last 1 min. This latter represents a stationary time series, since the augmented Dickey-Fuller test (ADF) results with a p value equal to 0. 000208 (and hence smaller than 0.05). In order to forecast future values of the average load, a smoothing is first applied and then an Auto-ARIMA model is trained. One advantage of this approach is that the best fit of the hyper-parameters are set automatically by applying a selction model criterio to be minimized: the AIC (Akaike Information Criterion).

In order to evaluate the Auto-ARIMA model, we divided our data set into 80% for the training phase and 20% for the test set. The test on the test set has a small value of the mean absolute percentage error (MAPE) that is equal to 6% and a high correlation factor between the actual and the forecast values that is equal to 82.5 %. The forecasting of the future overload in the proxy is illustrated in Fig. 5.

5.2 Failures prediction ML model

This second model aims to predict future failures in IMS ICSCF/S-CSCF machine. More specifically, the model train from data having 170,000 records and 21 attributes belonging to the following families: CPU usage, network inbound, network outbound and load. The aim is to forecast the future violation of latency in the future 15 min. The data are labeled with a binary classes representing the presence or absence of latency violation. The data set is imbalanced: 10.034 records represent latency violation whereas 166.966 records represent a normal status. The metrics are sampled every 1 s.

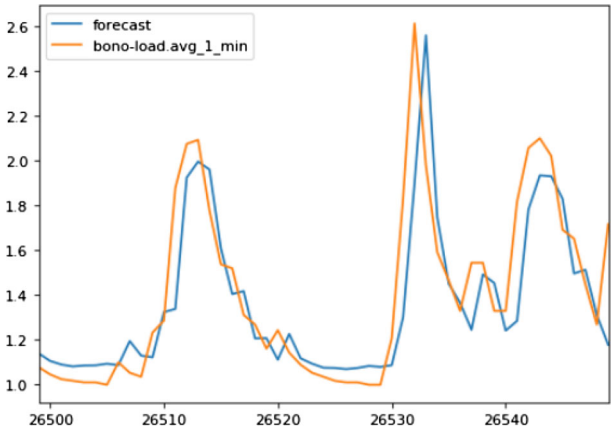


Fig. 5 Security prediction—AutoArima forecasting on test set

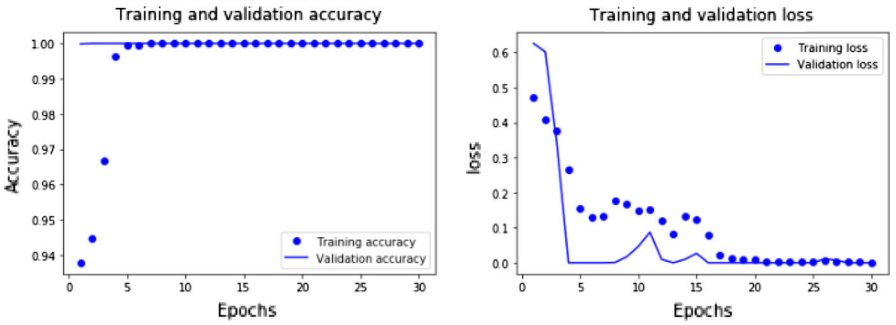


Fig. 6 Loss and accuracy of the failures prediction ML model over training and validation sets

A deep learning model is created in order to observe the last 2 h of the machine metrics in order to predict the future 15 min of latency violation.

80% of the data set are used for training and validation while the remaining 20% are kept for testing. A long short term model (LSTM) is trained with 30 epochs. The loss is computed using a cross entropy loss function. The results of the loss and accuracy over the training and the validation sets are illustrated by Fig. 6. After 16 iterations, the model starts the have a high accuracy and a low error with no over-fitting as proved by the two graphs. The evaluation over the test set also proves that the accuracy of the model is promising (100%) and the loss is still low (0.0568%) which proves the performance of the model.

5.3 Anomaly prediction ML model

For the third model, a new set of data was created from the metrics belonging to the memory family of the two virtual machines. As a result we have 177,000 records and six attributes. The objective is to observe the last 2 h in order to forecast excessive usage of the shared memory by using the common attributes that will occur in the

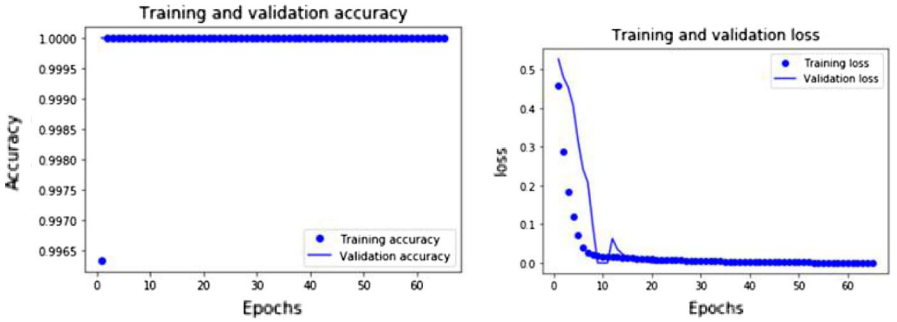


Fig. 7 Loss and accuracy of the anomaly detection model over training and validation sets

future 15 min. The class is binary and it corresponds to the presence or the absence of an excessive usage of the problem.

Similarly than the previous model, an LSTM model is trained with 65 epochs on 80% of the data set where 10% is kept for validation. The remaining 20% are used for testing. The results over the training and the validation sets are illustrated by Fig. 7. After the first 15 epochs, the model starts to converging with a promising validation and a training performance that showcase the absence of over-fitting problem. The accuracy on the test set is equal to 100% and the loss is equal to 0.0006.

6 Conclusion

This paper proposes a cognitive architecture containing one or several Attacks and Failures Prediction Agent (AFPA) that could be deployed at each system, host and function to monitor network attacks and failures in a common block. The proposed agent takes into account the correlation between security management system and failures management in order to develop a reliable solution that aims to learn at sub-slice level and at the level of the end-to-end slice which allows to have a fine view and also a general view of the network interactively.

As future work, several perspectives could be tackled. First, a more extensive experimental study of several AFPA agents will be proposed in order to showcase the collaboration and the re-enforcement mechanism. Second, a testbed of the overall architecture applied to 5G network should be developed in order to demonstrate the applicability of this framework in a 5G technology. Third, a mutual AFPA monitoring for accurate attacks and failure prediction should be proposed. It consists of a new reputation protocol that evaluates the behavior of system, host and function according to a couple (Trustl_Level, Failure_Level). Finally, the top level ML model in our AFPA framework could be infected by attackers and could generate a fault due to a failure. Therefore, it is mandatory to evaluate the trust and failure level of a monitored AFPA, where our reputation protocol is based on a Security and Maintenance reputation phases.

References

1. NGMN (2015) 5G white Paper. Technical report
2. Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv (CSUR) 41(3):15
3. Kbar G (2009) Security risk analysis based on probability of system failure, attacks and vulnerabilities, pp 874–879
4. Abhishek NV, Tandon A, Lim TJ, Sikdar B (2018) Detecting forwarding misbehavior in clustered iot networks. In: Proceedings of the 14th ACM international symposium on qos and security for wireless and mobile networks, series (Q2SWinet'18). Association for Computing Machinery, New York, NY, USA, p 16
5. Sriram VSS, Sahoo G, Agrawal KK (2010) Detecting and eliminating rogue access points in ieee-802.11 WLAN—a multi-agent sourcing methodology. In: 2010 IEEE 2nd international advance computing conference (IACC), pp 256–260
6. Singh PK, Kar K (2019) Countering data and control plane attack on olsr using passive neighbor policing and inconsistency identification. In: Proceedings of the 15th ACM international symposium on QoS and security for wireless and mobile networks, series (Q2SWinet'19). Association for Computing Machinery, New York, NY, USA, p 1928
7. Halder B, Barik MS, Mazumdar C (2018) Detection of flow violation in distributed SDN controller. In: 2018 Fifth international conference on emerging applications of information technology (EAIT), pp 1–6
8. Kralik L, Malanik D, Matysek M (2018) Cyber security resilience based on static factors as a part of converged security. In: 2018 5th international conference on mathematics and computers in sciences and industry (MCSI), pp 114–117
9. Arfaoui G, Vilchez JM Sanchez, Wary J (2017) Security and resilience in 5g: current challenges and future directions. In: 2017 IEEE Trustcom/BigDataSE/ICESS, pp 1010–1015
10. Slimen YB, Allio S, Jacques J (2017) Anomaly prevision in radio access networks using functional data analysis. In: 2017 IEEE global communications conference (GLOBECOM 2017), Singapore, December 4–8, 2017. IEEE, pp. 1–6
11. Hadj-Kacem I, Jemaa SB, Allio S, Slimen YB (2020) Anomaly prediction in mobile networks: a data driven approach for machine learning algorithm selection. In: NOMS 2020 - IEEE/IFIP network operations and management symposium, Budapest, Hungary, April 20–24, 2020. IEEE, pp 1–7
12. Bouattour H, Slimen YB, Mechteri M, Biallach H (2020) Root cause analysis of noisy neighbors in a virtualized infrastructure. In: 2020 IEEE wireless communications and networking conference (WCNC 2020), Seoul, Korea (South), May 25–28, 2020. IEEE, pp 1–6
13. SelfNet: a framework for self-organized network management in virtualized and software defined networks [Online]. https://selfnet-5g.eu/
14. CogNet building an intelligent system of insights and action for 5G network management [online]. https://5g-ppp.eu/cognet/
15. Kellerer W, Kalmbach P, Blenk A, Basta A, Reisslein M, Schmid S (2019) Adaptable and data-driven softwarized networks: review, opportunities, and challenges. Proc IEEE 107(4):711–731
16. Chollet F (2017) Deep learning with python. Manning
17. Tomasik B (2017) Ethical issues in artificial reinforcement learning. Technical report
18. 3GPPP (2017) 28.801: telecommunication management;Study on management and orchestration of network slicing for next generation network. Technical report
19. Jiang W, Strufe M, Schotten HD (2017) Intelligent network management for 5g systems: the SELFNET approach. In: 2017 European conference on networks and communications (EuCNC), pp 1–5
20. Kaloxylos A (2018) "Application of data mining in the 5g network architecture,"
21. Kibria MG, Nguyen K, Villardi GP, Zhao O, Ishizu K, Kojima F (2018) Big data analytics, machine learning, and artificial intelligence in next-generation wireless networks. IEEE Access 6:32 328–32 338
22. Kotulski Z, Nowak T, Sepczuk M, Tunia M, Artych R, Bocianiak K, Osko T, Wary J (2017) On end-to-end approach for slice isolation in 5g networks. fundamental challenges. In: 2017 federated conference on computer science and information systems (FedCSIS), pp 783–792
23. IETF (2017) SUPA policy-based management framework. Technical report